

Homenet Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: April 18, 2016

S. Barth  
Independent  
October 16, 2015

Home Network WiFi Roaming  
draft-barth-homenet-wifi-roaming-00

Abstract

This document describes a mechanism to manage host routes and statelessly proxy IPv6 Duplicate Address Detection messages between multiple WiFi links to allow client roaming.

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 18, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
2.1. Requirements Language . . . . .	3
3. Stateless Duplicate Address Detection (DAD) Proxy . . . . .	3
3.1. Receipt of DAD Messages and forwarding to routers . . . . .	3
3.2. Distributing DAD Messages received from other routers . . . . .	4
4. Maintaining Host Routes . . . . .	4
5. Roaming Interface Configuration . . . . .	5
6. Security Considerations . . . . .	5
7. IANA Considerations . . . . .	5
8. References . . . . .	5
8.1. Normative references . . . . .	5
8.2. Informative references . . . . .	6
Appendix A. Discussion Points [RFC Editor: please remove] . . . . .	6
Appendix B. Changelog [RFC Editor: please remove] . . . . .	6
Appendix C. Draft Source [RFC Editor: please remove] . . . . .	6
Appendix D. Acknowledgements . . . . .	7
Author's Address . . . . .	7

## 1. Introduction

In a multi router home network it can be desirable to have a WiFi network accessible in different physical locations. Synchronization of configuration parameters like ESSID and authentication allows clients to seamlessly roam the network. However, a client switching from one WiFi AP to another might suffer from service disruption if each AP uses a different IP address prefix. To mitigate this issue, all AP links could be bridged on layer 2, which would lead to increased traffic on the home network backbone. This draft offers an alternative solutions based on host routing and proxying of Duplicate Address Detection for IPv6.

In order to minimize additional complexity on routers, this draft either relies on existing state in the form the neighbor cache entries used for host routing or introduces only lightweight, stateless mechanism to distribute Duplicate Address Detection messages.

However, an additional mechanism is needed to identify and share information about which routers have roaming interfaces, to which roaming interface set they belong, under which addresses these routers are reachable and which specific roaming prefixes are assigned. The specific mechanism is out of scope for this draft, however a solution based on [I-D.ietf-homenet-hncp] is conceivable.

## 2. Terminology

ESSID	IEEE 802.11 Extended Service Set Identifier
host route	an IPv6 route with a prefix length of 128 intended to track a host on a roaming interface
roaming interface	a network interface (typically IEEE 802.11) which shares an address prefix with other similar interfaces to allow seamless client roaming
roaming interface set	the set of roaming interfaces which share a particular address prefix
roaming prefix	an IPv6 address prefix shared between all interfaces in a roaming interface set

### 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. Stateless Duplicate Address Detection (DAD) Proxy

Sharing a roaming prefix across multiple separate interfaces might lead to address collisions between hosts on different interfaces of the same set. Therefore, it needs to be ensured that DAD messages are shared across interfaces. The typical DAD process involves the querying host sending one or more Neighbor Solicitations using the unspecified address as a source and any colliding host replying to the all-nodes address as specified in [RFC4862] and [RFC4861]. Since the source addresses of the DAD solicitations and the destination addresses of the DAD advertisements are fixed, the whole process can happen statelessly.

### 3.1. Receipt of DAD Messages and forwarding to routers

The following requirements apply for receiving DAD traffic from hosts on roaming interfaces and forwarding them to other DAD proxy routers:

A router MUST listen for all Neighbor Solicitations with a target addresses from an assigned roaming prefix having the unspecified address as the source address. Similarly it MUST listen for all Neighbor Advertisements with a target address from an assigned roaming prefix and having the all-nodes multicast address as the destination address.

A router MUST forward all such messages via global unicast to all other routers having roaming interfaces sharing the roaming prefixes the target address of the respective message belongs to.

### 3.2. Distributing DAD Messages received from other routers

The following requirements apply for distributing DAD traffic, forwarded by other routers, to clients on roaming interfaces:

A router MUST listen for Neighbor Solicitations and Advertisements sent via global unicast from other routers having roaming interfaces with the same roaming prefix.

Upon reception of such a Neighbor Solicitation, the router MUST forward the message to all roaming interfaces having prefixes matching the target address. It MUST use the unspecified address as source and the respective solicited node multicast-address as destination address.

Upon reception of such a Neighbor Advertisement, the router MUST forward the message to all roaming interfaces having prefixes matching the target address. It MUST use the all-nodes multicast-address as destination address.

## 4. Maintaining Host Routes

Host routes are necessary in order to unambiguously forward packets to potentially roaming WiFi clients. This draft ties the announcement of host routes to the presence of Neighbor Cache [RFC4861] entries for addresses of roaming prefixes. The following requirements apply:

A router MUST announce a host route for an IPv6 address belonging to a roaming prefix whenever there is a corresponding entry for it in the roaming interface's neighbor cache with the states REACHABLE, STALE, DELAY or PROBE.

A router MUST retract any host route for an IPv6 address immediately when a neighbor lookup for it has finally failed on the roaming interface OR when a physical disconnect of the respective client (e.g., a WiFi disassociation) has been noticed.

A router SHOULD be able to manage and publish host routes for at least two IPv6 addresses for every connected host and IPv6 roaming prefix announced on a roaming interface.

The router SHOULD NOT flush neighbor cache entries in STALE state and respective host routes. Instead it SHOULD try to promote

STALE entries to REACHABLE and only upon failure remove the cache entry and route. This ensures that the address is actually not used anymore and the host does not happen to be merely temporarily inactive and potentially waiting for incoming packets.

## 5. Roaming Interface Configuration

The following requirements are suggested for interfaces intended to use the roaming features described in this draft:

Router Advertisements SHOULD only include Prefix Information Options for one more IPv6 roaming prefixes. Such Prefix Information Options MUST have the A-Bit set and the L-Bit cleared.

The IPv6 address fe80::1 SHOULD be used as fixed link-local address exclusively by the router on roaming interfaces. This is to ensure that the IPv6 default route for clients stays the same across roaming interfaces. Furthermore the address SHOULD be announced as recursive DNS server address should the router provide such a service.

Router Advertisements SHOULD be sent as described in [I-D.ietf-v6ops-reducing-ra-energy-consumption].

Stateful DHCPv6 MUST NOT be used to avoid the need to synchronize lease information and relay DHCPv6 packets. Similarly Router Advertisements MUST be sent with the M-Bit cleared.

NAT64 and DNS64 SHOULD be used if IPv4 connectivity is desired, alternatively different and disjoint native IPv4 prefixes MAY be used on each individual roaming interface.

## 6. Security Considerations

TBD

## 7. IANA Considerations

No action needed.

## 8. References

### 8.1. Normative references

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

- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.
- [I-D.ietf-v6ops-reducing-ra-energy-consumption] Yourtchenko, A. and L. Colitti, "Reducing energy consumption of Router Advertisements", draft-ietf-v6ops-reducing-ra-energy-consumption-02 (work in progress), October 2015.

## 8.2. Informative references

- [I-D.ietf-homenet-hncp] Stenberg, M., Barth, S., and P. Pfister, "Home Networking Control Protocol", draft-ietf-homenet-hncp-09 (work in progress), August 2015.

## Appendix A. Discussion Points [RFC Editor: please remove]

- o It is debatable which kind of transport could be used for the proxied DAD messages. Plain ICMPv6 could be possibly but might be dropped by certain hosts. Similarly UDP or DTLS encapsulation could be used easily if a port was allocated or exchanged.
- o Is additional reliability useful and if so what kind? Normally hosts only send a single DAD solicitation, so the probability of a total packet loss increases with each forwarding hop. Trivially one could simply duplicate all proxied DAD messages or use TCP framing or SCTP. However the announced Retrans Timer value in the RA and retransmission parameters of the transport need to be adjusted, so that any resend and matching replies happen before the host considers the DAD process to have resulted in no collisions.

## Appendix B. Changelog [RFC Editor: please remove]

draft-barth-homenet-wifi-roaming-00:

- o Initial version.

## Appendix C. 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 D. Acknowledgements

Thanks to Markus Stenberg for comments and feedback on the draft.

#### Author's Address

Steven Barth  
Independent  
Halle 06114  
Germany

Email: [cyrus@openwrt.org](mailto:cyrus@openwrt.org)

Homenet  
Internet-Draft  
Intended status: Standards Track  
Expires: April 18, 2016

L. Geng  
H. Deng  
China Mobile  
October 16, 2015

Use Cases for Multiple Provisioning Domain in Homenet  
draft-geng-homenet-mpvd-use-cases-02

Abstract

This document describes the use cases of multiple provisioning domain (MPVD) in homenet. Although most residential networks nowadays are connected to a single ISP and normally subscribed to standard internet service, it is expected that much wider range of devices and services will become common in home networks. Homenet defines such home network topologies with increasing number of devices with the assumption that it requires minimum configuration by residential user. As described in the homenet architecture ([RFC7368]), multihoming and multi-service residential network will be more common in the near future. Nodes in such network may commonly have multiple interfaces or subscribe to multiple services. Potential types of PVD-aware nodes concerning interface and service specific provisioning domains are introduced in this document. Based on this, different MPVD configuration examples are given. These examples illustrate how PVD may be implemented in home network. PVDs provide independent provisioning domains for different interfaces and services, which enables robust and flexible network configuration for these networks.

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 18, 2016.



## Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
1.1. Requirements Language . . . . .	3
2. Terminology and Abbreviations . . . . .	3
3. Homenet with Multiple PvDs . . . . .	3
4. Examples of MPvD Configurations in Home Network . . . . .	4
4.1. Single CE Router Connected to Single ISP with interior router . . . . .	4
4.2. Two CE Routers Connected to two ISPs with Shared Subnets	6
4.3. One CE Routers Connected to two ISPs with Shared Subnets	6
5. PvD-aware node in homenet . . . . .	6
6. IPv4 compatibility . . . . .	7
7. Conveying PvD information . . . . .	7
8. Acknowledgements . . . . .	7
9. IANA Considerations . . . . .	7
10. Security Considerations . . . . .	7
11. References . . . . .	8
11.1. Normative References . . . . .	8
11.2. Informative References . . . . .	8
Authors' Addresses . . . . .	8

## 1. Introduction

It is believed that future residential network will more commonly be multihomed, which potentially provides either resilience or more flexible services. At the same time, more internal routing and multiple subnets are expected to commonly exist in such networks. For example, customer may want independent subnets for private and guest usages. Homenet describes such future home network involving multiple routers and subnets ([RFC7368]).

Multihoming and the increasing number of subnets bring challenges on provisioning of the network. As stated in [RFC6418], such multihomed scenarios with nodes attached to multiple upstream networks may experience configuration conflicts, leading to a number of problems. To deal with these problem, draft-ietf-mif-mpvd-arch-10 provides a framework which introduces Provisioning Domain (PvD), which associates a certain interface and its related network configuration information. Hence, corresponding network configuration can be used when packets are delivered through a particular interface.

This document focuses on the MPvD use cases in residential network, particularly the IPV6-based homenet. Based on the homenet topology, use cases of MPvD in homenet are described for both singlehomed and multihomed network configurations.

### 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. Terminology and Abbreviations

The terminology and abbreviations used in this document are defined in this section.

- o ISP: Internet Service Provider. A traditional network operator who provides internet access to customers.

## 3. Homenet with Multiple PvDs

In the most common multihoming scenarios, the home network has multiple physical connections to the ISP networks. Section 3.2.2.2 and 3.2.2.3 in [RFC7368] give the topology examples of such homenet. In the examples, homenet hosts are connected to a single or multiple customer edge routers (CE router), the CE routers are then connected to separate ISP networks. For the particular topology with a single CE router given in Section 3.2.2.3 in [RFC7368], the CE router is a mif node since it has two interfaces connected to individual service provider routers. Given that the CE router is a PvD-aware node, it may have two PvDs provided by ISPs respectively.

Apart from the multihoming resulted from physical connections , PvDs in Homenet can also be used for service provisioning. For example, a host may subscribe one ISP for internet service, whilst subscribe to another ISP for Internet of Things (IoT) service given that the CE router have access to both ISPs. On the other hand, the host user may also subscribe to the same ISP for both services. In either

case, PvDs can be used for customized network configurations purposes. This enables the service providers to provide independent and flexible provisioning for different services. Meanwhile, IoT service providers may also want to use independent PvDs to avoid the configuration conflicts between each other as stated in RFC6418.

A typical example of a PvD in home network is the one associating corresponding network configuration with an HNCP routers. These includes both CE router and interior router in Homenet. As described in ([RFC7368]), an CE router in homenet may have one or more external interfaces with ISPs and internal interfaces with interior routers. For external interfaces, the CE router can receive associated PvD information from corresponding ISPs. For interior interfaces, the interior router can receive PvD information from connected CE router or other interior routers.

Hosts in homenet are expected to be multihomed as well. Hence, PvD may also be used in such cases to associate different network configurations. In this case, the PvD information is received from the HNCP router a host is attached to, either a CE router or a interior router.

#### 4. Examples of MPvD Configurations in Home Network

This section gives some examples of MPvD configurations in home network.

##### 4.1. Single CE Router Connected to Single ISP with interior router

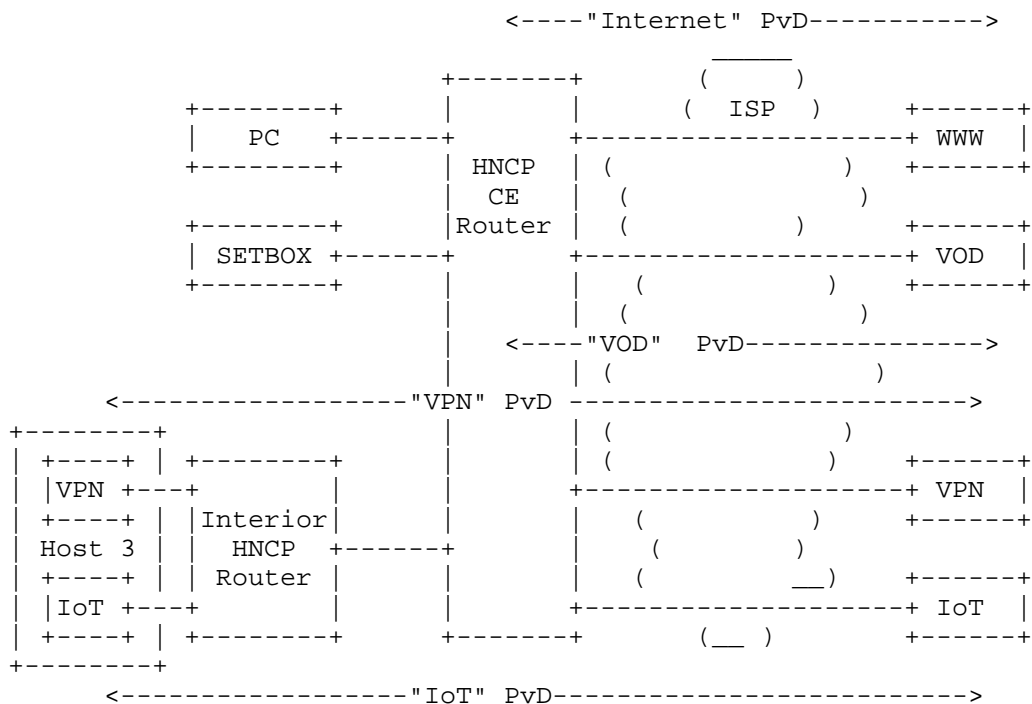


Figure 1

In this example, A homenet is connected with a single ISP as seen in Figure 1. Four different services are provided to this home network including Internet, VOD, VPN and IoT. There are 3 Hosts in this set-up. PC and setbox use "Internet" and "VoD" PvdS respectively and Host 3 uses both "VPN" and "IoT" Pvd.

The four PVDs could be either implicit or explicit PVDs. Explicit PVDs should be initially assigned to HNCP CE router by ISP. And then forwarded to interior routers and host. Given that all PVDs are explicit in the case above, the "Internet" PVD is forwarded to PC and "VOD" PVD to SETBOX by the CE router, and the "VPN" and "IoT" PVDs are forwarded to interior HNCP router and then Host 3. The PVD\_ID should be kept the same when the PVDs are forwarded. However, other associated network configuration (i.e. delegated prefixes) should be changed accordingly.

#### 4.2. Two CE Routers Connected to two ISPs with Shared Subnets

To be added

#### 4.3. One CE Routers Connected to two ISPs with Shared Subnets

To be added

### 5. PvD-aware node in homenet

As defined in MIF, "PvD-aware node is a node that supports the association of network configuration information into PvDs and the use of these PvDs to serve requests for network connections". In Homenet, the HNCP CE router, interior router and host are all PvD-aware nodes.

The HNCP CE router PvD-related functionality is define as follows:

- o Generates implicit PvDs for different uplink interfaces.
- o Requests and receives all explicit PvDs provided by the connected ISPs.
- o Generates explicit PvDs for interior routers and hosts referring to the ISP-provided PvDs and forwards them accordingly.
- o Creates and stores the PvD mapping between the PvD applied itself the the one forwarded to interior routers and hosts using the assigned PvD\_ID and prefix.
- o Identify the prefix received from homenet nodes and performs PvD selection based on PvD mapping.

The interior router PvD-related functionality is defined as follows:

- o Generates implicit PvDs for different homenet internal interface.
- o Requests and receives all explicit PvDs provided by connected homenet routers.
- o Generates explicit PvDs for interior routers and hosts referring to the homenet-router-provided PvDs and forwards them accordingly.
- o Creates and stores the PvD mapping between the PvD applied itself the the one forwarded to interior routers and hosts using the assigned PvD\_ID and prefix.

- o Identify the prefix received from homenet nodes and performs PvD selection based on PvD mapping.

The host PvD-related functionality is defined as follows:

- o Generates implicit PvDs for different interfaces between host and homenet routers.
- o Requests and receives all explicit PvDs provided by connected homenet routers.

#### 6. IPv4 compatibility

PvD in homenet can be either single-family or dual-stack. For single-family PvD, the IPV4 and IPV6 configurations should be managed in separate PvDs with different PvD identities. For Dual-stack PvDs, IPV4 and IPV6 configurations can exist in the same PvD. In both cases, there can either be only one IPV4 PvD for each interface or multiple IPV4 PvDs with different default gateway addresses.

#### 7. Conveying PvD information

At the time this document was written, the conveying of PvD information was still under discussion in mif working group. Popular choices include DNS, DHCP and Route Advertisement. For PvD information provided from ISP to CE router and router to host, the approaches for PvD information delivery defined by mif may be directly used. For PvD information delivery within homenet between HNCP-enabled routers, HNCP-based approach need to be defined. The detail of how homenet could support the delivery of PvD information between routers is subjected to further discussions and will be addressed in a separate document.

#### 8. Acknowledgements

The author would like to thank Ted Lemon, Mark Townsley, Markus Stenberg and Steven Barth for valuable discussions and contributions to this document.

#### 9. IANA Considerations

This memo includes no request to IANA.

#### 10. Security Considerations

TBA

## 11. References

### 11.1. Normative References

- [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>>.
- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, DOI 10.17487/RFC2629, June 1999, <<http://www.rfc-editor.org/info/rfc2629>>.

### 11.2. Informative References

- [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>>.
- [RFC7368] Chown, T., Ed., Arkko, J., Brandt, A., Troan, O., and J. Weil, "IPv6 Home Networking Architecture Principles", RFC 7368, DOI 10.17487/RFC7368, October 2014, <<http://www.rfc-editor.org/info/rfc7368>>.

## Authors' Addresses

Liang Geng  
China Mobile

Email: [gengliang@chinamobile.com](mailto:gengliang@chinamobile.com)

Hui Deng  
China Mobile

Email: [denghui@chinamobile.com](mailto:denghui@chinamobile.com)

Homenet Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 16, 2016

M. Stenberg  
S. Barth  
Independent  
October 14, 2015

Distributed Node Consensus Protocol  
draft-ietf-homenet-dncp-11

Abstract

This document describes the Distributed Node Consensus Protocol (DNCP), a generic state synchronization protocol that uses the Trickle algorithm and hash trees. DNCP is an abstract protocol, and must be combined with a specific profile to make a complete implementable protocol.

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 16, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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



## Table of Contents

1. Introduction . . . . .	3
1.1. Applicability . . . . .	3
2. Terminology . . . . .	5
2.1. Requirements Language . . . . .	7
3. Overview . . . . .	7
4. Operation . . . . .	8
4.1. Hash Tree . . . . .	8
4.1.1. Calculating network state and node data hashes . . . . .	8
4.1.2. Updating network state and node data hashes . . . . .	9
4.2. Data Transport . . . . .	9
4.3. Trickle-Driven Status Updates . . . . .	10
4.4. Processing of Received TLVs . . . . .	11
4.5. Discovering, Adding and Removing Peers . . . . .	14
4.6. Data Liveliness Validation . . . . .	15
5. Data Model . . . . .	16
6. Optional Extensions . . . . .	17
6.1. Keep-Alives . . . . .	18
6.1.1. Data Model Additions . . . . .	18
6.1.2. Per-Endpoint Periodic Keep-Alives . . . . .	19
6.1.3. Per-Peer Periodic Keep-Alives . . . . .	19
6.1.4. Received TLV Processing Additions . . . . .	19
6.1.5. Peer Removal . . . . .	19
6.2. Support For Dense Multicast-Enabled Links . . . . .	20
7. Type-Length-Value Objects . . . . .	21
7.1. Request TLVs . . . . .	22
7.1.1. Request Network State TLV . . . . .	22
7.1.2. Request Node State TLV . . . . .	22
7.2. Data TLVs . . . . .	22
7.2.1. Node Endpoint TLV . . . . .	22
7.2.2. Network State TLV . . . . .	23
7.2.3. Node State TLV . . . . .	23
7.3. Data TLVs within Node State TLV . . . . .	24
7.3.1. Peer TLV . . . . .	24
7.3.2. Keep-Alive Interval TLV . . . . .	25
8. Security and Trust Management . . . . .	25
8.1. Pre-Shared Key Based Trust Method . . . . .	25
8.2. PKI Based Trust Method . . . . .	26
8.3. Certificate Based Trust Consensus Method . . . . .	26
8.3.1. Trust Verdicts . . . . .	26
8.3.2. Trust Cache . . . . .	27
8.3.3. Announcement of Verdicts . . . . .	28
8.3.4. Bootstrap Ceremonies . . . . .	29
9. DNPCP Profile-Specific Definitions . . . . .	30
10. Security Considerations . . . . .	31
11. IANA Considerations . . . . .	32
12. References . . . . .	33

12.1. Normative references . . . . .	33
12.2. Informative references . . . . .	33
Appendix A. Alternative Modes of Operation . . . . .	34
A.1. Read-only Operation . . . . .	34
A.2. Forwarding Operation . . . . .	34
Appendix B. DNCP Profile Additional Guidance . . . . .	34
B.1. Unicast Transport - UDP or TCP? . . . . .	34
B.2. (Optional) Multicast Transport . . . . .	35
B.3. (Optional) Transport Security . . . . .	35
Appendix C. Example Profile . . . . .	36
Appendix D. Some Questions and Answers [RFC Editor: please remove] . . . . .	37
Appendix E. Changelog [RFC Editor: please remove] . . . . .	37
Appendix F. Draft Source [RFC Editor: please remove] . . . . .	39
Appendix G. Acknowledgements . . . . .	39
Authors' Addresses . . . . .	39

## 1. Introduction

DNCP is designed to provide a way for each participating node to publish a small set of TLV (Type-Length-Value) tuples (at most 64 KB), and to provide a shared and common view about the data published by every currently bidirectionally reachable DNCP node in a network.

For state synchronization a hash tree is used. It is formed by first calculating a hash for the dataset published by each node, called node data, and then calculating another hash over those node data hashes. The single resulting hash, called network state hash, is transmitted using the Trickle algorithm [RFC6206] to ensure that all nodes share the same view of the current state of the published data within the network. The use of Trickle with only short network state hashes sent infrequently (in steady state, once the maximum Trickle interval per link or unicast connection has been reached) makes DNCP very thrifty when updates happen rarely.

For maintaining liveness of the topology and the data within it, a combination of Trickled network state, keep-alives, and "other" means of ensuring reachability are used. The core idea is that if every node ensures its peers are present, transitively, the whole network state also stays up-to-date.

### 1.1. Applicability

DNCP is useful for cases like autonomous bootstrapping, discovery and negotiation of embedded network devices like routers. Furthermore it can be used as a basis to run distributed algorithms like [I-D.ietf-homenet-prefix-assignment] or usecases as described in Appendix C. The topology of the devices is not limited and

automatically discovered. If globally scoped addresses are used, DNCP peers do not necessarily need to be on the same link. Autonomous discovery features are usually used in local network scenario however - with security enabled - DNCP can also be used over unsecured public networks. Network size is restricted merely by the capabilities of the devices, i.e., each DNCP node needs to be able to store the entirety of the data published by all nodes.

DNCP is most suitable for data that changes only infrequently to gain the maximum benefit from using Trickle. As the network of nodes grows, or the frequency of data changes per node increases, Trickle is eventually used less and less and the benefit of using DNCP diminishes. In these cases Trickle just provides extra complexity within the specification and little added value.

The suitability of DNCP for a particular application can roughly be evaluated by considering the expected average network-wide state change interval  $A_{NC\_I}$ ; it is computed by dividing the mean interval at which a node originates a new TLV set by the number of participating nodes. If keep-alives are used,  $A_{NC\_I}$  is the minimum of the computed  $A_{NC\_I}$  and the keep-alive interval. If  $A_{NC\_I}$  is less than the (application-specific) Trickle minimum interval, DNCP is most likely unsuitable for the application as Trickle will not be utilized most of the time.

If constant rapid state changes are needed, the preferable choice is to use an additional point-to-point channel whose address or locator is published using DNCP. Nevertheless, if doing so does not raise  $A_{NC\_I}$  above the (sensibly chosen) Trickle interval parameters for a particular application, using DNCP is probably not suitable for the application.

Another consideration is the size of the published TLV set by a node compared to the size of deltas in the TLV set. If the TLV set published by a node is very large, and has frequent small changes, DNCP as currently specified in this specification may be unsuitable as it lacks a delta synchronization scheme to keep implementation simple.

DNCP can be used in networks where only unicast transport is available. While DNCP uses the least amount of bandwidth when multicast is utilized, even in pure unicast mode, the use of Trickle (ideally with  $k < 2$ ) results in a protocol with an exponential backoff timer and fewer transmissions than a simpler protocol not using Trickle.

## 2. Terminology

DNCP profile	the values for the set of parameters, given in Section 9. They are prefixed with DNCP_ in this document. The profile also specifies the set of optional DNCP extensions to be used. For a simple example DNCP profile, see Appendix C.
DNCP-based protocol	a protocol which provides a DNCP profile, according to Section 9, and zero or more TLV assignments from the per-DNCP profile TLV registry as well as their processing rules.
DNCP node	a single node which runs a DNCP-based protocol.
Link	a link-layer media over which directly connected nodes can communicate.
DNCP network	a set of DNCP nodes running DNCP-based protocol(s) with matching DNCP profile(s). The set consists of nodes that have discovered each other using the transport method defined in the DNCP profile, via multicast on local links, and / or by using unicast communication.
Node identifier	an opaque fixed-length identifier consisting of DNCP_NODE_IDENTIFIER_LENGTH bytes which uniquely identifies a DNCP node within a DNCP network.
Interface	a node's attachment to a particular link.
Address	an identifier used as source or destination of a DNCP message flow, e.g., a tuple (IPv6 address, UDP port) for an IPv6 UDP transport.
Endpoint	a locally configured termination point for (potential or established) DNCP message flows. An endpoint is the source and destination for separate unicast message flows to individual nodes and optionally for multicast messages to all thereby reachable nodes (e.g., for node discovery). Endpoints are usually in one of the transport modes specified in Section 4.2.
Endpoint identifier	a 32-bit opaque and locally unique value, which identifies a particular endpoint of a particular DNCP node. The value 0 is reserved for DNCP and DNCP-based protocol purposes and not used to

	identify an actual endpoint. This definition is in sync with the interface index definition in [RFC3493], as the non-zero small positive integers should comfortably fit within 32 bits.
Peer	another DNCP node with which a DNCP node communicates using a particular local and remote endpoint pair.
Node data	a set of TLVs published and owned by a node in the DNCP network. Other nodes pass it along as-is, even if they cannot fully interpret it.
Origination Time	the (estimated) time when the node data set with the current sequence number was published.
Node state	a set of metadata attributes for node data. It includes a sequence number for versioning, a hash value for comparing equality of stored node data, and a timestamp indicating the time passed since its last publication (i.e., since the origination time). The hash function and the length of the hash value are defined in the DNCP profile.
Network state hash	a hash value which represents the current state of the network. The hash function and the length of the hash value are defined in the DNCP profile. Whenever a node is added, removed or updates its published node data this hash value changes as well. For calculation, please see Section 4.1.
Trust verdict	a statement about the trustworthiness of a certificate announced by a node participating in the certificate based trust consensus mechanism.
Effective trust verdict	the trust verdict with the highest priority within the set of trust verdicts announced for the certificate in the DNCP network.
Topology graph	the undirected graph of DNCP nodes produced by retaining only bidirectional peer relationships between nodes.
Bidirectionally reachable	a peer is locally unidirectionally reachable if a consistent multicast or any unicast DNCP message has been received by the local node (see Section 4.5). If said peer in return also considers the local node unidirectionally reachable, then

bidirectionally reachability is established. As this process is based on publishing peer relationships and evaluating the resulting topology graph as described in Section 4.6, this information is available to the whole DNCP network.

**Trickle Instance** a distinct Trickle [RFC6206] algorithm state kept by a node (Section 5) and related to an endpoint or a particular (peer, endpoint) tuple with Trickle variables *I*, *t* and *c*. See Section 4.3.

### 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. Overview

DNCP operates primarily using unicast exchanges between nodes, and may use multicast for Trickle-based shared state dissemination and topology discovery. If used in pure unicast mode with unreliable transport, Trickle is also used between peers.

DNCP is based on exchanging TLVs (Section 7) and defines a set of mandatory and optional ones for its operation. They are categorized into TLVs for requesting information (Section 7.1), transmitting data (Section 7.2) and being published as data (Section 7.3). DNCP based protocols usually specify additional ones to extend the capabilities.

DNCP discovers the topology of the nodes in the DNCP network and maintains the liveness of published node data by ensuring that the publishing node is bidirectionally reachable. New potential peers can be discovered autonomously on multicast-enabled links, their addresses may be manually configured or they may be found by some other means defined in the particular DNCP profile. The DNCP profile may specify, for example, a well-known anycast address or provisioning the remote address to contact via some other protocol such as DHCPv6 [RFC3315].

A hash tree of height 1, rooted in itself, is maintained by each node to represent the state of all currently reachable nodes (see Section 4.1) and the Trickle algorithm is used to trigger synchronization (see Section 4.3). The need to check peer nodes for state changes is thereby determined by comparing the current root of their respective hash trees, i.e., their individually calculated network state hashes.

Before joining a DNCP network, a node starts with a hash tree that has only one leaf if the node publishes some TLVs, and no leaves otherwise. It then announces the network state hash calculated from the hash tree by means of the Trickle algorithm on all its configured endpoints.

When an update is detected by a node (e.g., by receiving a different network state hash from a peer) the originator of the event is requested to provide a list of the state of all nodes, i.e., all the information it uses to calculate its own hash tree. The node uses the list to determine whether its own information is outdated and - if necessary - requests the actual node data that has changed.

Whenever a node's local copy of any node data and its hash tree are updated (e.g., due to its own or another node's node state changing or due to a peer being added or removed) its Trickle instances are reset which eventually causes any update to be propagated to all of its peers.

## 4. Operation

### 4.1. Hash Tree

Each DNCP node maintains an arbitrary width hash tree of height 1. The root of the tree represents the overall network state hash and is used to determine whether the view of the network of two or more nodes is consistent and shared. Each leaf represents one bidirectionally reachable DNCP node. Every time a node is added or removed from the topology graph (Section 4.6) it is likewise added or removed as a leaf. At any time the leaves of the tree are ordered in ascending order of the node identifiers of the nodes they represent.

#### 4.1.1. Calculating network state and node data hashes

The network state hash and the node data hashes are calculated using the hash function defined in the DNCP profile (Section 9) and truncated to the number of bits specified therein.

Individual node data hashes are calculated by applying the function and truncation on the respective node's node data as published in the Node State TLV. Such node data sets are always ordered as defined in Section 7.2.3.

The network state hash is calculated by applying the function and truncation on the concatenated network state. This state is formed by first concatenating each node's sequence number (in network byte order) with its node data hash to form a per-node datum for each node. These per-node data are then concatenated in ascending order

of the respective node's node identifier, i.e., in the order that the nodes appear in the hash tree.

#### 4.1.2. Updating network state and node data hashes

The network state hash and the node data hashes are updated on-demand and whenever any locally stored per-node state changes. This includes local unidirectional reachability encoded in the published Peer TLVs (Section 7.3.1) and - when combined with remote data - results in awareness of bidirectional reachability changes.

#### 4.2. Data Transport

DNCP has few requirements for the underlying transport; it requires some way of transmitting either unicast datagram or stream data to a peer and, if used in multicast mode, a way of sending multicast datagrams. As multicast is used only to identify potential new DNCP nodes and to send status messages which merely notify that a unicast exchange should be triggered, the multicast transport does not have to be secured. If unicast security is desired and one of the built-in security methods is to be used, support for some TLS-derived transport scheme - such as TLS [RFC5246] on top of TCP or DTLS [RFC6347] on top of UDP - is also required. They provide for integrity protection and confidentiality of the node data, as well as authentication and authorization using the schemes defined in Security and Trust Management (Section 8). A specific definition of the transport(s) in use and their parameters MUST be provided by the DNCP profile.

TLVs (Section 7) are sent across the transport as is, and they SHOULD be sent together where, e.g., MTU considerations do not recommend sending them in multiple batches. DNCP does not fragment or reassemble TLVs thus it MUST be ensured that the underlying transport performs these operations should they be necessary. If this document indicates sending one or more TLVs, then the sending node does not need to keep track of the packets sent after handing them over to the respective transport, i.e., reliable DNCP operation is ensured merely by the explicitly defined timers and state machines such as Trickle (Section 4.3). TLVs in general are handled individually and statelessly (and thus do not need to be sent in any particular order) with one exception: To form bidirectional peer relationships DNCP requires identification of the endpoints used for communication. As bidirectional peer relationships are required for validating liveness of published node data as described in Section 4.6, a DNCP node MUST send a Node Endpoint TLV (Section 7.2.1). When it is sent varies, depending on the underlying transport, but conceptually it should be available whenever processing a Network State TLV:



- o If using a stream transport, the TLV MUST be sent at least once per connection, but SHOULD NOT be sent more than once.
- o If using a datagram transport, it MUST be included in every datagram that also contains a Network State TLV (Section 7.2.2) and MUST be located before any such TLV. It SHOULD also be included in any other datagram, to speed up initial peer detection.

Given the assorted transport options as well as potential endpoint configuration, a DNCP endpoint may be used in various transport modes:

Unicast:

- \* If only reliable unicast transport is used, Trickle is not used at all. Whenever the locally calculated network state hash changes, a single Network State TLV (Section 7.2.2) is sent instead to every unicast peer. Additionally, recently changed Node State TLVs (Section 7.2.3) MAY be included.
- \* If only unreliable unicast transport is used, Trickle state is kept per peer and it is used to send Network State TLVs intermittently, as specified in Section 4.3.

Multicast+Unicast: If multicast datagram transport is available on an endpoint, Trickle state is only maintained for the endpoint as a whole. It is used to send Network State TLVs periodically, as specified in Section 4.3. Additionally, per-endpoint keep-alives MAY be defined in the DNCP profile, as specified in Section 6.1.2.

MulticastListen+Unicast: Just like Unicast, except multicast transmissions are listened to in order to detect changes of the highest node identifier. This mode is used only if the DNCP profile supports dense multicast-enabled link optimization (Section 6.2).

#### 4.3. Trickle-Driven Status Updates

The Trickle algorithm [RFC6206] is used to ensure protocol reliability over unreliable multicast or unicast transports. For reliable unicast transports, its actual algorithm is unnecessary and omitted (Section 4.2). DNCP maintains multiple Trickle states as defined in Section 5. Each such state can be based on different parameters (see below) and is responsible for ensuring that a specific peer or all peers on the respective endpoint are regularly provided with the node's current locally calculated network state

hash for state comparison, i.e., to detect potential divergence in the perceived network state.

Trickle defines 3 parameters: Imin, Imax and k. Imin and Imax represent the minimum value for I and the maximum number of doublings of Imin, where I is the time interval during which at least k Trickle updates must be seen on an endpoint to prevent local state transmission. The actual suggested Trickle algorithm parameters are DNCP profile specific, as described in Section 9.

The Trickle state for all Trickle instances defined in Section 5 is considered inconsistent and reset if and only if the locally calculated network state hash changes. This occurs either due to a change in the local node's own node data, or due to receipt of more recent data from another node as explained in Section 4.1. A node MUST NOT reset its Trickle state merely based on receiving a Network State TLV (Section 7.2.2) with a network state hash which is different from its locally calculated one.

Every time a particular Trickle instance indicates that an update should be sent, the node MUST send a Network State TLV (Section 7.2.2) if and only if:

- o the endpoint is in Multicast+Unicast transport mode, in which case the TLV MUST be sent over multicast.
- o the endpoint is NOT in Multicast+Unicast transport mode, and the unicast transport is unreliable, in which case the TLV MUST be sent over unicast.

A (sub)set of all Node State TLVs (Section 7.2.3) MAY also be included, unless it is defined as undesirable for some reason by the DNCP profile, or to avoid exposure of the node state TLVs by transmitting them within insecure multicast when using secure unicast.

#### 4.4. Processing of Received TLVs

This section describes how received TLVs are processed. The DNCP profile may specify when to ignore particular TLVs, e.g., to modify security properties - see Section 9 for what may be safely defined to be ignored in a profile. Any 'reply' mentioned in the steps below denotes sending of the specified TLV(s) to the originator of the TLV being processed. All such replies MUST be sent using unicast. If the TLV being replied to was received via multicast and it was sent to a multiple access link, the reply MUST be delayed by a random timespan in  $[0, Imin/2]$ , to avoid potential simultaneous replies that may cause problems on some links, unless specified differently in the

DNCP profile. Sending of replies MAY also be rate-limited or omitted for a short period of time by an implementation. However, if the TLV is not forbidden by the DNCP profile, an implementation MUST reply to retransmissions of the TLV with a non-zero probability to avoid starvation which would break the state synchronization.

A DNCP node MUST process TLVs received from any valid (e.g., correctly scoped) address, as specified by the DNCP profile and the configuration of a particular endpoint, whether this address is known to be the address of a peer or not. This provision satisfies the needs of monitoring or other host software that needs to discover the DNCP topology without adding to the state in the network.

Upon receipt of:

- o Request Network State TLV (Section 7.1.1): The receiver MUST reply with a Network State TLV (Section 7.2.2) and a Node State TLV (Section 7.2.3) for each node data used to calculate the network state hash. The Node State TLVs SHOULD NOT contain the optional node data part to avoid redundant transmission of node data, unless explicitly specified in the DNCP profile.
- o Request Node State TLV (Section 7.1.2): If the receiver has node data for the corresponding node, it MUST reply with a Node State TLV (Section 7.2.3) for the corresponding node. The optional node data part MUST be included in the TLV.
- o Network State TLV (Section 7.2.2): If the network state hash differs from the locally calculated network state hash, and the receiver is unaware of any particular node state differences with the sender, the receiver MUST reply with a Request Network State TLV (Section 7.1.1). These replies MUST be rate limited to only at most one reply per link per unique network state hash within Imin. The simplest way to ensure this rate limit is a timestamp indicating requests, and sending at most one Request Network State TLV (Section 7.1.1) per Imin. To facilitate faster state synchronization, if a Request Network State TLV is sent in a reply, a local, current Network State TLV MAY also be sent.
- o Node State TLV (Section 7.2.3):
  - \* If the node identifier matches the local node identifier and the TLV has a greater sequence number than its current local value, or the same sequence number and a different hash, the node SHOULD re-publish its own node data with a sequence number significantly (e.g., 1000) greater than the received one, to reclaim the node identifier. This difference is needed in order to ensure that it is higher than any potentially

lingering copies of the node state in the network. This may occur normally once due to the local node restarting and not storing the most recently used sequence number. If this occurs more than once or for nodes not re-publishing their own node data, the DNCP profile MUST provide guidance on how to handle these situations as it indicates the existence of another active node with the same node identifier.

- \* If the node identifier does not match the local node identifier, and one or more of the following conditions are true:
  - + The local information is outdated for the corresponding node (local sequence number is less than that within the TLV).
  - + The local information is potentially incorrect (local sequence number matches but the node data hash differs).
  - + There is no data for that node altogether.

Then:

- + If the TLV contains the Node Data field, it SHOULD also be verified by ensuring that the locally calculated hash of the Node Data matches the content of the H(Node Data) field within the TLV. If they differ, the TLV SHOULD be ignored and not processed further.
- + If the TLV does not contain the Node Data field, and the H(Node Data) field within the TLV differs from the local node data hash for that node (or there is none), the receiver MUST reply with a Request Node State TLV (Section 7.1.2) for the corresponding node.
- + Otherwise the receiver MUST update its locally stored state for that node (node data based on Node Data field if present, sequence number and relative time) to match the received TLV.

For comparison purposes of the sequence number, a looping comparison function MUST be used to avoid problems in case of overflow. The comparison function  $a < b \Leftrightarrow ((a - b) \% (2^{32})) \& (2^{31}) \neq 0$  where  $(a \% b)$  represents the remainder of a modulo  $b$  and  $(a \& b)$  represents bitwise conjunction of  $a$  and  $b$  is RECOMMENDED unless the DNCP profile defines another.

- o Any other TLV: TLVs not recognized by the receiver MUST be silently ignored unless they are sent within another TLV (for example, TLVs within the Node Data field of a Node State TLV).

If secure unicast transport is configured for an endpoint, any Node State TLVs received over insecure multicast MUST be silently ignored.

#### 4.5. Discovering, Adding and Removing Peers

Peer relations are established between neighbors using one or more mutually connected endpoints. Such neighbors exchange information about network state and published data directly and through transitivity this information then propagates throughout the network.

New peers are discovered using the regular unicast or multicast transport defined in the DNCP profile (Section 9). This process is not distinguished from peer addition, i.e., an unknown peer is simply discovered by receiving regular DNCP protocol TLVs from it and dedicated discovery messages or TLVs do not exist. For unicast-only transports, the individual node's transport addresses are preconfigured or obtained using an external service discovery protocol. In the presence of a multicast transport, messages from unknown peers are handled in the same way as multicast messages from peers that are already known, thus new peers are simply discovered when sending their regular DNCP protocol TLVs using multicast.

When receiving a Node Endpoint TLV (Section 7.2.1) on an endpoint from an unknown peer:

- o If received over unicast, the remote node MUST be added as a peer on the endpoint and a Peer TLV (Section 7.3.1) MUST be created for it.
- o If received over multicast, the node MAY be sent a (possibly rate-limited) unicast Request Network State TLV (Section 7.1.1).

If keep-alives specified in Section 6.1 are NOT sent by the peer (either the DNCP profile does not specify the use of keep-alives or the particular peer chooses not to send keep-alives), some other existing local transport-specific means (such as Ethernet carrier-detection or TCP keep-alive) MUST be used to ensure its presence. If the peer does not send keep-alives, and no means to verify presence of the peer are available, the peer MUST be considered no longer present and it SHOULD NOT be added back as a peer until it starts sending keep-alives again. When the peer is no longer present, the Peer TLV and the local DNCP peer state MUST be removed. DNCP does not define an explicit message or TLV for indicating the termination

of DNCP operation by the terminating node, however a derived protocol could specify an extension, if the need arises.

If the local endpoint is in the Multicast-Listen+Unicast transport mode, a Peer TLV (Section 7.3.1) MUST NOT be published for the peers not having the highest node identifier.

#### 4.6. Data Liveliness Validation

Maintenance of the hash tree (Section 4.1) and thereby network state hash updates depend on up-to-date information on bidirectional node reachability derived from the contents of a topology graph. This graph changes whenever nodes are added to or removed from the network or when bidirectional connectivity between existing nodes is established or lost. Therefore the graph MUST be updated either immediately or with a small delay shorter than the DNCP profile-defined Trickle Imin, whenever:

- o A Peer TLV or a whole node is added or removed, or
- o the origination time (in milliseconds) of some node's node data is less than current time -  $2^{32} + 2^{15}$ .

The artificial upper limit for the origination time is used to gracefully avoid overflows of the origination time and allow for the node to republish its data as noted in Section 7.2.3.

