Improved Recursive DNS Server Selection for Multi-Interfaced Nodes
draft-ietf-mif-dns-server-selection-09

Abstract

A multi-interfaced node is connected to multiple networks, some of which may be utilizing private DNS namespaces. A node commonly receives recursive DNS server configuration information from all connected networks. Some of the recursive DNS servers may have information about namespaces other servers do not have. When a multi-interfaced node needs to utilize DNS, the node has to choose which of the recursive DNS servers to contact to. This document describes DHCPv4 and DHCPv6 options that can be used to configure nodes with information required to perform informed recursive DNS server selection decisions.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents 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 "work in progress."

This Internet-Draft will expire on November 26, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction .................................................. 4
   1.1. Requirements Language ................................ 5
2. Private namespaces and problems for multi-interfaced nodes ... 5
   2.1. Fully qualified domain names with limited scopes ... 5
   2.2. Network interface specific IP addresses .......... 6
   2.3. A problem not fully solved by the described solution ... 8
3. Deployment scenarios .......................................... 8
   3.1. CPE deployment scenario ................................. 8
   3.2. Cellular network scenario ............................... 9
   3.3. VPN scenario ............................................ 9
   3.4. Dual-stack accesses ..................................... 9
4. Improved RDNSS selection ...................................... 9
   4.1. Procedure for prioritizing RDNSSes and handling responses ................. 9
   4.2. RDNSS selection DHCPv6 option .......................... 12
   4.3. RDNSS selection DHCPv4 option .......................... 14
   4.4. Limitations on use .................................... 16
   4.5. Coexistence of various RDNSS configuration tools .......... 16
   4.6. Considerations on follow-up queries .................... 17
5. Example of a node behavior ................................... 18
6. Scalability considerations ..................................... 20
7. Considerations for network administrators ................... 20
8. Acknowledgements ............................................. 20
9. IANA Considerations ........................................... 21
10. Security Considerations ...................................... 21
   10.1. Attack vectors ....................................... 21
   10.2. Trust levels of network interfaces ................... 21
   10.3. Importance of following the algorithm .............. 21
11. References .................................................. 22
   11.1. Normative References .................................. 22
   11.2. Informative References ................................. 22
Appendix A. Possible alternative practices for RDNSS selection .... 23
   A.1. Sending queries out on multiple interfaces in parallel ... 23
   A.2. Search list option for DNS forward lookup decisions ........ 24
   A.3. More specific routes for reverse lookup decision ....... 24
   A.4. Longest matching prefix for reverse lookup decision .... 24
Appendix B. DNSSEC and multiple answers validating with different trust anchors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
1. Introduction

A multi-interfaced node faces several problems a single-homed node does not encounter, as is described in [RFC6418]. This document studies in detail the problems private namespaces may cause for multi-interfaced nodes and provides a solution. The node may be implemented as a host or as a router.

We start from the premise that network operators sometimes include private namespaces in the answers they provide from Recursive DNS Servers (RDNSS), and that those private namespaces are at least as useful to nodes as the answers from the public DNS. When private namespaces are visible for a node, some RDNSSes have information other RDNSSes do not have. The node ought to be able to ask right RDNSS for the information it needs.

An example of an application that benefits from multi-interfacing is a web browser that commonly accesses many different destinations, each of which is available only on one network. The browser therefore needs to be able to communicate over different network interfaces, depending on the destination it is trying to reach.

In deployments where private namespaces are present, selection of correct route and destination and source addresses for the actual IP connection is crucial as well, as the resolved destination’s IP addresses may be only usable on the network interface over which the name was resolved on. Hence solution described in this document is assumed to be commonly used in combination with tools for delivering additional routing and source and destination address selection policies.

This document is organized in the following manner. Background information about problem descriptions and example deployment scenarios are included in Section 2 and Section 3. Section 4 contains all normative descriptions for DHCP options and node behavior. Informative Section 5 illustrates behavior of a node implementing functionality described in the Section 4. Section 6 contains informational considerations about scalability. Section 7 contains normative guidelines related to creation of private namespaces. Informational Section 10 summarizes identified security considerations.

