### Multiple Provisioning Domain Architecture
### draft-ietf-mif-mpvd-arch-00

Abstract

   This document is a product of the work of MIF architecture design
   team.  It outlines a solution framework for some of the issues,
   experienced by nodes that can be attached to multiple networks.  The
   framework defines the notion of a Provisioning Domain (PVD) - a
   consistent set of network configuration information, and PVD-aware
   nodes - nodes which learn PVDs from the attached network(s) and/or
   other sources and manage and use multiple PVDs for connectivity
   separately and consistently.

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 04, 2014.

Table of Contents

## 1.  Introduction

Nodes attached to multiple networks may encounter problems due to
conflict of the networks configuration  and/or simultaneous use of
the multiple available networks.  While existing implementations
apply various techniques ([RFC6419]) to tackle such problems, in many
cases the issues may still appear.  The MIF problem statement
document [RFC6418] describes the general landscape as well as
discusses many specific issues and scenarios details.

Problems, enumerated in [RFC6418], can be grouped into 3 categories:

1.  Lack of consistent and distinctive management of configuration
    elements, associated with different networks.

2.  Inappropriate mixed use of configuration elements, associated
    with different networks, in the course of a particular network
    activity / connection.

3.  Use of a particular network, not consistent with the intent of
    the scenario / involved parties, leading to connectivity failure
    and / or other undesired consequences.

An example of (1) is a single node-scoped list of DNS server IP
addresses, learned from different networks, leading to failures or
delays in resolution of names from particular namespaces; an example
of (2) is use of an attempt to resolve a name of a HTTP proxy server,
learned from a network A, with a DNS server, learned from a network
B, that is likely to fail; an example of (3) is use of an employer-
sponsored VPN connection for peer-to-peer connectivity, unrelated to
employment activities.

This architecture describes a solution to these categories of
problems, respectively, by:

1.  Introducing a formal notion of the PVD, including PVD identity,
    and ways for nodes to learn the intended associations among
    acquired network configuration information elements.

2.  Introducing a reference model for a PVD-aware node, preventing
    inadvertent mixed use of the configuration information, which may
    belong to different PVDs.

3.  Providing recommendations on PVD selection based on PVD identity
    and connectivity tests for common scenarios.

## 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.  Definitions and types of PVDs

   Provisioning Domain: a consistent set of network configuration
   information.  Classically, the entire set available on a single
   interface is provided by a single source, such as network
   administrator, and can therefore be treated as a single provisioning
   domain.  In modern IPv6 networks, multihoming can result in more than
   one provisioning domain being present on a single link.  In some
   scenarios, it is also possible for elements of the same domain to be
   present on multiple links.

   Typical examples of information in a provisioning domain, learned
   from the network, are: source address prefixes that can be used by
   connections within the provisioning domain, IP address of DNS server,
   name of HTTP proxy server if available, DNS suffixes associated with
   the network etc.

   PVD-aware node: a node that supports association of network
   configuration information into PVDs, and using these PVDs to serve
   requests for network connections in ways, consistent with
   recommendations of this architecture.

2.1.  Explicit PVDs

   A node may receive explicit information from the network and/or other
   sources, about presence of PVDs and association of particular network
   information with a particular PVD.  PVDs, constructed based on such
   information, are referred to in this document as "explicit".

   Protocol changes/extensions will likely be required to support the
   explicit PVDs by IETF-defined mechanisms.  As an example, one could
   think of one or several DHCP options, carrying PVD identity and / or
   its elements.  A different approach could be to introduce a DHCP
   option, which only carries identity of a PVD, while the association
   of network information elements with that identity, is implemented by
   the respective protocols - such as e.g., with a Router Discovery
   [RFC4861] option associating an address range with a PVD.

   Specific, existing or new, features of networking protocols to enable
   delivery of PVD identity and association with various network
   information elements will be defined in companion design documents.

   Link-specific and/or vendor-proprietary mechanisms for discovery of
   PVD information, different from the IETF-defined mechanisms, can be
   used by the nodes separately from or together with IETF-defined
   mechanisms, as long as they allow to discover necessary elements of
   the PVD(s). Another example of a delivery mechanism for PVDs are key
   exchange or tunneling protocols, such as  IKEv2 [RFC5996] that allow
   transporting host configuration information.  In all cases, by

default nodes must ensure that the lifetime of all dynamically

discovered PVD configuration is appropriately limited by the relevant
events - for example, if an interface media state change was
indicated, the previously discovered information may no longer be
valid and needs to be re-discovered or confirmed.

