

Internet Engineering Task Force  
Internet-Draft  
Intended status: Standards Track  
Expires: July 11, 2011

T. Savolainen  
Nokia  
J. Kato  
NTT  
January 7, 2011

**Improved DNS Server Selection for Multi-Homed Nodes**  
**draft-savolainen-mif-dns-server-selection-06**

Abstract

A multi-homed node can be connected to multiple networks that may utilize different DNS namespaces. The node often receives DNS server configuration information from all connected networks. Some of the DNS servers may have information about namespaces other servers do not have. When the multi-homed node needs to utilize DNS, it has to choose which of the servers to contact to. This document describes a policy based method for helping on selection of DNS server for both forward and reverse DNS lookup procedures with help of DNS suffix and IPv6 prefix information received via DHCPv6.

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 July 11, 2011.

Copyright Notice

Copyright (c) 2011 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

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">1.1.</a>	Requirements Language . . . . .	<a href="#">3</a>
2.	Problem description for local namespaces with multi-homed nodes . . . . .	<a href="#">4</a>
<a href="#">2.1.</a>	Fully qualified domain names with limited scopes . . . . .	<a href="#">4</a>
<a href="#">2.2.</a>	Network interface specific IP addresses . . . . .	<a href="#">5</a>
<a href="#">3.</a>	Improved DNS server selection . . . . .	<a href="#">7</a>
<a href="#">3.1.</a>	Procedure for prioritizing DNS servers and handling responses . . . . .	<a href="#">7</a>
<a href="#">3.2.</a>	DNS server selection option . . . . .	<a href="#">9</a>
<a href="#">3.3.</a>	Coexistence with <a href="#">RFC3646</a> . . . . .	<a href="#">11</a>
<a href="#">4.</a>	Example of a node behavior . . . . .	<a href="#">12</a>
<a href="#">5.</a>	Scalability considerations . . . . .	<a href="#">15</a>
<a href="#">6.</a>	Considerations for network administrators . . . . .	<a href="#">15</a>
<a href="#">7.</a>	Acknowledgements . . . . .	<a href="#">15</a>
<a href="#">8.</a>	IANA Considerations . . . . .	<a href="#">15</a>
<a href="#">9.</a>	Security Considerations . . . . .	<a href="#">16</a>
<a href="#">10.</a>	References . . . . .	<a href="#">16</a>
<a href="#">10.1.</a>	Normative References . . . . .	<a href="#">16</a>
<a href="#">10.2.</a>	Informative References . . . . .	<a href="#">16</a>
<a href="#">Appendix A.</a>	Best Current Practice for DNS server selection . . . . .	<a href="#">17</a>
<a href="#">A.1.</a>	Sending queries out on multiple interfaces in parallel . . . . .	<a href="#">18</a>
<a href="#">A.2.</a>	Search list option for DNS forward lookup decisions . . . . .	<a href="#">18</a>
<a href="#">A.3.</a>	More specific routes for reverse lookup decision . . . . .	<a href="#">19</a>
<a href="#">A.4.</a>	Longest matching prefix for reverse lookup decision . . . . .	<a href="#">19</a>
	Authors' Addresses . . . . .	<a href="#">19</a>



## **1. Introduction**

A multi-homed node faces several problems over single-homed node as is described in [[I-D.ietf-mif-problem-statement](#)]. This document studies in detail the problems local namespaces may cause for multi-homed nodes in the IPv4 and IPv6 domains and provides a solution. The node may be implemented as a host, or as a router such as Consumer Premises Equipment.

When multiple namespaces are visible for a node, some DNS servers have information other servers do not have. Because of that, a multi-homed node cannot assume every DNS server is able to provide any piece of information, but instead the node must be able to ask right server for the information it needs.

An example of an application that benefits from multi-homing is a web browser that commonly accesses many different destinations and should be able to dynamically communicate over different network interfaces.

However, as the IPv4 is being phased out and often uses NATs to achieve similar functions, this document describes a solution only for the IPv6 domain.

In deployments where multiple namespaces are present, selection of the correct destination and source addresses for the actual IP connection is usually crucial as well, as the resolved destination's IP address may be only usable on the network interface over which it was resolved on. Hence solution described in this document is assumed to be often used in combination of tools delivering source and destination address selection policies.

Node multihoming in general may introduce new attack vectors. This document includes security considerations that will help against possible new attack vectors and also to some existing attack vectors.