The topology graph update starts with the local node marked as reachable and all other nodes marked as unreachable. Other nodes are then iteratively marked as reachable using the following algorithm: A candidate not-yet-reachable node N with an endpoint NE is marked as reachable if there is a reachable node R with an endpoint RE that meet all of the following criteria:

- o The origination time (in milliseconds) of R's node data is greater than current time -  $2^{32} + 2^{15}$ .
- o R publishes a Peer TLV with:
  - \* Peer Node Identifier = N's node identifier
  - \* Peer Endpoint Identifier = NE's endpoint identifier
  - \* Endpoint Identifier = RE's endpoint identifier
- o N publishes a Peer TLV with:
  - \* Peer Node Identifier = R's node identifier

- \* Peer Endpoint Identifier = RE's endpoint identifier
- \* Endpoint Identifier = NE's endpoint identifier

The algorithm terminates, when no more candidate nodes fulfilling these criteria can be found.

DNCP nodes that have not been reachable in the most recent topology graph traversal **MUST NOT** be used for calculation of the network state hash, be provided to any applications that need to use the whole TLV graph, or be provided to remote nodes. They **MAY** be forgotten immediately after the topology graph traversal, however it is **RECOMMENDED** to keep them at least briefly to improve the speed of DNCP network state convergence. This reduces the number of queries needed to reconverge during both initial network convergence and when a part of the network loses and regains bidirectional connectivity within that time period.

## 5. Data Model

This section describes the local data structures a minimal implementation might use. This section is provided only as a convenience for the implementor. Some of the optional extensions (Section 6) describe additional data requirements, and some optional parts of the core protocol may also require more.

A DNCP node has:

- o A data structure containing data about the most recently sent Request Network State TLVs (Section 7.1.1). The simplest option is keeping a timestamp of the most recent request (required to fulfill reply rate limiting specified in Section 4.4).

A DNCP node has for every DNCP node in the DNCP network:

- o Node identifier: the unique identifier of the node. The length, how it is produced, and how collisions are handled, is up to the DNCP profile.
- o Node data: the set of TLV tuples published by that particular node. As they are transmitted ordered (see Node State TLV (Section 7.2.3) for details), maintaining the order within the data structure here may be reasonable.
- o Latest sequence number: the 32-bit sequence number that is incremented any time the TLV set is published. The comparison function used to compare them is described in Section 4.4.

- o Origination time: the (estimated) time when the current TLV set with the current sequence number was published. It is used to populate the Milliseconds Since Origination field in a Node State TLV (Section 7.2.3). Ideally it also has millisecond accuracy.

Additionally, a DNCP node has a set of endpoints for which DNCP is configured to be used. For each such endpoint, a node has:

- o Endpoint identifier: the 32-bit opaque locally unique value identifying the endpoint within a node. It SHOULD NOT be reused immediately after an endpoint is disabled.
- o Trickle instance: the endpoint's Trickle instance with parameters I, T, and c (only on an endpoint in Multicast+Unicast transport mode).

and one (or more) of the following:

- o Interface: the assigned local network interface.
- o Unicast address: the DNCP node it should connect with.
- o Set of addresses: the DNCP nodes from which connections are accepted.

For each remote (peer, endpoint) pair detected on a local endpoint, a DNCP node has:

- o Node identifier: the unique identifier of the peer.
- o Endpoint identifier: the unique endpoint identifier used by the peer.
- o Peer address: the most recently used address of the peer (authenticated and authorized, if security is enabled).
- o Trickle instance: the particular peer's Trickle instance with parameters I, T, and c (only on an endpoint in Unicast mode, when using an unreliable unicast transport) .

## 6. Optional Extensions

This section specifies extensions to the core protocol that a DNCP profile may specify to be used.



## 6.1. Keep-Alives

While DNCP provides mechanisms for discovery and adding of new peers on an endpoint (Section 4.5), as well as state change notifications, another mechanism may be needed to get rid of old, no longer valid peers if the transport or lower layers do not provide one as noted in Section 4.6.

If keep-alives are not specified in the DNCP profile, the rest of this subsection MUST be ignored.

A DNCP profile MAY specify either per-endpoint (sent using multicast to all DNCP nodes connected to a multicast-enabled link) or per-peer (sent using unicast to each peer individually) keep-alive support.

For every endpoint that a keep-alive is specified for in the DNCP profile, the endpoint-specific keep-alive interval MUST be maintained. By default, it is DNCP\_KEEPA\_LIVE\_INTERVAL. If there is a local value that is preferred for that for any reason (configuration, energy conservation, media type, ..), it can be substituted instead. If a non-default keep-alive interval is used on any endpoint, a DNCP node MUST publish appropriate Keep-Alive Interval TLV(s) (Section 7.3.2) within its node data.

### 6.1.1. Data Model Additions

The following additions to the Data Model (Section 5) are needed to support keep-alives:

For each configured endpoint that has per-endpoint keep-alives enabled:

- o Last sent: If a timestamp which indicates the last time a Network State TLV (Section 7.2.2) was sent over that interface.

For each remote (peer, endpoint) pair detected on a local endpoint, a DNCP node has:

- o Last contact timestamp: a timestamp which indicates the last time a consistent Network State TLV (Section 7.2.2) was received from the peer over multicast, or anything was received over unicast. Failing to update it for a certain amount of time as specified in Section 6.1.5 results in the removal of the peer. When adding a new peer, it is initialized to the current time.
- o Last sent: If per-peer keep-alives are enabled, a timestamp which indicates the last time a Network State TLV (Section 7.2.2) was

sent to to that point-to-point peer. When adding a new peer, it is initialized to the current time.

#### 6.1.2. Per-Endpoint Periodic Keep-Alives

If per-endpoint keep-alives are enabled on an endpoint in Multicast+Unicast transport mode, and if no traffic containing a Network State TLV (Section 7.2.2) has been sent to a particular endpoint within the endpoint-specific keep-alive interval, a Network State TLV (Section 7.2.2) MUST be sent on that endpoint, and a new Trickle interval started, as specified in the step 2 of Section 4.2 of [RFC6206]. The actual sending time SHOULD be further delayed by a random timespan in  $[0, I_{min}/2]$ .

#### 6.1.3. Per-Peer Periodic Keep-Alives

If per-peer keep-alives are enabled on a unicast-only endpoint, and if no traffic containing a Network State TLV (Section 7.2.2) has been sent to a particular peer within the endpoint-specific keep-alive interval, a Network State TLV (Section 7.2.2) MUST be sent to the peer, and a new Trickle interval started, as specified in the step 2 of Section 4.2 of [RFC6206].

#### 6.1.4. Received TLV Processing Additions

If a TLV is received over unicast from the peer, the Last contact timestamp for the peer MUST be updated.

On receipt of a Network State TLV (Section 7.2.2) which is consistent with the locally calculated network state hash, the Last contact timestamp for the peer MUST be updated in order to maintain it as a peer.

#### 6.1.5. Peer Removal

For every peer on every endpoint, the endpoint-specific keep-alive interval must be calculated by looking for Keep-Alive Interval TLVs (Section 7.3.2) published by the node, and if none exist, using the default value of `DNCP_KEEPAIVE_INTERVAL`. If the peer's Last contact timestamp has not been updated for at least locally chosen potentially endpoint-specific keep-alive multiplier (defaults to `DNCP_KEEPAIVE_MULTIPLIER`) times the peer's endpoint-specific keep-alive interval, the Peer TLV for that peer and the local DNCP peer state MUST be removed.

## 6.2. Support For Dense Multicast-Enabled Links

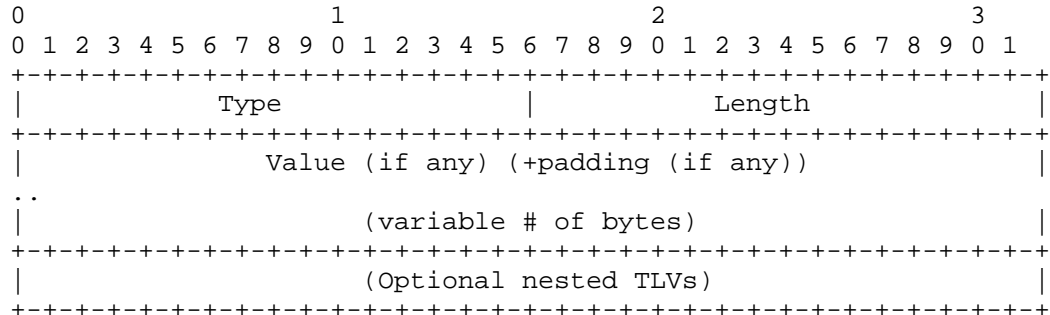
This optimization is needed to avoid a state space explosion. Given a large set of DNCP nodes publishing data on an endpoint that uses multicast on a link, every node will add a Peer TLV (Section 7.3.1) for each peer. While Trickle limits the amount of traffic on the link in stable state to some extent, the total amount of data that is added to and maintained in the DNCP network given  $N$  nodes on a multicast-enabled link is  $O(N^2)$ . Additionally if per-peer keep-alives are used, there will be  $O(N^2)$  keep-alives running on the link if liveliness of peers is not ensured using some other way (e.g., TCP connection lifetime, layer 2 notification, per-endpoint keep-alive).

An upper bound for the number of peers that are allowed for a particular type of link that an endpoint in Multicast+Unicast transport mode is used on SHOULD be provided by a DNCP profile, but MAY also be chosen at runtime. The main consideration when selecting a bound (if any) for a particular type of link should be whether it supports multicast traffic, and whether a too large number of peers case is likely to happen during the use of that DNCP profile on that particular type of link. If neither is likely, there is little point specifying support for this for that particular link type.

If a DNCP profile does not support this extension at all, the rest of this subsection MUST be ignored. This is because when this extension is used, the state within the DNCP network only contains a subset of the full topology of the network. Therefore every node must be aware of the potential of it being used in a particular DNCP profile.

If the specified upper bound is exceeded for some endpoint in Multicast+Unicast transport mode and if the node does not have the highest node identifier on the link, it SHOULD treat the endpoint as a unicast endpoint connected to the node that has the highest node identifier detected on the link, therefore transitioning to Multicast-listen+Unicast transport mode. See Section 4.2 for implications on the specific endpoint behavior. The nodes in Multicast-listen+Unicast transport mode MUST keep listening to multicast traffic to both receive messages from the node(s) still in Multicast+Unicast mode, and as well to react to nodes with a greater node identifier appearing. If the highest node identifier present on the link changes, the remote unicast address of the endpoints in Multicast-Listen+Unicast transport mode MUST be changed. If the node identifier of the local node is the highest one, the node MUST switch back to, or stay in Multicast+Unicast mode, and form peer relationships with all peers as specified in Section 4.5.

## 7. Type-Length-Value Objects



Each TLV is encoded as:

- o a 2 byte Type field
- o a 2 byte Length field which contains the length of the Value field in bytes; 0 means no Value
- o the Value itself (if any)
- o padding bytes with value of zero up to the next 4 byte boundary if the Length is not divisible by 4.

While padding bytes MUST NOT be included in the number stored in the Length field of the TLV, if the TLV is enclosed within another TLV, then the padding is included in the enclosing TLV's Length value.

Each TLV which does not define optional fields or variable-length content MAY be sent with additional sub-TLVs appended after the TLV to allow for extensibility. When handling such TLV types, each node MUST accept received TLVs that are longer than the fixed fields specified for the particular type, and ignore the sub-TLVs with either unknown types, or not supported within that particular TLV type. If any sub-TLVs are present, the Length field of the TLV describes the number of bytes from the first byte of the TLV's own Value (if any) to the last (padding) byte of the last sub-TLV.

For example, type=123 (0x7b) TLV with value 'x' (120 = 0x78) is encoded as: 007B 0001 7800 0000. If it were to have sub-TLV of type=124 (0x7c) with value 'y', it would be encoded as 007B 000C 7800 0000 007C 0001 7900 0000.

In this section, the following special notation is used:

.. = octet string concatenation operation.

$H(x)$  = non-cryptographic hash function specified by DNCP profile.

## 7.1. Request TLVs

### 7.1.1. Request Network State TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Type: REQ-NETWORK-STATE (1) | Length: >= 0 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

This TLV is used to request response with a Network State TLV (Section 7.2.2) and all Node State TLVs (Section 7.2.3) (without node data).

### 7.1.2. Request Node State TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Type: REQ-NODE-STATE (2) | Length: > 0 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Node Identifier                               |
|                               (length fixed in DNCP profile)                 |
| ...                           |
|                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

This TLV is used to request a Node State TLV (Section 7.2.3) (including node data) for the node with the matching node identifier.

## 7.2. Data TLVs

### 7.2.1. Node Endpoint TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Type: NODE-ENDPOINT (3) | Length: > 4 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Node Identifier                               |
|                               (length fixed in DNCP profile)                 |
| ...                           |
|                               |
|                               Endpoint Identifier                             |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

This TLV identifies both the local node's node identifier, as well as the particular endpoint's endpoint identifier. Section 4.2 specifies when it is sent.

#### 7.2.2. Network State TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Type: NETWORK-STATE (4)   |   Length: > 0   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   H(sequence number of node 1 .. H(node data of node 1) ..
|   .. sequence number of node N .. H(node data of node N))
|   (length fixed in DNCP profile)
|
...
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

This TLV contains the current locally calculated network state hash, see Section 4.1 for how it is calculated.

#### 7.2.3. Node State TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Type: NODE-STATE (5)   |   Length: > 8   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Node Identifier
|   (length fixed in DNCP profile)
|
...
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Sequence Number
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Milliseconds Since Origination
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               H(Node Data)
|   (length fixed in DNCP profile)
|
...
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   (optionally) Node Data (a set of nested TLVs)
|
...
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

This TLV represents the local node's knowledge about the published state of a node in the DNCP network identified by the Node Identifier field in the TLV.

Every node, including the node publishing the node data, MUST update the Milliseconds Since Origination whenever it sends a Node State TLV based on when the node estimates the data was originally published. This is, e.g., to ensure that any relative timestamps contained within the published node data can be correctly offset and interpreted. Ultimately, what is provided is just an approximation, as transmission delays are not accounted for.

Absent any changes, if the originating node notices that the 32-bit milliseconds since origination value would be close to overflow (greater than  $2^{32}-2^{16}$ ), the node MUST re-publish its TLVs even if there is no change. In other words, absent any other changes, the TLV set MUST be re-published roughly every 48 days.

The actual node data of the node may be included within the TLV as well in the optional Node Data field. The set of TLVs MUST be strictly ordered based on ascending binary content (including TLV type and length). This enables, e.g., efficient state delta processing and no-copy indexing by TLV type by the recipient. The Node Data content MUST be passed along exactly as it was received. It SHOULD be also verified on receipt that the locally calculated  $H(\text{Node Data})$  matches the content of the field within the TLV, and if the hash differs, the TLV SHOULD be ignored.

### 7.3. Data TLVs within Node State TLV

These TLVs are published by the DNCP nodes, and therefore only encoded in the Node Data field of Node State TLVs. If encountered outside Node State TLV, they MUST be silently ignored.

#### 7.3.1. Peer TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|          Type: PEER (8)          |          Length: > 8          |
+-----+-----+-----+-----+-----+-----+-----+-----+
|          Peer Node Identifier          |
|          (length fixed in DNCP profile)          |
|          ...          |
+-----+-----+-----+-----+-----+-----+-----+-----+
|          Peer Endpoint Identifier          |
+-----+-----+-----+-----+-----+-----+-----+-----+
|          (Local) Endpoint Identifier          |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

This TLV indicates that the node in question vouches that the specified peer is reachable by it on the specified local endpoint.

The presence of this TLV at least guarantees that the node publishing it has received traffic from the peer recently. For guaranteed up-to-date bidirectional reachability, the existence of both nodes' matching Peer TLVs needs to be checked.

### 7.3.2. Keep-Alive Interval TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Type: KEEP-ALIVE-INTERVAL (9) | Length: >= 8 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Endpoint Identifier                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Interval                                         |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

This TLV indicates a non-default interval being used to send keep-alives specified in Section 6.1.

Endpoint identifier is used to identify the particular (local) endpoint for which the interval applies on the sending node. If 0, it applies for ALL endpoints for which no specific TLV exists.

Interval specifies the interval in milliseconds at which the node sends keep-alives. A value of zero means no keep-alives are sent at all; in that case, some lower layer mechanism that ensures presence of nodes MUST be available and used.

## 8. Security and Trust Management

If specified in the DNCP profile, either DTLS [RFC6347] or TLS [RFC5246] may be used to authenticate and encrypt either some (if specified optional in the profile), or all unicast traffic. The following methods for establishing trust are defined, but it is up to the DNCP profile to specify which ones may, should or must be supported.

### 8.1. Pre-Shared Key Based Trust Method

A PSK-based trust model is a simple security management mechanism that allows an administrator to deploy devices to an existing network by configuring them with a pre-defined key, similar to the configuration of an administrator password or WPA-key. Although limited in nature it is useful to provide a user-friendly security mechanism for smaller networks.



## 8.2. PKI Based Trust Method

A PKI-based trust-model enables more advanced management capabilities at the cost of increased complexity and bootstrapping effort. It however allows trust to be managed in a centralized manner and is therefore useful for larger networks with a need for an authoritative trust management.

## 8.3. Certificate Based Trust Consensus Method

For some scenarios - such as bootstrapping a mostly unmanaged network - the methods described above may not provide a desirable tradeoff between security and user experience. This section includes guidance for implementing an opportunistic security [RFC7435] method which DNCP profiles can build upon and adapt for their specific requirements.

The certificate-based consensus model is designed to be a compromise between trust management effort and flexibility. It is based on X.509-certificates and allows each DNCP node to provide a trust verdict on any other certificate and a consensus is found to determine whether a node using this certificate or any certificate signed by it is to be trusted.

A DNCP node not using this security method **MUST** ignore all announced trust verdicts and **MUST NOT** announce any such verdicts by itself, i.e., any other normative language in this subsection does not apply to it.

The current effective trust verdict for any certificate is defined as the one with the highest priority from all trust verdicts announced for said certificate at the time.

### 8.3.1. Trust Verdicts

Trust verdicts are statements of DNCP nodes about the trustworthiness of X.509-certificates. There are 5 possible trust verdicts in order of ascending priority:

0 (Neutral): no trust verdict exists but the DNCP network should determine one.

1 (Cached Trust): the last known effective trust verdict was Configured or Cached Trust.

2 (Cached Distrust): the last known effective trust verdict was Configured or Cached Distrust.

3 (Configured Trust): trustworthy based upon an external ceremony or configuration.

4 (Configured Distrust): not trustworthy based upon an external ceremony or configuration.

Trust verdicts are differentiated in 3 groups:

- o Configured verdicts are used to announce explicit trust verdicts a node has based on any external trust bootstrap or predefined relation a node has formed with a given certificate.
- o Cached verdicts are used to retain the last known trust state in case all nodes with configured verdicts about a given certificate have been disconnected or turned off.
- o The Neutral verdict is used to announce a new node intending to join the network so a final verdict for it can be found.

The current effective trust verdict for any certificate is defined as the one with the highest priority within the set of trust verdicts announced for the certificate in the DNCP network. A node **MUST** be trusted for participating in the DNCP network if and only if the current effective trust verdict for its own certificate or any one in its certificate hierarchy is (Cached or Configured) Trust and none of the certificates in its hierarchy have an effective trust verdict of (Cached or Configured) Distrust. In case a node has a configured verdict, which is different from the current effective trust verdict for a certificate, the current effective trust verdict takes precedence in deciding trustworthiness. Despite that, the node still retains and announces its configured verdict.

#### 8.3.2. Trust Cache

Each node **SHOULD** maintain a trust cache containing the current effective trust verdicts for all certificates currently announced in the DNCP network. This cache is used as a backup of the last known state in case there is no node announcing a configured verdict for a known certificate. It **SHOULD** be saved to a non-volatile memory at reasonable time intervals to survive a reboot or power outage.

Every time a node (re)joins the network or detects the change of an effective trust verdict for any certificate, it will synchronize its cache, i.e., store new effective trust verdicts overwriting any previously cached verdicts. Configured verdicts are stored in the cache as their respective cached counterparts. Neutral verdicts are never stored and do not override existing cached verdicts.

### 8.3.3. Announcement of Verdicts

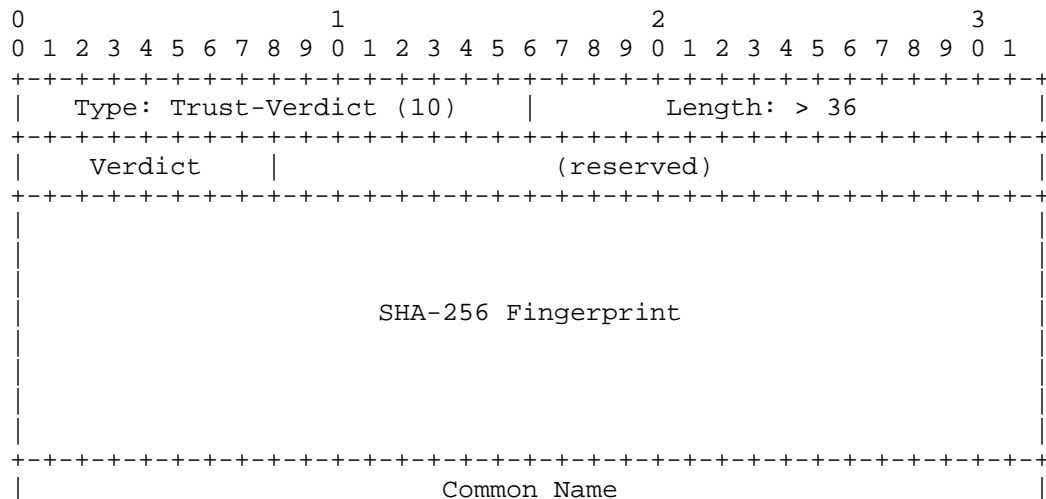
A node SHOULD always announce any configured trust verdicts it has established by itself, and it MUST do so if announcing the configured trust verdict leads to a change in the current effective trust verdict for the respective certificate. In absence of configured verdicts, it MUST announce cached trust verdicts it has stored in its trust cache, if one of the following conditions applies:

- o The stored trust verdict is Cached Trust and the current effective trust verdict for the certificate is Neutral or does not exist.
- o The stored trust verdict is Cached Distrust and the current effective trust verdict for the certificate is Cached Trust.

A node rechecks these conditions whenever it detects changes of announced trust verdicts anywhere in the network.

Upon encountering a node with a hierarchy of certificates for which there is no effective trust verdict, a node adds a Neutral Trust-Verdict-TLV to its node data for all certificates found in the hierarchy, and publishes it until an effective trust verdict different from Neutral can be found for any of the certificates, or a reasonable amount of time (10 minutes is suggested) with no reaction and no further authentication attempts has passed. Such trust verdicts SHOULD also be limited in rate and number to prevent denial-of-service attacks.

Trust verdicts are announced using Trust-Verdict TLVs:



Verdict represents the numerical index of the trust verdict.

(reserved) is reserved for future additions and MUST be set to 0 when creating TLVs and ignored when parsing them.

SHA-256 Fingerprint contains the SHA-256 [RFC6234] hash value of the certificate in DER-format.

Common Name contains the variable-length (1-64 bytes) common name of the certificate.

#### 8.3.4. Bootstrap Ceremonies

The following non-exhaustive list of methods describes possible ways to establish trust relationships between DNCP nodes and node certificates. Trust establishment is a two-way process in which the existing network must trust the newly added node and the newly added node must trust at least one of its peer nodes. It is therefore necessary that both the newly added node and an already trusted node perform such a ceremony to successfully introduce a node into the DNCP network. In all cases an administrator MUST be provided with external means to identify the node belonging to a certificate based on its fingerprint and a meaningful common name.

##### 8.3.4.1. Trust by Identification

A node implementing certificate-based trust MUST provide an interface to retrieve the current set of effective trust verdicts, fingerprints and names of all certificates currently known and set configured trust verdicts to be announced. Alternatively it MAY provide a companion DNCP node or application with these capabilities with which it has a pre-established trust relationship.

##### 8.3.4.2. Preconfigured Trust

A node MAY be preconfigured to trust a certain set of node or CA certificates. However such trust relationships MUST NOT result in unwanted or unrelated trust for nodes not intended to be run inside the same network (e.g., all other devices by the same manufacturer).

##### 8.3.4.3. Trust on Button Press

A node MAY provide a physical or virtual interface to put one or more of its internal network interfaces temporarily into a mode in which it trusts the certificate of the first DNCP node it can successfully establish a connection with.

#### 8.3.4.4. Trust on First Use

A node which is not associated with any other DNCP node MAY trust the certificate of the first DNCP node it can successfully establish a connection with. This method MUST NOT be used when the node has already associated with any other DNCP node.

### 9. DNCP Profile-Specific Definitions

Each DNCP profile MUST specify the following aspects:

- o Unicast and optionally multicast transport protocol(s) to be used. If multicast-based node and status discovery is desired, a datagram-based transport supporting multicast has to be available.
- o How the chosen transport(s) are secured: Not at all, optionally or always with the TLS scheme defined here using one or more of the methods, or with something else. If the links with DNCP nodes can be sufficiently secured or isolated, it is possible to run DNCP in a secure manner without using any form of authentication or encryption.
- o Transport protocols' parameters such as port numbers to be used, or multicast address to be used. Unicast, multicast, and secure unicast may each require different parameters, if applicable.
- o When receiving TLVs, what sort of TLVs are ignored in addition - as specified in Section 4.4 - e.g., for security reasons. While the security of the node data published within the Node State TLVs is already ensured by the base specification (if secure mode is enabled, Node State TLVs are sent only via unicast as multicast ones are ignored on receipt), if a profile adds TLVs that are sent outside the node data, a profile should indicate whether or not those TLVs should be ignored if they are received via multicast or non-secured unicast. A DNCP profile may define the following DNCP TLVs to be safely ignored:
  - \* Anything received over multicast, except Node Endpoint TLV (Section 7.2.1) and Network State TLV (Section 7.2.2).
  - \* Any TLVs received over unreliable unicast or multicast at too high rate; Trickle will ensure eventual convergence given the rate slows down at some point.
- o How to deal with node identifier collision as described in Section 4.4. Main options are either for one or both nodes to assign new node identifiers to themselves, or to notify someone about a fatal error condition in the DNCP network.

- o Imin, Imax and k ranges to be suggested for implementations to be used in the Trickle algorithm. The Trickle algorithm does not require these to be the same across all implementations for it to work, but similar orders of magnitude helps implementations of a DNCP profile to behave more consistently and to facilitate estimation of lower and upper bounds for convergence behavior of the network.
- o Hash function H(x) to be used, and how many bits of the output are actually used. The chosen hash function is used to handle both hashing of node specific data, and network state hash, which is a hash of node specific data hashes. SHA-256 defined in [RFC6234] is the recommended default choice, but a non-cryptographic hash function could be used as well.
- o DNCP\_NODE\_IDENTIFIER\_LENGTH: The fixed length of a node identifier (in bytes).
- o Whether to send keep-alives, and if so, whether per-endpoint (requires multicast transport), or per-peer. Keep-alive has also associated parameters:
  - \* DNCP\_KEEPLIVE\_INTERVAL: How often keep-alives are to be sent by default (if enabled).
  - \* DNCP\_KEEPLIVE\_MULTIPLIER: How many times the DNCP\_KEEPLIVE\_INTERVAL (or peer-supplied keep-alive interval value) a node may not be heard from to be considered still valid. This is just a default used in absence of any other configuration information, or particular per-endpoint configuration.
- o Whether to support dense multicast-enabled link optimization (Section 6.2) or not.

For some guidance on choosing transport and security options, please see Appendix B.

## 10. Security Considerations

DNCP-based protocols may use multicast to indicate DNCP state changes and for keep-alive purposes. However, no actual published data TLVs will be sent across that channel. Therefore an attacker may only learn hash values of the state within DNCP and may be able to trigger unicast synchronization attempts between nodes on a local link this way. A DNCP node MUST therefore rate-limit its reactions to multicast packets.

When using DNCP to bootstrap a network, PKI based solutions may have issues when validating certificates due to potentially unavailable accurate time, or due to inability to use the network to either check Certificate Revocation Lists or perform on-line validation.

The Certificate-based trust consensus mechanism defined in this document allows for a consenting revocation, however in case of a compromised device the trust cache may be poisoned before the actual revocation happens allowing the distrusted device to rejoin the network using a different identity. Stopping such an attack might require physical intervention and flushing of the trust caches.

## 11. IANA Considerations

IANA should set up a registry for the (decimal 16-bit) "DNCP TLV Types" under "Distributed Node Consensus Protocol (DNCP)", with the following initial contents: ([RFC Editor: please remove] ideally as <http://www.iana.org/assignments/dncp-registry>)

- 0: Reserved
- 1: Request network state
- 2: Request node state
- 3: Node endpoint
- 4: Network state
- 5: Node state
- 6: Reserved (was: Custom)
- 7: Reserved (was: Fragment count)
- 8: Peer
- 9: Keep-alive interval
- 10: Trust-Verdict
- 11-31: Free - policy of standards action [RFC5226] should be used
- 32-511: Reserved for per-DNCP profile use
- 512-767: Free - policy of standards action [RFC5226] should be used

768-1023: Private use [RFC5226]

1024-65535: Reserved for future protocol evolution (for example, DNCP version 2)

## 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.
- [RFC6206] Levis, P., Clausen, T., Hui, J., Gnawali, O., and J. Ko, "The Trickle Algorithm", RFC 6206, March 2011.
- [RFC6234] Eastlake, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, May 2011.
- [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>>.

### 12.2. Informative references

- [RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, February 2003.
- [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.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <<http://www.rfc-editor.org/info/rfc7435>>.
- [I-D.ietf-homenet-prefix-assignment] Pfister, P., Paterson, B., and J. Arkko, "Distributed Prefix Assignment Algorithm", draft-ietf-homenet-prefix-assignment-08 (work in progress), August 2015.



## Appendix A. Alternative Modes of Operation

Beyond what is described in the main text, the protocol allows for other uses. These are provided as examples.

### A.1. Read-only Operation

If a node uses just a single endpoint and does not need to publish any TLVs, full DNCP node functionality is not required. Such limited node can acquire and maintain view of the TLV space by implementing the processing logic as specified in Section 4.4. Such node would not need Trickle, peer-maintenance or even keep-alives at all, as the DNCP nodes' use of it would guarantee eventual receipt of network state hashes, and synchronization of node data, even in presence of unreliable transport.

### A.2. Forwarding Operation

If a node with a pair of endpoints does not need to publish any TLVs, it can detect (for example) nodes with the highest node identifier on each of the endpoints (if any). Any TLVs received from one of them would be forwarded verbatim as unicast to the other node with highest node identifier.

Any tinkering with the TLVs would remove guarantees of this scheme working; however passive monitoring would obviously be fine. This type of simple forwarding cannot be chained, as it does not send anything proactively.

## Appendix B. DNCP Profile Additional Guidance

This appendix explains implications of design choices made when specifying DNCP profile to use particular transport or security options.

### B.1. Unicast Transport - UDP or TCP?

The node data published by a DNCP node is limited to 64KB due to the 16-bit size of the length field of the TLV it is published within. Some transport choices may decrease this limit; if using e.g. UDP datagrams for unicast transport the upper bound of node data size is whatever the nodes and the underlying network can pass to each other as DNCP does not define its own fragmentation scheme. A profile which chooses UDP has to be limited to small node data (e.g. somewhat smaller than IPv6 default MTU if using IPv6), or specify a minimum which all nodes have to support. Even then, if using non-link-local communications, there is some concern about what middleboxes do to fragmented packets. Therefore, the use of stream transport such as

TCP is probably a good idea if either non-link-local communication is desired, or fragmentation is expected to cause problems.

TCP also provides some other facilities, such as a relatively long built-in keep-alive which in conjunction with connection closes occurring from eventual failed retransmissions may be sufficient to avoid the use of in-protocol keep-alive defined in Section 6.1. Additionally it is reliable, so there is no need for Trickle on such unicast connections.

The major downside of using TCP instead of UDP with DNCP-based profiles lies in the loss of control over the time at which TLVs are received; while unreliable UDP datagrams also have some delay, TLVs within reliable stream transport may be delayed significantly due to retransmissions. This is not a problem if no relative time dependent information is stored within the TLVs in the DNCP-based protocol; for such a protocol, TCP is a reasonable choice for unicast transport if it is available.

#### B.2. (Optional) Multicast Transport

Multicast is needed for dynamic peer discovery and to trigger unicast exchanges; for that, unreliable datagram transport (=typically UDP) is the only transport option defined within this specification, although DNCP-based protocols may themselves define some other transport or peer discovery mechanism (e.g. based on mDNS or DNS).

If multicast is used, a well-known address should be specified, and for e.g. IPv6 respectively the desired address scopes. In most cases link-local and possibly site-local are useful scopes.

#### B.3. (Optional) Transport Security

In terms of provided security, DTLS and TLS are equivalent; they also consume similar amount of state on the devices. While TLS is on top of a stream protocol, using DTLS also requires relatively long session caching within the DTLS layer to avoid expensive re-authentication/authorization steps if and when any state within the DNCP network changes or per-peer keep-alive (if enabled) is sent.

TLS implementations (at the time of the writing of the specification) seem more mature and available (as open source) than DTLS ones. This may be due to a long history of use with HTTPS.

Some libraries seem not to support multiplexing between insecure and secure communication on the same port, so specifying distinct ports for secured and unsecured communication may be beneficial.

## Appendix C. Example Profile

This is the DNCP profile of SHSP, an experimental (and for the purposes of this document fictional) home automation protocol. The protocol itself is used to make key-value store published by each of the nodes available to all other nodes for distributed monitoring and control of a home infrastructure. It defines only one additional TLV type: a key=value TLV which contains a single key=value assignment for publication.

- o Unicast transport: IPv6 TCP on port EXAMPLE-P1 since only absolute timestamps are used within the key=value data and since it focuses primarily on Linux-based nodes which support both protocols well. Connections from and to non-link-local addresses are ignored to avoid exposing this protocol outside the secure links.
- o Multicast transport: IPv6 UDP on port EXAMPLE-P2 to link-local scoped multicast address ff02:EXAMPLE. At least one node per link in the home is assumed to facilitate node discovery without depending on any other infrastructure.
- o Security: None. It is to be used only on trusted links (WPA2-x wireless, physically secure wired links).
- o Additional TLVs to be ignored: None. No DNCP security is specified, and no new TLVs are defined outside of node data.
- o Node identifier length (DNCP\_NODE\_IDENTIFIER\_LENGTH): 32 bits that are randomly generated.
- o Node identifier collision handling: Pick new random node identifier.
- o Trickle parameters: Imin = 200ms, Imax = 7, k = 1. It means at least one multicast per link in 25 seconds in stable state ( $0.2 * 2^7$ ).
- o Hash function H(x) + length: SHA-256, only 128 bits used. Relatively fast, and 128 bits should be plenty to prevent random conflicts (64 bits would most likely be sufficient, too).
- o No in-protocol keep-alives (Section 6.1); TCP keep-alive is to be used. In practice TCP keep-alive is seldom encountered anyway as changes in network state cause packets to be sent on the unicast connections, and those that fail sufficiently many retransmissions are dropped much before keep-alive actually would fire.

- o No support for dense multicast-enabled link optimization (Section 6.2); SHSP is a simple protocol for few nodes (network-wide, not even to mention on a single link), and therefore would not provide any benefit.

#### Appendix D. Some Questions and Answers [RFC Editor: please remove]

Q: 32-bit endpoint id?

A: Here, it would save 32 bits per peer if it was 16 bits (and less is not realistic). However, TLVs defined elsewhere would not seem to even gain that much on average. 32 bits is also used for ifindex in various operating systems, making for simpler implementation.

Q: Why have topology information at all?

A: It is an alternative to the more traditional seq#/TTL-based flooding schemes. In steady state, there is no need to, e.g., re-publish every now and then.

#### Appendix E. Changelog [RFC Editor: please remove]

draft-ietf-homenet-dncp-10:

- o Added profile guidance section, as well as example profile.

draft-ietf-homenet-dncp-09:

- o Reserved 1024+ TLV types for future versions (=versioning mechanism); private use section moved from 192-255 to 512-767.
- o Added applicability statement and clarified some text based on reviews.

draft-ietf-homenet-dncp-08:

- o Removed fragmentation as it is somewhat underspecified and unimplemented. It may be specified in some future extension draft or new version of DNCP.
- o Added generic sub-TLV extensibility mechanism.

draft-ietf-homenet-dncp-06:

- o Removed custom TLV.
- o Made keep-alive multipliers local implementation choice, profiles just provide guidance on sane default value.

- o Removed the DNCP\_GRACE\_INTERVAL as it is really implementation choice.
- o Simplified the suggested structures in data model.
- o Reorganized the document and provided an overview section.

draft-ietf-homenet-dncp-04:

- o Added mandatory rate limiting for network state requests, and optional slightly faster convergence mechanism by including current local network state in the remote network state requests.

draft-ietf-homenet-dncp-03:

- o Renamed connection -> endpoint.
- o !!! Backwards incompatible change: Renumbered TLVs, and got rid of node data TLV; instead, node data TLV's contents are optionally within node state TLV.

draft-ietf-homenet-dncp-02:

- o Changed DNCP "messages" into series of TLV streams, allowing optimized round-trip saving synchronization.
- o Added fragmentation support for bigger node data and for chunking in absence of reliable L2 and L3 fragmentation.

draft-ietf-homenet-dncp-01:

- o Fixed keep-alive semantics to consider unicast requests also updates of most recently consistent, and added proactive unicast request to ensure even inconsistent keep-alive messages eventually triggering consistency timestamp update.
- o Facilitated (simple) read-only clients by making Node Connection TLV optional if just using DNCP for read-only purposes.
- o Added text describing how to deal with "dense" networks, but left actual numbers and mechanics up to DNCP profiles and (local) configurations.

draft-ietf-homenet-dncp-00: Split from pre-version of draft-ietf-homenet-hncp-03 generic parts. Changes that affect implementations:

- o TLVs were renumbered.

- o TLV length does not include header (=-4). This facilitates, e.g., use of DHCPv6 option parsing libraries (same encoding), and reduces complexity (no need to handle error values of length less than 4).
- o Trickle is reset only when locally calculated network state hash is changes, not as remote different network state hash is seen. This prevents, e.g., attacks by multicast with one multicast packet to force Trickle reset on every interface of every node on a link.
- o Instead of 'ping', use 'keep-alive' (optional) for dead peer detection. Different message used!

#### Appendix F. 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 G. Acknowledgements

Thanks to Ole Troan, Pierre Pfister, Mark Baugher, Mark Townsley, Juliusz Chroboczek, Jiazi Yi, Mikael Abrahamsson, Brian Carpenter, Thomas Clausen, DENG Hui and Margaret Cullen for their contributions to the draft.

Thanks to Kaiwen Jin and Xavier Bonnetain for their related research work.

#### Authors' Addresses

Markus Stenberg  
Independent  
Helsinki 00930  
Finland

Email: markus.stenberg@iki.fi

Steven Barth  
Independent  
Halle 06114  
Germany

Email: cyrus@openwrt.org

HOMENET  
Internet-Draft  
Intended status: Standards Track  
Expires: March 26, 2016

D. Migault (Ed)  
Ericsson  
R. Weber  
Nominum  
R. Hunter  
Globis Consulting BV  
C. Griffiths

W. Cloetens  
SoftAtHome  
September 23, 2015

Outsourcing Home Network Authoritative Naming Service  
draft-ietf-homenet-front-end-naming-delegation-04.txt

## Abstract

RFC7368 'IPv6 Home Networking Architecture Principles' section 3.7 describes architecture principles related to naming and service discovery in residential home networks.

Customer Edge Routers and other Customer Premises Equipment (CPEs) are designed to provide IP connectivity to home networks. Most CPEs assign IP addresses to the nodes of the home network which makes them good candidates for hosting the naming service. IPv6 provides global connectivity, and nodes from the home network will be reachable from the global Internet. As a result, the naming service is expected to be exposed on the Internet.

However, CPEs have not been designed to host such a naming service exposed on the Internet. Running a naming service visible on the Internet may expose the CPEs to resource exhaustion and other attacks, which could make the home network unreachable, and most probably would also affect the internal communications of the home network.

In addition, regular end users may not understand, or possess the necessary skills to be able to perform, DNSSEC management and configuration. Misconfiguration may also result in naming service disruption, thus these end users may prefer to rely on third party name service providers.

This document describes a homenet naming architecture, where the CPEs manage the DNS zones associated with its own home network, and outsource elements of the naming service (possibly including DNSSEC management) to a third party running on the Internet.

## 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 March 26, 2016.

## Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Requirements notation . . . . .	3
2. Introduction . . . . .	3
2.1. Scope of the Document . . . . .	4
3. Terminology . . . . .	5
4. Architecture Description . . . . .	7
4.1. Architecture Overview . . . . .	7
4.2. Example: Homenet Zone . . . . .	9
4.3. Example: CPE necessary parameters for outsourcing . . . . .	11
5. Synchronization between CPE and the Synchronization Server . . . . .	12
5.1. Synchronization with a Hidden Primary . . . . .	12
5.2. Securing Synchronization . . . . .	13
5.3. CPE Security Policies . . . . .	15
6. DNSSEC compliant Homenet Architecture . . . . .	15
6.1. Zone Signing . . . . .	15



6.2. Secure Delegation . . . . .	17
7. Handling Different Views . . . . .	18
7.1. Misleading Reasons for Local Scope DNS Zone . . . . .	18
7.2. Consequences . . . . .	19
7.3. Guidance and Recommendations . . . . .	19
8. Homenet Reverse Zone . . . . .	20
9. Renumbering . . . . .	20
9.1. Hidden Primary . . . . .	21
9.2. Synchronization Server . . . . .	22
10. Privacy Considerations . . . . .	23
11. Security Considerations . . . . .	23
11.1. Names are less secure than IP addresses . . . . .	23
11.2. Names are less volatile than IP addresses . . . . .	24
11.3. DNS Reflection Attacks . . . . .	24
11.3.1. Reflection Attack involving the Hidden Primary . . . . .	24
11.3.2. Reflection Attacks involving the Synchronization Server . . . . .	26
11.3.3. Reflection Attacks involving the Public Authoritative Servers . . . . .	27
11.4. Flooding Attack . . . . .	27
11.5. Replay Attack . . . . .	27
12. IANA Considerations . . . . .	28
13. Acknowledgment . . . . .	28
14. References . . . . .	28
14.1. Normative References . . . . .	28
14.2. Informational References . . . . .	31
Appendix A. Document Change Log . . . . .	32
Authors' Addresses . . . . .	34

## 1. Requirements notation

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].

## 2. Introduction

IPv6 provides global end to end IP reachability. End users prefer to use names instead of long and complex IPv6 addresses when accessing services hosted in the home network.

Customer Edge Routers and other Customer Premises Equipment (CPEs) are already providing IPv6 connectivity to the home network, and generally provide IPv6 addresses or prefixes to the nodes of the home network. This makes CPEs good candidates to manage the binding between names and IP addresses of nodes. In addition, [RFC7368] recommends that home networks be resilient to connectivity disruption from the ISP. This could be achieved by a dedicated device inside

the home network that builds the Homenet Zone, thus providing bindings between names and IP addresses. All this makes the CPE the natural candidate for populating the Homenet zone.

CPEs are usually low powered devices designed for the home network, but not for terminating heavy traffic. As a result, hosting an authoritative DNS service on the Internet may expose the home network to resource exhaustion and other attacks. This may isolate the home network from the Internet and also impact the services hosted by the CPEs, thus affecting overall home network communication.

In order to avoid resource exhaustion and other attacks, this document describes an architecture that outsources the authoritative naming service of the home network. More specifically, the Homenet Zone built by the CPE is outsourced to an Outsourcing Infrastructure. The Outsourcing Infrastructure publishes the corresponding Public Homenet Zone on the Internet. Section 4.1 describes the architecture. In order to keep the Public Homenet Zone up-to-date Section 5 describes how the Homenet Zone and the Public Homenet Zone can be synchronized. The proposed architecture aims at deploying DNSSEC, and the Public Homenet Zone is expected to be signed with a secure delegation. The zone signing and secure delegation may be performed either by the CPE or by the Outsourcing Infrastructure. Section 6 discusses these two alternatives. Section 7 discusses the consequences of publishing multiple representations of the same zone also commonly designated as views. This section provides guidance to limit the risks associated with multiple views. Section 8 discusses management of the reverse zone. Section 9 discusses how renumbering should be handled. Finally, Section 10 and Section 11 respectively discuss privacy and security considerations when outsourcing the Homenet Zone.

## 2.1. Scope of the Document

The scope of the document is to describe an architecture that outsources the authoritative naming service of the home network and more specifically the interactions between the home network and the Outsourcing Infrastructure. Considerations or descriptions on inner communications or organization of the home network are provided for completeness and for clarifying the interface between the home network and the Outsourcing Infrastructure

For sake of simplicity, this document designates by "CPE" that connects the home network with the Outsourcing Infrastructure - and so performs most of the operations for outsourcing the home network naming architecture. On the other hand, CPE are usually associated to home router. These two functions - e.g routing and naming - are not correlated and could be split in multiple devices. More

specifically, this document does not consider the home network has a single CPE nor a single ISP.

In fact this document considers the CPE as a set of functionalities that can be collocated on a single device or split between multiple devices. The split of these functions between different devices as well as their potential associated communications is considered as implementation dependent and out of scope of the architecture. The only limitation this architecture considers is that the Hidden Primary be located in a single place as long as the distributed Primary architecture has not been defined. As a result, if there are multiple candidates for hosting the Hidden Primary, the home network should select a single device.

