Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: May 17, 2017

G. Chen
China Mobile
C. Williams
Consultant
D. Wing
A. Yourtchenko
Cisco Systems, Inc.
November 13, 2016

Happy Eyeballs Extension for Multiple Interfaces
draft-ietf-mif-happy-eyeballs-extension-11

Abstract

This memo proposes extensions to the Happy Eyeball's algorithm requirements defined in RFC6555 for use with the multiple provisioning domain architecture. The Happy Eyeballs in MIF would make the selection process smoother by using connectivity tests over pre-filtered interfaces according to defined policy. This would choose the best interface with an automatic fallback mechanism.

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 May 17, 2017.

Copyright Notice

Table of Contents

1. Introduction

The MIF problem statement [RFC6418] describes problems specific for nodes attached to multiple provisioning domains. Specifically, there is a issue description that a node has selected an interface and obtained a valid IP address from the network, but Internet connectivity is not available. This memo intends to address the issue and elaborate more in Section 3.1.

[RFC7556] describes the multiple provisioning domain architecture. It refers to using connectivity tests to validate a Provisioning Domain (PvD). Given a number of implicit/explicit PvDs, plus preferences/policy, what is the process to follow to select the best PvD to use for any given connection. In the event that two or more are deemed to be best, how are the Happy Eyeballs (HE) techniques applied to find the best and deal with resilience. This memo also proposes process requirements using Happy Eyeballs (HE) extensions.

There are a variety of algorithms that can be envisioned. This document describes additional parameters and processes that need to be considered in addition to the HE algorithm requirements defined in [RFC6555] necessary to support multiple interfaces, so that a node with multiple interfaces can select the best path for a particular connection-oriented flow (e.g., TCP, SCTP).

2. Terminology

This document makes use of following terms:

- Happy Eyeballs (HE): specifies requirements for an algorithm that reduces the user-visible connection delay for dual-stack hosts with a single interface per-protocol.

- Happy Eyeballs - Multi-Interface (HE-MIF): Extends the Happy Eyeballs concept to the multiple provisioning domain architecture. It describes additional requirements for algorithms that offer connectivity tests on PVD-aware or non-PVD-aware nodes [RFC7556] to select the best interface for a specific connection request.

3. Use Cases

The section describes scenarios the HE-MIF targeted to use.

3.1. WiFi is broken

Assuming a MIF node has both a 3GPP mobile network interface and a WiFi interface, a common practice would be to always prefer the WiFi connection when the node enters an area with WiFi available. In this

situation, a node might assume that because a valid IP address has been allocated, the WiFi link provides connectivity to destinations through the Internet. However, this might not be the case for several reasons:

- o WiFi access-point authentication requirements
- o WiFi has no global Internet connectivity
- o Instability at layer 2

In order to resolve this problem, the user would need to disable the device's interface preferences, e.g. by disabling the WiFi interface. HE-MIF offers users the possibility of configuring their preferences for the choice of the most suitable network interface to use, such as via setting on their mobile phone.

In this case, users may prefer to wait an appropriate time period for connections to be established over a WiFi path. If no connection can be made it will fall back to attempting the connection over a 3GPP mobile network path.

3.2. Policy Conflict

A node has network access via both WiFi and 3GPP networks. In a mobile network, IPv6-only may be preferable since IPv6 has the potential to be simpler than dual-stack. The WiFi access offers IPv4 only. In this scenario, the combination of source address selection [RFC6724] and preferring the WiFi interface may cause a problem. The transition to IPv6 may mean that IPv6 is the preferred protocol, so the 3GPP interface should be chosen even though it could be considered a suboptimal selection e.g. the WiFi interface likely is less expensive.

4. Happiness Parameters

This section provides input parameter proposal that HE-MIF should catch. Two sets of "Happiness" parameters have been defined. It serves applications and initiates HE-MIF connection tests subsequently. By following the process described below, MIF nodes can select an appropriate interface that best meets the configuration parameters defined by the user. The two sets of "Happiness" parameters are called Hard Set and Soft Set respectively.

4.1. Hard Set

Hard set contains parameters which should be complied with. It helps to select candidate interfaces through which a particular flow should be directed. These should be seen as constraints on the choice, such as provider policies, support for IPv4 or IPv6, and other parameters which would prevent a particular interface and transport from being used by a particular flow. Parameters in the hard set should be easy to use and understand. When several parameters in the hard set are in conflict, the user’s preference should be prioritized.

4.1.1. Operator Policy

Operators may deliver the customized policies for a particular network environment because of geo-location or service regulation considerations. One example relevant for 3GPP networks is an operator delivering policies from an Access Network Discovery and Selection function (ANDSF) [TS23.402].

The ANDSF provides a node with policies and network selection information to influence the selection between different access technologies, such as 3GPP mobile networks, WiFi access. The ANDSF can provide the node with three types of information[TS24.302].

- Access network discovery and selection information: it includes a list of access networks available in the vicinity of the node. The information may include the access technology types (e.g. WiFi), network identifiers (e.g. SSID in the case of WiFi) as well as validity conditions (e.g. where and when).

- Inter-System Mobility Policies (ISMPs): they are a set of operator-defined rules and preferences that affect the inter-system mobility decisions, e.g. decisions about whether to use 3GPP mobile network or a WiFi network.

- Inter-System Routing Policies (ISRPs): the node uses ISRPs when it can route IP traffic simultaneously over multiple radio access networks. It could provide routing policies in an IP flow granularity.

4.1.2. User Preference

User’s preference: users may express preferences which likely not have a formally technical language , like "No 3/4G while roaming", "Only download applications larger than 20Mb over WiFi", etc. Those information are normally input from User Interface (UI).

4.2. Soft Set

Soft set contains factors which impact the selection of the path across which a particular flow should be transmitted among the available interfaces and transports which meet the hard set requirements described above.

4.2.1. Provisioning Domain Identity

A PVD-aware node uses PvD Identity(PvD-ID) to select a PvD with a matching ID for special-purpose connection requests. The PvD-ID may be generated by the node implicitly or received from the network explicitly. for explicit PvDs, the node could take the parameter from PvD ID Option [I-D.ietf-mif-mpvd-id] via the configuration protocols ([I-D.ietf-mif-mpvd-dhcp-support] or [I-D.ietf-mif-mpvd-ndp-support]). A PVD-aware node may decide to use one preferred PVD or allow the use of multiple PVDs simultaneously for applications. The node behavior should be consistent with MPVD architecture [RFC7556].

4.2.2. DNS Selection

At the name service lookup step, the node has to choose a recursive DNS server to use. A HE-MIF node should take the parameter of RDNSS Selection DHCP Option [RFC6731] to select an interface for a particular namespace.

4.2.3. Next Hop

[RFC4191] allows the configuration of specific routes to a destination. A HE-MIF node should take the parameters of router preference and route information to identify the next hop.

4.2.4. Source Address Selection

For each destination, once the best next hop is found, the node should consider IP prefix and precedence parameter in policy table to select the best source address according to the rule defined in [RFC6724].

4.2.5. Common Practice

There is relevant common practice related to interface selection, e.g. Prefer WiFi over a 3GPP interface, if available. Such conventions should also be considered.

5. HE-MIF Process Requirements

An HE-MIF node may use the two sets of parameters as two steps in the interface selection process. The first step is to use the Hard Set to synthesize policies from different actors (e.g., users or network operators). These hard set parameters will provide a filter which will exclude not qualifying interfaces from any further consideration.

The second step is to influence how a node makes a connection when multiple interfaces still remain in the candidate list after first step. This is essentially sorting behavior. In the multiple provisioning domain architecture, a PVD aware node makes connectivity tests as described in Section 5.3 of [RFC7556]. A PVD agnostic node take other parameters apart from PVD-ID in the Soft Set to proceed the sort process.

The two steps are described in more details in the following sub-sections. It should be noted that HE-MIF does not prescribe such two-step model. It will be very specific to particular cases and implementations. The two step model mainly describes requirements for how to use the hard/soft set.

5.1. First Step, Filter

One goal of the filter is to reconcile multiple selection policies from users or operators. Afterwards, merged demands would be mapped to a set of candidate interfaces, which are judged as qualified.

Decision on the reconciliation of different policies will depend very much on the deployment scenario. An implementation may not be able to determine priority for each policies without explicit configuration provided by users or administrator. For example, an implementation may by default always prefer the WiFi because of cost saving consideration. Whereas, other users may turn off a device’s WiFi interface to guarantee use of a 3GPP network interface to assure higher reliability or security.

The decision on mergence of policies may be made by implementations, or by node administrators. However, it’s worth to note that a demand from users should be normally considered higher priority than from other actors.

The merged policies serve as a filter which is iterated across the list of available interfaces. Qualified interfaces are selected and the proceed to the second step.

5.2. Second Step, Sort

5.2.1. Interface Validation

The Sort process aims to select the best interface and provide fallback capacities. As stated in [RFC7556], a PVD-aware node shall perform connectivity tests and, only after validation of the PVD, consider using it to serve application connections requests. In current implementations, some nodes already implement this, e.g., by trying to reach a dedicated web server (see Section 3.1.2 [RFC6419]). If anything is abnormal, it assumes there is a proxy on the path. This status detection is recommended to be used in HE-MIF to detect DNS interception or an HTTP proxy that forces a login or a click-through. Unexamined PVDs or interfaces should be accounted as "unconnected". It should not join the sort process.

5.2.2. Name Resolution

Name resolution is executed on the validated interfaces. Before the requests are initiated, it should check if there is a matching PVD ID for the destination name. A PVD agnostic node may request DNS server selection DHCP option [RFC6731] for interface selection guidance. Those information may weight a particular interface to be preferred to others sending resolving requests. If the node can’t find useful information in the Soft Set, DNS queries would be sent out on multiple interfaces in parallel to maximize chances for connectivity. Some additional discussions of DNS selection consideration of HE-MIF are described in Section 7.3.

5.2.3. Connection Establishment

Once a destination address was resolved, a connection is to be setup. For the given destination address, a PVD-aware node selects a next-hop and source address associated with that PVD in the name resolution process. A PVD agnostic node may receive certain next hop in a RA message [RFC4191], the node selects best source address according to the rules [RFC6724].

The interface identified by the source address should be treated to initiate the connection prior to others. This could avoid thrashing the network, by not making simultaneous connection attempts on multiple interfaces. After making a connection attempt on the preferred pairs and failing to establish a connection within a certain time period (see Section 7.2), a HE-MIF implementation will decide to initiate connection attempt using rest of interfaces in parallel. This fallback consideration will make subsequent connection attempts successful on non-preferable interfaces.

The node would cache information regarding the outcome of each connection attempt. Cache entries would be flushed periodically. A system-defined timeout may take place to age the state. Maximum on the order of 10 minutes defined in [RFC6555] is recommended to keep the interface state changes synchronizing with IP family states.

If there is no specific Soft Set provided, all selected interfaces should be treated equally. for a node implementing multipath transports (for example, Multipath TCP (MPTCP) [RFC6182]), the interfaces could be treated as valid to perform subsequent multipath process, such as starting subflow. A node only supporting single physical transport would initiate on several interface simultaneously. The goal here is to provide the most fast connection for users, by quickly attempting to connect using each candidate interface. Afterwards, the node would do the same caching and flushing process as described above.

6. Implementation Framework

The simplest way to implement the processes described in this document is within the application itself. This would not require any specific support from the operating system beyond the commonly available APIs that provide transport service. It could also be implemented using a high-level API approach, linking to the MIF-API [I-D.ietf-mif-api-extension].