The [Appendix A](#) describes best current practices possible with tools preceding this document and on networks not supporting this specification. As it is possible to solve the problem with less efficient and less explicit manners, this document can be considered as an optimization. However, in some environments this optimization is considered essential.

### **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](#)].



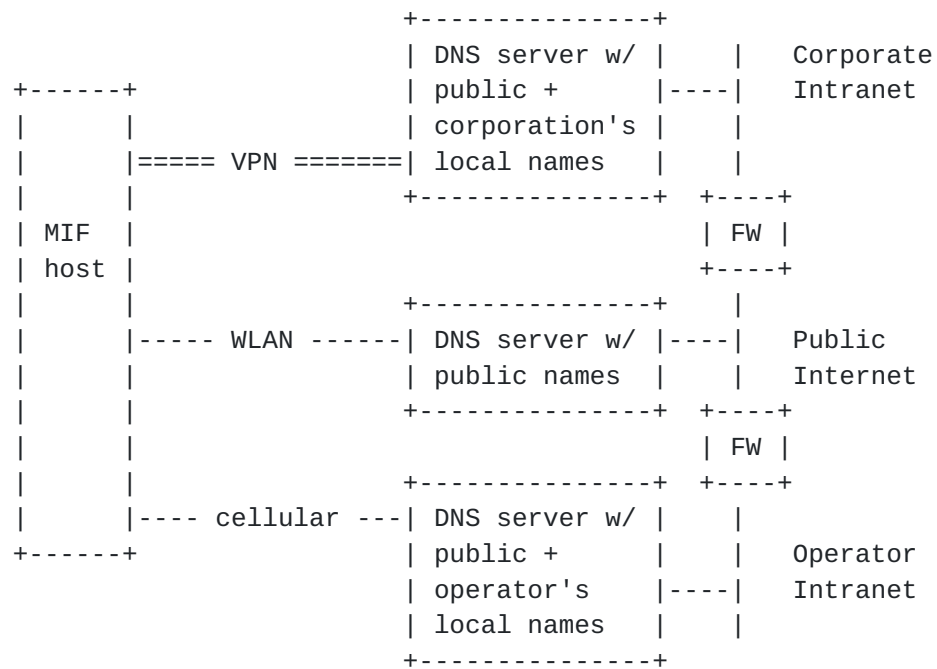
## **2. Problem description for local namespaces with multi-homed nodes**

This chapter describes two host multi-homing related local namespace scenarios for which the procedure described in chapter 3 provides a solution. Essentially the same challenges may be faced by Consumer Premises Equipment as is described in [\[I-D.troan-multi-homing-without-nat66\]](#).

### **2.1. Fully qualified domain names with limited scopes**

A multi-homed host may be connecting to one or more networks that are using local namespaces. As an example, the host may have simultaneously open a wireless LAN (WLAN) connection to public Internet, cellular connection to an operator network, and a virtual private network (VPN) connection to a corporate network. When an application initiates a connection establishment to an FQDN, the host needs to be able to choose the right network interface for making a successful DNS query. This is illustrated in the figure 1. An FQDN for a public name can be resolved with any DNS server of any network interface, but for an FQDN of corporation's or operator's service's local name the host would need to be able to correctly select the right network interface for the DNS resolution, i.e. do interface selection already before destination's IP address is known.





Split DNS and locally scoped names illustrated

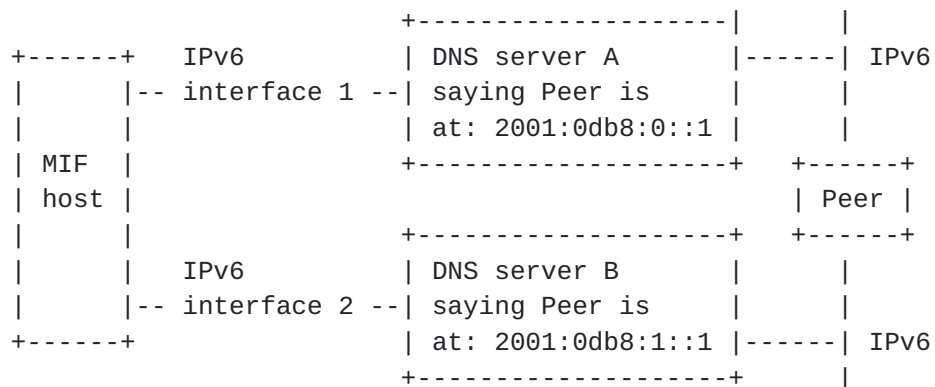
Figure 1

## 2.2. Network interface specific IP addresses

In the second problem an FQDN as such 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. This is a problem when a host is single-homed, but for multi-homed host this results in additional challenges: the host's 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.





