

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: August 23, 2009

T. Savolainen
Nokia
February 19, 2009

DNS Server Selection on Multi-Homed Hosts
draft-savolainen-mif-dns-server-selection-00

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

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 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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/1id-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on August 23, 2009.

Copyright Notice

Copyright (c) 2009 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.

Abstract

A multi-homed host may receive DNS server configuration information from multiple physical and/or virtual network interfaces. In split

Internet-Draft

MIF and DNS server selection

February 2009

DNS scenarios not all DNS servers are able to provide the same information. When the multi-homed host needs to utilize DNS, it has to select which of the servers to contact to. This document describes problems of split DNS for multi-homed hosts and also a method for selecting the DNS server with help of DNS suffix information received dynamically for each network interface. The method is useful in split DNS scenarios where private names are used and where correct DNS server selection is mandatory for successful DNS resolution.

Table of Contents

1.	Introduction	3
1.1.	Requirements Language	4
2.	Problem description for split DNS with multi-homed hosts . . .	4
2.1.	Private fully qualified domain names	4
2.2.	Network interface specific IP addresses	5
3.	DNS server selection procedure	6
3.1.	DNS suffixes as hints	7
3.1.1.	Learning of the DNS suffixes	8
3.1.2.	Changes to DNS resolution procedures	9
4.	Considerations for network administrators	10
5.	Further considerations	10
6.	Acknowledgements	10
7.	IANA Considerations	11
8.	Security Considerations	11
9.	Normative References	11
	Author's Address	12

1. Introduction

A multi-homed host faces several problems over single-homed host as described in [[I-D.blanchet-mif-problem-statement](#)]. This document studies problems split DNS may cause for multi-homed hosts and for which optimal behaviour should be defined. The problems are the same in IPv4 and IPv6 domains.

In the split DNS scenario different DNS servers have different information. Therefore DNS related information, which otherwise could be consider global for a single-homed host, in a multi-homed host has to be handled as local to a network interface.

An obvious solution for the problem would be for network administrators to cease utilizing any form of split DNS, or have split DNS used only in deployments where hosts are not allowed to multi-home. However, currently split DNS is deployed and multi-homed hosts have to cope with that.

If an application is bound to utilize only a specific network interface at a time, it essentially makes the host behave single-interface way for that particular application and avoids the problems of split DNS (assuming that also application's DNS requests are handled strictly with DNS service available in that particular network interface). If all applications in a host are bound to use only single network interface at a time, even if the used network interfaces were different, the problems are generally avoided. The procedure described in chapter 3 applies when applications are allowed to utilize multiple interfaces in parallel.

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

The solution presented in this memo is intended to be fully backwards

compatible and one that can be fully ignored by hosts and networks that are not experiencing the described problem scenarios.

In deployments where split DNS is used, selection of the correct destination and source addresses for the actual IP connection is crucial, as the resolved destination's IP address may be only usable on the network interface over which it was resolved on. However, the actual IP address selection logic is not at the scope of 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.](#) Problem description for split DNS with multi-homed hosts

This chapter describes two multi-homing related split DNS problem scenarios for which the procedure described in chapter 3 is targeted at. (DISCUSS: Even more more known problem scenarios caused by split DNS for multi-homed hosts?)