7. Additional Considerations

7.1. Usage Scope

Connection-oriented transports (e.g., TCP, SCTP) are directly applied as scoped in [RFC6555]. For connectionless transport protocols (e.g., UDP), a similar mechanism can be used if the application has request/response semantics. Further investigations are out of the document scope.

7.2. Fallback Timeout

When the preferred interface was failed, HE-MIF would trigger a fallback process to start connection initiation on several candidate interfaces. A period of time should be set to invalidate the interface and fallback to others. Aggressive timeouts may achieve quick interface handover, but at the cost of traffic that may be chargeable on certain networks, e.g. the handover from WiFi to 3GPP networks brings a charge to customers. Considering the reasons, it is recommended to prioritize the input from users (e.g., real customers or applications) through user interface. For default-setting on a system, a hard error [RFC1122] in replied ICMP could

serve as a trigger for the fallback process. When the ICMP soft error is present or non-response was received, it’s recommended that the timeout should be large enough to allow connection retransmission. [RFC1122] states that such timer must be at least 3 minutes to provide TCP retransmission. However, several minutes delay may not inappropriate for user experiences. A widespread practice [RFC5461] sets 75 seconds to optimize connection process.

More optimal timer may be expected. The particular setting will be very specific to implementations and cases. The memo didn’t try to provide a concrete value because of following concerns.

- RTT (Round-Trip Time) on different interfaces may vary quite a lot. A particular value of timeout may not accurately help to make a decision that this interface doesn’t work at all. On the contrary, it may cause a misjudgment on a interface, which is not very fast. In order to compensate the issues, the timeout setting based on past experiences of a particular interface may help to make a fair decision. Whereas, it’s going beyond the capability of Happy Eyeballs [RFC6555]. Therefore, it leaves a particular implementation.

- In some cases, fast interface may not be treated as "best". For example, a interface could be evaluated in the principle of bandwidth-delay, termed "Bandwidth-Delay-Product ". Happy Eyeballs measures only connection speed. That is, how quickly a TCP connection is established . It does not measure bandwidth. If the fallback has to take various factors into account and make balanced decision, it’s better to resort to a specific context and implementation.

7.3. DNS Selections

During the Sort process, HE-MIF prioritizes PVD-ID match or [RFC6731] inputs to select a proper server. It could help to address following two cases.

- A DNS answer may be only valid for a specific provisioning domain, but the DNS resolver may not be aware of that because the DNS reply is not kept with the provisioning from which the answer comes. The situation may become worse if asking internal name with public address response or asking public name with private address answers.

- Some FQDNs can be resolvable only by sending queries to the right server (e.g., intranet services). Otherwise, a response with NXDOMAIN is replied. Fast response is treated as optimal only if

the record is valid. That may cause messy for data connections, since NXDOMAIN doesn't provide useful information.

HE-MIF can help to solve the issues of DNS interception with captive portal. The DNS server modified and replied the answer with the IP address of captive portal rather than the intended destination address. In those cases, TCP connection may succeed, but Internet connectivity is not available. It results in lack of service unless user has authenticated. HE-MIF recommended using network connectivity status probes to examine a pre-configured URL for detecting DNS interception on the path (see more in Section 5.2). The node will be able to automatically rely upon other interfaces to select right DNS servers by excluding the unexamined interfaces.

7.4. Flow Continuity

[I-D.deng-mif-api-session-continuity-guide] describes session continuity guidance for application developers. The flow continuity topic is beyond this document scope.

7.5. Interworking with Happy Eyeball

HE-MIF process could cooperate with HE [RFC6555]. HE is executed on an interface which is selected to make connection establishment (see Section 5.2.3). for example, a node following PvD policy to pick a interface and make both IPv4/IPv6 connection attempts in consistent with HE requirements. The interface state management in HE-MIF is designed to synchronize with IP family states. It could facilitate the HE executions.

7.6. Multipath Applicability

Some nodes may support transports that provide an abstraction of a single connection, aggregating multiple underlying connections. Multipath TCP (MPTCP) [RFC6182] is an example of such a transport protocol. For connections provided by such transports, a node may leverage the "happiness" parameters and process on the underlying connections. Following the HE-MIF requirements, each connection could be performed consistently with user/operator's preference and corresponding provisioning domain information.

8. IANA Considerations

This memo does not include any IANA requests.

9. Security Considerations

The security consideration is following the statement in [RFC6555] and [RFC6418].

10. Acknowledgements

The authors would like to thank Margaret Wasserman, Hui Deng, Erik Kline, Stuart Cheshire, Teemu Savolainen, Jonne Soininen, Simon Perreault, Zhen Cao, Dmitry Anipko, Ted Lemon, Daniel Migault, Russ White and Bing Liu for their helpful comments.

Many thanks to Ralph Droms, Ian Farrer, Jouni Korhonen, Mirja Khlewind and Suresh Krishnan for their detailed reviews.

11. References

11.1. Normative References

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <http://www.rfc-editor.org/info/rfc1122>.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <http://www.rfc-editor.org/info/rfc4191>.

[RFC6555] Wing, D. and A. Yourtchenko, "Happy Eyeballs: Success with Dual-Stack Hosts", RFC 6555, DOI 10.17487/RFC6555, April 2012, <http://www.rfc-editor.org/info/rfc6555>.

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <http://www.rfc-editor.org/info/rfc6724>.

[RFC6731] Savolainen, T., Kato, J., and T. Lemon, "Improved Recursive DNS Server Selection for Multi-Interfaced Nodes", RFC 6731, DOI 10.17487/RFC6731, December 2012, <http://www.rfc-editor.org/info/rfc6731>.

[TS23.402] 3rd Generation Partnership Project, 3GPP., "Architecture enhancements for non-3GPP accesses v8.8.0", December 2009.

[TS24.302]
3rd Generation Partnership Project, 3GPP., "Access to the 3GPP Evolved Packet Core (EPC) via non-3GPP access networks v14.0.0", June 2016.

11.2. Informative References

[I-D.deng-mif-api-session-continuity-guide]
Deng, H., Krishnan, S., Lemon, T., and M. Wasserman, "Guide for application developers on session continuity by using MIF API", draft-deng-mif-api-session-continuity-guide-04 (work in progress), July 2014.

[I-D.ietf-mif-api-extension]
Liu, D., Lemon, T., Ismailov, Y., and Z. Cao, "MIF API consideration", draft-ietf-mif-api-extension-05 (work in progress), February 2014.

[I-D.ietf-mif-mpvd-dhcp-support]
Krishnan, S., Korhonen, J., and S. Bhandari, "Support for multiple provisioning domains in DHCPv6", draft-ietf-mif-mpvd-dhcp-support-02 (work in progress), October 2015.

[I-D.ietf-mif-mpvd-id]
Krishnan, S., Korhonen, J., Bhandari, S., and S. Gundavelli, "Identification of provisioning domains", draft-ietf-mif-mpvd-id-02 (work in progress), October 2015.

[I-D.ietf-mif-mpvd-ndp-support]
Korhonen, J., Krishnan, S., and S. Gundavelli, "Support for multiple provisioning domains in IPv6 Neighbor Discovery Protocol", draft-ietf-mif-mpvd-ndp-support-03 (work in progress), February 2016.

[RFC5461] Gont, F., "TCP’s Reaction to Soft Errors", RFC 5461, DOI 10.17487/RFC5461, February 2009, <http://www.rfc-editor.org/info/rfc5461>.

[RFC6182] Ford, A., Raiciu, C., Handley, M., Barre, S., and J. Iyengar, "Architectural Guidelines for Multipath TCP Development", RFC 6182, DOI 10.17487/RFC6182, March 2011, <http://www.rfc-editor.org/info/rfc6182>.

[RFC6418] Blanchet, M. and P. Seite, "Multiple Interfaces and Provisioning Domains Problem Statement", RFC 6418, DOI 10.17487/RFC6418, November 2011, <http://www.rfc-editor.org/info/rfc6418>.

[RFC6419] Wasserman, M. and P. Seite, "Current Practices for Multiple-Interface Hosts", RFC 6419, DOI 10.17487/RFC6419, November 2011, <http://www.rfc-editor.org/info/rfc6419>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <http://www.rfc-editor.org/info/rfc7556>.

Authors' Addresses

Gang Chen
China Mobile
29, Jinrong Avenue
Xicheng District,
Beijing 100033
China

Email: phdgang@gmail.com, chengang@chinamobile.com

Carl Williams
Consultant
El Camino Real
Palo Alto, CA 94306
USA

Email: carlw@mcsr-labs.org

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134
USA

Email: dwing@cisco.com

Andrew Yourtchenko
Cisco Systems, Inc.
De Kleetlaan, 7
Diegem B-1831
Belgium

Email: ayourtch@cisco.com

DHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2016

S. Krishnan
Ericsson
J. Korhonen
Broadcom Corporation
S. Bhandari
Cisco Systems
October 19, 2015

Support for multiple provisioning domains in DHCPv6
draft-ietf-mif-mpvd-dhcp-support-02

Abstract

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks. One part of the solution requires associating configuration information with provisioning domains. This document details how configuration information provided through DHCPv6 can be associated with provisioning domains.

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 April 21, 2016.

Copyright Notice

Table of Contents

1. Introduction

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks based on the Multiple Provisioning Domains (MPVD) architecture work [RFC7556]. One part of the solution requires associating configuration information with provisioning domains. This document describes a DHCPv6 mechanism for explicitly indicating provisioning domain information along with any configuration that will be provided. The proposed mechanism uses a DHCPv6 option that indicates the identity of the provisioning domain and encapsulates the options that contain the configuration information as well as any accompanying authentication/authorization information.

2. Terminology

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 [RFC2119].

3. PVD Container option

The PVD container option is used to encapsulate and group together all the configuration options that belong to the explicitly identified provisioning domain. The PVD container option MUST encapsulate exactly one OPTION_PVD_ID. The PVD container option MAY occur multiple times in the same message, but each of these PVD container options MUST have a different PVD identity specified under its PVD identity option. The PVD container option SHOULD contain at most one (i.e. zero or one) OPTION_PVD_AUTH.

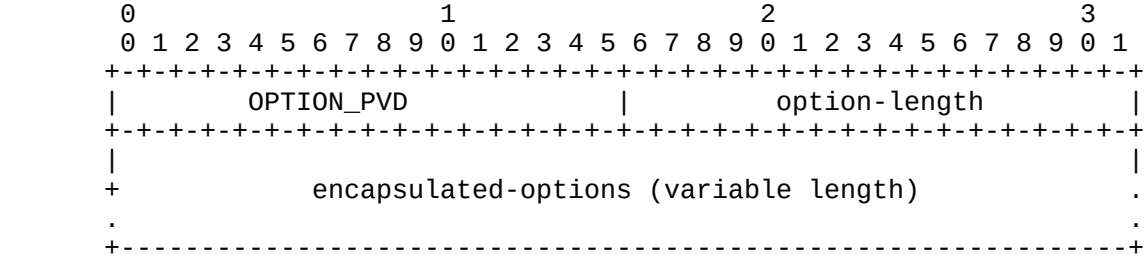


Figure 1: PVD Container Option

- option-code: OPTION_PVD (TBA1)
- option-length: Length of encapsulated options
- encapsulated-options: options associated with this provisioning domain.

4. PVD Identity option

The PVD identity option is used to explicitly indicate the identity of the provisioning domain that is associated with the configuration information encapsulated by the PVD container option.

