Network Working Group
Internet-Draft
Intended status: Informational
Expires: March 26, 2015

T. Lemon
Nominum, Inc.
T. Mrugalski
Internet Systems Consortium, Inc.
September 22, 2014

# Customizing DHCP Configuration on the Basis of Network Topology
## draft-ietf-dhc-topo-conf-03

Abstract

DHCP servers have evolved over the years to provide significant functionality beyond that which is described in the DHCP base specifications. One aspect of this functionality is support for context-specific configuration information. This memo describes some such features and makes recommendations as to how they can be used.

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, 2015.

Table of Contents



## 1. Introduction

The DHCPv4 [RFC2131] and DHCPv6 [RFC3315] protocol specifications describe how addresses can be allocated to clients based on network topology information provided by the DHCP relay infrastructure. Address allocation decisions are integral to the allocation of addresses and prefixes in DHCP.

The DHCP protocol also describes mechanisms for provisioning devices with additional configuration information; for example, DNS [RFC1034] server addresses, default DNS search domains, and similar information.

Although it was the intent of the authors of these specifications that DHCP servers would provision devices with configuration information appropriate to each device's location on the network, this practice was never documented, much less described in detail.

Existing DHCP server implementations do in fact provide such capabilities; the goal of this document is to describe those capabilities for the benefit both of operators and of protocol designers who may wish to use DHCP as a means for configuring their own services, but may not be aware of the capabilities provided by most modern DHCP servers.

## 2. Terminology

- Routable IP address: an IP address with a scope of use wider than the local link.
- PE router: Provider Edge Router. The provider router closest to the customer.
- CPE device: customer premise equipment device. Typically a router belonging to the customer that connects directly to the provider link.
- Shared subnet: a case where two or more subnets of the same protocol family are available on the same link. 'Share subnet' terminology is typically used in Unix environments. It is typically called 'multinet' in Windows environment. The administrative configuration inside a Microsoft DHCP server is called 'DHCP Superscope'.

## 3. Locality

Figure 1 illustrates a simple hierarchy of network links with Link D serving as a backbone to which the DHCP server is attached.

Figure 2 illustrates a more complex case. Although some of its aspects are unlikely to be seen in an actual production networks, they are beneficial for explaining finer aspects of the DHCP protocols. Note that some nodes act as routers (which forward all IPv6 traffic) and some are relay agents (i.e. run DHCPv6 specific software that forwards only DHCPv6 traffic).

