INTERNET-DRAFT
<draft-ietf-dnsext-ipv6-name-auto-reg-01.txt>
Expires in six months

H. Kitamura
NEC Corporation

23 July 2003

Domain Name Auto-Registration for Plugged-in IPv6 Nodes
<draft-ietf-dnsext-ipv6-name-auto-reg-01.txt>

Status of this Memo

Abstract

This document describes a scheme of "Domain Name Auto-Registration for Plugged-in IPv6 Nodes" mechanism that makes it possible to register both regular and inverse domain name information of plugged-in IPv6 nodes to DNS servers automatically.

Since IPv6 addresses are too long to remember and EUI64-based addresses are too complicated to remember, there are strong requirements to use logical names that are easy to remember instead of IPv6 addresses to specify IPv6 nodes and to register domain name information of plugged-in IPv6 nodes automatically.

In order to meet the requirements, a mechanism is proposed as one of the IPv6 auto-configuration (plug and play) functions. After the Address Autoconfiguration [ADDR-AUTO] has been executed, it works as a succeeding plug and play mechanism.

This document clarifies problems that we meet when we apply the Dynamic Updates in the DNS [DYN-DNS] to automatic domain name information registration mechanisms. This document describes the Domain Name Auto-Registration mechanism as a solution to these problems.

The Domain Name Auto-Registration mechanism, in addition to its main functionality, provides two types of additional benefits.

One is that IP address information that should be registered to the DNS is collected automatically. The mechanism can also be used under (non plug and play) manual configuration situations in a different manner from its main functionality. Under such situations, network administrators meet a problem that it is not easy to collect IP address information to register to the DNS. The mechanism solves it.

The other is that a plugged-in IPv6 node can obtain its domain name information (FQDN and DNS zone suffix) without having new functions installed into it. By simply executing inverse DNS name resolving procedures with its IPv6 address argument, the plugged-in IPv6 node can obtain its FQDN and DNS zone suffix with ease.

1. Introduction

This document describes a scheme of "Domain Name Auto-Registration for Plugged-in IPv6 Nodes" mechanism that makes it possible to register both regular and inverse domain name information of plugged-in IPv6 nodes to DNS servers automatically.

In order to specify destination nodes to communicate, SOME identifiers are necessary for users. Since IPv6 addresses are too long to remember and EUI64-based addresses are too complicated to remember, they are not suitable for such identifiers. Logical names are suitable identifiers because they are easy to remember. Therefore, there are strong requirements to use logical names instead of IPv6 addresses to specify IPv6 destination nodes and to register domain name information of plugged-in IPv6 nodes automatically.

In order to meet the requirements, a mechanism is proposed as one of the IPv6 auto-configuration (plug and play) functions. After the Address Autoconfiguration [ADDR-AUTO] has been executed, it works as a succeeding plug and play mechanism.

It is known that the Dynamic Updates in the DNS [DYN-DNS] have already been defined and that they can help automatic domain name information registration mechanisms. However, some problems need to be solved to apply this idea to actual situations.

This document clarifies problems that we meet when we apply the Dynamic Updates in the DNS [DYN-DNS] to automatic domain name information registration mechanisms. This document describes the Domain Name Auto-Registration mechanism as a solution to these problems.

Basic target situations for the mechanism are plug and play situations. Accordingly, it has been designed for plugged-in IPv6 nodes under plug and play situations.

We have to consider the following issues to design the "Domain Name Auto-Registration for Plugged-in IPv6 Nodes" mechanism.

1. Plugged-in IPv6 nodes do not have sufficient capability to show their preferences. In most cases, it is difficult for plugged-in IPv6 nodes to show their preferences for their domain names.

2. Since it is not easy to install new function to all IPv6 nodes, it is desirable to achieve the mechanism without installing new functions into plugged-in IPv6 nodes.

3. It is essential to have (register) SOME domain name for a plugged-in node. It is NOT main concern for a plugged-in node which actual name is assigned to it.

Thus, the idea of "default domain name" is introduced. When a new plugged-in IPv6 node appears, its appearance is automatically detected and a default domain name is selected for it, and both regular and inverse information of the default domain name are registered to appropriate DNS servers.

This document does not deal with cases where IPv6 nodes want to register domain names that they absolutely prefer. Such cases do not fall within the target range of plug and play situations; they will be supported under manual configuration situations.

There are various types of plugged-in IPv6 nodes that can/cannot show their preferences for their domain names. In order to meet various plug and play situations, this document considers several cases.