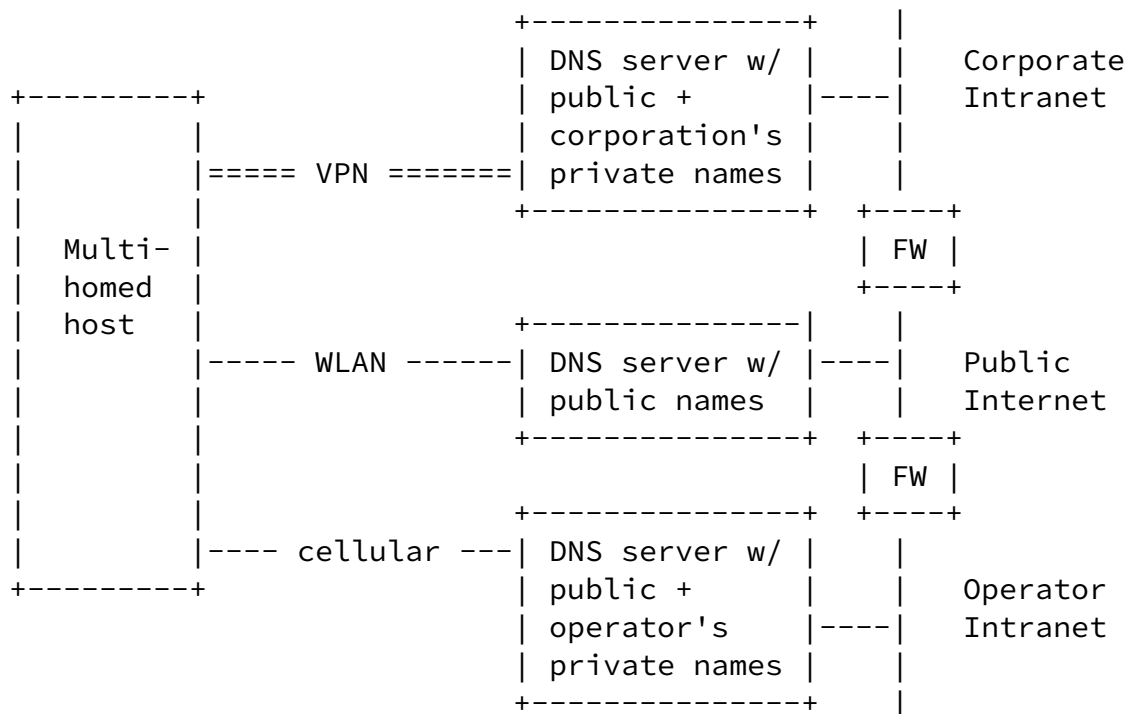
[2.1.](#) Private fully qualified domain names

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

Internet-Draft

MIF and DNS server selection

February 2009

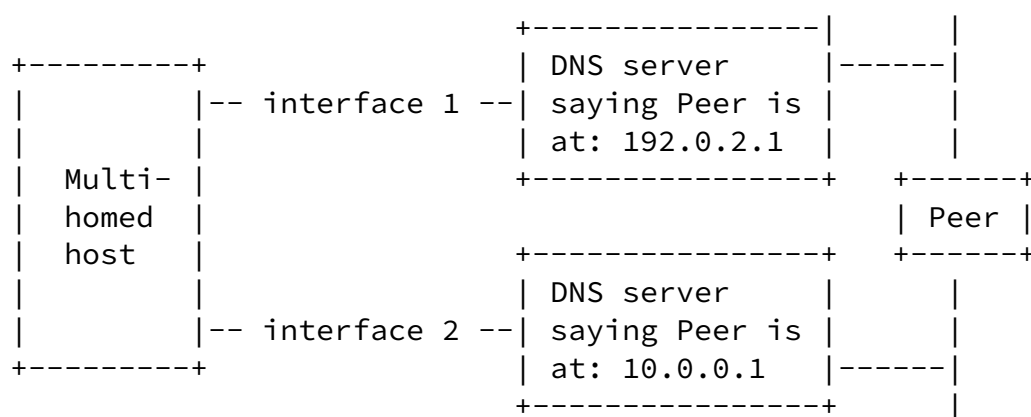


Split DNS and private names illustrated

Figure 1

2.2. Network interface specific IP addresses

In the second problem an FQDN is resolvable via different network interfaces, but to different and not necessarily globally reachable IP addresses. This is not so much 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 interface 1 and 2.

Figure 2

More complex scenario of this problem is an FQDN, which in addition to resolving into network interface specific IP addresses, also on different network interfaces identifies completely different peers with potentially different set of service offering.

