IPv6 Stateless DNS Discovery
<draft-ietf-ipv6-dns-discovery-04.txt>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet Drafts are valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet Drafts as reference material or to cite them other than as a "work in progress".

To view the list Internet-Draft Shadow Directories, see http://www.ietf.org/shadow.html.

Abstract

This document specifies the steps a host takes in deciding how to autoconfigure the addresses of DNS Servers required for name resolution in IP version 6. The autoconfiguration process includes determining whether such information should be obtained through the stateless mechanism, the stateful mechanism, or both. This document defines the process for acquiring a list of DNS server addresses. Approaches for acquiring a domain search path, and the domain name of the host via a stateless mechanism are
included in an appendix for further study. The details of autoconfiguration using the stateful protocol are specified elsewhere.

Copyright Notice

Copyright (C) The Internet Society (2002). All Rights Reserved.
1. Introduction

RFC 2462 [ADDRCONF] specifies "OtherConfigFlag" as a per-interface variable, which is set from the value of the "O" ("Other stateful configuration") flag in Router Advertisements received. When OtherConfigFlag is set on an interface, information related to name resolution is obtained using the stateful autoconfiguration mechanism. However, when OtherConfigFlag is not set, it does not describe how to obtain such information. This is the purpose of this document.

For a host to effectively resolve names of other hosts, and potentially allow resolution of its name to be performed, the following parameters are typically required:

- One or more addresses of Domain Name Service (DNS) [RFC1034, RFC1035] servers. The function of name-to-address resolution (or vice versa) in IP is performed by DNS, which requires that at least one DNS Server be known and reachable by a device desiring to perform name resolution.

- A per-interface domain name of the host itself, and is equivalent to the Domain Name option in [DHCP]. This can be used when Multicast DNS is enabled, and the host responds to queries for its own name, as well as when DNS Dynamic Update is enabled. Depending on the implementation, the per-interface domain name may or may not be the same thing as the primary domain name of the host.

- Search path. It is currently common practice for the search path to be computed by a device based on its domain name(s). However, a DHCP option [DOMSEARCH] has been proposed, and so search path configuration is likely to be a requirement in general.

- mDNS-enabled flag. This parameter controls whether Multicast DNS [MDNS] should be enabled on the host’s interface.

A design team recommendation [DDDT] contains an analysis of the requirements and solution space, which was used as the basis for this document. One result of this analysis was that there is a spectrum of configuration/discovery mechanisms. At one end, a single protocol is used to configure/discovery all types of information. This style works well in an administered environment where a network administrator wants to have a central point of
control, and apply policies, etc. At the other end, each protocol
is self-configuring, rather than depending on any other protocol
or server. This style provides optimal fate sharing, and works
well in zero-configuration environments such as adhoc networks and
simple networks without network administrator staff. The former
mechanism is used with the "Other stateful configuration" flag is
used, and this draft provides the benefits (and limitations) of
the latter approach when "Other stateful configuration" is not
set.

Note: This document only includes in its body a solution for
obtaining the address of Domain Name Service servers. Mechanisms
for obtaining the other parameters listed above are included in an
Appendix A for further study. These may be moved to a separate
document in the future.

2. Overview

A set of three well-known site-local IPv6 addresses are reserved
for autodiscovery of DNS servers. These addresses may be used as
unicast addresses assigned to different servers. The use of the
addresses as anycast addresses with one of them being assigned to
all DNS servers in the site, or any combination of anycast and
unicast addresses, is for further study.

Host routes for these addresses are propagated in the site’s
routing tables. This document proposes that these three addresses
be:

\[
\begin{align*}
  \text{fec0:0:0:ffff::1} \\
  \text{fec0:0:0:ffff::2} \\
  \text{fec0:0:0:ffff::3}
\end{align*}
\]

This list of three addresses may be hardcoded into a host.

Furthermore, we define two levels of functionality. Level 1 is
defined below. Level 2 is described in Appendix A and is for
further study.
3. Level 1 Compliance

Level 1 compliance entails using the three addresses above for actual name resolution. It provides only rudimentary functionality. In particular, it does not provide the ability to have separate configuration for hosts on different subnets, nor to provide hosts with a domain name other than one for which the DNS server is authoritative.