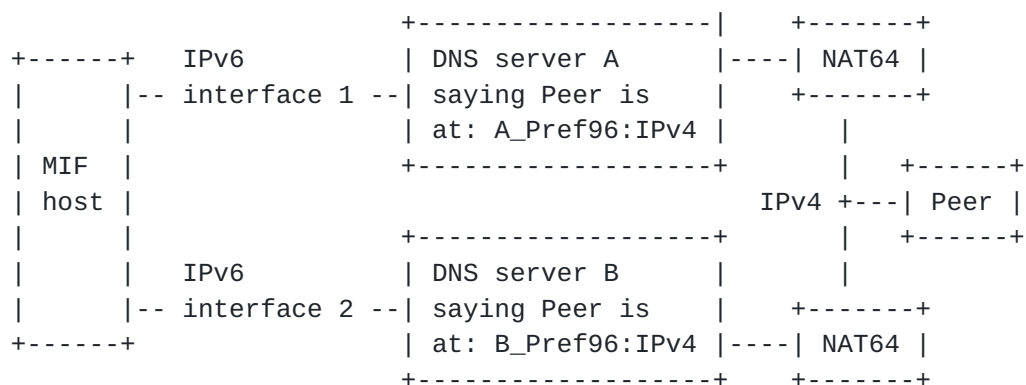


Split DSN and different IP addresses for an FQDN on interfaces 1 and 2.

Figure 2

Similar situation can happen when IPv6 protocol translation is used in combination with AAAA record synthesis procedure [I-D.ietf-behave-dns64]. A synthesised AAAA record is guaranteed to be valid only on a network interface it was synthesized on. Figure 3 illustrates a scenario where the peer's IPv4 address is synthesized into different IPv6 addresses by DNS servers A and B. The same problem can happen in the IPv4 domain as well if A record synthesis is done, for example as described in Bump-In-the-Stack [RFC2767].

For a related problem for dual-stack hosts in a network with DNS64, where IPv4 should be prioritized over synthesized IPv6, please see [I-D.wing-behave-dns64-config].



AAAA synthesis results in interface specific IPv6 addresses.

Figure 3



A more complex scenario is an FQDN, which in addition to resolving into network interface specific IP addresses, identifies on different network interfaces completely different peer entities with potentially different set of service offering. 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, some of the problems may be solvable only by user intervention.

A thing worth noting is that interface specific IP addresses can cause problems also for a single-homed host, if the host retains its DNS cache during movement from one network interface to another. After the interface change a host could have both positive and negative DNS cache entries invalid for the new network interface. Because of this the cached DNS information should be considered network interface local instead of node global.

### **3. Improved DNS server selection**

This chapter describes a procedure a (stub / proxy) resolver may utilize for improved DNS server selection in face of multiple namespaces and multiple simultaneously active network interfaces.

#### **3.1. Procedure for prioritizing DNS servers and handling responses**

The resolver SHALL dynamically build for each DNS query a priority list of DNS servers it will contact to. To prioritize DNS servers in safe and optimal way, a node SHOULD ask with DHCPv6 which DNS servers of each network interface are most likely able to successfully serve forward lookup requests matching to specific DNS suffixes or reverse (PTR record) lookup requests matching to specific IPv6 prefixes.

A resolver lacking more explicit information shall assume that all information is available from any DNS server of any network interface. The DNS servers learnt by other DNS server address configuration methods MUST be handled as medium priority default servers.

When a resource record is to be resolved, the resolver SHOULD give highest precedence to the DNS servers explicitly known to serve matching suffixes or prefixes. However, the resolver MUST take into account different trust levels of pieces of DNS server selection information the resolver has received from node's network interfaces. The resolver MUST generally prefer more trusted DNS servers and less trusted DNS server MAY be of highest priority only if trusted interfaces specifically configure DNS servers with low priority. The



non-exhaustive list on table 1 illustrates how the different trust levels of received DNS server selection information MUST influence the DNS server selection logic.

