Internet Engineering Task Force J. Bound
INTERNET DRAFT Compaq Computer Corp.
DHC Working Group M. Carney
Obsoletes: draft-ietf-dhc-dhcpv6exts-11.txt Sun Microsystems, Inc
C. Perkins
Nokia Research Center
5 May 2000

Extensions for the Dynamic Host Configuration Protocol for IPv6
draft-ietf-dhc-dhcpv6exts-12.txt

Status of This Memo

Abstract

The Dynamic Host Configuration Protocol for IPv6 [4] (DHCP) provides a framework for passing configuration information to hosts on a TCP/IP network. Configuration parameters and other control information are carried in typed data items that are stored in the ``extensions'' field of the DHCP message. The data items themselves are also called ``extensions.'' This document specifies the initial set of DHCP extensions, which will be periodically updated as new extensions are defined until this document reaches proposed standard.

After that time, individual extensions will be defined in separate documents, to be reviewed by the DHCP WG (if it still exists) and the IESG.

Contents

1. Introduction

This document specifies extensions for use with the Dynamic Host

Configuration Protocol for IP version 6 (DHCP). The DHCP message formats are described in the DHCP protocol document [4]. In this document, several words are used to signify the requirements of the specification, in accordance with RFC 2119 [5]. These words (MUST, SHOULD, MAY, MUST NOT, etc) are often capitalized.

This document defines the overall format of information in the ``extensions'' field of DHCP messages. The extensions defined within this document specify a generalized way to distribute information useful to a wide class of devices, operating systems and configurations. Sites with a DHCP server that is shared among heterogeneous clients may choose to define other, site-specific formats for the use of the ``extensions'' field.

Section 2 of this memo describes the formats of DHCP extensions. Information on registering new extensions is contained in section 8. The other sections organize the format descriptions of various extensions according to their general type, as follows:

- Releasable Resource Extensions (section 4)

- General Extensions (section 5)

    - TCP-specific Extensions

- DHCP-specific Extensions (section 6)

Future applications will make extensive use of an ever-increasing number and variety of network services. It is expected that client requirements for locating these network services will be satisfied by the Service Location Protocol [20], and not the DHCP. The DHCP is expected to be used for the kinds of configuration that enable clients to become fully functional as self-contained network entities.

2. DHCP Extension Field Format

Extensions may be fixed length or variable length. All extensions begin with a ``Type'' field, which is a an two octet unsigned network-order integer that uniquely identifies the extension. Every extension has a two octet unsigned network-order integer ``Length'' field following the ``Type'' field.  The ``Length'' field contains the number of octets of extension data that follow the ``Length'' field.  Thus, the ``Length'' field value does not include the number of octets needed to carry the ``Type'' and ``Length'' fields. For some extensions, the ``Length'' field is always the same number, but it MUST still be specified.  There is no requirement for alignment of data fields within existing DHCP extensions.  Any extensions defined subsequent to this document MUST contain a two-octet ``Length'' field even if the value it would contain would always be fixed or zero.

Unrecognized extensions MUST be silently ignored by skipping over the number of octets specified in the ``Length'' field, and processing continued for subsequent extensions.  Unless and until specified otherwise by use of the ``Maximum DHCP message size'' extension (section 6.1), DHCP implementations MUST assume that that the maximum DHCP message size including extensions is limited to 1280 octets.

All multi-octet quantities are in network-order.

Extension Type 0 (zero) is reserved.

There can be 65535 different extensions, which are divided up into the following ranges:

Releasable Resource Range (1--8192)

Extensions carrying data which identifies a resource which is leased by the server to a client for a finite period of time

known as a ``lease''. The client agrees to stop using the resource when the lease expires, and the server guarantees that it will not allocate the resource to another client until the lease expires or the client signals that it is done using the resource.

A server MUST NOT return a releasable resource to a client unless the client explicitly requests an instance of that resource from the server. This requirement ensures that only clients capable of managing a releasable resource receive them.

A client MUST remember which server allocated the client a releasable resource, in order to contact that server to extend the lease on the resource or release the resource back to the server when it is finished with it.

As of this writing, the only example of a type of releasable resource is an IP address, carried in the ``IP Address Extension'' (section 4.1). See the ``DHCP for IPv6'' companion document ([4]) for more details.

General Range (8193-49152)

Extensions in this range are informational in nature, and may point to resources which may be shared by any number of nodes. General extension proposals are reviewed by the DHC WG and IESG for general usefulness to the IETF community at large.

Servers MAY return any general range extension to clients if administrative policy requires it; however, a server SHOULD only return general extensions if the client requests them using the ``Extension Request Extension'' (ERE) (section 6.3).

Examples of general extensions include Domain Name Service parameters, Network Time Protocol (NTP, [14]) parameters, those carrying information pertaining to the DHCP, such as the ``Maximum DHCP Message Size'' (section 6.1), ``DHCP Retransmission and Configuration Parameter'' (section 6.2, and ``Platform Specific Information'' (section 6.5) Extensions.

Site-specific Range (49153--65535)

Extensions in this range are reserved for use by Site managers (administrators of the DHCP domain). Their type and content is entirely up to the administrator. DHCP implementations SHOULD permit the definition of site-specific extensions, including such information as data type and format. Note that both client and server implementations MAY need to be configured in order to properly exchange site extensions.

A server SHOULD only return site-specific extensions to the client if it explicitly requests them using the ``Extension Request Extension'' (ERE) (section 6.3).