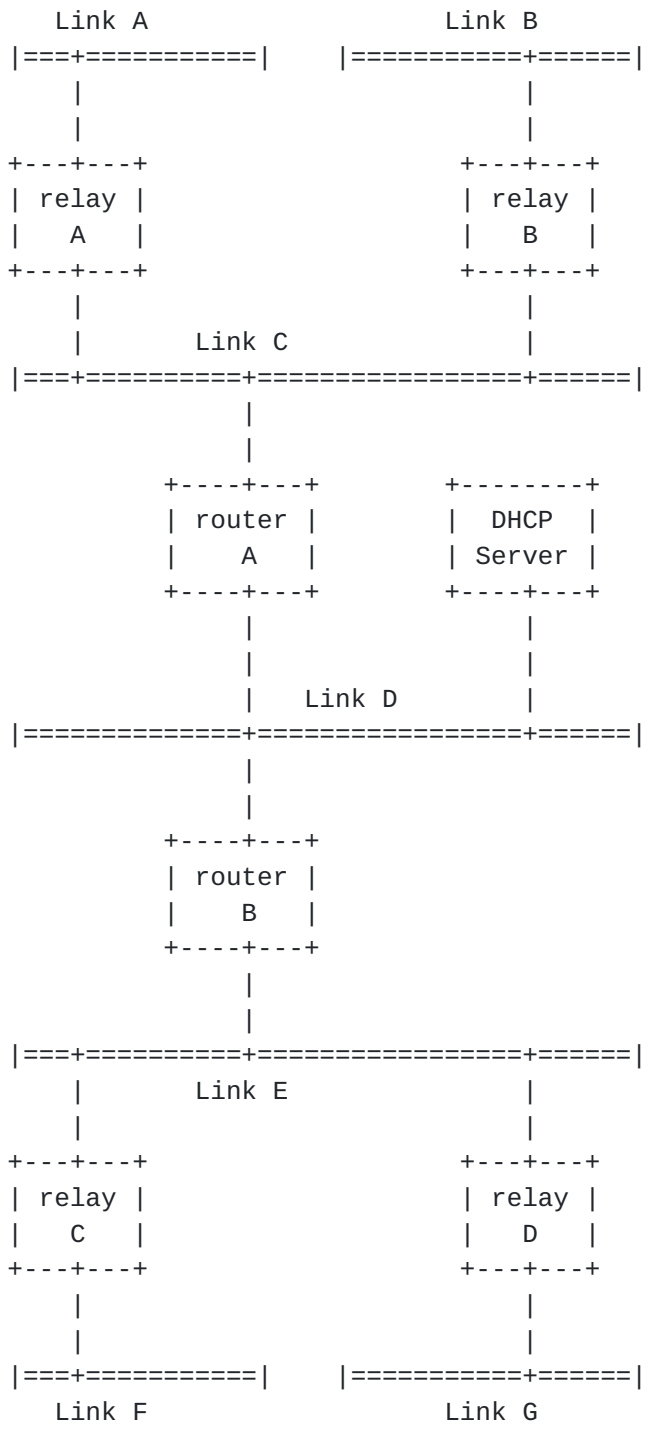


Figure 1: A simple network

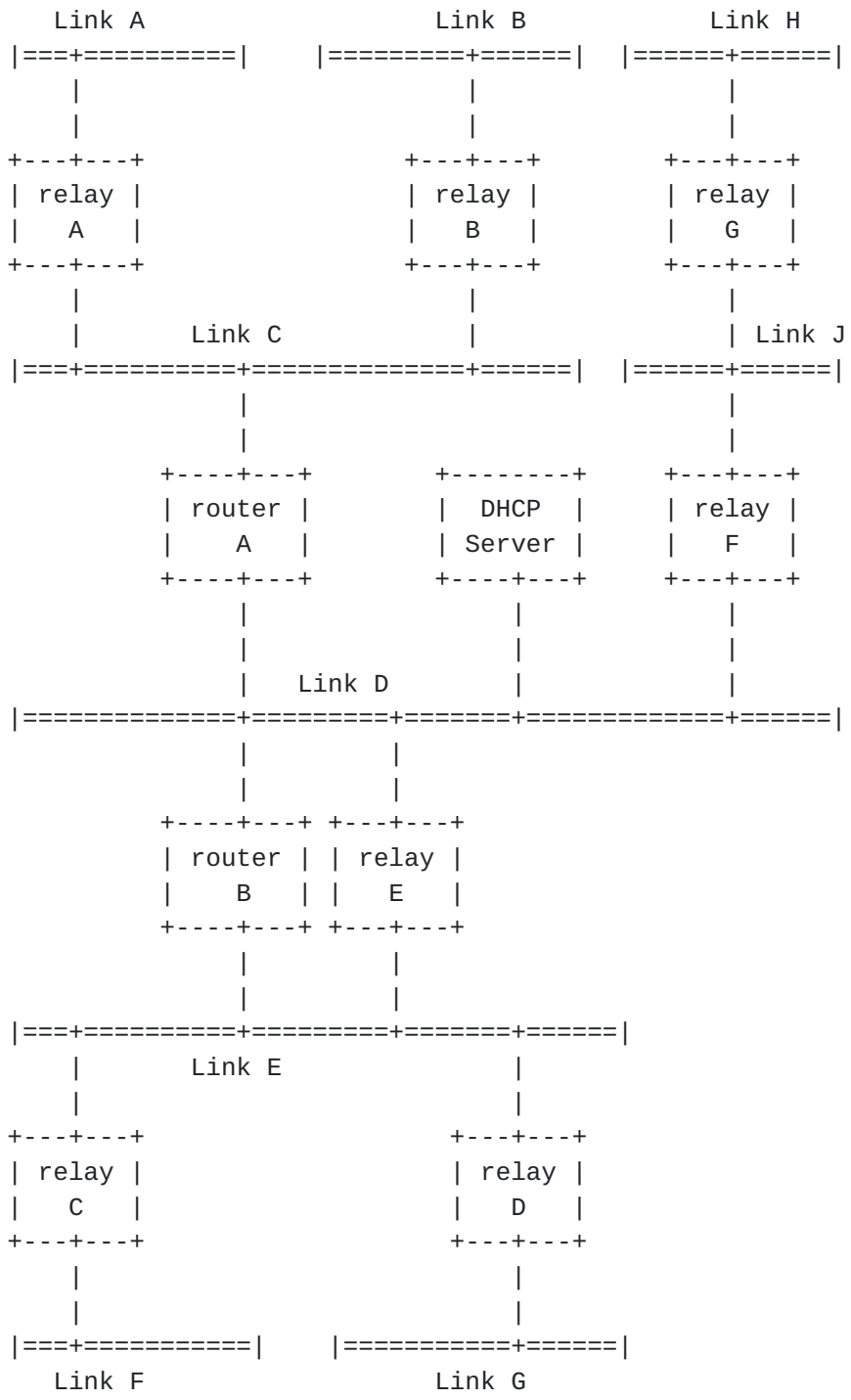


Figure 2: Complex network

This diagram allows us to represent a variety of different network configurations and illustrate how existing DHCP servers can provide configuration information customized to the particular location from which a client is making its request.

It's important to understand the background of how DHCP works when considering this diagram. DHCP clients are assumed not to have routable IP addresses when they are attempting to obtain configuration information.

The reason for making this assumption is that one of the functions of DHCP is to bootstrap the DHCP client's IP address configuration; if the client does not yet have an IP address configured, it cannot route packets to an off-link DHCP server, therefore some kind of relay mechanism is required.

The details of how packet delivery between clients and servers works are different between DHCPv4 and DHCPv6, but the essence is the same: whether or not the client actually has an IP configuration, it generally communicates with the DHCP server by sending its requests to a DHCP relay agent on the local link; this relay agent, which has a routable IP address, then forwards the DHCP requests to the DHCP server. In some cases in DHCPv4, when a DHCP client has a routable IPv4 address, the message is unicast to the DHCP server rather than going through a relay agent. In DHCPv6 that is also possible in case where the server is configured with a Server Unicast option (see Section 22.12 in [RFC3315]) and clients are able to take advantage of it. In such case once the clients get their (presumably global) addresses, they are able to contact server directly, bypassing relays. It should be noted that such a mode is completely controllable by administrators in DHCPv6. (They may simply choose to not configure server unicast option, thus forcing clients to send their messages always via relay agents).

In all cases, the DHCP server is able to obtain an IP address that it knows is on-link for the link to which the DHCP client is connected: either the DHCPv4 client's routable IPv4 address, or the relay agent's IPv4 address on the link to which the client is connected. So in every case the server is able to determine the client's point of attachment and select appropriate subnet- or link-specific configuration.