The Appendix A describes best current practices possible with tools preceding this document and that may be possibilities on networks not supporting the solution described in this document.
1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Private namespaces and problems for multi-interfaced nodes

This section describes two node multi-interfacing related private namespace scenarios for which the procedure described in Section 4 provides a solution for. Additionally, Section 2.3 describes a problem for which this document provides only partial solution.

2.1. Fully qualified domain names with limited scopes

A multi-interfaced node may be connected to one or more networks that are using private namespaces. As an example, the node may have simultaneously open a wireless LAN (WLAN) connection to the public Internet, cellular connection to an operator network, and a virtual private network (VPN) connection to an enterprise network. When an application initiates a connection establishment to an FQDN, the node needs to be able to choose the right RDNSS for making a successful DNS query. This is illustrated in the figure 1. An FQDN for a public name can be resolved with any RDNSS, but for an FQDN of enterprise’s or operator’s service’s private name the node needs to be able to correctly select the right RDNSS for the DNS resolution, i.e. do also network interface selection already before destination’s IP address is known.
2.2. Network interface specific IP addresses

In the second problem an FQDN is valid and resolvable via different network interfaces, but to different and not necessarily globally reachable IP addresses, as is illustrated in the figure 2. Node’s routing and source and destination address selection mechanism must ensure the destination’s IP address is only used in combination with source IP addresses of the network interface the name was resolved on.
Private DNS namespaces and different IP addresses for an FQDN on interfaces 1 and 2.

Figure 2

Similar situation can happen with IPv6 protocol translation and AAAA record synthesis [RFC6147]. A synthetic AAAA record is guaranteed to be valid only on a network it was synthesized on. Figure 3 illustrates a scenario where the peer’s IPv4 address is synthesized into different IPv6 addresses by RDNSSes A and B.

AAAA synthesis results in network interface specific IPv6 addresses.

Figure 3

It is worth noting is that network specific IP addresses can cause problems also for a single-homed node, if the node retains DNS cache during movement from one network to another. After the network change, a node may have entries in its DNS cache that are no longer correct or appropriate for its new network position.
2.3. A problem not fully solved by the described solution

A more complex scenario is an FQDN, which in addition to possibly resolving into network interface specific IP addresses, identifies on different network interfaces completely different peer entities with potentially different set of service offerings. In even more complex scenario, an FQDN identifies unique peer entity, but one that provides different services on its different network interfaces. The solution described in this document is not able to tackle these higher layer issues. In fact, these problems may be solvable only by manual user intervention.

However, when DNSSEC is used, the DNSSEC validation procedure may provide assistance for selecting correct responses for some, but not all, use cases. A node may prefer to use the DNS answer that validates with the preferred trust anchor.

3. Deployment scenarios

This document has been written with three particular deployment scenarios in mind. First being a Consumer Premises Equipment (CPE) with two or more uplink VLAN connections. Second scenario involves a cellular device with two uplink Internet connections: WLAN and cellular. Third scenario is for VPNs, where use of local RDNSS may be preferred for latency reasons, but enterprise’s RDNSS must be used to resolve private names used by the enterprise.

3.1. CPE deployment scenario

A home gateway may have two uplink connections leading to different networks, as is described in [I-D.ietf-v6ops-ipv6-multihoming-without-ipv6nat]. In the two uplinks scenario only one uplink connection leads to the Internet, while another uplink connection leads to a private network utilizing private namespaces.

It is desirable that the CPE does not have to send DNS queries over both uplink connections, but instead CPE should send default queries to the RDNSS of the interface leading to the Internet, and queries related to private namespace to the RDNSS of the private network.