It shall be possible for sources of PVD information to communicate
that some of their configuration elements could be used within a
context of other networks/PVDs.  PVD-aware nodes, based on such
declaration and their policies, may choose to inject such elements
into some or all other PVDs they connect to.

In some network topologies, the network infrastructure elements may
need to advertise multiple PVDs.  The details of how this is done
generally will be defined in the individual companion design
documents.  However, where different design choices are possible, the
choice that requires smaller number of packets shall be preferred for
efficiency.

## 2.2.  Implicit PVDs and incremental adoption of the explicit PVDs

It is likely that for a long time there may be networks which do not
advertise explicit PVD information, since deployment of any new
features in networking protocols is a relatively slow process.

When connected to networks, which don't advertise explicit PVD
information, PVD-aware node shall automatically create separate PVDs
for configuration received on multiple interfaces.  Such PVDs are
referred to in this document as "implicit".

With implicit PVDs, PVD-aware nodes may still provide benefits to
their users as compared to non-PVD aware nodes, by using network
information from different interfaces separately and consistently to
serve network connection requests, following best practices described
in Section 5.

In the mixed mode, where e.g., multiple networks are available on the
link the interface is attached to, and only some of the networks
advertise PVD information, the PVD-aware node shall create explicit
PVDs based on explicitly learned PVD information, and associate the
rest of the configuration with an implicit PVD created for that
interface.

## 2.3.  Relationship between PVDs and interfaces

Implicit PVDs are limited to network configuration information
received on a single interface.  Explicit PVDs, in practice will
often also be scoped to a configuration related to a particular
interface, however per this architecture there is no such requirement
or limitation and as defined in this architecture, explicit PVDs may
include information related to more than one interfaces, if the node

learns presence of the same PVD on those interfaces and the

authentication of the PVD ID meets the level required by the node
policy.

It is an intent of this architecture to support such scenarios among
others.  Hence, it shall be noted that no hierarchical relationship
exists between interfaces and PVDs: it is possible for multiple PVDs
to be simultaneously accessible over one interface, as well as single
PVD to be simultaneously accessible over multiple interfaces.

## 2.4.  PVD identity/naming

For explicit PVDs, PVD ID (globally unique ID, that possibly is
human-readable) is received as part of that information.  For
implicit PVDs, the node assigns a locally generated globally unique
ID to each implicit PVD.

PVD-aware node may use these IDs to choose a PVD with matching ID for
special-purpose connection requests, in accordance with node policy
or choice by advanced applications, and/or to present human-readable
representation of the IDs to the end-user for selection of Internet-
connected PVDs.

A single network provider may operate multiple networks, including
networks at different locations.  In such cases, the provider may
chose whether to advertise single or multiple PVD identities at all
or some of those networks, as it suits their business needs.  This
architecture doesn't impose specific requirements in this regard.

When multiple nodes are connected to the same link, where one or more
explicit PVDs are available, this architecture assumes that the
information about all available PVDs is advertized by the networks to
all the connected nodes.  At the same time, the connected nodes may
have different heuristics, policies and/or other settings, including
configured set of their trusted PVDs, which may lead to different
PVDs actually being used by different nodes for their connections.

Possible extensions, where different sets of PVDs may be advertised
by the networks to different connected nodes, are out of scope of
this document.

## 2.5.  Relationship to dual-stack networks

When applied to dual-stack networks, the PVD definition allows for
multiple PVDs to be created, where each PVD contain information for
only one address family, or for a single PVD that contains
information about multiple address families.  This architecture
requires that accompanying design documents for the PVD-related
protocol changes must support PVDs containing information from

multiple address families.  PVD-aware nodes must be capable of
dealing with both single-family and multi-family PVDs.

For explicit PVDs, the choice of either of the approaches is a policy
decision of a network administrator and/or node user/administrator.
Since some of the IP configuration information that can be learned
from the network can be applicable to multiple address families (for
instance DHCP address selection option [RFC7078]), it is likely that
dual-stack networks will deploy single PVDs for both address
families.

For implicit PVDs, by default PVD-aware nodes shall including
multiple IP families into single implicit PVD created for an
interface.  At the time of writing of this document in dual-stack
networks it appears to be a common practice for configuration of both
address families to be provided by a single source.

A PVD-aware node that provides API to use / enumerate / inspect PVDs
and/or their properties shall provide ability to filter PVDs and/or
their properties by address family.