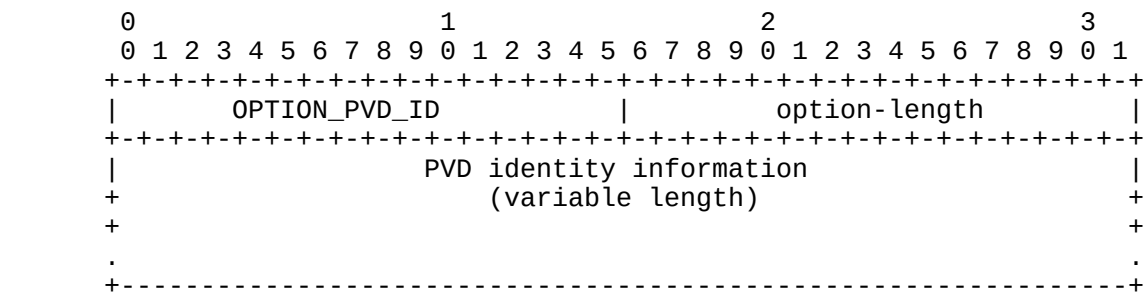


Figure 2: PVD ID Option

- option-code: OPTION_PVD_ID (TBA2)

- option-length: Length of PVD identity information

- PVD identity information: The provisioning domain identity. The contents of this field is defined in a separate document [I-D.ietf-mif-mpvd-id].

5. PVD Authentication and Authorization option

The PVD authentication and authorization option contains information that could be used by the DHCPv6 client to verify whether the configuration information provided was not tampered with by the DHCPv6 server as well as establishing that the DHCPv6 server was authorized to advertise the information on behalf of the PVD per OPTION_PVD basis. The contents of the authentication/authorization information is provided by the owner of the provisioning domain and is completely opaque to the DHCPv6 server that passes along the information unmodified. Every OPTION_PVD option SHOULD contain at most one (i.e. zero or one) OPTION_PVD_AUTH option. If present, the OPTION_PVD_AUTH option MUST be the last option inside the OPTION_PVD option.

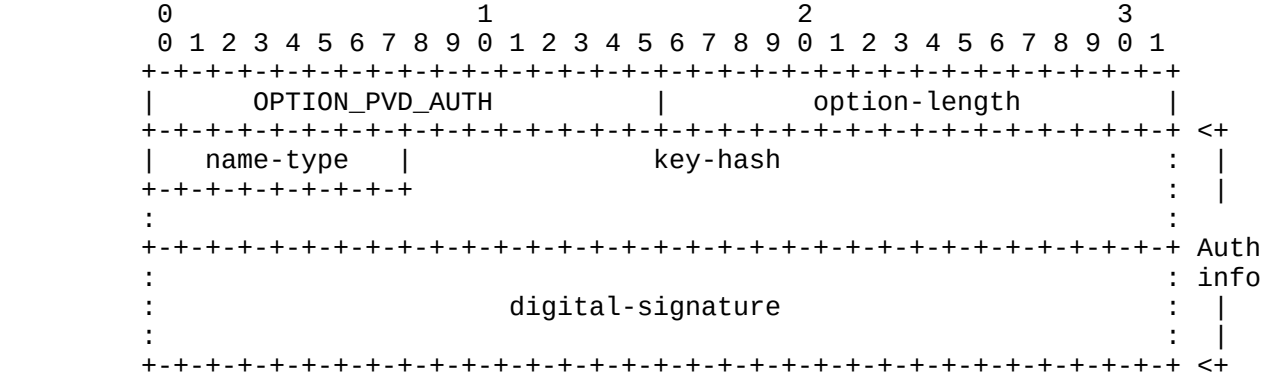


Figure 3: PVD Auth Option

- option-code: OPTION_PVD_AUTH (TBA3)

- option-length: Length of the Auth info

- name-type: Names the algorithm used to identify a specific X.509 certificate using the method defined for the Subject Key Identifier (SKI) extension for the X.509 certificates. The usage and the Name Type registry aligns with the mechanism

defined for SeND [RFC6494][RFC6495].
Name Type values starting
from 3 are supported and an implementation MUST at least support SHA-1 (value 3).

- key-hash: A hash of the public key using the algorithm identified by the Name Type. The procedure how the Key Hash is calculated is defined in [RFC3971] and [RFC6495].

- digital-signature: A signature calculated over the encapsulating OPTION_PVD including all option data from the beginning of the option while setting the digital-signature field to zero. The procedure of calculating the signature is identical to the one defined for SeND [RFC3971].

5.1. Authentication is optional

The OPTION_PVD_AUTH will be sent only if it is requested in the ORO.If the client does not request this option, it will not get a signed PVD option. Clients that request the OPTION_PVD with the intention of redistributing configuration information to other clients (e.g. CPE routers) SHOULD NOT request the OPTION_PVD_AUTH in the ORO. This allows them to send out a subset of the received options to their clients and also allows them to support PVD unaware clients by sending out the options without PVD information.

6. Set of allowable options

The PVD container option MAY be used to encapsulate any allocated DHCPv6 options but MUST NOT be used to encapsulate another OPTION_PVD option.

7. Behaviour of DHCPv6 entities

This section describes role of DHCPv6 entities involved in requesting and receiving DHCPv6 configuration or prefix and address allocation.

7.1. Client and Requesting Router Behavior

DHCPv6 client or requesting router can request for configuration from provisioning domain in the following ways:

- In the SOLICIT message it MAY include OPTION_PVD_ID requesting configuration for the specific PVD ID indicated in the OPTION_PVD_ID option. It can include multiple OPTION_PVD_ID options to indicate its preference for more than one provisioning domain. The PVD ID it requests is learnt via configuration or any other out of band mechanism not defined in this document.

- In the SOLICIT message include an OPTION_ORO option with the OPTION_PVD option code to request configuration from all the PVDs that the DHCPv6 server can provide.

The client or requesting router parses OPTION_PVD options in the response message. The Client or Requesting router MUST then include all or subset of the received OPTION_PVD options in the REQUEST message so that it will be responsible for the configuration information selected.

If DHCPv6 client or requesting router receives OPTION_PVD options but does not support PVD, it SHOULD ignore the received option(s).

7.2. Relay Agent Behavior

If the relay agent supports both the Relay-Supplied DHCP Option (RSOO) [RFC6422] and the PVD, and it is configured to request configuration data for clients in one or more provisioning domains, then the relay agent MAY include the RSOO in the Relay-Forward message. The RSOO MAY contain zero or more OPTION_PVD options. The relay agent MUST NOT include any OPTION_PVD options into the RSOO unless the client has indicated support for the PVD as described in Section Section 7.1.

7.3. Server and Delegating Router Behavior

If the Server or Delegating router supports PVD and it is configured to provide configuration data in one or more provisioning domains, it selects configuration for the PVD based allocation in the following way:

- If OPTION_PVD option code within OPTION_ORO is not present in the request, it MUST NOT include provisioning domain based configuration. It MAY select configuration and prefix allocation from a default PVD defined.

- If OPTION_PVD_ID is included, it selects information to be offered from that specific PVD if available.

- If OPTION_PVD option code within OPTION_ORO is included, then based on its configuration and policy it MAY offer configuration from the available PVD(s).

When PVD information and configuration are selected for address and prefix allocation the server or delegating router responds with an ADVERTISE message after populating OPTION_PVD.

If OPTION_PVD is not included, then the server or delegating router MAY allocate the prefix and provide configuration as specified in [RFC3315] and[RFC3633] and MUST NOT include OPTION_PVD option in the response.

If OPTION_ORO option includes the OPTION_PVD option code but the server or delegating router does not support PVD, then it SHOULD ignore the OPTION_PVD and OPTION_PVD_ID options received.

If both client/requesting router and server/delegating router support PVD but cannot offer configuration with PVD for any other reason, it MUST respond to client/requesting router with appropriate status code as specified in [RFC3315] and [RFC3633].

Similarly, if the OPTION_PVD is received in the RSOO from the relay agent the above described procedures apply for including the PVD specific configuration information back to the client.

8. Redistributing configuration information

Clients that redistribute configuration information further to legacy clients (e.g. homenet CPEs) after stripping out the PVD information need to exercise caution when removing information from the PVD containers and distributing them as top level options. Configuration information that is consistent within a PVD container may not work equally well when mixed with other top level configuration information or information from other PVD containers. e.g. Using DNS server information from one PVD container while using address/prefix from another may lead to unexpected results.

9. Security Considerations

An attacker may attempt to modify the information provided inside the PVD container option. These attacks can easily be prevented by using the DHCPv6 AUTH option [RFC3315] that would detect any form of tampering with the DHCPv6 message contents.

A compromised DHCPv6 server or relay agent may insert configuration information related to PVDs it is not authorized to advertise. e.g. A coffee shop DHCPv6 server may provide configuration information purporting to be from an enterprise and may try to attract enterprise related traffic. The only real way to avoid this is that the PVD container contains embedded authentication and authorization information from the owner of the PVD. Then, this attack can be detected by the client by verifying the authentication and authorization information provided inside the PVD container option after verifying its trust towards the PVD owner (e.g. a certificate with a well-known/common trust anchor).

A compromised configuration source or an on-link attacker may try to capture advertised configuration information and replay it on a different link or at a future point in time. This can be avoided by including some replay protection mechanism such as a timestamp or a nonce inside the PVD container to ensure freshness of the provided information.

10. IANA Considerations

This document defines three new DHCPv6 options to be allocated out of the registry at http://www.iana.org/assignments/dhcpv6-parameters/

OPTION_PVD (TBA1)
OPTION_PVD_ID (TBA2)
OPTION_PVD_AUTH (TBA3)

This document also adds OPTION_PVD (TBA1) into the "Options Permitted in the Relay-Supplied Options Option" registry at http://www.iana.org/assignments/dhcpv6-parameters/

11. Acknowledgements

The authors would like to thank the members of the MIF architecture design team for their comments that led to the creation of this draft. The authors would also thank Ian Farrer and Steven Barth for their reviews and comments.

12. References

12.1. Normative References

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

[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.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <http://www.rfc-editor.org/info/rfc4122>.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, DOI 10.17487/RFC4282, December 2005, <http://www.rfc-editor.org/info/rfc4282>.

[RFC6422] Lemon, T. and Q. Wu, "Relay-Supplied DHCP Options", RFC 6422, DOI 10.17487/RFC6422, December 2011, <http://www.rfc-editor.org/info/rfc6422>.

[RFC6494] Gagliano, R., Krishnan, S., and A. Kukec, "Certificate Profile and Certificate Management for SEcure Neighbor Discovery (SEND)", RFC 6494, DOI 10.17487/RFC6494, February 2012, <http://www.rfc-editor.org/info/rfc6494>.

[RFC6495] Gagliano, R., Krishnan, S., and A. Kukec, "Subject Key Identifier (SKI) SEcure Neighbor Discovery (SEND) Name Type Fields", RFC 6495, DOI 10.17487/RFC6495, February 2012, <http://www.rfc-editor.org/info/rfc6495>.

12.2. Informative References

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <http://www.rfc-editor.org/info/rfc7556>.

Authors' Addresses

Suresh Krishnan
Ericsson
8400 Decarie Blvd.
Town of Mount Royal, QC
Canada

Phone: +1 514 345 7900 x42871
Email: suresh.krishnan@ericsson.com

Jouni Korhonen
Broadcom Corporation
3151 Zanker Road
San Jose, CA 95134
USA

Email: jouni.nospam@gmail.com

Shwetha Bhandari
Cisco Systems
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Phone: +91 80 4426 0474
Email: shwethab@cisco.com

mif Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2016

S. Krishnan
Ericsson
J. Korhonen
Broadcom Corporation
S. Bhandari
Cisco Systems
S. Gundavelli
Cisco

October 19, 2015

Identification of provisioning domains
draft-ietf-mif-mpvd-id-02

Abstract

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks. This document describes several methods of generating identification information for provisioning them and a format for carrying such identification in configuration protocols.