The Domain Name Auto-Registration mechanism is basically designed for domain name registrations for global unicast addresses. By setting the query scope of the target DNS server appropriately, the mechanism will be able to be applied to domain name registrations for site-local and link-local scope unicast addresses.

The Domain Name Auto-Registration mechanism, in addition to its main functionality, provides two types of additional benefits.

One is that IP address information that should be registered to the DNS is collected automatically. The mechanism can also be used under (non plug and play) manual configuration situations in a different manner from its main functionality. Under such situations where network is maintained by administrators manually, administrators meet

a problem that it is not easy to collect IP address information to register to the DNS. The mechanism solves the problem, and IP address information to register to the DNS is collected automatically.

Under manual configuration situations, the automatic DNS registration procedure that is the last procedure of the mechanism can be replaced by the administrators' manual registration (not by the Dynamic Updates).

The other is that a plugged-in IPv6 node can obtain its domain name information (FQDN and DNS zone suffix) with ease. The plugged-in IPv6 nodes know its IPv6 address that are automatically configured by the Address Autoconfiguration [ADDR-AUTO]. By simply executing inverse DNS name resolving procedures with the IPv6 address argument, the plugged-in IPv6 node can obtain information on its domain names (FQDN and DNS zone suffix) easily. Since all IPv6 nodes have DNS name resolving functions for both regular and inverse queries, this procedure can be achieved without installing new functions into IPv6 nodes.

2. Problems in applying the Dynamic Updates mechanism

This section clarifies problems that we meet when we apply the Dynamic Updates in the DNS [DYN-DNS] to automatic domain name information registration mechanisms.

1: Ordinary DNS servers accept Dynamic Updates messages only from trusted nodes.

   Since it is difficult for plugged-in IPv6 nodes to become trusted nodes of the DNS servers, Dynamic Updates messages from plugged-in IPv6 nodes are usually not accepted by the DNS servers.

2: It is difficult for plugged-in IPv6 nodes to know the location of the appropriate DNS server to register their domain name information to.
   ([DNS-DISC] discusses on issues of this type.)

3: It is difficult for plugged-in IPv6 nodes to prepare sufficient domain name information to register. They need to know their DNS zone suffix information to prepare FQDN for registration, but it is difficult for them to acquire it.
   ([DNS-DISC] also discusses on issues of this type.)

4: There is no explicit method to avoid duplicated, conflicting name registrations.

When a DNS server receives Dynamic Updates messages that are correctly formatted and authenticated, the server accepts them and updates DNS database with them without checking for duplication. (It is essentially difficult for a DNS server to distinguish overwrite (update) registrations from duplicate registrations.)

5: Basically, there is no mechanism to control (restrict) the number of issued Dynamic Updates messages for plugged-in nodes.

In order to minimize the effects of malicious or misconfigured registration requests, it is necessary to control them.

6: It is not clear when domain name registration requests must be issued. It is necessary to define trigger timings to start registrations.

3. Basic Design of the Domain Name Auto-Registration

This section describes the basic design of the Domain Name Auto-Registration mechanism. The mechanism solves all of the above-mentioned problems.

3.1 Design Policies

The Domain Name Auto-Registration mechanism is composed of two new functions. One is the "Detector" function, which detects appearances of new plugged-in IPv6 nodes. The other is the "Registrar" function, which registers detected domain name information to DNS servers. These functions are introduced into the IPv6 network system to achieve the mechanism.

There are several reasons why the mechanism is implemented as two functions.

1. To make the mechanism easy to control

   By concentrating administrative operations only on the Registrar side, administrative costs are reduced and the mechanism is basically maintained by administering only Registrars.

   The number of DNS servers' trusted nodes that require much administrative operation is reduced.

   Since registration information is aggregated at Registrars, it becomes easy to control registrations and minimize the effects of malicious or misconfigured registration requests.

2. To make Detector easy to implement

   There are certain constraints in placing Detectors on the IPv6 network. Thus, it is necessary for Detectors to be simple enough to be installed on IPv6 nodes of any type.

   This need is filled by putting all the intelligent operations into Registrars.

   Furthermore, the system becomes well balanced since intelligent operations are not placed on each end link.

3. To make the mechanism flexible and enable it to be applied to various environments (office networks, home networks, etc.)

   If the mechanism is applied to home networks, Registrars can be placed at the Provider side, and Detectors can be placed at the User side.

Figure 1 and 2 show typical examples that indicate locations where Detector and Registrar functions are placed on the IPv6 network.