## 2.6.  Elements of PVD

## 3.  Conveying PVD information using DHCPv6 and Router Advertisements

DHCPv6 and Router Advertisements are the two most common methods of
configuring hosts and they would need to be extended to convey
explicit PVD information.  There are several things that need to be
considered before finalizing a mechanism to augment DHCPv6 and RAs
with PvD information.

## 3.1.  Separate messages or one message

When information from several PVDs is available at the same
configuration source, there are two possibilities regarding how to
send these out.  One way is to send information from different
provisioning domains in separate messages.  The other is to combine
information from several PVDs onto one message.  The latter method
has the advantage of being more efficient but could have issues due
to authentication and authorization issues as well as potential
issues with accommodating common information and information not
tagged with any PVD information.

## 3.2.  Securing the PVD information

DHCPv6 and RAs both provide some form of authentication that ensures
the identity of the source as well as the integrity of the contents
that have been secured.  While this is useful, the authenticity of
the information provides no information whether the configuration
source is actually allowed to provide information from a given PVD.

In order to do be able to do this, there must be a mechanism for the

owner of the PVD to attach some form of authorization token to the
configuration information that is delivered.

## 3.3.  Backward compatibility

The extensions to RAs and DHCPv6 should be defined in such a manner
than unmodified hosts (i.e.  hosts not aware of PvDs) will continue
to function as well as they did before the PvD information got added.
This could imply that some information may need to be duplicated in
order to be conveyed to legacy hosts.  Similarly PvD aware hosts need
to be able to handle legacy configuration sources which do not
provide PvD information.  There are also several initiatives ongoing
that are aimed at adding some form of additional information to
prefixes [refs to draft-bhandari and draft-korhonen] and any new
mechanism should try to consider co-existence with these existing
mechanisms.

## 3.4.  Selective propagation

When a configuration source has information regarding several PvDs it
is not clear whether it should provide information about all of them
to any host that requests info from it.  While it may be reasonable
in some cases, this might become an unreasonable burden once the
number of PvDs starts increasing.  One way to restrict the
propagation of useless information is for the host to select the PvD
information they desire in their request to the configuration source.
One way this could be accomplished is by using an ORO with the PvDs
that are of interest.  The configuration source can then respond with
only the requested information.

By default, a configuration source SHOULD provide information related
to all provisioning domains without expecting the client to select
the PvD(s) it requires.  This is necessary to ensure that hosts that
do not support requesting selective PvD information will continue to
work.  Also note that IPv6 neighbor discovery does not provide any
functionality analogous to the DHCPv6 ORO.

In this case, when a host receives PvD information it does not
require, the information can simply be discarded.  Also, in
constrained networks such as LLNs, the amount of configuration
information needs to be restricted to ensure that the load on the
hosts is bearable while keeping the information identical across all
the hosts.

In case selective propogation is required, some form of PvD discovery
mechanism needs to be specified so that hosts/applications can be
pre-provisioned to request a specific PvD. Alternately, the set of

   PvDs that the network can provide to the host can be propogated to
   the host using RAs or stateless DHCPv6. The discovery mechanism may
   potentially support the discovery of available PvDs on a per-host
   basis.

3.5.  Retracting/updating PvD information

   After the PvD information is provided to the host it may be outdated
   or updated with newer information before the hosts would normally
   request updates.  Thos would require the mchanism to be able to
   update and/or withdraw all (or some subset) of information related to
   a given PvD. For efficiency reasons, there should be a way to specify
   that all the information from the PvD needs to be reconfigured
   instead of individually updating each item associated with the PvD.

3.6.  Conveying configuration information using IKEv2

   Internet Key Exchange protocol version 2 (IKEv2) [RFC5996] [RFC5739]
   is another widely used and a popular method of configuring IP
   information in a host.  In the case of IKEv2 the provisioning domain
   could actually be implicitly learnt from the Identification -
   Responder (IDr) payloads the IKEv2 initiator and the responder inject
   during the IKEv2 exchange.  The IP configuration may depend on the
   named IDr.  Another possibility could be adding specific provisioning
   domain identifying payload extensions to IKEv2. All of the
   considerations listed above for DHCPv6 and RAs potentialy apply to
   IKEv2 as well.

4.  Example network configurations and number of PVDs

5.  Reference model of PVD-aware node

5.1.  Constructions and maintenance of separate PVDs

   It is assumed that normally, configuration information contained in a
   single PVD, shall be sufficient for a node to fulfill a network
   connection request by an application, and hence there should be no
   need to attempt to merge information across different PVDs.