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 April 21, 2016.

Table of Contents

1. Introduction

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks based on the Multiple Provisioning Domains (MPVD) architecture work [RFC7556]. This document describes a format for carrying identification information along with a few alternatives for reasonable sources for Provisioning Domain (PVD) identification. Since the PVD identities (PVD ID) are expected to be unique, the identification sources provide some level of uniqueness using either a hierarchical structure (e.g. FQDNs and OIDs) or some form of randomness (e.g. UUID and ULAs). Any source that does not provide either guaranteed or probabilistic uniqueness is probably not a good candidate for identifying provisioning domains.

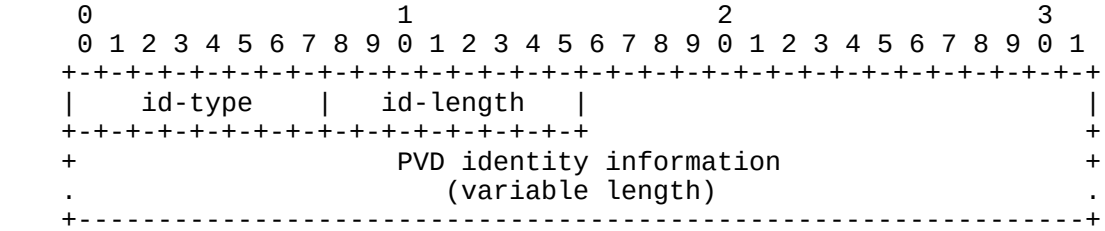
2. Terminology

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 [RFC2119].

3. Provisioning domain identity format

The identity of the PVD is independent of the configuration protocol used to communicate it. Furthermore, the PVD identity SHOULD only contain information related to the identification purposes and not encode additional provisioning domain specific configuration information. The configuration protocol used and its extensions are meant for that purpose [I-D.ietf-mif-mpvd-dhcp-support]

[I-D.ietf-mif-mpvd-ndp-support]. The PVD identity is formatted as follows.



PVD ID Option

- id-type: Describes the type of identification information. This document defines six types of PVD identity information
  0x01: UUID [RFC4122]
  0x02: UTF-8 string
  0x03: OID [OID]
  0x04: NAI Realm [RFC4282]
  0x05: FQDN [RFC1035]
  0x06: ULA Prefix [RFC4193]
  Further types can be added by IANA action.

- id-length: Length of the PVD identification in octets not including the id-type and id-length fields. The length of the PVD identity is dependent on the id-type and is defined by the document that specifies that kind of ID.

- PVD identity information: The PVD identification that is based on the id-typ.The format of the PVD identity is dependent on the id-type and is defined by the document that specifies that kind of ID.

4. Security Considerations

An attacker may attempt to modify the PVD identity provided in a configuration protocol. These attacks can be prevented by using the configuration protocol mechanisms such as SEND [RFC3971] and DHCPv6 AUTH option [RFC3315] that detect any form of tampering with the configuration.

A compromised configuration source, on the other hand, cannot easily be detected by a configuration client. The only real way to avoid this is that the PVD identification is directly associable to some form of authentication and authorization information from the owner

of the PVD (e.g. an FQDN can be associated with a DANE cert). Then, this attack can be detected by the client by verifying the authentication and authorization information provided inside the PVD container option (such as the OPTION_PVD_AUTH inside OPTION_PVD [I-D.ietf-mif-mpvd-dhcp-support] or the Key Hash and Digital Signature inside PVD_CO [I-D.ietf-mif-mpvd-ndp-support]) verifying its trust towards the PVD owner (e.g. a certificate with a well-known /common trust anchor that).

5. IANA Considerations

This document creates a new registry for PVD id types. The initial values are listed below

0x01: UUID [RFC4122]
0x02: UTF-8 string
0x03: OID [OID]
0x04: NAI Realm [RFC4282]
0x05: FQDN [RFC1035]
0x06: ULA Prefix [RFC4193]

6. Acknowledgements

The authors would like to thank the members of the MIF architecture design team, Ted Lemon, Brian Carpenter, Bernie Volz and Alper Yegin for their contributions to this draft. The authors also thank Ian Farrer, Erik Kline, Dave Thaler and Steven Barth for their reviews and comments that improved this draft.

7. References

7.1. Normative References

[OID] IANA, "PRIVATE ENTERPRISE NUMBERS", SMI Network Management Private Enterprise Codes, http://www.iana.org/assignments/enterprise-numbers/enterprise-numbers, March 2013.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

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

[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.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <http://www.rfc-editor.org/info/rfc4122>.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, DOI 10.17487/RFC4282, December 2005, <http://www.rfc-editor.org/info/rfc4282>.

7.2. Informative References

[I-D.ietf-mif-mpvd-dhcp-support] Krishnan, S., Korhonen, J., and S. Bhandari, "Support for multiple provisioning domains in DHCPv6", draft-ietf-mif-mpvd-dhcp-support-01 (work in progress), March 2015.

[I-D.ietf-mif-mpvd-ndp-support] Korhonen, J., Krishnan, S., and S. Gundavelli, "Support for multiple provisioning domains in IPv6 Neighbor Discovery Protocol", draft-ietf-mif-mpvd-ndp-support-01 (work in progress), February 2015.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <http://www.rfc-editor.org/info/rfc7556>.

Authors' Addresses

Suresh Krishnan
Ericsson
8400 Decarie Blvd.
Town of Mount Royal, QC
Canada

Phone: +1 514 345 7900 x42871
Email: suresh.krishnan@ericsson.com

Jouni Korhonen
Broadcom Corporation
3151 Zanker Road
San Jose, CA 95134
USA

Email: jouni.nospam@gmail.com

Shwetha Bhandari
Cisco Systems
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Phone: +91 80 4426 0474
Email: shwethab@cisco.com

Sri Gundavelli
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: sgundave@cisco.com

MIF
Internet-Draft
Intended status: Standards Track
Expires: August 28, 2016

J. Korhonen
Broadcom Limited
S. Krishnan
Ericsson
S. Gundavelli
Cisco Systems

February 25, 2016

Support for multiple provisioning domains in IPv6 Neighbor Discovery Protocol
draft-ietf-mif-mpvd-ndp-support-03

Abstract

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks. One part of the solution requires associating configuration information with provisioning domains. This document details how configuration information provided through IPv6 Neighbor Discovery Protocol can be associated with provisioning domains.

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 August 28, 2016.

Copyright Notice

Table of Contents

1. Introduction

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks based on the Multiple Provisioning Domains (MPVD) architecture work [RFC7556]. One part of the solution requires associating configuration information with Provisioning Domains (PVD). This document describes an IPv6 Neighbor Discovery Protocol (NDP) [RFC4861] mechanism for explicitly indicating provisioning domain information along with any configuration which is associated with that provisioning domain. The proposed mechanism uses an NDP option that indicates the identity of the provisioning domain and encapsulates the options that contain the configuration information as well as optional authentication/authorization information. The solution defined in this document aligns as much as possible with the existing IPv6 Neighbor Discovery security, namely with Secure Neighbor Discovery (SeND) [RFC3971].

2. Terminology

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 [RFC2119].

3. PVD Container option

The PVD container option (PVD_CO) is used to encapsulate the configuration options that belong to the explicitly identified provisioning domain. The PVD container option always encapsulates exactly one PVD identity. The PVD container option MAY occur multiple times in a Router Advertisement (RA) message. In this case each PVD container MUST belong to a different provisioning domain. The PVD container options MUST NOT be nested. The PVD Container option is defined only for the RA NDP message.

Since implementations are required to ignore any unrecognized options [RFC4861], the backward compatibility and the reuse of existing NDP options is implicitly enabled. Implementations that do not recognize the PVD container option will ignore it, and any PVD container option "encapsulated" NDP options without associating them into any provisioning domain (since the implementation has no notion of provisioning domains). For example, the PVD container could "encapsulate" a Prefix Information Option (PIO), which would mark that this certain advertised IPv6 prefix belongs and originates from a specific provisioning domain. However, if the implementation does not understand provisioning domains, then this specific PIO is also skipped and not configured on the interface.

The optional security for the PVD container is based on X.509 certificates [RFC6487] and reuses mechanisms already defined for SeND [RFC3971] [RFC6495]. However, the use of PVD containers does not assume or depend on SeND being deployed or even implemented. The PVD containers SHOULD be signed per PVD certificates, which provides both integrity protection and proves that the configuration information source is authorized for advertising the given information. See [RFC6494] for discussion how to enable deployments where the certificates needed to sign PVD containers belong to different administrative domains i.e., to different provisioning domains.

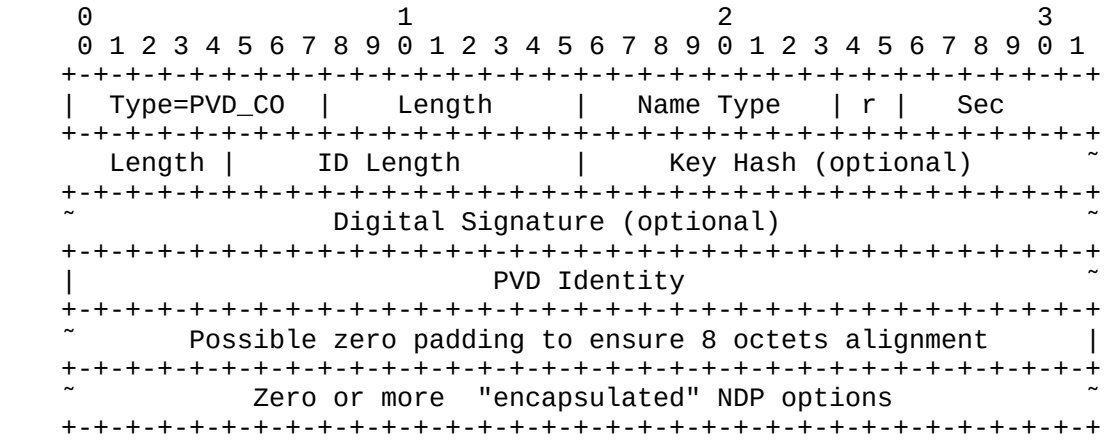


Figure 1: PVD Container Option

Type

PVD Container; Set to TBD1.

Length

Length of the PVD_CO. The actual length depends on the number of "encapsulated" NDP options, length of the PVD Identity, and the optional Key Hash/Digital Signature/Padding.

Name Type

Names the algorithm used to identify a specific X.509 certificate using the method defined for the Subject Key Identifier (SKI) extension for the X.509 certificates. The usage and the Name Type registry aligns with the mechanism defined for SeND [RFC6495]. Name Type values starting from 3 are supported and an implementation MUST at least support SHA-1 (value 3). Note that if Sec Length=0 the Name field serves no use and MUST be set to 0.

r

Reserved. MUST be set to 0 and ignored when received.

Sec Length

11-bit length of the Key Hash and Digital Signature in a units of 1 octet. When no security is enabled the Sec Length MUST be set to value of 0.

ID Length

11-bit length of the PVD Identity in a units of 1 octet. The ID Length MUST be greater than 0.