In this scenario the legacy hosts can be supported by deploying DNS proxy on the CPE and configuring hosts in the LAN to talk to the DNS proxy. However, updated hosts would be able to talk directly to the correct RDNSS of each uplink ISP’s RDNSS. It is deployment decision whether the updated hosts would be pointed to DNS proxy or to actual RDNSSes.
Depending on actual deployments, all VLAN connections might be considered as trusted.

3.2. Cellular network scenario

A cellular device may have both WLAN and cellular network interfaces up. In such a case it is often desirable to use WLAN by default, except for those connections cellular network operator wants to go over cellular interface. The cellular network may utilize private names and hence the cellular device needs to ask for those through the cellular interface.

In this scenario cellular interface can be considered trusted and WLAN oftentimes untrusted.

3.3. VPN scenario

Depending on a deployment, there may be interest to use VPN only for the traffic destined to a enterprise network. The enterprise may be using private namespace, and hence related DNS queries should be send over VPN to the enterprise’s RDNSS, while by default RDNSS of a local access network may be used.

In this scenario VPN interface can be considered trusted and local access network untrusted.

3.4. Dual-stack accesses

In all three scenarios one or more of the connected networks may support both IPv4 and IPv6. In such a case both or either of DHCPv4 and DHCPv6 can be used to learn RDNSS selection information.

4. Improved RDNSS selection

This section describes DHCP options and a procedure that a (stub / proxy) resolver may utilize for improved RDNSS selection in the face of private namespaces and multiple simultaneously active network interfaces.

4.1. Procedure for prioritizing RDNSSes and handling responses

A resolver SHALL build a priority list of RDNSSes it will contact to depending on the query. To build the list in an optimal way, a node SHOULD ask with DHCP which RDNSSes of each network interface are most likely to be able to successfully serve forward lookup requests matching to specific domain or reverse (PTR record) lookup requests matching to specific network addresses (later referred as "network").
For security reasons the RDNSS selection information MUST be used only when it is safe to do so, see Section 4.4 for details.

The node SHOULD create node specific routes for RDNSS addresses learned via DHCP. The route must point to the interface each RDNSS address was learned on. This is required to ensure DNS queries are sent out via the right network interface.

A resolver lacking more specific information shall assume that all information is available from any RDNSS of any network interface. The RDNSSes learnt by other RDNSS address configuration methods MUST be handled as medium priority default RDNSSes (see also Section 4.5).

When a DNS query needs to be made, the resolver SHOULD give highest precedence to the RDNSSes explicitly known to serve matching domain or network. The resolver MUST take into account differences in trust levels of pieces of received RDNSS selection information. The resolver MUST prefer RDNSSes of trusted interfaces. The RDNSSes of untrusted interfaces may be of highest priority only if trusted interfaces specifically configure low priority RDNSSes. The non-exhaustive list on figure 4 illustrates how the different trust levels of received RDNSS selection information SHOULD influence the RDNSS selection logic.

Trustworthiness of an interface and configuration information received over the interface is implementation and/or node deployment dependent. Trust may be based on, for example, on the nature of an interface. For example, an authenticated and encrypted VPN or layer 2 connections to a trusted home network may be considered as trusted, and an unauthenticated and unencrypted connection to an unknown visited network may be considered as untrusted. In some occasions an interface may be considered trusted only if explicitly configured to be trusted.

A resolver SHOULD prioritize between equally trusted RDNSSes with help of the DHCP option preference field. The resolver MUST NOT prioritize less trusted RDNSSes higher than trusted, even in the case of less trusted RDNSS would apparently have additional information. In the case of all other things being equal the resolver shall make the prioritization decision based on its internal preferences.
<table>
<thead>
<tr>
<th>Information from more trusted interface A</th>
<th>Information from less trusted interface B</th>
<th>Resulting RDNSS priority selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Medium priority default</td>
<td>Medium priority default</td>
<td>Default: A, then B</td>
</tr>
<tr>
<td>2. Medium priority default</td>
<td>High priority default</td>
<td>Default: A, then B</td>
</tr>
<tr>
<td>3. Low priority default</td>
<td>Medium priority default</td>
<td>Default: B, then A</td>
</tr>
<tr>
<td>4. Low priority default</td>
<td>Medium priority default</td>
<td>Default: B, then A</td>
</tr>
</tbody>
</table>