Figure 1 shows a case for a single link, and Figure 2 shows a case for multiple links.

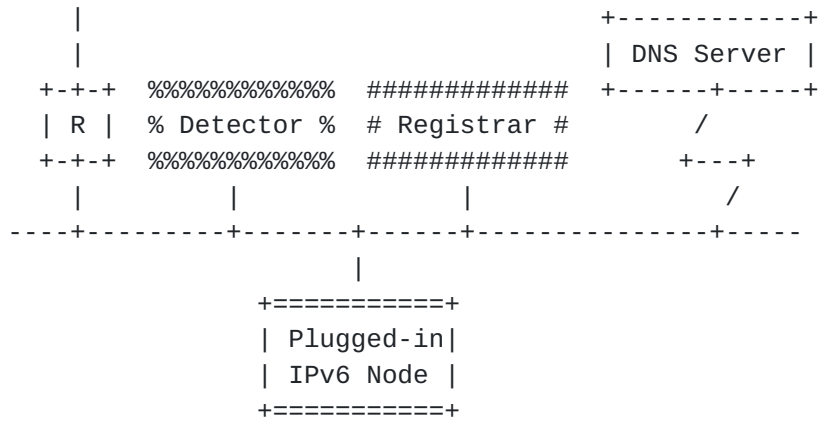

Fig. 1 Single-Link Case Example

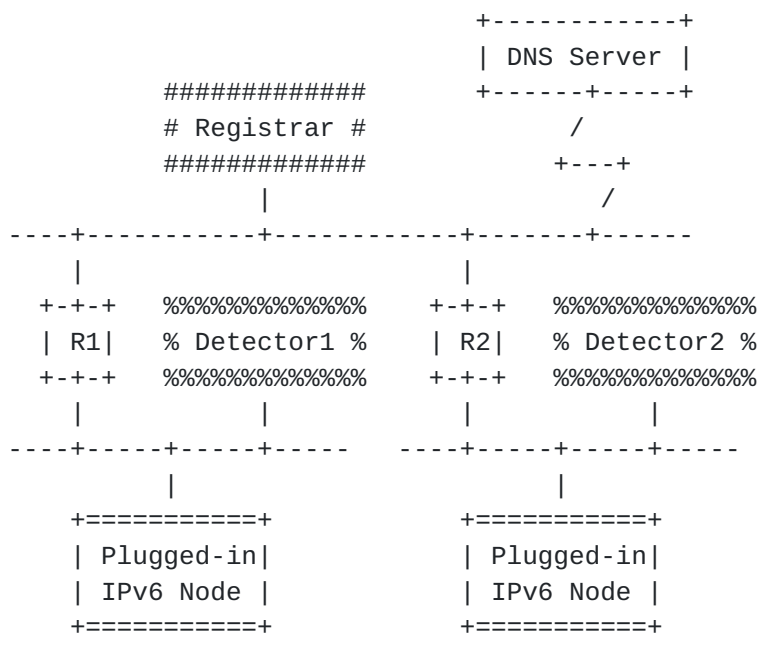

Fig. 2 Multiple-Link Case Example

One Registrar can take charge of multiple Detectors, and one Registrar can cover multiple DNS zones.

Multiple Detectors can provide detected information for one DNS zone. If the corresponding Registrars of these Detectors are different, multiple Registrars can cover one DNS zone.

Therefore, Registrars must be designed to support both cases.

3.2 Detector Function

The role of a "Detector" is to detect appearances of new plugged-in IPv6 nodes and to send the detected information to a "Registrar" without applying any selection rules to it.

Detectors are NOT required to perform any "intelligent" operations.

A Detector knows the location of its corresponding Registrar. (This location is configured manually.) Detected information must be sent securely from the Detector to the Registrar by using some kind of secure communication method (e.g., [TSIG]-like authentication, IPsec (AH, ESP), [TLS]).

Since a Detector must be placed where appearances of new plugged-in IPv6 nodes can be detected, the Detector location is restricted.

In typical cases, appearances are detected by watching for DAD packets that are issued from plugged-in IPv6 nodes (see section 3.4). So, the Detector must be placed where it can listen to link-local scope multicast packets. In other words, a Detector must be placed on each link to achieve the mechanism.

The Detector function can be implemented on routers, because its operations are simple and lightweight and routers are located at suitable places for listening to link-local scope multicast packets that are issued from plugged-in IPv6 nodes.