All of the extensions described in this document MUST also have their default values specified, if any. Whenever an extension is received as part of a DHCP message, any reserved fields of the message MUST be ignored, and processing continued as if the reserved fields were zero. Typically, the value of the ``Type'' field is shown directly in the format illustration, and for some fixed-length extensions the value of the ``Length'' field is also shown in the format illustration for the extension.

3. DHCP Relay Considerations

The DHCP Relay MUST NOT change any information in any DHCP Extension fields. All Extension information flows between DHCP Server and DHCP Client without modification by any Relay.

4. Releasable Resource Extensions

Releasable resource extensions contain data identifying a specific resource leased by the server to the client for a specific period of time known as a ``lease''. Because the allocation of such extensions requires extension-specific management of the lease by both the client and the server, these extensions MUST only be returned to the client if they have been explicitly requested by the client.

How the resource and its lease is managed is resource-specific (extension-specific).

A client MUST remember in non-volatile storage which server allocated which releasable resource, in order to appropriately manage the lease associated with that resource.

As of this writing, the only example of a releasable resource is an IP address, which is carried in the ``IP Address Extension'', discussed below.

4.1. IP Address Extension

The IP address extension is used by clients and servers to refer to a particular IP address and related information such as the status of the host name associated with the IP address. All information relevant to a particular IP address allocation has been collected together within the ``IP address'' extension.