Key Hash

This field is only present when Sec Length>0. A hash of the public key using the algorithm identified by the Name Type. The procedure how the Key Hash is calculated is defined in [RFC3971] and [RFC6495].

Digital Signature

This field is only present when Sec Length>0. A signature calculated over the PVD_CO option including all option data from the beginning of the option until to the end of the container. The procedure of calculating the signature is identical to the one defined for SeND [RFC3971]. During the signature calculation the contents of the Digital Signature option MUST be treated as all zero.

PVD Identity

The provisioning domain identity. The contents of this field is defined in a separate document [I-D.ietf-mif-mpvd-id].

Implementations MUST ensure that the PVD container option meets the 8 octets NDP option alignment requirement as described in [RFC4861].

If the PVD_CO does not contain a digital signature, then other means to secure the integrity of the NDP message SHOULD be provided, such as utilizing SeND. However, the security provided by SeND is for the entire NDP message and does not allow verifying whether the sender of the NDP message is actually authorized for the information for the provisioning domain.

If the PVD_CO contains a signature and the verification fails, then the whole PVD_CO option MUST be silently ignored and the event SHOULD be logged.

4. Set of allowable options

The PVD container option MAY be used to encapsulate any allocated IPv6 NDP options, which may appear more than once in a NDP message. The PVD container option MUST NOT be used to encapsulate other PVD_CO option(s).

5. Security Considerations

An attacker may attempt to modify the information provided inside the PVD container option. These attacks can easily be prevented by using SeND [RFC3971] or per PVD container signature that would detect any form of tampering with the IPv6 NDP message contents.

A compromised router may advertise configuration information related to provisioning domains it is not authorized to advertise. e.g. A coffee shop router may provide configuration information purporting to be from an enterprise and may try to attract enterprise related traffic. The only real way to avoid this is that the provisioning domain container contains embedded authentication and authorization information from the owner of the provisioning domain. Then, this attack can be detected by the client by verifying the authentication and authorization information provided inside the PVD container option after verifying its trust towards the provisioning domain owner (e.g. a certificate with a well-known/common trust anchor).

A compromised configuration source or an on-link attacker may try to capture advertised configuration information and replay it on a different link or at a future point in time. This can be avoided by including some replay protection mechanism such as a timestamp or a nonce inside the PVD container to ensure freshness of the provided information. This specification does not define a replay protection solution. Rather it is assumed that if replay protection is required, the access network and hosts also deploy existing security solutions such as SeND [RFC3971].

6. IANA Considerations

This document defines two new IPv6 NDP options into the "IPv6 Neighbor Discovery Option Formats" registry. Option TBD1 is described in Section 3.

7. Acknowledgements

The authors would like to thank the members of the MIF architecture design team for their comments that led to the creation of this draft.

8. References

8.1. Normative References

[I-D.ietf-mif-mpvd-id]
Krishnan, S., Korhonen, J., Bhandari, S., and S. Gundavelli, "Identification of provisioning domains", draft-ietf-mif-mpvd-id-02 (work in progress), October 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <http://www.rfc-editor.org/info/rfc3971>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

[RFC6494] Gagliano, R., Krishnan, S., and A. Kukec, "Certificate Profile and Certificate Management for SEcure Neighbor Discovery (SEND)", RFC 6494, DOI 10.17487/RFC6494, February 2012, <http://www.rfc-editor.org/info/rfc6494>.

[RFC6495] Gagliano, R., Krishnan, S., and A. Kukec, "Subject Key Identifier (SKI) SEcure Neighbor Discovery (SEND) Name Type Fields", RFC 6495, DOI 10.17487/RFC6495, February 2012, <http://www.rfc-editor.org/info/rfc6495>.

8.2. Informative References

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <http://www.rfc-editor.org/info/rfc6487>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <http://www.rfc-editor.org/info/rfc7556>.

Appendix A. Examples

A.1. One implicit PVD and one explicit PVD

Figure 2 shows how the NDP options are laid out in an RA for one implicit provisioning domain and one explicit provisioning domain. The example does not include security (and signing of the PVD container). The assumption is the PVD identity consumes total 18 octets (for example encoding a NAI Realm string "dana.example.com").

The explicit provisioning domain contains a specific PIO for 2001:db8:abad:cafe::/64 and the MTU of 1337 octets. The implicit provisioning domain configures a prefix 2001:db8:cafe:babe::/64 and the link MTU of 1500 octets. There are two cases: 1) the host receiving the RA implements provisioning domains and 2) the host does not understand provisioning domains.

1. The host recognizes the PVD_CO and "starts" a provisioning domain specific configuration. Security is disabled, thus there are no Key Hash or Digital Signature fields to process. The prefix 2001:db8:abad:cafe::/64 is found and configured on the interface. Once the PVD_ID option is located the interface prefix configuration for 2001:db8:abad:cafe::/64 and the MTU of 1337 octets can be associated to the provisioning domain found in the PVD_CO option.

   The rest of the options are parsed and configured into the implicit provisioning domain since there is no encapsulating provisioning domain. The interface is configured with prefix 2001:db8:cafe:babe::/64. The implicit provisioning domain uses the link MTU of 1500 octets, whereas the "dana.example.com" provisioning domain uses the MTU of 1337 octets (this means when packets are sourced using 2001:db8:abad:cafe::/64 prefix the link MTU is different than when sourcing packets using 2001:db8:cafe:babe::/64 prefix).

2. The host ignores the PVD_CO and ends up configuring one prefix on its interface (2001:db8:cafe:babe::/64) with a link MTU of 1500 octets.

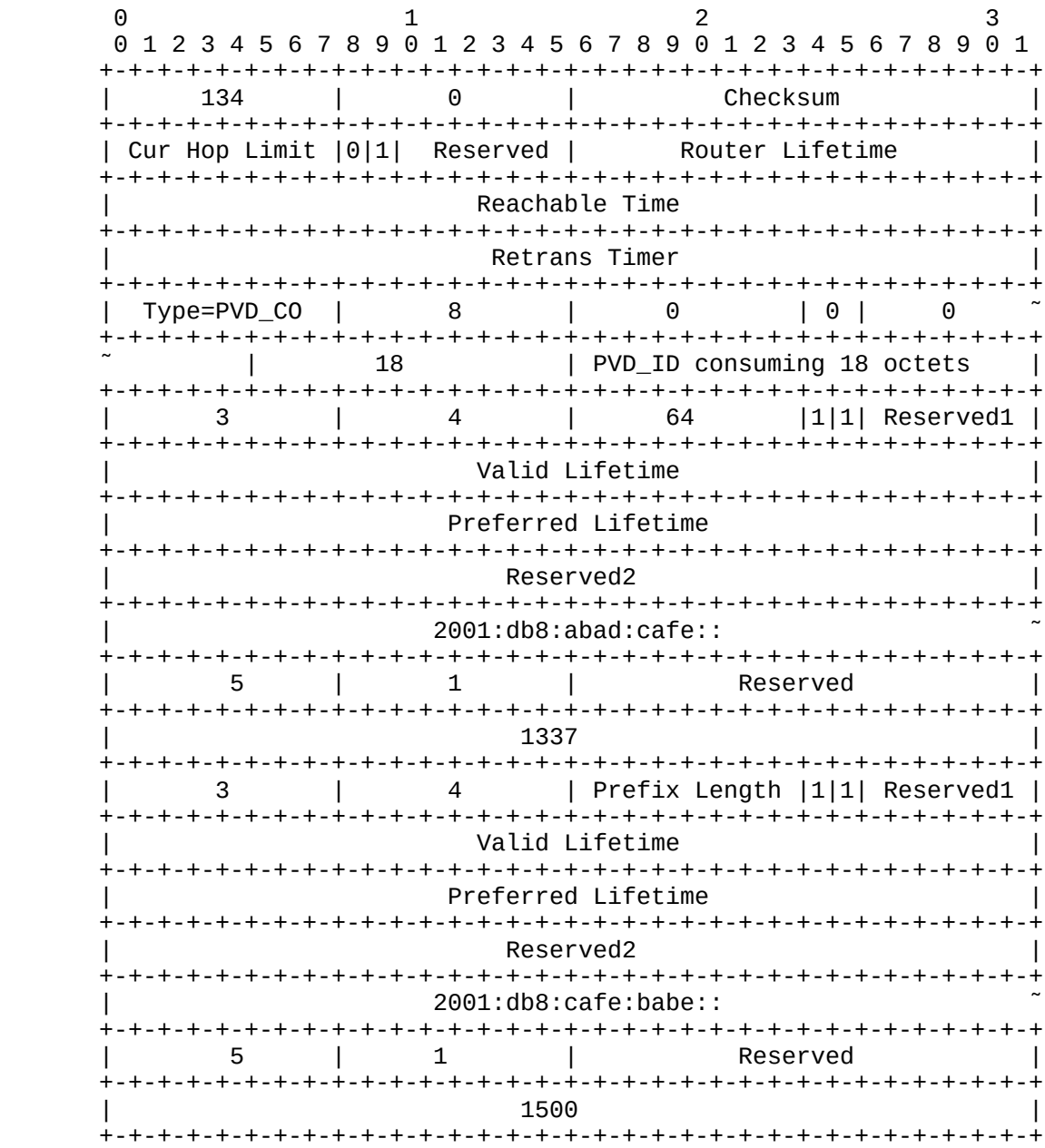


Figure 2: An RA with one implicit PVD and one explicit PVD

Authors' Addresses

Jouni Korhonen
Broadcom Limited
3151 Zanker Road
San Jose, CA 95134
USA

Email: jouni.nospam@gmail.com

Suresh Krishnan
Ericsson
8400 Decarie Blvd.
Town of Mount Royal, QC
Canada

Phone: +1 514 345 7900 x42871
Email: suresh.krishnan@ericsson.com

Sri Gundavelli
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
USA

Email: sgundave@cisco.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: May 04, 2016

E. Kline
Google Japan KK

November 01, 2015

Multiple Provisioning Domains API Requirements
draft-kline-mif-mpvd-api-reqs-00

Abstract

RFC 7556 [RFC7556] provides the essential conceptual guidance an API designer would need to support use of PvDs. This document aims to capture the requirements for an API that can be used by applications that would be considered "advanced", according to section 6.3 [1] of RFC 7556 [RFC7556]. The "basic" [2] and "intermediate" [3] API support levels can in principle be implemented by means of layers wrapping the advanced API.

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 May 04, 2016.

Copyright Notice

Table of Contents

1. Introduction

RFC 7556 [RFC7556] provides the essential conceptual guidance an API designer would need to support use of PvDs. This document aims to capture the requirements for an API that can be used by applications that would be considered "advanced", according to section 6.3 [4] of RFC 7556 [RFC7556]. The "basic" [5] and "intermediate" [6] API support levels can in principle be implemented by means of layers wrapping the advanced API.

This document also attempts to make some of the API implementation requirements more concrete by discussion and example.

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. High level requirements

As described in section 2 [7] of RFC 7556 [RFC7556], a Provisioning Domain ("PvD") is fundamentally a "consistent set of network configuration information." This includes information like:

- the list of participating interfaces
- IPv4 and IPv6 addresses
- IPv4 and IPv6 routes: both default routes and more specifics (such as may be learned via RFC 4191 [RFC4191] Route Information Options ("RIOs"))
- DNS nameservers, search path, et cetera
- HTTP proxy configuration

and undoubtedly many more configuration elements yet to be specified (like metering hints, transmission medium and speed, captive portal URL, et cetera).