3.1. DNS Server Configuration

Level 1 functionality requires no DNS server configuration other than assigning one of the well-known addresses to one of the server’s interfaces. A host route must then be injected into the routing domain, e.g., by configuring a static host route on the server’s router, and redistributing it into the intra-domain routing protocol.

A host route must then be injected into the site’s routing infrastructure. Route injection can be done via any of several methods, including but not limited to:

a) Run the server on a router, and configure it to inject the host route.

b) Run a routing protocol on the server, and configure it to inject the host route. Note that this requires that the server and its router(s) must run the same routing protocol, at least for communication between the router(s) and the server(s) on the link. However, a server does not need to participate fully in the routing protocol, it only needs to be able to inject routes.

c) Run multiple servers on the same link(s), and configure their local router(s) to inject host routes for the well-known address into the site’s routing infrastructure. Running multiple servers on the same link provides robustness to the failure of a server, while routing provides robustness to the loss of routers and other links. There may still be some failures, however, such as a unidirectional failure of the router’s interface, which are not handled by this option.

d) Modify the routers on the link to periodically solicit (using Neighbor Discovery) the well-known address, and inject the
host route based on whether a reply is received.

3.2. Host Behavior

The host sets its DNS server list equal to the set of three addresses listed above. The search path is not discovered, but is generated from the host’s domain name(s) as is currently common practice. If desired, a per-interface domain name can be obtained by sending a query (with the Recursion Desired (RD) bit set), doing a reverse lookup for the well-known site-local IPv6 destination address, and extracting the domain name from the NS record in the reply.

4. Security Considerations

Ensuring that queries reach a legitimate DNS server relies on the security of the IPv6 routing infrastructure. The issues here are the same as those for protecting basic IPv6 connectivity.

IPsec/IKE can be used when the well-known addresses are used as unicast addresses.

The payload can be protected as follows. If the client can preconfigure a well known private or public key then TSIG [TSIG] can be used with the same packets presented for the query. If this is not the case, then TSIG keys will have to be negotiated using [TKEY]. After the client has the proper key then the query can be performed.

5. IANA Considerations

The IANA should assign the following site-local IPv6 addresses to be used as well-known addresses assigned to DNS servers:

```
   fec0:0:0:fff::1
   fec0:0:0:fff::2
   fec0:0:0:fff::3.
```

[Note to readers: the above addresses are tentative, but the ffff is intended to be consistent with a simultaneous proposal to reserve the ffff SLA for use with IANA-assigned addresses such as these.]
6. Acknowledgements

The IPv6 DNS Discovery Design Team [DDDT] provided recommendations that formed the basis of this specification. Rob Austein and the IPv6 Working Group provided valuable feedback on the mechanism. Aaron Schrader provided helpful comments as well. Robert Hinden edited this version of the document.

7. References

[ADDRCONF]

[ANYCAST]

[DDDT]

[DIFFSEC]

[DNSSEC]

[DOMSEARCH]

[MDNS]

[TKEY]
[TSIG]

8. Authors’ Addresses

Dave Thaler
Microsoft
One Microsoft Way
Redmond, CA 98052, USA
Email: dthaler@microsoft.com

Jun-ichiro itojun HAGINO
Research Laboratory, Internet Initiative Japan Inc.
Takebashi Yasuda Bldg.,
3-13 Kanda Nishiki-cho,
Chiyoda-ku, Tokyo 101-0054, JAPAN
Email: itojun@iijlab.net
9. Appendix A - Level 2 Compliance

Level 2 compliance allows site administrators to have site-wide defaults for all name resolution parameters, and optionally to have subnet-specific overrides. However, it defines a new DNS record type to hold name resolution configuration information. In this way, DNS becomes self-configuring.

9.1. DNS Server Configuration

A new record type, CFG, is defined, with a syntax as follows:

```
<owner> <class> <ttl> CFG "<attribute name>=<attribute value>"
```

The set of attribute names is described below. This set may be extended by other RFCs, but any new attributes MUST be specific to name resolution.