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, and thus on the new network interface host has cache entirely invalid for that network interface. Because of this the DNS cache information should be considered network interface local instead of node global.

3. DNS server selection procedure

This chapter documents a possible procedure a host may utilize for DNS server selection on multi-homed scenarios.

Essentially the host shall dynamically for each DNS query build a list of DNS servers it will try to contact to and cycle through until a positive reply is received or all selected DNS servers have been contacted or timed out. (DISCUSS: What about those DNS servers that instead of negative answer always return positive reply with an IP address of some default HTTP server, which purpose is just to say 'page not found'?)

When building the list, the host shall prioritize DNS servers in a optimal way for the query at hand. Host can utilize any information it may have, e.g. possible user's preferences, host's general preferences between network interfaces, differences on trust levels

of network interfaces (see Security Considerations), DNS suffix information possibly available, or any other piece of information.

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.

In private FQDN deployments a negative reply from a DNS server implies only that the DNS server at hand is not able to serve the query. However, it is not probable that the secondary DNS servers on the same network interface would be able to serve either due likely being in the same administrative domain. Therefore the next DNS server host contacts should be from another network interface.

A host may optimize its behaviour by sending DNS requests in parallel to multiple DNS servers of different network interfaces, but this approach is not always practical:

- o It may unnecessary trigger activation of a radio and thus increase battery consumption.
- o It may unnecessarily reveal private names to outsiders.
- o It may be a privacy issues as it would reveal all records host is resolving to all DNS servers.

[3.1.](#) DNS suffixes as hints

To help prioritize DNS servers in an optimal way, a host may learn which DNS servers are most likely able to successfully serve requests related to specific DNS suffixes.

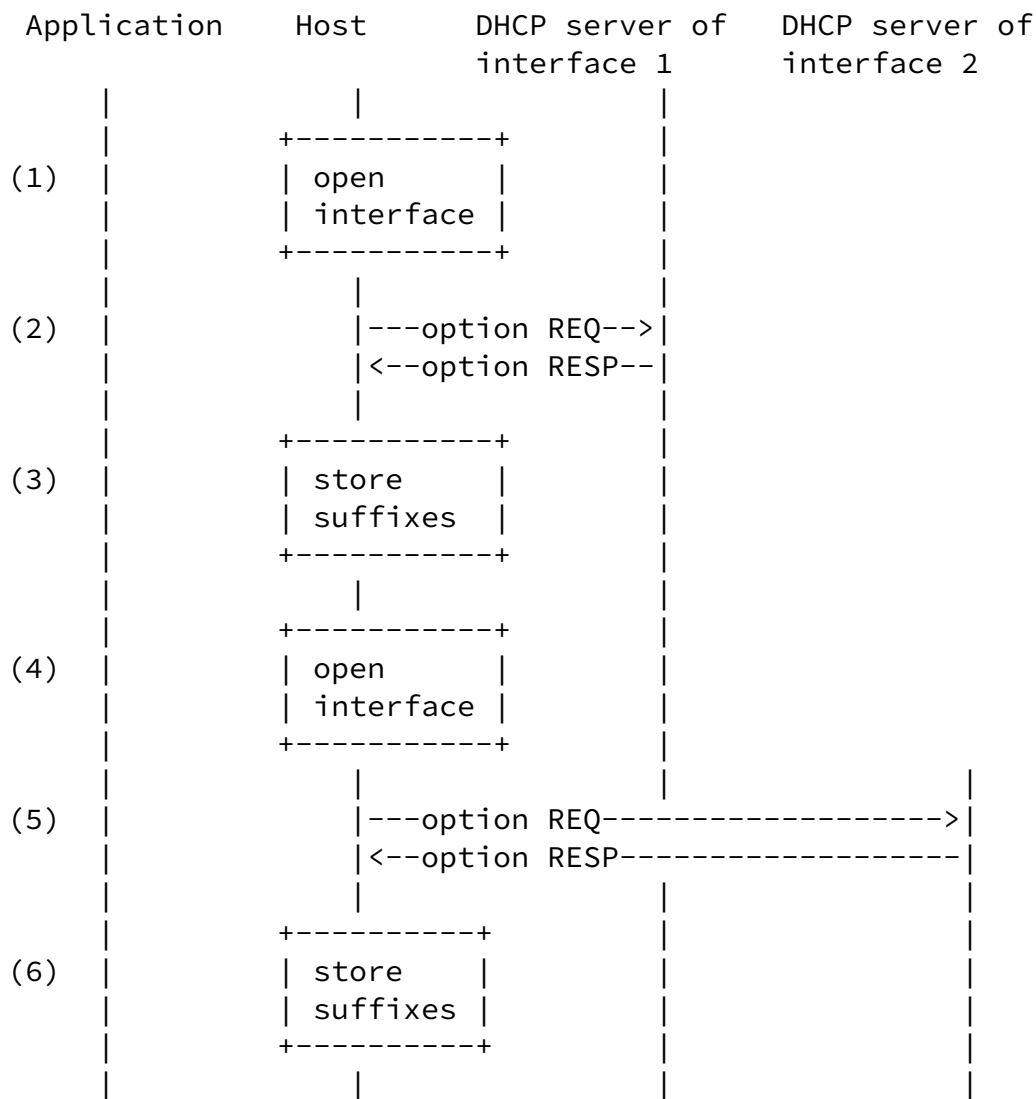
By default a host should assume all information is available via all DNS servers of any network interface.