Figure 4: RDNSS selection in the case of different trust levels

Because DNSSEC provides cryptographic assurance of the integrity of DNS data, data that can be validated under DNSSEC is necessarily to be preferred over data that cannot be. There are two ways that a node can determine that data is valid under DNSSEC. The first is to perform DNSSEC validation itself. The second is to have a secure connection to an authenticated RDNSS, and to rely on that RDNSS to perform DNSSEC validation (signalling that it has done so using the AD bit). If a DNS response is not proven to be unmolested using DNSSEC, then a node cannot make a decision to prefer data from any interface with any great assurance: any response could be forged, and there is no way to detect the forgery without DNSSEC.

A node SHALL send requests to RDNSSes in the order defined by the priority list until an acceptable reply is received, all replies are received, or a time out occurs. In the case of a requested name matching to a specific domain or network rule accepted from any interface, a DNSSEC-aware resolver MUST NOT proceed with a reply that cannot be validated using DNSSEC until all RDNSSes on the priority list have been contacted or timed out. This protects against possible redirection attacks. In the case of the requested name not matching to any specific domain or network, first received response from any RDNSS MAY be considered acceptable. A DNSSEC-aware node MAY always contact all RDNSSes in an attempt to receive a response that can be validated, but contacting all RDNSSes is not mandated for the default case as in some deployments that would consume excess resources.

The resolver SHOULD avoid sending queries over different network interfaces in parallel as that wastes resources such as energy. The
amount of wasted energy can be significant, for example when radio interfaces has to be started just for the queries.

In the case of validated NXDOMAIN response being received from a RDNSS that can provide answers for the queried name a node MUST NOT accept non-validated replies from other RDNSSes (see Appendix B for considerations related to multiple trust anchors.

4.2. RDNSS selection DHCPv6 option

DHCPv6 option described below can be used to inform resolvers which RDNSS should be contacted when initiating forward or reverse DNS lookup procedures. This option is DNS record type agnostic and applies, for example, equally to both A and AAAA queries.
option-code:  OPTION_DNS_SERVER_SELECT (TBD)

option-len:  Length of the option in octets

DNS-recursive-name-server:  An IPv6 address of RDNSS

Reserved:  Field reserved for the future. MUST be set to zero.

prf:  RDNSS preference, for selecting between equally trusted RDNSSes:
  01 High
  00 Medium
  11 Low
  10 Reserved

Domains and networks:  The list of domains for forward DNS lookup and networks for reverse DNS lookup the RDNSS has special knowledge about. Field MUST be encoded as specified in Section "Representation and use of domain names" of [RFC3315]. Special domain of "." is used to indicate capability to resolve global names and act as a default RDNSS. Lack of "." domain on the list indicates RDNSS only has information related to listed domains and networks. Networks for reverse mapping are encoded as defined for ip6.arpa [RFC3596] or in-addr.arpa [RFC2317].

DHCPv6 option for explicit domain configuration

Figure 5
A node SHOULD include an OPTION_ORO option in a DHCPv6 request with the OPTION_DNS_SERVER_SELECT option code to inform the DHCPv6 server about the support for the improved RDNSS selection logic. DHCPv6 server receiving this information MAY then choose to provision RDNSS addresses only with the OPTION_DNS_SERVER_SELECT.

The OPTION_DNS_SERVER_SELECT contains one or more domains the related RDNSS has particular knowledge of. The option can occur multiple times in a single DHCPv6 message, if multiple RDNSS are to be configured.

IPv6 networks should cover all the domains configured in this option. Networks should be as long as possible to avoid potential collision with information received on other option instances or with options received from DHCPv6 servers of other network interfaces. Overlapping IPv6 networks are interpreted so that the resolver can use any of the RDNSSes for queries matching the networks.