This configuration information as a whole may not be able to be learned atomically, may need to be synthesized from multiple sources including administrative provisioning, and cannot be presumed to be unchanging over the lifetime of a node’s association with a given PvD.

In order for an application to make consistent use [8] of a given PvD’s network configuration several requirements are placed upon the API itself and the host operating system providing the API.

2.1. Requirements for an API

At the highest level, the requirements for an API that enables applications to make sophisticated use of multiple PvDs amount to providing mechanisms by which they can:

R1 observe accessible PvDs

It MUST be possible for an application to be informed of the set of all PvDs it can currently access, and to be informed of changes to this set.

R2 observe configuration elements of an accessible PvD

It MUST be possible to learn requested configuration information of any accessible PvD, and to be informed of any changes to the configuration information comprising an accessible PvD.

R3 scope networking functionality to a specified PvD

For every existing API function that interacts with the node's networking stack, be it at a relatively high level like getaddrinfo() [9] or at the level of something like Sockets API's sendmsg(), there MUST be a means by which an application can specify the PvD within which networking operations are to be restricted.

R4 use one and only specified scope per networking functionality invocation

For every unique invocation of a networking API function, there MUST only be one specified PvD to which networking functionality is to be restricted. At any given point in an application's lifetime there MAY be several encapsulating layers of unspecified PvDs (Section 3.2) through which the implementation must progressively search to find a specified PvD, but ultimately a networking function MUST use one and only one PvD for its operations, even if that PvD is a "null PvD" (Section 3.3).

R5 make consistent use of programmatic references to PvDs

For uniformity and simplicity, every PvD-aware API functional element SHOULD use (as return values of function calls, function arguments, et cetera) the same programmatic reference for PvDs, e.g. a construct containing a PvD identifier [10] or some equivalent shorthand reference token (see Section 4.6 for a discussion of implementation considerations). Regardless of the implementation strategy chosen, a given programmatic reference MUST remain constant over the lifetime of the node's continuous attachment to the PvD to which it refers (until a disconnection or disassociation event occurs). Additionally, references MAY change with successive re-associations to the same PvD whereas PvD identifiers, by definition, will not.

It is important to note that there is always a provisioning domain within which networking functionality is scoped. For simply-

connected hosts this may be the implicit PvD [11] created by a single networking interface connected to a traditional, shared LAN segment. For multihomed hosts the "default provisioning domain" is likely a matter of policy, but MAY be a "null" PvD, i.e. one completely devoid of networking configuration information (no addresses, no routes, et cetera). See Section 3 for further discussion.

The utility of such an API (allowing applications to learn of and control the scope of networking functionality) suggests that the Provisioning Domain is perhaps a more useful operational definition for the original IPv6 concept of a "site-local scope" than the ill-fated [RFC3879], "ill-defined concept" [12] of a site. It also suggests one possible way by which operating system support for a PvD-aware API might be implemented.

2.2. Requirements for supporting operating systems

The multiple PvD model of host behaviour is perhaps closer to the Strong End System Model than the Weak End System Model characterized in RFC 1122 [RFC1122] section 3.3.4.2 [13], but owing to its recognition of a many-to-many relationship between interfaces and PvDs should be considered a unique model unto itself.

In the PvD-aware End System Model, the "two key requirement issues related to multihoming" are restated as:

a. A host MAY silently discard an incoming datagram whose destination address does not correspond to any PvD associated with the physical (or virtual) interface through which it is received.

b. A host MUST restrict itself to sending (non-source-routed) IP datagrams only through the physical (or virtual) interfaces that correspond to the PvD associated with the IP source address of the datagrams.

In order to support a PvD-aware application’s use of multiple PVDs, several additional requirements must be met by the host operating system, especially when performing functions on behalf of applications or when no direct application intervention is possible, as discussed in the following sections.

2.2.1. Source address selection

Whenever a source address is to be selected on behalf of an application it is essential for consistent use that only source addresses belonging to the specified PvD be used a candidate set. (See RFC 6418 [RFC6418] section 3.5 [14] for references to issues arising from poor source address selection.)

For nodes following the PvD-aware End System Model, RFC 6724 [RFC6724] section 4 [15] is amended as follows:

R6 The candidate source addresses MUST be restricted to the set of unicast addresses associated with the concurrently specified PvD.

Additionally, source address selection policies from PvDs other than the concurrently specified PvD MUST NOT be applied.

2.2.2. Route isolation

Whenever a routing lookup for a given destination is to be performed, it is essential that only routes belonging to the currently specified PvD be consulted. Applications and libraries that use the inherent routing reachability check (and subsequent source address selection) performed during something like the Sockets API connect() call on a UDP socket to learn reachability information cheaply cannot function correctly otherwise. RFC 6418 [RFC6418] section 4.2 [16] contains more discussion and references to issues arising from insufficiently isolated routing information.

For nodes following the PvD-aware End System Model:

R7 The set of routes consulted for any routing decision MUST be restricted to the routes associated with the concurrently specified PvD.

2.2.3. Automatic PvD metadata marking

In many cases, an application can examine a source address or the destination address of a received datagram and use that address’s association with a given PvD to learn, for example, the PvD with which an incoming connection may be associated. It may, however, be impossible for an application to make this determination on its own if, for example, an incoming TCP connection is destined to a RFC 1918 [RFC1918] address that happens to be configured in multiple PvDs at the same time. In such circumstances, the supporting operating system will need to provide additional assistance.

For nodes following the PvD-aware End System Model:

R8 When performing networking functionality on behalf of an application, the supporting operating system MUST record and make available to the application either (1) all the information the application might need to make a determination of the applicable PvD on its own or (2) the API's PvD programmatic reference directly.

A supporting operating system SHOULD record and make available the API's PvD programmatic reference; other approaches invite ambiguity among applications' interpretation of available information.

2.2.4. Additional system and library support

Frequently, operating systems have several additional supporting libraries and services for more advance networking functionality. Using the system's own PvD API, and fulfilling the above requirements, it should be possible to extend these services to provide correct per-PvD isolation of information and enable consistent application use of PvDs.

3. Conceptual PvDs

3.1. The 'default' PvD

Because there is always one specified provisioning domain to which an individual invocation of networking functionality is restricted (Section 2.1) there must necessarily exist a system "default PvD". This provisioning domain is the one which networking functionality MUST use when no other specified PvD can be determined.

Using the system's default PvD enables support of basic [17] uses of the PvD API (i.e. backward compatibility for unmodified applications).

The operating system MAY change the default PvD accordingly to policy. It is expected that nodes will use a variety of information, coupled with administrative policy, to promote one of any number of concurrently available PvDs to be the system's default PvD.

R9 A PvD-aware API implementation MUST include a mechanism for applications to learn the programmatic reference to the system's concurrent default PvD.

R10 A PvD-aware API implementation SHOULD contain a mechanism enabling an application to be notified of changes to the concurrent default PvD in a comparatively efficient manner (i.e. more efficient than polling).

3.2. The ’unspecified’ PvD

An application may at some times wish to be specific about which PvD should be used for networking operations and at other times may prefer to defer the choice of specific PvD to one specified elsewhere (including the system default PvD).

For example, if an application has specified the PvD to be used for all functions called by its process and child processes (Section 4.3), it may indicate that certain invocations should instead use the system default PvD by using a programmatic reference to the "unspecified PvD".

R11 API implementors MUST reserve a programmatic reference to represent an "unspecified PvD": an indication that the application defers the selection of a specific PvD.

R12 When invoked without a specific PvD, or with a programmatic reference to the "unspecified PvD", networking functionality MUST find a specific PvD to be used by examining the successive encapsulating layers of possible specificity supported by the API (Section 4.3), e.g. look first for a "fiber-specific default" PvD, then a "thread-specific default" PvD, a "process-specific default" PvD, and ultimately use the system’s default PvD if no other specified PvD can be found.

3.3. The ’null’ PvD

If there are no PvDs accessible to an application, whether as a matter of policy (insufficient privileges) (Section 4.5) or as a matter of natural circumstance (the node is not connected to any network), the construct of a ’null’ PvD may be useful to ensure networking functions fail (and fail quickly).

R13 API implementors MAY reserve a programmatic reference to represent a "null PvD": an unchanging provisioning domain devoid of any and all networking configuration information.

It is possible for operating systems to enforce that only PvD-aware applications may function normally by administratively configuring the default PvD to be the "null PvD".

3.4. The ’loopback’ PvD

TBD: is it useful to have a "loopback" PvD, i.e. one consisting solely of all addresses configured on the node and all locally delivered routes?

4. Requirements for new API functionality

4.1. Learning PvD availability

R14 A PvD-aware API MUST implement a mechanism whereby an application can receive a set of the API's PvD programmatic references representing the complete set of PvDs (both explicit [18] and implicit [19]) with which the node is currently associated.

R15 A PvD-aware API implementation SHOULD contain a mechanism enabling an application to be notified of changes in the above set of actively associated PvDs in a comparatively efficient manner (i.e. more efficient than polling).

In may also be of use to applications to receive notifications of pending changes to the set of currently connected PvDs. For example, if it is known that a connection to a PvD is scheduled to be terminated shortly, an application may be able to take some appropriate action (migrate connections to another PvD, send notifications, et cetera).

4.2. Learning network configuration information comprising a PvD

R16 A PvD-aware API MUST include a mechanism whereby by an application, using the API's PvD programmatic reference, can receive elements of the network configuration information that comprise a PvD. At a minimum, this mechanism MUST be capable of answering queries for:

* the PvD identifier
* all participating interfaces
* all IPv4 and all non-deprecated IPv6 addresses
* all configured DNS nameservers

A PvD's network configuration information is neither guaranteed to be learned atomically nor is it guaranteed to be static. Addresses, routes, and even DNS nameservers and participating interfaces may each change over the lifetime of the node's association to a given PvD. Timely notification of such changes may be of particular importance to some applications.

R17 A PvD-aware API implementation SHOULD contain a mechanism enabling an application to be notified of changes in the networking configuration information comprising a PvD in a comparatively efficient manner (i.e. more efficient than polling).

R18 A network configuration query API implementation SHOULD take extensibility into account, to support querying for configuration information not yet conceived of with minimal adverse impact to applications.

4.3. Scoping functionality to a specific PvD

R19 A PvD-aware API implementation MUST include a mechanism for an application to specify the programmatic reference of the PvD to which all networking functionality MUST be restricted when not otherwise explicitly specified (a configurable, application-specific "default PvD").

R20 The API implementation MUST support setting such a "default PvD" for an application’s entire process (and by extension its child processes). Additionally, the API SHOULD support an application setting a "default PvD" at every granularity of "programming parallelization", i.e. not only per-process, but also per-thread, per-fiber, et cetera. At every supported layer of granularity, if no PvD reference has been set the next coarser layer’s setting MUST be consulted (up to and including the system’s default PvD) when identifying the specified PvD to be used.

R21 For every degree of granularity at which an application may specify a "default PvD" there MUST exist a corresponding mechanism to retrieve any concurrently specified implementation-specific PvD programmatic reference. If no PvD has been specified for at the granularity of a given query, the "unspecified PvD" must be returned.

With access to this functionality it is possible to start non-PvD-aware applications within a single PvD context with no adverse impact. Furthermore, with judicious use of a sufficiently granular API, existing general purpose networking APIs can be wrapped to appear PvD-aware.

4.4. Explicit versus Implicit PvDs

R22 Because programmatic references to PvDs are returned for both explicit and implicit PvDs, the MPvD API implementation MUST be equally applicable and useful for any valid type of PvD; it MUST NOT be necessary for a PvD-aware application to distinguish between explicit and implicit PvDs to function properly.