In order to identify Detectors, each Detector must have its own Detector ID. Since a Detector is placed on each link, the Detector's IP address that is connected to its watching link can be used for the Detector ID. (Default Address Selection for IPv6 [DEF-ADDR] algorithm is also applied here.) When a Detector sends detected information to a Registrar, the Detector ID is attached to it.

In order to meet "temporary address" [RFC3041] issues (see section 5), a link-layer address of a detected IP address is also attached to detected information.

Some simple protocol is necessary to send detected information from the Detector to the Registrar. In Appendix A, [HTTP]-based or [TLS-HTTP]-based simple protocol is shown.

3.3 Registrar Function

The role of a "Registrar" is to prepare appropriate domain name information for registration and to register it by sending Dynamic Update messages to the corresponding "DNS servers".

Appropriate domain name information for registration is created from detected information that is sent from the Detector. Some sort of intelligent algorithm is necessary in such procedures. One of the roles of the algorithm is to minimize the effects of malicious or misconfigured registration requests.

Registrars are required to perform "intelligent" operations.

By using some sort of algorithm, the Registrar verifies (checks) whether detected information must be registered (see section 3.5). After the verification procedures are completed, the Registrar selects a "default domain name" for the detected information.

In order to prepare appropriate domain name information, the Registrar must know the appropriate DNS zone suffix for detected information. (The suffix is configured manually.) The DNS zone suffix depends on the Detector ID information.

A Registrar must know the locations of "DNS servers" that correspond to detected information for registration (both regular DNS zone prefix and its inverse zone). Registrars must be trusted nodes of the DNS servers and Dynamic Update messages must be sent securely from the Registrar to the DNS servers by using some sort of secure communication methods. The [TSIG] technique would be suitable for authenticating the messages.

A Registrar has a database table to manage such knowledge. The following elements are managed in the database table:
Detector IDs, DNS zone suffixes, locations of DNS servers, applied algorithms (naming rules, how to deal with link-local or site-local scope addresses, etc.) and keys for secure communications.

A Registrar can be placed anywhere in the IPv6 network, because the Registrar communicates only with Detectors and DNS servers, all communications are unicast.

In order to optimize the communication path for packets between them, the Registrar is usually placed in the network upstream from the Detectors (see Fig.2).

Detected information that is sent from Detectors is aggregated at the Registrar.

The Registrar may frequently execute inverse DNS name resolving procedures to verify (check) whether detected information must be registered. It is recommended to put a DNS cache server function on the same node where the Registrar is placed to reduce inverse DNS name resolving traffic (see section 3.5).

3.4 Methods of Detecting Appearances of New Plugged-in IPv6 Nodes

In order to detect appearances of new plugged-in IPv6 nodes, the Detector must watch or receive packets from new plugged-in nodes. Accordingly, detection methods on the Detector are categorized into two types.

One is detection of the appearance of "standard" plugged-in nodes that do not issue special packets to show their appearance. The other is detection of the appearance of "active" plugged-in nodes that issue special packets to show their appearance.

We can assume there will be complex cases in which standard and active plugged-in nodes are mixed together. For purposes of simplification, such cases are not discussed here.

3.4.1 Detecting Appearance of "Standard" Plugged-in IPv6 Nodes

In this case, plugged-in nodes do NOT issue special dedicated packets to show their appearance. (Current standard networks are composed of such nodes.) So, the Detector must watch for packets that are issued somewhere from new plugged-in nodes.

The initial procedure for a standard plugged-in IPv6 node is to auto-configure its address and do DAD (Duplicate Address Detection) [ADDR-AUTO].

DAD packets have sufficient characteristics for an appearance-detection method, because they are issued only when IPv6 nodes are plugged in, and address information for the plugged-in IPv6 nodes is included in DAD packets.

By watching for only DAD packets, the Detector can detect appearances of new plugged-in IPv6 nodes, and DAD packets become triggers to start Domain Name Auto-Registration.

This method enables the mechanism to function without introducing new protocols and without installing new functions into plugged-in IPv6 nodes.

DAD packets are issued not only for global addresses but also for link-local or site-local scope addresses. All detected information is sent to the Registrar, and the manner of dealing with information for non-global addresses is determined by Registrar algorithms that are indicated by Detector IDs of the detected information.

This method works effectively on ordinary IPv6 links where DAD packets are issued. However, on extraordinary IPv6 links where DAD packets are not issued, it does not work. On such links, there must be another initial procedure that substitutes the DAD function. Such a procedure can be used as a trigger for a detection method on extraordinary IPv6 links.

(IP addresses can be assigned by other methods (e.g., DHCP). Domain name registration mechanisms for such cases will be discussed further in other documents.)