In the DHCPv6 protocol, there are two mechanisms defined in [RFC3315] that allow server to distinguish which link the relay agent is connected to. The first mechanism is a link-address field in the RELAY-FORW and RELAY-REPL messages. Somewhat contrary to its name, relay agents insert in the link-address field an address that is typically global and can be used to uniquely identify the link on which the client is located. In normal circumstances this is the solution that is easiest to maintain. It requires, however, for the relay agent to have an address with a scope larger than link-local configured on its client-facing interface. If for whatever reason that is not feasible (e.g. because the relay agent does not have a

global address configured on the client-facing interface), the relay agent includes an Interface-Id option (see Section 22.18 of [RFC3315]) that identifies the link clients are connected to. It is up to administrator to make sure that the interface-id is unique within his administrative domain. It should be noted that RELAY-FORW and RELAY-REPL messages are exchanged between relays and servers only. Clients are never exposed to those messages. Also, servers never receive RELAY-REPL messages. Relay agents must be able to process both RELAY-FORW (sending already relayed message further towards the server, when there is more than one relay agent in a chain) and RELAY-REPL (when sending back the response towards the client, when there is more than one relay agent in a chain).

For completeless, we also mention an uncommon, but valid case, where relay agents set link-local address in the link-address field in relayed RELAY-FORW messages. This may happen if the relay agent doesn't have any address with a larger scope. Even though link local addresses can't be automatically used to associate relay agent with a given link, with sufficient information provided the server is still able to correctly select proper link. That requires the DHCP server software to be able to specify relay agent link-address or a feature similar to 'shared subnets' (see Section 9). Network administrator has to manually configure additional information that a given subnet uses a relay agent with link-address X. Alternatively, if the relay agent uses link address X and relays messages from a subnet A, an administrator can configure that subnet A is a shared subnet with a very small X/128 subnet. That is not a recommended configuration, but in cases where it is impossible for relay agents to get an address from the subnet they are relaying from, it may be a viable solution.