The ``lease'' feature of the ``IP address'' extension is implemented by the ``Preferred lifetime'' and ``Valid lifetime'' fields.

```
    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 = 1            |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    status     |C|I|L|Q|A|P|   reserved    |scope| prefix-len  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         (if present)                          |
   |                    IP address (16 octets)                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          (if present) preferred lifetime (4 octets)           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            (if present) valid lifetime (4 octets)             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         (if present) DNS name (variable length)  ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type   1

Length   (unsigned integer, variable) The length of the Extension in octets.

status   The receiver's result of its attempt to honor the sender's request.

C   If the ``C'' bit is set, the field containing the IP address is present in the extension.

I   If the ``I'' bit is set, the client is informing the server that the IP address listed *was not* received from DHCP (e.g.  received from Stateless Autoconfiguration, or manually configured).  The ``I'' bit MUST NOT be set if the ``C'' bit is not set (IP address field MUST be present).  The ``L'' bit MUST NOT be set if the ``I'' bit is set.

L   If the ``L'' bit is set, the preferred and valid lifetimes are present in the extension.

Q   If the ``Q'' bit is set, the fields included by the client are required, and must be made available by the server or else the extension must be rejected.

A   If the ``A'' bit is set, the client requests that the server updates DNS with a new AAAA/A6 record, as specified by the client's FQDN.

P   If the ``P'' bit is set, the client requests that the server updates DNS with a new PTR record, as specified by the client's FQDN.

reserved MUST be zero.

scope

This 3-bit field is used by the client to request an IP address of a certain scope. The 3 bits form a number (0--7) which can have the following settings:

0 Don't Care

1 Globally-scoped address

2 Site local-scoped address

3 reserved

4 reserved

5 reserved

6 reserved

7 reserved

prefix-len

If the IP address field is present (the ``C'' bit is set), a non-zero prefix-len is the number of left-most bits of the IP address which make up the subnet prefix. Otherwise, if the ``C'' bit is not set, prefix-len MUST be zero. The prefix-len field is 7-bits in length.

IP address

The IP address to be conveyed to the receiver from the sender. (16 octets long).

preferred lifetime

The preferred lifetime of the IP address in seconds.

valid lifetime

The valid lifetime of the IP address in seconds.

DNS name

The DNS name (a string of NVT-ASCII octets) associated with the IP address.

The following values for the status field are defined within this document:

0   request granted, no errors

1   IP address is already in use by a different node

2   Extension settings (bit combinations) illegal

3   IP address scope requested is not available

4   IP address requested by client is not available

18   Security parameters failed for this client

20   Resource AAAA/A6 Record Parameter Problem

21   Resource PTR Record Parameter Problem

23   DNS name string error

24   dynDNS Not Implemented

25   Authoritative DNS Server could not be found

33   The name server was unable to interpret the request due to a format error.

34   dynDNS unavailable at this time (SERVFAIL)

35   Some name that ought to exist, does not exist (NXDOMAIN)

36   The name server does not support the specified Opcode (NOTIMP)

37   The name server refuses to perform the specified operation for policy or security reasons (REFUSED)

38   Some name that ought not to exist, does exist (YXDOMAIN)

39   Some RRset that ought not to exist, does exist (YXRRSET)

40   Some RRset that ought to exist, does not exist (NXRRSET)

41   The server is not authoritative for the zone named in the Zone Section (NOTAUTH)

42   A name used in the Prerequisite or Update Section is not within the zone denoted by the Zone Section (NOTZONE)

Status values 33 through 42 are described more fully within Dynamic Updates to DNS (RFC 2136 [21]). Up-to-date values for the values of the status field are specified in the most recent ``Assigned Numbers'' document [17].

The DNS name can be a host name, which does not contain the ``.'' ASCII character as a separator between DNS hierarchy components. Any name containing the ``.'' is treated as a Fully Qualified Domain Name (FQDN). The length of the DNS name may be determined by subtracting, from the Length, the length of those fixed length fields which are present.

If the ``Q'' bit is set, the values or actions requested by the C, I, L, A, P bits and the scope field are required, and MUST be provided, or the extension MUST be rejected with the appropriate ``status'' field value, indicating the reason why the server was unable to fulfill the required extension attributes. The ``Q'' bit is NEVER used by the server in ``IP address'' extensions it generates.

A DHCP client can include an IP address in its IP Address extension and set the ``I'' bit and the ``A'' bit and/or ``P'' bit to ask the DHCP Server to use the information contained in the extension to update the DNS on the client's behalf. This would be done for IP addresses obtained by a method other than the DHCP, such as Stateless Address Autoconfiguration, RFC 2462 [19].

If the client wishes to have its FQDN associated with one of several existing IP addresses which it has received from the DHCP Server, the client MUST supply that IP address in the IP address extension along with the FQDN.

By default, the client SHOULD update the AAAA/A6 (See [7] for information about the A6 record type) record, and the server SHOULD update the PTR record. The IP Address extension permit clients and servers to use a different behavior than the default through the use of the 'Q' and 'A' bits and associated fields.

4.1.1. IP Address Lifetimes

The lifetime value contained in the ``preferred'' and ``valid'' fields is a value relative to when the sender sent the message containing the ``IP address'' extension.  The receiver adds these times to the current clock time in order to determine the absolute times for the ``preferred'' and ``valid'' lifetimes.

4.1.2. Client Considerations

Sent in a DHCP Request Message

In a DHCP Request message (for each IP address extension), a client MUST initialize the ``status'' field value to zero.

A client may include multiple IP Address extensions in a single DHCP Request, in order to request as many IP addresses of varying scopes or subnet prefixes as it requires.

In a DHCP Request (for each address extension), a client MAY:

- o Request that any IP address of a specific scope be returned. The client does this by setting the scope value to the desired value. The ``C'' bit and prefix-len fields MUST NOT be set, as the client is not requesting a particular IP address.

- o Request the lease of some IP address on a specific network (subnet prefix) or a specific IP address (interface ID specified). The client does this by setting the ``C'' bit and including the desired information in the ``IP address'' and ``prefix-len'' fields in the extension. Note that the client MUST set the prefix-len field to the number of left-most bits representing the subnet prefix, even if it is requesting a specific IP address.

- o Ask that the IP address returned have the ``preferred'' and ``valid'' lifetimes suggested by the client. The client does this by setting the ``L'' bit and including the desired lifetime values in the ``preferred'' and ``valid'' lifetime fields. The client MUST use the lifetimes returned by the DHCP server.

- o Request that the DHCP server perform a DNS AAAA/A6 record update (``A'' bit is set) and/or DNS PTR record update (``P'' bit is set) for either an IP address the server will assign (``I'' bit not set) or the IP address the client has provided (``I'' bit set) which it acquired through a means other than DHCP, for the host name or Fully Qualified Domain Name (FQDN) the client has provided.

o Specify that the attributes of the request carried by the ``IP address'' extension are required by the client by setting the ``Q'' bit.

Received in a DHCP Reply Message in response to a DHCP Request

When the client receives an IP address extension within a DHCP Reply message, it first validates that the bits / fields set in the extension are valid. If they aren't, the client generates a DHCP Release message including the ill-formed IP address extension, and sets the ``status'' field to 2, and sends it to the server.  If the extension is valid, the client inspects the ``status'' field value to see whether the client's request has been granted. If the status is nonzero, the client should log the error, and display the error condition for action by the user and/or the network administrator. Non-zero status almost always indicates that the client will be need to modify its request before it could be satisfied by the replying DHCP server, or alternatively that the replying DHCP server will need to be given updated configuration information for the client.