   Nevertheless, even when a PVD lack some parts of the configuration,
   merging of information from different PVD(s) shall not be done
   automatically, since typically it would lead to issues described in
   [RFC6418].

   A node may use other sources, such as e.g., node local policy, user
   input or other mechanisms, not defined by IETF, to either construct a
   PVD entirely (analogously to static IP configuration of an
   interface), or supplement with particular elements all or some PVDs
   learned from the network, or potentially merge information from
   different PVDs, if such merge is known to the node to be safe, based

on explicit policies.

As an example, node administrator could inject a not ISP-specific DNS
server into PVDs for any of the networks the node could become
attached to.  Such creation / augmentation of PVD(s) could be static
or dynamic.  The particular implementation mechanisms are outside of
the scope of this document.

## 5.2.  Consistent use of PVDs for network connections

PVDs enable PVD-aware nodes to use consistently a correct set of
configuration elements to serve the specific network requests from
beginning to end.  This section describes specific examples of such
consistent use.

### 5.2.1.  Name resolution

When PVD-aware node needs to resolve a name of the destination used
by a connection request, the node could decide to use one, or
multiple PVDs for a given name lookup.

The node shall chose one PVD, if e.g., the node policy required to
use a particular PVD for a particular purpose (e.g.  to download an
MMS using a specific APN over a cellular connection).  To make the
choice, the node could use a match of the PVD DNS suffix or other
form of PVD ID, as determined by the node policy.

The node may pick multiple PVDs, if e.g., they are general purpose
PVDs providing connectivity to the Internet, and the node desires to
maximize chances for connectivity in Happy Eyeballs style.  In this
case, the node could do the lookups in parallel, or in sequence.
Alternatively, the node may use for the lookup only one PVD, based on
the PVD connectivity properties, user choice of the preferred
Internet PVD, etc.

In either case, by default the node uses information obtained in a
name service lookup to establish connections only within the same PVD
from which the lookup results were obtained.

For simplicity, when we say that name service lookup results were
obtained from a PVD, what we mean is that the name service query was
issued against a name service the configuration of which is present
in a particular PVD.   In that sense, the results are "from" that
particular PVD.

Some nodes may support transports and/or APIs, which provide an
abstraction of a single connection, aggregating multiple underlying
connections.  MPTCP [RFC6182] is an example of such transport
protocol.  For the connections provided by such transports/APIs, a
PVD-aware node may use different PVDs for servicing of that logical
connection, provided that all operations on the underlying

connections are done consistently within their corresponding PVD(s).

## 5.2.2.  Next-hop and source address selection

For the purpose of this discussion, let's assume the preceding name
lookup succeeded in a particular PVD.  For each obtained destination
address, the node shall perform a next-hop lookup among routers,
associated with that PVD. As an example, such association could be
determined by the node via matching the source address prefixes/
specific routes advertized by the router against known PVDs, or
receiving explicit PVD affiliation advertized through a new Router
Discovery [RFC4861] option.

For each destination, once the best next-hop is found, the node
selects best source address according to the [RFC6724] rules, but
with a constraint that the source address must belong to a range
associated with the used PVD. If needed, the node would use the
prefix policy from the same PVD for the best source address selection
among multiple candidates.

When destination/source pairs are identified, then they are sorted
using the [RFC6724] destination sorting rules and the prefix policy
table from the used PVD.

## 5.3.  Connectivity tests

Although some PVDs may appear as valid candidates for PVD selection
(e.g.  good link quality, consistent connection parameters, etc.),
they may provide limited or no connectivity to the desired network or
the Internet.  For example, some PVDs provide limited IP connectivity
(e.g., scoped to the link or to the access network), but require the
node to authenticate through a web portal to get full access to the
Internet.  This may be more likely to happen for PVDs, which are not
trusted by the given PVD-aware node.

An attempt to use such PVD may lead to limited network connectivity
or connection failures for applications.  To prevent the latter, a
PVD-aware node may perform connectivity test for the PVD, before
using it to serve network connection requests of the applications.
In current implementations, some nodes do that, for instance, by
trying to reach a dedicated web server (e.g., see [RFC6419]).

Per Section 5.2, a PVD-aware node shall maintain and use multiple
PVDs separately.  The PVD-aware node shall perform connectivity test
and, only after validation of the PVD, consider using it to serve
application connections requests.  Ongoing connectivity tests are
also required, since during the IP session, the end-to-end
connectivity could be disrupted for various reasons (e.g.  poor L2,

IP QoS issues); hence a connectivity monitoring function is needed to
check the connectivity status and remove the PVD from the set of
usable PVDs if necessary.