### 3. Terminology

- Customer Premises Equipment: (CPE) is a router providing connectivity to the home network. It might be configured and managed by the end user. In this document, the CPE might also host services such as DHCPv6. This device might be provided by the ISP. A home network may have multiple CPE, and each of them may be connected to an Internet Service Provider. In this document, the CPE represents the set of functions involved in the naming architecture. These functions may be collocated on a single device, or distributed between multiple devices. How these functions communicate is out of the scope of this document and is left for implementation. See Section 2.1 for more details.
- Registered Homenet Domain: is the Domain Name associated to the home network.
- Homenet Zone: is the DNS zone associated with the home network. It is designated by its Registered Homenet Domain. This zone is built by the CPE and contains the bindings between names and IP addresses of the nodes in the home network. The CPE synchronizes the Homenet Zone with the Synchronization Server via a hidden primary / secondary architecture. The Outsourcing Infrastructure may process the Homenet Zone - for example providing DNSSEC signing - to generate the Public Homenet Zone. This Public Homenet Zone is then transmitted to the Public Authoritative Server(s) that publish it on the Internet.
- Public Homenet Zone: is the public version of the Homenet Zone. It is expected to be signed with DNSSEC. It is hosted by the Public Authoritative Server(s), which are authoritative for this zone. The Public Homenet Zone and the Homenet Zone might be different. For example some names might not become

reachable from the Internet, and thus not be hosted in the Public Homenet Zone. Another example of difference may also occur when the Public Homenet Zone is signed whereas the Homenet Zone is not signed.

- Outsourcing Infrastructure:    is the combination of the Synchronization Server and the Public Authoritative Server(s).
- Public Authoritative Servers:    are the authoritative name servers hosting the Public Homenet Zone. Name resolution requests for the Homenet Domain are sent to these servers. For resiliency the Public Homenet Zone SHOULD be hosted on multiple servers.
- Synchronization Server:    is the server with which the CPE synchronizes the Homenet Zone. The Synchronization Server is configured as a secondary and the CPE acts as primary. There MAY be multiple Synchronization Servers, but the text assumes a single server. In addition, the text assumes the Synchronization Server is a separate entity. This is not a requirement, and when the CPE signs the zone, the synchronization function might also be operated by the Public Authoritative Servers.
- Homenet Reverse Zone:    The reverse zone file associated with the Homenet Zone.
- Reverse Public Authoritative Servers:    are the authoritative name server(s) hosting the Public Homenet Reverse Zone. Queries for reverse resolution of the Homenet Domain are sent to this server. Similarly to Public Authoritative Servers, for resiliency, the Homenet Reverse Zone SHOULD be hosted on multiple servers.
- Reverse Synchronization Server:    is the server with which the CPE synchronizes the Homenet Reverse Zone. It is configured as a secondary and the CPE acts as primary. There MAY be multiple Reverse Synchronization Servers, but the text assumes a single server. In addition, the text assumes the Reverse Synchronization Server is a separate entity. This is not a requirement, and when the CPE signs the zone, the synchronization function might also be operated by the Reverse Public Authoritative Servers.
- Hidden Primary:    designates the primary server of the CPE, that synchronizes the Homenet Zone with the Synchronization Server. A primary / secondary architecture is used between the CPE and the Synchronization Server. The hidden primary is not expected to serve end user queries for the Homenet Zone as a regular

primary server would. The hidden primary is only known to its associated Synchronization Server.

#### 4. Architecture Description

This section describes the architecture for outsourcing the authoritative naming service from the CPE to the Outsourcing Infrastructure. Section 4.1 describes the architecture, Section 4.2 and Section 4.3 illustrates this architecture and shows how the Homenet Zone should be built by the CPE. It also lists the necessary parameters the CPE needs to be able to outsource the authoritative naming service. These two sections are informational and non-normative.

##### 4.1. Architecture Overview

Figure 1 provides an overview of the architecture.

The home network is designated by the Registered Homenet Domain Name -- example.com in Figure 1. The CPE builds the Homenet Zone associated with the home network. How the Homenet Zone is built is out of the scope of this document. The CPE may host or interact with multiple services to determine name-to-address mappings, such as a web GUI, DHCP [RFC6644] or mDNS [RFC6762]. These services may coexist and may be used to populate the Homenet Zone. This document assumes the Homenet Zone has been populated with domain names that are intended to be publicly published and that are publicly reachable. More specifically, names associated with services or devices that are not expected to be reachable from outside the home network or names bound to non-globally reachable IP addresses MUST NOT be part of the Homenet Zone.

Once the Homenet Zone has been built, the CPE does not host an authoritative naming service, but instead outsources it to the Outsourcing Infrastructure. The Outsourcing Infrastructure takes the Homenet Zone as an input and publishes the Public Homenet Zone. If the CPE does not sign the Homenet Zone, the Outsourcing Infrastructure may instead sign it on behalf of the CPE. Figure 1 provides a more detailed description of the Outsourcing Infrastructure, but overall, it is expected that the CPE provides the Homenet Zone. Then the Public Homenet Zone is derived from the Homenet Zone and published on the Internet.

As a result, DNS queries from the DNS resolvers on the Internet are answered by the Outsourcing Infrastructure and do not reach the CPE. Figure 1 illustrates the case of the resolution of node1.example.com.

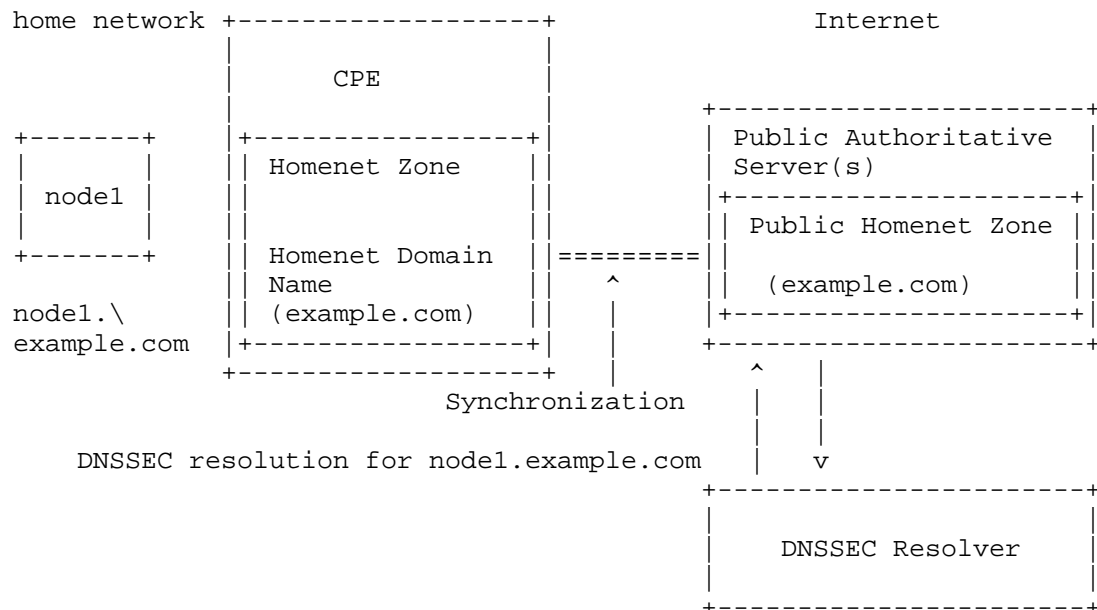


Figure 1: Homenet Naming Architecture Description

The Outsourcing Infrastructure is described in Figure 2. The Synchronization Server receives the Homenet Zone as an input. The received zone may be transformed to output the Public Homenet Zone. Various operations may be performed here, however this document only considers zone signing as a potential operation. This should occur only when the CPE outsources this operation to the Synchronization Server. On the other hand, if the CPE signs the Homenet Zone itself, the zone would be collected by the Synchronization Server and directly transferred to the Public Authoritative Server(s). These policies are discussed and detailed in Section 6 and Section 7.

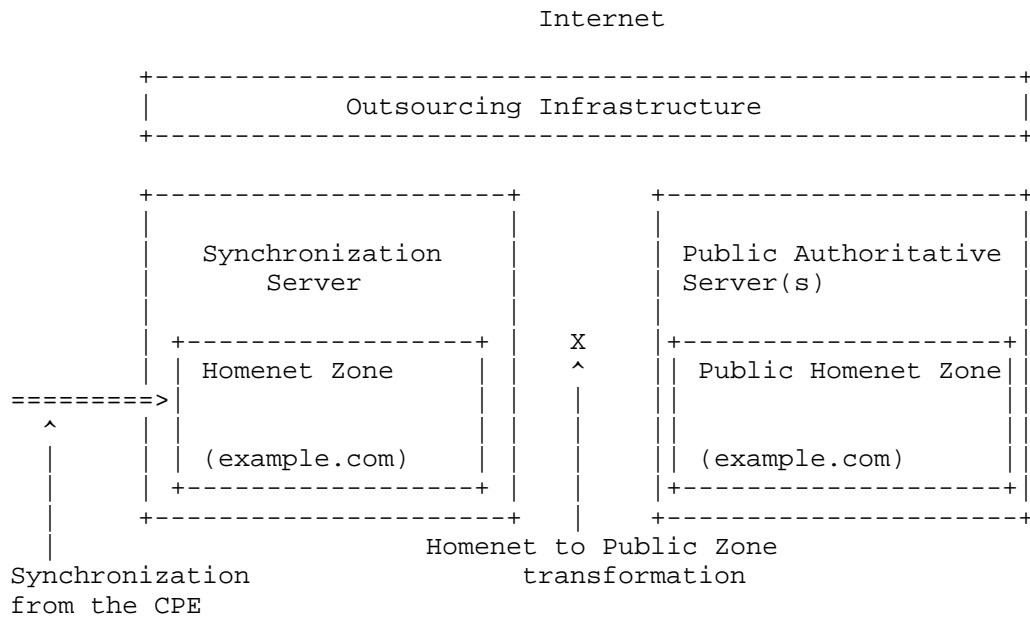


Figure 2: Outsourcing Infrastructure Description

#### 4.2. Example: Homenet Zone

This section is not normative and intends to illustrate how the CPE builds the Homenet Zone.

As depicted in Figure 1 and Figure 2, the Public Homenet Zone is hosted on the Public Authoritative Server(s), whereas the Homenet Zone is hosted on the CPE. Motivations for keeping these two zones identical are detailed in Section 7, and this section considers that the CPE builds the zone that will be effectively published on the Public Authoritative Server(s). In other words "Homenet to Public Zone transformation" is the identity also commonly designated as "no operation" (NOP).

In that case, the Homenet Zone should configure its Name Server RRset (NS) and Start of Authority (SOA) with the values associated with the Public Authoritative Server(s). This is illustrated in Figure 3. `public.primary.example.net` is the FQDN of the Public Authoritative Server(s), and IP1, IP2, IP3, IP4 are the associated IP addresses. Then the CPE should add the additional new nodes that enter the home network, remove those that should be removed, and sign the Homenet Zone.

```
$ORIGIN example.com
$TTL 1h

@ IN SOA public.primary.example.net
    hostmaster.example.com. (
        2013120710 ; serial number of this zone file
        1d         ; secondary refresh
        2h         ; secondary retry time in case of a problem
        4w         ; secondary expiration time
        1h         ; maximum caching time in case of failed
                   ; lookups
    )

@ NS public.authoritative.servers.example.net

public.primary.example.net A @IP1
public.primary.example.net A @IP2
public.primary.example.net AAAA @IP3
public.primary.example.net AAAA @IP4
```

Figure 3: Homenet Zone

The SOA RRset is defined in [RFC1033], [RFC1035] and [RFC2308]. This SOA is specific, as it is used for the synchronization between the Hidden Primary and the Synchronization Server and published on the DNS Public Authoritative Server(s)..

- MNAME: indicates the primary. In our case the zone is published on the Public Authoritative Server(s), and its name MUST be included. If multiple Public Authoritative Server(s) are involved, one of them MUST be chosen. More specifically, the CPE MUST NOT include the name of the Hidden Primary.
- RNAME: indicates the email address to reach the administrator. [RFC2142] recommends using hostmaster@domain and replacing the '@' sign by '.'.
- REFRESH and RETRY: indicate respectively in seconds how often secondaries need to check the primary, and the time between two refresh when a refresh has failed. Default values indicated by [RFC1033] are 3600 (1 hour) for refresh and 600 (10 minutes) for retry. This value might be too long for highly dynamic content. However, the Public Authoritative Server(s) and the CPE are expected to implement NOTIFY [RFC1996]. So whilst shorter refresh timers might increase the bandwidth usage for secondaries hosting large number of zones, it will have little practical impact on the elapsed time required to achieve



synchronization between the Outsourcing Infrastructure and the Hidden Master. As a result, the default values are acceptable.

EXPIRE:    is the upper limit data SHOULD be kept in absence of refresh. The default value indicated by [RFC1033] is 3600000 (approx. 42 days). In home network architectures, the CPE provides both the DNS synchronization and the access to the home network. This device may be plugged and unplugged by the end user without notification, thus we recommend a long expiry timer.

MINIMUM:   indicates the minimum TTL. The default value indicated by [RFC1033] is 86400 (1 day). For home network, this value MAY be reduced, and 3600 (1 hour) seems more appropriate.

#### 4.3. Example: CPE necessary parameters for outsourcing

This section specifies the various parameters required by the CPE to configure the naming architecture of this document. This section is informational, and is intended to clarify the information handled by the CPE and the various settings to be done.

Synchronization Server may be configured with the following parameters. These parameters are necessary to establish a secure channel between the CPE and the Synchronization Server as well as to specify the DNS zone that is in the scope of the communication:

- Synchronization Server:    The associated FQDNs or IP addresses of the Synchronization Server. IP addresses are optional and the FQDN is sufficient. To secure the binding name and IP addresses, a DNSSEC exchange is required. Otherwise, the IP addresses should be entered manually.
- Authentication Method:    How the CPE authenticates itself to the Synchronization Server. This MAY depend on the implementation but this should cover at least IPsec, DTLS and TSIG
- Authentication data:    Associated Data. PSK only requires a single argument. If other authentication mechanisms based on certificates are used, then CPE private keys, certificates and certification authority should be specified.
- Public Authoritative Server(s):    The FQDN or IP addresses of the Public Authoritative Server(s). It MAY correspond to the data that will be set in the NS RRsets and SOA of the Homenet Zone. IP addresses are optional and the FQDN is sufficient. To secure the binding between name and IP addresses, a DNSSEC

exchange is required. Otherwise, the IP addresses should be entered manually.

- Registered Homenet Domain: The domain name used to establish the secure channel. This name is used by the Synchronization Server and the CPE for the primary / secondary configuration as well as to index the NOTIFY queries of the CPE when the CPE has been renumbered.

Setting the Homenet Zone requires the following information.

- Registered Homenet Domain: The Domain Name of the zone. Multiple Registered Homenet Domains may be provided. This will generate the creation of multiple Public Homenet Zones.
- Public Authoritative Server(s): The Public Authoritative Server(s) associated with the Registered Homenet Domain. Multiple Public Authoritative Server(s) may be provided.

## 5. Synchronization between CPE and the Synchronization Server

The Homenet Reverse Zone and the Homenet Zone MAY be updated either with DNS UPDATE [RFC2136] or using a primary / secondary synchronization. The primary / secondary mechanism is preferred as it scales better and avoids DoS attacks: First the primary notifies the secondary that the zone must be updated and leaves the secondary to proceed with the update when possible. Then, a NOTIFY message is sent by the primary, which is a small packet that is less likely to load the secondary. Finally, the AXFR query performed by the secondary is a small packet sent over TCP (section 4.2 [RFC5936]), which mitigates reflection attacks using a forged NOTIFY. On the other hand, DNS UPDATE (which can be transported over UDP), requires more processing than a NOTIFY, and does not allow the server to perform asynchronous updates.

This document RECOMMENDS use of a primary / secondary mechanism instead of the use of DNS UPDATE. This section details the primary / secondary mechanism.

### 5.1. Synchronization with a Hidden Primary

Uploading and dynamically updating the zone file on the Synchronization Server can be seen as zone provisioning between the CPE (Hidden Primary) and the Synchronization Server (Secondary Server). This can be handled either in band or out of band.

Note that there is no standard way to distribute a DNS primary between multiple devices. As a result, if multiple CPEs are

candidate for hosting the Hidden Primary, some specific mechanisms should be designed so the home network only selects a single CPE for the Hidden Primary. Selection mechanisms based on HNCP are good candidates [XXX].

The Synchronization Server is configured as a secondary for the Homenet Domain Name. This secondary configuration has been previously agreed between the end user and the provider of the Synchronization Server. In order to set the primary / secondary architecture, the CPE acts as a Hidden Primary Server, which is a regular authoritative DNS Server listening on the WAN interface.

The Hidden Primary Server SHOULD accept SOA [RFC1033], AXFR [RFC1034], and IXFR [RFC1995] queries from its configured secondary DNS server(s). The Hidden Primary Server SHOULD send NOTIFY messages [RFC1996] in order to update Public DNS server zones as updates occur. Because, the Homenet Zones are likely to be small, the CPE MUST implement AXFR and SHOULD implement IXFR.

Hidden Primary Server differs from a regular authoritative server for the home network by:

- Interface Binding:    the Hidden Primary Server listens on the WAN Interface, whereas a regular authoritative server for the home network would listen on the home network interface.
- Limited exchanges:    the purpose of the Hidden Primary Server is to synchronize with the Synchronization Server, not to serve any zones to end users. As a result, exchanges are performed with specific nodes (the Synchronization Server). Further, exchange types are limited. The only legitimate exchanges are: NOTIFY initiated by the Hidden Primary and IXFR or AXFR exchanges initiated by the Synchronization Server. On the other hand, regular authoritative servers would respond to any hosts, and any DNS query would be processed. The CPE SHOULD filter IXFR/AXFR traffic and drop traffic not initiated by the Synchronization Server. The CPE MUST listen for DNS on TCP and UDP and MUST at least allow SOA lookups of the Homenet Zone.

## 5.2.    Securing Synchronization

Exchange between the Synchronization Server and the CPE MUST be secured, at least for integrity protection and for authentication.

TSIG [RFC2845] or SIG(0) [RFC2931] MAY be used to secure the DNS communications between the CPE and the Synchronization Server. TSIG uses a symmetric key which can be managed by TKEY [RFC2930]. Management of the key involved in SIG(0) is performed through zone

updates. How keys are rolled over with SIG(0) is out-of-scope of this document. The advantage of these mechanisms is that they are only associated with the DNS application. Not relying on shared libraries eases testing and integration. On the other hand, using TSIG, TKEY or SIG(0) requires these mechanisms to be implemented on the CPE, which adds code and complexity. Another disadvantage is that TKEY does not provide authentication mechanisms.

Protocols like TLS [RFC5246] / DTLS [RFC6347] MAY be used to secure the transactions between the Synchronization Server and the CPE. The advantage of TLS/DTLS is that this technology is widely deployed, and most of the devices already embed TLS/DTLS libraries, possibly also taking advantage of hardware acceleration. Further, TLS/DTLS provides authentication facilities and can use certificates to authenticate the Synchronization Server and the CPE. On the other hand, using TLS/DTLS requires implementing DNS exchanges over TLS/DTLS, as well as a new service port. This document therefore does NOT RECOMMEND this option.

IPsec [RFC4301] IKEv2 [RFC7296] MAY also be used to secure transactions between the CPE and the Synchronization Server. Similarly to TLS/DTLS, most CPEs already embed an IPsec stack, and IKEv2 supports multiple authentication mechanisms via the EAP framework. In addition, IPsec can be used to protect DNS exchanges between the CPE and the Synchronization Server without any modifications of the DNS server or client. DNS integration over IPsec only requires an additional security policy in the Security Policy Database (SPD). One disadvantage of IPsec is that NATs and firewall traversal may be problematic. However, in our case, the CPE is connected to the Internet, and IPsec communication between the CPE and the Synchronization Server should not be impacted by middle boxes.

How the PSK can be used by any of the TSIG, TLS/DTLS or IPsec protocols: Authentication based on certificates implies a mutual authentication and thus requires the CPE to manage a private key, a public key, or certificates, as well as Certificate Authorities. This adds complexity to the configuration especially on the CPE side. For this reason, we RECOMMEND that the CPE MAY use PSK or certificate base authentication, and that the Synchronization Server MUST support PSK and certificate based authentication.

Note also that authentication of message exchanges between the CPE and the Synchronization Server SHOULD NOT use the external IP address of the CPE to index the appropriate keys. As detailed in Section 9, the IP addresses of the Synchronization Server and the Hidden Primary are subject to change, for example while the network is being renumbered. This means that the necessary keys to authenticate

transaction SHOULD NOT be indexed using the IP address, and SHOULD be resilient to IP address changes.

### 5.3. CPE Security Policies

This section details security policies related to the Hidden Primary / Secondary synchronization.

The Hidden Primary, as described in this document SHOULD drop any queries from the home network. This could be implemented via port binding and/or firewall rules. The precise mechanism deployed is out of scope of this document.

The Hidden Primary SHOULD drop any DNS queries arriving on the WAN interface that are not issued from the Synchronization Server.

The Hidden Primary SHOULD drop any outgoing packets other than DNS NOTIFY query, SOA response, IXFR response or AXFR responses.

The Hidden Primary SHOULD drop any incoming packets other than DNS NOTIFY response, SOA query, IXFR query or AXFR query.

The Hidden Primary SHOULD drop any non protected IXFR or AXFR exchange, depending on how the synchronization is secured.

## 6. DNSSEC compliant Homenet Architecture

[RFC7368] in Section 3.7.3 recommends DNSSEC to be deployed on both the authoritative server and the resolver. The resolver side is out of scope of this document, and only the authoritative part of the server is considered.

Deploying DNSSEC requires signing the zone and configuring a secure delegation. As described in Section 4.1, signing can be performed either by the CPE or by the Outsourcing Infrastructure. Section 6.1 details the implications of these two alternatives. Similarly, the secure delegation can be performed by the CPE or by the Outsourcing Infrastructure. Section 6.2 discusses these two alternatives.

### 6.1. Zone Signing

This section discusses the pros and cons when zone signing is performed by the CPE or by the Outsourcing Infrastructure. It is RECOMMENDED that the CPE signs the zone unless there is a strong argument against this, such as a CPE that is not capable of signing the zone. In that case zone signing MAY be performed by the Outsourcing Infrastructure on behalf of the CPE.

Reasons for signing the zone by the CPE are:

- 1: Keeping the Homenet Zone and the Public Homenet Zone equal to securely optimize DNS resolution. As the Public Zone is signed with DNSSEC, RRsets are authenticated, and thus DNS responses can be validated even though they are not provided by the authoritative server. This provides the CPE the ability to respond on behalf of the Public Authoritative Server(s). This could be useful for example if, in the future, the CPE announces to the home network that the CPE can act as a local authoritative primary or equivalent for the Homenet Zone. Currently the CPE is not expected to receive authoritative DNS queries, as its IP address is not mentioned in the Public Homenet Zone. On the other hand most CPEs host a resolving function, and could be configured to perform a local lookup to the Homenet Zone instead of initiating a DNS exchange with the Public Authoritative Server(s). Note that outsourcing the zone signing operation means that all DNSSEC queries SHOULD be cached to perform a local lookup, otherwise a resolution with the Public Authoritative Server(s) would be performed.
- 2: Keeping the Homenet Zone and the Public Homenet Zone equal to securely address the connectivity disruption independence detailed in [RFC7368] section 4.4.1 and 3.7.5. As local lookups are possible in case of network disruption, communications within the home network can still rely on the DNSSEC service. Note that outsourcing the zone signing operation does not address connectivity disruption independence with DNSSEC. Instead local lookup would provide DNS as opposed to DNSSEC responses provided by the Public Authoritative Server(s).
- 3: Keeping the Homenet Zone and the Public Homenet Zone equal to guarantee coherence between DNS responses. Using a unique zone is one way to guarantee uniqueness of the responses among servers and places. Issues generated by different views are discussed in more details in Section 7.
- 2: Privacy and Integrity of the DNSSEC Homenet Zone are better guaranteed. When the Zone is signed by the CPE, it makes modification of the DNS data -- for example for flow redirection -- impossible. As a result, signing the Homenet Zone by the CPE provides better protection for end user privacy.

Reasons for signing the zone by the Outsourcing Infrastructure are:

- 1: The CPE may not be capable of signing the zone, most likely because its firmware does not support this function. However this reason is expected to become less and less valid over time.
- 2: Outsourcing DNSSEC management operations. Management operations involve key roll-over, which can be performed automatically by the CPE and transparently for the end user. Avoiding DNSSEC management is mostly motivated by bad software implementations.
- 3: Reducing the impact of CPE replacement on the Public Homenet Zone. Unless the CPE private keys can be extracted and stored off-device, CPE hardware replacement will result in an emergency key roll-over. This can be mitigated by using relatively small TTLs.
- 4: Reducing configuration impact on the end user. Unless there are zero configuration mechanisms in place to provide credentials between the new CPE and the Synchronization Server, authentication associations between the CPE and the Synchronization Server would need to be re-configured. As CPE replacement is not expected to happen regularly, end users may not be at ease with such configuration settings. However, mechanisms as described in [I-D.ietf-homenet-naming-architecture-dhc-options] use DHCP Options to outsource the configuration and avoid this issue.
- 5: The Outsourcing Infrastructure is more likely to handle private keys more securely than the CPE. However, having all private keys in one place may also nullify that benefit.

## 6.2. Secure Delegation

Secure delegation is achieved only if the DS RRset is properly set in the parent zone. Secure delegation can be performed by the CPE or the Outsourcing Infrastructures (that is the Synchronization Server or the Public Authoritative Server(s)).

The DS RRset can be updated manually with nsupdate for example. This requires the CPE or the Outsourcing Infrastructure to be authenticated by the DNS server hosting the parent of the Public Homenet Zone. Such a trust channel between the CPE and the parent DNS server may be hard to maintain with CPEs, and thus may be easier to establish with the Outsourcing Infrastructure. In fact, the Public Authoritative Server(s) may use Automating DNSSEC Delegation Trust Maintenance [RFC7344].

## 7. Handling Different Views

The Homenet Zone provides information about the home network. Some users may be tempted to have provide responses dependent on the origin of the DNS query. More specifically, some users may be tempted to provide a different view for DNS queries originating from the home network and for DNS queries coming from the Internet. Each view could then be associated with a dedicated Homenet Zone. Note that this document does not specify how DNS queries originating from the home network are addressed to the Homenet Zone. This could be done via hosting the DNS resolver on the CPE for example.

This section is not normative. Section 7.1 details why some nodes may only be reachable from the home network and not from the global Internet. Section 7.2 briefly describes the consequences of having distinct views such as a "home network view" and an "Internet view". Finally, Section 7.3 provides guidance on how to resolve names that are only significant in the home network, without creating different views.

### 7.1. Misleading Reasons for Local Scope DNS Zone

The motivation for supporting different views is to provide different answers dependent on the origin of the DNS query, for reasons such as:

- 1: An end user may want to have services not published on the Internet. Services like the CPE administration interface that provides the GUI to administer your CPE might not seem advisable to publish on the Internet. Similarly, services like the mapper that registers the devices of your home network may also not be desirable to be published on the Internet. In both cases, these services should only be known or used by the network administrator. To restrict the access of such services, the home network administrator may choose to publish these pieces of information only within the home network, where it might be assumed that the users are more trusted than on the Internet. Even though this assumption may not be valid, at least this may reduce the surface of any attack.
- 2: Services within the home network may be reachable using non global IP addresses. IPv4 and NAT may be one reason. On the other hand IPv6 may favor link-local or site-local IP addresses. These IP addresses are not significant outside the boundaries of the home network. As a result, they MAY be published in the home network view, and SHOULD NOT be published in the Public Homenet Zone.



## 7.2. Consequences

Enabling different views leads to a non-coherent naming system. Depending on where resolution is performed, some services will not be available. This may be especially inconvenient with devices with multiple interfaces that are attached both to the Internet via a 3G/4G interface and to the home network via a WLAN interface. Devices may also cache the results of name resolution, and these cached entries may no longer be valid if a mobile device moves between a homenet connection and an internet connection e.g. a device temporarily loses wifi signal and switches to 3G.

Regarding local-scope IP addresses, such devices may end up with poor connectivity. Suppose, for example, that DNS resolution is performed via the WLAN interface attached to the CPE, and the response provides local-scope IP addresses, but the communication is initiated on the 3G/4G interface. Communications with local-scope addresses will be unreachable on the Internet, thus aborting the communication. The same situation occurs if a device is flip / flopping between various WLAN networks.

Regarding DNSSEC, if the CPE does not sign the Homenet Zone and outsources the signing process, the two views are different, because one is protected with DNSSEC whereas the other is not. Devices with multiple interfaces will have difficulty securing the naming resolution, as responses originating from the home network may not be signed.

For devices with all its interfaces attached to a single administrative domain, that is to say the home network, or the Internet. Incoherence between DNS responses may still also occur if the device is able to perform DNS resolutions both using the DNS resolving server of the home network, or one of the ISP. DNS resolution performed via the CPE or the ISP resolver may be different than those performed over the Internet.

## 7.3. Guidance and Recommendations

As documented in Section 7.2, it is RECOMMENDED to avoid different views. If network administrators choose to implement multiple views, impacts on devices' resolution SHOULD be evaluated.

As a consequence, the Homenet Zone is expected to be an exact copy of the Public Homenet Zone. As a result, services that are not expected to be published on the Internet SHOULD NOT be part of the Homenet Zone, local-scope addresses SHOULD NOT be part of the Homenet Zone, and when possible, the CPE SHOULD sign the Homenet Zone.

The Homenet Zone is expected to host public information only. It is not the scope of the DNS service to define local home network boundaries. Instead, local scope information is expected to be provided to the home network using local scope naming services. mDNS [RFC6762] DNS-SD [RFC6763] are two examples of these services. Currently mDNS is limited to a single link network. However, future protocols are expected to leverage this constraint as pointed out in [RFC7558].

## 8. Homenet Reverse Zone

This section is focused on the Homenet Reverse Zone.

Firstly, all considerations for the Homenet Zone apply to the Homenet Reverse Zone. The main difference between the Homenet Reverse Zone and the Homenet Zone is that the parent zone of the Homenet Reverse Zone is most likely managed by the ISP. As the ISP also provides the IP prefix to the CPE, it may be able to authenticate the CPE using mechanisms outside the scope of this document e.g. the physical attachment point to the ISP network. If the Reverse Synchronization Server is managed by the ISP, credentials to authenticate the CPE for the zone synchronization may be set automatically and transparently to the end user. [I-D.ietf-homenet-naming-architecture-dhc-options] describes how automatic configuration may be performed.

With IPv6, the domain space for IP addresses is so large that reverse zone may be confronted with scalability issues. How the reverse zone is generated is out of scope of this document. [I-D.howard-dnsop-ip6rdns] provides guidance on how to address scalability issues.

## 9. Renumbering

This section details how renumbering is handled by the Hidden Primary server or the Synchronization Server. Both types of renumbering are discussed i.e. "make-before-break" and "break-before-make".

In the make-before-break renumbering scenario, the new prefix is advertised, the network is configured to prepare the transition to the new prefix. During a period of time, the two prefixes old and new coexist, before the old prefix is completely removed. In the break-before-make renumbering scenario, the new prefix is advertised making the old prefix obsolete.

Renumbering has been extensively described in [RFC4192] and analyzed in [RFC7010] and the reader is expected to be familiar with them before reading this section.

### 9.1. Hidden Primary

In a renumbering scenario, the Hidden Primary is informed it is being renumbered. In most cases, this occurs because the whole home network is being renumbered. As a result, the Homenet Zone will also be updated. Although the new and old IP addresses may be stored in the Homenet Zone, we recommend that only the newly reachable IP addresses be published.

To avoid reachability disruption, IP connectivity information provided by the DNS SHOULD be coherent with the IP plane. In our case, this means the old IP address SHOULD NOT be provided via the DNS when it is not reachable anymore. Let for example TTL be the TTL associated with a RRset of the Homenet Zone, it may be cached for TTL seconds. Let  $T\_NEW$  be the time the new IP address replaces the old IP address in the Homenet Zone, and  $T\_OLD\_UNREACHABLE$  the time the old IP is not reachable anymore. In the case of the make-before-break, seamless reachability is provided as long as  $T\_OLD\_UNREACHABLE - T\_NEW > 2 * TTL$ . If this is not satisfied, then devices associated with the old IP address in the home network may become unreachable for  $2 * TTL - (T\_OLD\_UNREACHABLE - T\_NEW)$ . In the case of a break-before-make,  $T\_OLD\_UNREACHABLE = T\_NEW$ , and the device may become unreachable up to  $2 * TTL$ .

Once the Homenet Zone file has been updated on the Hidden Primary, the Hidden Primary needs to inform the Outsourcing Infrastructure that the Homenet Zone has been updated and that the IP address to use to retrieve the updated zone has also been updated. Both notifications are performed using regular DNS exchanges. Mechanisms to update an IP address provided by lower layers with protocols like SCTP [RFC4960], MOBIKE [RFC4555] are not considered in this document.

The Hidden Primary SHOULD inform the Synchronization Server that the Homenet Zone has been updated by sending a NOTIFY payload with the new IP address. In addition, this NOTIFY payload SHOULD be authenticated using SIG(0) or TSIG. When the Synchronization Server receives the NOTIFY payload, it MUST authenticate it. Note that the cryptographic key used for the authentication SHOULD be indexed by the Registered Homenet Domain contained in the NOTIFY payload as well as the RRSIG. In other words, the IP address SHOULD NOT be used as an index. If authentication succeeds, the Synchronization Server MUST also notice the IP address has been modified and perform a reachability check before updating its primary configuration. The routability check MAY be performed by sending a SOA request to the Hidden Primary using the source IP address of the NOTIFY. This exchange is also secured, and if an authenticated response is received from the Hidden Primary with the new IP address, the

Synchronization Server SHOULD update its configuration file and retrieve the Homenet Zone using an AXFR or a IXFR exchange.

Note that the primary reason for providing the IP address is that the Hidden Primary is not publicly announced in the DNS. If the Hidden Primary were publicly announced in the DNS, then the IP address update could have been performed using the DNS as described in Section 9.2.

## 9.2. Synchronization Server

Renumbering of the Synchronization Server results in the Synchronization Server changing its IP address. The Synchronization Server is a secondary, so its renumbering does not impact the Homenet Zone. In fact, exchanges to the Synchronization Server are restricted to the Homenet Zone synchronization. In our case, the Hidden Primary MUST be able to send NOTIFY payloads to the Synchronization Server.

If the Synchronization Server is configured in the Hidden Primary configuration file using a FQDN, then the update of the IP address is performed by DNS. More specifically, before sending the NOTIFY, the Hidden Primary performs a DNS resolution to retrieve the IP address of the secondary.

As described in Section 9.1, the Synchronization Server DNS information SHOULD be coherent with the IP plane. Let TTL be the TTL associated with the Synchronization Server FQDN, T\_NEW the time the new IP address replaces the old one and T\_OLD\_UNREACHABLE the time the Synchronization Server is not reachable anymore with its old IP address. Seamless reachability is provided as long as  $T\_OLD\_UNREACHABLE - T\_NEW > 2 * TTL$ . If this condition is not met, the Synchronization Server may be unreachable during  $2 * TTL - (T\_OLD\_UNREACHABLE - T\_NEW)$ . In the case of a break-before-make,  $T\_OLD\_UNREACHABLE = T\_NEW$ , and it may become unreachable up to  $2 * TTL$ .

Some DNS infrastructure uses the IP address to designate the secondary, in which case, other mechanisms must be found. The reason for using IP addresses instead of names is generally to reach an internal interface that is not designated by a FQDN, and to avoid potential bootstrap problems. Such scenarios are considered as out of scope in the case of home networks.

## 10. Privacy Considerations

Outsourcing the DNS Authoritative service from the CPE to a third party raises a few privacy related concerns.

The Homenet Zone contains a full description of the services hosted in the network. These services may not be expected to be publicly shared although their names remain accessible through the Internet. Even though DNS makes information public, the DNS does not expect to make the complete list of services public. In fact, making information public still requires the key (or FQDN) of each service to be known by the resolver in order to retrieve information about the services. More specifically, making mywebsite.example.com public in the DNS, is not sufficient to make resolvers aware of the existence web site. However, an attacker may walk the reverse DNS zone, or use other reconnaissance techniques to learn this information as described in [I-D.ietf-opsec-ipv6-host-scanning].

In order to prevent the complete Homenet Zone being published on the Internet, AXFR queries SHOULD be blocked on the Public Authoritative Server(s). Similarly, to avoid zone-walking NSEC3 [RFC5155] SHOULD be preferred over NSEC [RFC4034].

When the Homenet Zone is outsourced, the end user should be aware that it provides a complete description of the services available on the home network. More specifically, names usually provides a clear indication of the service and possibly even the device type, and as the Homenet Zone contains the IP addresses associated with the service, they also limit the scope of the scan space.

In addition to the Homenet Zone, the third party can also monitor the traffic associated with the Homenet Zone. This traffic may provide an indication of the services an end user accesses, plus how and when they use these services. Although, caching may obfuscate this information inside the home network, it is likely that outside your home network this information will not be cached.

## 11. Security Considerations

The Homenet Naming Architecture described in this document solves exposing the CPE's DNS service as a DoS attack vector.

### 11.1. Names are less secure than IP addresses

This document describes how an end user can make their services and devices from his home network reachable on the Internet by using names rather than IP addresses. This exposes the home network to attackers, since names are expected to include less entropy than IP

addresses. In fact, with IP addresses, the Interface Identifier is 64 bits long leading to up to  $2^{64}$  possibilities for a given subnetwork. This is not to mention that the subnet prefix is also of 64 bits long, thus providing up to  $2^{64}$  possibilities. On the other hand, names used either for the home network domain or for the devices present less entropy (livebox, router, printer, nicolas, jennifer, ...) and thus potentially exposes the devices to dictionary attacks.

#### 11.2. Names are less volatile than IP addresses

IP addresses may be used to locate a device, a host or a service. However, home networks are not expected to be assigned a time invariant prefix by ISPs. As a result, observing IP addresses only provides some ephemeral information about who is accessing the service. On the other hand, names are not expected to be as volatile as IP addresses. As a result, logging names over time may be more valuable than logging IP addresses, especially to profile an end user's characteristics.

PTR provides a way to bind an IP address to a name. In that sense, responding to PTR DNS queries may affect the end user's privacy. For that reason end users may choose not to respond to PTR DNS queries and MAY instead return a NXDOMAIN response.

#### 11.3. DNS Reflection Attacks

An attacker performs a reflection attack when it sends traffic to one or more intermediary nodes (reflectors), that in turn send back response traffic to the victim. Motivations for using an intermediary node might be anonymity of the attacker, as well as amplification of the traffic. Typically, when the intermediary node is a DNSSEC server, the attacker sends a DNSSEC query and the victim is likely to receive a DNSSEC response. This section analyzes how the different components may be involved as a reflector in a reflection attack. Section 11.3.1 considers the Hidden Primary, Section 11.3.2 the Synchronization Server, and Section 11.3.3 the Public Authoritative Server(s).

##### 11.3.1. Reflection Attack involving the Hidden Primary

With the specified architecture, the Hidden Primary is only expected to receive DNS queries of type SOA, AXFR or IXFR. This section analyzes how these DNS queries may be used by an attacker to perform a reflection attack.

DNS queries of type AXFR and IXFR use TCP and as such are less subject to reflection attacks. This makes SOA queries the only

remaining practical vector of attacks for reflection attacks, based on UDP.

SOA queries are not associated with a large amplification factor compared to queries of type "ANY" or to query of non existing FQDNs. This reduces the probability a DNS query of type SOA will be involved in a DDoS attack.

SOA queries are expected to follow a very specific pattern, which makes rate limiting techniques an efficient way to limit such attacks, and associated impact on the naming service of the home network.

Motivations for such a flood might be a reflection attack, but could also be a resource exhaustion attack performed against the Hidden Primary. The Hidden Primary only expects to exchange traffic with the Synchronization Server, that is its associated secondary. Even though secondary servers may be renumbered as mentioned in Section 9, the Hidden Primary is likely to perform a DNSSEC resolution and find out the associated secondary's IP addresses in use. As a result, the Hidden Primary is likely to limit the origin of its incoming traffic based on the origin IP address.

With filtering rules based on IP address, SOA flooding attacks are limited to forged packets with the IP address of the secondary server. In other words, the only victims are the Hidden Primary itself or the secondary. There is a need for the Hidden Primary to limit that flood to limit the impact of the reflection attack on the secondary, and to limit the resource needed to carry on the traffic by the CPE hosting the Hidden Primary. On the other hand, mitigation should be performed appropriately, so as to limit the impact on the legitimate SOA sent by the secondary.

The main reason for the Synchronization Server sending a SOA query is to update the SOA RRset after the TTL expires, to check the serial number upon the receipt of a NOTIFY query from the Hidden Primary, or to re-send the SOA request when the response has not been received. When a flood of SOA queries is received by the Hidden Primary, the Hidden Primary may assume it is involved in an attack.

There are few legitimate time slots when the secondary is expected to send a SOA query. Suppose  $T\_NOTIFY$  is the time a NOTIFY is sent by the Hidden Primary,  $T\_SOA$  the last time the SOA has been queried,  $TTL$  the TTL associated to the SOA, and  $T\_REFRESH$  the refresh time defined in the SOA RRset. The specific time SOA queries are expected can be for example  $T\_NOTIFY$ ,  $T\_SOA + 2/3\ TTL$ ,  $T\_SOA + TTL$ ,  $T\_SOA + T\_REFRESH$ , and. Outside a few minutes following these specific time slots, the probability that the CPE discards a legitimate SOA query

is very low. Within these time slots, the probability the secondary may have its legitimate query rejected is higher. If a legitimate SOA is discarded, the secondary will re-send SOA query every "retry time" second until "expire time" seconds occurs, where "retry time" and "expire time" have been defined in the SOA.

As a result, it is RECOMMENDED to set rate limiting policies to protect CPE resources. If a flood lasts more than the expired time defined by the SOA, it is RECOMMENDED to re-initiate a synchronization between the Hidden Primary and the secondaries.

#### 11.3.2. Reflection Attacks involving the Synchronization Server

The Synchronization Server acts as a secondary coupled with the Hidden Primary. The secondary expects to receive NOTIFY query, SOA responses, AXFR and IXFR responses from the Hidden Primary.

Sending a NOTIFY query to the secondary generates a NOTIFY response as well as initiating an SOA query exchange from the secondary to the Hidden Primary. As mentioned in [RFC1996], this is a known "benign denial of service attack". As a result, the Synchronization Server SHOULD enforce rate limiting on sending SOA queries and NOTIFY responses to the Hidden Primary. Most likely, when the secondary is flooded with valid and signed NOTIFY queries, it is under a replay attack which is discussed in Section 11.5. The key thing here is that the secondary is likely to be designed to be able to process much more traffic than the Hidden Primary hosted on a CPE.

This paragraph details how the secondary may limit the NOTIFY queries. Because the Hidden Primary may be renumbered, the secondary SHOULD NOT perform permanent IP filtering based on IP addresses. In addition, a given secondary may be shared among multiple Hidden Primaries which make filtering rules based on IP harder to set. The time at which a NOTIFY is sent by the Hidden Primary is not predictable. However, a flood of NOTIFY messages may be easily detected, as a NOTIFY originated from a given Homenet Zone is expected to have a very limited number of unique source IP addresses, even when renumbering is occurring. As a result, the secondary, MAY rate limit incoming NOTIFY queries.

On the Hidden Primary side, it is recommended that the Hidden Primary sends a NOTIFY as long as the zone has not been updated by the secondary. Multiple SOA queries may indicate the secondary is under attack.



#### 11.3.3. Reflection Attacks involving the Public Authoritative Servers

Reflection attacks involving the Public Authoritative Server(s) are similar to attacks on any Outsourcing Infrastructure. This is not specific to the architecture described in this document, and thus are considered as out of scope.

In fact, one motivation of the architecture described in this document is to expose the Public Authoritative Server(s) to attacks instead of the CPE, as it is believed that the Public Authoritative Server(s) will be better able to defend itself.

#### 11.4. Flooding Attack

The purpose of flooding attacks is mostly resource exhaustion, where the resource can be bandwidth, memory, or CPU for example.

One goal of the architecture described in this document is to limit the surface of attack on the CPE. This is done by outsourcing the DNS service to the Public Authoritative Server(s). By doing so, the CPE limits its DNS interactions between the Hidden Primary and the Synchronization Server. This limits the number of entities the CPE interacts with as well as the scope of DNS exchanges - NOTIFY, SOA, AXFR, IXFR.

The use of an authenticated channel with SIG(0) or TSIG between the CPE and the Synchronization Server, enables detection of illegitimate DNS queries, so appropriate action may be taken - like dropping the queries. If signatures are validated, then most likely, the CPE is under a replay attack, as detailed in Section 11.5

In order to limit the resource required for authentication, it is recommended to use TSIG that uses symmetric cryptography over SIG(0) that uses asymmetric cryptography.