Upon reception of a new IP address, the client MUST perform Duplicate Address Detection (DAD) as specified in RFC 2462 [19]. If the IP address has already been allocated to the client and the client is merely requesting a renewal of the lifetime of the IP address, the client MUST NOT perform DAD, as it is using this IP address. If the client finds that the new IP address is in use by another node, the client forms a DHCP Release message including the IP address extension containing the in-use IP address, and sets the ``status'' field value to 1, and sends the Release to the DHCP server.

If the client receives an IP address with zero valid lifetime and:

- The DHCP Reply message has been authenticated, the client MUST immediately discontinue using that IP address.

- The DHCP Reply message has no authentication, the client sets the valid lifetime for the address to 2 hours.

When the preferred lifetime of an IP address leased from the DHCP server is 80% exhausted, the client SHOULD begin sending DHCP requests to the server requesting a renewal of the lease on that IP address. If the client is unsuccessful at its attempts and the valid lifetime expires, the client MUST immediately stop using that IP address.

Sent in a DHCP Release Message

A client sends IP address extension(s) in a DHCP Release message when:

- o It is releasing an IP address back to the server because it is finished using it.

- o It has discovered through DAD that the IP address assigned by the DHCP server is already being used by a different node.

- o The IP address extension received from the DHCP server has an illegal combination of bit/fields settings.

- o The client wishes to delete the DNS records associated with the IP address/hostname it will present to the server.

4.1.3. Server Considerations

This section contains information specifying the handling of the ``IP Address'' extension by DHCP servers.

Note that a server implementation MUST scan its client bindings from time to time to locate bindings whose lifetimes have expired. Those bindings SHOULD be deleted, and any DNS operations performed which are recorded in those bindings MUST be reversed.

The DHCP Advertise Message and the ``IP address'' Extension

The ``IP address'' extension is not used in the DHCP Advertise message.

Received in a DHCP Request Message

When a server processes an ``IP address'' extension within a DHCP Request, the server first validates the combination of bits / fields contained within the extension. If these bits / fields are set incorrectly, the server generates a DHCP Reply message, which includes the incorrect IP address extension from the client's request, with the ``status'' field set to 2, thereby notifying the client of the error. If the IP address extension is correct, the server processes the extension as follows:

- o If no IP address field is present, then the client is requesting that the server allocate an IP address of the scope identified by the ``scope'' field value. If the IP address field is present and the ``I'' bit is not set, then the client is requesting the assignment of:

  * A specific IP address (interface ID present)

  * Any IP address in the specified network (interface ID is zero)

  The prefix-len field specifies the length of the subnet prefix in either case.

  The server consults its allocation tables and attempts to select an IP address meeting the client's request which is appropriate for the link to which the client is attached. The link can be determined by the contents of the relay-agent address and prefix-len fields of the DHCP Request message. If these fields are set, then the client is off-link, otherwise the client is attached to one of the same links as the server.

- If the client is requesting that the server update the DNS on its behalf (either for the IP address the server will assign or the one it provided which was acquired through some other means (not the DHCP)), the server makes the appropriate DNS dynamic update requests and records the status of the update within the ``status'' field of the IP address extension it will include in the DHCP Reply message sent to the client. If the ``Q'' bit is set, then the server will ensure that the DNS operation has completed successfully before responding to the client. If the ``Q'' bit is not set, then the server SHOULD register the update request with the DNS, and MAY immediately return its DHCP Reply without waiting for the result of the DNS operation.

  If the client has requested that the server perform DNS updates as part of the IP address allocation and configuration, the server MUST maintain this fact as part of the client's binding. Then, if the client eventually releases the IP address by sending a DHCP Release message or the lifetimes associated with the IP address expire because the client has not renewed them, the server MUST perform the reverse service by updating DNS again to remove the changes it has made on the client's behalf.

- If the client has set the ``Q'' bit, then all fields within the IP address extension which represent attributes of interest to the client are requirements, and must be met, otherwise a DHCP Reply message is generated with the ``status'' field set identifying the portion of the request the server could not fulfill. Note that if more than one attribute of the request could not be provided, the server can only identify one of the problems in the ``status'' field.

Sent in a DHCP Reply Message in response to a DHCP Request

If the server is assigning an IP address (or extending the lifetimes of an existing IP address binding the client holds), the server MUST include an IP address extension for the IP address with the following settings:

- the preferred lifetime

- the valid lifetime

If the DHCP Reply is a response to a DHCP Release, the lifetimes MUST both be zero.

If the server has performed DNS operations on behalf of the client, it sets the ``A'' and ``P'' bits if the AAAA/A6 record and PTR record respectively have been updated by the DNS.

The ``status'' field of the extension MUST be set by the server indicating the result of the server's attempt to honor the client's IP address-related request.

If the server receives a DHCP Request from one of its clients whose address it wishes to invalidate, it can cause the client to discontinue use of the old address by including valid and preferred lifetimes with a value of zero.

To perform renumbering, the server will include two IP address extensions, one to reduce the preferred and valid lifetimes for the old address, and another to give the client its new address.

Received in a DHCP Release Message

When a server processes an ``IP address'' extension within a DHCP Release, the server first validates the combination of bits / fields contained within the extension. If these bits / fields are set incorrectly, the server generates a DHCP Reply message, which includes the incorrect IP address extension from the client's request, with the ``status'' field set to 2, thereby notifying the client of the error. If the IP address extension is correct, the server continues to processes the extension.

The client generates a DHCP Release for the following reasons:

- o Client is finished with the IP address. In this case, the client has determined it no longer needs the IP address, and is returning it to the server for use by other clients.

  The server removes the IP address from the client's binding, returning it to the general pool of IP addresses. If the server has performed DNS operations on behalf of the client regarding this IP address, the server contacts the DNS service and deletes the changes it has made regarding the FQDN/IP address. The server generates a DHCP Reply including the client's IP address extension, with the ``status'' field set to indicate the results of the release operation.

- o Client has discovered through DAD that the IP address is already in use by another node. The server MUST mark the errant IP address as unavailable for assignment, and SHOULD generate a log message indicating the problem to the administrator. The server then generates a DHCP Reply message containing the client's IP address extension, with the ``status'' field set to 0 to indicate that it has received the client's release.

- o The client has requested that the DHCP server serve as a DNS update proxy for a name associated with an IP address that it acquired outside of the DHCP. The server will undo the DNS operations it performed on behalf of this client, deletes its knowledge of those operations, and generates a DHCP Reply message including the client's IP address extension with the ``status'' field set to indicate the result of the release operation.

The ``IP Address'' Extension and the DHCP Reconfigure-init Messages

A server MUST NOT include an ``IP address'' extension in DHCP Reconfigure or DHCP Reconfigure-init messages. IP addresses may be changed during the DHCP Request/Reply exchange set in motion by DHCP Reconfigure-init message(s).

5. General Extensions

General extensions (in the range 8193-49152) are important for many DHCP clients, and are not specific to any upper-level protocol.

5.1. IEEE 1003.1 POSIX Timezone Extension

This extension allows delivery of timezone information in the form of a IEEE 1003.1 POSIX Timezone specifier, as detailed in section 5.1.1.