If the OPTION_DNS_SERVER_SELECT contains a RDNSS address already learned from other DHCPv6 servers of the same network, and contains new domains or networks, the node SHOULD append the information to the information received earlier. The node MUST NOT remove previously obtained information. However, the node SHOULD NOT extend lifetime of earlier information either. In the case of conflicting RDNSS address is learned from less trusted interface, the node MUST ignore the option.

As the RDNSS options of [RFC3646], the OPTION_DNS_SERVER_SELECT option MUST NOT appear in any other than the following DHCPv6 messages: Solicit, Advertise, Request, Renew, Rebind, Information-Request, and Reply.

The information conveyed in OPTION_DNS_SERVER_SELECT is considered valid until changed or refreshed by general events that trigger DHCPv6 action. In the event that it is desired for the client to request a refresh of the information, use of generic DHCPv6 Information Refresh Time Option, as specified in [RFC4242] is RECOMMENDED.

4.3. RDNSS selection DHCPv4 option

DHCPv4 option described below can be used to inform resolvers which RDNSS should be contacted when initiating forward or reverse DNS lookup procedures. This option is DNS record type agnostic and applies, for example, equally to both A and AAAA queries.
<table>
<thead>
<tr>
<th>CODE</th>
<th>Len</th>
<th>Reserved</th>
<th>prf</th>
<th>Primary ..</th>
</tr>
</thead>
<tbody>
<tr>
<td>.. DNS-recursive-name-server’s IPv4 address</td>
<td>Secondary ..</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.. DNS-recursive-name-server’s IPv4 address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+</td>
<td>Domains and networks (variable length)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

option-code: OPTION_DNS_SERVER_SELECT (TBD)

option-len: Length of the option in octets

Reserved: Field reserved for the future. MUST be set to zero.

prf: RDNSSes preference, for selecting between equally trusted RDNSSes:

01 High
00 Medium
11 Low
10 Reserved

Primary DNS-recursive-name-server’s IPv4 address: Address of a primary RDNSS

Secondary DNS-recursive-name-server’s IPv4 address: Address of a secondary RDNSS or 0.0.0.0 if not configured

Domains and networks: The list of domains for forward DNS lookup and networks for reverse DNS lookup the RDNSSes have special knowledge about. Field MUST be encoded as specified in Section "Representation and use of domain names" of [RFC3315]. Special domain of "." is used to indicate capability to resolve global names and act as default RDNSS. Lack of "." domain on the list indicates RDNSSes only have information related to listed domains and networks. Networks for reverse mapping are encoded as defined for ip6.arpa [RFC3596] or in-addr.arpa [RFC2317].

DHCPv4 option for explicit domain configuration
The OPTION_DNS_SERVER_SELECT contains one or more domains the primary and secondary RDNSSes have particular knowledge of. If the length of the domains and networks field causes option length to exceed the maximum permissible for a single option (255 octets), then multiple options MAY be used, as described in "Encoding Long Options in the Dynamic Host Configuration Protocol (DHCPv4)" [RFC3396]. When multiple options are present, the data portions of all option instances are concatenated together.

If the OPTION_DNS_SERVER_SELECT contains a RDNSS address already learned from other DHCPv4 servers of the same network, and contains new domains or networks, the node SHOULD append the information to the information received earlier. The node MUST NOT remove previously obtained information. However, the node SHOULD NOT extent lifetime of earlier information either. In the case of conflicting RDNSS address is learned from less trusted interface, the node MUST ignore the option.

4.4. Limitations on use

Use of OPTION_DNS_SERVER_SELECT is ideal in the following environments, but SHOULD NOT be enabled by default otherwise:

1. The RDNSS selection option is delivered across a secure, trusted channel.

2. The RDNSS selection option is not secured, but the client on a node does DNSSEC validation.

3. The RDNSS selection option is not secured, the resolver does DNSSEC validation, and the client communicates with the resolver configured with RDNSS selection option over a secure, trusted channel.