#### 11.5. Replay Attack

Replay attacks consist of an attacker either resending or delaying a legitimate message that has been sent by an authorized user or process. As the Hidden Primary and the Synchronization Server use an authenticated channel, replay attacks are mostly expected to use forged DNS queries in order to provide valid traffic.

From the perspective of an attacker, using a correctly authenticated DNS query may not be detected as an attack and thus may generate a response. Generating and sending a response consumes more resources than either dropping the query by the defender, or generating the query by the attacker, and thus could be used for resource exhaustion

attacks. In addition, as the authentication is performed at the DNS layer, the source IP address could be impersonated in order to perform a reflection attack.

Section 11.3 details how to mitigate reflection attacks and Section 11.4 details how to mitigate resource exhaustion. Both sections assume a context of DoS with a flood of DNS queries. This section suggests a way to limit the attack surface of replay attacks.

As SIG(0) and TSIG use inception and expiration time, the time frame for replay attack is limited. SIG(0) and TSIG recommends a fudge value of 5 minutes. This value has been set as a compromise between possibly loose time synchronization between devices and the valid lifetime of the message. As a result, better time synchronization policies could reduce the time window of the attack.

## 12. IANA Considerations

This document has no actions for IANA.

## 13. Acknowledgment

The authors wish to thank Philippe Lemordant for its contributions on the early versions of the draft; Ole Troan for pointing out issues with the IPv6 routed home concept and placing the scope of this document in a wider picture; Mark Townsley for encouragement and injecting a healthy debate on the merits of the idea; Ulrik de Bie for providing alternative solutions; Paul Mockapetris, Christian Jacquenet, Francis Dupont and Ludovic Eschard for their remarks on CPE and low power devices; Olafur Gudmundsson for clarifying DNSSEC capabilities of small devices; Simon Kelley for its feedback as dnsmasq implementer; Andrew Sullivan, Mark Andrew, Ted Lemon, Mikael Abrahamson, Michael Richardson and Ray Bellis for their feedback on handling different views as well as clarifying the impact of outsourcing the zone signing operation outside the CPE; Mark Andrew and Peter Koch for clarifying the renumbering.

## 14. References

### 14.1. 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>>.

- [RFC1995]   Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <<http://www.rfc-editor.org/info/rfc1995>>.
- [RFC1996]   Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <<http://www.rfc-editor.org/info/rfc1996>>.
- [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>>.
- [RFC2136]   Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <<http://www.rfc-editor.org/info/rfc2136>>.
- [RFC2142]   Crocker, D., "Mailbox Names for Common Services, Roles and Functions", RFC 2142, DOI 10.17487/RFC2142, May 1997, <<http://www.rfc-editor.org/info/rfc2142>>.
- [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>>.
- [RFC2845]   Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <<http://www.rfc-editor.org/info/rfc2845>>.
- [RFC2930]   Eastlake 3rd, D., "Secret Key Establishment for DNS (TKEY RR)", RFC 2930, DOI 10.17487/RFC2930, September 2000, <<http://www.rfc-editor.org/info/rfc2930>>.
- [RFC2931]   Eastlake 3rd, D., "DNS Request and Transaction Signatures ( SIG(0)s )", RFC 2931, DOI 10.17487/RFC2931, September 2000, <<http://www.rfc-editor.org/info/rfc2931>>.
- [RFC4034]   Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <<http://www.rfc-editor.org/info/rfc4034>>.
- [RFC4301]   Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<http://www.rfc-editor.org/info/rfc4301>>.

- [RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, DOI 10.17487/RFC4555, June 2006, <<http://www.rfc-editor.org/info/rfc4555>>.
- [RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <<http://www.rfc-editor.org/info/rfc4960>>.
- [RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, DOI 10.17487/RFC5155, March 2008, <<http://www.rfc-editor.org/info/rfc5155>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<http://www.rfc-editor.org/info/rfc5246>>.
- [RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <<http://www.rfc-editor.org/info/rfc5936>>.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<http://www.rfc-editor.org/info/rfc6347>>.
- [RFC6644] Evans, D., Droms, R., and S. Jiang, "Rebind Capability in DHCPv6 Reconfigure Messages", RFC 6644, DOI 10.17487/RFC6644, July 2012, <<http://www.rfc-editor.org/info/rfc6644>>.
- [RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <<http://www.rfc-editor.org/info/rfc6762>>.
- [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>>.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<http://www.rfc-editor.org/info/rfc7296>>.

## 14.2. Informational References

- [I-D.howard-dnsop-ip6rdns]  
Howard, L., "Reverse DNS in IPv6 for Internet Service Providers", draft-howard-dnsop-ip6rdns-00 (work in progress), June 2014.
- [I-D.ietf-homenet-naming-architecture-dhc-options]  
Migault, D., Cloetens, W., Griffiths, C., and R. Weber, "DHCP Options for Homenet Naming Architecture", draft-ietf-homenet-naming-architecture-dhc-options-02 (work in progress), May 2015.
- [I-D.ietf-opsec-ipv6-host-scanning]  
Gont, F. and T. Chown, "Network Reconnaissance in IPv6 Networks", draft-ietf-opsec-ipv6-host-scanning-08 (work in progress), August 2015.
- [RFC1033]    Lottor, M., "Domain Administrators Operations Guide", RFC 1033, DOI 10.17487/RFC1033, November 1987,  
<<http://www.rfc-editor.org/info/rfc1033>>.
- [RFC4192]    Baker, F., Lear, E., and R. Droms, "Procedures for Renumbering an IPv6 Network without a Flag Day", RFC 4192, DOI 10.17487/RFC4192, September 2005,  
<<http://www.rfc-editor.org/info/rfc4192>>.
- [RFC7010]    Liu, B., Jiang, S., Carpenter, B., Venaas, S., and W. George, "IPv6 Site Renumbering Gap Analysis", RFC 7010, DOI 10.17487/RFC7010, September 2013,  
<<http://www.rfc-editor.org/info/rfc7010>>.
- [RFC7344]    Kumari, W., Gudmundsson, O., and G. Barwood, "Automating DNSSEC Delegation Trust Maintenance", RFC 7344, DOI 10.17487/RFC7344, September 2014,  
<<http://www.rfc-editor.org/info/rfc7344>>.
- [RFC7368]    Chown, T., Ed., Arkko, J., Brandt, A., Troan, O., and J. Weil, "IPv6 Home Networking Architecture Principles", RFC 7368, DOI 10.17487/RFC7368, October 2014,  
<<http://www.rfc-editor.org/info/rfc7368>>.
- [RFC7558]    Lynn, K., Cheshire, S., Blanchet, M., and D. Migault, "Requirements for Scalable DNS-Based Service Discovery (DNS-SD) / Multicast DNS (mDNS) Extensions", RFC 7558, DOI 10.17487/RFC7558, July 2015,  
<<http://www.rfc-editor.org/info/rfc7558>>.

Appendix A.   Document Change Log

[RFC Editor: This section is to be removed before publication]

-08

- 1: Clarification of the meaning of CPE. The architecture does not consider a single CPE. The CPE represents multiple functions.

-07:

- 1: Ray Hunter is added as a co-author.

-06:

- 2: Ray Hunter is added in acknowledgment.
- 3: Adding Renumbering section with comments from Dallas meeting
- 4: Replacing Master / Primary - Slave / Secondary

Security Consideration has been updated with Reflection attacks, flooding attacks, and replay attacks.

-05:

\*Clarifying on handling different views:

- 1: How the CPE may be involved in the resolution and responds without necessarily requesting the Public Authoritative Server(s) (and eventually the Hidden Primary)
- 2: How to handle local scope resolution that is link-local, site-local and NAT IP addresses as well as Private domain names that the administrator does not want to publish outside the home network.

Adding a Privacy Considerations Section

Clarification on pro/cons outsourcing zone-signing

Documenting how to handle reverse zones

Adding reference to RFC 2308

-04:

\*Clarifications on zone signing

- \*Rewording

- \*Adding section on different views

- \*architecture clarifications

-03:

- \*Simon's comments taken into consideration

- \*Adding SOA, PTR considerations

- \*Removing DNSSEC performance paragraphs on low power devices

- \*Adding SIG(0) as a mechanism for authenticating the servers

- \*Goals clarification: the architecture described in the document 1) does not describe new protocols, and 2) can be adapted to specific cases for advance users.

-02:

- \*remove interfaces: "Public Authoritative Server Naming Interface" is replaced by "Public Authoritative Server(s)y(ies)". "Public Authoritative Server Management Interface" is replaced by "Synchronization Server".

-01.3:

- \*remove the authoritative / resolver services of the CPE.  
Implementation dependent

- \*remove interactions with mdns and dhcp.   Implementation dependent.

- \*remove considerations on low powered devices

- \*remove position toward homenet arch

- \*remove problem statement section

-01.2:

- \* add a CPE description to show that the architecture can fit CPEs

- \* specification of the architecture for very low powered devices.

- \* integrate mDNS and DHCP interactions with the Homenet Naming Architecture.

- \* Restructuring the draft. 1) We start from the homenet-arch draft to derive a Naming Architecture, then 2) we show why CPE need mechanisms that do not expose them to the Internet, 3) we describe the mechanisms.

- \* I remove the terminology and expose it in the figures A and B.

- \* remove the Front End Homenet Naming Architecture to Homenet Naming  
-01:

- \* Added C. Griffiths as co-author.

- \* Updated section 5.4 and other sections of draft to update section on Hidden Primary / Slave functions with CPE as Hidden Primary/Homenet Server.

- \* For next version, address functions of MDNS within Homenet Lan and publishing details northbound via Hidden Primary.

-00: First version published.

#### Authors' Addresses

Daniel Migault  
Ericsson  
8400 Boulevard Decarie  
Montreal, QC H4P 2N2  
Canada

Phone: +1 (514) 452-2160  
Email: [daniel.migault@ericsson.com](mailto:daniel.migault@ericsson.com)

Ralf Weber  
Nominum  
2000 Seaport Blvd #400  
Redwood City, CA 94063  
US

Email: [ralf.weber@nominum.com](mailto:ralf.weber@nominum.com)  
URI: <http://www.nominum.com>



Ray Hunter  
Globis Consulting BV  
Weegschaalstraat 3  
5632CW Eindhoven  
The Netherlands

Email: [v6ops@globis.net](mailto:v6ops@globis.net)  
URI: <http://www.globis.net>

Chris Griffiths

Email: [cgriffiths@gmail.com](mailto:cgriffiths@gmail.com)

Wouter Cloetens  
SoftAtHome  
vaartdijk 3 701  
3018 Wijgmaal  
Belgium

Email: [wouter.cloetens@softathome.com](mailto:wouter.cloetens@softathome.com)

Homenet Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: March 3, 2016

M. Stenberg  
S. Barth  
Independent  
P. Pfister  
Cisco Systems  
August 31, 2015

Home Networking Control Protocol  
draft-ietf-homenet-hncp-09

Abstract

This document describes the Home Networking Control Protocol (HNCP), an extensible configuration protocol and a set of requirements for home network devices. HNCP is described as a profile of and extension to the Distributed Node Consensus Protocol (DNCP). HNCP enables discovery of network borders, automated configuration of addresses, name resolution, service discovery, and the use of any routing protocol which supports routing based on both source and destination address.

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 March 3, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
1.1. Applicability . . . . .	4
2. Terminology . . . . .	5
2.1. Requirements language . . . . .	6
3. DNCP Profile . . . . .	7
4. HNCP Versioning and Router Capabilities . . . . .	8
5. Interface Classification . . . . .	9
5.1. Interface Categories . . . . .	9
5.2. DHCP Aided Auto-Detection . . . . .	10
5.3. Algorithm for Border Discovery . . . . .	10
6. Autonomous Address Configuration . . . . .	11
6.1. Common Link . . . . .	11
6.2. External Connections . . . . .	12
6.3. Prefix Assignment . . . . .	13
6.3.1. Prefix Assignment Algorithm Parameters . . . . .	13
6.3.2. Making New Assignments . . . . .	15
6.3.3. Applying Assignments . . . . .	16
6.3.4. DHCPv6 Prefix Delegation . . . . .	16
6.4. Node Address Assignment . . . . .	16
6.5. Local IPv4 and ULA Prefixes . . . . .	17
7. Configuration of Hosts and non-HNCP Routers . . . . .	18
7.1. IPv6 Addressing and Configuration . . . . .	19
7.2. DHCPv6 for Prefix Delegation . . . . .	19
7.3. DHCPv4 for Addressing and Configuration . . . . .	20
7.4. Multicast DNS Proxy . . . . .	20
8. Naming and Service Discovery . . . . .	20
9. Securing Third-Party Protocols . . . . .	21
10. Type-Length-Value Objects . . . . .	22
10.1. HNCP Version TLV . . . . .	22
10.2. External Connection TLV . . . . .	23
10.2.1. Delegated Prefix TLV . . . . .	24
10.2.2. DHCPv6 Data TLV . . . . .	25
10.2.3. DHCPv4 Data TLV . . . . .	26
10.3. Assigned Prefix TLV . . . . .	26
10.4. Node Address TLV . . . . .	27
10.5. DNS Delegated Zone TLV . . . . .	28
10.6. Domain Name TLV . . . . .	29
10.7. Node Name TLV . . . . .	29
10.8. Managed PSK TLV . . . . .	30
11. General Requirements for HNCP Nodes . . . . .	31

12. Security Considerations . . . . .	32
12.1. Interface Classification . . . . .	33
12.2. Security of Unicast Traffic . . . . .	33
12.3. Other Protocols in the Home . . . . .	34
13. IANA Considerations . . . . .	34
14. References . . . . .	35
14.1. Normative references . . . . .	35
14.2. Informative references . . . . .	36
Appendix A. Changelog [RFC Editor: please remove] . . . . .	37
Appendix B. Draft source [RFC Editor: please remove] . . . . .	38
Appendix C. Implementation [RFC Editor: please remove] . . . . .	38
Appendix D. Acknowledgements . . . . .	38
Authors' Addresses . . . . .	39

## 1. Introduction

HNCP is designed to facilitate sharing of state among home routers to fulfill the needs of the IPv6 homenet architecture [RFC7368], which assumes zero-configuration operation, multiple subnets, multiple home routers and (potentially) multiple upstream service providers providing (potentially) multiple prefixes to the home network. While RFC7368 sets no requirements for IPv4 support, HNCP aims to support dual-stack mode of operation, and therefore the functionality is designed with that in mind. The state is shared as TLVs among the routers (and potentially advanced hosts) to enable:

- o Autonomic discovery of network borders (Section 5.3) based on DNCP topology.
- o Automated portioning of prefixes delegated by the service providers as well as assigned prefixes to both HNCP and non-HNCP routers (Section 6.3) using [I-D.ietf-homenet-prefix-assignment]. Prefixes assigned to HNCP routers are used to:
  - \* Provide addresses to non-HNCP aware nodes (using SLAAC and DHCP).
  - \* Provide space in which HNCP nodes assign their own addresses (Section 6.4).
- o Internal and external name resolution, as well as multi-link service discovery (Section 8).
- o Other services not defined in this document, that do need to share state among homenet nodes, and do not cause rapid and constant TLV changes (see following applicability section).

HNCP is a DNCP [I-D.ietf-homenet-dncp]-based protocol and includes a DNCP profile which defines transport and synchronization details for sharing state across nodes defined in Section 3. The rest of the document defines behavior of the services noted above, how the required TLVs are encoded (Section 10), as well as additional requirements on how HNCP nodes should behave (Section 11).

### 1.1. Applicability

While HNCP does not deal with routing protocols directly (except potentially informing them about internal and external interfaces if classification specified in Section 5.3 is used), in homenet environments where multiple IPv6 source-prefixes can be present, routing based on source and destination address is necessary [RFC7368]. Ideally, the routing protocol is also zero-configuration (e.g., no need to configure identifiers or metrics) although HNCP can be used also with a manually configured routing protocol.

As HNCP uses DNCP as the actual state synchronization protocol, the applicability statement of DNCP can be used here as well; HNCP should not be used for any data that changes rapidly and constantly, and locators to find such services should be published using it instead. This is why the naming and service discovery (Section 8) TLVs contain only DNS server addresses, and no actual per-name or per-service data of hosts.

HNCP TLVs specified within this document, in steady state, stay constant, with one exception: as Delegated Prefix TLVs (Section 10.2.1) do contain lifetimes, they force re-publishing of that data every time the valid or preferred lifetimes of prefixes are updated (significantly). Therefore, it is desirable for ISPs to provide large enough valid and preferred lifetimes to avoid unnecessary HNCP state churn in homes, but even given non-cooperating ISPs, the state churn is proportional only to the number of externally received delegated prefixes and not the home network size, and should therefore be relatively low.

HNCP assumes a certain level of control over host configuration servers (e.g., DHCP [RFC2131]) on links that are managed by its routers. Some HNCP functionality (such as border discovery or some aspects of naming) might be affected by existing DHCP servers not aware of the HNCP-managed network and thus might need to be reconfigured to not result in unexpected behavior.

While HNCP is designed to be used by (home) routers, it can also be used by advanced hosts that want to do, e.g., their own address assignment and routing.

HNCP is link layer agnostic; if a link supports IPv6 (link-local) multicast and unicast, HNCP will work on it. Trickle retransmissions and keep-alives will handle both packet loss and non-transitive connectivity, ensuring eventual convergence.

## 2. Terminology

The following terms are used as they are defined in [I-D.ietf-homenet-prefix-assignment]:

- o Advertised Prefix Priority
- o Advertised Prefix
- o Assigned Prefix
- o Delegated Prefix
- o Prefix Adoption
- o Private Link
- o Published Assigned Prefix
- o Applied Assigned Prefix
- o Shared Link

The following terms are used as they are defined in [I-D.ietf-homenet-dncp]:

- o DNCP profile
- o Node identifier
- o Link
- o Interface

(HNCP) node	A device implementing this specification.
(HNCP) router	A device implementing this specification, which forwards traffic on behalf of other devices.
Border	separation point between administrative domains; in this case, between the home network and any other network, i.e., usually an ISP network.
Internal link	a link that does not cross borders.
Internal interface	an interface that is connected to an internal link.
External interface	an interface that is connected to a link which is not an internal link.
Interface category	a local configuration denoting the use of a particular interface. The interface category determines how a HNCP node should treat the particular interface. External and internal category mark the interface as out of or within the network border; there are also a number of sub-categories to internal that further affect local node behavior. See Section 5.1 for a list of interface categories and how they behave. The internal or external categories may also be auto-detected (Section 5.3).
Border router	a router announcing external connectivity and forwarding traffic across the network border.
Common Link	a set of nodes on a link which share a common view of it, i.e., they see each other's traffic and the same set of hosts. Unless configured otherwise transitive connectivity is assumed.
DHCPv4	refers to Dynamic Host Configuration Protocol [RFC2131] in this document.
DHCPv6	refers to Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC3315] in this document.
DHCP	refers to cases which apply to both DHCPv4 and DHCPv6 in this document.

## 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. DNCP Profile

The DNCP profile of HNCP is defined as follows:

- o HNCP uses UDP datagrams on port HNCP-UDP-PORT as a transport over link-local scoped IPv6, using unicast and multicast (All-Homenet-Nodes is the HNCP group address). Received datagrams where either or both of the IPv6 source or destination address is not link-local scoped MUST be ignored. Replies to multicast and unicast messages MUST be sent to the IPv6 source address and port of the original message. Each node MUST be able to receive (and potentially reassemble) UDP datagrams with a payload of at least 4000 bytes.
- o HNCP operates on multicast-capable interfaces only. HNCP nodes MUST assign a locally unique non-zero 32-bit endpoint identifier to each interface for which HNCP is enabled. The value zero is not used in DNCP TLVs, but it has a special meaning in HNCP TLVs (see Section 10.3 and Section 6.4). Implementations MAY use a value equivalent to the IPv6 link-local scope identifier for the given interface.
- o HNCP uses opaque 32-bit node identifiers (DNCP\_NODE\_IDENTIFIER\_LENGTH = 32). A node implementing HNCP SHOULD use a random node identifier. If there is a node identifier collision (as specified in the Node State TLV handling of Section 4.4 of [I-D.ietf-homenet-dncp]), the node MUST immediately generate and use a new random node identifier which is not used by any other node at the time, based on the current DNCP network state.
- o HNCP nodes MUST use the leading 64 bits of MD5 [RFC1321] as DNCP non-cryptographic hash function H(x).
- o HNCP nodes MUST use DNCP's per-endpoint keep-alive extension on all endpoints. The following parameters are suggested:
  - \* Default keep-alive interval (DNCP\_KEEPA\_LIVE\_INTERVAL): 20 seconds.
  - \* Multiplier (DNCP\_KEEPA\_LIVE\_MULTIPLIER): 2.1 on virtually lossless links works fine as it allows for one lost keep-alive. If used on a lossy link, considerably higher multiplier, such as 15, should be used instead. In that case, an implementation might prefer shorter keep-alive intervals on that link as well to ensure that DNCP\_KEEPA\_LIVE\_INTERVAL \* DNCP\_KEEPA\_LIVE\_MULTIPLIER timeout after which (entirely) lost nodes time out is low enough.



- o HNCP nodes use the following Trickle parameters for the per-interface Trickle instances:
  - \* k SHOULD be 1, as the timer reset when data is updated and further retransmissions should handle packet loss. Even on a non-transitive lossy link, the eventual per-endpoint keep-alives should ensure status synchronization occurs.
  - \* Imin SHOULD be 200 milliseconds but MUST NOT be lower. Note: Earliest transmissions may occur at  $Imin / 2$ .
  - \* Imax SHOULD be 7 doublings of Imin (i.e., 25.6 seconds) but MUST NOT be lower.
- o HNCP unicast traffic SHOULD be secured using DTLS [RFC6347] as described in DNCP if exchanged over unsecured links. UDP on port HNCP-DTLS-PORT is used for this purpose. A node implementing HNCP security MUST support the DNCP Pre-Shared Key method, SHOULD support the DNCP Certificate Based Trust Consensus and MAY support the PKI-based trust method.
- o HNCP nodes MUST ignore all Node State TLVs received via multicast on a link which has DNCP security enabled in order to prevent spoofing of node state changes.

#### 4. HNCP Versioning and Router Capabilities

Multiple versions of HNCP based on compatible DNCP profiles may be present in the same network when transitioning between HNCP versions and for troubleshooting purposes it might be beneficial to identify the HNCP agent version running. Therefore each node MUST include an HNCP-Version TLV (Section 10.1) in its Node Data and MUST ignore (except for DNCP synchronization purposes) any TLVs with a type greater than 32 published by nodes not also publishing an HNCP-Version TLV.

HNCP routers may also have different capabilities regarding interactions with hosts, e.g., for configuration or service discovery. These are indicated by M, P, H and L values. The combined "capability value" is a metric indicated by interpreting the bits as an integer, i.e.,  $(M \ll 12 \mid P \ll 8 \mid H \ll 4 \mid L)$ . These values are used to elect certain servers on a Common Link, as described in Section 7. Nodes that are not routers MUST announce the value 0 for all capabilities. Any node announcing the value 0 is considered to not advertise the respective capability and thus does not take part in the respective election.

## 5. Interface Classification

### 5.1. Interface Categories

HNCP specifies the following categories interfaces can be configured to be in:

**Internal category:** This declares an interfaces to be internal, i.e., within the borders of the HNCP network. HNCP traffic **MUST** be sent and received. Routers **MUST** forward traffic with appropriate source addresses between their internal interfaces and allow internal traffic to reach external networks. All nodes **MUST** implement this category and nodes not implementing any other category implicitly use it as a fixed default.

**External category:** This declares an interface to be external, i.e., not within the borders of the HNCP network. HNCP traffic **MUST** neither be sent nor received. Accessing internal resources from external interfaces is restricted, i.e., the use of [RFC6092] is **RECOMMENDED**. HNCP routers **SHOULD** announce acquired configuration information for use in the network as described in Section 6.2, if the interface appears to be connected to an external network. HNCP routers **MUST** implement this category.

**Leaf category:** This declares an interface used by client devices only. Such an interface uses the Internal category with the exception that HNCP traffic **MUST NOT** be sent on the interface, and all such traffic received on the interface **MUST** be ignored. This category **SHOULD** be supported by HNCP routers.

**Guest category:** This declares an interface used by untrusted client devices only. In addition to the restrictions of the Leaf category, HNCP routers **MUST** filter traffic from and to the interface such that connected devices are unable to reach other devices inside the HNCP network or query services advertised by them unless explicitly allowed. This category **SHOULD** be supported by HNCP routers.

**Ad-hoc category:** This configures an interface to use the Internal category but no assumption is made about the the link's transitivity. All other interface categories assume transitive connectivity. This affects the Common Link (Section 6.1) definition. Support for this category is **OPTIONAL**.

**Hybrid category:** This declares an interface to use the Internal category while still trying to acquire (external) configuration information on it, e.g., by running DHCP clients. This is useful, e.g., if the link is shared with a non-HNCP router under control

and still within the borders of the same network. Detection of this category automatically in addition to manual configuration is out of scope of this document. Support for this category is OPTIONAL.

## 5.2. DHCP Aided Auto-Detection

Auto-detection of interface categories is possible based on interaction with DHCPv4 [RFC2131] and DHCPv6-PD [RFC3633] servers on connected links. HNCP defines special DHCP behavior to differentiate its internal servers from external ones in order to achieve this. Therefore all internal devices (including HNCP nodes) running DHCP servers on links where auto-detection is used by any HNCP node MUST use the following mechanism based on The User Class Option for DHCPv4 [RFC3004] and its DHCPv6 counterpart [RFC3315]:

- o The device MUST ignore or reject DHCP-Requests containing a DHCP User-Class consisting of the ASCII-String "HOMENET".

Not following this rule (e.g., running unmodified DHCP servers) might lead to false positives when auto-detection is used, i.e., HNCP nodes assume an interface to not be internal, even though it was intended to be.

## 5.3. Algorithm for Border Discovery

This section defines the interface classification algorithm. It is suitable for both IPv4 and IPv6 (single or dual-stack) and detects the category of an interface either automatically or based on a fixed configuration. By determining the category for all interfaces, the network borders are implicitly defined, i.e., all interfaces not belonging to the External category are considered to be within the borders of the network, all others are not.

The following algorithm MUST be implemented by any node implementing HNCP. However, if the node does not implement auto-detection, only the first step is required. The algorithm works as follows, with evaluation stopping at first match:

1. If a fixed category is configured for an interface, it is used.
2. If a delegated prefix could be acquired by running a DHCPv6 client, it is considered external. The DHCPv6 client MUST have included a DHCPv6 User-Class consisting of the ASCII-String "HOMENET" in all of its requests.
3. If an IPv4 address could be acquired by running a DHCPv4 client on the interface, it is considered external. The DHCPv4 client

MUST have included a DHCP User-Class consisting of the ASCII-String "HOMENET" in all of its requests.

4. The interface is considered internal.

Note that as other HNCP nodes will ignore the client due to the user class option, any server that replies is clearly external (or a malicious internal node).

An HNCP router SHOULD allow setting the fixed category for each interface which may be connected to either an internal or external device (e.g., an Ethernet port that can be connected to a modem, another HNCP router or a client).

An HNCP router using auto-detection on an interface MUST run the appropriately configured DHCP clients as long as the interface without a fixed category is active (including states where auto-detection considers it to be internal) and rerun the algorithm above to react to conditions resulting in a different interface category. The router SHOULD wait for a reasonable time period (5 seconds as a default), during which the DHCP clients can acquire a lease, before treating a newly activated or previously external interface as internal.

6. Autonomous Address Configuration

This section specifies how HNCP nodes configure host and node addresses. At first border routers share information obtained from service providers or local configuration by publishing one or more External Connection TLVs (Section 10.2). These contain other TLVs such as Delegated Prefix TLVs (Section 10.2.1) which are then used for prefix assignment. Finally, HNCP nodes obtain addresses either statelessly or using a specific stateful mechanism (Section 6.4). Hosts and non-HNCP routers are configured using SLAAC, DHCP or DHCPv6-PD.

6.1. Common Link

HNCP uses the concept of Common Link both in autonomic address configuration and naming and service discovery (Section 8). A Common Link refers to the set of interfaces of nodes that see each other's traffic and presumably also the traffic of all hosts that may use the nodes to, e.g., forward traffic. Common Links are used, e.g., to determine where prefixes should be assigned or which peers participate in the election of a DHCP server. The Common Link is computed separately for each local internal interface, and it always contains the local interface. Additionally, if the local interface is not set to ad-hoc category (see Section 5.1), it also contains the

set of interfaces that are bidirectionally reachable from the given local interface, that is, every remote interface of a remote node meeting all of the following requirements:

- o The local node publishes a Peer TLV with:
  - \* Peer Node Identifier = remote node's node identifier
  - \* Peer Endpoint Identifier = remote interface's endpoint identifier
  - \* Endpoint Identifier = local interface's endpoint identifier
- o The remote node publishes a Peer TLV with:
  - \* Peer Node Identifier = local node's node identifier
  - \* Peer Endpoint Identifier = local interface's endpoint identifier
  - \* Endpoint Identifier = remote interface's endpoint identifier

A node MUST be able to detect whether two of its local internal interfaces are connected, e.g., by detecting an identical remote interface being part of the Common Links of both local interfaces.

## 6.2. External Connections

Each HNCP router MAY obtain external connection information such as address prefixes, DNS server addresses and DNS search paths from one or more sources, e.g., DHCPv6-PD [RFC3633], NETCONF [RFC6241] or static configuration. Each individual external connection to be shared in the network is represented by one External Connection TLV (Section 10.2).

Announcements of individual external connections may consist of the following components:

**Delegated Prefixes:** address space available for assignment to internal links announced using Delegated Prefix TLVs (Section 10.2.1). Some address spaces might have special properties which are necessary to understand in order to handle them (e.g., information similar to [RFC6603]). This information is encoded using DHCPv6 Data TLVs (Section 10.2.2) inside the respective Delegated Prefix TLVs.

**Auxiliary Information:** information about services such as DNS or time synchronization regularly used by hosts in addition to

addressing and routing information. This information is encoded using DHCPv6 Data TLVs (Section 10.2.2) and DHCPv4 Data TLVs (Section 10.2.3).

Whenever information about reserved parts (e.g., as specified in [RFC6603]) is received for a delegated prefix, the reserved parts MUST be advertised using Assigned Prefix TLVs (Section 10.3) with the highest priority (i.e., 15), as if they were assigned to a Private Link.

Some connections or delegated prefixes may have a special meaning and are not regularly used for internal or internet connectivity, instead they may provide access to special services like VPNs, sensor networks, VoIP, IPTV, etc. Care must be taken that these prefixes are properly integrated and dealt with in the network, in order to avoid breaking connectivity for devices who are not aware of their special characteristics or to only selectively allow certain devices to use them. Such prefixes are distinguished using Prefix Policy TLVs (Section 10.2.1.1). Their contents MAY be partly opaque to HNCP nodes, and their identification and usage depends on local policy. However the following general rules MUST be adhered to:

Special rules apply when making address assignments for prefixes with Prefix Policy TLVs with type 131, as described in Section 6.3.2

In presence of any type 1 to 128 Prefix Policy TLV the prefix is specialized to reach destinations denoted by any such Prefix Policy TLV, i.e., in absence of a type 0 Prefix Policy TLV it is not usable for general internet connectivity. An HNCP router MAY enforce this restriction with appropriate packet filter rules.

### 6.3. Prefix Assignment

HNCP uses the Prefix Assignment Algorithm [I-D.ietf-homenet-prefix-assignment] in order to assign prefixes to HNCP internal links and uses some of the terminology (Section 2) defined there. HNCP furthermore defines the Assigned Prefix TLV (Section 10.3) which MUST be used to announce Published Assigned Prefixes.

#### 6.3.1. Prefix Assignment Algorithm Parameters

All HNCP nodes running the prefix assignment algorithm use the following values for its parameters:

Node IDs: HNCP node identifiers are used. The comparison operation is defined as bit-wise comparison.

**Set of Delegated Prefixes:** The set of prefixes encoded in Delegated Prefix TLVs which are not strictly included in prefixes encoded in other Delegated Prefix TLVs. Note that Delegated Prefix TLVs included in ignored External Connection TLVs are not considered. It is dynamically updated as Delegated Prefix TLVs are added or removed.

**Set of Shared Links:** The set of Common Links associated with interfaces with internal, leaf, guest or ad-hoc category. It is dynamically updated as interfaces are added, removed, or switch from one category to another. When multiple interfaces are detected as belonging to the same Common Link, prefix assignment is disabled on all of these interfaces except one.

**Set of Private Links:** This document defines Private Links representing DHCPv6-PD clients or as a mean to advertise prefixes included in the DHCPv6 Exclude Prefix option. Other implementation-specific Private Links may be defined whenever a prefix needs to be assigned for a purpose that does not require a consensus with other HNCP nodes.

**Set of Advertised Prefixes:** The set of prefixes included in Assigned Prefix TLVs advertised by other HNCP nodes (Prefixes advertised by the local node are not in this set). The associated Advertised Prefix Priority is the priority specified in the TLV. The associated Shared Link is determined as follows:

- \* If the Link Identifier is zero, the Advertised Prefix is not assigned on a Shared Link.
- \* If the other node's interface identified by the Link Identifier is included in one of the Common Links used for prefix assignment, it is considered as assigned on the given Common Link.
- \* Otherwise, the Advertised Prefix is not assigned on a Shared Link.

Advertised Prefixes as well as their associated priorities and associated Shared Links MUST be updated as Assigned Prefix TLVs are added, updated or removed, and as Common Links are modified.

**ADOPT\_MAX\_DELAY:** The default value is 0 seconds (i.e., prefix adoption MAY be done instantly).

**BACKOFF\_MAX\_DELAY:** The default value is 4 seconds.

**RANDOM\_SET\_SIZE:** The default value is 64.

Flooding Delay: The default value is 5 seconds.

Default Advertised Prefix Priority: When a new assignment is created or an assignment is adopted - as specified in the prefix assignment algorithm routine - the default Advertised Prefix Priority to be used is 2.

### 6.3.2. Making New Assignments

Whenever the prefix assignment algorithm subroutine (Section 4.1 of [I-D.ietf-homenet-prefix-assignment]) is run on a Common Link and whenever a new prefix may be assigned (case 1 of the subroutine: no Best Assignment and no Current Assignment), the decision of whether the assignment of a new prefix is desired MUST follow these rules in order:

If the Delegated Prefix TLV contains a DHCPv6 Data TLV, and the meaning of one of the DHCP options is not understood by the HNCP node, the creation of a new prefix is not desired. This rule applies to TLVs inside Delegated Prefix TLVs but not to those inside External Connection TLVs.

If the remaining preferred lifetime of the prefix is 0 and there is another delegated prefix of the same IP version used for prefix assignment with a non-zero preferred lifetime, the creation of a new prefix is not desired.

If the Delegated Prefix does not include a Prefix Policy TLV indicating restrictive assignment (type 131) or if local policy exists to identify it based on, e.g., other Prefix Policy TLV values and allows assignment, the creation of a new prefix is desired.

Otherwise, the creation of a new prefix is not desired.

If the considered delegated prefix is an IPv6 prefix, and whenever there is at least one available prefix of length 64, a prefix of length 64 MUST be selected unless configured otherwise. In case no prefix of length 64 would be available, a longer prefix MAY be selected even without configuration.

If the considered delegated prefix is an IPv4 prefix (Section 6.5 details how IPv4 delegated prefixes are generated), a prefix of length 24 SHOULD be preferred.

In any case, an HNCP router making an assignment MUST support a mechanism suitable to distribute addresses from the considered prefix if the link is intended to be used by clients. In this case a router



assigning an IPv4 prefix MUST announce the L-capability and a router assigning an IPv6 prefix with a length greater than 64 MUST announce the H-capability as defined in Section 4.

#### 6.3.3. Applying Assignments

The prefix assignment algorithm indicates when a prefix is applied to the respective Common Link. When that happens each router connected to said link:

MUST forward traffic destined to said prefix to the respective link.

MUST participate in the client configuration election as described in Section 7, if the link is intended to be used by clients.

MAY add an address from said prefix to the respective network interface as described in Section 6.4, e.g., if it is to be used as source for locally originating traffic.

#### 6.3.4. DHCPv6 Prefix Delegation

When an HNCP router announcing the P-Capability (Section 4) receives a DHCPv6-PD request from a client, it SHOULD assign one prefix per delegated prefix in the network. This set of assigned prefixes is then delegated to the client, after it has been applied as described in the prefix assignment algorithm. Each DHCPv6-PD client MUST be considered as an independent Private Link and delegation MUST be based on the same set of Delegated Prefixes as the one used for Common Link prefix assignments, however the prefix length to be delegated MAY be smaller than 64.

The assigned prefixes MUST NOT be given to DHCPv6-PD clients before they are applied, and MUST be withdrawn whenever they are destroyed. As an exception to this rule, in order to shorten delays of processed requests, a router MAY prematurely give out a prefix which is advertised but not yet applied if it does so with a valid lifetime of not more than 30 seconds and ensures removal or correction of lifetimes as soon as possible.

#### 6.4. Node Address Assignment

This section specifies how HNCP nodes reserve addresses for their own use. Nodes MAY, at any time, try to reserve a new address from any Applied Assigned Prefix. Each HNCP node SHOULD announce an IPv6 address and - if it supports IPv4 - MUST announce an IPv4 address, whenever matching prefixes are assigned to at least one of its Common Links. These addresses are published using Node Address TLVs and

used to locally reach HNCP nodes for other services. Nodes SHOULD NOT create and announce more than one assignment per IP version to avoid cluttering the node data with redundant information unless a special use case requires it.

Stateless assignment based on Semantically Opaque Interface Identifiers [RFC7217] SHOULD be used for address assignment whenever possible (e.g., the prefix length is 64), otherwise (e.g., for IPv4 if supported) the following method MUST be used instead: For any assigned prefix for which stateless assignment is not used, the first quarter of the addresses are reserved for HNCP based address assignments, whereas the last three quarters are left to the DHCP elected router (Section 4 specifies the DHCP server election process). For example, if the prefix 192.0.2.0/24 is assigned and applied to a Common Link, addresses included in 192.0.2.0/26 are reserved for HNCP nodes and the remaining addresses are reserved for the elected DHCPv4 server.

HNCP nodes assign themselves addresses, and then (to ensure eventual lack of conflicting assignments) publish the assignments using the Node Address TLV (Section 10.4).

The process of obtaining addresses is specified as follows:

- o A node MUST NOT start advertising an address if it is already advertised by another node.
- o An assigned address MUST be part of an assigned prefix currently applied on a Common Link which includes the interface specified by the endpoint identifier.
- o An address MUST NOT be used unless it has been advertised for at least ADDRESS\_APPLY\_DELAY consecutive seconds, and is still currently being advertised. The default value for ADDRESS\_APPLY\_DELAY is 3 seconds.
- o Whenever the same address is advertised by more than one node, all but the one advertised by the node with the highest node identifier MUST be removed.

#### 6.5. Local IPv4 and ULA Prefixes

HNCP routers can create a ULA or private IPv4 prefix to enable connectivity between local devices. These prefixes are inserted in HNCP as if they were delegated prefixes of a (virtual) external connection (Section 6.2). The following rules apply:

An HNCP router SHOULD create a ULA prefix if there is no other IPv6 prefix with a preferred time greater than 0 in the network. It MAY also do so, if there are other delegated IPv6 prefixes, but none of which is locally generated (i.e., without any Prefix Policy TLV) and has a preferred time greater than 0. However, it MUST NOT do so otherwise. In case multiple locally generated ULA prefixes are present, only the one published by the node with the highest node identifier is kept among those with a preferred time greater than 0 - if there is any.

An HNCP router MUST create a private IPv4 prefix [RFC1918] whenever it wishes to provide IPv4 internet connectivity to the network and no other private IPv4 prefix with internet connectivity currently exists. It MAY also enable local IPv4 connectivity by creating a private IPv4 prefix if no IPv4 prefix exists but MUST NOT do so otherwise. In case multiple IPv4 prefixes are announced, only the one published by the node with the highest node identifier is kept among those with a Prefix Policy of type 0 - if there is any. The router publishing a prefix with internet connectivity MUST forward IPv4 traffic to the internet and perform NAT on behalf of the network as long as it publishes the prefix, other routers in the network MAY choose not to.

Creation of such ULA and IPv4 prefixes MUST be delayed by a random timespan between 0 and 10 seconds in which the router MUST scan for others trying to do the same.

When a new ULA prefix is created, the prefix is selected based on the configuration, using the last non-deprecated ULA prefix, or generated based on [RFC4193].

## 7. Configuration of Hosts and non-HNCP Routers

HNCP routers need to ensure that hosts and non-HNCP downstream routers on internal links are configured with addresses and routes. Since DHCP clients can usually only bind to one server at a time, a per-link and per-service election takes place.

HNCP routers may have different capabilities for configuring downstream devices and providing naming services. Each router MUST therefore indicate its capabilities as specified in Section 4 in order to participate as a candidate in the election.

### 7.1. IPv6 Addressing and Configuration

In general Stateless Address Autoconfiguration [RFC4861] is used for client configuration for its low overhead and fast renumbering capabilities. Therefore each HNCP router sends Router Advertisements on interfaces which are intended to be used by clients and MUST at least include a Prefix Information Option for each Applied Assigned Prefix which it assigned to the respective link in every such advertisement. However, stateful DHCPv6 can be used in addition by administrative choice, to, e.g., collect hostnames and use them to provide naming services or whenever stateless configuration is not applicable.

The designated stateful DHCPv6 server for a Common Link (Section 6.1) is elected based on the capabilities described in Section 4. The winner is the router (connected to the Common Link) advertising the greatest H-capability. In case of a tie, Capability Values (Section 4) are compared, and the router with the greatest value is elected. In case of another tie, the router with the highest node identifier is elected among the routers with tied Capability Values.

The elected router MUST serve stateful DHCPv6 and SHOULD provide naming services for acquired hostnames as outlined in Section 8, all others nodes MUST NOT. Stateful addresses SHOULD be assigned in a way not hindering fast renumbering even if the DHCPv6 server or client do not support the DHCPv6 reconfigure mechanism, e.g., by only handing out leases from locally-generated (ULA) prefixes and prefixes with a length different from 64, and by using low renew and rebind times (i.e., not longer than 5 minutes). In case no router was elected, stateful DHCPv6 is not provided. Routers which cease to be elected DHCP servers SHOULD - when applicable - invalidate remaining existing bindings in order to trigger client reconfiguration.

### 7.2. DHCPv6 for Prefix Delegation

The designated DHCPv6 server for prefix-delegation on a Common Link is elected based on the capabilities described in Section 4. The winner is the router (connected to the Common Link) advertising the greatest P-capability. In case of a tie, Capability Values (Section 4) are compared, and the router with the greatest value is elected. In case of another tie, the router with the highest node identifier is elected among the routers with tied Capability Values.

The elected router MUST provide prefix-delegation services [RFC3633] on the given link (and follow the rules in Section 6.3.4), all other nodes MUST NOT.

### 7.3. DHCPv4 for Addressing and Configuration

The designated DHCPv4 server on a Common Link (Section 6.1) is elected based on the capabilities described in Section 4. The winner is the router (connected to the Common Link) advertising the greatest L-capability. In case of a tie, Capability Values (Section 4) are compared, and the router with the greatest value is elected. In case of another tie, the router with the highest node identifier is elected among the routers with tied Capability Values.

The elected router **MUST** provide DHCPv4 services on the given link, all other nodes **MUST NOT**. The elected router **MUST** provide IP addresses from the pool defined in Section 6.4 and **MUST** announce itself as router [RFC2132] to clients.

DHCPv4 lifetimes renew and rebind times (T1 and T2) **SHOULD** be short (i.e., not longer than 5 minutes) in order to provide reasonable response times to changes. Routers which cease to be elected DHCP servers **SHOULD** - when applicable - invalidate remaining existing bindings in order to trigger client reconfiguration.

### 7.4. Multicast DNS Proxy

The designated MDNS [RFC6762] proxy on a Common Link is elected based on the capabilities described in Section 4. The winner is the router (connected to the Common Link) advertising the greatest M-capability. In case of a tie, Capability Values (Section 4) are compared, and the router with the greatest value is elected. In case of another tie, the router with the highest node identifier is elected among the routers with tied Capability Values.

The elected router **MUST** provide an MDNS-proxy on the given link and announce it as described in Section 8.

## 8. Naming and Service Discovery

Network-wide naming and service discovery can greatly improve the user-friendliness of a network. The following mechanism provides means to setup and delegate naming and service discovery across multiple HNCP routers.

Each HNCP router **SHOULD** provide and advertise a recursive name resolving server to clients which honors the announcements made in Delegated Zone TLVs (Section 10.5), Domain Name TLVs (Section 10.6) and Node Name TLVs (Section 10.7), i.e., delegate queries to the designated name servers and hand out appropriate A, AAAA and PTR records according to the mentioned TLVs.

Each HNCP router SHOULD provide and announce an auto-generated or user-configured name for each internal Common Link (Section 6.1) for which it is the designated DHCPv4, stateful DHCPv6 server, MDNS proxy, or for which it provides forward or reverse DNS services on behalf of connected devices. This announcement is done using Delegated Zone TLVs (Section 10.5) and MUST be unique in the whole network. In case of a conflict the announcement of the node with the highest node identifier takes precedence and all other nodes MUST cease to announce the conflicting TLV. HNCP routers providing recursive name resolving services MUST use the included DNS server address within the TLV to resolve names belonging to the zone as if there was an NS record.