```
 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 = 8193           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    IEEE 1003.1 POSIX Timezone string (variable length) ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

If a DHCP client finds that the POSIX Timezone extension value is misformatted, it SHOULD notify the the user of the problem and MUST discard the entire extension value.

5.1.1. IEEE 1003.1 POSIX Timezone specifier

The format of the IEEE 1003.1 POSIX timezone string is specified as

```
StdOffset[Dst[Offset],[Start[/Time],End[/Time]]]
```

where '[' and ']' enclose optional fields, '|' indicates choice of exactly one of the alternatives, ',' and '/' represent literal characters present in the string, and:

Std   three or more octets for the standard timezone (Std). Any characters (or case) except a leading colon, digits, comma, minus or plus sign are allowed.

Offset   Indicates the value one must add to local time to arrive at UTC, of the form: [+|-]hh[:mm[:ss]]. Offset following Std is required. Digits are always interpreted as decimal number. If preceded by a '-', the timezone is east of the Prime Meridian, otherwise it is west ('+' is optional) The permissible values for hh[:mm[:ss]] are as follows:

hh	0 <= hh <= 23
mm	0 <= mm <= 60
ss	0 <= ss <= 60

Offset has no default value.

Dst three or more octets for the daylight savings timezone. If Dst is missing, then daylight savings time does not apply in this locale. If no Offset follows Dst, then Dst is assumed to be one hour ahead of standard time. Any characters (or case) except a leading colon, digits, comma, minus or plus sign are allowed.

Start Indicates the day of the year, in one of the formats indicated below, when to change to daylight savings time. The ``Time'' field (which follows immediately after a ``/'' character, if present) indicates when the change is made, in local time.

End Indicate the day of the year, in one of the formats indicated below, when to change back from daylight savings time. The ``Time'' field (which follows immediately after a ``/'' character, if present) indicates when the change is made, in local time.

Time Time has the same format as Offset, except that no leading ``-'' or ``+'' is permitted. The default is 02:00:00.

The day of the year can be given in one of the following formats:

Jn The julian day n, (1 <= n <= 365). Leap days are not counted.

n Zero-based julian day, (0 <= n <= 365). Leap days are counted so it is possible to refer to Feb 29.

Mm.n.d The ``d''th day, (0 <= d <= 6) of week ``n'' of month ``m'' of the year (1 <= n <= 5, 1 <= m <= 12, where week 5 means last ``d'' day in month ``m'' which may occur in either the fourth or the fifth week. Week ``1'' is the first week in which the ``d'' day occurs.

5.1.2. An Example:

For Eastern USA time zone, 1986, the Posix timezone string is as follows:

```
EST5EDT4,116/02:00:00,298/02:00:00
```

Here, ``5'' is the Offset for Std, and ``4'' is the Offset for Dst. Start is the 116th day at 2am, and End is 298th day at 2am.

5.2. Domain Name Server Extension

```
 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 = 8194          |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Domain Name System server addresses              |
|                       (16 octets each)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Domain Name Server extension specifies a list of Domain Name System (STD13 [16]) name servers available to the client. Servers SHOULD be listed in order of preference.

The minimum Length for this extension is 16 octets, and MUST always be a multiple of 16.

5.3. Domain Name Suffix Extension

This extension specifies the default domain name suffix that client should use when resolving hostnames via the Domain Name System.

```
 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 = 8195         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Domain Name Suffix (variable length)  ...        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The minimum length for this extension is 1. The domain name is a NVT-ASCII string, ``Length'' octets in size. If the Domain Name Suffix extension is not specified, and the IP Address extension received by a client contains a FQDN, then the client MAY take the part of the FQDN after the first ``.'' octet as the Domain Name Suffix.

5.4. Service Location Protocol Directory Agent Extension

Entities using the Service Location Protocol (SLP) [20] need to find out the address of one or more Directory Agents in order to transact messages, and possibly the correct scope to be used in conjunction with the service attributes which are exchanged using the Service Location Protocol.

The Directory Agent extension requests or specifies a Directory Agent (DA), along with zero or more scopes supported by that DA. Note that this extension MAY be included multiple times in the same DHCP Request or DHCP Reply. If so, then the extensions SHOULD be included in order of decreasing preference.

```
 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 = 8196         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|D|F|M|T|        reserved       |           DA length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Directory Agent (variable length) ...
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        (if present) Typed-Scope-List (variable length) ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Length   (unsigned integer, variable) The length of the Extension in octets.

D   If the ``D'' bit is set, the Directory Agent field and the DA Length fields are present.

F   If the ``F'' bit is set, the Directory Agent is indicated by including its variable length host name or Fully Qualified Domain Name (FQDN) instead of its IP address.

M   If the ``M'' bit is set, the Directory Agent address MUST be present, and multicast methods for discovering Directory Agents MUST NOT be used.

T   If the ``T'' bit is set, the Typed-Scope-List is present.

rsv   reserved; ignored upon reception; MUST be sent as zero

DA Length
The length (in octets) of the Directory Agent field.

Directory Agent
The FQDN, host name, or IP address of the Directory Agent.

Typed-Scope-List
The string denoting the typed-scope-list formatted as explained in the description of the service scope extension (section 5.5).

In order to simplify administration of the configuration of DAs for clients using SLP, the DA can be indicated by presenting its host name or FQDN instead of its IP address. This allows renumbering to proceed more smoothly as outlined in RFC1900 [6]. When the FQDN or host name is used, the server sets the ``F'' bit. The host name can be distinguished from the FQDN by the presence of a ``.'' character. In any case, the DA length field is set to be the length of the Directory Agent field. When the ``F'' bit is not set, the DA Length MUST be 16.

Note that more than one Directory Agent extension may be present in a DHCP message. Each such extension may have the same or different typed-scope-list. The client may request any Directory Agent with a particular scope, by including the Directory Agent extension in a DHCP Request message with no Directory Agent address included (the ``D'' bit set to zero), and a nonempty typed-scope-list. The length of the Typed-Scope-List is only indicated implicitly by the overall length of the extension.

The format of the Typed-Scope-List field is described in the service scope extension (section 5.5).

The ``M'' bit MUST NOT be set when the extension is used as part of a DHCP Request message.

Extension type 8196 MUST include one or more scopes if a DA address is returned. Using extension type 8196, it is not possible for different service types on the same node to be configured with different directory agents. In other words, all service agents of the same service type on the same node will be configured with the same directory agent.

5.5. Service Location Protocol Service Scope Extension

This extension indicates a scope that should be used by a Service Agent (SA) as described in RFC 2165 [20], when responding to Service Request messages as specified by the Service Location Protocol (SLP).

This extension MAY be included multiple times in the same DHCP Request or DHCP Reply.