When a resource record is to be resolved, a host shall give higher precedence to DNS servers of the network interface(s) advertising corresponding DNS suffix.

For example, when a resolution of an FQDN has been requested and the host is prioritizing DNS servers of different network interfaces, the host may prioritize higher DNS server(s) of the network interface(s) with matching DNS suffix than it otherwise would have.

[3.1.1.](#) Learning of the DNS suffixes

A host can learn the 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]. This is illustrated in example figure 3 below.



Learning DNS suffixes

Figure 3

Flow explanations:

1. A host opens its first network interface
2. The host obtains DNS suffix information for the new interface 1 from DHCP server

3. The host stores the learned DNS suffixes for later use
4. The host opens its seconds network interface 2
5. The host obtains DNS suffix, say 'example.com' information for the new interface 2 from DHCP server
6. The host stores the learned DNS suffixes for later use

[3.1.2.](#) Changes to DNS resolution procedures

When a DNS resolver in a host is requested by an application to do DNS resolution for an FQDN to an IP address, the host should look if any of the available network interfaces is known to advertise DNS suffix matching to the FQDN. If there is a matching DNS suffix, then DNS server(s) of that that particular interface should be prioritized higher, i.e. be used for name resolution procedures. This is illustrated in figure 4 below.

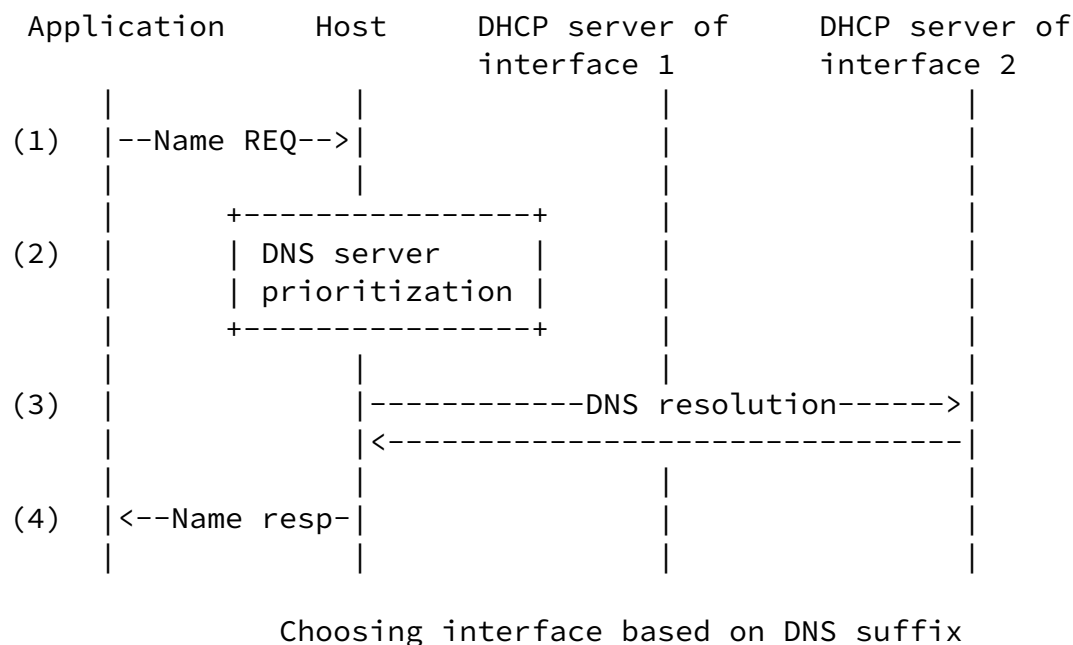


Figure 4

Flow explanations:

1. An application makes a request for resolving an FQDN, e.g. 'private.example.com'
2. A host creates list of DNS servers to contact to and uses stored

3. The host has chosen interface 2, as from DHCP it was learned earlier that the interface 2 has DNS suffix 'example.com'. The host then resolves the requested name using interface 2's DNS server to IP 192.0.2.1
4. The host replies to application with resolved IP address 192.0.2.1

[4.](#) Considerations for network administrators

Due to the problems caused by split DNS for multi-homed hosts, network administrators should consider carefully deployment of split DNS.