Each HNCP node SHOULD announce a node name for itself to be easily reachable and MAY do so on behalf of other devices. Announcements are made using Node Name TLVs (Section 10.7) and MUST be unique in the whole network. In case of a conflict the announcement of the node with the highest node identifier takes precedence and all other nodes MUST cease to announce the conflicting TLV. HNCP routers providing recursive name resolving services as described above MUST resolve such announced names to their respective IP addresses as if there were corresponding A/AAAA records.

Names and unqualified zones are used in an HNCP network to provide naming and service discovery with local significance. A network-wide zone is appended to all single labels or unqualified zones in order to qualify them. ".home" is the default, however an administrator MAY configure announcing of a Domain Name TLV (Section 10.6) for the network to use a different one. In case multiple are announced, the domain of the node with the greatest node identifier takes precedence.

## 9. Securing Third-Party Protocols

Pre-shared keys (PSKs) are often required to secure (for example) IGPs and other protocols which lack support for asymmetric security. The following mechanism manages PSKs using HNCP to enable bootstrapping of such third-party protocols. The scheme SHOULD be used only in conjunction with secured HNCP unicast transport (=DTLS), as transferring the PSK in plain-text anywhere in the network is a potential risk, especially as the originator may not know about security (and use of DNCP security) on all links. The following rules define how such a PSK is managed and used:

- o If no Managed PSK TLV (Section 10.8) is currently being announced, an HNCP node using this mechanism MUST create one after a random delay of 0 to 10 seconds with a 32 bytes long random key and add it to its node data.



This TLV is used to indicate the supported version and router capabilities of an HNCP node as described in Section 4.

Reserved: Bits are reserved for future use. They MUST be set to zero when creating this TLV, and their value MUST be ignored when processing the TLV.

M-capability: Priority value used for electing the on-link MDNS [RFC6762] proxy. It MUST be set to some value between 1 and 7 included (4 is the default) if the router is capable of proxying MDNS and 0 otherwise. The values 8-15 are reserved for future use.

P-capability: Priority value used for electing the on-link DHCPv6-PD server. It MUST be set to some value between 1 and 7 included (4 is the default) if the router is capable of providing prefixes through DHCPv6-PD (Section 6.3.4) and 0 otherwise. The values 8-15 are reserved for future use.

H-capability: Priority value used for electing the on-link DHCPv6 server offering non-temporary addresses. It MUST be set to some value between 1 and 7 included (4 is the default) if the router is capable of providing such addresses and 0 otherwise. The values 8-15 are reserved for future use.

L-capability: Priority value used for electing the on-link DHCPv4 server. It MUST be set to some value between 1 and 7 included (4 is the default) if the router is capable of running a legacy DHCPv4 server offering IPv4 addresses to clients and 0 otherwise. The values 8-15 are reserved for future use.

User-Agent: The user-agent is a human-readable UTF-8 string that describes the name and version of the current HNCP implementation.

## 10.2. External Connection TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Type: EXTERNAL-CONNECTION (33) |                               Length |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               (Optional nested TLVs)                               |

```

An External Connection TLV is a container TLV used to gather network configuration information associated with a single external connection (Section 6.2) to be shared across the HNCP network. A node MAY publish an arbitrary number of instances of this TLV to share the desired number of external connections. Upon reception,



the information transmitted in any nested TLVs is used for the purposes of prefix assignment (Section 6.3) and host configuration (Section 7).

#### 10.2.1. Delegated Prefix TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Type: DELEGATED-PREFIX (34) | Length: >= 9 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Valid Lifetime Since Origination |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Preferred Lifetime Since Origination |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Prefix Length |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Prefix |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
...
| | 0-pad if any |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| (Optional nested TLVs) |

```

The Delegated Prefix TLV is used by HNCP routers to advertise prefixes which are allocated to the whole network and can be used for prefix assignment. Delegated Prefix TLVs are only valid inside External Connection TLVs and their prefixes MUST NOT overlap with those of other such TLVs in the same container.

**Valid Lifetime Since Origination:** The time in seconds the delegated prefix was valid for at the origination time of the node data containing this TLV. The value MUST be updated whenever the node republishes its Node State TLV.

**Preferred Lifetime Since Origination:** The time in seconds the delegated prefix was preferred for at the origination time of the node data containing this TLV. The value MUST be updated whenever the node republishes its Node State TLV.

**Prefix Length:** The number of significant bits in the Prefix.

**Prefix:** Significant bits of the prefix padded with zeroes up to the next byte boundary.

##### 10.2.1.1. Prefix Policy TLV

```

0                                     1                                     2                                     3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Type: PREFIX-POLICY (43)   |   Length: >= 1   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Policy Type |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Value                                     +
|

```

The Prefix Policy TLV contains information about the policy or applicability of a delegated prefix. This information can be used to determine whether prefixes for a certain usecase (e.g., local reachability, internet connectivity) do exist or should be acquired and to make decisions about assigning prefixes to certain links or to fine-tune border firewalls. See Section 6.2 for a more in-depth discussion. This TLV is only valid inside a Delegated Prefix TLV.

Policy Type: The type of the policy identifier.

- 0 : Internet connectivity (no Value).
- 1-128 : Explicit destination prefix with the Policy Type being the actual length of the prefix (Value contains significant bits of the destination prefix padded with zeroes up to the next byte boundary).
- 129 : DNS Zone (Value contains an RFC 1035 [RFC1035] encoded DNS label sequence).
- 130 : Opaque UTF-8 string (e.g., for administrative purposes).
- 131 : Restrictive Assignment (no Value).
- 132-255: Reserved for future additions.

Value: A variable length identifier of the given type.

#### 10.2.2. DHCPv6 Data TLV

```

0                                     1                                     2                                     3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Type: DHCPV6-DATA (37)   |   Length: > 0   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     DHCPv6 option stream                                     |

```

This TLV is used to encode auxiliary IPv6 configuration information (e.g., recursive DNS servers) encoded as a stream of DHCPv6 options.

It is only valid in an External Connection TLV or a Delegated Prefix TLV encoding an IPv6 prefix and MUST NOT occur more than once in any single container. When included in an External Connection TLV, it contains DHCPv6 options relevant to the External Connection as a whole. When included in a Delegated Prefix, it contains options mandatory to handle said prefix.

DHCPv6 option stream: DHCPv6 options encoded as specified in [RFC3315].

#### 10.2.3. DHCPv4 Data TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Type: DHCPV4-DATA (38)      |      Length: > 0      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                DHCPv4 option stream                                |

```

This TLV is used to encode auxiliary IPv4 configuration information (e.g., recursive DNS servers) encoded as a stream of DHCPv4 options. It is only valid in an External Connection TLV and MUST NOT occur more than once in any single container. It contains DHCPv4 options relevant to the External Connection as a whole.

DHCPv4 option stream: DHCPv4 options encoded as specified in [RFC2131].

#### 10.3. Assigned Prefix TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Type: ASSIGNED-PREFIX (35)      |      Length: >= 6      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                Endpoint Identifier                                |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|  Rsv. | Prty. | Prefix Length |                                |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                Prefix                                |
...
|                                | 0-pad if any |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                (Optional nested TLVs)                                |

```

This TLV is used to announce Published Assigned Prefixes for the purposes of prefix assignment (Section 6.3).

Endpoint Identifier: The endpoint identifier of the local interface the prefix is assigned to, or 0 if it is assigned to a Private Link (e.g., when the prefix is assigned for downstream prefix delegation).

Rsv.: Bits are reserved for future use. They MUST be set to zero when creating this TLV, and their value MUST be ignored when processing the TLV.

Prty: The Advertised Prefix Priority from 0 to 15.

0-1 : Low priorities.

2 : Default priority.

3-7 : High priorities.

8-11 : Administrative priorities. MUST NOT be used unless configured otherwise.

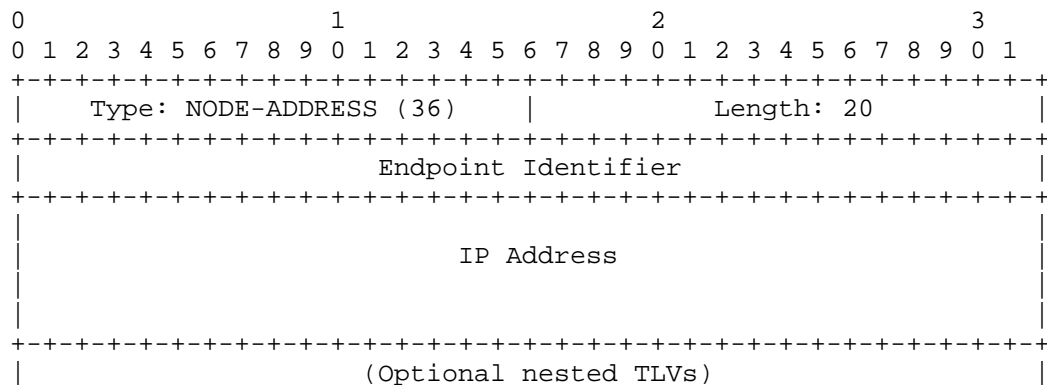
12-14: Reserved for future use.

15 : Provider priorities. MAY only be used by the router  
advertising the corresponding delegated prefix and based on  
static or dynamic configuration (e.g., for excluding a prefix  
based on DHCPv6-PD Prefix Exclude Option [RFC6603]).

Prefix Length: The number of significant bits in the Prefix field.

Prefix: The significant bits of the prefix padded with zeroes up to the next byte boundary.

#### 10.4. Node Address TLV



This TLV is used to announce addresses assigned to an HNCP node as described in Section 6.4.

**Endpoint Identifier:** The endpoint identifier of the local interface the prefix is assigned to, or 0 if it is not assigned on an HNCP enabled link.

**IP Address:** The globally scoped IPv6 address, or the IPv4 address encoded as an IPv4-mapped IPv6 address [RFC4291].

#### 10.5. DNS Delegated Zone TLV

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Type: DNS-DELEGATED-ZONE (39) | Length: >= 17 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               IP Address                               |
|                               |                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Reserved |L|B|S|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Zone (DNS label sequence - variable length) |
|                               |                               |
...                               | 0-pad if any |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               (Optional nested TLVs)                               |

```

This TLV is used to announce a forward or reverse DNS zone delegation in the HNCP network. Its meaning is roughly equivalent to specifying an NS and A/AAAA record for said zone. Details are specified in Section 8.

**IP Address :** The IPv6 address of the authoritative DNS server for the zone; IPv4 addresses are represented as IPv4-mapped addresses [RFC4291]. The special value of :: (all-zero) means the delegation is available in the global DNS-hierarchy.

**Reserved :** Those bits MUST be set to zero when creating the TLV and ignored when parsing it unless defined in a later specification.

**L-bit :** DNS-SD [RFC6763] Legacy-Browse, indicates that this delegated zone should be included in the network's DNS-SD legacy browse list of domains at lb.\_dns-sd.\_udp.(DOMAIN-NAME). Local forward zones SHOULD have this bit set, reverse zones SHOULD NOT.

B-bit : (DNS-SD [RFC6763] Browse) indicates that this delegated zone should be included in the network's DNS-SD browse list of domains at `b._dns-sd._udp. (DOMAIN-NAME)`. Local forward zones SHOULD have this bit set, reverse zones SHOULD NOT.

S-bit : (fully-qualified DNS-SD [RFC6763] domain) indicates that this delegated zone consists of a fully-qualified DNS-SD domain, which should be used as base for DNS-SD domain enumeration, i.e., `_dns-sd._udp.(Zone)` exists. Forward zones MAY have this bit set, reverse zones MUST NOT. This can be used to provision DNS search path to hosts for non-local services (such as those provided by an ISP, or other manually configured service providers). Zones with this flag SHOULD be added to the search domains advertised to clients.

Zone : The label sequence of the zone, encoded as the domain names are encoded DNS messages as specified in [RFC1035]. The last label in the zone MUST be empty.

#### 10.6. Domain Name TLV

```

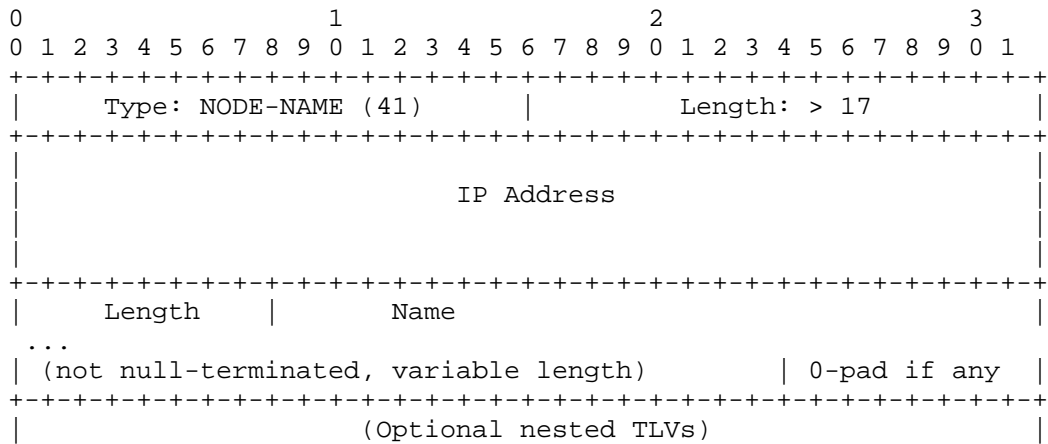
0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|  Type: DOMAIN-NAME (40)  |      Length: > 0      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Domain (DNS label sequence - variable length)      |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

This TLV is used to indicate the base domain name for the network as specified in Section 8. This TLV MUST NOT be announced unless the domain name was explicitly configured by an administrator.

Domain: The label sequence encoded according to [RFC1035]. Compression MUST NOT be used. The zone MUST end with an empty label.

#### 10.7. Node Name TLV



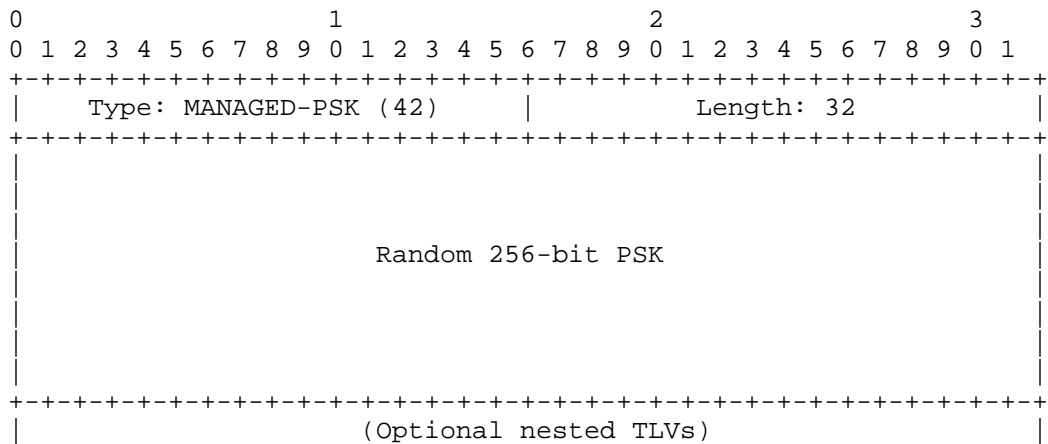
This TLV is used to assign the name of a node in the network to a certain IP address as specified in Section 8.

**IP Address:** The IP address associated with the name. IPv4 addresses are encoded using IPv4-mapped IPv6 addresses.

**Length:** The length of the name (0-63).

**Name:** The name of the node as a single DNS label.

#### 10.8. Managed PSK TLV



This TLV is used to announce a PSK for securing third-party protocols exclusively supporting symmetric cryptography as specified in Section 9.

## 11. General Requirements for HNCP Nodes

Each node implementing HNCP is subject to the following requirements:

- o It MUST implement HNCP-Versioning (Section 4) and Interface Classification (Section 5).
- o It MUST implement and run the method for securing third-party protocols (Section 9) whenever it uses the security mechanism of HNCP.

If the node is acting as a router, then the following requirements apply in addition:

- o It MUST support Autonomous Address Configuration (Section 6) and Configuration of Hosts and non-HNCP Routers (Section 7).
- o It SHOULD implement support for the Service Discovery and Naming (Section 8) as defined in this document.
- o It MAY be able to provide connectivity to IPv4-devices using DHCPv4.
- o It SHOULD be able to delegate prefixes to legacy IPv6 routers using DHCPv6-PD (Section 6.3.4).
- o In addition, normative language of Basic Requirements for IPv6 Customer Edge Routers [RFC7084] applies with the following adjustments:
  - \* The generic requirements G-4 and G-5 are relaxed such that any known default router on any interface is sufficient for a router to announce itself as default router, similarly only the loss of all such default routers results in self-invalidation.
  - \* The section "WAN-Side Configuration" applies to interfaces classified as external.
  - \* If the CE sends a size-hint as indicated in WPD-2, the hint MUST NOT be determined by the number of LAN-interfaces of the CE, but SHOULD instead be large enough to at least accommodate prefix assignments announced for existing delegated or ULA-prefixes, if such prefixes exist and unless explicitly configured otherwise.
  - \* The dropping of packets with a destination address belonging to a delegated prefix mandated in WPD-5 MUST NOT be applied to



destinations that are part of any prefix announced using an Assigned Prefix TLV by any HNCP router in the network.

- \* The section "LAN-Side Configuration" applies to interfaces not classified as external.
- \* The requirement L-2 to assign a separate /64 to each LAN interface is replaced by the participation in the prefix assignment mechanism (Section 6.3) for each such interface.
- \* The requirement L-9 is modified, in that the M flag MUST be set if and only if a router connected to the respective Common Link is advertising a non-zero H-capability. The O flag SHOULD always be set.
- \* The requirement L-12 to make DHCPv6 options available is adapted, in that a CER SHOULD publish the subset of options using the DHCPv6 Data TLV in an External Connection TLV. Similarly it SHOULD do the same for DHCPv4 options in a DHCPv4 Data TLV. DHCPv6 options received inside an OPTION\_IAPREFIX [RFC3633] MUST be published using a DHCPv6 Data TLV inside the respective Delegated Prefix TLV. HNCP routers SHOULD make relevant DHCPv6 and DHCPv4 options available to clients, i.e., options contained in External Connection TLVs that also include delegated prefixes from which a subset is assigned to the respective link.
- \* The requirement L-13 to deprecate prefixes is applied to all delegated prefixes in the network from which assignments have been made on the respective interface. Furthermore the Prefix Information Options indicating deprecation MUST be included in Router Advertisements for the remainder of the prefixes' respective valid lifetime, but MAY be omitted after at least 2 hours have passed.

## 12. Security Considerations

HNCP enables self-configuring networks, requiring as little user intervention as possible. However this zero-configuration goal usually conflicts with security goals and introduces a number of threats.

General security issues for existing home networks are discussed in [RFC7368]. The protocols used to set up addresses and routes in such networks to this day rarely have security enabled within the configuration protocol itself. However these issues are out of scope for the security of HNCP itself.

HNCP is a DNCP-based state synchronization mechanism carrying information with varying threat potential. For this consideration the payloads defined in DNCP and this document are reviewed:

- o Network topology information such as HNCP nodes and their common links.
- o Address assignment information such as delegated and assigned prefixes for individual links.
- o Naming and service discovery information such as auto-generated or customized names for individual links and nodes.

#### 12.1. Interface Classification

As described in Section 5.3, an HNCP node determines the internal or external state on a per-interface basis. A firewall perimeter is set up for the external interfaces, and for internal interfaces, HNCP traffic is allowed, with the exception of leaf and guest sub-categories.

Threats concerning automatic interface classification cannot be mitigated by encrypting or authenticating HNCP traffic itself since external routers do not participate in the protocol and often cannot be authenticated by other means. These threats include propagation of forged uplinks in the homenet in order to, e.g., redirect traffic destined to external locations and forged internal status by external routers to, e.g., circumvent the perimeter firewall.

It is therefore imperative to either secure individual links on the physical or link-layer or preconfigure the adjacent interfaces of HNCP routers to an appropriate fixed category in order to secure the homenet border. Depending on the security of the external link eavesdropping, man-in-the-middle and similar attacks on external traffic can still happen between a homenet border router and the ISP, however these cannot be mitigated from inside the homenet. For example, DHCPv4 has defined [RFC3118] to authenticate DHCPv4 messages, but this is very rarely implemented in large or small networks. Further, while PPP can provide secure authentication of both sides of a point to point link, it is most often deployed with one-way authentication of the subscriber to the ISP, not the ISP to the subscriber.

#### 12.2. Security of Unicast Traffic

Once the homenet border has been established there are several ways to secure HNCP against internal threats like manipulation or eavesdropping by compromised devices on a link which is enabled for

HNCP traffic. If left unsecured, attackers may perform arbitrary eavesdropping, spoofing or denial of service attacks on HNCP services such as address assignment or service discovery.

Detailed interface categories like "leaf" or "guest" can be used to integrate not fully trusted devices to various degrees into the homenet by not exposing them to HNCP traffic or by using firewall rules to prevent them from reaching homenet-internal resources.

On links where this is not practical and lower layers do not provide adequate protection from attackers, DNCP secure mode **MUST** be used to secure traffic.

### 12.3. Other Protocols in the Home

IGPs and other protocols are usually run alongside HNCP therefore the individual security aspects of the respective protocols must be considered. It can however be summarized that many protocols to be run in the home (like IGPs) provide - to a certain extent - similar security mechanisms. Most of these protocols do not support encryption and only support authentication based on pre-shared keys natively. This influences the effectiveness of any encryption-based security mechanism deployed by HNCP as homenet routing information is thus usually not encrypted.

### 13. IANA Considerations

IANA should set up a registry for the (decimal values within range 0-1023) "HNCP TLV Types" under "Distributed Node Consensus Protocol (DNCP)", with the following initial contents:

- 0-31: Reserved - specified in the DNCP registry
- 32: HNCP-Version
- 33: External-Connection
- 34: Delegated-Prefix
- 35: Assigned-Prefix
- 36: Node-Address
- 37: DHCPv4-Data
- 38: DHCPv6-Data
- 39: DNS-Delegated-Zone

40: Domain-Name

41: Node-Name

42: Managed-PSK

43: Prefix-Policy

44-512: Free - policy of 'RFC required' should be used.

512-767: Reserved - specified in the DNCP registry

768-1023: Reserved - to be used for per-implementation experimentation. How collision is avoided is out of scope of this document.

HNCP requires allocation of UDP port numbers HNCP-UDP-PORT and HNCP-DTLS-PORT, as well as an IPv6 link-local multicast address All-Homenet-Nodes.

## 14. References

### 14.1. Normative references

- [I-D.ietf-homenet-dncp]  
Stenberg, M. and S. Barth, "Distributed Node Consensus Protocol", draft-ietf-homenet-dncp-09 (work in progress), August 2015.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC6603] Korhonen, J., Savolainen, T., Krishnan, S., and O. Troan, "Prefix Exclude Option for DHCPv6-based Prefix Delegation", RFC 6603, May 2012.
- [I-D.ietf-homenet-prefix-assignment]  
Pfister, P., Paterson, B., and J. Arkko, "Distributed Prefix Assignment Algorithm", draft-ietf-homenet-prefix-assignment-08 (work in progress), August 2015.

## 14.2. Informative references

- [RFC3004] Stump, G., Droms, R., Gu, Y., Vyaghrapuri, R., Demirtjis, A., Beser, B., and J. Privat, "The User Class Option for DHCP", RFC 3004, November 2000.
- [RFC3118] Droms, R. and W. Arbaugh, "Authentication for DHCP Messages", RFC 3118, June 2001.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, 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.
- [RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.
- [RFC7368] Chown, T., Arkko, J., Brandt, A., Troan, O., and J. Weil, "IPv6 Home Networking Architecture Principles", RFC 7368, October 2014.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.
- [RFC6234] Eastlake, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, May 2011.
- [RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.
- [RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, February 2013.
- [RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, February 2013.

- [RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.
- [RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.
- [RFC7084] Singh, H., Beebe, W., Donley, C., and B. Stark, "Basic Requirements for IPv6 Customer Edge Routers", RFC 7084, November 2013.
- [RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, April 2014.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.
- [RFC6092] Woodyatt, J., Ed., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, DOI 10.17487/RFC6092, January 2011, <<http://www.rfc-editor.org/info/rfc6092>>.

#### 14.3. URIs

[3] <http://www.openwrt.org>

[4] <http://www.homewrt.org/doku.php?id=run-conf>

#### Appendix A. Changelog [RFC Editor: please remove]

draft-ietf-homenet-hncp-09: Added nested TLV definitions for variable length TLVs. NOTE: Node name TLV encoding includes now length byte. Version TLV now itself indicates version.

draft-ietf-homenet-hncp-08: Editorial reorganization.

draft-ietf-homenet-hncp-07: Using version 1 instead of version 0, as existing implementations already use it.

draft-ietf-homenet-hncp-06: Various edits based on feedback, hopefully without functional delta.

draft-ietf-homenet-hncp-05: Renamed "Adjacent Link" to "Common Link". Changed single IPv4 uplink election from MUST to MAY. Added explicit indication to distinguish (IPv4)-PDs for local connectivity and ones with uplink connectivity allowing, e.g., better local-only IPv4-connectivity.

draft-ietf-homenet-hncp-04: Change the responsibility for sending RAs to the router assigning the prefix.

draft-ietf-homenet-hncp-03: Split to DNCP (generic protocol) and HNCP (homenet profile).

draft-ietf-homenet-hncp-02: Removed any built-in security. Relying on IPsec. Reorganized interface categories, added requirements languages, made manual border configuration a MUST-support. Redesigned routing protocol election to consider non-router devices.

draft-ietf-homenet-hncp-01: Added (MAY) guest, ad-hoc, hybrid categories for interfaces. Removed old hnetv2 reference, and now pointing just to OpenWrt + github. Fixed synchronization algorithm to spread also same update number, but different data hash case. Made purge step require bidirectional connectivity between nodes when traversing the graph. Edited few other things to be hopefully slightly clearer without changing their meaning.

draft-ietf-homenet-hncp-00: Added version TLV to allow for TLV content changes pre-RFC without changing IDs. Added link id to assigned address TLV.

#### Appendix B. Draft source [RFC Editor: please remove]

This draft is available at <https://github.com/fingon/ietf-drafts/> in source format. Issues and pull requests are welcome.

#### Appendix C. Implementation [RFC Editor: please remove]

A GPLv2-licensed implementation of HNCP is currently under development at <https://github.com/sbyx/hnetd/> and binaries are available in the OpenWrt [3] package repositories. See [4] for more information. Feedback and contributions are welcome.

#### Appendix D. Acknowledgements

Thanks to Ole Troan, Mark Baugher, Mark Townsley, Juliusz Chroboczek and Thomas Clausen for their contributions to the draft.

Thanks to Eric Kline for the original border discovery work.

Authors' Addresses

Markus Stenberg  
Independent  
Helsinki 00930  
Finland

Email: markus.stenberg@iki.fi

Steven Barth  
Independent  
Halle 06114  
Germany

Email: cyrus@openwrt.org

Pierre Pfister  
Cisco Systems  
Paris  
France

Email: pierre.pfister@darou.fr



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

M. Stenberg  
Independent  
October 15, 2015

Auto-Configuration of a Network of Hybrid Unicast/Multicast DNS-Based  
Service Discovery Proxy Nodes  
draft-ietf-homenet-hybrid-proxy-zeroconf-02

Abstract

This document describes how a proxy functioning between Unicast DNS-Based Service Discovery and Multicast DNS can be automatically configured using an arbitrary network-level state sharing 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 April 17, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Requirements language . . . . .	3
3. Hybrid proxy - what to configure . . . . .	3
3.1. Conflict resolution within network . . . . .	4
3.2. Per-link DNS-SD forward zone names . . . . .	4
3.3. Reasonable defaults . . . . .	5
3.3.1. Network-wide unique link name (scheme 1) . . . . .	5
3.3.2. Node name (scheme 2) . . . . .	5
3.3.3. Link name (scheme 2) . . . . .	5
4. TLVs . . . . .	5
4.1. DNS Delegated Zone TLV . . . . .	6
4.2. Domain Name TLV . . . . .	7
4.3. Node Name TLV . . . . .	7
5. Desirable behavior . . . . .	7
5.1. DNS search path in DHCP requests . . . . .	8
5.2. Hybrid proxy . . . . .	8
5.3. Hybrid proxy network zeroconf daemon . . . . .	8
6. Limited zone stitching for host name resolution . . . . .	8
7. Security Considerations . . . . .	9
8. IANA Considerations . . . . .	9
9. References . . . . .	9
9.1. Normative references . . . . .	9
9.2. Informative references . . . . .	10
Appendix A. Example configuration . . . . .	10
A.1. Used topology . . . . .	10
A.2. Zero-configuration steps . . . . .	11
A.3. TLV state . . . . .	11
A.4. DNS zone . . . . .	12
A.5. Interaction with hosts . . . . .	13
Appendix B. Implementation . . . . .	13
Appendix C. Why not just proxy Multicast DNS? . . . . .	13
C.1. General problems . . . . .	14
C.2. Stateless proxying problems . . . . .	14
C.3. Stateful proxying problems . . . . .	15
Appendix D. Acknowledgements . . . . .	15
Appendix E. Changelog [RFC Editor: please remove] . . . . .	15
Author's Address . . . . .	16

## 1. Introduction

Section 3 ("Hybrid Proxy Operation") of [I-D.ietf-dnssd-hybrid] describes how to translate queries from Unicast DNS-Based Service Discovery described in [RFC6763] to Multicast DNS described in [RFC6762], and how to filter the responses and translate them back to unicast DNS.

This document describes what sort of configuration the participating hybrid proxy servers require, as well as how it can be provided using any network-wide state sharing mechanism such as link-state routing protocol or Home Networking Control Protocol [I-D.ietf-homenet-hncp]. The document also describes a naming scheme which does not even need to be same across the whole covered network to work as long as the specified conflict resolution works. The scheme can be used to provision both forward and reverse DNS zones which employ hybrid proxy for heavy lifting.

This document does not go into low level encoding details of the Type-Length-Value (TLV) data that we want synchronized across a network. Instead, we just specify what needs to be available, and assume every node that needs it has it available.

We go through the mandatory specification of the language used in Section 2, then describe what needs to be configured in hybrid proxies and participating DNS servers across the network in Section 3. How the data is exchanged using arbitrary TLVs is described in Section 4. Finally, some overall notes on desired behavior of different software components is mentioned in Section 5.

## 2. 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 [RFC2119].

## 3. Hybrid proxy - what to configure

Beyond the low-level translation mechanism between unicast and multicast service discovery, the hybrid proxy draft [I-D.ietf-dnssd-hybrid] describes just that there have to be NS records pointing to hybrid proxy responsible for each link within the covered network.

In zero-configuration case, choosing the links to be covered is also non-trivial choice; we can use the border discovery functionality (if available) to determine internal and external links. Or we can use some other protocol's presence (or lack of it) on a link to determine internal links within the covered network, and some other signs (depending on the deployment) such as DHCPv6 Prefix Delegation (as described in [RFC3633]) to determine external links that should not be covered.

For each covered link we want forward DNS zone delegation to an appropriate node which is connected to a link, and running hybrid proxy. Therefore the links' forward DNS zone names should be unique

across the network. We also want to populate reverse DNS zone similarly for each IPv4 or IPv6 prefix in use.

There should be DNS-SD browse domain list provided for the network's domain which contains each physical link only once, regardless of how many nodes and hybrid proxy implementations are connected to it.

Yet another case to consider is the list of DNS-SD domains that we want hosts to enumerate for browse domain lists. Typically, it contains only the local network's domain, but there may be also other networks we may want to pretend to be local but are in different scope, or controlled by different organization. For example, a home user might see both home domain's services (TBD-TLD), as well as ISP's services under `isp.example.com`.

### 3.1. Conflict resolution within network

Any naming-related choice on node may have conflicts in the network given that we require only distributed loosely synchronized database. We assume only that the underlying protocol used for synchronization has some concept of precedence between nodes originating conflicting information, and in case of conflict, the higher precedence node MUST keep the name they have chosen. The one(s) with lower precedence MUST either try different one (that is not in use at all according to the current link state information), or choose not to publish the name altogether.

If a node needs to pick a different name, any algorithm works, although simple algorithm choice is just like the one described in Multicast DNS[RFC6762]: append -2, -3, and so forth, until there are no conflicts in the network for the given name.

### 3.2. Per-link DNS-SD forward zone names

How to name the links of a whole network in automated fashion? Two different approaches seem obvious:

1. Unique link name based - `(unique-link).(domain)`.
2. Node and link name - `(link).(unique-node).(domain)`.

The first choice is appealing as it can be much more friendly (especially given manual configuration). For example, it could mean just `lan.example.com` and `wlan.example.com` for a simple home network. The second choice, on the other hand, has a nice property of being local choice as long as node name can be made unique.

The type of naming scheme to use can be left as implementation option. And the actual names themselves SHOULD be also overridable, if the end-user wants to customize them in some way.

### 3.3. Reasonable defaults

Note that any manual configuration, which SHOULD be possible, MUST override the defaults provided here or chosen by the creator of the implementation.

#### 3.3.1. Network-wide unique link name (scheme 1)

It is not obvious how to produce network-wide unique link names for the (unique-link).(domain) scheme. One option would be to base it on type of physical network layer, and then hope that the number of the networks won't be significant enough to confuse (e.g. "lan", or "wlan").

The network-wide unique link names should be only used in small networks. Given a larger network, after conflict resolution, identifying which link is 'lan-42.example.com' may be challenging.

#### 3.3.2. Node name (scheme 2)

Our recommendation is to use some short form which indicates the type of node it is, for example, "openwrt.example.com". As the name is visible to users, it should be kept as short as possible. In theory even more exact model could be helpful, for example, "openwrt-buffalo-wzr-600-dhr.example.com". In practice providing some other records indicating exact node information (and access to management UI) is more sensible.

#### 3.3.3. Link name (scheme 2)

Recommendation for (link) portion of (link).(node).(domain) is to use physical network layer type as base, or possibly even just interface name on the node if it's descriptive enough. For example, "eth0.openwrt.example.com" and "wlan0.openwrt.example.com" may be good enough.

## 4. TLVs

To implement this specification fully, support for following three different TLVs is needed. However, only the DNS Delegated Zone TLVs MUST be supported, and the other two SHOULD be supported.

#### 4.1. DNS Delegated Zone TLV

This TLV is effectively a combined NS and A/AAAA record for a zone. It MUST be supported by implementations conforming to this specification. Implementations SHOULD provide forward zone per link (or optimizing a bit, zone per link with Multicast DNS traffic). Implementations MAY provide reverse zone per prefix using this same mechanism. If multiple nodes advertise same reverse zone, it should be assumed that they all have access to the link with that prefix. However, as noted in Section 5.3, mainly only the node with highest precedence on the link should publish this TLV.

##### Contents:

- o Address field is IPv6 address (e.g. 2001:db8::3) or IPv4 address mapped to IPv6 address (e.g. ::FFFF:192.0.2.1) where the authoritative DNS server for Zone can be found. If the address field is all zeros, the Zone is under global DNS hierarchy and can be found using normal recursive name lookup starting at the authoritative root servers (This is mostly relevant with the S bit below).
- o S-bit indicates that this delegated zone consists of a full DNS-SD domain, which should be used as base for DNS-SD domain enumeration (that is, (field).\_dns-sd.\_udp.(zone) exists). Forward zones MAY have this set. Reverse zones MUST NOT have this set. This can be used to provision DNS search path to hosts for non-local services (such as those provided by ISP, or other manually configured service providers).
- o B-bit indicates that this delegated zone should be included in network's DNS-SD browse list of domains at b.\_dns-sd.\_udp.(domain). Local forward zones SHOULD have this set. Reverse zones SHOULD NOT have this set.
- o L-bit indicates that this delegated zone should be included in the network's DNS-SD legacy browse list of domains at lb.\_dns-sd.\_udp.(DOMAIN-NAME). Local forward zones SHOULD have this bit set, reverse zones SHOULD NOT.
- o Zone is the label sequence of the zone, encoded according to section 3.1. ("Name space definitions") of [RFC1035]. Note that name compression is not required here (and would not have any point in any case), as we encode the zones one by one. The zone MUST end with an empty label.

In case of a conflict (same zone being advertised by multiple parties with different address or bits), conflict should be addressed according to Section 3.1.

#### 4.2. Domain Name TLV

This TLV is used to indicate the base (domain) to be used for the network. If multiple nodes advertise different ones, the conflict resolution rules in Section 3.1 should result in only the one with highest precedence advertising one, eventually. In case of such conflict, user SHOULD be notified somehow about this, if possible, using the configuration interface or some other notification mechanism for the nodes. Like the Zone field in Section 4.1, the Domain Name TLV's contents consist of a single DNS label sequence.

This TLV SHOULD be supported if at all possible. It may be derived using some future DHCPv6 option, or be set by manual configuration. Even on nodes without manual configuration options, being able to read the domain name provided by a different node could make the user experience better due to consistent naming of zones across the network.

By default, if no node advertises domain name TLV, hard-coded default (TBD) should be used.

#### 4.3. Node Name TLV

This TLV is used to advertise a node's name. After the conflict resolution procedure described in Section 3.1 finishes, there should be exactly zero to one nodes publishing each node name. The contents of the TLV should be a single DNS label.

This TLV SHOULD be supported if at all possible. If not supported, and another node chooses to use the (link).(node) naming scheme with this node's name, the contents of the network's domain may look misleading (but due to conflict resolution of per-link zones, still functional).

If the node name has been configured manually, and there is a conflict, user SHOULD be notified somehow about this, if possible, using the configuration interface or some other notification mechanism for the nodes.

#### 5. Desirable behavior

### 5.1. DNS search path in DHCP requests

The nodes following this specification SHOULD provide the used (domain) as one item in the search path to it's hosts, so that DNS-SD browsing will work correctly. They also SHOULD include any DNS Delegated Zone TLVs' zones, that have S bit set.

### 5.2. Hybrid proxy

The hybrid proxy implementation SHOULD support both forward zones, and IPv4 and IPv6 reverse zones. It SHOULD also detect whether or not there are any Multicast DNS entities on a link, and make that information available to the network zeroconf daemon (if implemented separately). This can be done by (for example) passively monitoring traffic on all covered links, and doing infrequent service enumerations on links that seem to be up, but without any Multicast DNS traffic (if so desired).

Hybrid proxy nodes MAY also publish it's own name via Multicast DNS (both forward A/AAAA records, as well as reverse PTR records) to facilitate applications that trace network topology.

### 5.3. Hybrid proxy network zeroconf daemon

The daemon should avoid publishing TLVs about links that have no Multicast DNS traffic to keep the DNS-SD browse domain list as concise as possible. It also SHOULD NOT publish delegated zones for links for which zones already exist by another node with higher precedence.

The daemon (or other entity with access to the TLVs) SHOULD generate zone information for DNS implementation that will be used to serve the (domain) zone to hosts. Domain Name TLV described in Section 4.2 should be used as base for the zone, and then all DNS Delegated Zones described in Section 4.1 should be used to produce the rest of the entries in zone (see Appendix A.4 for example interpretation of the TLVs in Appendix A.3).

## 6. Limited zone stitching for host name resolution

Section 4.1 of the hybrid proxy specification [I-D.ietf-dnssd-hybrid] notes that the stitching of multiple .local zones into a single DNS-SD zone is to be defined later. This specification does not even attempt that, but for the purpose of host name resolution, it is possible to use the set of DNS Delegated Zone TLVs with S-bit or B-bit set to also provide host naming for the (domain). It is done by simply rewriting A/AAAA queries for (name).(domain) to every (name).(ddz-subdomain).(domain), and providing response to the host



when the first non-empty one is received, rewritten back to (name).(domain).

While this scheme is not very scalable, as it multiplies the number of queries by the number of links (given no response in cache), it does work in small networks with relatively few sub-domains.

## 7. Security Considerations

There is a trade-off between security and zero-configuration in general; if used network state synchronization protocol is not authenticated (and in zero-configuration case, it most likely is not), it is vulnerable to local spoofing attacks. We assume that this scheme is used either within (lower layer) secured networks, or with not-quite-zero-configuration initial set-up.

If some sort of dynamic inclusion of links to be covered using border discovery or such is used, then effectively service discovery will share fate with border discovery (and also security issues if any).

## 8. IANA Considerations

This document has no actions for IANA.

## 9. References

### 9.1. Normative references

- [I-D.ietf-dnssd-hybrid] Cheshire, S., "Hybrid Unicast/Multicast DNS-Based Service Discovery", draft-ietf-dnssd-hybrid-00 (work in progress), November 2014.
- [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>>.
- [RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <<http://www.rfc-editor.org/info/rfc6762>>.

[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>>.

## 9.2. Informative references

[I-D.ietf-homenet-hncp] Stenberg, M., Barth, S., and P. Pfister, "Home Networking Control Protocol", draft-ietf-homenet-hncp-09 (work in progress), August 2015.

[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>>.

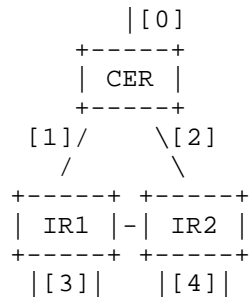
## 9.3. URIs

[1] <https://github.com/sbyx/hnetd/>

## Appendix A. Example configuration

### A.1. Used topology

Let's assume home network that looks like this:



We're not really interested about links [0], [1] and [2], or the links between IRs. Given the optimization described in Section 4.1, they should not produce anything to network's Multicast DNS state (and therefore to DNS either) as there isn't any Multicast DNS traffic there.

The user-visible set of links are [3] and [4]; each consisting of a LAN and WLAN link. We assume that ISP provides 2001:db8:1234::/48 prefix to be delegated in the home via [0].

#### A.2. Zero-configuration steps

Given implementation that chooses to use the second naming scheme (link).(node).(domain), and no configuration whatsoever, here's what happens (the steps are interleaved in practice but illustrated here in order):

1. Network-level state synchronization protocol runs, nodes get effective precedences. For ease of illustration, CER winds up with 2, IR1 with 3, and IR2 with 1.
2. Prefix delegation takes place. IR1 winds up with 2001:db8:1234:11::/64 for LAN and 2001:db8:1234:12::/64 for WLAN. IR2 winds up with 2001:db8:1234:21::/64 for LAN and 2001:db8:1234:22::/64 for WLAN.
3. IR1 is assumed to be reachable at 2001:db8:1234:11::1 and IR2 at 2001:db8:1234:21::1.
4. Each node wants to be called 'node' due to lack of branding in drafts. They announce that using the node name TLV defined in Section 4.3. They also advertise their local zones, but as that information may change, it's omitted here.
5. Conflict resolution ensues. As IR1 has precedence over the rest, it becomes "node". CER and IR2 have to rename, and (depending on timing) one of them becomes "node-2" and other one "node-3". Let us assume IR2 is "node-2". During conflict resolution, each node publishes TLVs for it's own set of delegated zones.
6. CER learns ISP-provided domain "isp.example.com" using DHCPv6 domain list option defined in [RFC3646]. The information is passed along as S-bit enabled delegated zone TLV.

#### A.3. TLV state

Once there is no longer any conflict in the system, we wind up with following TLVs (NN is used as abbreviation for Node Name, and DZ for Delegated Zone TLVs):

```
(from CER)
DZ {s=1,zone="isp.example.com"}

(from IR1)
NN {name="node"}

DZ {address=2001:db8:1234:11::1, b=1,
    zone="lan.node.example.com."}
DZ {address=2001:db8:1234:11::1,
    zone="1.1.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa."}

DZ {address=2001:db8:1234:11::1, b=1,
    zone="wlan.node.example.com."}
DZ {address=2001:db8:1234:11::1,
    zone="2.1.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa."}

(from IR2)
NN {name="node-2"}

DZ {address=2001:db8:1234:21::1, b=1,
    zone="lan.node-2.example.com."}
DZ {address=2001:db8:1234:21::1,
    zone="1.2.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa."}

DZ {address=2001:db8:1234:21::1, b=1,
    zone="wlan.node-2.example.com."}
DZ {address=2001:db8:1234:21::1,
    zone="2.2.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa."}
```

#### A.4. DNS zone

In the end, we should wind up with following zone for (domain) which is example.com in this case, available at all nodes, just based on dumping the delegated zone TLVs as NS+AAAA records, and optionally domain list browse entry for DNS-SD:

```
b._dns_sd._udp PTR lan.node
b._dns_sd._udp PTR wlan.node

b._dns_sd._udp PTR lan.node-2
b._dns_sd._udp PTR wlan.node-2

node AAAA 2001:db8:1234:11::1
node-2 AAAA 2001:db8:1234:21::1

node NS node
node-2 NS node-2
```

```
1.1.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa. NS node.example.com.
2.1.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa. NS node.example.com.
1.2.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa. NS node-2.example.com.
2.2.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa. NS node-2.example.com.
```

Internally, the node may interpret the TLVs as it chooses to, as long as externally defined behavior follows semantics of what's given in the above.