```
 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 = 8197         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Typed-Scope-List ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Length (unsigned integer, variable) The length of the Extension in octets.

Typed-Scope-List

In Service Location Protocol (SLP) [20], multiple service types can be hosted on the same network node. It is possible that different service types on the same computer would be administered from different scopes. Thus, extension types 8196 and 8197 have additional syntax to allow this more detailed style of service configuration.

In particular, the list of scopes contained in the extensions is syntactically separated into lists pertaining to each service type.

Grammatically, a typed-scope-list extension in a DHCP Reply is structured as follows:

```
     typed-scope-list = one or more
      maybe-typed-scope-items,
separated by commas
     maybe-typed-scope-item =
      typed-scope-item,
or scope-list
     typed-scope-item = '(' service-type
      '=' scope-list ')'
     scope-list = one or more
      scope-items, comma-separated
```

A typed-scope-list extension in a DHCP Request is structured as follows:

```
     typed-scope-list = one or more
      maybe-typed-scope-items,
separated by commas
```

```
     maybe-typed-scope-item = typed-scope-item,
      or maybe-empty-scope-list
     typed-scope-item = '(' service-type '='
      maybe-empty-scope-list ')'
     maybe-empty-scope-list = zero or
      more scope-items, comma-separated
```

A service type has the format defined in RFC 2609 ([9]), and a scope-item has the format defined in RFC 2608 ([10]) for ``strval''. Basically, a scope-item is a character string that has alphanumeric characters not including control characters or `(',`)',`,', \',`!',`<',`=',`>', or `"' Service schemes are special cases of schemes as defined for general URLs in RFC 1738 ([3]).

The typed-scope-list MAY contain both untyped-scope-lists and typed-scope-lists. Each scope-item in each untyped-scope-list applies to every service type on the node. The string containing the typed-scope-list is NOT null-terminated. The typed-scope-list string must be UTF-8 character encoded.

As an example, the scope-list ``A,B,C'' denotes scopes A, B and C for all service types on the client. In a DHCP Request, this scope string would indicate that the client wishes a directory agent which supports ANY of these three scopes. In a DHCP Reply, the scope indicates that the directory agent supports ALL of the three scopes.

Suppose instead that service types ``netman'' and ``proxystuff'' are residing on a DHCP client. Then, the typed-scope-list in a DHCP Reply could be:

(netman=mgmt),(proxystuff=math-dept,labs)
Assuming the DHCP client with two service types ``netman'' and ``proxystuff'' did not make any scope restriction, a corresponding typed-scope-list in a DHCP Request could be:

(netman=),(proxystuff=)
asking for scopes for those service types.

The Typed-Scope-List is described in section 5.5. The DHCP client (i.e., user agent or service agent) which receives this extension will use the indicated scope for in all SLP requests and registrations.

DHCP clients MAY use extension 8197 to request scopes for one or more particular service types. Note that more than one Service Scope extension may be present in a DHCP message. The length of the typed-scope-list is only indicated implicitly by the overall length of the extension.

5.6. Network Time Protocol Servers Extension

This extension specifies a list of IP addresses indicating NTP [13] servers available to the client. Servers SHOULD be listed in order of preference.

```
    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 = 8198          |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     NTP server addresses                      |
   |                       (16 octets each)                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The minimum Length for this extension is 16, and the Length MUST be a multiple of 16.

5.7. Network Information Service (NIS) Domain Name Extension

This extension specifies the name of the client's NIS domain. The domain is formatted as a character string consisting of characters from the NVT-ASCII character set. The minimum length for this extension is 1.

```
    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 = 8199          |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              NIS Domain Name (variable length)  ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

5.8. Network Information Service (NIS) Servers Extension

This extension specifies a list of IP addresses indicating NIS servers available to the client. Servers SHOULD be listed in order of preference.

```
 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 = 8200          |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     NIS server addresses                      |
|                       (16 octets each)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The minimum Length for this extension is 16, and the length MUST be a multiple of 16.

5.9. Network Information Service V2 (NIS+) Domain Extension

This extension specifies the name of the client's NIS+ domain. The domain is formatted as a character string consisting of characters from the NVT-ASCII character set. The minimum Length for this extension is 1.