DHCPv6 also has support for more finely grained link identification, using Lightweight DHCPv6 Relay Agents [RFC6221] (LDRA). In this case, in addition to receiving an IPv6 address that is on-link for the link to which the client is connected, the DHCPv6 server also receives an Interface-Id option from the relay agent that can be used to more precisely identify the client's location on the network.

What this means in practice is that the DHCP server in all cases has sufficient information to pinpoint, at the very least, the layer 3 link to which the client is connected, and in some cases which layer 2 link the client is connected to, when the layer 3 link is aggregated out of multiple layer 2 links.

In all cases, then, the DHCP server will have a link-identifying IP address, and in some cases it may also have a link-specific identifier (e.g. Interface-Id Option or Link Address Option defined in Section 5 of [RFC6977]). It should be noted that there is no

guarantee that the link-specific identifier will be unique outside the scope of the link-identifying IP address.

It is also possible for link-specific identifiers to be nested, so that the actual identifier that identifies the link is an aggregate of two or more link-specific identifiers sent by a set of LDRAs in a chain; in general this functions exactly as if a single identifier were received from a single LDRA, so we do not treat it specially in the discussion below, but sites that use chained LDRA configurations will need to be aware of this when configuring their DHCP servers.

So let's examine the implications of this in terms of how a DHCP server can deliver targeted supplemental configuration information to DHCP clients.

## 4. Simple Subnetted Network

Consider Figure 1 in the context of a simple subnetted network. In this network, there are four leaf subnets: links A, B, F and G, on which DHCP clients will be configured. Relays A, B, C and D in this example are represented in the diagram as IP routers with an embedded relay function, because this is a very typical configuration, but the relay function can also be provided in a separate node on each link.

In a simple network like this, there may be no need for link-specific configuration in DHCPv6, since local routing information is delivered through router advertisements. However, in IPv4, it is very typical to configure the default route using DHCP; in this case, the default route will be different on each link. In order to accomplish this, the DHCP server will need link-specific configuration for the default route.

To illustrate, we will use an example from a hypothetical DHCP server that uses a simple JSON notation for configuration. Although we know of no DHCP server that uses this specific syntax, most modern DHCP server provides similar functionality.

```
{
    "prefixes": {
        "192.0.2.0/26": {
            "options": {
                "routers": ["192.0.2.1"]
            },
        "on-link": ["a"]
        },
        "192.0.2.64/26": {
            "options": {
                "routers": ["192.0.2.65"]
            },
       "on-link": ["b"]
        },
        "192.0.2.128/26": {
            "options": {
                "routers": ["192.0.2.129"]
            },
       "on-link": ["f"]
        },
        "192.0.2.192/26": {
            "options": {
                "routers": ["192.0.2.193"]
            },
       "on-link": ["g"]
        }
    }
}
```

Figure 3: Configuration example

In Figure 3, we see a configuration example for this scenario: a set of prefixes, each of which has a set of options and a list of links for which it is on-link. We have defined one option for each prefix: a routers option. This option contains a list of values; each list only has one value, and that value is the IP address of the router specific to the prefix.

When the DHCP server receives a request, it searches the list of prefixes for one that encloses the link-identifying IP address provided by the client or relay agent. The DHCP server then examines the options list associated with that prefix and returns those options to the client.

So for example a client connected to link A in the example would have a link-identifying IP address within the 192.0.2.0/26 prefix, so the DHCP server would match it to that prefix. Based on the configuration, the DHCP server would then return a routers option

containing a single IP address: 192.0.2.1. A client on link F would have a link-identifying address in the 192.0.2.128/26 prefix, and would receive a routers option containing the IP address 192.0.2.129.

## 5. Relay agent running on a host

Relay agent is a DHCP software that may be run on any IP node. Although it is typically run on a router, this is by no means required by the DHCP protocol. The relay agent is simply a service that operates on a link, receiving link-local multicasts or broadcasts and relaying them, using IP routing, to a DHCP server. As long as the relay has an IP address on the link, and a default route or more specific route through which it can reach a DHCP server, it need not be a router, or even have multiple interfaces.