The DNS server list is expressed with a string of the form

```
dnsservers=<address>[,<address>]*
```

where the attribute value is a comma-separated list of one or more addresses (either IPv4 or IPv6) in string literal format suitable for passing to getaddrinfo.

The domain name to use is expressed with a string of the form

```
domainname=<domain>
```

where the attribute value is the domain name the client should use.

The domain suffix search path is expressed with a string of the form

```
searchpath=<domain>[,<domain>]*
```

where the attribute value is a comma-separated list of domain suffixes.

The mDNS enabled flag is expressed with a string of the form

```
mDNSenabled=<value>
```

where the attribute value is either "true" or "false".

Name resolution information can be expressed as defaults for the entire site, as well as per-subnet overrides if desired. To express site defaults, the record owner used is a wildcard, namely *".local.arpa. The format of per-subnet overrides is described in the next section.

[NOTE WELL: the use of "local.arpa" and the CFG record syntax shown above are just placeholders until DNS experts figure out what the right thing is.]
Each of the attributes described herein are optional, and any combination may be used, except that only one record per attributename should be present per owner (site or subnet) string.

```
*.local.arpa   IN   CFG   "dnsservers=fec0:0:1::1,fec0:0:2::2"
*.local.arpa   IN   CFG   "domainname=example.com"
*.local.arpa   IN   CFG   "searchpath=foo.example.com,
                        bar.example.com,example.com"
*.local.arpa   IN   CFG   "mdnsenabled=true"
```

Table 1: Example configuration

The DNS server must also be assigned one of the well-known site-local addresses, and a host route must be injected into the site’s routing infrastructure, e.g. using one of the methods described above in Section 3.1.

9.2. Host Behavior

When an interface comes up, and a host determines that the OtherConfigFlag on the interface is off, then it takes the following actions.

The host constructs a DNS query for CFG records for "<subnet>.local.arpa.", where <subnet> is constructed from an onlink prefix as follows:

1) Determine the onlink prefix to use. Any onlink site-local prefix is used, if present. Otherwise, any onlink global prefix is used. If no other onlink prefixes are present (e.g., if no routers are present), the link-local prefix is used as a last resort.

2) Convert the subnet prefix to a string by taking the lower case string literal representation, with no zero compression, and replacing all colons with underscores. Table 2 below shows some examples. This notation is used so that it uses only one token, is unique per prefix, and is human readable.
Once the query is formed, the host initially sends out the query using UDP to each discovery address in turn until a reply is received, or until the end of the list is reached. To avoid implosion problems when an entire site reboots such as after a power outage, the first request should wait 3 seconds for a reply, with the wait period doubling for each subsequent request.

Since the destination address may be an anycast address, the reply will necessarily come from a different address. The host must not discard the reply simply because the source address is different. A more detailed discussion of this issue can be found in [ANYCAST].

Upon receiving a response, the host parses the CFG records and acts on the information as follows.

If a dnsservers attribute is present, the list of server addresses is extracted from the value. If no such attribute is present (or if no reply is received), an implementation-specific default list is used. For example:

- an implementation MAY use an empty list (effectively disabling name resolution),
- a host MAY use a DNS server list containing only the anycast address, subject to the limitations discussed in the next section,
- a host MAY use mDNS [MDNS] only, or
- a host MAY use some combination of the above.

In general, the list obtained is used in the same way as if the list had been obtained (or failed to be obtained) via DHCP.

If a domainname attribute is present, the domain name is extracted
from the value. The domain name (or lack thereof) is used in the same way as if the list had been obtained (or failed to be obtained) via DHCP.

If the searchpath attribute is present, the search path is extracted from the value. If not present, the host uses the search path it would use if no path had been obtained if DHCP were in use.

If the mdnsenabled attribute is present, the value is extracted. If not present, mDNS is not enabled.
10. Full Copyright Statement

Copyright (C) The Internet Society (2001). All Rights Reserved.

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

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

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