Network Working Group
Request for Comments: 4862
Obsoletes: 2462
Category: Standards Track

S. Thomson
Cisco
T. Narten
IBM
T. Jinmei
Toshiba

September 2007

IPv6 Stateless Address Autoconfiguration

Status of This Memo

Abstract

This document specifies the steps a host takes in deciding how to autoconfigure its interfaces in IP version 6. The autoconfiguration process includes generating a link-local address, generating global addresses via stateless address autoconfiguration, and the Duplicate Address Detection procedure to verify the uniqueness of the addresses on a link.

Table of Contents

1. Introduction

This document specifies the steps a host takes in deciding how to autoconfigure its interfaces in IP version 6 (IPv6). The autoconfiguration process includes generating a link-local address, generating global addresses via stateless address autoconfiguration, and the Duplicate Address Detection procedure to verify the uniqueness of the addresses on a link.

The IPv6 stateless autoconfiguration mechanism 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.

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. On the other hand, Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC3315] is used when a site requires tighter control over exact address assignments. Both stateless address autoconfiguration and DHCPv6 may be used simultaneously.

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 an address is in a deprecated state, its use 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 ensure 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, independently of whether they are obtained via stateless autoconfiguration or DHCPv6. 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 IPv6 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. The protocol described in this document will be used on all types of links unless specified otherwise in the link-type-specific document describing how to operate IP on the link in line with [RFC4861].

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 [RFC4291].

solicited-node multicast address - a multicast address to which Neighbor Solicitation messages are sent. The algorithm for computing the address is given in [RFC4291].

link-layer address - a link-layer identifier for an interface. Examples include IEEE 802 addresses for Ethernet links and E.164 addresses for Integrated Services Digital Network (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.

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 latter 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 [RFC4291]. 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., [RFC2464]). Note that the address architecture [RFC4291] also defines the length of the interface identifiers for some set of addresses, but the two sets of definitions must be consistent. In many cases, the identifier will be derived from 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 [RFC2119].

Note that this document intentionally limits the use of the keywords to the protocol specification (Section 5).

3. Design Goals

Stateless autoconfiguration is designed with the following goals in mind:

- 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.

- Small sites consisting of a set of machines attached to a single link should not require the presence of a DHCPv6 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 an interface identifier to the link-local prefix.

- A large site with multiple networks and routers should not require the presence of a DHCPv6 server for address configuration. In order to generate 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.

- 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.

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 an identifier of the interface to the well-known link-local prefix [RFC4291].

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 the address 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 can do. Note that the DHCPv6 service for address configuration may still be available even if no routers are present.

Routers send Router Advertisements periodically, but the delay between successive advertisements will generally be longer than a host performing autoconfiguration will want to wait [RFC4861]. To obtain an advertisement quickly, a host sends one or more Router Solicitations to the all-routers multicast group.

Router Advertisements also contain zero or more Prefix Information options that contain information used by stateless address autoconfiguration to generate global addresses. It should be noted that a host may use both stateless address autoconfiguration and DHCPv6 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.

By default, all addresses should be tested for uniqueness prior to their assignment to an interface for safety. The test should individually be performed on all addresses obtained manually, via stateless address autoconfiguration, or via DHCPv6. 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 and are described in [RFC3484].

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-capable 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., [RFC2464]).

Autoconfiguration also assumes the presence of the variable RetransTimer as defined in [RFC4861]. 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 Structures

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

A host 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. This includes the case where the attached link is dynamically changed due to a change of the access point of wireless networks.

- The interface becomes enabled by system management after having been administratively disabled.

A link-local address is formed by combining the well-known link-local prefix FE80::0 [RFC4291] (of appropriate length) with an interface identifier as follows:

1. The left-most 'prefix length' bits of the address are those of the link-local prefix.

2. The bits in the address to the right of the link-local prefix are set to all zeroes.

3. If the length of the interface identifier is N bits, the right-most N bits of the address are replaced by the interface identifier.

If the sum of the link-local prefix length and N is larger than 128, autoconfiguration fails and manual configuration is required. The length of the interface identifier is defined in a separate link-type-specific document, which should also be consistent with the address architecture [RFC4291] (see Section 2). These documents will carefully define the length so that link-local addresses can be autoconfigured on the link.