Relay agent can be run on a host connected to two links. That case is presented in Figure 2. There is router B that is connected to links D and E. At the same time there is also a host that is connected to the same links. The relay agent software is running on that host. That is uncommon, but legal configuration.

## 6. Cascade relays

Let's observe another case shown in Figure 2. Note that in typical configuration, the clients connected to link G will send their requests to relay D which will forward its packets directly to the DHCP server. That is typical, but not the only possible configuration. It is possible to configure relay agent D to forward client messages to relay E which in turn will send it to the DHCP server. This configuration is sometimes referred to as cascade relay agents.

Note that the relaying mechanism works differently in DHCPv4 and in DHCPv6. In DHCPv4 only the first relay is able to set the GIADDR field in the DHCPv4 packet. Any following relays that receive that packet will not change it as the server needs GIADDR information from the first relay (i.e. the closest to the client). Server will send the response back to the GIADDR address, which is the address of the first relay agent that saw the client's message. That means that the client messages travel on a different path than the server's responses. A message from client connected to link G will travel via relay D, relay E and to the server. A response message will be sent from the server to relay D via router B, and relay D will send it to the client on link G.

Relaying in DHCPv6 is more structured. Each relay agent encapsulates a packet that is destined to the server and sends it towards the server. Depending on the configuration that can be server's unicast

address, a multicast address or next relay agent address. The next relay repeats the encapsulation process. Although the resulting packet is more complex (may have up to 32 levels of encapsulation if traveled through 32 relays), every relay may insert its own options and it is clear which relay agent inserted which option.

## 7. Regional Configuration Example

In this example, link C is a regional backbone for an ISP. Link E is also a regional backbone for that ISP. Relays A, B, C and D are PE routers, and Links A, B, F and G are actually link aggregators with individual layer 2 circuits to each customer--for example, the relays might be DSLAMs or cable head-end systems. At each customer site we assume there is a single CPE device attached to the link.

We further assume that links A, B, F and G are each addressed by a single prefix, although it would be equally valid for each CPE device to be numbered on a separate prefix.

In a real-world deployment, there would likely be many more than two PE routers connected to each regional backbone; we have kept the number small for simplicity.

In the example presented in Figure 4, the goal is to configure all the devices within a region with server addresses local to that region, so that service traffic does not have to be routed between regions unnecessarily.

```
{
    "prefixes": {
        "2001:db8:0:0::/40": {
            "on-link": ["A"]
        },
        "2001:db8:100:0::/40": {
            "on-link": ["B"]
        },
        "2001:db8:200:0::/40": {
            "on-link": ["F"]
        },
        "2001:db8:300:0::/40": {
            "on-link": ["G"]
        }
    },
    "links": {
        "A": {"region": "omashu"},
        "B": {"region": "omashu"},
        "F": {"region": "gaoling"},
        "G": {"region": "gaoling"}
    },
   "regions": {
       "omashu": {
           "options": {
               "sip-servers": ["sip.omashu.example.org"],
               "dns-servers": ["dns1.omashu.example.org",
                               "dns2.omashu.example.org"]
           }
       },
       "gaoling": {
           "options": {
               "sip-servers": ["sip.gaoling.example.org"],
               "dns-servers": ["dns1.gaoling.example.org",
                               "dns2.gaoling.example.org"]
           }
        }
    }
}
```

Figure 4: An example regions configuration

In this example, when a request comes in to the DHCP server with a link-identifying IP address in the 2001:DB8:0:0::/40 prefix, it is identified as being on link A. The DHCP server then looks on the list of links to see what region the client is in. Link A is identified as being in omashu. The DHCP server then looks up omashu in the set of regions, and discovers a list of region-specific options.

The DHCP server then resolves the domain names listed in the options and sends a sip-server option containing the IP addresses that the resolver returned for sip.omashu.example.org, and a dns-server option containing the IP addresses returned by the resolver for dns1.omashu.example.org and dns2.omashu.example.org. Depending on the server capability and configuration, it may cache resolved responses for specific period of time, repeat queries every time or even keep the response until reconfiguration or shutdown.

Similarly, if the DHCP server receives a request from a DHCP client where the link-identifying IP address is contained by the prefix 2001:DB8:300:0::/40, then the DHCP server identifies the client as being connected to link G. The DHCP server then identifies link G as being in the gaoling region, and returns the sip-servers and dns-servers options specific to that region.