There may be cases where a connectivity test for PVD selection may be
not appropriate and should be complemented, or replaced, by PVD
selection based on other factors.  This could be realized e.g., by
leveraging some 3GPP and IEEE mechanisms, which would allow to expose
some PVD characteristics to the node (e.g.  3GPP Access Network
Discovery and Selection Function (ANDSF) [TS23.402], IEEE 802.11u
[IEEE802.11u]/ANQP).

## 5.4.  Relationship to interface management and connection managers

Current devices such as mobile handsets make use of proprietary
mechanisms and custom applications to manage connectivity in
environments with multiple interfaces and multiple sets of network
configurations.  These mechanisms or applications are commonly known
as connection managers [RFC6419].

Connection managers sometimes rely on policy servers to allow the
node, connected to multiple networks, perform the network selection.
They can also make use of routing guidance from the network (e.g.
3GPP ANDSF [TS23.402]).  Although connection managers solve some
connectivity problems, they rarely address the network selection
problems in a comprehensive manner.  With proprietary solutions, it
is challenging to present a coherent behaviour to the end user of the
device, as different platforms present different behaviours even when
connected to the same network, with the same type of interface, and
for the same purpose.

## 6.  PVD support in APIs

In all cases changes in available PVDs must be somehow exposed,
appropriately for each of the approaches.

## 6.1.  Basic

Applications are not PVD-aware in any manner, and only submit
connection requests.  The node performs PVD selection implicitly,
without any otherwise applications participation, and based purely on
node-specific administrative policies and/or choices made by the user
in a user interface provided by the operating environment, not by the
application.

As an example, such PVD selection can be done at the name service
lookup step, by using the relevant configuration elements, such as
e.g., those described in [RFC6731].  As another example, the PVD
selection could be done based on application identity or type (i.e.,
a node could always use a particular PVD for a VOIP application).

## 6.2.  Intermediate

Applications indirectly participate in selection of PVD by specifying
hard requirements and soft preferences.  The node performs PVD
selection, based on applications inputs and policies and/or user
preferences.  Some / all properties of the resultant PVD may be
exposed to applications.

## 6.3.  Advanced

PVDs are directly exposed to applications, for enumeration and
selection.  Node polices and/or user choices, may still override the
application preferences and limit which PVD(s) can be enumerated and/
or used by the application, irrespectively of any preferences which
application may have specified.  Depending on the implementation,
such restrictions, imposed per node policy and/or user choice, may or
may not be visible to the application.

## 7.  PVD-aware nodes trust to PVDs

## 7.1.  Untrusted PVDs

Implicit and explicit PVDs for which no trust relationship exists are
considered untrusted.   Only PVDs, which meet the requirements in
Section 7.2, are trusted; any other PVD is untrusted.

In order to avoid various forms of misinformation that can be
asserted when PVDs are untrusted, nodes that implement PVD separation
cannot assume that two explicit PVDs with the same identifier are
actually the same PVD.  A node that did make this assumption would be
vulnerable to attacks where for example an open Wifi hotspot might
assert that it was part of another PVD, and thereby might draw
traffic intended for that PVD onto its own network.

Since implicit PVD identifiers are synthesized by the node, this
issue cannot arise with implicit PVDs.

Mechanisms exist (for example, [RFC6731]) whereby a PVD can provide
configuration information that asserts special knowledge about the
reachability of resources through that PVD.   Such assertions cannot
be validated unless the node has a trust relationship with the PVD;
assertions of this type therefore must be ignored by nodes that
receive them from untrusted PVDs.   Failure to ignore such assertions
could result in traffic being diverted from legitimate destinations
to spoofed destinations.

## 7.2.  Trusted PVDs

Trusted PVDs are PVDs for which two conditions apply.   First, a
trust relationship must exist between the node that is using the PVD
configuration and the source that provided that configuration; this
is the authorization portion of the trust relationship.   Second,
there must be some way to validate the trust relationship.   This is
the authentication portion of the trust relationship.   Two
mechanisms for validating the trust relationship are defined.

It shall be possible to validate the trust relationship for all
advertised elements of a trusted PVD, irrespectively of whether the
PVD elements are communicated as a whole, e.g.  in a single DHCP
option, or separately, e.g.  in supplementary RA options.  Whether or
not this is feasible to provide mechanisms to implement trust
relationship for all PVD elements, will be determined in the
respective companion design documents.