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

5.4. Duplicate Address Detection

Duplicate Address Detection MUST be performed on all unicast addresses prior to assigning them to an interface, regardless of whether they are obtained through stateless autoconfiguration, DHCPv6, or manual configuration, with the following exceptions:

- An interface whose DupAddrDetectTransmits variable is set to zero does not perform Duplicate Address Detection.

- Duplicate Address Detection MUST NOT be performed on anycast addresses (note that anycast addresses cannot syntactically be distinguished from unicast addresses).

- Each individual unicast address SHOULD be tested for uniqueness. Note that there are implementations deployed that only perform Duplicate Address Detection for the link-local address and skip the test for the global address that uses the same interface identifier as that of the link-local address. Whereas this document does not invalidate such implementations, this kind of

"optimization" is NOT RECOMMENDED, and new implementations MUST NOT do that optimization. This optimization came from the assumption that all of an interface's addresses are generated from the same identifier. However, the assumption does actually not stand; new types of addresses have been introduced where the interface identifiers are not necessarily the same for all unicast addresses on a single interface [RFC4941] [RFC3972]. Requiring that Duplicate Address Detection be performed for all unicast addresses will make the algorithm robust for the current and future special interface identifiers.

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. Note that the "other packets" include Neighbor Solicitation and Advertisement messages that have the tentative (i.e., unicast) address as the IP destination address and contain the tentative address in the Target Address field. Such a case should not happen in normal operation, though, since these messages are multicasted in the Duplicate Address Detection procedure.

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 [RFC4861]. A Neighbor Solicitation or Advertisement message that passes these validity checks is called a valid solicitation or valid advertisement, respectively.

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 ensures that the node receives Neighbor Advertisements from other nodes already using the address; the latter ensures that two nodes attempting to use the same address simultaneously should 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 going to be the first message sent from an interface after interface (re)initialization, the node SHOULD delay joining the solicited-node multicast address by a random delay between 0 and MAX_RTR_SOLICITATION_DELAY as specified in [RFC4861]. 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.

Even if the Neighbor Solicitation is not going to be the first message sent, the node SHOULD delay joining the solicited-node multicast address by a random delay between 0 and MAX_RTR_SOLICITATION_DELAY if the address being checked is configured by a router advertisement message sent to a multicast address. The delay will avoid similar congestion when multiple nodes are going to configure addresses by receiving the same single multicast router advertisement.

Note that when a node joins a multicast address, it typically sends a Multicast Listener Discovery (MLD) report message [RFC2710] [RFC3810] for the multicast address. In the case of Duplicate Address Detection, the MLD report message is required in order to inform MLD-snooping switches, rather than routers, to forward multicast packets. In the above description, the delay for joining the multicast address thus means delaying transmission of the corresponding MLD report message. Since the MLD specifications do not request a random delay to avoid race conditions, just delaying Neighbor Solicitation would cause congestion by the MLD report messages. The congestion would then prevent the MLD-snooping switches from working correctly and, as a result, prevent Duplicate Address Detection from working. The requirement to include the delay for the MLD report in this case avoids this scenario. [RFC3590] also talks about some interaction issues between Duplicate Address Detection and MLD, and specifies which source address should be used for the MLD report in this case.

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 during the delay period. This does not necessarily conflict with the requirement that joining the multicast group be delayed. In fact, in some cases it is possible for a node to start listening to the group during the delay period before MLD report transmission. It should be noted, however, that in some link-layer environments, particularly with MLD-snooping switches, no multicast reception will be available until the MLD report is sent.

5.4.3. Receiving Neighbor Solicitation Messages

On receipt of a valid Neighbor Solicitation message on an interface, node behavior depends on whether or not the target address is tentative. If the target address is not tentative (i.e., it is assigned to the receiving interface), the solicitation is processed as described in [RFC4861]. 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.

Implementer'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 Appendix A 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 before one is sent, 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 joining the solicited-node multicast address and 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 loop back the 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:

1. If the target address is tentative, the tentative address is not unique.

2. If the target address matches a unicast address assigned to the receiving interface, it would possibly indicate that the address is a duplicate but it has not been detected by the Duplicate Address Detection procedure (recall that Duplicate Address Detection is not completely reliable). How to handle such a case is beyond the scope of this document.