As with the previous example, the exact configuration syntax and structure shown above does not precisely match what existing DHCP servers do, but the behavior illustrated in this example can be accomplished with most existing modern DHCP servers.

## 8. Dynamic Lookup

In the Regional example, the configuration listed several domain names as values for the sip-servers and dns-servers options. The wire format of both of these options contains one or more IPv6 addresses--there is no way to return a domain name to the client.

This was understood to be an issue when the original DHCP protocol was defined, and historical implementations even from the very early days would accept domain names and resolve them. Some early DHCP implementations, particularly those based on earlier BOOTP implementations, had very limited capacity for reconfiguration.

However, most modern DHCP servers handle name resolution by querying the resolver each time a DHCP packet comes in. This means that if DHCP servers and DNS servers are managed by different administrative entities, there is no need for the administrators of the DHCP servers and DNS servers to communicate when changes are made. When changes are made to the DNS server, these changes are promptly and automatically adopted by the DHCP server. Similarly, when DHCP server configurations change, DNS server administrators need not be aware of this.

However, it should be noted that even though the DHCP server may be configured to query the DNS server every time it uses configured names, the changes made in the DNS zone may not be visible to the server until the DNS cache expires. If this is not desired, the DHCP

server can be configured to query the authoritative DNS server directly, bypassing any caching DNS servers.

It's worth noting that DNS is not the only way to resolve names, and not all DHCP servers support other techniques (e.g., NIS+ or WINS). However, since these protocols have all but vanished from common use, this won't be an issue in new deployments.

## 9. Multiple subnets on the same link

There are scenarios where there is more than one subnet from the same protocol family (i.e. two or more IPv4 subnets or two or more IPv6 subnets) configured on the same layer 3 link. One example is a slow network renumbering where some services are migrated to the new addressing scheme, but some aren't yet. Second example is a cable network, where cable modems and the devices connected behind them are connected to the same layer 2 link. However, operators want the cable modems and user devices to get addresses from distinct address spaces, so users couldn't easily access their modems management interfaces. Such a configuration is often referred to as 'shared subnets' in Unix environments or 'multinet' in Microsoft terminology.

To support such an configuration, additional differentiating information is required. Many DHCP server implementations offer a feature that is typically called client classification. The server segregates incoming packets into one or more classes based on certain packet characteristics, e.g. presence or value of certains options or even a match between existing options. Servers require additional information to handle such configuration, as it can't use the topographical property of the relay addresses alone to properly choose a subnet. Such information is always implementation specific.

## 10. Acknowledgments

Thanks to Dave Thaler for suggesting that even though "everybody knows" how DHCP servers are deployed in the real world, it might be worthwhile to have an IETF document that explains what everybody knows, because in reality not everybody is an expert in how DHCP servers are administered. Thanks to Andre Kostur, Carsten Strotmann, Simon Perreault, Jinmei Tatuya and Suresh Krishnan for their reviews, comments and feedback.

## 11. Security Considerations

This document explains existing practice with respect to the use of Dynamic Host Configuration Protocol [RFC2131] and Dynamic Host Configuration Protocol Version 6 [RFC3315]. The security considerations for these protocols are described in their

specifications and in related documents that extend these protocols. This document introduces no new functionality, and hence no new security considerations.

## 12. IANA Considerations

The IANA is hereby absolved of any requirement to take any action in relation to this document.

## 13. References

### 13.1. Normative References

[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.

### 13.2. Informative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC6221] Miles, D., Ooghe, S., Dec, W., Krishnan, S., and A. Kavanagh, "Lightweight DHCPv6 Relay Agent", RFC 6221, May 2011.

[RFC6977] Boucadair, M. and X. Pougnard, "Triggering DHCPv6 Reconfiguration from Relay Agents", RFC 6977, July 2013.

## Authors' Addresses

Ted Lemon
Nominum, Inc.
2000 Seaport Blvd
Redwood City, CA 94063
USA

Phone: +1-650-381-6000
Email: Ted.Lemon@nominum.com

Tomek Mrugalski
Internet Systems Consortium, Inc.
950 Charter Street
Redwood City, CA 94063
USA

Phone: +1 650 423 1345
Email: tomasz.mrugalski@gmail.com