4. The IP address of RDNSS that is being recommended in the RDNSS selection option is known and trusted by the client; that is, the RDNSS selection option serves not to introduce the client to a new RDNSS, but rather to inform it that RDNSS it has already been configured to trust is available to it for resolving certain domains.

4.5. Coexistence of various RDNSS configuration tools

The DHCPv4 and DHCPv6 OPTION_DNS_SERVER_SELECT options are designed to coexist between each other and with other tools used for RDNSS address configuration.
For RDNSS selection purposes information received from all tools should be combined together into a single list, as discussed in Section 4.1.

In the case of DHCPv4 and DHCPv6 providing conflicting RDNSS selection information on the same interface, or on the equally trusted interfaces, the node SHALL firstly prefer DHCP-version possibly securing OPTION_DNS_SERVER_SELECT, and secondly prefer DHCPv6 over DHCPv4.

The RDNSSes learned via other tools than OPTION_DNS_SERVER_SELECT MUST be handled as default RDNSSes, with medium preference, when building a list of RDNSSes to talk to (see Section 4.1).

The non-exhaustive list of possible other sources for RDNSS address configuration are:

(1) DHCPv6 OPTION_DNS_SERVERS defined in [RFC3646].

(2) DHCPv4 Domain Name Server Option defined in [RFC2132].

(3) IPv6 Router Advertisement RDNSS Option defined in [RFC6106].

When the OPTION_DNS_SERVER_SELECT contains default RDNSS address and other sources are providing RNDSS addresses, the resolver MUST make the decision which one to prefer based on RDNSS preference field value. If OPTION_DNS_SERVER_SELECT defines medium preference then RDNSS from OPTION_DNS_SERVER_SELECT SHALL be selected.

If multiple sources are providing same RDNSS(es) IP address(es), each address MUST be added to the RDNSS list only once.

If a node had indicated support for OPTION_DNS_SERVER_SELECT in DHCPv6 request, the DHCPv6 server may choose to omit sending of OPTION_DNS_SERVERS. This enables offloading use case where network administrator wishes to only advertise low priority default RDNSSes.

4.6. Considerations on follow-up queries

Any follow-up queries that are performed on the basis of an answer received on an interface MUST continue to use the same interface, irrespective of the RDNSS selection settings on any other interface. For example, if a node receives a reply with a canonical name (CNAME) or delegation name (DNAME) the follow-up queries MUST be sent to RDNSS(es) of the same interface, or to same RDNSS, irrespectively of the FQDN received. Otherwise referrals may fail.
5. Example of a node behavior

Figure 7 illustrates node behavior when it initializes two network interfaces for parallel usage and learns domain and network information from DHCPv6 servers.

Flow explanations:

1. A node opens its first network interface

2. The node obtains domain ‘domain1.example.com’ and IPv6 network ‘0.8.b.d.0.1.0.0.2.ip6.arpa’ for the new interface 1 from DHCPv6 server
3. The node stores the learned domains and IPv6 networks for later use

4. The node opens its second network interface 2

5. The node obtains domain ‘domain2.example.com’ and IPv6 network information, say ‘1.8.b.d.0.1.0.0.2.ip6.arpa’ for the new interface 2 from DHCPv6 server

6. The node stores the learned domains and networks for later use

Figure 8 below illustrates how a resolver uses the learned domain information. Network information use for reverse lookups is not illustrated, but that would go as the figure 7 example.

<table>
<thead>
<tr>
<th>Application</th>
<th>Node</th>
<th>RDNSS on interface 1</th>
<th>RDNSS on interface 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2)</td>
<td></td>
<td>RDNSS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>prioritization</td>
<td></td>
</tr>
<tr>
<td>(3)</td>
<td></td>
<td>-------------------DNS resolution-----&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;--------------------</td>
<td></td>
</tr>
<tr>
<td>(4)</td>
<td>&lt;---Name resp-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example on choosing interface based on domain

Flow explanations:

1. An application makes a request for resolving an FQDN, e.g. ‘private.domain2.example.com’

2. A node creates list of RDNSSes to contact to and uses configured RDNSS selection information and stored domain information on prioritization decisions.