3. Otherwise, the advertisement is processed as described in [RFC4861].

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 based on the hardware address, which is supposed to be uniquely assigned (e.g., EUI-64 for an Ethernet interface), IP operation on the interface SHOULD be disabled. By disabling IP operation, the node will then:

- not send any IP packets from the interface,

- silently drop any IP packets received on the interface, and

- not forward any IP packets to the interface (when acting as a router or processing a packet with a Routing header).

In this case, the IP address duplication probably means duplicate hardware addresses are in use, and trying to recover from it by configuring another IP address will not result in a usable network. In fact, it probably makes things worse by creating problems that are harder to diagnose than just disabling network operation on the interface; the user will see a partially working network where some things work, and other things do not.

On the other hand, if the duplicate link-local address is not formed from an interface identifier based on the hardware address, which is supposed to be uniquely assigned, IP operation on the interface MAY be continued.

Note: as specified in Section 2, "IP" means "IPv6" in the above description. While the background rationale about hardware address is independent of particular network protocols, its effect on other protocols is beyond the scope of this document.

5.5. Creation of Global Addresses

Global 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 addresses 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 [RFC4861].

5.5.2. Absence of Router Advertisements

Even if a link has no routers, the DHCPv6 service to obtain addresses may still be available, and hosts may want to use the service. 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 [RFC4861].

Note that it is possible that there is no router on the link in this sense, but there is a node that has the ability to forward packets. In this case, the forwarding node's address must be manually configured in hosts to be able to send packets off-link, since the only mechanism to configure the default router's address automatically is the one using Router Advertisements.

5.5.3. Router Advertisement Processing

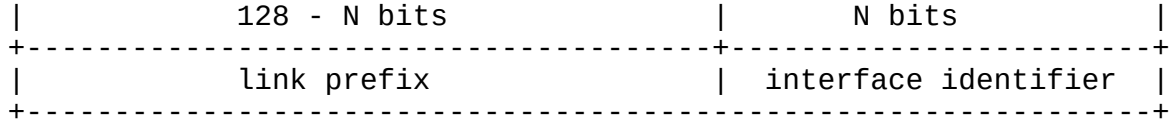
For each Prefix-Information option in the Router Advertisement:

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 is not equal to the prefix of an address configured by stateless autoconfiguration already in the list of addresses associated with the interface (where "equal" means the two prefix lengths are the same and the first prefix-length bits of the prefixes are identical), and if the Valid Lifetime is not 0, form an address (and add it to the list) by combining the advertised prefix with an interface identifier of the link as follows:



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. The length of the interface identifier is defined in a separate link-type specific document, which should also be consistent with the address architecture [RFC4291] (see Section 2).

It is the responsibility of the system administrator to ensure that the lengths of prefixes contained in Router Advertisements are consistent with the length of interface identifiers for that link type. It should be noted, however, that this does not mean the advertised prefix length is meaningless. In fact, the advertised length has non-trivial meaning for on-link determination in [RFC4861] where the sum of the prefix length and the interface identifier length may not be equal to 128. Thus, it should be safe to validate the advertised prefix length here, in order to detect and avoid a configuration error specifying an invalid prefix length in the context of address autoconfiguration.

Note that a future revision of the address architecture [RFC4291] and a future link-type-specific document, which will still be consistent with each other, could potentially allow for an interface identifier of length other than the value defined in the current documents. Thus, an implementation should not assume a particular constant. Rather, it should expect any lengths of interface identifiers.

If an address is formed successfully and the address is not yet in the list, 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. Note that the check against the prefix performed at the beginning of this step cannot always detect the address conflict in the list. It could be possible that an address already in the list, configured either manually or by DHCPv6, happens to be identical to the newly created address, whereas such a case should be atypical.

e) If the advertised prefix is equal to the prefix of an address configured by stateless autoconfiguration in the list, the preferred lifetime of the address is reset to the Preferred Lifetime in the received advertisement. The specific action to perform for the valid lifetime of the address depends on the Valid Lifetime in the received advertisement and the remaining time to the valid lifetime expiration of the previously autoconfigured address. We call the remaining time "RemainingLifetime" in the following discussion:

1. If the received Valid Lifetime is greater than 2 hours or greater than RemainingLifetime, set the valid lifetime of the corresponding address to the advertised Valid Lifetime.