Information from from more trusted interface A	Information from less trusted interface B	Resulting DNS server priority selection
<b>1. Medium priority default</b>	Medium priority default	Default: A, then B
<b>2. Medium priority default</b>	High priority default High priority specific	Default: A, then B Specific: A, then B
<b>3. Low priority default</b>	Medium priority default	Default: B, then A
<b>4. Low priority default High priority specific</b>	Medium priority default	Default: B, then A Specific: A, then B

Figure 4: DNS server selection in case of different trust levels

The resolver SHOULD avoid sending queries to different interfaces in parallel as that may waste resources, sometimes significantly, and would also unnecessary reveal information about ongoing communications. Independently of whether DNS queries are sent in series or parallel, replies for DNS queries MUST be waited until acceptable positive reply is received, all replies are received, or time out occurs. Specifically, the resolver MUST NOT proceed with a positive reply from less trusted server that cannot be validated with DNSSEC if the DNS query sent to more trusted server is still pending. (DISCUSS: What about those DNS servers that instead of negative answer always return positive reply with an IP address of some captive portal?)

For the scenario where an FQDN maps to same service but different IP addresses on different network interfaces, the source address selection algorithm must be able to pick a source address from the network interface that was used for DNS resolution.

When local namespace are present a negative reply from a DNS server implies only that the particular DNS server was not able to serve the query. However, it is not probable that the secondary DNS servers on the same network interface, on a same administrative domain, would be able to serve either. Therefore, the next DNS server resolver contacts SHOULD be from another network interface.



Resolver SHOULD use DNSSEC to validate all received DNS replies. In the case DNSSEC validation fails the resolver MUST resend the query to the next preferred DNS server.

### **3.2. DNS server selection option**

A DHCPv6 option described below is used to inform resolvers which DNS server should be contacted when initiating forward or reverse DNS lookup procedures.



```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|  OPTION_DNS_SERVER_SELECT      |      option-len      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                |                          |
|      DNS-recursive-name-server (IPv6 address)           |
|                                |                          |
|                                |                          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|prf| Reserved |                                          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                |      DNS suffixes and prefixes      |
|                                |      (variable length)                |
|                                |                                          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

option-code:   OPTION\_DNS\_SERVER\_SELECT (TBD)

option-len:    Length of the option in octets

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

prf:           DNS server preference:

```

      01 High
      00 Medium
      11 Low
      10 Reserved - MUST NOT be sent

```

The preference is used when selecting between  
equally trusted DNS servers.

(Editor's note: this field is under consideration  
- really needed or not?)

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

DNS suffixes and prefixes: The list of DNS suffixes for forward DNS  
lookup and prefixes for reverse DNS lookup the DNS server  
has special knowledge about. Field MUST be encoded as  
specified in section "Representation and use of  
domain names" of [[RFC3315](#)].  
Additionally, special suffix of "." is used to indicate  
capability to resolve global names. Lack of "."  
suffix on the list indicates DNS server has only local  
information. Prefixes for reverse mapping are encoded as  
defined for ip6.arpa [[RFC3152](#)].



## DHCPv6 option for explicit DNS suffix configuration

Figure 5

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

The `OPTION_DNS_SERVER_SELECT` contains one or more DNS suffixes the related DNS server has particular knowledge of (i.e. local namespaces). The option can occur multiple times in a single DHCPv6 message, if multiple DNS servers are to be configured.

A resolver SHOULD prioritize between equally trusted DNS servers with help of the preference field. The resolver MUST NOT prioritize less trusted DNS servers higher than trusted, even in the case of less trusted server would apparently have additional information. In case all other things being equal the resolver shall make the prioritization decision based on its internal preferences.

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

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

The node SHOULD create a host specific route for the DNS server address. The route must point to the interface DNS server address was learned on. This is required to ensure DNS queries are sent out via the right interface.

In the case of a DNS server replying negatively to a question having matching suffix, it will be for implementation to decide whether to consider that as a final response, or whether to ask also from other DNS servers. The implementation decision may be based, for example, on deployment or trust models.

### 3.3. Coexistence with [RFC3646](#)

The `OPTION_DNS_SERVER_SELECT` is designed to coexist with `OPTION_DNS_SERVERS` defined in [[RFC3646](#)]. The DNS servers configured



via `OPTION_DNS_SERVERS` MUST BE considered as default name servers with medium preference. When both options are received from the same network interface and the `OPTION_DNS_SERVER_SELECT` contains default DNS server address, the resolver SHOULD make decision which one to prefer based on preferences. If `OPTION_DNS_SERVER_SELECT` defines medium preference then DNS server selection decision is implementation specific. All default servers are assumed to be able to resolve queries for global names.

If both `OPTION_DNS_SERVERS` and `OPTION_DNS_SERVER_SELECT` contain the same DNS server(s) IPv6 address(es), only one instance of each DNS servers' IPv6 addresses shall be added to the DNS server list.

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 DNS servers.

#### **4. Example of a node behavior**

Figure 5 illustrates node behavior when it initializes two network interfaces for parallel usage and learns DNS suffix and prefix information from DHCPv6 servers.



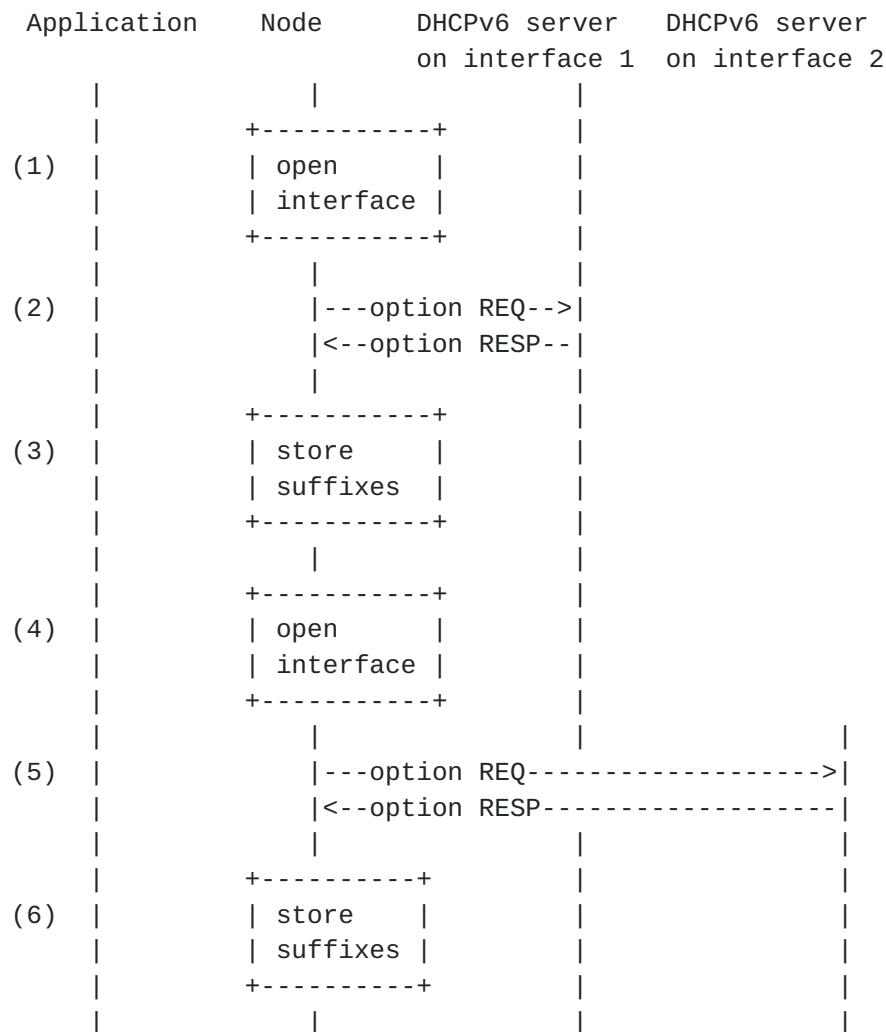


Illustration of learning DNS suffixes

Figure 6

Flow explanations:

1. A node opens its first network interface
2. The node obtains DNS suffix and IPv6 prefix information for the new interface 1 from DHCPv6 server
3. The node stores the learned DNS suffixes and IPv6 prefixes for later use
4. The node opens its seconds network interface 2
5. The node obtains DNS suffix, say 'example.com', and IPv6 prefix information, say '8.b.d.0.1.0.0.2.ip6.arpa' for the new interface

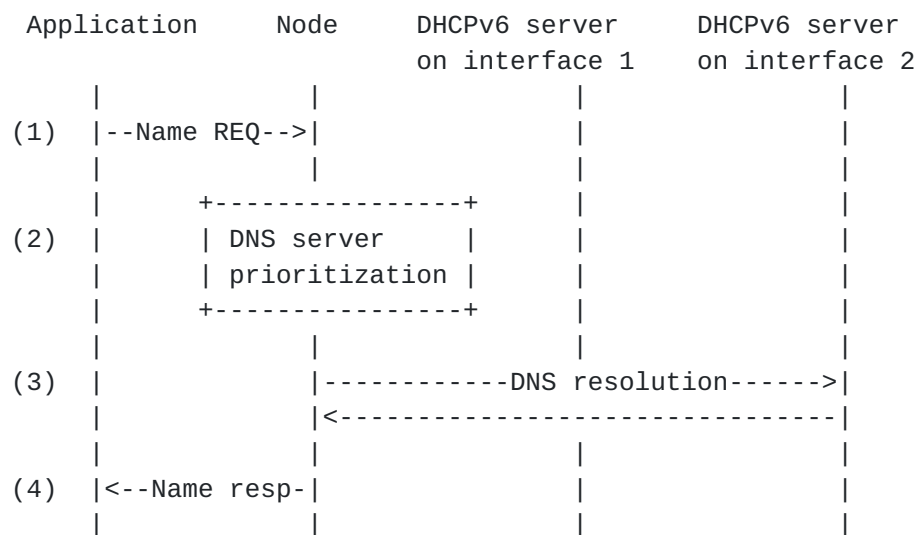




2 from DHCPv6 server

6. The node stores the learned DNS suffixes and prefixes for later use