4.5. Policy restrictions

This document does not make recommendations about policies governing the use of any or all elements of a PvD API, save only to note that some restrictions on use may be deemed necessary or appropriate.

R23 A PvD API implementation MAY implement policy controls whereby access to PvD availability information, configuration elements, and/or explicit scoping requests is variously permitted or denied to certain applications.

4.6. Programmatic reference implementation considerations

PvD identifiers may be of a length or form not easily handled directly in some programming environments, and unauthenticated PvD identifiers are assumed to be only probabilistically unique [20]. As such, API implementations should consider using some alternative programmatic reference (a node-specific "handle" or "token"), which is fully under the control of the operating system, to identify an instance of a single provisioning domain’s network configuration information.

Even though a PvD identifier may uniquely correspond to, say, a network operator, there is no guarantee that the configuration information (delegated prefixes, configured IP addresses, and so on) will be the same with every successive association to the same PvD identifier. An implementation may elect to change the value of the programmatic reference to a given PvD identifier for each temporally distinct association. Doing so presents some advantages worth considering:

Collisions in the PvD identifier space will inherently be treated as distinct by applications not concerned solely with identifiers.

Changing the value of a reference can disabuse application writers of inappropriately caching configuration information from one association instance to another.

Whether two PvDs are "identical" is perhaps better left to applications to decide since "PvD equivalence" for a given application may alternatively be determined by successfully accessing some restricted resource.

This document makes no specific requirement on the type of programmatic reference used by the API.

5. Existing networking APIs

5.1. Updating existing APIs

From the perspective of a PvD-aware operating system, all previously existing non-PvD-enabled networking functionality had historically been executed within the context of a single, implicit provisioning domain. A sufficiently granular API to specify which PvD is to be used to scope subsequent networking functionality (Section 4.3) can be used to wrap non-PvD-aware APIs, giving them this new PvD-aware capability. However,

R24 Operating system implementors SHOULD consider updating existing networking APIs to take or return programmatic references to PvDs directly.

This may mean creating new functions with an additional PvD programmatic reference argument, adding a PvD programmatic reference field to an existing structure or class that is itself an argument or return type, or finding other means by which to use a programmatic reference with minimal or no disruption to existing applications or libraries.

5.2. Requirements for name resolution APIs

RFC 3493 [RFC3493] getaddrinfo() [21] and getnameinfo() [22] APIs deserve explicit discussion. Previously stated requirements make it clear that it MUST be possible for an application to perform normal name resolution constrained to the DNS configuration within a specified PVD. This MUST be possible using at least the techniques of Section 4.3.

The following additional requirements are places on PvD-aware implementations of these functions:

R25 All DNS protocol communications with a PvD’s nameservers MUST be restricted to use only source addresses and routes associated with the PvD.

R26 If getaddrinfo() is called with the AI_ADDRCONFIG flag specified, IPv4 addresses shall be returned only if an IPv4 address is configured within the specified provisioning domain and IPv6 addresses shall be returned only if an IPv6 address is configured within the specified provision domain. The loopback address is (still) not considered for this case as valid as a configured address.

6. Acknowledgements

The core concepts presented in this document were developed during the Android multinetworking effort by Lorenzo Colitti, Robert Greenwalt, Paul Jensen, and Sreeram Ramachandran.

Additional thanks to the coffee shops of Tokyo.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

An important new security impact of a PvD-aware API is that it becomes much simpler (by design) to write a well-functioning application to create a bridging data path between two PvDs that would not otherwise have been so easily connected.

For some operating systems, existing APIs already make this bridging possible, though some functionality like DNS resolution may have been difficult to implement. Indeed, the very aim of an MPvD API is to make implementing a PvD-aware application simple and to make its functioning more "correct" ("first class" support for such functionality).

Operating system implementations have several points of potential policy control including:

- use of certain PvDs MAY be restricted by policy (e.g. only approved users, groups, or applications might be permitted access), and/or
- use of more than one PvD (or the MPvD API itself) MAY be similarly restricted.

9. References

9.1. Normative References

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <http://www.rfc-editor.org/info/rfc1122>.

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

[RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, DOI 10.17487/RFC3493, February 2003, <http://www.rfc-editor.org/info/rfc3493>.

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <http://www.rfc-editor.org/info/rfc6724>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <http://www.rfc-editor.org/info/rfc7556>.

9.2. Informative References

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[RFC3879] Huitema, C. and B. Carpenter, "Deprecating Site Local Addresses", RFC 3879, DOI 10.17487/RFC3879, September 2004, <http://www.rfc-editor.org/info/rfc3879>.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <http://www.rfc-editor.org/info/rfc4191>.

[RFC6418] Blanchet, M. and P. Seite, "Multiple Interfaces and Provisioning Domains Problem Statement", RFC 6418, DOI 10.17487/RFC6418, November 2011, <http://www.rfc-editor.org/info/rfc6418>.

Author’s Address

Erik Kline
Google Japan KK
6-10-1 Roppongi
Mori Tower, 44th floor
Minato, Tokyo 106-6126
JP

Email: ek@google.com

MIF Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 17, 2016

M. Stenberg
S. Barth
Independent

October 15, 2015

Multiple Provisioning Domains using Domain Name System
draft-stenberg-mif-mpvd-dns-00

Abstract

This document describes a mechanism to transmit and secure provisioning domain information for IPv6 and IPv4 addresses by using reverse DNS resolution. In addition it specifies backwards-compatible extensions to IPv6 host configuration to support special-purpose global IPv6 prefixes which can only be used to access certain isolated services.

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 April 17, 2016.

Copyright Notice

Table of Contents

1. Introduction

Given multiple address prefixes or multiple interfaces, hosts require more information to make educated choices about the interfaces and addresses to use. [RFC7556] describes the provision domains (PVDs) that provide the additional information the hosts need.

This document describes where and how the provision domain information is encoded in the Domain Name System (DNS). For optional authentication DNSSEC is used.

A backwards compatible way of adding IPv6 prefixes without generic internet connectivity is also provided so that the hosts that are not aware of the provisioning domain prefixes do not inadvertly use those for general network access.

2. Terminology

PVD	a provisioning domain, usually with a set of provisioning domain information; for more information, see [RFC7556].
special-purpose IPv6 prefix	a global IPv6 source prefix that cannot be used to reach the public IPv6 internet but instead only allows access to certain special services (e.g., VoIP, IPTV).

2.1. Requirements Language

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

3. PVD information discovery using DNS

Each PVD is stored within the DNS, encoded as a single TXT record [RFC1035]. Section 3.1 describes the syntax and Section 3.2 the semantics of the enclosed data.

To find the per-PVD TXT records that apply to a source address, the host queries the DNS for PTR records of the domain _pvd.<domain>. <domain> is a .arpa domain for reverse lookups derived from the respective prefix or subnet the source address is assigned from and generated as specified in [RFC3596] for IPv6 addresses and [RFC1034] for IPv4 addresses respectively. If the query returned any PTR records the host then subsequently queries the DNS for TXT records located in each domain indicated in the PTR records and handles their contents as individual PVDs.

As prefixes can be sub-delegated arbitrarily, PTR records SHOULD be provided for any subprefixes contained within a particular prefix. For example, given a prefix 2001:db8:8765:4321::/64, a host with an address of 2001:db8:8765:4321:1234:5678:abcd:beef queries for PTR records in _pvd.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.2.3.4.5.6.7.8.8.b.d .0.1.0.0.2.ip6.arpa However, if the address is assigned from 2001:db8:8765:4321:1234::/80, the request would be made for _pvd.0.0. 0.0.0.0.0.0.0.0.0.0.4.3.2.1.1.2.3.4.5.6.7.8.8.b.d.0.1.0.0.2.ip6.arpa instead.

If for example, the retrieved PTR record is assumed to point at the FQDN _pvd.example.com. The next query is then sent for TXT records in this domain, and if successful, the node has retrieved up-to-date information about PVDs applicable to that particular address.

A host MUST support handling multiple PTR records for the initial .arpa domain as well as multiple TXT records for all domains pointed to by the PTR records. This facilitates handling of multiple PVDs with minimal amount of state in the network. A host MUST honor both the time-to-live of the received records, and negative replies that conform to [RFC2308]. A host MUST NOT use addresses from a prefix as the source for new packet flows once the TTL has passed until it did successfully retrieve updated PVD information.

3.1. PVD TXT Record Fomat

PVD information within DNS is encoded using TXT records, similar to those of DNS-SD [RFC6763] and defined as follows. TXT records consist of key/value pairs, each encoded as a string of up to 255 octets preceded by a length byte storing the number of octets. The strings are in the form "key=value" or simply "key" (without quotation marks) where everything up to the first ’=’ character (if any, otherwise the whole string) is the key and everything after it (if any, including subsequent ’=’ characters) is the value. Due to the use of a length byte, quotation marks or similar are not required around the value. Keys are case-sensitive. Hosts MUST ignore TXT records which do not conform to these rules.

3.2. PVD TXT Record Keys

The following keys are defined to be used inside PVD TXT records. Unknown keys inside PVD Information MUST be ignored.

3.2.1. Reachable Services

The following set of keys can be used to specify the set of services for which the respective PVD should be used. If present they MUST be honored by the client, i.e., if the PVD is marked as not usable for internet access it MUST NOT be used for internet access, if the usability is limited to a certain set of domain or address prefixes, then a different PVD MUST be used for other destinations.

Key	Description	Value	Example
n	User-visible service name	human-readable UTF-8 string	n=Foobar Service
s	Internet inaccessible	(none)	s
z	DNS zones accessible	comma-separated list of DNS zone	z=foo.com,sub.bar.com
6	IPv6-prefixes accessible	comma-separated list of IPv6 prefixes	6=2001:db8:a::/48,2001:db8:b:c::/64
4	IPv4-prefixes accessible	comma-separated list of IPv4 prefixes in CIDR	4=1.2.3.0/24,2.3.0.0/16

3.2.2. DNS Configuration

The following set of keys can be used to specify the DNS configuration for the respective PVD. If present, they MUST be honored and used by the client whenever it wishes to access a resource of the PVD.

Key	Description	Value	Example
r	Recursive DNS server	comma-separated list of IPv6 and IPv4 addresses	r=2001:db8::1,192.0.2.2
d	DNS search domains	comma-separated list of search domains	d=foo.com,sub.bar.com

3.2.3. Connectivity Characteristics

The following set of keys can be used to signal certain characteristics of the connection towards the PVD.

Key	Description	Value	Example
bw	Maximum achievable bandwidth	1 symmetrical or 2 comma-separated ingress, egress values in kilobits per second	bw=5000 or bw=1000,100
lt	Minimum achievable latency	1 symmetrical or 2 comma-separated ingress, egress values in milliseconds	lt=20 ot lt=20,100
rl	Maximum achievable reliability	1 symmetrical or 2 comma-separated ingress, egress values in 1/1000	rl=1000 or rl=900,800
tm	Traffic metered (cut-off / limited over threshold)	(none) or traffic threshold in kilobytes	tm or tm=1000000
cp	Captive portal	(none)	cp
nat	IPv4 NAT in place	(none)	nat

3.2.4. Private Extensions

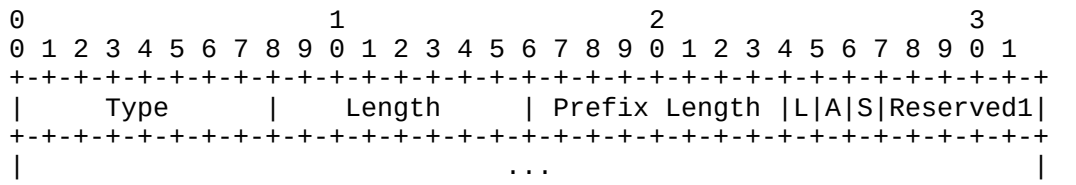
keys starting with "x-" are reserved for private use and can be utilized to provide vendor-, user- or enterprise-specific information. It is RECOMMENDED to use one of the patterns "x-FQDN-KEY[=VALUE]" or "x-PEN-KEY[=VALUE]" where FQDN is a fully qualified domain name or PEN is a private enterprise number [PEN] under control of the author of the extension to avoid collisions.