2. If RemainingLifetime is less than or equal to 2 hours, ignore the Prefix Information option with regards to the valid lifetime, unless the Router Advertisement from which this option was obtained has been authenticated (e.g., via Secure Neighbor Discovery [RFC3971]). If the Router Advertisement was authenticated, the valid lifetime of the corresponding address should be set to the Valid Lifetime in the received option.

3. Otherwise, reset the valid lifetime of the corresponding address to 2 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 Valid Lifetimes could cause all of a node's addresses to expire prematurely. The above rules ensure that legitimate advertisements (which are sent periodically) will "cancel" the short Valid Lifetimes before they actually take effect.

Note that the preferred lifetime of the corresponding address is always reset to the Preferred Lifetime in the received Prefix Information option, regardless of whether the valid lifetime is also reset or ignored. The difference comes from the fact that the possible attack for the preferred lifetime is relatively minor. Additionally, it is even undesirable to ignore the preferred lifetime when a valid administrator wants to deprecate a particular address by sending a short preferred lifetime (and the valid lifetime is ignored by accident).

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 to initiate new communications if an alternate (non-deprecated) address of sufficient scope can easily be used instead.

Note that the feasibility of initiating new communication using a non-deprecated address may be an application-specific decision, as only the application may have knowledge about whether the (now) deprecated address was (or still is) in use by the application. For

example, if an application explicitly specifies that the protocol stack use a deprecated address as a source address, the protocol stack must accept that; the application might request it because that IP address is used in higher-level communication and there might be a requirement that the multiple connections in such a grouping use the same pair of IP addresses.

IP and higher layers (e.g., TCP, UDP) MUST continue to accept and process datagrams destined to a deprecated address as normal since a deprecated address is still a valid address for the interface. In the case of TCP, this means TCP SYN segments sent to a deprecated address are responded to using the deprecated address as a source address in the corresponding SYN-ACK (if the connection would otherwise be allowed).

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.

Other subtle cases should also be noted about source address selection. For example, the above description does not clarify which address should be used between a deprecated, smaller-scope address and a non-deprecated, sufficient scope address. The details of the address selection including this case are described in [RFC3484] and are beyond the scope of this document.

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 autoconfiguration and DHCPv6 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 DHCPv6. 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 Neighbor Discovery and DHCPv6.

If inconsistent information is learned from different sources, an implementation may want to give information learned securely precedence over information learned without protection. For

instance, Section 8 of [RFC3971] discusses how to deal with information learned through Secure Neighbor Discovery conflicting with information learned through plain Neighbor Discovery. The same discussion can apply to the preference between information learned through plain Neighbor Discovery and information learned via secured DHCPv6, and so on.

In any case, if there is no security difference, the most recently obtained values SHOULD have precedence over information learned earlier.

5.7. Retaining Configured Addresses for Stability

An implementation that has stable storage may want to retain addresses in the storage when the addresses were acquired using stateless address autoconfiguration. Assuming the lifetimes used are reasonable, this technique implies that a temporary outage (less than the valid lifetime) of a router will never result in losing a global address of the node even if the node were to reboot. When this technique is used, it should also be noted that the expiration times of the preferred and valid lifetimes must be retained, in order to prevent the use of an address after it has become deprecated or invalid.

Further details on this kind of extension are beyond the scope of this document.

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 stateless address autoconfiguration and Duplicate Address Detection opens up the possibility of several denial-of-service attacks. For example, any node can respond to Neighbor Solicitations for a tentative address, causing the other node to reject the address as a duplicate. A separate document [RFC3756] discusses details about these attacks, which can be addressed with the Secure Neighbor Discovery protocol [RFC3971]. It should also be noted that [RFC3756] points out that the use of IP security is not always feasible depending on network environments.

7. Acknowledgements

Thomas Narten and Susan Thompson were the authors of RFCs 1971 and 2462. For this revision of the RFC, Tatuya Jinmei was the sole editor.

The authors of RFC 2461 would like to thank the members of both the IPNG (which is now IPV6) 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.

A number of people have contributed to identifying issues with RFC 2461 and to proposing resolutions to the issues as reflected in this version of the document. In addition to those listed above, the contributors include Jari Arkko, James Carlson, Brian E. Carpenter, Gregory Daley, Elwyn Davies, Ralph Droms, Jun-ichiro Itojun Hagino, Christian Huitema, Suresh Krishnan, Soohong Daniel Park, Markku Savela, Pekka Savola, Hemant Singh, Bernie Volz, Margaret Wasserman, and Vlad Yasevich.