Figure 6 below illustrates how a resolver uses the learned suffix information. Prefix information use for reverse lookups is not illustrated, but that would go as the figure 6 example.



Example on choosing interface based on DNS suffix

Figure 7

Flow explanations:

1. An application makes a request for resolving an FQDN, e.g. 'private.example.com'
2. A node creates list of DNS servers to contact to and uses configured DNS server information and stored DNS suffix information on prioritization decisions.
3. The node has chosen interface 2, as from DHCPv6 it was learned earlier that the interface 2 has DNS suffix 'example.com'. The node then resolves the requested name using interface 2's DNS server to an IPv6 address
4. The node replies to application with the resolved IPv6 address



## **5. Scalability considerations**

The size limitations of DHCPv6 messages limit the number of suffixes and prefixes that can be carried in a configuration option. Including the suffixes and prefixes in a DHCPv6 option is best suited for deployments where relatively few carefully selected suffixes and prefixes are adequate.

## **6. Considerations for network administrators**

Network administrators deploying private namespaces should assist advanced hosts in the DNS server selection by providing information described in this memo for nodes. To ensure nodes' source and destination IP address selection also works correctly, network administrators should also deploy related technologies for that purpose.

The solution described herein is best for selecting a DNS server having knowledge of some namespaces. The solution is not able to make the right decision in a scenario where same name points to different services on different network interfaces. Network administrators are recommended to avoid overloading of namespaces in such manner.

To mitigate against attacks against local namespaces, administrators utilizing this tool should deploy DNSSEC for their zone.

## **7. Acknowledgements**

The author would like to thank following people for their valuable feedback and improvement ideas: Mark Andrews, Jari Arkko, Marcelo Bagnulo, Stuart Cheshire, Lars Eggert, Tomohiro Fujisaki, Peter Koch, Suresh Krishnan, Edward Lewis, Kurtis Lindqvist, Arifumi Matsumoto, Erik Nordmark, Steve Padgett, Fabien Rapin, 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.

## **8. IANA Considerations**

This memo includes a new DHCPv6 option that requires allocation of a new code point.



## **9. Security Considerations**

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 DNS suffixes and prefixes from very small to very large scope or simply by trying to place attacker's DNS server as the highest priority default server.

The main countermeasure against these attacks is to systematically prioritize more trusted DNS servers higher than less trusted ones. Additionally, resolvers should implement DNSSEC to be able to validate DNS responses received via any of its 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.

A node MAY also choose, or be configured, to obtain DNS server selection rules only from selected trusted interfaces, in which case it would be in the hands of administrators of these trusted interfaces whether or not to allow redirection, offloading, of DNS queries to untrusted interfaces (case 4 of table 1).

## **10. References**

### **10.1. Normative References**

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC3152] Bush, R., "Delegation of IP6.ARPA", [BCP 49](#), [RFC 3152](#), August 2001.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", [RFC 3315](#), July 2003.

### **10.2. Informative References**

- [I-D.ietf-behave-dns64]  
Bagulo, M., Sullivan, A., Matthews, P., and I. Beijnum,



"DNS64: DNS extensions for Network Address Translation from IPv6 Clients to IPv4 Servers",  
[draft-ietf-behave-dns64-11](#) (work in progress),  
October 2010.

[I-D.ietf-mif-problem-statement]

Blanchet, M. and P. Seite, "Multiple Interfaces and Provisioning Domains Problem Statement",  
[draft-ietf-mif-problem-statement-09](#) (work in progress),  
October 2010.

[I-D.troan-multihoming-without-nat66]