3. The node has chosen interface 2, as from DHCPv6 it was learned earlier that the interface 2 has domain ‘domain2.example.com’. The node then resolves the requested name using interface 2’s
4. The node replies to application with the resolved IPv6 address

6. Scalability considerations

The size limitations of DHCP messages limit the number of domains and networks that can be carried in configuration options. Including the domains and networks in a DHCP option is best suited for deployments where relatively few carefully selected domains and networks are adequate.

7. Considerations for network administrators

Network administrators deploying private namespaces should assist advanced nodes in their RDNSS selection process by providing information described within this document.

Private namespaces MUST be globally unique in order to keep DNS unambiguous and henceforth avoiding caching related issues and destination selection problems (see Section 2.3). Exceptions to this rule are domains utilized for local name resolution (such as .local).

Private namespaces MUST only consist of subdomains of domains for which the relevant operator provides authoritative name service. Thus, subdomains of example.com are permitted in the private namespace served by an operator’s RDNSSes only if the same operator provides an SOA record for example.com.

To counter against attacks against private namespaces, administrators utilizing this tool SHOULD deploy DNSSEC for their zone.

8. Acknowledgements

The author would like to thank following people for their valuable feedback and improvement ideas: Mark Andrews, Jari Arkko, Marcelo Bagnulo, Brian Carpenter, Stuart Cheshire, Lars Eggert, Tomohiro Fujisaki, Peter Koch, Suresh Krishnan, Murray Kucherawy, Edward Lewis, Kurtis Lindqvist, Arifumi Matsumoto, Erik Nordmark, Steve Padgett, Fabien Rapin, Matthew Ryan, Dave Thaler, Margaret Wasserman, Dan Wing, and Dec Wojciech. Ted Lemon and Julien Laganier receive special thanks for their contributions to security considerations.

This document was prepared using xml2rfc template and the related web-tool.
9. IANA Considerations

This memo requests IANA to assign two new option codes. First option code is requested to be assigned for DHCPv4 RDNSS Selection option (TBD) from the DHCP option code space defined in section "New DHCP option codes" of RFC 2939. Second option code is requested to be assigned to the DHCPv6 RDNSS Selection option (TBD) from the DHCPv6 option code space defined in section "IANA Considerations" of RFC 3315.

10. Security Considerations

10.1. Attack vectors

It is possible that attackers might try to utilize OPTION_DNS_SERVER_SELECT option to redirect some or all DNS queries sent by a resolver to undesired destinations. The purpose of an attack might be denial-of-service, preparation for man-in-the-middle attack, or something akin.

Attackers might try to lure specific traffic by advertising domains and networks from very small to very large scope or simply by trying to place attacker’s RDNSS as the highest priority default RDNSS.

The best countermeasure for nodes is to implement validating DNSSEC aware resolvers. Trusting on validation done by a RDNSS is a possibility only if a node trusts the RDNSS and can use a secure channel for DNS messages.

10.2. Trust levels of network interfaces

Decision on trust levels of network interfaces depends very much on deployment scenario and types of network interfaces. For example, unmanaged WLAN may be considered less trustworthy than managed cellular or VPN connections. An implementation may not be able to determine trust levels without explicit configuration provided by user or administrator. Therefore, for example, an implementation may not by default trust configuration received even over VPN interfaces.

The decision on levels of trust may be made by implementation, by node administrators, or for example by other standards defining organizations as part of system design work.

10.3. Importance of following the algorithm

The Section 4 uses normative language for describing node internal behavior in order to ensure nodes would not open up new attack
vectors by accidental use of RDNSS selection options. During the standards work consensus was that it is safer to not enable this option always by default, but only when deemed useful and safe.