#### A.5. Interaction with hosts

So, what do the hosts receive from the nodes? Using e.g. DHCPv6 DNS options defined in [RFC3646], DNS server address should be one (or multiple) that point at DNS server that has the zone information described in Appendix A.4. Domain list provided to hosts should contain both "example.com" (the hybrid-enabled domain), as well as the externally learned domain "isp.example.com".

When hosts start using DNS-SD, they should check both b.\_dns-sd.\_udp.example.com, as well as b.\_dns-sd.\_udp.isp.example.com for list of concrete domains to browse, and as a result services from two different domains will seem to be available.

#### Appendix B. Implementation

There is an prototype implementation of this draft at [hnetd github repository](#) [1] which contains variety of other homenet WG-related things' implementation too.

#### Appendix C. Why not just proxy Multicast DNS?

Over the time number of people have asked me about how, why, and if we should proxy (originally) link-local Multicast DNS over multiple links.

At some point I meant to write a draft about this, but I think I'm too lazy; so some notes left here for general amusement of people (and to be removed if this ever moves beyond discussion piece).

### C.1. General problems

There are two main reasons why Multicast DNS is not proxyable in the general case.

First reason is the conflict resolution depends on the RRsets staying constant. That is not possible across multiple links (due to e.g. link-local addresses having to be filtered). Therefore, conflict resolution breaks, or at least requires ugly hacks to work around.

A simple, but not really working workaround for this is to make sure that in conflict resolution, propagated resources always loses. Given that the proxy function only removes records, the result SHOULD be consistently original set of records winning. Even with that, the conflict resolution will effectively cease working, allowing for two instances of same name to exist (as both think they 'own' the name due to locally seen higher precedence).

Given some more extra logic, it is possible to make this work by having proxies be aware of both the original record sets, and effectively enforcing the correct conflict resolution results by (for example) passing the unfiltered packets to the losing party just to make sure they renumber, or by altering the RR sets so that they will consistently win (by inserting some lower rrclass/rrtype records). As the conflicts happen only in rrclass=1/rrtype=28, it is easy enough to add e.g. extra TXT record (rrtype 16) to force precedence even when removing the later rrtype 28 record. Obviously, this new RRset must never wind up near the host with the higher precedence, or it will cause spurious renaming loops.

Second reason is timing, which is relatively tight in the conflict resolution phase, especially given lossy and/or high latency networks.

### C.2. Stateless proxying problems

In general, typical stateless proxy has to involve flooding, as Multicast DNS assumes that most messages are received by every host. And it won't scale very well, as a result.

The conflict resolution is also harder without state. It may result in Multicast DNS responder being in constant probe-announce loop, when it receives altered records, notes that it's the one that should own the record. Given stateful proxying, this would be just a

transient problem but designing stateless proxy that won't cause this is non-trivial exercise.

### C.3. Stateful proxying problems

One option is to write proxy that learns state from one link, and propagates it in some way to other links in the network.

A big problem with this case lies in the fact that due to conflict resolution concerns above, it is easy to accidentally send packets that will (possibly due to host mobility) wind up at the originator of the service, who will then perform renaming. That can be alleviated, though, given clever hacks with conflict resolution order.

The stateful proxying may be also too slow to occur within the timeframe allocated for announcing, leading to excessive later renamings based on delayed finding of duplicate services with same name

A work-around exists for this though; if the game doesn't work for you, don't play it. One option would be simply not to propagate ANY records for which conflict has seen even once. This would work, but result in rather fragile, lossy service discovery infrastructure.

There are some other small nits too; for example, Passive Observation Of Failure (POOF) will not work given stateful proxying. Therefore, it leads to requiring somewhat shorter TTLs, perhaps.

### Appendix D. Acknowledgements

Thanks to Stuart Cheshire for the original hybrid proxy draft and interesting discussion in Orlando, where I was finally convinced that stateful Multicast DNS proxying is a bad idea.

Also thanks to Mark Baugher, Ole Troan, Shwetha Bhandari and Gert Doering for review comments.

### Appendix E. Changelog [RFC Editor: please remove]

draft-ietf-homenet-hybrid-proxy-zeroconf-02:

- o Added subsection on simple zone stitching for host naming purposes.

draft-ietf-homenet-hybrid-proxy-zeroconf-01:

- o Refreshed the draft while waiting on progress of draft-ietf-dnssd-hybrid.

Author's Address

Markus Stenberg  
Independent  
Helsinki 00930  
Finland

Email: markus.stenberg@iki.fi



HOMENET  
Internet-Draft  
Intended status: Standards Track  
Expires: November 20, 2015

D. Migault (Ed)  
Ericsson  
W. Cloetens  
SoftAtHome  
C. Griffiths  
Dyn  
R. Weber  
Nominum  
May 19, 2015

DHCP Options for Homenet Naming Architecture  
draft-ietf-homenet-naming-architecture-dhc-options-02.txt

Abstract

CPEs are usually constraint devices with reduced network and CPU capacities. As such, a CPE hosting on the Internet the authoritative naming service for its home network may become vulnerable to resource exhaustion attacks. One way to avoid exposing CPE is to outsource the authoritative service to a third party. This third party can be the ISP or any other independent third party.

Outsourcing the authoritative naming service to a third party requires setting up an architecture which may be unappropriated for most end users. To leverage this issue, this document proposes DHCP Options so any agnostic CPE can automatically proceed to the appropriated configuration and outsource the authoritative naming service for the home network. This document shows that in most cases, these DHCP Options make outsourcing to a third party (be it the ISP or any ISP independent service provider) transparent for the end user.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 20, 2015.

#### Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

#### Table of Contents

1. Requirements notation . . . . .	3
2. Terminology . . . . .	3
3. Introduction . . . . .	4
4. Protocol Overview . . . . .	6
4.1. Architecture and DHCP Options Overview . . . . .	7
4.2. Mechanisms Securing DNS Transactions . . . . .	10
4.3. Primary / Secondary Synchronization versus DNS Update . .	11
5. CPE Configuration . . . . .	11
5.1. CPE Primary / Secondary Synchronization Configurations .	11
5.1.1. CPE / Public Authoritative Name Server Set . . . . .	12
5.1.2. CPE / Reverse Public Authoritative Name Server Set .	12
5.2. CPE DNS Data Handling and Update Policies . . . . .	12
5.2.1. DNS Homenet Zone Template . . . . .	12
5.2.2. DNS (Reverse) Homenet Zone . . . . .	13
6. Payload Description . . . . .	13
6.1. Security Field . . . . .	14
6.2. Update Field . . . . .	14
6.3. DHCP Public Key Option . . . . .	15
6.4. DHCP Zone Template Option . . . . .	16
6.5. DHCP Public Authoritative Name Server Set Option . . . .	16
6.6. DHCP Reverse Public Authoritative Name Server Set Option	17
7. DHCP Behavior . . . . .	18
7.1. DHCPv6 Server Behavior . . . . .	18
7.2. DHCPv6 Client Behavior . . . . .	19
7.3. DHCPv6 Relay Behavior . . . . .	19
8. IANA Considerations . . . . .	19
9. Security Considerations . . . . .	19
9.1. DNSSEC is recommended to authenticate DNS hosted data . .	19
9.2. Channel between the CPE and ISP DHCP Server MUST be	

secured . . . . .	19
9.3. CPEs are sensitive to DoS . . . . .	20
10. Acknowledgment . . . . .	20
11. References . . . . .	20
11.1. Normative References . . . . .	20
11.2. Informational References . . . . .	22
Appendix A. Scenarios and impact on the End User . . . . .	22
A.1. Base Scenario . . . . .	22
A.2. Third Party Registered Homenet Domain . . . . .	23
A.3. Third Party DNS Infrastructure . . . . .	23
A.4. Multiple ISPs . . . . .	25
Appendix B. Document Change Log . . . . .	26
Authors' Addresses . . . . .	27

## 1. Requirements notation

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].

## 2. Terminology

- Customer Premises Equipment: (CPE) is the router providing connectivity to the home network. It is configured and managed by the end user. In this document, the CPE might also hosts services such as DHCPv6. This device might be provided by the ISP.
- Public Key: designates a public Key generated by the CPE. This key is used as an authentication credential for the CPE.
- Registered Homenet Domain: is the Domain Name associated to the home network.
- DNS Homenet Zone: is the DNS zone associated to the home network. This zone is set by the CPE and essentially contains the bindings between names and IP addresses of the nodes of the home network. In this document, the CPE does neither perform any DNSSEC management operations such as zone signing nor provide an authoritative service for the zone. Both are delegated to the Public Authoritative Server. The CPE synchronizes the DNS Homenet Zone with the Public Authoritative Server via a hidden primary / secondary architecture. The Public Authoritative Server might use specific servers for the synchronization of the DNS Homenet Zone: the Public Authoritative Name Server Set.

- DNS Homenet Zone Template: The template used as a basis to generate the DNS Homenet Zone.
- DNS Template Server: The DNS server that hosts the DNS Homenet Zone Template.
- DNS Homenet Reverse Zone: The reverse zone file associated to the DNS Homenet Zone.
- Public Authoritative Primary(ies): are the visible name server hosting the DNS Homenet Zone. End users' resolutions for the Homenet Domain are sent to this server, and this server is a primary for the zone.
- Public Authoritative Name Server Set: is the server the CPE synchronizes the DNS Homenet Zone. It is configured as a secondary and the CPE acts as primary. The CPE sends information so the DNSSEC zone can be set and served.
- Reverse Public Authoritative Primary(ies): are the visible name server hosting the DNS Homenet Reverse Zone. End users' resolutions for the Homenet Domain are sent to this server, and this server is a primary for the zone.
- Reverse Public Authoritative Name Server Set: is the server the CPE synchronizes the DNS Homenet Reverse Zone. It is configured as a secondary and the CPE acts as primary. The CPE sends information so the DNSSEC zone can be set and served.

### 3. Introduction

CPEs are usually constraint devices with reduced network and CPU capacities. As such, a CPE hosting on the Internet the authoritative naming service for its home network may become vulnerable to resource exhaustion attacks. One way to avoid exposing CPE is to outsource the authoritative service to a third party. This third party can be the ISP or any other independent third party.

Outsourcing the authoritative naming service to a third party requires setting up an architecture which may be unappropriated for most end users. To leverage this issue, this document proposes DHCP Options so any agnostic CPE can automatically proceed to the appropriated configuration and outsource the authoritative naming service for the home network. This document shows that in most cases, these DHCP Options make outsourcing to a third party (be it the ISP or any ISP independent service provider) transparent for the end user.

When the CPE is plugged, the DHCP Options described in the document enable the CPE:

- 1. To build the DNS Homenet Zone: Building the DNS Homenet Zone requires filling the zone with appropriated bindings likes name / IP addresses of the different devices in the home networks. Such information can be provided for example by the DHCP Server hosted on the CPE. On the other hand, it also requires configuration parameters like the name of the Registered Domain Name associated to the home network or the Public Authoritative Primary(ies) the DNS Homenet Zone is outsourced to. These configuration parameters are stored in the DNS Homenet Zone Template. This document describes the DHCP Zone Template Option. This option carries a DNS Homenet Zone Template FQDN. In order to retrieve the DNS Homenet Zone Template, the CPE sends a query of type AXFR [RFC1034] [RFC5936]for the DNS Homenet Zone Template FQDN.
- 2. To upload the DNS(SEC) Homenet Zone to the appropriated server: This server is designated as the Public Authoritative Name Server Set. It is in charge of publishing the DNS(SEC) Homenet Zone on the Public Authoritative Primary(ies). This document describes the DHCP Public Authoritative Name Server Set Option that provides the FQDN of the appropriated server. Note that, in the document we do not consider whether the DNS(SEC) Homenet Zone is signed or not and if signed who signs it. Such questions are out of the scope of the current document.
- 3. To upload the DNS Homenet Reverse Zone to the appropriated server: This server is designated as the Reverse Public Authoritative Name Server Set. It is in charge of publishing the DNS Homenet Reverse Zone on the Reverse Public Authoritative Primary(ies). This document describes the DHCP Reverse Public Authoritative Name Server Set Option that provides the FQDN of the appropriated server. Similarly to item 2., we do not consider in this document if the DNS Homenet Reverse Zone is signed or not, and if signed who signs it.
- 4. To provide authentication credential (a public key) to the DHCP Server: Information stored in the DNS Homenet Zone Template, the DNS(SEC) Homenet Zone and DNS Homenet Reverse Zone belongs to the CPE, and only the CPE should be able to update or upload these zones. To authenticate the CPE, this document defines the DHCP Public Key Option. This option is sent by the CPE to the DHCP Server and provides the Public Key the CPE uses to authenticate itself. The DHCP Server is then responsible to provide the Public Key to the various DNS servers.

As a result, the DHCP Options described in this document enable an agnostic CPE to outsource its naming infrastructure without any configuration from the end user. The main reason no configuration is required by the end user is that there are privileged links: first between the CPE and the DHCP Server and then between the DHCP Server and the various DNS servers (DNS Homenet Zone Server, the Reverse Public Authoritative Name Server Set, Public Authoritative Name Server Set). This enables the CPE to send its authentication credentials (a Public Key) to the DHCP Server that in turn forward it to the various DNS servers. With the authentication credential on the DNS servers set, the CPE is able to update the various zones in a secure way.

If the DHCP Server cannot provide the public key to one of these servers (most likely the Public Authoritative Name Server Set) and the CPE needs to interact with the server, then, the end user is expected to provide the CPE's public key to these servers (the Reverse Public Authoritative Name Server Set or the Public Authoritative Name Server Set) either manually or using other mechanisms. Such mechanisms are outside the scope of this document. In that case, the authentication credentials need to be provided every time the key is modified. Appendix A provides more details on how different scenarios impact the end users.

The remaining of this document is as follows. Section 4 provides an overview of the DHCP Options as well as the expected interactions between the CPE and the various involved entities. This section also provides an overview of available mechanisms to secure DNS transactions and update DNS Data. Section 5 describes how the CPE may securely synchronize and update DNS data. Section 6 describes the payload of the DHCP Options and Section 7 details how DHCP Client DHCP Server and DHCP Relay behave. Section 8 lists the new parameters to be registered at the IANA, Section 9 provides security considerations. Finally, Appendix A describes how the CPE may behave and be configured regarding various scenarios.

#### 4. Protocol Overview

This section provides an overview of the how the CPE is expect to interact with various entities, as well as how the CPE is expected to be configured via DHCP Options. Section 4.1 describes the entities the CPE is expected to interact with. Interaction with each entities is defined via DHCP Options that are expected to configure the CPE. Once configured, the CPE is expected to be able to update some DNS Data hosted by the different entities. As a result security and updating mechanisms play an important role in the specification. Section 4.2 provides an overview of the different security mechanisms considered for securing the CPE transactions and Section 4.3

considers the different update mechanisms considered for the CPE to update the DNS Data.

#### 4.1. Architecture and DHCP Options Overview

This section illustrates how a CPE configures its naming infrastructure to outsource its authoritative naming service. All configurations and settings are performed using DHCP Options. This section, for the sake of simplicity, assumes that the DHCP Server is able to communicate to the various DNS servers and to provide them the public key associated to the CPE. Once each server got the public key, the CPE can proceed to updates in a authenticated and secure way.

This scenario has been chosen as it is believed to be the most popular scenario. This document does not ignore that scenarios where the DHCP Server does not have privileged relations with the Public Authoritative Name Server Set must be considered. These cases are discussed latter in Appendix A. Such scenario does not necessarily require configuration for the end user and can also be Zero Config.

The scenario is represented in Figure 1.

- 1: The CPE provides its Public Key to the DHCP Server using a DHCP Public Key Option (OPTION\_PUBLIC\_KEY) and sends a DHCP Option Request Option (ORO) for the DHCP Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE), the DHCP Public Authoritative Name Server Set Option (OPTION\_NAME\_SERVER\_SET) and the DHCP Reverse Public Authoritative Name Server Set Option (OPTION\_REVERSE\_NAME\_SERVER\_SET).
- 2: The DHCP Server makes the Public Key available to the DNS servers, so the CPE can secure its DNS transactions. Note that the Public Key alone is not sufficient to perform the authentication and the key should be, for example, associated with an identifier, or the concerned domain name. How the binding is performed is out of scope of the document. It can be a centralized database or various bindings may be sent to the different servers. Figure 1 represents the specific case where the DHCP Server forwards the set (Public Key, Zone Template FQDN) to the DNS Template Server, the set (Public Key, IPv6 subnet) to the Reverse Public Authoritative Name Server Set and the set (Public Key, Registered Homenet Domain) to the Public Authoritative Name Server Set.
- 3.: The DHCP Server responds to the CPE with the requested DHCP Options, i.e. the DHCP Public Key Option (OPTION\_PUBLIC\_KEY), DHCP Zone Template Option OPTION\_DNS\_ZONE\_TEMPLATE, DHCP Public

Authoritative Name Server Set Option (OPTION\_NAME\_SERVER\_SET),  
DHCP Reverse Public Authoritative Name Server Set Option  
(OPTION\_REVERSE\_NAME\_SERVER\_SET).

- 4.: Upon receiving the DHCP Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE), the CPE performs an AXFR DNS query for the Zone Template FQDN. The exchange is secured according to the security protocols defined in the Security field of the DHCP option. Once the CPE has retrieved the DNS Zone Template, the CPE can build the DNS Homenet Zone and the DNS Homenet Reverse Zone. Eventually the CPE signs these zones.
- 5.: Once the DNS(SEC) Homenet Reverse Zone has been set, the CPE uploads the zone to the Reverse Public Authoritative Name Server Set. The DHCP Reverse Public Authoritative Name Server Set Option (OPTION\_REVERSE\_NAME\_SERVER\_SET) provides the Reverse Public Authoritative Name Server Set FQDN as well as the upload method, and the security protocol to secure the upload.
- 6.: Once the DNS(SEC) Homenet Zone has been set, the CPE uploads the zone to the Public Authoritative Name Server Set. The DHCP Public Authoritative Name Server Set Option (OPTION\_NAME\_SERVER\_SET) provides the Public Authoritative Name Server Set FQDN as well as the upload method and the security protocol to secure the upload.



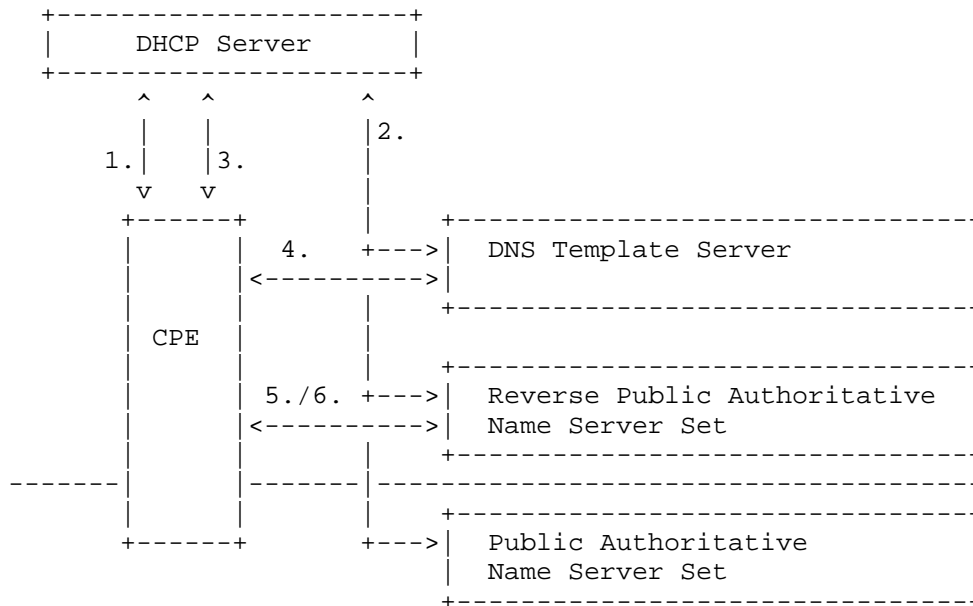


Figure 1: Protocol Overview

As described above, the CPE is likely to interact with various DNS content. This section is focused on DNS Data the CPE is likely to update. More specifically, the CPE is likely to update the:

- DNS Homenet Zone Template: may be updated by the CPE if the configuration of the zone may be changed. This can include additional Public Authoritative Primary(ies), a different Registered Homenet Domain as the one initially proposed, or a redirection to another domain.
- DNS Homenet Reverse Zone: may be updated every time a new device is connected or dis-connected.
- DNS Homenet Zone: may be updated every time a new device is connected, dis-connected.

In fact, the CPE must be able to perform these updates in a secure manner. There are multiple ways to secure a DNS transaction and this document considers two mechanisms to update a DNS Data (nsupdate and primary/secondary synchronization). Which security mechanism to use to secure a DNS transaction depends on the expected security (authentication of the authoritative server, mutual authentication, confidentiality...). The expected security may also depends on the kind of transaction performed by the CPE. Section 4.2 describes the

different security mechanisms considered in the document as well as their respective goals. Which mechanism to use to update the DNS Data depends on the kind of update. Frequency of the update, size of the DNS Data to update may be some useful criteria. Section 4.3 positions the nsupdate and primary/secondary synchronization mechanisms.

#### 4.2. Mechanisms Securing DNS Transactions

Multiple protocols like IPsec [RFC4301] or TLS / DTLS [RFC5246] / [RFC6347] may be used to secure DNS transactions between the CPE and the DNS servers. This document restricts the scope of security protocols to those that have been designed specifically for DNS. This includes DNSSEC [RFC4033], [RFC4034], [RFC4035] that authenticates and provides integrity protection of DNS data, TSIG [RFC2845], [RFC2930] that use a shared secret to secure a transaction between two end points and SIG(0) [RFC2931] authenticates the DNS packet exchanged.

The key issue with TSIG is that a shared secret must be negotiated between the CPE and the server. On the other hand, TSIG performs symmetric cryptography which is light in comparison with asymmetric cryptography used by SIG(0). As a result, over large zone transfer, TSIG may be preferred to SIG(0).

This document does not provides means to distribute shared secret for example using a specific DHCP Option. The only assumption made is that the CPE generates or is assigned a public key.

As a result, when the document specifies the transaction is secured with TSIG, it means that either the CPE and the DNS Server have been manually configured with a shared secret, or the shared secret has been negotiated using TKEY [RFC2930], and the TKEY exchanged are secured with SIG(0).

Exchange with the DNS Template Server to retrieve the DNS Homenet Zone Template may be protected by SIG(0), TSIG or DNSSEC. When DNSSEC is used, it means the DNS Template Server only provides integrity protection, and does not necessarily prevents someone else to query the DNS Homenet Zone Template. In addition, DNSSEC is only a way to protect the AXFR queries transaction, in other words, DNSSEC cannot be used to secure updates. If DNSSEC is used to provide integrity protection for the AXFR response, the CPE should proceed to the DNSSEC signature checks. If signature check fails, it MUST reject the response. If the signature check succeeds, the CPE removes all DNSSEC related RRsets (DNSKEY, RRSIG, NSEC\* ...) before building the DNS Homenet Zone. In fact, these DNSSEC related fields

are associated to the DNS Homenet Zone Template and not the DNS Homenet Zone.

Any update exchange should use SIG(0) or TSIG to authenticate the exchange.

#### 4.3. Primary / Secondary Synchronization versus DNS Update

As updates only concern DNS zones, this document only considers DNS update mechanisms such as DNS update [RFC2136] [RFC3007] or a primary / secondary synchronization.

The DNS Homenet Zone Template can only be updated with DNS update. The reason is that the DNS Homenet Zone Template contains static configuration data that is not expected to evolve over time.

The DNS Homenet Reverse Zone and the DNS Homenet Zone can be updated either with DNS update or using a primary / secondary synchronization. As these zones may be large, with frequent updates, we recommend to use the primary / secondary architecture as described in [I-D.ietf-homenet-front-end-naming-delegation]. The primary / secondary mechanism is preferred as it better scales and avoids DoS attacks: First the primary notifies the secondary the zone must be updated, and leaves the secondary to proceed to the update when possible. Then, the NOTIFY message sent by the primary is a small packet that is less likely to load the secondary. At last, the AXFR query performed by the secondary is a small packet sent over TCP (section 4.2 [RFC5936]) which makes unlikely the secondary to perform reflection attacks with a forged NOTIFY. On the other hand, DNS updates can use UDP, packets require more processing than a NOTIFY, and they do not provide the server the opportunity to post-pone the update.

### 5. CPE Configuration

#### 5.1. CPE Primary / Secondary Synchronization Configurations

The primary / secondary architecture is described in [I-D.ietf-homenet-front-end-naming-delegation]. The CPE is configured as a primary whereas the DNS Server is configured as a secondary. The DNS Server represents the Public Authoritative Name Server Set or the Reverse Public Authoritative Name Server Set.

When the CPE is plugged its IP address may be unknown to the secondary. The section details how the CPE or primary communicate the necessary information to set up the secondary.

In order to set the primary / secondary configuration, both primary and secondaries must agree on 1) the zone to be synchronized, 2) the IP address and ports used by both primary and secondary.

#### 5.1.1. CPE / Public Authoritative Name Server Set

The CPE knows the zone to be synchronized by reading the Registered Homenet Domain in the DNS Homenet Zone Template provided by the DHCP Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE). The IP address of the secondary is provided by the DHCP Public Authoritative Name Server Set Option (OPTION\_NAME\_SERVER\_SET).

The Public Authoritative Name Server Set has been configured with the Registered Homenet Domain and the Public Key that identifies the CPE. The only thing missing is the IP address of the CPE. This IP address is provided by the CPE by sending a NOTIFY [RFC1996].

When the CPE has built its DNS Homenet Zone, it sends a NOTIFY message to the Public Authoritative Name Server Sets. Upon receiving the NOTIFY message, the secondary reads the Registered Homenet Domain and checks the NOTIFY is sent by the authorized primary. This can be done using the shared secret (TSIG) or the public key (SIG(0)). Once the NOTIFY has been authenticated, the Public Authoritative Name Server Sets might consider the source IP address of the NOTIFY query to configure the primaries attributes.

#### 5.1.2. CPE / Reverse Public Authoritative Name Server Set

The CPE knows the zone to be synchronized by looking at its assigned prefix. The IP address of the secondary is provided by the DHCP Reverse Public Authoritative Name Server Set Option (OPTION\_REVERSE\_NAME\_SERVER\_SET).

Configuration of the secondary is performed as illustrated in Section 5.1.1.

### 5.2. CPE DNS Data Handling and Update Policies

#### 5.2.1. DNS Homenet Zone Template

The DNS Homenet Zone Template contains at least the related fields of the Public Authoritative Primary(ies) as well as the Homenet Registered Domain, that is SOA, and NS fields. This template might be generated automatically by the owner of the DHCP Server. For example, an ISP might provide a default Homenet Registered Domain as well as default Public Authoritative Primary(ies). This default settings should provide the CPE the necessary pieces of information to set the homenet naming architecture.

If the DNS Homenet Zone Template is not subject to modifications or updates, the owner of the template might only use DNSSEC to enable integrity check.

The DNS Homenet Zone Template might be subject to modification by the CPE. The advantage of using the standard DNS zone format is that standard DNS update mechanism can be used to perform updates. These updates might be accepted or rejected by the owner of the DNS Homenet Zone Template. Policies that defines what is accepted or rejected is out of scope of this document. However, in this document we assume the Registered Homenet Domain is used as an index by the Public Authoritative Name Server Set, and SIG(0), TSIG are used to authenticate the CPE. As a result, the Registered Homenet Domain should not be modified unless the Public Authoritative Name Server Set can handle with it.

#### 5.2.2. DNS (Reverse) Homenet Zone

The DNS Homenet Zone might be generated from the DNS Homenet Zone Template. How the DNS Homenet Zone is generated is out of scope of this document. In some cases, the DNS Homenet Zone might be the exact copy of the DNS Homenet Zone Template. In other cases, it might be generated from the DNS Homenet Zone Template with additional RRsets. In some other cases, the DNS Homenet Zone might be generated without considering the DNS Homenet Zone Template, but only considering specific configuration rules.

In the current document the CPE only sets a single zone that is associated with one single Homenet Registered Domain. The domain might be assigned by the owner of the DNS Homenet Zone Template. This constrain does not prevent the CPE to use multiple domain names. How additional domains are considered is out of scope of this document. One way to handle these additional zones is to configure static redirections to the DNS Homenet Zone using CNAME [RFC2181], [RFC1034], DNAME [RFC6672] or CNAME+DNAME [I-D.sury-dnsext-cname-dname].

### 6. Payload Description

This section details the payload of the DHCP Options. A few DHCP Options are used to advertise a server the CPE may be expect to interact with. Interaction may require to define how the update is expected to be performed as well as how the communication is secured. Security and Update are shared by multiple DHCP Options and are described in separate sections. Section 6.1 describes the security field, Section 6.2 describes the update fields, the remaining sections Section 6.3, Section 6.4, Section 6.5, Section 6.6 describe the DHCP Options.

### 6.1. Security Field

The Security Field of the DHCP Option is represented in Figure 2. It indicates the security mechanism supported by the DNS Server. One of these mechanism MUST be chosen by the CPE in order to perform a transaction with the DNS server. See Section 4.2 for more details.

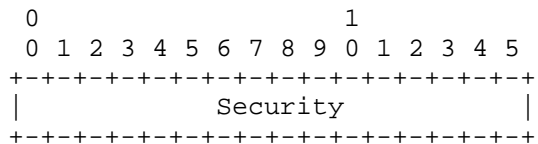


Figure 2: Security Field

- DNS (Bit 0): indicates, when set to 1, that DNS without any security extension is supported.
- DNSSEC (Bit 1): indicates, when set to 1, that DNSSEC provides integrity protection. This can only be used for read operations like retrieving the DNS Homenet Zone Template.
- SIG(0) (Bit 2): indicates, when set to 1, that transaction protected by SIG(0) are supported.
- TSIG (Bit 3): indicates, when set to 1, that transaction using TSIG is supported. Note that if a shared secret has not been previously negotiated between the two party, it should be negotiated using TKEY. The TKEY exchanges MUST be protected with SIG(0) even though SIG(0) is not supported.
- Remaining Bits (Bit 4-15): MUST be set to 0 by the DHCP Server and ignored by the DHCP Client.

A Security field with all bits set to zero indicates the operation is not permitted. The Security field may be set to zero when updates operations are not permitted for the DNS Homenet Template. In any other case this is an error.

### 6.2. Update Field

The Update Field of the DHCP Option is represented in Figure 3. It indicates the update mechanism supported by the DNS server. See Section 4.3 for more details.

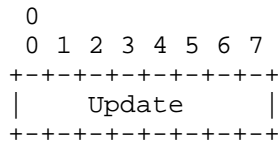


Figure 3: Update Field

- Primary / Secondary (Bit 0): indicates, when set to 1, that DNS Server supports data synchronization using a Primary / Secondary mechanism.
- DNS Update (Bit 1): indicates, when set to 1, that DNS Server supports data synchronization using DNS Updates.
- Remaining Bits (Bit 2-7): MUST be set to 0 by the DHCP Server and ignored by the DHCP Client.

### 6.3. DHCP Public Key Option

The DHCP Public Key Option (OPTION\_PUBLIC\_KEY) indicates the Public Key that is used to authenticate the CPE.

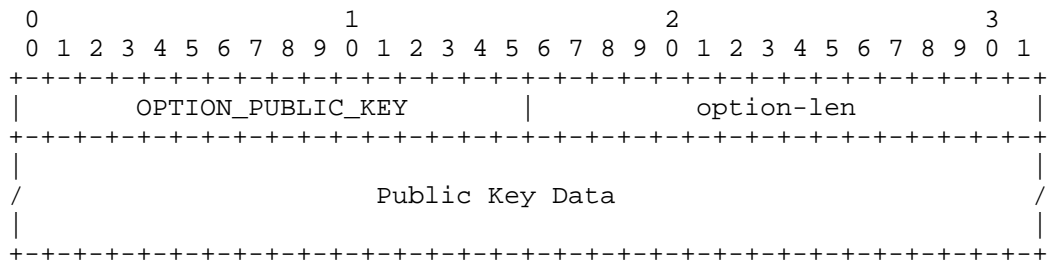


Figure 4: DHCP Public Key Option

- OPTION\_PUBLIC\_KEY (variable): the option code for the DHCP Public Key Option.
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Public Key Data: contains the Public Key. The format is the DNSKEY RDATA format as defined in [RFC4034].

#### 6.4. DHCP Zone Template Option

The DHCP Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE) Option indicates the CPE how to retrieve the DNS Homenet Zone Template. It provides a FQDN the CPE SHOULD query with a DNS query of type AXFR. The option also specifies which security protocols are available on the authoritative server. DNS Homenet Zone Template update, if permitted MUST use the DNS Update mechanism.

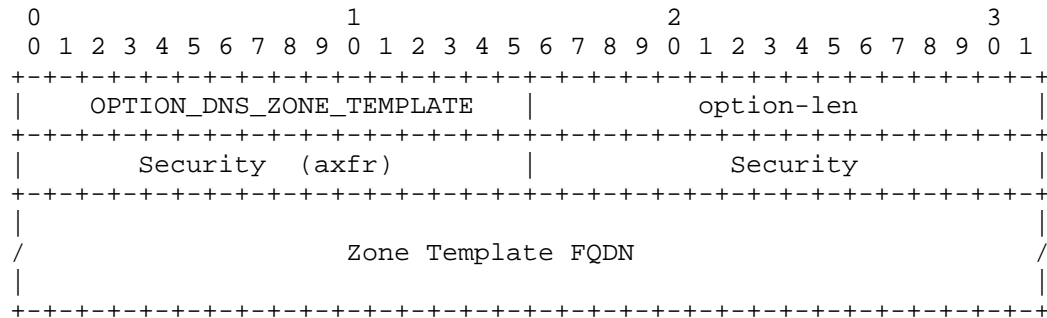


Figure 5: DHCP Zone Template Option

- OPTION\_DNS\_ZONE\_TEMPLATE (variable): the option code for the DHCP Zone Template Option.
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Security (axfr) (16 bits): defines which security protocols are supported by the DNS server. This field concerns the AXFR and consultation queries, not the update queries. See Section 6.1 for more details.
- Security (16 bits): defines which security protocols are supported by the DNS server. This field concerns the update. See Section 6.1 for more details.
- Zone Template FQDN FQDN (variable): the FQDN of the DNS server hosting the DNS Homenet Zone Template.

#### 6.5. DHCP Public Authoritative Name Server Set Option

The DHCP Public Authoritative Name Server Set Option (OPTION\_NAME\_SERVER\_SET) provides information so the CPE can upload the DNS Homenet Zone to the Public Authoritative Name Server Set. Finally, the option provides the security mechanisms that are



available to perform the upload. The upload is performed via a DNS primary / secondary architecture or DNS updates.

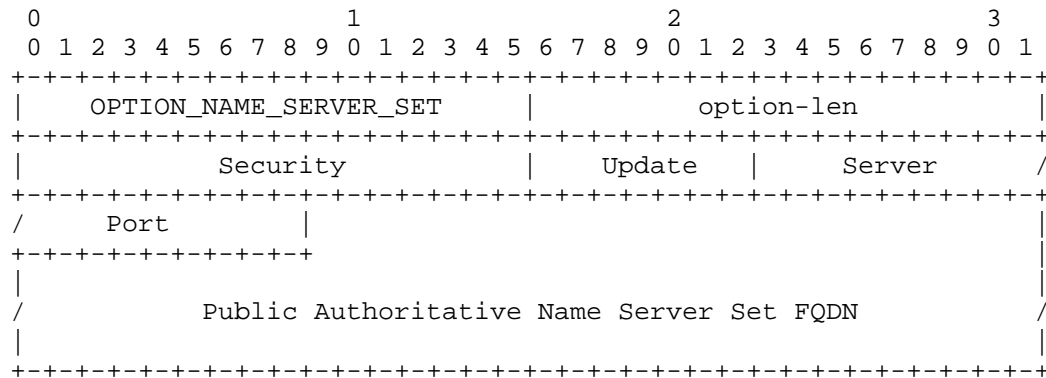


Figure 6: DHCP Public Authoritative Name Server Set Option

- `OPTION_NAME_SERVER_SET` (16 bits): the option code for the DHCP Public Authoritative Name Server Set Option.
- `option-len` (16 bits): length in octets of the option-data field as described in [RFC3315].
- `Security` (16 bits): defines which security protocols are supported by the DNS server. See Section 6.1 for more details.
- `Update` (8 bits): defines which update mechanisms are supported by the DNS server. See Section 4.3 for more details.
- `Server Port` (16 bits): defines the port the Public Authoritative Name Server Set is listening.
- `Public Authoritative Name Server Set FQDN` (variable): the FQDN of the Public Authoritative Name Server Set.

#### 6.6. DHCP Reverse Public Authoritative Name Server Set Option

The DHCP Reverse Public Authoritative Name Server Set Option (`OPTION_REVERSE_NAME_SERVER_SET`) provides information so the CPE can upload the DNS Homenet Zone to the Public Authoritative Name Server Set. The option provides the security mechanisms that are available to perform the upload. The upload is performed via a DNS primary / secondary architecture or DNS updates.

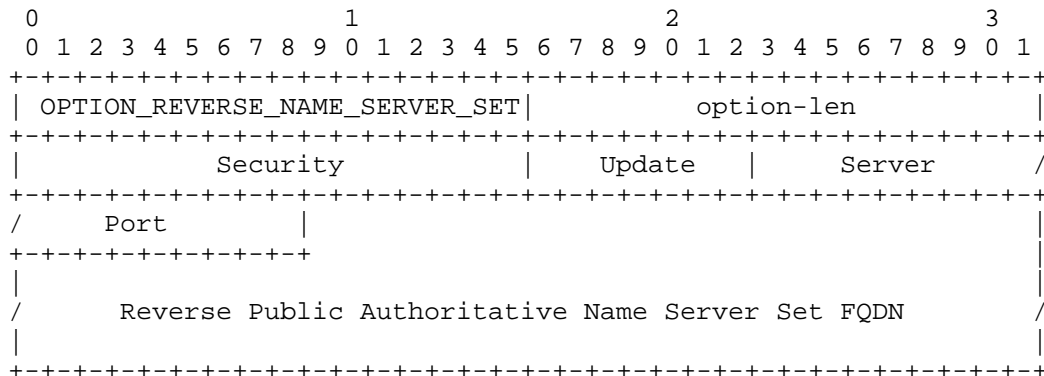


Figure 7: DHCP Reverse Public Authoritative Name Server Set Option

- OPTION\_REVERSE\_NAME\_SERVER\_SET (16 bits): the option code for the DHCP Reverse Public Authoritative Name Server Set Option.
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Security (16 bits): defines which security protocols are supported by the DNS server. See Section 6.1 for more details.
- Update (8 bits): defines which update mechanisms are supported by the DNS server. See Section 4.3 for more details.
- Server Port (16 bits): defines the port the Public Authoritative Name Server Set is listening.
- Reverse Public Authoritative Name Server Set FQDN (variable): The FQDN of the Reverse Public Authoritative Name Server Set.

## 7. DHCP Behavior

### 7.1. DHCPv6 Server Behavior

The DHCP Server sends the DHCP Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE), DHCP Public Authoritative Name Server Set Option (OPTION\_NAME\_SERVER\_SET), DHCP Reverse Public Authoritative Name Server Set Option (OPTION\_REVERSE\_NAME\_SERVER\_SET) upon request by the DHCP Client.

The DHCP Server MAY receive a DHCP Public Key Option (OPTION\_PUBLIC\_KEY) from the CPE. Upon receipt of this DHCP Option, the DHCP Server is expected to communicate this credential to the available DNS Servers like the DNS Template Server, the Public

Authoritative Name Server Set and the Reverse Public Authoritative Name Server Set.

## 7.2. DHCPv6 Client Behavior

The DHCP Client MAY send a DHCP Public Key Option (OPTION\_PUBLIC\_KEY) to the DHCP Server. This Public Key authenticates the CPE.

The DHCP Client sends a DHCP Option Request Option (ORO) with the necessary DHCP options.

A CPE SHOULD only send the an ORO request for DHCP Options it needs or for information that needs to be up-to-date.

Upon receiving a DHCP option described in this document, the CPE SHOULD retrieve or update DNS zones using the associated security and update protocols.

## 7.3. DHCPv6 Relay Behavior

DHCP Relay behavior are not modified by this document.

## 8. IANA Considerations

The DHCP options detailed in this document is:

- OPTION\_DNS\_ZONE\_TEMPLATE: TBD
- OPTION\_NAME\_SERVER\_SET: TBD
- OPTION\_REVERSE\_NAME\_SERVER\_SET: TBD
- OPTION\_PUBLIC\_KEY: TBD

## 9. Security Considerations

### 9.1. DNSSEC is recommended to authenticate DNS hosted data

It is recommended that the (Reverse) DNS Homenet Zone is signed with DNSSEC. The zone may be signed by the CPE or by a third party. We recommend the zone to be signed by the CPE, and that the signed zone is uploaded.

### 9.2. Channel between the CPE and ISP DHCP Server MUST be secured

The document considers that the channel between the CPE and the ISP DHCP Server is trusted. More specifically, the CPE is authenticated and the exchanged messages are protected. The current document does

not specify how to secure the channel. [RFC3315] proposes a DHCP authentication and message exchange protection, [RFC4301], [RFC7296] propose to secure the channel at the IP layer.

In fact, the channel **MUST** be secured because the CPE provides authentication credentials. Unsecured channel may result in CPE impersonation attacks.

### 9.3. CPEs are sensitive to DoS

CPE have not been designed for handling heavy load. The CPE are exposed on the Internet, and their IP address is publicly published on the Internet via the DNS. This makes the Home Network sensitive to Deny of Service Attacks. The resulting outsourcing architecture is described in [I-D.ietf-homenet-front-end-naming-delegation]. This document shows how the outsourcing architecture can be automatically set.

## 10. Acknowledgment

We would like to thank Tomasz Mrugalski, Marcin Siodelski and Bernie Volz for their comments on the design of the DHCP Options. We would also like to thank Mark Andrews, Andrew Sullivan and Lorenzo Colliti for their remarks on the architecture design. The designed solution has been largely been inspired by Mark Andrews's document [I-D.andrews-dnsop-pd-reverse] as well as discussions with Mark.

## 11. References

### 11.1. Normative References

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.
- [RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, August 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

- [RFC2845] Vixie, P., Gudmundsson, O., Eastlake, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.
- [RFC2930] Eastlake, D., "Secret Key Establishment for DNS (TKEY RR)", RFC 2930, September 2000.
- [RFC2931] Eastlake, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, September 2000.
- [RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, November 2000.
- [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.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [RFC5936] Lewis, E. and A. Hoenes, "DNS Zone Transfer Protocol (AXFR)", RFC 5936, June 2010.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC6672] Rose, S. and W. Wijngaards, "DNAME Redirection in the DNS", RFC 6672, June 2012.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, October 2014.

## 11.2. Informational References

[I-D.andrews-dnsop-pd-reverse]

Andrews, M., "Automated Delegation of IP6.ARPA reverse zones with Prefix Delegation", draft-andrews-dnsop-pd-reverse-02 (work in progress), November 2013.

[I-D.ietf-homenet-front-end-naming-delegation]

Migault, D., Cloetens, W., Griffiths, C., and R. Weber, "Outsourcing Home Network Authoritative Naming Service", draft-ietf-homenet-front-end-naming-delegation-02 (work in progress), May 2015.

[I-D.sury-dnsexst-cname-dname]

Sury, O., "CNAME+DNAME Name Redirection", draft-sury-dnsexst-cname-dname-00 (work in progress), April 2010.

## Appendix A. Scenarios and impact on the End User

This section details various scenarios and discuss their impact on the end user.

### A.1. Base Scenario

The base scenario is the one described in Section 4. It is typically the one of an ISP that manages the DHCP Server, and all DNS servers.

The end user subscribes to the ISP (foo), and at subscription time registers for example.foo as its Registered Homenet Domain example.foo. Since the ISP knows the Registered Homenet Domain and the Public Authoritative Primary(ies) the ISP is able to build the DNS Homenet Zone Template.

The ISP manages the DNS Template Server, so it is able to load the DNS Homenet Zone Template on the DNS Template Server.

When the CPE is plugged (at least the first time), it provides its Public Key to the DHCP Server. In this scenario, the DHCP Server and the DNS Servers are managed by the ISP so the DHCP Server can provide authentication credentials of the CPE to enable secure authenticated transaction between the CPE and these DNS servers. More specifically, credentials are provided to:

- Public Authoritative Name Server Set
- Reverse Public Authoritative Name Server Set
- DNS Template Server

The CPE can update the zone using DNS update or a primary / secondary configuration in a secure way.

The main advantage of this scenario is that the naming architecture is configured automatically and transparently for the end user.

The drawbacks are that the end user uses a Registered Homenet Domain managed by the ISP and that it relies on the ISP naming infrastructure.

#### A.2. Third Party Registered Homenet Domain

This section considers the case when the end user wants its home network to use example.com as a Registered Homenet Domain instead of example.foo that has been assigned by the ISP. We also suppose that example.com is not managed by the ISP.