Troan, O., Miles, D., Matsushima, S., Okimoto, T., and D. Wing, "IPv6 Multihoming without Network Address Translation", [draft-troan-multihoming-without-nat66-01](#) (work in progress), July 2010.

[I-D.wing-behave-dns64-config]

Wing, D., "DNS64 Resolvers and Dual-Stack Hosts",  
[draft-wing-behave-dns64-config-02](#) (work in progress),  
February 2010.

[RFC2767] Tsuchiya, K., HIGUCHI, H., and Y. Atarashi, "Dual Stack Hosts using the "Bump-In-the-Stack" Technique (BIS)",  
[RFC 2767](#), February 2000.

[RFC3397] Aboba, B. and S. Cheshire, "Dynamic Host Configuration Protocol (DHCP) Domain Search Option", [RFC 3397](#),  
November 2002.

[RFC3646] Droms, R., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", [RFC 3646](#),  
December 2003.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", [RFC 4191](#), November 2005.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", [RFC 4193](#), October 2005.

[RFC5006] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Option for DNS Configuration",  
[RFC 5006](#), September 2007.

## [Appendix A](#). Best Current Practice for DNS server selection

On some split-DNS deployments explicit policies for DNS server





selection are not available. This section describes ways for hosts 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 host SHOULD implement DNSSEC in order to be able to reject responses that cannot be validated. Selection between legitimate answers is implementation specific, but positive replies 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 DNS servers is increased. However, local caching of results mitigates these problems, and a node might also learn interfaces that seem to be able to provide more responses than other and prefer those - without forgetting fallback required for cases when node is connected to more than one network using local namespaces.

Another downside is revealing to all DNS servers the names a host is connecting to. For example, a DNS server of public hotspot could learn all the private names host is trying to connect on other interfaces.

#### **A.2. Search list option for DNS forward lookup decisions**

A host can learn the special DNS suffixes of attached network interfaces from DHCP search list options; DHCPv4 Domain Search Option number 119 [[RFC3397](#)] and DHCPv6 Domain Search List Option number 24 [[RFC3646](#)]. The host behavior is very similar as is illustrated in the example at [section 3.3](#). While these DHCP options are not intended to be used in DNS server selection, they may be used by the host for smart DNS server 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 DNS suffixes learned from search lists also for name resolution purposes. This may not be a problem in deployments where DNS search list options contain few DNS suffixes like 'example.com, private.example.com', but can become a problem if many suffixes are configured.



### **[A.3.](#) More specific routes for reverse lookup decision**

[RFC4191] defines how more specific routes can be provisioned for hosts. This information is not intended to be used in DNS server selection, but nevertheless a host can use this information as a hint about which interface would be best to try first for reverse lookup procedures. A DNS server configured via the same interface as more specific routes is likely more capable to answer reverse lookup questions than DNS server of an another interface. The likelihood of success is possibly higher if DNS server address is received in the same RA [[RFC5006](#)] as the more specific route information.

### **[A.4.](#) Longest matching prefix for reverse lookup decision**

A host may utilize the longest matching prefix approach when deciding which DNS server to contact for reverse lookup purposes. Namely, the host may send a DNS query to a DNS server 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 host or server within the same network (for example intranet).

#### Authors' Addresses

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

Email: [teemu.savolainen@nokia.com](mailto:teemu.savolainen@nokia.com)

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

Email: [kato@syce.net](mailto:kato@syce.net)