4. Special-purpose IPv6 prefixes

A service provider might wish to assign certain global unicast address prefixes which can be used to reach a limited set of services only. In the presence of legacy hosts it must be ensured however that these prefixes are not mistakenly used as source addresses for other destinations. This section therefore defines optional extensions to NDP [RFC4861], DHCPv6 [RFC3315] and DHCPv6-PD [RFC3633] to indicate this state.

4.1. Extensions to Stateless Address Autoconfiguration

NDP [RFC4861] defines the Prefix Information option to announce prefixes for stateless address configuration. The "A-bit" is set, whenever hosts may autonomously derive addresses from a given prefix. For special-purpose prefixes this document defines the first bit of the Reserved1-field as the "S-Bit".



The following additional requirements apply to hosts intending to support global special-purpose IPv6 prefixes:

Upon reception of a Prefix Information Option with the S-Bit set, it should behave as if the A-Bit was set, however (unless the A-Bit was actually set by the sending router) it MUST delay using any addresses derived from the prefix until it has queried, retrieved and honored (see Section 3) at least all mandatory provisioning domain information related to the given prefix.

A host SHOULD NOT interpret the S-Bit being clear as an indicator that no provisioning domain information is available for a given prefix.

The following additional requirements apply to routers:

A router MUST NOT set the A-Bit for global unicast address prefixes which cannot be used to reach the public IPv6 internet.

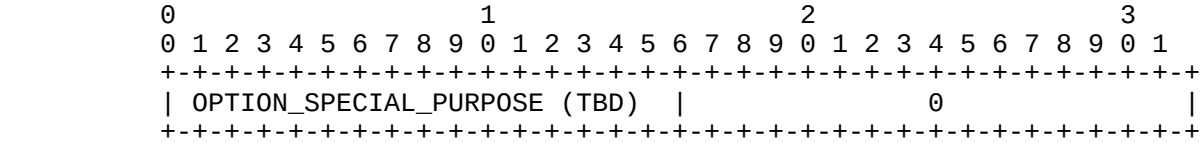
A router SHOULD use the S-Bit to indicate that PVD-aware hosts can statelessly assign themselves addresses from a given prefix. It MAY use the S-Bit in addition to the A-Bit to indicate that a prefix usable to reach the public IPv6 internet has additional (optional) provisioning domain information.

A router announcing one or more Prefix Information options with the S-Bit set MUST also announce one or more recursive DNS servers using a Recursive DNS Server Option [RFC6106]. If none of the Prefix Information Options it announces have the A-Bit set then at least one of these recursive DNS servers MUST be reachable using a link-local address.

4.2. Extensions to DHCPv6

Using DHCPv6 [RFC3315] and DHCPv6-PD [RFC3633] servers can actively decide which addresses or prefixes are assigned to clients and requesting routers, however a mechanism is needed to distinguish PVD-aware devices and in the same sense PVD-aware devices need to be able to detect which prefixes and addresses are special-purpose. Therefore, a zero-length DHCPv6 option OPTION_SPECIAL_PURPOSE is

defined to be added as a suboption to OPTION_IAADDR and OPTION_IAPREFIX options.



The following additional requirements apply to clients and requesting routers intending to support global special-purpose IPv6 prefixes via DHCPv6:

> A client or requesting router MUST include the option code OPTION_SPECIAL_PURPOSE in an option OPTION_ORO in its SOLICIT and REQUEST messages, whenever it wishes to accept special-purpose prefixes in its identity associations.
>
> Upon reception of an OPTION_IAADDR or OPTION_IAPREFIX option having an embedded OPTION_SPECIAL_PURPOSE option it MUST delay using any addresses derived from the prefix as source address for its own communication until it has queried, retrieved and honored (see Section 3) at least all mandatory provisioning domain information related to the given prefix or address. If it is a requesting router, it MAY however subdelegate prefixes or assign addresses from special-purpose prefixes to clients without doing so as long as the requirements in the following paragraph are honored.

The following additional requirements apply to routers assigning addresses from or delegating (parts of) special-purpose prefixes using DHCPv6:

> A router MUST include a zero-length suboption of type OPTION_SPECIAL_PURPOSE in every OPTION_IAADDR and OPTION_IAPREFIX option it assigns or delegates containing a global unicast address or prefix which cannot be used to reach the public IPv6 internet. It MUST NOT assign or delegate such an address or prefix to a client or requesting router not including the option code of OPTION_SPECIAL_PURPOSE inside an OPTION_ORO option.
>
> A router announcing one or more addresses or prefixes with an embedded OPTION_SPECIAL_PURPOSE option MUST also announce one or more recursive DNS servers using a OPTION_DNS_SERVERS option [RFC3646]. If all of the addresses in a DHCPv6 reply carry the embedded OPTION_SPECIAL_PURPOSE option then at least one of the announced recursive DNS servers MUST be reachable using a link-local address.

5. Security Considerations

The security implications of using PVDs in general depend on two factors: what they are allowed to do, and whether or not the authentication and authorization of the PVD information received is sufficient for the particular usecase. As the defined scheme uses DNS for retrieval of the connection parameters, the retrieval of both the PTR and the TXT records should be secured, if they are to be trusted. The PVD information allows for the following types of attacks:

- Traffic redirection, both by providing custom DNS server, as well as actual potentially different next-hop and/or source address selection.

- Faking of connection capabilities to e.g. prefer some connection fraudulently over others.

If a host requires DNSSEC authentication and the retrieved information is not sufficiently secured, they MUST be ignored as the defined way of using them in Section 3.2 requires honoring the supplied information.

Security properties of NDP and DHCPv6 are inherited for the respective extensions, therefore relevant sections of [RFC4861] and [RFC3315] should be consulted. In any case, signaling addresses and prefixes to be special-purpose does not have a significant impact on the underlying assignment and delegation mechanisms.

6. IANA Considerations

IANA is requested to setup a PVD DNS TXT Record Key registry with the initial types: s, z, 6, 4 (Section 3.2.1); r, d (Section 3.2.2); bw, lt, rl, tm, cp, nat (Section 3.2.3) and a prefix x- (Section 3.2.4) for Private Use [RFC5226]. The policy for further additions to the registry is requested to be RFC Required [RFC5226].

This document defines a new bit for the NDP Prefix Information Option (the S-bit). There is currently no registration procedure for such bits, so IANA should not take any action on this matter.

IANA should assign a DHCPv6 option code OPTION_SPECIAL_PURPOSE to the DHCPv6 option code space defined in [RFC3315].

7. References

7.1. Normative references

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", RFC 2308, DOI 10.17487/RFC2308, March 1998, <http://www.rfc-editor.org/info/rfc2308>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, DOI 10.17487/RFC3633, December 2003, <http://www.rfc-editor.org/info/rfc3633>.

[RFC3646] Droms, R., Ed., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, DOI 10.17487/RFC3646, December 2003, <http://www.rfc-editor.org/info/rfc3646>.

[RFC6106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 6106, DOI 10.17487/RFC6106, November 2010, <http://www.rfc-editor.org/info/rfc6106>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <http://www.rfc-editor.org/info/rfc7556>.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, DOI 10.17487/RFC3596, October 2003, <http://www.rfc-editor.org/info/rfc3596>.

7.2. Informative references

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <http://www.rfc-editor.org/info/rfc6763>.

[PEN] IANA, "Private Enterprise Numbers", <https://www.iana.org/assignments/enterprise-numbers>.

Appendix A. This solution compared to doing this in DHCPv6/NDP [RFC Editor: please remove]

The angle of attack of the MIF work to date (autumn 2015) has been to add container options and their transfer mechanisms to DHCPv6 and NDP. This document details a different approach, and therefore it is sensible to compare it to to the existing solutions in terms of (highly subjective) pros and cons. The authors consider pros of this proposal to be:

- No overhead for hosts that do not care (possibly most; no spurious RA options, ...)
- No problems with relaying data; if the first-hop device does not care, DNS requests propagate onward.
- Little/no changes to DHCP, DHCPv6, DHCPv6-PD or RA.
- Much more scalable; no worries about multicast packet size limits.
- No duplication of specifications / TLVs for DHCP, DHCPv6 and RA.
- Solves m:n prefix <-> PVD elegantly: no need to either duplicate applying prefix for each PVD or duplicate each PVD for each applying prefix.

- Easily extensible (TXT records, no TLV definitions, parsing and generation necessary)
- Probably not affected by IPR on draft-ietf-mif-mpvd-dhcp-support
- Reuses the existing reverse DNS infrastructure

The authors consider cons of this proposal to be:

- This scheme requires DNS servers ’close’ on the path to the user, if changed information is to be sent; otherwise centralized solution would work (with some synthesized records).
- Security using either DNSSEC or in-band hashes is rather painful (but possibly not more than the scheme in the current DHCP/RA drafts), so the default would most likely be insecure. That is not much different from DHCP*/RA, which are also 99.999...% of the time not secured.

Appendix B. Discussion Points [RFC Editor: please remove]

- Besides special purpose prefixes, it might be desirable to have special purpose routers which only provide access to certain services but not the entire internet. These services could be announced by only using more-specific routes, however if the destination addresses are possibly changing, extension of the RIO mechanism might be needed. One possibility would be to add a new RIO S-flag with semantics like: "When the host receives a Route Information Option with the S-Bit set, it MUST ignore the value in the Prf-field (even if it is (10)) and instead assume the preference to have a value greater than (11). However, it MUST only use the route for packets having a source prefix announced by the same router.". This would allow selective routes (Prf=(10)) only applying to MIF-hosts.
- DNSSEC delegation of reverse zones might be difficult in some cases. It is debatable, whether we want a complementary in-band signing mechanism as well, e.g., pre-shared public keys associated the domain name of the TXT records and "sig-X=..." keys (where X identifies the specific key) and ... is an EdDSA or ECDSA signature over all records not starting with "sig-". Care would need to be taken wrt. TTL and negative caching though.
- Should PVD-aware hosts be recommended or even required to always prefer routers that announced the respective source address in a PIO over those that didn’t when making routing decisions? This takes on the points made in draft-baker-6man-multi-homed-host.

Appendix C. Changelog [RFC Editor: please remove]

draft-stenberg-mif-mpvd-dns-00:

- Initial version.

Appendix D. Draft Source [RFC Editor: please remove]

As usual, this draft is available at https://github.com/fingon/ietf-drafts/ in source format (with nice Makefile too). Feel free to send comments and/or pull requests if and when you have changes to it!

Appendix E. Acknowledgements

Thanks to Eric Vyncke for the original idea of using DNS for transmitting PVD information.

Authors' Addresses

Markus Stenberg
Independent
Helsinki 00930
Finland

Email: markus.stenberg@iki.fi

Steven Barth
Independent
Halle 06114
Germany

Email: cyrus@openwrt.org