```
 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 = 8201          |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              NIS+ Client Domain Name (variable length)  ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

5.10. Network Information Service V2 (NIS+) Servers Extension

This extension specifies a list of IP addresses indicating NIS+ servers available to the client. Servers SHOULD be listed in order of preference.

```
     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 = 8202          |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                     NIS+ server addresses                     |
    |                       (16 octets each)                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The minimum Length for this extension is 16, and the length MUST be a multiple of 16.

5.11. TCP-specific Extensions

This section lists the extensions that affect the operation of the TCP layer on a per-interface basis.

5.11.1. TCP Keepalive Interval Extension

```
     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 = 8203          |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                    Keepalive Time Interval                    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

This extension specifies the interval (in seconds) that the client TCP should wait before sending a keepalive message on a TCP connection. The time is specified as a 32-bit unsigned integer. A value of zero indicates that the client should not generate keepalive messages on connections unless specifically requested by an application.

The length for this extension is 4.

6. DHCP-specific Extensions

This section details the extensions that are used by the DHCP.

6.1. Maximum DHCP Message Size Extension

This extension specifies the maximum size in octets of any DHCP message that the sender of the extension is willing to accept. The size is specified as an unsigned 32-bit integer. A client may use the maximum DHCP message size extension in DHCP Request messages, but MUST NOT use the extension in other DHCP messages.

```
 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 = 8204         |          Length = 4           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Max DHCP Message Length                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Length for this extension is 4. The minimum permissible value is 1280, as specified in RFC 2460 [8].

6.2. DHCP Retransmission and Configuration Parameter Extension

This extension allows configuration of values for DHCP retransmission and configuration variables, as specified for use when sending or receiving DHCP messages. These variables are discussed in detail in the section on ``Configuration Variables'' in the ``DHCP for IPv6'' companion document [4].

```
 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 = 8205         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Configuration Variable Identifier (2 octets)          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 New Variable Value  (8 octets)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The length for this extension is 10 octets. The ``Configuration Variable Identifier'' field carries an unsigned 16-bit network-order integer representing the configuration variable. The ``New Variable Value'' field carries a 64-bit network-order integer representing the value of the configuration variable. If a client uses this extension in a DHCP Request message, then the ``New Variable Value'' field MUST be 0 (zero). If a client does not receive a setting for the ``Configuration Variable Identifier'' it has requested, it MUST use

the default values defined in the ``Configuration Variables'' section of the ``DHCP for IPv6'' document [4].

6.3. Extension Request Extension (ERE)

The ``Extension Request Extension'' (ERE) MAY be used by DHCP implementations to indicate which DHCP extensions they are interested in (client), or what DHCP information (e.g. extensions) are available (server).

```
 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 = 8206         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| A | B | C | D | E | ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The extension contains a list of extension ``Type'' values indicating the extension of interest. Since an extension ``Type'' field is an unsigned two-octet network-order integer, each extension is identified by two-octets. Thus, the Length field MUST always be an even number. Extension types listed in the ERE are listed in priority order, with the extensions of highest priority listed before those of lower priority. One and only one ERE extension is permitted within a DHCP message.

6.3.1. Client Considerations

If the client implementation supports it, the client SHOULD generate a Extensions Request Extension identifying which extensions it is interested in and include it in its DHCP Request messages.

6.3.2. Server Considerations

If a server receives a DHCP request with an ERE extension present, the server SHOULD attempt to provide valid values for all the information requested.

6.4. Subnet Prefix Extension

The ``Subnet Prefix'' extension is a DHCP server-only extension used to advertise what networks are available on the client's link. Each extension carries a single subnet prefix.

```
 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 = 8207           |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Prefix Len (number of left-most bits) (1 octet)        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |       Subnet Prefix Octets (variable number of octets)        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

A subnet prefix is specified by the ``Subnet Prefix Octets'' field and the ``Prefix-Len'' field. The ``Subnet Prefix Octets'' field is large enough to contain all the bits of the subnet prefix. Any unused bits in the last octet of this field MUST be set to off (zero).

The length of this extension is variable.

6.4.1. Client Considerations

Clients MAY use the ``Subnet Prefix'' extension value to request one or more IP addresses on that network. A client does this by forming an ``IP address'' extension with the value of the ``Subnet Prefix Octets'' field copied into the high-order portion of the ``IP address'' field (``C'' bit set) and the ``Prefix-len'' value copied into the ``prefix-len'' field of the ``IP address'' extension for each IP address desired on the advertised network.

6.4.2. Server Considerations

In response to a client's DHCP Solicit message (``P'' bit set), a server SHOULD return one ``Prefix'' extension for each of the networks it is configured to manage that exist on the client's link in the resultant DHCP Advertise message. A server SHOULD NOT include ``Prefix'' extensions in its Advertise messages if the client has not requested them (``P'' bit NOT set).

If a server receives a DHCP Request message with ``Prefix'' extension(s), that DHCP Request message MUST be dropped.

6.5. Platform Specific Information Extension

A platform is defined as the combination of hardware and operating system (OS).

This extension is used by clients and servers to exchange client-platform-specific information. The information is an opaque collection of data, presumably interpreted by platform-specific code on the clients. The definition of this information is platform specific. Clients identify their platform through the use of the Platform Class identifier extension (see Section 6.6). Clients which do not receive platform specific information SHOULD make an attempt to operate without it, although they may do so (and announce that they are doing so) in a degraded mode.