Network administrators deploying split DNS should assist hosts in DNS server selection by configuring their DHCP servers with proper DNS suffix information, which hosts then can use as hints. To ensure hosts' source and destination IP address selection works correctly, network administrators should also consider deployment of additional technologies to help with that.

Network administrators can continue using DHCP DNS search list options as before, but administrators should take into account that multi-homed hosts may choose to use the DNS suffix information also for DNS server selection purposes.

[5.](#) Further considerations

Overloading of existing DNS search list options is not without problems, though: hosts 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.

[6.](#) Acknowledgements

The author would like to thank following people for their comments: Jari Arkko, Marcelo Bagnulo, Lars Eggert, Kurtis Lindqvist, Fabien Rapin, Dave Thaler, Margaret Wasserman, Dec Wojciech.

This document was prepared using xml2rfc template and related web-tool.

[7.](#) IANA Considerations

This memo includes no request to IANA.

[8.](#) Security Considerations

An attacker may try to lure traffic from multi-homed host to his network by advertising DNS suffixes attacker wishes to intercept or deny service of. The host's security should not be based on correct functionality of DNS server selection, but risks of this attack can be mitigated by properly prioritizing network interfaces with conflicting DNS suffix advertisements. The prioritization could be based on trust level of a network interface over which DNS suffix was learned from, like for example:

1. Managed tunnel interfaces (such as VPN) considered most trustworthy
2. Managed networks being on the middle
3. Unmanaged networks having lowest priority

Now, for example, if all of the three abovementioned networks would advertise 'corporation.com' DNS suffix, the host would prefer the VPN network interface for related DNS resolution requests.

[9.](#) Normative References

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

Blanchet, M., "Multiple Interfaces Problem Statement",

[draft-blanchet-mif-problem-statement-00](#) (work in progress), December 2008.

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

Savolainen

Expires August 23, 2009

[Page 11]

Internet-Draft

MIF and DNS server selection

February 2009

Author's Address

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

Email: teemu.savolainen@nokia.com