3.4.2 Detecting Appearance of "Active" Plugged-in Nodes

In this case, plugged-in nodes issue special dedicated packets to show their appearance. The Detector must listen for and receive packets from the new plugged-in nodes.

Since plugged-in nodes do not know the location of the Detector, anycast or multicast packets are used for the special dedicated packets.

In this method, plugged-in nodes can actively show their preference for their domain names. However, it will be difficult to show their preference under plug and play situations.

In order to achieve the method, new protocols must be defined and new functions must be installed into plugged-in IPv6 nodes.
(This will be discussed further in other documents.)

3.5 Methods of Controlling Registration

If received Dynamic Update messages are correctly formatted and authenticated, the DNS server accepts them without checking for any duplication, because the DNS server can not distinguish overwrite (update) registrations from duplicate registrations. It is difficult to achieve a mechanism for avoiding duplicated registrations on the DNS server side.

Therefore, registrations by the Dynamic Update messages must be controlled on the Registrar side. This control mechanism also helps to minimize the effects of malicious or misconfigured registration requests.

Plugged-in nodes may switch on and off frequently and issue DAD packets frequently. Since the Detector sends detected information without applying any selection rules to it, all detected information is sent to the Registrar. Thus, the Registrar must have some information verification mechanism to avoid duplicated registrations.

All candidate information (detected addresses) for registration is checked by using inverse DNS resolving queries of them. If there is FQDN information that matches the detected address, such registration candidates are not registered.

Only when FQDN information for it is NOT found and it is verified that the detected information is based on first appearance of the plugged-in node, appropriate domain name information for registration is prepared and both regular and inverse domain name information for it are registered to the DNS servers by the Dynamic Update messages.

By using this verification mechanism, the Registrar does not have to have a local database to maintain the status of the detected information and no DNS registration inconsistency problems are caused.

By restricting the number of Dynamic Update messages that are sent from the Registrar per unit of time, the effects of malicious or misconfigured registration requests are minimized.

3.6 Naming Rules for Default Domain Names

This section describes a method of setting "default domain names" for plugged-in nodes.

A fully qualified "default domain name" is composed of a node's original prefix part and a DNS zone suffix part that is the same for each site or link.

Since a DNS zone suffix is given to the Registrar manually, only the naming rules for a node's original prefix are discussed here. A naming rule algorithm for a node's prefix is given to the Registrar manually.

It is not necessary to define naming rules for a node's prefix explicitly in this document. Each site can define its own naming rules (algorithms) per link according to site policy.

This document shows some example naming rules for a node's prefix name.

1. Prefix Letter(s) + Number

   This is the easiest rule. First, prefix letter(s) that depends on each link (Detector ID) is/are selected, and the following number is selected after that.

   The following numbers comprise sequential numbers. In order to achieve this, the Registrar must remember the last selected number.

   There are some situations where using sequential numbers is not favorable because the next number could be easily predicted. In those cases, random numbers can be used, which makes it necessary to implement the Registrar with a duplicate number check mechanism.

2. Predefined Names

The Registrar prepares predefined names (e.g., names of flowers) that are used for prefix names for plugged-in nodes. Random or sequential numbers can be used to prepare predefined names.

This method can be used for an environment where the number of plugged-in nodes can be estimated and the number is not excessively large.

3. Use Preferences of Plugged-in Nodes.

The Registrar inquires the preference or property of a plugged-in node, and uses the obtained information as a hint to define a prefix name for the plugged-in node.

There are two types of methods for plugged-in nodes to indicate their preference or property.

One is "passive" indication. Plugged-in nodes do NOT become an initiator to indicate their preferences. The Registrar becomes an initiator and issues query packets to plugged-in nodes. Existing protocol (e.g., Node Information Query [NIQ], SNMP) is used for it.

For a detected global address, the Registrar can use Node Information Query [NIQ] to obtain hint information to define a name for the plugged-in node.

By using [SNMP], the Registrar can also obtain hint information to define a name for the plugged-in node. Plugged-in nodes use parts of MIB to indicate their preferences or properties. It is possible to define a special MIB for this purpose. Also, some parts of currently existing MIB can be used for it. Most plugged-in nodes have already set some property information (OS type, version, etc.) to their MIB when they are plugged in. Such information can be used for a hint to define a prefix name. (The Registrar must have an appropriate read access right to such MIB information.)