11. References

11.1. Normative References


11.2. Informative References


[RFC3646] Droms, R., "DNS Configuration options for Dynamic Host
On some private namespace deployments explicit policies for RDNSS selection are not available. This section describes ways for nodes to mitigate the problem by sending wide-spread queries and by utilizing possibly existing indirect information elements as hints.

A.1. Sending queries out on multiple interfaces in parallel

A possible current practice is to send DNS queries out of multiple interfaces and pick up the best out of the received responses. A node SHOULD implement DNSSEC in order to be able to reject responses that cannot be validated. Selection between legitimate answers is implementation specific, but replies from trusted RDNSS should be preferred.

A downside of this approach is increased consumption of resources. Namely power consumption if an interface, e.g. wireless, has to be brought up just for the DNS query that could have been resolved also via cheaper interface. Also load on RDNSSes is increased. However, local caching of results mitigates these problems, and a node might also learn interfaces that seem to be able to provide ‘better’ responses than other and prefer those - without forgetting fallback required for cases when node is connected to more than one network using private namespaces.
A.2. Search list option for DNS forward lookup decisions

A node can learn the special domains of attached network interfaces from IPv6 Router Advertisement DNS Search List Option [RFC6106] or DHCP search list options; DHCPv4 Domain Search Option number 119 [RFC3397] and DHCPv6 Domain Search List Option number 24 [RFC3646]. The node behavior is very similar as is illustrated in the example at Section 5. While these options are not intended to be used in RDNSS selection, they may be used by the nodes as hints for smarter RDNSS prioritization purposes in order to increase likelihood of fast and successful DNS query.

Overloading of existing DNS search list options is not without problems: resolvers would obviously use the domains learned from search lists also for name resolution purposes. This may not be a problem in deployments where DNS search list options contain few domains like ‘example.com, private.example.com’, but can become a problem if many domains are configured.

A.3. More specific routes for reverse lookup decision

[RFC4191] defines how more specific routes can be provisioned for nodes. This information is not intended to be used in RDNSS selection, but nevertheless a node can use this information as a hint about which interface would be best to try first for reverse lookup procedures. A RDNSS configured via the same interface as more specific routes is more likely capable to answer reverse lookup questions correctly than RDNSS of an another interface. The likelihood of success is possibly higher if RDNSS address is received in the same RA [RFC6106] as the more specific route information.

A.4. Longest matching prefix for reverse lookup decision

A node may utilize the longest matching prefix approach when deciding which RDNSS to contact for reverse lookup purposes. Namely, the node may send a DNS query to a RDNSS learned over an interface having longest matching prefix to the address being queried. This approach can help in cases where ULA [RFC4193] addresses are used and when the queried address belongs to a node or server within the same network (for example intranet).

Appendix B. DNSSEC and multiple answers validating with different trust anchors

When validating DNS answers with DNSSEC, a validator might order the list of trust anchors it uses to start validation chains, in terms of the node’s preferences for those trust anchors. A node could use
this ability in order to select among alternative DNS results from different interfaces. Suppose that a node has a trust anchor for the public DNS root, and also has a special-purpose trust anchor for example.com. An answer is received on interface i1 for www.example.com, and the validation for that succeeds by using the public trust anchor. Also, an answer is received on interface i2 for www.example.com, and the validation for that succeeds by using the trust anchor for example.com. In this case, the node has evidence for relying on i2 for answers in the example.com zone.

Authors’ Addresses

Teemu Savolainen  
Nokia  
Hermiankatu 12 D  
TAMPERE, FI-33720  
FINLAND  

Email: teemu.savolainen@nokia.com

Jun-ya Kato  
NTT  
9-11, Midori-Cho 3-Chome Musashino-Shi  
TOKYO, 180-8585  
JAPAN  

Email: kato@syce.net

Ted Lemon  
Nominum, Inc.  
2000 Seaport Boulevard  
Redwood City, CA 94063  
USA  

Phone: +1 650 381 6000  
Email: Ted.Lemon@nominum.com