### 7.2.1.  Authenticated PVDs

One way to validate the trust relationship between a node and the
source of a PVD is through the combination of cryptographic
authentication and an identifier configured on the node.   In some
cases, the two could be the same; for example, if authentication is
done with a shared secret, the secret would have to be associated
with the PVD identifier.   Without a (PVD Identifier, shared key)
tuple, authentication would be impossible, and hence authentication
and authorization are combined.

However, if authentication is done using some public key mechanism
such as a TLS cert or DANE, authentication by itself isn't enough,
since theoretically any PVD could be authenticated in this way.   In
addition to authentication, the node would need to be configured to
trust the identifier being authenticated.  Validating the
authenticated PVD name against a list of PVD names configured as
trusted on the node would constitute the authorization step in this
case.

### 7.2.2.  PVDs trusted by attachment

In some cases a trust relationship may be validated by some means
other than described in Section 7.2.1, simply by virtue of the
connection through which the PVD was obtained.   For instance, a
handset connected to a mobile network may know through the mobile
network infrastructure that it is connected to a trusted PVD, and
whatever mechanism was used to validate that connection constitutes
the authentication portion of the PVD trust relationship.
Presumably such a handset would be configured from the factory, or
else through mobile operator or user preference settings, to trust
the PVD, and this would constitute the authorization portion of this

type of trust relationship.

**8.  Acknowledgements**

This document was created as a product of a MIF architecture design team.

## 9.  IANA Considerations

This memo includes no request to IANA.

## 10.  Security Considerations

There are at least three different form of attacks that can be performed using configuration sources that use multiple provisioning domains.

Tampering with configuration information provided An attacker may attempt to modify the information provided inside the PVD container option.  These attacks can easily be prevented by using the message integrity features provided by the underlying protocol used to carry the configuration information.  e.g.  SEND [RFC3971] would detect any form of tampering with the RA contents and the DHCPv6 [RFC3315] AUTH option that would detect any form of tampering with the DHCPv6 message contents.  This attack can also be performed by a compromised configuration source by modifying information inside a specific , in which case the mitigations proposed in the next subsection may be helpful.

Rogue configuration source A compromised configuration source such as a router or a DHCPv6 server may advertise information about PvDs that it is not authorized to advertise.  e.g.  A coffee shop may advertise 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 related configuration 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).

Replay attacks 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.

## 11.  References

## 11.1.  Normative References

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

[IEEE802.11u]
          IEEE, "IEEE Standard 802.11u-2011 (Amendment 9:
          Interworking with External Networks)", 2011.

[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.

[RFC4861]  Narten, T., Nordmark, E., Simpson, W. and H. Soliman,
          "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861,
          September 2007.

[RFC5739]  Eronen, P., Laganier, J. and C. Madson, "IPv6
          Configuration in Internet Key Exchange Protocol Version 2
          (IKEv2)", RFC 5739, February 2010.

[RFC5996]  Kaufman, C., Hoffman, P., Nir, Y. and P. Eronen, "Internet
          Key Exchange Protocol Version 2 (IKEv2)", RFC 5996,
          September 2010.

[RFC6182]  Ford, A., Raiciu, C., Handley, M., Barre, S. and J.
          Iyengar, "Architectural Guidelines for Multipath TCP
          Development", RFC 6182, March 2011.

[RFC6418]  Blanchet, M. and P. Seite, "Multiple Interfaces and
          Provisioning Domains Problem Statement", RFC 6418,
          November 2011.

[RFC6419]  Wasserman, M. and P. Seite, "Current Practices for
          Multiple-Interface Hosts", RFC 6419, November 2011.

[RFC6724]  Thaler, D., Draves, R., Matsumoto, A. and T. Chown,
          "Default Address Selection for Internet Protocol Version 6
          (IPv6)", RFC 6724, September 2012.

[RFC6731]  Savolainen, T., Kato, J. and T. Lemon, "Improved Recursive
          DNS Server Selection for Multi-Interfaced Nodes", RFC
          6731, December 2012.

[RFC7078]  Matsumoto, A., Fujisaki, T. and T. Chown, "Distributing
          Address Selection Policy Using DHCPv6", RFC 7078, January
          2014.

[TS23.402]
          3GPP, "3GPP TS 23.402; Architecture enhancements for non-
          3GPP accesses; release 12", .

Author's Address

Dmitry Anipko, editor
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA

Phone: +1 425 703 7070
Email: dmitry.anipko@microsoft.com