The other is "active" indication. Plugged-in nodes become an initiator to indicate their preferences and issue special dedicated packets for it. Since plugged-in nodes do not know the location of the Detector or Registrar, anycast or multicast packets are used for them. It is possible to attach name preference information to packets that are used for showing the appearance of plugged-in nodes. The Registrar can receive such information via the Detector.

In order to achieve the "active" indication method, new protocols must be defined and new functions must be installed into plugged-in IPv6 nodes.
(This will be discussed further in other documents.)

4. Procedures of the Domain Name Auto-Registration

Figure 3 shows an example of typical Domain Name Auto-Registration procedures at IPv6 links where DAD packets are issued. DAD packets are used for the appearance detection method (for standard plugged-in IPv6 nodes).

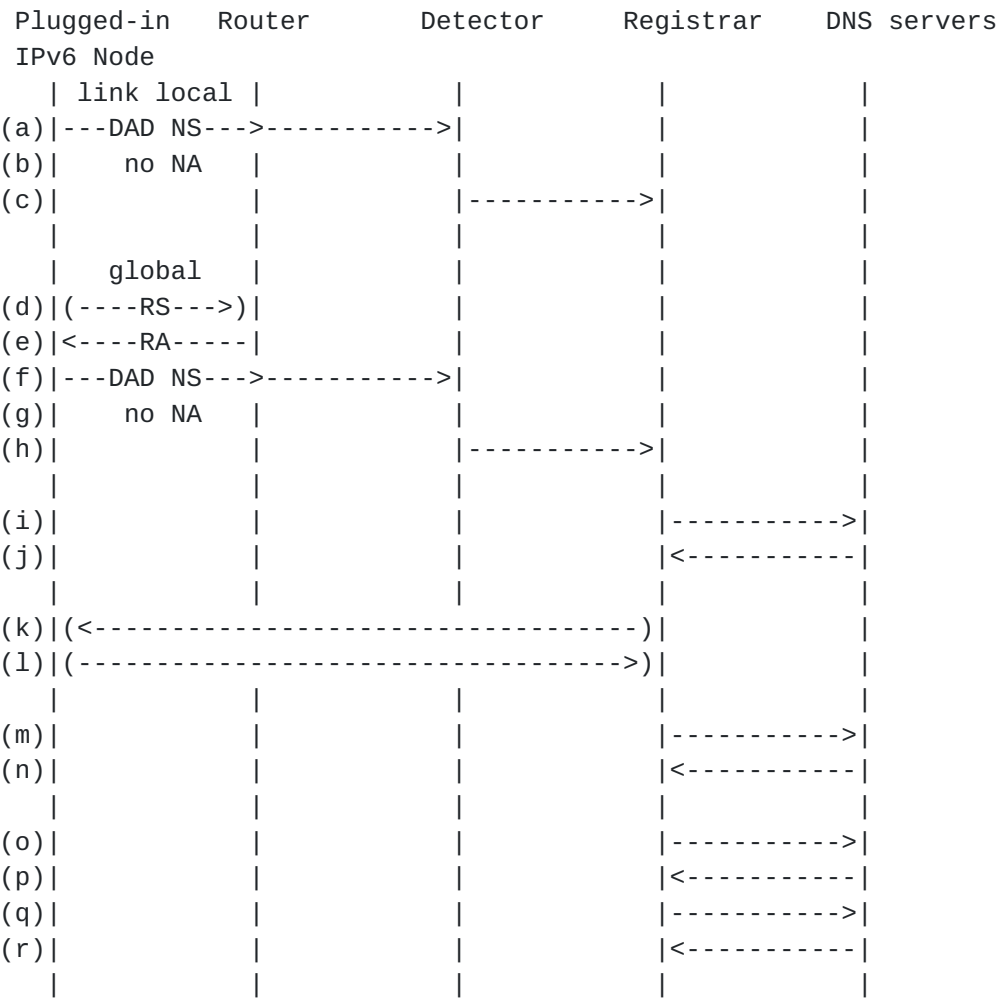

Fig. 3 Example of Typical Auto-Registration Procedures

(a) and (b) are DAD procedures for the link-local address of the Plugged-in Node. (b) is a procedure to verify that there is no NA (reply to NS) and the link-local address is not duplicated on the link.

The Detector watches (a) and (b), and detects the appearance of new plugged-in IPv6 nodes. (c) is a procedure for sending the detected information, to which the Detector ID is attached. The scope of the detected address is not checked at the Detector.

After the Registrar receives the detected information by the procedure (c), the scope of the detected address and the decision algorithm (which depends on the Detector ID) are checked on the Registrar.