This can also be achieved without any configuration. When the end user buys the domain name example.com, it may request to redirect the name example.com to example.foo using static redirection with CNAME [RFC2181], [RFC1034], DNAME [RFC6672] or CNAME+DNAME [I-D.sury-dnsex-cname-dname].

This configuration is performed once when the domain name example.com is registered. The only information the end user needs to know is the domain name assigned by the ISP. Once this configuration is done no additional configuration is needed anymore. More specifically, the CPE may be changed, the zone can be updated as in Appendix A.1 without any additional configuration from the end user.

The main advantage of this scenario is that the end user benefits from the Zero Configuration of the Base Scenario Appendix A.1. Then, the end user is able to register for its home network an unlimited number of domain names provided by an unlimited number of different third party providers.

The drawback of this scenario may be that the end user still rely on the ISP naming infrastructure. Note that the only case this may be inconvenient is when the DNS Servers provided by the ISPs results in high latency.

#### A.3. Third Party DNS Infrastructure

This scenario considers that the end user uses example.com as a Registered Homenet Domain, and does not want to rely on the authoritative servers provided by the ISP.

In this section we limit the outsourcing to the Public Authoritative Name Server Set and Public Authoritative Primary(ies) to a third party. All other DNS Servers DNS Template Server, Reverse Public Authoritative Primary(ies) and Reverse Public Authoritative Name Server Set remain managed by the ISP. The reason we consider that Reverse Public Authoritative Primary(ies) and Reverse Public Authoritative Name Server Set remains managed by the ISP are that the prefix is managed by the ISP, so outsourcing these resources requires some redirection agreement with the ISP. More specifically the ISP will need to configure the redirection on one of its Reverse DNS Servers. That said, outsourcing these resources is similar as outsourcing Public Authoritative Name Server Set and Public Authoritative Primary(ies) to a third party. Similarly, the DNS Template Server can be easily outsourced as detailed in this section

Outsourcing Public Authoritative Name Server Set and Public Authoritative Primary(ies) requires:

- 1) Updating the DNS Homenet Zone Template: this can be easily done as detailed in Section 4.3 as the DNS Template Server is still managed by the ISP. Such modification can be performed once by any CPE. Once this modification has been performed, the CPE can be changed, the Public Key of the CPE may be changed, this does not need to be done another time. One can imagine a GUI on the CPE asking the end user to fill the field with Registered Homenet Domain, optionally Public Authoritative Primary(ies), with a button "Configure DNS Homenet Zone Template".
- 2) Updating the DHCP Server Information. In fact the Reverse Public Authoritative Name Server Set returned by the ISP is modified. One can imagine a GUI interface that enables the end user to modify its profile parameters. Again, this configuration update is done once-for-ever.
- 3) Upload the authentication credential of the CPE, that is the Public Key of the CPE, to the third party. Unless we use specific mechanisms, like communication between the DHCP Server and the third party, or a specific token that is plugged into the CPE, this operation is likely to be performed every time the CPE is changed, and every time the Public Key generated by the CPE is changed.

The main advantage of this scenario is that the DNS infrastructure is completely outsourced to the third party. Most likely the Public Key that authenticate the CPE need to be configured for every CPE. Configuration is expected to be CPE live-long.



#### A.4. Multiple ISPs

This scenario considers a CPE connected to multiple ISPs.

Firstly, suppose the CPE has been configured with the based scenarios exposed in Appendix A.1. The CPE has multiple interfaces, one for each ISP, and each of these interface is configured using DHCP. The CPE sends to each ISP its DHCP Public Key Option as well as a request for a DHCP Zone Template Option, a DHCP Public Authoritative Name Server Set Option and a DHCP Reverse Public Authoritative Name Server Set Option. Each ISP provides the requested DHCP options, with different values. Note that this scenario assumes, the home network has a different Registered Homenet Domain for each ISP as it is managed by the ISP. On the other hand, the CPE Public Key may be shared between the CPE and the multiple ISPs. The CPE builds the associate DNS(SEC) Homenet Zone, and proceeds to the various settings as described in Appendix A.1.

The protocol and DHCP Options described in this document are fully compatible with a CPE connected to multiple ISPs with multiple Registered Homenet Domains. However, the CPE should be able to handle different Registered Homenet Domains. This is an implementation issue which is outside the scope of the current document. More specifically, multiple Registered Homenet Domains leads to multiple DNS(SEC) Homenet Zones. A basic implementation may erase the DNS(SEC) Homenet Zone that exists when it receives DHCP Options, and rebuild everything from scratch. This will work for an initial configuration but comes with a few drawbacks. First, updates to the DNS(SEC) Homenet Zone may only push to one of the multiple Registered Homenet Domain, the latest Registered Homenet Domain that has been set, and this is most likely expected to be almost randomly chosen as it may depend on the latency on each ISP network at the boot time. As a results, this leads to unsynchronized Registered Homenet Domains. Secondly, if the CPE handles in some ways resolution, only the latest Registered Homenet Domain set may be able to provide naming resolution in case of network disruption.

Secondly, suppose the CPE is connected to multiple ISP with a single Registered Homenet Domain. In this case, the one party is chosen to host the Registered Homenet Domain. This entity may be one of the ISP or a third party. Note that having multiple ISPs can be motivated for bandwidth aggregation, or connectivity fail-over. In the case of connectivity fail-over, the fail-over concerns the access network and a failure of the access network may not impact the core network where the Public Authoritative Name Server Set and Public Authoritative Primaries are hosted. In that sense, choosing one of the ISP even in a scenario of multiple ISPs may make sense. However, for sake of simplicity, this scenario assumes that a third party has

be chosen to host the Registered Homenet Domain. The DNS settings for each ISP is described in Appendix A.2 and Appendix A.3. With the configuration described in Appendix A.2, the CPE is expect to be able to handle multiple Homenet Registered Domain, as the third party redirect to one of the ISPs Servers. With the configuration described in Appendix A.3, DNS zone are hosted and maintained by the third party. A single DNS(SEC) Homenet Zone is built and maintained by the CPE. This latter configuration is likely to match most CPE implementations.

The protocol and DHCP Options described in this document are fully compatible with a CPE connected to multiple ISPs. To configure or not and how to configure the CPE depends on the CPE facilities. Appendix A.1 and Appendix A.2 require the CPE to handle multiple Registered Homenet Domain, whereas Appendix A.3 does not have such requirement.

#### Appendix B. Document Change Log

[RFC Editor: This section is to be removed before publication]

-05: changing Master to Primary, Slave to Secondary

-04: Working Version Major modifications are:

- Re-structuring the draft: description and comparison of update and security mechanisms have been intergrated into the Overview section. a Configuration section has been created to describe both configuration and corresponding behavior of the CPE.
- Adding Ports parameters: Server Set can configure a port. The Port Server parameter have been added in the DHCP Option payloads because middle boxes may not be configured to let port 53 packets and it may also be useful to split servers among different ports, assigning each end user a different port.
- Multiple ISP scenario: In order to address comments, the multiple ISPs scenario has been described to explicitly show that the protocol and DHCP Options do not prevent a CPE connected to multiple independent ISPs.

-03: Working Version Major modifications are:

- Redesigning options/scope: according to feed backs received from the IETF89 presentation in the dhc WG.

- Redesigning architecture: according to feed backs received from the IETF89 presentation in the homenet WG, discussion with Mark and Lorenzo.
- 02: Working Version Major modifications are:
  - Redesigning options/scope: As suggested by Bernie Volz
- 01: Working Version Major modifications are:
  - Remove the DNS Zone file construction: As suggested by Bernie Volz
  - DHCPv6 Client behavior: Following options guide lines
  - DHCPv6 Server behavior: Following options guide lines
- 00: version published in the homenet WG. Major modifications are:
  - Reformatting of DHCP Options: Following options guide lines
  - DHCPv6 Client behavior: Following options guide lines
  - DHCPv6 Server behavior: Following options guide lines
- 00: First version published in dhc WG.

#### Authors' Addresses

Daniel Migault  
Ericsson  
8400 boulevard Decarie  
Montreal, QC H4P 2N2  
Canada

Email: daniel.migault@ericsson.com

Wouter Cloetens  
SoftAtHome  
vaartdijk 3 701  
3018 Wijgmaal  
Belgium

Email: wouter.cloetens@softathome.com

Chris Griffiths  
Dyn  
150 Dow Street  
Manchester, NH 03101  
US

Email: [cgriffiths@dyn.com](mailto:cgriffiths@dyn.com)  
URI: <http://dyn.com>

Ralf Weber  
Nominum  
2000 Seaport Blvd #400  
Redwood City, CA 94063  
US

Email: [ralf.weber@nominum.com](mailto:ralf.weber@nominum.com)  
URI: <http://www.nominum.com>

HOMENET  
Internet-Draft  
Intended status: Standards Track  
Expires: April 21, 2016

D. Migault (Ed)  
Ericsson  
T. Mrugalski  
ISC  
C. Griffiths

R. Weber  
Nominum  
W. Cloetens  
SoftAtHome  
October 19, 2015

DHCPv6 Options for Homenet Naming Architecture  
draft-ietf-homenet-naming-architecture-dhc-options-03.txt

Abstract

Customer Premises Equipment (CPE) devices are usually constrained devices with reduced network and CPU capabilities. As such, a CPE exposing the authoritative naming service for its home network on the Internet may become vulnerable to resource exhaustion attacks. One way to avoid exposing CPE is to outsource the authoritative service to a third party, e.g. ISP. Such an outsource requires setting up an architecture which may be inappropriate for most end users. This document defines DHCPv6 options so any agnostic CPE can automatically proceed to the appropriate configuration and outsource the authoritative naming service for the home network. In most cases, the outsourcing mechanism is transparent for the end user.

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

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Requirements notation . . . . .	3
2. Terminology . . . . .	3
3. Introduction . . . . .	3
4. Protocol Overview . . . . .	6
4.1. Architecture and DHCPv6 Options Overview . . . . .	6
4.2. Mechanisms Securing DNS Transactions . . . . .	9
4.3. Primary / Secondary Synchronization versus DNS Update . .	10
5. CPE Configuration . . . . .	11
5.1. CPE Primary / Secondary Synchronization Configurations .	11
5.1.1. CPE / Synchronization Server . . . . .	11
5.1.2. CPE / Reverse Synchronization Server . . . . .	11
5.2. CPE DNS Data Handling and Update Policies . . . . .	12
5.2.1. Homenet Zone Template . . . . .	12
5.2.2. DNS (Reverse) Homenet Zone . . . . .	12
6. Payload Description . . . . .	13
6.1. Supported Authentication Methods Field . . . . .	13
6.2. Update Field . . . . .	14
6.3. Client Public Key Option . . . . .	14
6.4. Zone Template Option . . . . .	14
6.5. Synchronization Server Option . . . . .	15
6.6. Reverse Synchronization Server Option . . . . .	16
7. DHCP Behavior . . . . .	17
7.1. DHCPv6 Server Behavior . . . . .	17
7.2. DHCPv6 Client Behavior . . . . .	18
7.3. DHCPv6 Relay Agent Behavior . . . . .	18
8. IANA Considerations . . . . .	18
9. Security Considerations . . . . .	19
9.1. DNSSEC is recommended to authenticate DNS hosted data .	19
9.2. Channel between the CPE and ISP DHCP Server MUST be secured . . . . .	19
9.3. CPEs are sensitive to DoS . . . . .	19

10. Acknowledgments . . . . .	19
11. References . . . . .	19
11.1. Normative References . . . . .	20
11.2. Informational References . . . . .	21
Appendix A. Scenarios and impact on the End User . . . . .	22
A.1. Base Scenario . . . . .	22
A.2. Third Party Registered Homenet Domain . . . . .	23
A.3. Third Party DNS Infrastructure . . . . .	23
A.4. Multiple ISPs . . . . .	25
Appendix B. Document Change Log . . . . .	26
Authors' Addresses . . . . .	27

## 1. Requirements notation

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].

## 2. Terminology

The reader is expected to be familiar with [I-D.ietf-homenet-front-end-naming-delegation] and its terminology section. This section defines terms that have not been defined in [I-D.ietf-homenet-front-end-naming-delegation]:

- Client Public Key: designates a public key generated by the CPE. This key is used as an authentication credential for the CPE.
- Homenet Zone Template: The template used as a basis to generate the Homenet Zone.
- DNS Template Server: The DNS server that hosts the Homenet Zone Template.
- Homenet Reverse Zone: The reverse zone file associated to the Homenet Zone.

## 3. Introduction

CPEs are usually constrained devices with reduced network and CPU capacities. As such, a CPE hosting on the Internet the authoritative naming service for its home network may become vulnerable to resource exhaustion attacks. Outsourcing the authoritative service to a third party avoids exposing the CPE to such attacks. This third party can be the ISP or any other independent third party.

Outsourcing the authoritative naming service to a third party requires setting up an architecture designated in this document as

the Outsourcing Infrastructure. These settings may be inappropriate for most end users that do not have the sufficient knowledge. To address this issue, this document proposes DHCPv6 options so any agnostic CPE can automatically set the Outsourcing Infrastructure. In most cases, these DHCPv6 options are sufficient and do not require any additional interaction from the end user, thus achieving a zero-config settings. In some other cases, the end user is expected to perform some limited manual configuration.

When the CPE is plugged, the DHCPv6 options described in the document enable:

- 1. To build the Homenet Zone: Building the Homenet Zone requires filling the zone with appropriated bindings such as bindings between the names and the IP addresses of the different devices of the home networks. How the CPE is aware of these binding is out of scope of the document. They may be provided, for example, by the DHCPv6 server hosted on the CPE. On the other hand, building the Homenet Zone also requires configuration parameters like the name of the Registered Domain Name associated to the home network or the Public Authoritative Server(s) the Homenet Zone is outsourced to. These configuration parameters are stored in the Homenet Zone Template. This document describes the Zone Template Option which carries the FQDN associated to the Homenet Zone Template. In order to retrieve the Homenet Zone Template, the CPE sends a query of type AXFR [RFC1034], [RFC5936].
- 2. To upload the Homenet Zone to the Synchronization Server, in charge of publishing the Homenet Zone on the Public Authoritative Server(s). This document describes the Synchronization Server Option that provides the FQDN of the appropriated server. Note that, the document does not consider whether the Homenet Zone is signed or not, and if signed, which entity is responsible to sign it. Such questions are out of the scope of the current document.
- 3. To upload the Homenet Reverse Zone to the Reverse Synchronization Server in charge of publishing the Homenet Reverse Zone on the Reverse Public Authoritative Server(s). This document describes the Reverse Synchronization Server Option that provides the FQDN of the appropriated server. Similarly to item 2., we do not consider in this document if the Homenet Reverse Zone is signed or not, and if signed who signs it.
- 4. To provide authentication credential (a public key) to the DHCP Server: Information stored in the Homenet Zone Template, the



Homenet Zone and Homenet Reverse Zone belongs to the CPE, and only the CPE should be able to update or upload these zones. To authenticate the CPE, this document defines the Client Public Key Option. This option is sent by the CPE to the DHCPv6 server and provides the Client Public Key the CPE uses to authenticate itself. This document does not describe mechanisms used to transmit the Client Public Key from the DHCPv6 server to the appropriate entities. If the DHCPv6 server is not able to provide the Client Public Key to the appropriated entities, then the end user is likely to provide manually the Client Public Key to these entities. This document illustrates two scenarios: one where the DHCPv6 server is responsible for distributing the Client Public Key to the Synchronization Servers and Reverse Synchronization Server. In the other scenarios, the Client Public Key is distributed out of band.

The DHCPv6 options described in this document make possible to configure an Outsourcing Infrastructure with no or little configurations from the end user. A zero-config setting is achieved if the the link between the CPE and the DHCPv6 server and the link between the DHCPv6 server and the various DNS servers (Homenet Zone Server, the Reverse Synchronization Server, Synchronization Server) are trusted. For example, one way to provide a trustworthy connection between the CPE and the DHCPv6 server is defined in [I-D.ietf-dhc-sedhcpv6]. When both links are trusted, the CPE is able to provide its authentication credentials (a Client Public Key) to the DHCPv6 server, that in turn forwards it to the various DNS servers. With the authentication credentials on the DNS servers, the CPE is able to securely update.

If the DHCPv6 server cannot provide the Client Public Key to one of these servers (most likely the Synchronization Server) and the CPE needs to interact with the server, then, the end user is expected to provide the CPE's Client Public Key to these servers (the Reverse Synchronization Server or the Synchronization Server) either manually or using other mechanisms. Such mechanisms are outside the scope of this document. In that case, the authentication credentials need to be provided every time the key is modified. Appendix A provides more details on how different scenarios impact the end users.

The remaining of this document is structured as follows. Section 4 provides an overview of the DHCPv6 options as well as the expected interactions between the CPE and the various involved entities. This section also provides an overview of available mechanisms to secure DNS transactions and update DNS data. Section 5 describes how the CPE may securely synchronize and update DNS data. Section 6 describes the payload of the DHCPv6 options and Section 7 details how

DHCPv6 client, server and relay agent behave. Section 8 lists the new parameters to be registered at the IANA, Section 9 provides security considerations. Finally, Appendix A describes how the CPE may behave and be configured regarding various scenarios.

#### 4. Protocol Overview

This section provides an overview of the CPE's interactions with the Outsourcing Infrastructure in Section 4.1, and so the necessary for its setting. In this document, the configuration is provided via DHCPv6 options. Once configured, the CPE is expected to be able to update and publish DNS data on the different components of the Outsourcing Infrastructure. As a result authenticating and updating mechanisms play an important role in the specification. Section 4.2 provides an overview of the different authentication methods and Section 4.3 provides an overview of the different update mechanisms considered to update the DNS data.

##### 4.1. Architecture and DHCPv6 Options Overview

This section illustrates how a CPE receives the necessary information via DHCPv6 options to outsource its authoritative naming service on the Outsourcing Infrastructure. For the sake of simplicity, this section assumes that the DHCPv6 server is able to communicate to the various DNS servers and to provide them the public key associated with the CPE. Once each server got the public key, the CPE can proceed to transactions in an authenticated and secure way.

This scenario has been chosen as it is believed to be the most popular scenario. This document does not ignore that scenarios where the DHCP Server does not have privileged relations with the Synchronization Server must be considered. These cases are discussed latter in Appendix A. Such scenario does not necessarily require configuration for the end user and can also be zero-config.

The scenario is represented in Figure 1.

- 1: The CPE provides its Client Public Key to the DHCP Server using a Client Public Key Option (OPTION\_PUBLIC\_KEY) and includes the following option codes in its Option Request Option (ORO): Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE), the Synchronization Server Option (OPTION\_SYNC\_SERVER) and the Reverse Synchronization Server Option (OPTION\_REVERSE\_SYNC\_SERVER).
- 2: The DHCP Server makes the Client Public Key available to the DNS servers, so the CPE can secure its DNS transactions. How the Client Public Key is transmitted to the various DNS servers

is out of scope of this document. Note that the Client Public Key alone is not sufficient to perform the authentication and the key should be, for example, associated with an identifier, or the concerned domain name. How the binding is performed is out of scope of the document. It can be a centralized database or various bindings may be sent to the different servers. Figure 1 represents the specific case where the DHCP Server forwards the set (Client Public Key, Zone Template FQDN) to the DNS Template Server, the set (Client Public Key, IPv6 subnet) to the Reverse Synchronization Server and the set (Client Public Key, Registered Homenet Domain) to the Synchronization Server.

- 3: The DHCP Server responds to the CPE with the requested DHCPv6 options, i.e. the Client Public Key Option (OPTION\_PUBLIC\_KEY), Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE), Synchronization Server Option (OPTION\_SYNC\_SERVER), Reverse Synchronization Server Option (OPTION\_REVERSE\_SYNC\_SERVER). Note that this step may be performed in parallel to step 2, or even before. In other words, there is no requirements that step 3 is conducted after step 2.
- 4: Upon receiving the Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE), the CPE performs an AXFR DNS query for the Zone Template FQDN. The exchange is authenticated according to the authentication methods defined in the Supported Authentication Methods field of the DHCP option. Once the CPE has retrieved the DNS Zone Template, the CPE can build the Homenet Zone and the Homenet Reverse Zone. Eventually the CPE signs these zones.
- 5: Once the Homenet Reverse Zone has been set, the CPE uploads the zone to the Reverse Synchronization Server. The Reverse Synchronization Server Option (OPTION\_REVERSE\_SYNC\_SERVER) provides the Reverse Synchronization Server FQDN as well as the upload method, and the Supported Authentication Methods protocol to secure the upload.
- 6: Once the Homenet Zone has been set, the CPE uploads the zone to the Synchronization Server. The Synchronization Server Option (OPTION\_SYNC\_SERVER) provides the Synchronization Server FQDN as well as the upload method and the authentication method to secure the upload.

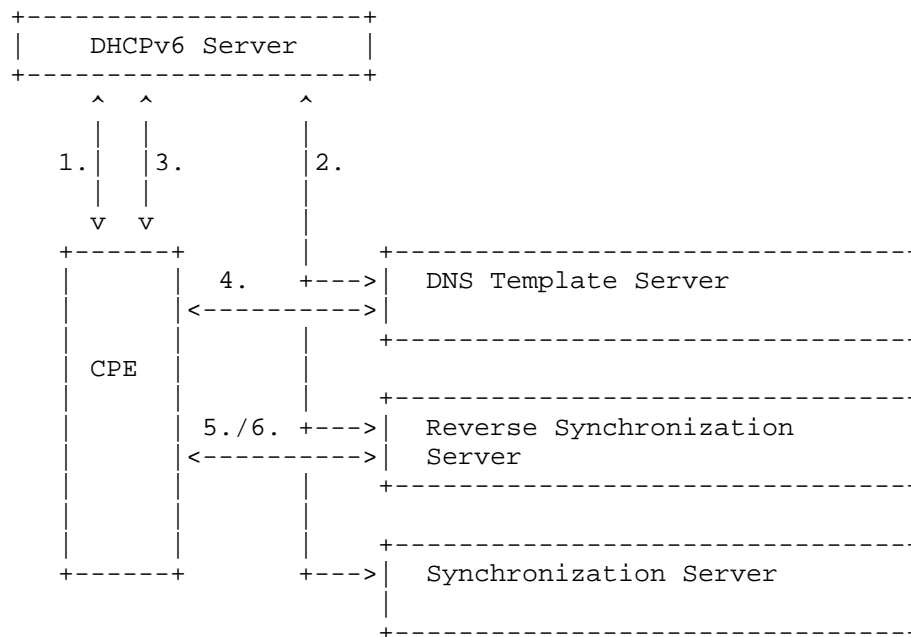


Figure 1: Protocol Overview

As described above, the CPE is likely to interact with various DNS content. More specifically, the CPE is likely to update the:

- Homenet Zone Template: if the configuration of the zone may be changed. This may include additional Public Authoritative Server(s), a different Registered Homenet Domain as the one initially proposed, or a redirection to another domain.
- Homenet Reverse Zone: every time a new device is connected or dis-connected.
- Homenet Zone: every time a new device is connected, dis-connected.

Step 2 and step 3 should be considered as independent steps and could be re-ordered. In fact, the DHCPv6 server does not have to wait for a confirmation from the DNS servers the Client Public Key has been properly received, and is operational by the DNS servers. The DHCP Server is expected to reply upon receiving the Client Public Key Option. The reply to the message with a Client Public Key Option from the DHCP Server is interpreted by the DHCPv6 client as a confirmation of the reception of the option by the DHCP Server only. It does not indicate whether the server had processed the option or

not. Debugging configurations errors or transmission error with one of the DNS servers is let to the CPE and thus is outside of the scope of the DHCPv6. First, it is unlikely a DNS server can validate that the Client Public Key will be operational for the CPE, as multiple causes of errors could occur. For example, the Client Public Key may have been changed during the transmission or by the DHCP Server, or the DNS server may be misconfigured. Second, the number of error codes would be too complex. In addition to multiple causes of errors, multiple architectures and multiple DNS servers may be involved. Third, this may cause significant DHCP Server performance degradation.

In fact, the CPE performs these updates in a secure manner. There are multiple ways to secure a DNS transaction and this document considers two mechanisms: nsupdate and primary/secondary synchronization. Section 4.2 describes the authentication method that may be use to secure the DNS transactions of the CPE. The appropriate authentication methods may, for example, be chosen according to the level of confidentiality or the level of authentication requested by the CPE transactions. Section 4.3 positions the nsupdate and primary/secondary synchronization mechanisms. The update appropriate update mechanism may depend on the for example on the update frequency or the size of the DNS data to update.

#### 4.2. Mechanisms Securing DNS Transactions

Multiple protocols like IPsec [RFC4301] or TLS / DTLS [RFC5246] / [RFC6347] may be used to secure DNS transactions between the CPE and the DNS servers. This document limits its scope to authentication method that have been designed specifically for DNS. This includes DNSSEC [RFC4033], [RFC4034], [RFC4035] that authenticates and provides integrity protection of DNS data, TSIG [RFC2845], [RFC2930] that use a shared secret to secure a transaction between two end points and SIG(0) [RFC2931] authenticates the DNS packet exchanged.

The key issue with TSIG is that a shared secret must be negotiated between the CPE and the server. On the other hand, TSIG performs symmetric cryptography which is light in comparison with asymmetric cryptography used by SIG(0). As a result, over large zone transfer, TSIG may be preferred to SIG(0).

This document does not provide means to distribute shared secret for example using a specific DHCPv6 option. The only assumption made is that the CPE generates or is assigned a public key.

As a result, when the document specifies the transaction is secured with TSIG, it means that either the CPE and the DNS server have been

manually configured with a shared secret, or the shared secret has been negotiated using TKEY [RFC2930], and the TKEY exchanged are secured with SIG(0).

Exchanges with the DNS Template Server to retrieve the Homenet Zone Template may be protected by SIG(0), TSIG or DNSSEC. When DNSSEC is used, it means the DNS Template Server only provides integrity protection, and does not necessarily prevent someone else to query the Homenet Zone Template. In addition, DNSSEC is only a way to protect the AXFR queries transaction, in other words, DNSSEC cannot be used to secure updates. If DNSSEC is used to provide integrity protection for the AXFR response, the CPE should proceed to the DNSSEC signature checks. If signature check fails, it MUST reject the response. If the signature check succeeds, the CPE removes all DNSSEC related RRsets (DNSKEY, RRSIG, NSEC\* ...) before building the Homenet Zone. In fact, these DNSSEC related fields are associated to the Homenet Zone Template and not the Homenet Zone.

Any update exchange should use SIG(0) or TSIG to authenticate the exchange.

#### 4.3. Primary / Secondary Synchronization versus DNS Update

As updates only concern DNS zones, this document only considers DNS update mechanisms such as DNS update [RFC2136] [RFC3007] or a primary / secondary synchronization.

The Homenet Zone Template SHOULD be updated with DNS update as it contains static configuration data that is not expected to evolve over time.

The Homenet Reverse Zone and the Homenet Zone can be updated either with DNS update or using a primary / secondary synchronization. As these zones may be large, with frequent updates, we recommend to use the primary / secondary architecture as described in [I-D.ietf-homenet-front-end-naming-delegation]. The primary / secondary mechanism is preferred as it better scales and avoids DoS attacks: First the primary notifies the secondary the zone must be updated, and leaves the secondary to proceed to the update when possible. Then, the NOTIFY message sent by the primary is a small packet that is less likely to load the secondary. At last, the AXFR query performed by the secondary is a small packet sent over TCP (section 4.2 [RFC5936]) which makes unlikely the secondary to perform reflection attacks with a forged NOTIFY. On the other hand, DNS updates can use UDP, packets require more processing than a NOTIFY, and they do not provide the server the opportunity to postpone the update.

## 5. CPE Configuration

### 5.1. CPE Primary / Secondary Synchronization Configurations

The primary / secondary architecture is described in [I-D.ietf-homenet-front-end-naming-delegation]. The CPE hosts a Hidden Primary that synchronizes with a Synchronization Server or the Reverse Synchronization Server.

When the CPE is plugged its IP address may be unknown to the secondary. The section details how the CPE or primary communicates the necessary information to set up the secondary.

In order to set the primary / secondary configuration, both primary and secondaries must agree on 1) the zone to be synchronized, 2) the IP address and ports used by both primary and secondary.

#### 5.1.1. CPE / Synchronization Server

The CPE is aware of the zone to be synchronized by reading the Registered Homenet Domain in the Homenet Zone Template provided by the Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE). The IP address of the secondary is provided by the Synchronization Server Option (OPTION\_SYNC\_SERVER).

The Synchronization Server has been configured with the Registered Homenet Domain and the Client Public Key that identifies the CPE. The only missing information is the IP address of the CPE. This IP address is provided by the CPE by sending a NOTIFY [RFC1996].

When the CPE has built its Homenet Zone, it sends a NOTIFY message to the Synchronization Servers. Upon receiving the NOTIFY message, the secondary reads the Registered Homenet Domain and checks the NOTIFY is sent by the authorized primary. This can be done using the shared secret (TSIG) or the public key (SIG(0)). Once the NOTIFY has been authenticated, the Synchronization Servers might consider the source IP address of the NOTIFY query to configure the primaries attributes.

#### 5.1.2. CPE / Reverse Synchronization Server

The CPE is aware of the zone to be synchronized by looking at its assigned prefix. The IP address of the secondary is provided by the Reverse Synchronization Server Option (OPTION\_REVERSE\_SYNC\_SERVER).

Configuration of the secondary is performed as illustrated in Section 5.1.1.

## 5.2. CPE DNS Data Handling and Update Policies

### 5.2.1. Homenet Zone Template

The Homenet Zone Template contains at least the related fields of the Public Authoritative Server(s) as well as the Homenet Registered Domain, that is SOA, and NS fields. This template might be generated automatically by the owner of the DHCP Server. For example, an ISP might provide a default Homenet Registered Domain as well as default Public Authoritative Server(s). This default settings should provide the CPE the necessary pieces of information to set the homenet naming architecture.

If the Homenet Zone Template is not subject to modifications or updates, the owner of the template might only use DNSSEC to enable integrity check.

On the other hand, the Homenet Zone Template might also be subject to modification by the CPE. The advantage of using the standard DNS zone format is that standard DNS update mechanism can be used to perform updates. These updates might be accepted or rejected by the owner of the Homenet Zone Template. Policies that defines what is accepted or rejected is out of scope of this document. However, this document assumes the Registered Homenet Domain is used as an index by the Synchronization Server, and SIG(0), TSIG are used to authenticate the CPE. As a result, the Registered Homenet Domain should not be modified unless the Synchronization Server can handle with it.

### 5.2.2. DNS (Reverse) Homenet Zone

The Homenet Zone might be generated from the Homenet Zone Template. How the Homenet Zone is generated is out of scope of this document. In some cases, the Homenet Zone might be the exact copy of the Homenet Zone Template. In other cases, it might be generated from the Homenet Zone Template with additional RRsets. In some other cases, the Homenet Zone might be generated without considering the Homenet Zone Template, but only considering specific configuration rules.

In the current document the CPE only sets a single zone that is associated with one single Homenet Registered Domain. The domain might be assigned by the owner of the Homenet Zone Template. This constraint does not prevent the CPE to use multiple domain names. How additional domains are considered is out of scope of this document. One way to handle these additional zones is to configure static redirections to the Homenet Zone using CNAME [RFC2181], [RFC1034], DNAME [RFC6672] or CNAME+DNAME [I-D.sury-dnsext-cname-dname].



## 6. Payload Description

This section details the payload of the DHCPv6 options. A few DHCPv6 options are used to advertise a server the CPE may be expected to interact with. Interaction may require to define update and authentication methods. Update fields are shared by multiple DHCPv6 options and are described in separate sections. Section 6.1 describes the Supported Authentication Method field, Section 6.2 describes the Update field, the remaining Section 6.3, Section 6.4, Section 6.5, Section 6.6 describe the DHCPv6 options.

### 6.1. Supported Authentication Methods Field

The Supported Authentication Methods field of the DHCPv6 option represented in Figure 2 indicates the authentication method supported by the DNS server. One of these mechanism MUST be chosen by the CPE in order to perform a transaction with the DNS server. See Section 4.2 for more details.

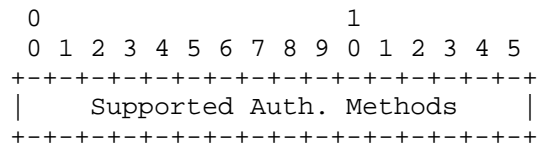


Figure 2: Supported Authentication Methods Filed

- DNS (Bit 0): indicates, when set to 1, that DNS without any security extension is supported.
- DNSSEC (Bit 1): indicates, when set to 1, that DNSSEC provides integrity protection. This can only be used for read operations like retrieving the Homenet Zone Template.
- SIG(0) (Bit 2): indicates, when set to 1, that transaction protected by SIG(0) are supported.
- TSIG (Bit 3): indicates, when set to 1, that transaction using TSIG is supported. Note that if a shared secret has not been previously negotiated between the two party, it should be negotiated using TKEY. The TKEY exchanges MUST be protected with SIG(0) even though SIG(0) is not supported.
- Remaining Bits (Bit 4-15): MUST be set to 0 by the DHCP Server and MUST be ignored by the DHCPv6 client.

A Supported Authentication Methods field with all bits set to zero indicates the operation is not permitted. The Supported

Authentication Methods field may be set to zero when updates operations are not permitted for the DNS Homenet Template. In any other case this is an error.

## 6.2. Update Field

The Update Field of the DHCPv6 option is represented in Figure 3. It indicates the update mechanism supported by the DNS server. See Section 4.3 for more details.

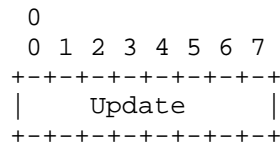


Figure 3: Update Field

- Primary / Secondary (Bit 0): indicates, when set to 1, that DNS Server supports data synchronization using a Primary / Secondary mechanism.
- DNS Update (Bit 1): indicates, when set to 1, that DNS Server supports data synchronization using DNS Updates.
- Remaining Bits (Bit 2-7): MUST be set to 0 by the DHCPv6 server and MUST be ignored by the DHCPv6 client.

## 6.3. Client Public Key Option

The Client Public Key Option (OPTION\_PUBLIC\_KEY) indicates the Client Public Key that is used to authenticate the CPE. This option is defined in [I-D.ietf-dhc-sedhcpv6].

## 6.4. Zone Template Option

The Zone Template Option (OPTION\_DNS\_ZONE\_TEMPLATE) Option indicates the CPE how to retrieve the Homenet Zone Template. It provides a FQDN the CPE SHOULD query with a DNS query of type AXFR as well as the authentication methods associated to the AXFR query or the nsupdate queries. Homenet Zone Template update, if permitted MUST use the DNS Update mechanism.

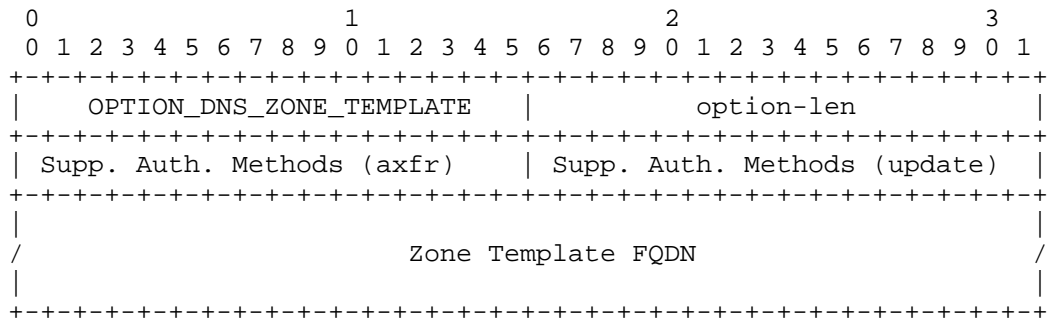


Figure 4: Zone Template Option

- option-code: (16 bits): `OPTION_DNS_ZONE_TEMPLATE`, the option code for the Zone Template Option (TBD1).
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Supported Authentication Methods(axfr) (16 bits): defines which authentication methods are supported by the DNS server. This field concerns the AXFR and consultation queries, not the update queries. See Section 6.1 for more details.
- Supported Authentication Methods (16 bits): defines which authentication methods are supported by the DNS server. This field concerns the update. See Section 6.1 for more details.
- Zone Template FQDN FQDN (variable): the FQDN of the DNS server hosting the Homenet Zone Template.

#### 6.5. Synchronization Server Option

The Synchronization Server Option (`OPTION_SYNC_SERVER`) provides information necessary for the CPE to upload the Homenet Zone to the Synchronization Server. Finally, the option provides the authentication methods that are available to perform the upload. The upload is performed via a DNS primary / secondary architecture or DNS updates.

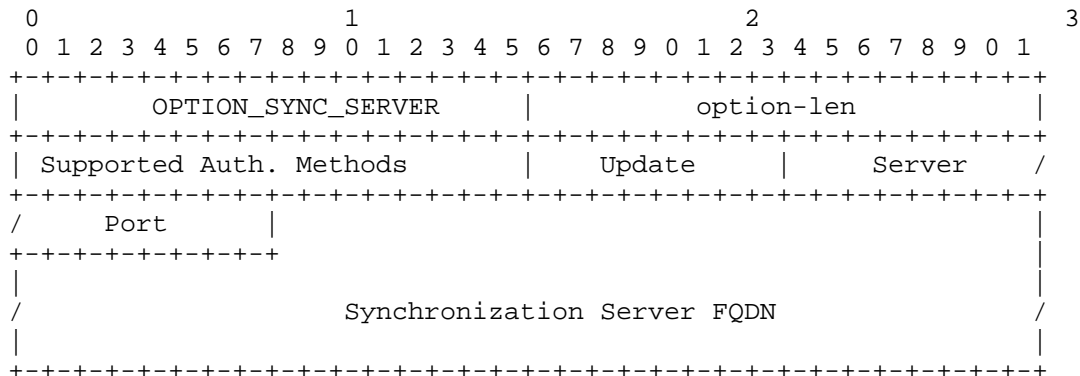


Figure 5: Synchronization Server Option

- option-code (16 bits): `OPTION_SYNC_SERVER`, the option code for the Synchronization Server Option (TBD2).
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Supported Authentication Methods (16 bits): defines which authentication methods are supported by the DNS server. See Section 6.1 for more details.
- Update (8 bits): defines which update mechanisms are supported by the DNS server. See Section 4.3 for more details.
- Server Port (16 bits): defines the port the Synchronization Server is listening. When multiple transport layers may be used, a single and unique Server Port value applies to all the transport layers. In the case of DNS for example, Server Port value considers DNS exchanges using UDP and TCP.
- Synchronization Server FQDN (variable): the FQDN of the Synchronization Server.

#### 6.6. Reverse Synchronization Server Option

The Reverse Synchronization Server Option (`OPTION_REVERSE_SYNC_SERVER`) provides information necessary for the CPE to upload the Homenet Zone to the Synchronization Server. The option provides the authentication methods that are available to perform the upload. The upload is performed via a DNS primary / secondary architecture or DNS updates.

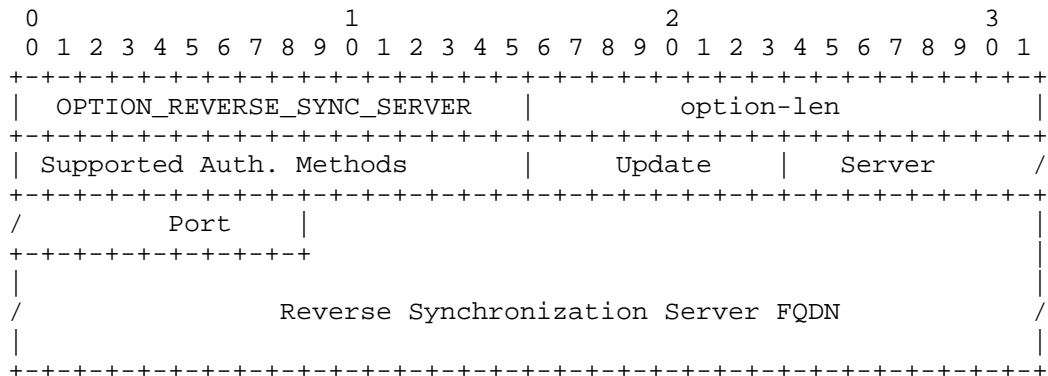


Figure 6: Reverse Synchronization Server Option

- option-code (16 bits): `OPTION_REVERSE_SYNC_SERVER`, the option code for the Reverse Synchronization Server Option (TBD3).
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Supported Authentication Methods (16 bits): defines which authentication methods are supported by the DNS server. See Section 6.1 for more details.
- Update (8 bits): defines which update mechanisms are supported by the DNS server. See Section 4.3 for more details.
- Server Port (16 bits): defines the port the Synchronization Server is listening.
- Reverse Synchronization Server FQDN (variable): The FQDN of the Reverse Synchronization Server.

## 7. DHCP Behavior

### 7.1. DHCPv6 Server Behavior

Sections 17.2.2 and 18.2 of [RFC3315] govern server operation in regards to option assignment. As a convenience to the reader, we mention here that the server will send option foo only if configured with specific values for foo and if the client requested it. In particular, when configured the DHCP Server sends the Zone Template Option, Synchronization Server Option, Reverse Synchronization Server Option when requested by the DHCPv6 client by including necessary option codes in its ORO.

The DHCP Server may receive a Client Public Key Option (OPTION\_PUBLIC\_KEY) from the CPE. Upon receipt of this DHCPv6 option, the DHCP Server SHOULD acknowledge the reception of the Client Public Key Option as described in Section 4.1 and communicate this credential to the available DNS Servers like the DNS Template Server, the Synchronization Server and the Reverse Synchronization Server, unless not configured to do so.

A CPE may update its Client Public Key by sending a new value in the Client Public Key Option (OPTION\_PUBLIC\_KEY) as this document assumes the link between the CPE and the DHCP Server is considered authenticated and trusted. The server SHOULD process received Client Public Key Option sent by the client (see step 2 in Section 4.1), unless not configured to do so.

## 7.2. DHCPv6 Client Behavior

The DHCPv6 client SHOULD send a Client Public Key Option (OPTION\_PUBLIC\_KEY) to the DHCP Server. This Client Public Key authenticates the CPE.

The DHCPv6 client sends a ORO with the necessary option codes: Zone Template Option, Synchronization Server Option and Reverse Synchronization Server Option.

Upon receiving a DHCP option described in this document in the Reply message, the CPE SHOULD retrieve or update DNS zones using the associated Supported Authentication Methods and update protocols, as described in Section 5.

## 7.3. DHCPv6 Relay Agent Behavior

There are no additional requirements for the DHCP Relay agents.

## 8. IANA Considerations

The DHCP options detailed in this document is:

- OPTION\_DNS\_ZONE\_TEMPLATE: TBD1
- OPTION\_SYNC\_SERVER: TBD2
- OPTION\_REVERSE\_SYNC\_SERVER: TBD3

## 9. Security Considerations

### 9.1. DNSSEC is recommended to authenticate DNS hosted data

It is recommended that the (Reverse) Homenet Zone is signed with DNSSEC. The zone may be signed by the CPE or by a third party. We recommend the zone to be signed by the CPE, and that the signed zone is uploaded.

### 9.2. Channel between the CPE and ISP DHCP Server MUST be secured

The channel MUST be secured because the CPE provides authentication credentials. Unsecured channel may result in CPE impersonation attacks.

The document considers that the channel between the CPE and the ISP DHCP Server is trusted. More specifically, the CPE is authenticated and the exchanged messages are protected. The current document does not specify how to secure the channel. [RFC3315] proposes a DHCP authentication and message exchange protection, [RFC4301], [RFC7296] propose to secure the channel at the IP layer.

### 9.3. CPEs are sensitive to DoS

CPE have not been designed for handling heavy load. The CPE are exposed on the Internet, and their IP address is publicly published on the Internet via the DNS. This makes the Home Network sensitive to Deny of Service Attacks. The resulting outsourcing architecture is described in [I-D.ietf-homenet-front-end-naming-delegation]. This document shows how the outsourcing architecture can be automatically set.

## 10. Acknowledgments

We would like to thank Marcin Siodelski and Bernie Volz for their comments on the design of the DHCPv6 options. We would also like to thank Mark Andrews, Andrew Sullivan and Lorenzo Colliti for their remarks on the architecture design. The designed solution has been largely been inspired by Mark Andrews's document [I-D.andrews-dnsop-pd-reverse] as well as discussions with Mark. We also thank Ray Hunter for its reviews, its comments and for suggesting an appropriated terminology.

## 11. References

### 11.1. 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>>.
- [RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <<http://www.rfc-editor.org/info/rfc1996>>.
- [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>>.
- [RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <<http://www.rfc-editor.org/info/rfc2136>>.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, DOI 10.17487/RFC2181, July 1997, <<http://www.rfc-editor.org/info/rfc2181>>.
- [RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <<http://www.rfc-editor.org/info/rfc2845>>.
- [RFC2930] Eastlake 3rd, D., "Secret Key Establishment for DNS (TKEY RR)", RFC 2930, DOI 10.17487/RFC2930, September 2000, <<http://www.rfc-editor.org/info/rfc2930>>.
- [RFC2931] Eastlake 3rd, D., "DNS Request and Transaction Signatures ( SIG(0)s )", RFC 2931, DOI 10.17487/RFC2931, September 2000, <<http://www.rfc-editor.org/info/rfc2931>>.
- [RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, DOI 10.17487/RFC3007, November 2000, <<http://www.rfc-editor.org/info/rfc3007>>.
- [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>>.



- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <<http://www.rfc-editor.org/info/rfc4033>>.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <<http://www.rfc-editor.org/info/rfc4034>>.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <<http://www.rfc-editor.org/info/rfc4035>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<http://www.rfc-editor.org/info/rfc4301>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<http://www.rfc-editor.org/info/rfc5246>>.
- [RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <<http://www.rfc-editor.org/info/rfc5936>>.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<http://www.rfc-editor.org/info/rfc6347>>.
- [RFC6672] Rose, S. and W. Wijngaards, "DNAME Redirection in the DNS", RFC 6672, DOI 10.17487/RFC6672, June 2012, <<http://www.rfc-editor.org/info/rfc6672>>.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<http://www.rfc-editor.org/info/rfc7296>>.

## 11.2. Informational References

- [I-D.andrews-dnsop-pd-reverse]  
Andrews, M., "Automated Delegation of IP6.ARPA reverse zones with Prefix Delegation", draft-andrews-dnsop-pd-reverse-02 (work in progress), November 2013.

[I-D.ietf-dhc-sedhcpv6]

Jiang, S., Shen, S., Zhang, D., and T. Jinmei, "Secure DHCPv6", draft-ietf-dhc-sedhcpv6-08 (work in progress), June 2015.

[I-D.ietf-homenet-front-end-naming-delegation]

Migault, D., Weber, R., Hunter, R., Griffiths, C., and W. Cloetens, "Outsourcing Home Network Authoritative Naming Service", draft-ietf-homenet-front-end-naming-delegation-04 (work in progress), September 2015.

[I-D.sury-dnsextns-cname-dname]

Sury, O., "CNAME+DNAME Name Redirection", draft-sury-dnsextns-cname-dname-00 (work in progress), April 2010.

## Appendix A. Scenarios and impact on the End User

This section details various scenarios and discuss their impact on the end user.

### A.1. Base Scenario

The base scenario is the one described in Section 4. It is typically the one of an ISP that manages the DHCP Server, and all DNS servers.

The end user subscribes to the ISP (foo), and at subscription time registers for example.foo as its Registered Homenet Domain example.foo. Since the ISP knows the Registered Homenet Domain and the Public Authoritative Server(s) the ISP is able to build the Homenet Zone Template.

The ISP manages the DNS Template Server, so it is able to load the Homenet Zone Template on the DNS Template Server.

When the CPE is plugged (at least the first time), it provides its Client Public Key to the DHCP Server. In this scenario, the DHCP Server and the DNS Servers are managed by the ISP so the DHCP Server can provide authentication credentials of the CPE to enable secure authenticated transaction between the CPE and these DNS servers. More specifically, credentials are provided to:

- Synchronization Server
- Reverse Synchronization Server
- DNS Template Server

The CPE can update the zone using DNS update or a primary / secondary configuration in a secure way.

The main advantage of this scenario is that the naming architecture is configured automatically and transparently for the end user.

The drawbacks are that the end user uses a Registered Homenet Domain managed by the ISP and that it relies on the ISP naming infrastructure.

#### A.2. Third Party Registered Homenet Domain

This section considers the case when the end user wants its home network to use example.com as a Registered Homenet Domain instead of example.foo that has been assigned by the ISP. We also suppose that example.com is not managed by the ISP.

This can also be achieved without any configuration. When the end user buys the domain name example.com, it may request to redirect the name example.com to example.foo using static redirection with CNAME [RFC2181], [RFC1034], DNAME [RFC6672] or CNAME+DNAME [I-D.sury-dnsex-cname-dname].

This configuration is performed once when the domain name example.com is registered. The only information the end user needs to know is the domain name assigned by the ISP. Once this configuration is done no additional configuration is needed anymore. More specifically, the CPE may be changed, the zone can be updated as in Appendix A.1 without any additional configuration from the end user.

The main advantage of this scenario is that the end user benefits from the Zero Configuration of the Base Scenario Appendix A.1. Then, the end user is able to register for its home network an unlimited number of domain names provided by an unlimited number of different third party providers.

The drawback of this scenario may be that the end user still rely on the ISP naming infrastructure. Note that the only case this may be inconvenient is when the DNS Servers provided by the ISPs results in high latency.

#### A.3. Third Party DNS Infrastructure

This scenario considers that the end user uses example.com as a Registered Homenet Domain, and does not want to rely on the authoritative servers provided by the ISP.

In this section we limit the outsourcing to the Synchronization Server and Public Authoritative Server(s) to a third party. All other DNS Servers DNS Template Server, Reverse Public Authoritative Server(s) and Reverse Synchronization Server remain managed by the ISP. The reason we consider that Reverse Public Authoritative Server(s) and Reverse Synchronization Server remains managed by the ISP are that the prefix is managed by the ISP, so outsourcing these resources requires some redirection agreement with the ISP. More specifically the ISP will need to configure the redirection on one of its Reverse DNS Servers. That said, outsourcing these resources is similar as outsourcing Synchronization Server and Public Authoritative Server(s) to a third party. Similarly, the DNS Template Server can be easily outsourced as detailed in this section

Outsourcing Synchronization Server and Public Authoritative Server(s) requires:

- 1) Updating the Homenet Zone Template: this can be easily done as detailed in Section 4.3 as the DNS Template Server is still managed by the ISP. Such modification can be performed once by any CPE. Once this modification has been performed, the CPE can be changed, the Client Public Key of the CPE may be changed, this does not need to be done another time. One can imagine a GUI on the CPE asking the end user to fill the field with Registered Homenet Domain, optionally Public Authoritative Server(s), with a button "Configure Homenet Zone Template".
- 2) Updating the DHCP Server Information. In fact the Reverse Synchronization Server returned by the ISP is modified. One can imagine a GUI interface that enables the end user to modify its profile parameters. Again, this configuration update is done once-for-ever.
- 3) Upload the authentication credential of the CPE, that is the Client Public Key of the CPE, to the third party. Unless we use specific mechanisms, like communication between the DHCP Server and the third party, or a specific token that is plugged into the CPE, this operation is likely to be performed every time the CPE is changed, and every time the Client Public Key generated by the CPE is changed.

The main advantage of this scenario is that the DNS infrastructure is completely outsourced to the third party. Most likely the Client Public Key that authenticate the CPE need to be configured for every CPE. Configuration is expected to be CPE live-long.

#### A.4. Multiple ISPs

This scenario considers a CPE connected to multiple ISPs.

Firstly, suppose the CPE has been configured with the based scenarios exposed in Appendix A.1. The CPE has multiple interfaces, one for each ISP, and each of these interface is configured using DHCP. The CPE sends to each ISP its Client Public Key Option as well as a request for a Zone Template Option, a Synchronization Server Option and a Reverse Synchronization Server Option. Each ISP provides the requested DHCP options, with different values. Note that this scenario assumes, the home network has a different Registered Homenet Domain for each ISP as it is managed by the ISP. On the other hand, the CPE Client Public Key may be shared between the CPE and the multiple ISPs. The CPE builds the associate DNS(SEC) Homenet Zone, and proceeds to the various settings as described in Appendix A.1.

The protocol and DHCPv6 options described in this document are fully compatible with a CPE connected to multiple ISPs with multiple Registered Homenet Domains. However, the CPE should be able to handle different Registered Homenet Domains. This is an implementation issue which is outside the scope of the current document. More specifically, multiple Registered Homenet Domains leads to multiple DNS(SEC) Homenet Zones. A basic implementation may erase the DNS(SEC) Homenet Zone that exists when it receives DHCPv6 options, and rebuild everything from scratch. This will work for an initial configuration but comes with a few drawbacks. First, updates to the DNS(SEC) Homenet Zone may only push to one of the multiple Registered Homenet Domain, the latest Registered Homenet Domain that has been set, and this is most likely expected to be almost randomly chosen as it may depend on the latency on each ISP network at the boot time. As a results, this leads to unsynchronized Registered Homenet Domains. Secondly, if the CPE handles in some ways resolution, only the latest Registered Homenet Domain set may be able to provide naming resolution in case of network disruption.

Secondly, suppose the CPE is connected to multiple ISP with a single Registered Homenet Domain. In this case, the one party is chosen to host the Registered Homenet Domain. This entity may be one of the ISP or a third party. Note that having multiple ISPs can be motivated for bandwidth aggregation, or connectivity fail-over. In the case of connectivity fail-over, the fail-over concerns the access network and a failure of the access network may not impact the core network where the Synchronization Server and Public Authoritative Primaries are hosted. In that sense, choosing one of the ISP even in a scenario of multiple ISPs may make sense. However, for sake of simplicity, this scenario assumes that a third party has be chosen to host the Registered Homenet Domain. The DNS settings for each ISP is

described in Appendix A.2 and Appendix A.3. With the configuration described in Appendix A.2, the CPE is expect to be able to handle multiple Homenet Registered Domain, as the third party redirect to one of the ISPs Servers. With the configuration described in Appendix A.3, DNS zone are hosted and maintained by the third party. A single DNS(SEC) Homenet Zone is built and maintained by the CPE. This latter configuration is likely to match most CPE implementations.

The protocol and DHCPv6 options described in this document are fully compatible with a CPE connected to multiple ISPs. To configure or not and how to configure the CPE depends on the CPE facilities. Appendix A.1 and Appendix A.2 require the CPE to handle multiple Registered Homenet Domain, whereas Appendix A.3 does not have such requirement.

#### Appendix B. Document Change Log

[RFC Editor: This section is to be removed before publication]

-05: changing Master to Primary, Slave to Secondary

-04: Working Version Major modifications are:

- Re-structuring the draft: description and comparison of update and authentication methods have been integrated into the Overview section. a Configuration section has been created to describe both configuration and corresponding behavior of the CPE.
- Adding Ports parameters: Server Set can configure a port. The Port Server parameter have been added in the DHCPv6 option payloads because middle boxes may not be configured to let port 53 packets and it may also be useful to split servers among different ports, assigning each end user a different port.
- Multiple ISP scenario: In order to address comments, the multiple ISPs scenario has been described to explicitly show that the protocol and DHCPv6 options do not prevent a CPE connected to multiple independent ISPs.

-03: Working Version Major modifications are:

- Redesigning options/scope: according to feed backs received from the IETF89 presentation in the dhc WG.
- Redesigning architecture: according to feed backs received from the IETF89 presentation in the homenet WG, discussion with Mark and Lorenzo.

- 02: Working Version Major modifications are:
  - Redesigning options/scope: As suggested by Bernie Volz
- 01: Working Version Major modifications are:
  - Remove the DNS Zone file construction: As suggested by Bernie Volz
  - DHCPv6 Client behavior: Following options guide lines
  - DHCPv6 Server behavior: Following options guide lines
- 00: version published in the homenet WG. Major modifications are:
  - Reformatting of DHCPv6 options: Following options guide lines
  - DHCPv6 Client behavior: Following options guide lines
  - DHCPv6 Server behavior: Following options guide lines
- 00: First version published in dhc WG.

#### Authors' Addresses

Daniel Migault  
Ericsson  
8400 boulevard Decarie  
Montreal, QC H4P 2N2  
Canada

Email: [daniel.migault@ericsson.com](mailto:daniel.migault@ericsson.com)

Tomek Mrugalski  
Internet Systems Consortium, Inc.  
950 Charter Street  
Redwood City, CA 94063  
USA

Email: [tomasz.mrugalski@gmail.com](mailto:tomasz.mrugalski@gmail.com)  
URI: <http://www.isc.org>

Chris Griffiths

Email: [cgriffiths@gmail.com](mailto:cgriffiths@gmail.com)

Ralf Weber  
Nominum  
2000 Seaport Blvd #400  
Redwood City, CA 94063  
US

Email: [ralf.weber@nominum.com](mailto:ralf.weber@nominum.com)  
URI: <http://www.nominum.com>

Wouter Cloetens  
SoftAtHome  
vaartdijk 3 701  
3018 Wijkmaal  
Belgium

Email: [wouter.cloetens@softathome.com](mailto:wouter.cloetens@softathome.com)



Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: February 25, 2016

P. Pfister  
B. Paterson  
Cisco Systems  
J. Arkko  
Ericsson  
August 24, 2015

Distributed Prefix Assignment Algorithm  
draft-ietf-homenet-prefix-assignment-08

Abstract

This document specifies a distributed algorithm for dividing a set of prefixes in a manner that allows for automatic assignment of sub-prefixes that are unique and non-overlapping. Used in conjunction with a protocol that provides flooding of information among a set of participating nodes, prefix configuration within a network may be automated.

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 February 25, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Definitions . . . . .	3
2.1. Subroutine Specific Terminology . . . . .	5
3. Applicability Statement . . . . .	7
4. Algorithm Specification . . . . .	8
4.1. Prefix Assignment Algorithm Subroutine . . . . .	9
4.2. Overriding and Destroying Existing Assignments . . . . .	12
4.3. Other Events . . . . .	13
5. Prefix Selection Considerations . . . . .	13
6. Implementation Capabilities and Node Behavior . . . . .	16
7. Algorithm Parameters . . . . .	17
8. Security Considerations . . . . .	17
9. IANA Considerations . . . . .	18
10. Acknowledgments . . . . .	18
11. References . . . . .	18
11.1. Normative References . . . . .	18
11.2. Informative References . . . . .	18
Appendix A. Static Configuration Example . . . . .	19
Authors' Addresses . . . . .	20

## 1. Introduction

This document specifies a distributed algorithm for automatic prefix assignment. The algorithm provides a generic alternative to centralized (human or software based) approaches for network prefix and address assignment. Although it does not require to be configured to operate properly, it supports custom configuration by means of variable priority assignments, and can therefore be used in fully autonomic as well as configured networks. This document focuses on the algorithm itself and therefore leaves as out of scope context-specific considerations, such as the process of selecting a prefix value and length when making a new assignment.

The algorithm makes use of a flooding mechanism allowing participating Nodes to advertise prefixes assigned to the links they are directly connected to or for other purposes, e.g., for private assignment or prefix delegation. Advertising a prefix therefore serves two purposes. It is a claim that a prefix is in use, meaning that no other Node may advertise an overlapping prefix (unless it has a greater priority). And it is a way for other Nodes to know which prefixes have been assigned to the links they are directly connected to.

The algorithm is given a set of delegated prefixes, and ensures that the following assertions are satisfied after a finite convergence period:

1. At most one prefix from each delegated prefix is assigned to each link.
2. Assigned prefixes are non-overlapping (i.e., an assigned prefix never includes another assigned prefix).
3. Assigned prefixes do not change in the absence of topology or configuration changes.

In the rest of this document the two first conditions are referred to as the correctness conditions of the algorithm while the third condition is referred to as its convergence condition.

Each assignment has a priority specified by the Node making the assignment, allowing for custom assignment policies. When multiple Nodes assign different prefixes from the same delegated prefix to the same link, or when multiple Nodes assign overlapping prefixes (to the same link or to different links), the assignment with the greatest priority is kept and other assignments are removed.

The prefix assignment algorithm requires that participating Nodes share information through a flooding mechanism. If the flooding mechanism ensures that all messages are propagated to all Nodes within a given time window, the algorithm also ensures that all assigned prefixes used for networking operations (e.g., host configuration) remain unchanged, unless another Node assigns an overlapping prefix with a higher assignment priority, or the topology changes and renumbering cannot be avoided.

## 2. Definitions

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

This document makes use of the following terminology. The terms defined here are ordered in such a way as to try to avoid forward references, and therefore are not sorted alphabetically.

**Node:** An entity executing the algorithm specified in this document and able to communicate with other Nodes using the Flooding Mechanism.

**Flooding Mechanism:** A mechanism allowing participating Nodes to reliably share information with all other participating Nodes.

**Link:** An object the distributed algorithm will assign prefixes to. A Node may only assign prefixes to Links it is directly connected to. A Link is either Shared or Private.

**Shared Link:** A Link multiple Nodes may be connected to. Most of the time, a Shared Link is a multi-access link or point-to-point link, virtual or physical, requiring prefixes to be assigned to it.

**Private Link:** A Private Link is an abstract concept defined for the sake of this document. It allows Nodes to make assignments for their private use or delegation. For instance, every DHCPv6-PD [RFC3633] requesting router may be considered as a different Private Link.

**Delegated Prefix:** A prefix provided to the algorithm and used as a prefix pool for Assigned Prefixes.

**Node ID:** A value identifying a given participating Node. The set of identifiers MUST be strictly and totally ordered (e.g., using the alphanumeric order). The mechanism used to assign Node IDs, whether manual or automated, is out of the scope of this document.

**Flooding Delay:** A value which MUST be provided by the Flooding Mechanism and SHOULD be a deterministic or likely upper bound on the information propagation delay among participating Nodes.

**Advertised Prefix:** A prefix advertised by another Node and delivered to the local Node by the Flooding Mechanism. It has an Advertised Prefix Priority and, when assigned to a directly connected Shared Link, is associated with that Shared Link.

**Advertised Prefix Priority:** A value that defines the priority of an Advertised Prefix received from the Flooding Mechanism or a published Assigned Prefix. Whenever multiple Advertised Prefixes are conflicting (i.e., overlapping or from the same Delegated Prefix and assigned to the same link), all Advertised Prefixes but the one with the greatest priority will eventually be removed. In case of a tie, the assignment advertised by the Node with the greatest Node ID is kept and others are removed. In order to ensure convergence, the range of priority values MUST have an upper bound.

**Assigned Prefix:** A prefix included in a Delegated Prefix and assigned to a Shared or Private Link. It represents a local

decision to assign a given prefix from a given Delegated Prefix to a given Link. The algorithm ensures that there is never more than one Assigned Prefix per Delegated Prefix and Link pair. When destroyed, an Assigned Prefix is set as not applied, ceases to be advertised, and is removed from the set of Assigned Prefixes.

**Applied (Assigned Prefix):** When an Assigned Prefix is applied, it MAY be used (e.g., for host configuration, routing protocol configuration, prefix delegation). When not applied, it MUST NOT be used for any purpose outside of the prefix assignment algorithm. Each Assigned Prefix is associated with a timer (Apply Timer) used to apply the Assigned Prefix. An Assigned Prefix is unapplied when destroyed.

**Published (Assigned Prefix):** The Assigned Prefix is advertised through the Flooding Mechanism as assigned to its associated Link. A published Assigned Prefix MUST have an Advertised Prefix Priority. It will appear as an Advertised Prefix to other Nodes, once received from the Flooding Mechanism.

**Destroy (an Assigned Prefix):** Local action of removing an Assigned Prefix from the set of Assigned Prefixes. If applied, the prefix is unapplied. If published, the prefix stops being advertised through the Flooding Mechanism.

**Prefix Adoption:** When an Advertised Prefix which does not conflict with any other Advertised Prefix or published Assigned Prefix stops being advertised, any other Node connected to the same Link may, after some random delay, start advertising the same prefix. This procedure is called adoption and provides seamless assignment transfer from a Node to another, e.g., in case of Node failure.

**Backoff Timer:** Every Delegated Prefix and Link pair is associated with a timer counting down to zero. It is used to reduce the probability of colliding assignments made by multiple Nodes by delaying the creation of new Assigned Prefixes or the advertisement of adopted Assigned Prefixes by a random amount of time.

**Renumbering:** Event occurring when an Assigned Prefix which was applied is destroyed. Renumbering is undesirable as it usually implies reconfiguring routers or hosts.

## 2.1. Subroutine Specific Terminology

In addition to the terms defined in Section 2, the subroutine specified in Section 4 makes use of the following terms.

**Current Assignment:** For a given Delegated Prefix and Link, the Current Assignment is the Assigned Prefix (if any) included in the Delegated Prefix and assigned to the given Link by the Node executing the algorithm. At some point in time, the Current Assignment from different Nodes may differ, but the algorithm ensures that eventually, all Nodes directly connected to a Shared Link have the same Current Assignment for any given Delegated Prefix.

**Precedence:** An Advertised Prefix takes precedence over an Assigned Prefix if and only if one of the following conditions is met:

- \* The Assigned Prefix is not published.
- \* The Assigned Prefix is published and the Advertised Prefix Priority from the Advertised Prefix is strictly greater than the Advertised Prefix Priority from the Assigned Prefix.
- \* The Assigned Prefix is published, the priorities are identical, and the Node ID from the Node advertising the Advertised Prefix is strictly greater than the local Node ID.

**Best Assignment:** For a given Delegated Prefix and Link, the Best Assignment is computed as the unique Advertised Prefix (if any) that:

- \* Includes or is included in the Delegated Prefix (i.e., the Advertised Prefix is a sub-prefix of the Delegated Prefix, or the Delegated Prefix is a sub-prefix of the Advertised Prefix).
- \* Is assigned on the given Link.
- \* Has the greatest Advertised Prefix Priority among Advertised Prefixes fulfilling the two preceding conditions (and, in case of a tie, the prefix advertised by the Node with the greatest Node ID among all prefixes with greatest priority).
- \* Takes precedence over the Current Assignment associated with the same Link and Delegated Prefix (if any).

**Valid (Assigned Prefix):** An Assigned Prefix is valid if and only if the following two conditions are met:

- \* No Advertised Prefix including or included in the Assigned Prefix takes precedence over the Assigned Prefix.

- \* No Advertised Prefix including or included in the same Delegated Prefix as the Assigned Prefix and assigned to the same Link takes precedence over the Assigned Prefix.

### 3. Applicability Statement

Although the algorithm was primarily designed as an autonomic prefix assignment tool for home networks, it is applicable to other areas. In particular, it can operate without any kind of configuration as well as using advanced prefix assignment rules. Additionally, it can be applied to any address space and can be used to manage multiple address spaces simultaneously. For instance, an implementation can make use of IPv4-mapped IPv6 addresses [RFC4291] in order to manage both IPv4 and IPv6 prefix assignment using a single prefix space.

Each Node MUST have a set of non-overlapping Delegated Prefixes (i.e., which do not include each other). This set MAY change over time and be different from one Node to another at some point, but Nodes MUST eventually have the same set of non-overlapping Delegated Prefixes.

Given this set of non-overlapping Delegated Prefixes, Nodes may assign available prefixes from each Delegated Prefix to the Links they are directly connected to. The algorithm ensures that at most one prefix from a given Delegated Prefix is assigned to any given Link. Prefixes may also be assigned for private use. For example, an assigned prefix may be delegated to some other entity that does not implement this algorithm [RFC3633], or associated with a high priority in order to prevent other nodes from assigning any overlapping prefix [RFC6603].

The algorithm supports dynamically changing topologies and therefore will converge if the topology remains unmodified for a long enough period of time (That time depends on the Flooding Mechanism properties). Nevertheless, some topology changes may induce renumbering, while others do not. In particular, Nodes joining the set of participating Nodes do not cause renumbering. Similarly, Nodes leaving the network may be dealt with without renumbering by using the prefix adoption procedure. On the other hand, Links junction or split may break correctness conditions, and therefore cause renumbering.

All Nodes MUST run a common Flooding Mechanism in order to share published Assigned Prefixes. The set of participating Nodes is defined as the set of Nodes participating in the Flooding Mechanism.

The Flooding Mechanism MUST:

- o Provide a way to flood Assigned Prefixes assigned to a directly connected Link along with their respective Advertised Prefix Priority and the Node ID of the Node which is advertising them.
- o Specify whether an Advertised Prefix is assigned to a directly connected Shared Link, and if so, on which one. This information also needs to be updated in case of links junction or split.
- o Provide a Flooding Delay value, which SHOULD represent a deterministic or likely upper bound on the information propagation delay among participating Nodes. Whenever the Flooding Mechanism is unable to adhere to the provided Flooding Delay, renumbering may happen. As such a delay often depends on the size of the network, it MAY change over time and MAY be different from one Node to another. Furthermore, the process of selecting this value is subject to a tradeoff between convergence speed and lower renumbering probability (e.g., the value 0 may be used when renumbering is harmless), and is therefore out of scope of this document.

The algorithm ensures that whenever the Flooding Delay is provided and held, and in the absence of any topology change or Delegated Prefix removal, renumbering only happens when a Node deliberately overrides an existing assignment. In the absence of such deliberate override, the algorithm converges within an absolute worst case timespan of  $2 * \text{Flooding Delay} * L$  seconds, where  $L$  is the number of links.

Each Node MUST have a Node ID. In the absence of colliding assignments, the algorithm will not suffer from multiple nodes having the same Node ID, but in order for collisions to be resolved, such situation MUST be transient.

Finally, leaving the Flooding Mechanism or Node ID assignment process unsecured makes the network vulnerable to deny of service attacks, as detailed in Section 8. Additionally, as this algorithm requires all Nodes to know which Node has made which assignment, it may be unsuitable depending on privacy requirements among participating Nodes.

#### 4. Algorithm Specification

This section specifies the behavior of Nodes implementing the prefix assignment algorithm. The terms 'Current Assignment', 'Precedence', 'Best Assignment' and 'Valid' are used as defined in Section 2.1.



#### 4.1. Prefix Assignment Algorithm Subroutine

This section specifies the prefix assignment algorithm subroutine. It is defined for a given Delegated Prefix and Link pair and takes a `BackoffTriggered` boolean as parameter (indicating whether the subroutine execution was triggered by the Backoff Timer or by another event). The subroutine also makes use of the two following configuration parameters: `ADOPT_MAX_DELAY` and `BACKOFF_MAX_DELAY`, which meanings are detailed in Section 7.

For a given Delegated Prefix and Link pair, the subroutine **MUST** be run with the `BackoffTriggered` boolean set to false whenever:

- o An Advertised Prefix including or included in the considered Delegated Prefix is added or removed.
- o An Assigned Prefix included in the considered Delegated Prefix and associated with a different Link than the considered Link was destroyed, while there is no Current Assignment associated with the given pair. This case **MAY** be ignored if the creation of a new Assigned Prefix associated with the considered pair is not desired.
- o The considered Delegated Prefix is added.
- o The considered Link is added.
- o The Node ID is modified.
- o An Assigned Prefix included in the considered Delegated Prefix and associated with the considered Link is destroyed outside of the context of the subroutine, as specified in Section 4.2.

Furthermore, for a given Delegated Prefix and Link pair, the subroutine **MUST** be run with the `BackoffTriggered` boolean set to true whenever:

- o The Backoff Timer associated with the considered Delegated Prefix and Link pair fires while there is no Current Assignment associated with the given pair.

When such an event occurs, a Node **MAY** delay the execution of the subroutine instead of executing it immediately, e.g., while receiving an update from the Flooding Mechanism, or for security reasons (see Section 8). Even if other events occur in the meantime, the subroutine **MUST** be run only once. It is also assumed that, whenever one of these events is the Backoff Timer firing while there is no

Current Assignment associated with the given pair, the subroutine is executed with the BackoffTriggered boolean set to true.

In order to execute the subroutine for a given Delegated Prefix and Link pair, first get the Current Assignment and compute the Best Assignment associated with the Delegated Prefix and Link pair, then execute the corresponding case:

1. If there is no Best Assignment and no Current Assignment: Decide whether the creation of a new assignment for the given Delegated Prefix and Link pair is desired (As any result would be valid, the process of making this decision is out of the scope of this document) and do the following:

- \* If it is not desired, stop the execution of the subroutine.
- \* Else if the Backoff Timer is running, stop the execution of the subroutine.
- \* Else if the BackoffTriggered boolean is set to false, set the Backoff Timer to some random delay between ADOPT\_MAX\_DELAY and BACKOFF\_MAX\_DELAY (see Section 7) and stop the execution of the subroutine.
- \* Else, continue the execution of the subroutine.

Select a prefix for the new assignment (see Section 5 for guidance regarding prefix selection). This prefix MUST be included in or be equal to the considered Delegated Prefix and MUST NOT include or be included in any Advertised Prefix. If a suitable prefix is found, use it to create a new Assigned Prefix:

- \* Assigned to the considered Link.
- \* Set as not applied.
- \* The Apply Timer set to '2 \* Flooding Delay'.
- \* Published with some selected Advertised Prefix Priority.

2. If there is a Best Assignment but no Current Assignment: First check if the Best Assignment is equal to or included in the Delegated Prefix. If not, stop the execution of the subroutine. Otherwise, cancel the Backoff Timer and use the prefix from the Best Assignment to create a new Assigned Prefix:

- \* Assigned to the considered Link.

- \* Set as not applied.
  - \* The Apply Timer set to '2 \* Flooding Delay'.
  - \* Set as not published.
3. If there is a Current Assignment but no Best Assignment:
- \* If the Current Assignment is not valid, destroy it, and execute the subroutine again with the BackoffTriggered boolean set to false.
  - \* If the Current Assignment is valid and published, stop the execution of the subroutine.
  - \* If the Current Assignment is valid and not published, the Node MUST either:
    - + Adopt the prefix by canceling the Apply Timer and set the Backoff Timer to some random delay between 0 and ADOPT\_MAX\_DELAY (see Section 7). This procedure is used to avoid renumbering when the Node advertising the prefix left the Shared Link, and SHOULD therefore be preferred.
    - + Destroy it and go to case 1, allowing a different prefix to be assigned, or the prefix to be removed. When the Current Assignment is applied, this causes renumbering.
4. If there is a Current Assignment and a Best Assignment:
- \* Cancel the Backoff Timer.
  - \* If the two prefixes are identical, set the Current Assignment as not published. If the Current Assignment is not applied and the Apply Timer is not set, set the Apply Timer to '2 \* Flooding Delay'.
  - \* If the two prefixes are not identical, destroy the Current Assignment and go to case 2.

When the prefix assignment algorithm subroutine requires an assignment to be created or adopted, any Advertised Prefix Priority value can be used. Other documents MAY provide restrictions over this value depending on the context the algorithm is operating in, or leave it as implementation-specific.

#### 4.2. Overriding and Destroying Existing Assignments

In addition to the behaviors specified in Section 4.1, the following procedures MAY be used in order to provide additional behavior options (Section 6):

Overriding Existing Assignments: For any given Link and Delegated Prefix, a Node MAY create a new Assigned Prefix using a chosen prefix and Advertised Prefix Priority such that:

- \* The chosen prefix is included in or is equal to the considered Delegated Prefix.
- \* The Current Assignment, if any, as well as all existing Assigned Prefixes which include or are included inside the chosen prefix, are destroyed.
- \* It is not applied.
- \* The Apply Timer is set to ' $2 * \text{Flooding Delay}$ '.
- \* It is published.
- \* The Advertised Prefix Priority is greater than the Advertised Prefix Priority from all Advertised Prefixes which include or are included in the chosen prefix.
- \* The Advertised Prefix Priority is greater than the Advertised Prefix Priority from all Advertised Prefixes which include or are included in the considered Delegated Prefix and are assigned to the considered Link.

In order to ensure algorithm convergence:

- \* Such overriding assignments MUST NOT be created unless there was a change in the Node configuration, a Link was added, or an Advertised Prefix was added or removed.
- \* The chosen Advertised Prefix Priority for the new Assigned Prefix SHOULD be greater than all priorities from the destroyed Assigned Prefixes. If not, simple topologies with only two Nodes may not converge. Nodes which do not adhere to this rule MUST implement a mechanism which detects whether the distributed algorithm does not converge and, whenever this would happen, stop creating overriding Assigned Prefixes which do not adhere to this rule. The specifications for such safety procedures are out of the scope of this document.

Removing an Assigned Prefix: A Node MAY destroy any Assigned Prefix which is published. Such an event reflects the desire of a Node to not assign a prefix from a given Delegated Prefix to a given Link anymore. In order to ensure algorithm convergence, such a procedure MUST NOT be executed unless there was a change in the Node configuration. Furthermore, whenever an Assigned Prefix is destroyed in this way, the prefix assignment algorithm subroutine MUST be run for the Delegated Prefix and Link pair associated with the destroyed Assigned Prefix.

The two procedures specified in this section are OPTIONAL. They could be used for various purposes, e.g., for providing custom prefix assignment configuration or reacting to prefix space exhaustion (by overriding short Assigned Prefixes and assigning longer ones).

#### 4.3. Other Events

When the Apply Timer fires, the associated Assigned Prefix MUST be applied.

When the Backoff Timer associated with a given Delegated Prefix and Link pair fires while there is a Current Assignment associated with the same pair, the Current Assignment MUST be published with some associated Advertised Prefix Priority and, if the prefix is not applied, the Apply Timer MUST be set to  $2 * \text{Flooding Delay}$ .

When a Delegated Prefix is removed from the set of Delegated Prefixes (e.g., when the Delegated Prefix expires), all Assigned Prefixes included in the removed Delegated Prefix MUST be destroyed.

When one Delegated Prefix is replaced by another one that includes or is included in the deleted Delegated Prefix, all Assigned Prefixes which were included in the deleted Delegated Prefix but are not included in the added Delegated Prefix MUST be destroyed. Others MAY be kept.

When a Link is removed, all Assigned Prefixes assigned to that Link MUST be destroyed.

#### 5. Prefix Selection Considerations

When the prefix assignment algorithm subroutine specified in Section 4.1 requires a new prefix to be selected, the prefix MUST be selected either:

- o Among prefixes included in the considered Delegated Prefix which were previously assigned and applied on the considered Link. For

that purpose, Applied Prefixes may be stored in stable storage along with their associated Link.

- o Randomly, picked in a set of at least RANDOM\_SET\_SIZE (see Section 7) prefixes included in the considered Delegated Prefix and not including or included in any Assigned or Advertised Prefix. If less than RANDOM\_SET\_SIZE candidates are found, the prefix MUST be picked among all candidates.
- o Based on some custom selection process specified in the configuration.

A simple implementation MAY randomly pick the prefix among all available prefixes, but this strategy is inefficient in terms of address space use as a few long prefixes may exhaust the pool of available short prefixes.

The rest of this section describes a more efficient approach which MAY be applied any time a Node needs to pick a prefix for a new assignment. The two following definitions are used:

**Available prefix:** The prefix of the form Prefix/PrefixLength is available if and only if it satisfies the three following conditions:

- \* It is included in the considered Delegated Prefix.
- \* It does not include and is not included in any Assigned or Advertised Prefix.
- \* It is equal to the considered Delegated Prefix or Prefix/(PrefixLength-1) includes an Assigned or Advertised Prefix.

**Candidate prefix:** A prefix of desired length which is included in or is equal to an available prefix.

The procedure described in this section takes the three following criteria into account:

**Prefix Stability:** In some cases, it is desirable that the selected prefix should remain the same across executions and reboots. For this purpose, prefixes previously applied on the Link or pseudo-random prefixes generated based on Node- and Link-specific values may be considered.

**Randomness:** When no stored or pseudo-random prefix is chosen, a prefix may be randomly picked among RANDOM\_SET\_SIZE candidates of

desired length. If less than RANDOM\_SET\_SIZE candidates can be found, the prefix is picked among all candidates.

Addressing-space usage efficiency: In the process of assigning prefixes, a small set of badly chosen long prefixes may prevent any shorter prefix from being assigned. For this reason, the set of RANDOM\_SET\_SIZE candidates is created from available prefixes with longest prefix lengths and, in case of a tie, preferring numerically small prefix values.

When executing the procedure, do as follows:

1. For each prefix stored in stable storage, check if the prefix is included in or equal to an available prefix. If so, pick that prefix and stop.
2. For each prefix length, count the number of available prefixes of the given length.
3. If the desired prefix length was not specified, select one. The available prefixes count computed previously may be used to help pick a prefix length such that:
  - \* There is at least one candidate prefix.
  - \* The prefix length is chosen large enough to not exhaust the address space.

Let N be the chosen prefix length.

4. Iterate over available prefixes starting with prefixes of length N down to length 0 and create a set of RANDOM\_SET\_SIZE candidate prefixes of length exactly N included in or equal to available prefixes. The end goal here is to create a set of RANDOM\_SET\_SIZE candidate prefixes of length N included in a set of available prefixes of maximized prefix length. In case of a tie, smaller prefix values (as defined by the bit-wise lexicographical order) are preferred.
5. Generate a set of prefixes of desired length, which are pseudo-randomly chosen based on Node- and Link-specific values. For each pseudo-random prefix, check if the prefix is equal to a candidate prefix. If so, pick that prefix and stop.
6. Choose a random prefix from the set of selected candidates.

The complexity of this procedure is equivalent to the complexity of iterating over available prefixes. Such operation may be

accomplished in linear time, e.g., by storing Advertised and Assigned Prefixes in a binary trie.

## 6. Implementation Capabilities and Node Behavior

Implementations of the prefix assignment algorithm may vary from very basic to highly customizable, enabling different types of fully interoperable behaviors. The three following behaviors are given as examples:

**Listener:** The Node only acts upon assignments made by other Nodes, i.e, it never creates new assignments nor adopts existing ones. Such behavior does not require the implementation of the considerations specified in Section 5 or Section 4.2. The Node never checks the validity of existing assignments, which makes this behavior particularly suited to lightweight devices which can rely on more capable neighbors to make assignments on directly connected Shared Links.

**Basic:** The Node is capable of assigning new prefixes or adopting prefixes which do not conflict with any other existing assignment. Such behavior does not require the implementation of the considerations specified in Section 4.2. It is suited to situations where there is no preference over which prefix should be assigned to which Link, and there is no priority between different Links.

**Advanced:** The Node is capable of assigning new prefixes, adopting existing ones, making overriding assignments and destroying existing ones. Such behavior requires the implementation of the considerations specified in Section 5 and Section 4.2. It is suited when the administrator desires some particular prefix to be assigned on a given Link, or some Link to be assigned prefixes with a greater priority when there are not enough prefixes available for all Links.

Note that if all Nodes directly connected to some Link are listener Nodes or none of these Nodes are willing to make an assignment from a given Delegated Prefix to the given Link, no prefix from the given Delegated Prefix will ever be assigned to the Link. This situation may be detected by watching whether no prefix from a given Delegated Prefix has been assigned to the Link for longer than `BACKOFF_MAX_DELAY` plus the Flooding Delay.



## 7. Algorithm Parameters

This document does not provide values for `ADOPT_MAX_DELAY`, `BACKOFF_MAX_DELAY` and `RANDOM_SET_SIZE`. The algorithm ensures convergence and correctness for any chosen values, even when these are different from Node to Node. They MAY be adjusted depending on the context, providing a tradeoff between convergence time, efficient addressing, reduced control traffic (generated by the Flooding Mechanism), and low collision probability.

`ADOPT_MAX_DELAY` (respectively `BACKOFF_MAX_DELAY`) represents the maximum backoff time a Node may wait before adopting an assignment (respectively making a new assignment). `BACKOFF_MAX_DELAY` MUST be greater than or equal to `ADOPT_MAX_DELAY`. The greater `ADOPT_MAX_DELAY` and `(BACKOFF_MAX_DELAY - ADOPT_MAX_DELAY)`, the lower the collision probability and the lesser the amount of control traffic, but the greater the convergence time.

`RANDOM_SET_SIZE` represents the desired size of the set a random prefix will be picked from. The greater `RANDOM_SET_SIZE`, the better the convergence time and the lower the collision probability, but the worse the addressing-space usage efficiency.

## 8. Security Considerations

The prefix assignment algorithm functions on top of two distinct mechanisms, the Flooding Mechanism and the Node ID assignment mechanism.

An attacker able to publish Advertised Prefixes through the Flooding Mechanism may perform the following attacks:

- \* Publish a single overriding assignment for a whole Delegated Prefix or for the whole address space, thus preventing any Node from assigning prefixes to Links.
- \* Quickly publish and remove Advertised Prefixes, generating traffic at the Flooding Mechanism layer and causing multiple executions of the prefix assignment algorithm in all participating Nodes.
- \* Publish and remove Advertised Prefixes in order to prevent the convergence of the algorithm.

An attacker able to prevent other Nodes from accessing a portion or the whole set of Advertised Prefixes may compromise the correctness of the algorithm.

An attacker able to cause repetitive Node ID changes may cause traffic to be generated in the Flooding Mechanism and multiple executions of the prefix assignment algorithm in all participating Nodes.

An attacker able to publish Advertised Prefixes using a Node ID used by another Node may impede the ability to resolve prefix assignment collisions.

Whenever the security of the Flooding Mechanism and Node ID assignment mechanism cannot be ensured, the convergence of the algorithm may be prevented. In environments where such attacks may be performed, the execution of the prefix assignment algorithm subroutine SHOULD be rate limited, as specified in Section 4.1.

## 9. IANA Considerations

This document has no actions for IANA.

## 10. Acknowledgments

The authors would like to thank those who participated in the previous document's version development as well as the present one. In particular, the authors would like to thank Tim Chown, Fred Baker, Mark Townsley, Lorenzo Colitti, Ole Troan, Ray Bellis, Markus Stenberg, Wassim Haddad, Joel Halpern, Samita Chakrabarti, Michael Richardson, Anders Brandt, Erik Nordmark, Laurent Toutain, Ralph Droms, Acee Lindem, Steven Barth and Juliusz Chroboczek for interesting discussions and document review.

## 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

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

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC6603] Korhonen, J., Savolainen, T., Krishnan, S., and O. Troan, "Prefix Exclude Option for DHCPv6-based Prefix Delegation", RFC 6603, May 2012.

#### Appendix A. Static Configuration Example

This section describes an example of how custom configuration of the prefix assignment algorithm may be implemented.

The Node configuration is specified as a finite set of rules. A rule is defined as:

- o A prefix to be used.
- o A Link on which the prefix may be assigned.
- o An Assigned Prefix Priority (smallest possible Assigned Prefix Priority if the rule may not override other Assigned Prefixes).
- o A rule priority (0 if the rule may not override existing Advertised Prefixes).

In order to ensure the convergence of the algorithm, the Assigned Prefix Priority MUST be an increasing function (not necessarily strictly) of the configuration rule priority (i.e., the greater is the configuration rule priority, the greater the Assigned Prefix Priority must be).

Each Assigned Prefix is associated with a rule priority. Assigned Prefixes which are created as specified in Section 4.1 are given a rule priority of 0.

Whenever the configuration is changed or the prefix assignment algorithm subroutine is run: For each Link/Delegated Prefix pair, look for the configuration rule with the greatest configuration rule priority such that:

- o The prefix specified in the configuration rule is included in the considered Delegated Prefix.
- o The Link specified in the configuration rule is the considered Link.
- o All the Assigned Prefixes which would need to be destroyed in case a new Assigned Prefix is created from that configuration rule (as specified in Section 4.2) have an associated rule priority which is strictly lower than the one of the considered configuration rule.

- o The assignment would be valid when published with an Advertised Prefix Priority equal to the one specified in the configuration rule.

If a rule is found, a new Assigned Prefix is created based on that rule as specified in Section 4.2. The new Assigned Prefix is associated with the Advertised Prefix Priority and the rule priority specified in the considered configuration rule.

Note that the use of rule priorities ensures the convergence of the algorithm.

#### Authors' Addresses

Pierre Pfister  
Cisco Systems  
Paris  
France

Email: pierre.pfister@darou.fr

Benjamin Paterson  
Cisco Systems  
Paris  
France

Email: paterson.b@gmail.com

Jari Arkko  
Ericsson  
Jorvas 02420  
Finland

Email: jari.arkko@piuha.net

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

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
2.1. Requirements Language . . . . .	3
3. PVD information discovery using DNS . . . . .	3
3.1. PVD TXT Record Fomat . . . . .	4
3.2. PVD TXT Record Keys . . . . .	4
3.2.1. Reachable Services . . . . .	4
3.2.2. DNS Configuration . . . . .	5
3.2.3. Connectivity Characteristics . . . . .	5
3.2.4. Private Extensions . . . . .	6
4. Special-purpose IPv6 prefixes . . . . .	6
4.1. Extensions to Stateless Address Autoconfiguration . . . . .	6
4.2. Extensions to DHCPv6 . . . . .	7
5. Security Considerations . . . . .	9
6. IANA Considerations . . . . .	9
7. References . . . . .	10
7.1. Normative references . . . . .	10
7.2. Informative references . . . . .	11
Appendix A. This solution compared to doing this in DHCPv6/NDP [RFC                      Editor: please remove] . . . . .	11
Appendix B. Discussion Points [RFC Editor: please remove] . . . . .	12
Appendix C. Changelog [RFC Editor: please remove] . . . . .	13
Appendix D. Draft Source [RFC Editor: please remove] . . . . .	13
Appendix E. Acknowledgements . . . . .	13
Authors' Addresses . . . . .	13

## 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 inadvertently 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.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.2.3.4.5.6.7.8.8.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.0.0.0.0.0.0.0.0.0.0.0.0.0.4.3.2.1.1.2.3.4.5.6.7.8.8.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=Fooobar 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 or 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".



defined to be added as a suboption to OPTION\_IAADDR and OPTION\_IAPREFIX options.

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| OPTION_SPECIAL_PURPOSE (TBD) |                               0 |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

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:

- o Traffic redirection, both by providing custom DNS server, as well as actual potentially different next-hop and/or source address selection.
- o 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 the existing solutions in terms of (highly subjective) pros and cons. The authors consider pros of this proposal to be:

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

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

The authors consider cons of this proposal to be:

- o 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).
- o 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]

- o 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.
- o 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.
- o 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:

- o 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