8. References

8.1. Normative References

[RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, December 1998.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

8.2. Informative References

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC3484] Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)", RFC 3484, February 2003.

[RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, March 2005.

[RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, October 1999.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[RFC3590] Haberman, B., "Source Address Selection for the Multicast Listener Discovery (MLD) Protocol", RFC 3590, September 2003.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[RFC3756] Nikander, P., Kempf, J., and E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756, May 2004.

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, August 1989.

[IEEE802.11] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", ANSI/IEEE STd 802.11, August 1999.

Appendix A. Loopback Suppression and 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 simply by 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. There is even a link-layer specification that requires that any such packets be discarded [IEEE802.11]. 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:

- If a node performing Duplicate Address Detection discards received packets that have the same source link-layer address as the receiving interface, it will also discard packets from other nodes that also use 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, if it is possible to disable the suppression.

- If a node that is already using a particular IP address discards received packets that have the same link-layer source address as the interface, it will also discard Duplicate Address Detection-related Neighbor Solicitation messages sent by another node that also use 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 interface's. It should also be noted that a link-layer specification can conflict with the condition necessary to make Duplicate Address Detection work.

Appendix B. Changes since RFC 1971

- Changed document to use term "interface identifier" rather than "interface token" for consistency with other IPv6 documents.

- Clarified definition of deprecated address to make clear it is OK to continue sending to or from deprecated addresses.

- 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.

- 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.

Appendix C. Changes since RFC 2462

Major changes that can affect existing implementations:

- Specified that a node performing Duplicate Address Detection delay joining the solicited-node multicast group, not just delay sending Neighbor Solicitations, explaining the detailed reason.

- Added a requirement for a random delay before sending Neighbor Solicitations for Duplicate Address Detection if the address being checked is configured by a multicasted Router Advertisements.

- Clarified that on failure of Duplicate Address Detection, IP network operation should be disabled and that the rule should apply when the hardware address is supposed to be unique.

Major clarifications:

- Clarified how the length of interface identifiers should be determined, described the relationship with the prefix length advertised in Router Advertisements, and avoided using a particular length hard-coded in this document.

- Clarified the processing of received neighbor advertisements while performing Duplicate Address Detection.

- Removed the text regarding the M and O flags, considering the maturity of implementations and operational experiences. ManagedFlag and OtherConfigFlag were removed accordingly. (Note that this change does not mean the use of these flags is deprecated.)

- Avoided the wording of "stateful configuration", which is known to be quite confusing, and simply used "DHCPv6" wherever appropriate.

- Recommended to perform Duplicate Address Detection for all unicast addresses more strongly, considering a variety of different interface identifiers, while keeping care of existing implementations.

- Clarified wording in Section 5.5.4 to make clear that a deprecated address specified by an application can be used for any communication.

- Clarified the prefix check described in Section 5.5.3 using more appropriate terms and that the check is done against the prefixes of addresses configured by stateless autoconfiguration.

- Changed the references to the IP security Authentication Header to references to RFC 3971 (Secure Neighbor Discovery). Also revised the Security Considerations section with a reference to RFC 3756.

- Added a note when an implementation uses stable storage for autoconfigured addresses.

- Added consideration about preference between inconsistent information sets, one from a secured source and the other learned without protection.

Other miscellaneous clarifications:

- Removed references to site-local and revised wording around the keyword.

- Removed redundant code in denial-of-service protection in Section 5.5.3.

- Clarified that a unicasted Neighbor Solicitation or Advertisement should be discarded while performing Duplicate Address Detection.

- Noted in Section 5.3 that an interface can be considered as becoming enabled when a wireless access point changes.

Authors' Addresses

Susan Thomson
Cisco Systems

EMail: sethomso@cisco.com

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

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

Tatuya Jinmei
Corporate Research & Development Center, Toshiba Corporation
1 Komukai Toshiba-cho, Saiwai-ku
Kawasaki-shi, Kanagawa 212-8582
Japan

Phone: +81 44-549-2230
EMail: jinmei@isl.rdc.toshiba.co.jp

Full Copyright Statement

Intellectual Property