In typical cases, the decision algorithm shows that link-local addresses are not candidates for registration. In such cases, the detected information for the link-local address is discarded at this point.

(d)(e)(f) and (g) are DAD procedures for the global address of the Plugged-in Node. (d) is an optional procedure. (g) is a procedure to verify that there is no NA (reply to NS) and that the global address is not duplicated.

The Detector watches (f) and (g), and detects the appearance of new plugged-in IPv6 nodes. (h) is a procedure for sending the detected information, to which the Detector ID is attached.

After the Registrar receives the detected information by the procedure (h), the scope of the detected address and decision algorithm (which depends on Detector ID) are checked on the Registrar.

In typical cases, the decision algorithm shows that global addresses are candidates for registration. In such cases, check procedures to avoid duplicated registrations are started at this point.

(i) and (j) are check procedures to verify that the detected address is must be registered to the DNS. The Registrar checks for the existence of FQDN information for the detected address by executing "inverse DNS name resolving" procedures with the detected address argument.

If the existence of FQDN information for the detected address is verified, such detected address information for registration is canceled and discarded at this point.

If the existence is not verified, the Registrar starts preparing "default domain name" information for the candidate IPv6 address. DNS zone suffix information that depends on the Detector ID is taken from the Registrar's manually configured database table, and the naming rule algorithm that depends on the Detector ID is also taken from it.

By following the defined naming rule algorithm, the plugged-in node's prefix name is selected.

(k) and (l) are optional procedures for preparing "default domain name." If the naming rule that is applied for the detected address stipulates inquiring the preference or property of the node, (k) and (l) are executed and such information is obtained by the Registrar. The obtained information is used as a hint to select the prefix name of the plugged-in node.

A candidate "default domain name" for the detected address is prepared here.

(m) and (n) are check procedures to verify that the candidate "default domain name" is not used by anyone. The Registrar checks for the existence of the candidate "default domain name" by executing "regular DNS name resolving" procedures with the candidate "default domain name."

If the existence is not verified, it becomes fully qualified "default domain name." If the existence is verified, the Registrar restarts and repeats preparing a candidate "default domain name" for the detected address.

After fully qualified "default domain name" information to register is prepared, (o)(p)(q) and (r) are executed to register both regular and inverse domain name information to the DNS servers by the Dynamic Update messages.

(Under manual configuration situations, (o)(p)(q) and (r) procedures are replaced by the administrators' manual registration (not by the Dynamic Updates).)

5. Treatment of "Temporary Addresses" in the Mechanism

"Temporary address" is defined in [RFC3041]. Temporary addresses are detected in this mechanism, because DAD packets are issued when temporary address are generated.

There are two views whether domain names for temporary addresses should be registered to the DNS or not.

One view is that domain names for temporary addresses should NOT be registered to the DNS, because temporary addresses are temporary and they are introduced to lessen privacy concerns.

The other view is domain names for temporary addresses should be registered to the DNS. [RFC3041] discusses on this issue at section 4 of [RFC3041]. In order to meet conventional inverse-DNS-based "authentication," nodes could register temporary addresses in the DNS using random names. The Domain Name Auto-Registration mechanism can provide a solution for this issue.

Since there are two views for domain names registration of temporary addresses, which view should be chosen is depends on site policies.

5.1 How to Distinguish "Temporary Addresses" from Public Addresses

In order to apply above discussed policies, it is necessary to distinguish "temporary addresses" from public addresses.

Only with IP address information, it is difficult to tell that a detected address is a temporary address or a public addresses. So, link-layer address information is utilized to achieve this operation (see section 3.2).

By utilizing link-layer address information, we can easily tell that a detected address is a EUI64-based address or not. (This operation is called a "EUI64 check" operation.)

If a detected address is a EUI64-based, it is not a temporary address. It is a normal target address for the Domain Name Auto-Registration mechanism.

If not, it must be a either temporary address or manually configured address. We can assume that a domain name for a manually configured address must have been registered in the DNS manually.

In the mechanism, an IP address whose domain name has been already registered does not become a candidate. It is verified by "inverse DNS name resolving" check procedures (see (i) and (j) procedures in Figure 3).

By applying a "EUI64 check" operation after "inverse DNS name resolving" check procedures, we can assume that non EUI64-based address must be a temporary address. Since temporary addresses are distinguished from public addresses, we can apply above discussed policies to temporary addresses.

6. Security Considerations

After the Address Autoconfiguration [ADDR-AUTO] has been executed, the Domain Name Auto-Registration works as a succeeding of the plug and play mechanism. The plugged-in IPv6 nodes' appearances detection method is depend on the Address Autoconfiguration.

Thus, the items that are described in the Security Considerations section of the Address Autoconfiguration [ADDR-AUTO] are also applicable to this document.

In addition, the following security issues are considered.

Since the Detector must send detected information to the Registrar securely, some sort of secure communication method (e.g., [TSIG]-like authentication, IPsec (AH, ESP), [TLS]) must be used.

The Registrars must be trusted nodes of the DNS servers and Dynamic Update messages must be sent securely from the Registrar to the DNS servers by using some sort of secure communication method. The [TSIG] technique would be suitable for authenticating the messages.

In order to minimize the effects of malicious or misconfigured registration requests, the Registrar restricts the number of Dynamic Update messages that are sent from the Registrar per unit of time.

Appendix A. HTTP-based simple protocol between Detector and Registrar

a. Design of HTTP parameters

- Request Parameters

```
method             = <registration protocol>
detectorID         = <detector identifier(address)>
IP-address         = <detected IP address>
link-layer-address = <detected link-layer address>
source             = <source type>
time-detected      = <detected time>
```

- Response Parameters

```
result             = <result>
address            = <registered address>
hostname           = <registered hostname>
namehint           = <name hint>
error              = <error>
time-accepted      = <accepted time>
```

b. Message Examples

- Request message

```
POST /cgi-bin/registrar.cgi HTTP/1.1
Host: registrar-host
Content-Length: mmm
User-Agent: DAD-detector
Content-type: application/x-pnp-dnar

method=register/2.0
detectorID=3ffe:xxxx::2a0:c9ff:fea6:7ff1
IP-address=3ffe:yyyy::202:b3ff:fe2d:68c0
link-layer-address=00:00:4c:zz:zz:zz
source=DAD-detector
time-detected=1013078377
```

- Response message

```
HTTP/1.1 200 OK
Content-Type : text/plain
Content-Length : nnn
Connection : close

result=REGISTER
address=3ffe:yyyy::202:b3ff:fe2d:68c0
hostname=host.example.com
namehint=none
time-accepted=1013078378
```

References

[IPv6] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC2460, December 1998.

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

[ADDR-AUTO] S. Thomson, T. Narten, "IPv6 Stateless Address Autoconfiguration," RFC2462, December 1998.

[DYN-DNS] P. Vixie, S. Thomson, Y. Rekhter, and J. Bound, "Dynamic Updates in the Domain Name System," RFC 2136, April 1997.

[TSIG] P. Vixie, O. Gudmundsson, D. Eastlake, D. and B. Wellington, "Secret Key Transaction Signatures for DNS (TSIG)," RFC 2845, May 2000.

[TLS] T. Dierks, C. Allen, "The TLS Protocol Version 1.0", RFC2246, January 1999

[DNS-SIG0] D. Eastlake, "DNS Request and Transaction Signatures (SIG(0)s)," RFC2931, September 2000.

[DYN-DNSSEC] B. Wellington, "Secure Domain Name System (DNS) Dynamic Update," RFC3007, November 2000.

[DNSSEC] B. Wellington, "Domain Name System Security (DNSSEC) Signing Authority," RFC 3008, November 2000.

[SNMP] J. Case, K. McCloghrie, M. Rose, S.Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)," RFC1905, January 1996.

[RFC3041] T. Narten, R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6," RFC3041, January 2001

[HTTP] R. Fielding, et al, "Hypertext Transfer Protocol -- HTTP/1.1" RFC2616, June 1999

[TLS-HTTP] R. Khare, S. Lawrence, "Upgrading to TLS Within HTTP/1.1" RFC2817, May 2000

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

[NIQ] M. Crawford, "IPv6 Node Information Queries,"
<draft-ietf-ipngwg-icmp-name-lookups-10.txt>, June 2003
"work in progress"

[DNS-DISC] A. Durand, J. Hagino, D. Thaler, "Well known site local
unicast addresses to communicate with recursive DNS servers,"
<draft-ietf-ipv6-dns-discovery-07.txt>, October 2002
"work in progress"

Author's Address:

Hiroshi Kitamura
Network Development Laboratories, NEC Corporation
(Igarashi Building 4F) 11-5, Shibaura 2-Chome,
Minato-Ku, Tokyo 108-8557, JAPAN

Phone: +81 (3) 5476-1071
Fax: +81 (3) 5476-1005
Email: kitamura@da.jp.nec.com