If a platform vendor encodes more than one item of information in this extension, then the vendor MUST encode the extension using ``Encapsulated platform-specific extensions'' as described below. The ``Encapsulated platform-specific extensions'' field MUST be encoded as a sequence of type/length/value fields of identical syntax to the form defined for DHCP extensions (see section 2), encapsulated within the ``Platform Specific Information Extension''.

```
 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 = 8208           |             Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Platform-specific extension information  ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The minimum length for this extension is 4.

DHCP servers which support the configuration of ``Platform Specific Information'' extensions, and which have been configured with configuration information specific to some number of ``Platform Class Identifiers'' MUST select and return only those platform-specific extensions which match the ``Platform Class Identifier'' provided by the DHCP client.

6.6. Platform Class Identifier Extension

This extension is used by a DHCP client to identify the hardware type and operating system platform it is hosted on. The extension value itself is an opaque value to a DHCP server, and is only used by the DHCP server to "lookup" Platform Specific Extensions associated with clients of a certain platform class. DHCP servers SHOULD also allow the association of other extensions (Releasable, General, etc) with clients of a certain platform class.

Note that unlike the ``User Class Identifier'' (see section 6.7, the ``Platform Class Identifier'' does not need to be echoed back to the

DHCP client because there can be one and only one ``Platform Class Identifier'' for a client.

```
 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 = 8209         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Platform Class Identifier ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The minimum length for this extension is 4.

The ``Platform Class Identifier'' is a string of UTF-8 characters of Length octets. The ``Platform Class Identifier'' represents the hardware and operating system class of which the client is a member.

In order to prevent collisions in the ``Platform Class Identifier'' namespace, DHCP client vendors MUST prefix their ``Platform Class Identifiers'' with their stock symbol or some other globally recognized organizational identifier. For example, ``Platform Class Identifiers'' for Sun Microsystems Inc platforms would be prefaced by ``SUNW'', the NASDAQ stock symbol for Sun. Those associated with Microsoft platforms would be prefaced by ``MSFT''.

6.6.1. Client Considerations

If the client wishes platform-specific data, it includes a platform class identifier extension identifying its platform type.

6.6.2. Server Considerations

Servers not equipped to interpret the platform class identifier specified by a client MUST ignore it (although it may be reported to the DHCP administrator). Otherwise, servers SHOULD respond with the set of extensions corresponding to the platform class identifier specified by the client.

6.7. User Class Identifier Extension

This extension is used by a DHCP client to optionally identify the type or category of user or applications it represents.

Network administrators may define specific user class identifiers to convey information about a client's software configuration or about its user's preferences. For example, an identifier may specify that a particular machine hosting a DHCP client is a member of the class ``accounting auditors'', which have special service needs such as a particular database server or printer.

```
 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 = 8210         |               Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      User Class Identifier ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The minimum length for this extension is 4.

The user class identifier is a UTF-8 string of Length octets. The value of the ``User Class Identifier'' is selected by the administrator of the DHCP domain containing the all members of this class. Thus, a ``User Class Identifier'' need only be unique within the DHCP domain, although the administrator MAY choose to prefix the ``User Class Identifier'' with the department name in order to reduce the possibility of ``User Class Identifier'' name space collisions.

6.7.1. Client Considerations

If the client is configured to request user class-specific data, it includes a User Class identifier extension for each user class it is configured with.

6.7.2. Server Considerations

Servers not equipped to interpret the user class identifier specified by a client MUST ignore it (although it may be reported to the network administrator). Otherwise, servers SHOULD respond with the set of extensions corresponding to the user class identifier specified by the client. Further, if the server responds with the set of extensions corresponding to the given user class identifier, it MUST echo the client's user class identifier extension back to the client.

6.8. Reconfigure Multicast Address Extension

A DHCP server can instruct its clients to join one or more multicast groups for the purposes of receiving DHCP Reconfigure or DHCP Reconfigure-init messages. The DHCP server accomplishes this by

returning a ``Reconfigure Multicast Address Extension'' for each multicast address associated with the group. See the ``DHCP for IPv6'' document [4] for more details on the use of this extension.

```
    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 = 8211         |          Length = 16          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 Reconfigure Multicast Address                 |
   |                          (16 octets)                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Length for this extension is 16.

6.9. Renumber DHCP Server Address Extension

A DHCP server can instruct its clients to change their internal records to reflect the server's newly renumbered IP address, by using the ``Renumber DHCP Server Address Extension''. This extension SHOULD be sent in the DHCP Reconfigure message.

The server includes both its previous IP address and its new IP address. Providing the previous IP address allows clients to update only those resource associations owned by this server.

```
    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 = 8212         |          Length = 32          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                  Previous DHCP Server Address                 |
   |                          (16 octets)                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    New DHCP Server Address                    |
   |                          (16 octets)                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Length for this extension is 32.

6.10. Client-Server Authentication Extension

Exactly one ``Client-Server Authentication Extension'' MAY be present in any DHCP message transmitted between a client and server (or vice-versa). If present, it MUST be the last extension.

```
 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 = 8213         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Security Parameters Index (SPI)               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Replay Protection                      |
|                           (8 octets)                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Authenticator (variable length) ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Length   (unsigned integer, variable) 4 for the SPI, plus 8 for the replay protection, plus the number of octets in the Authenticator.

SPI   A Security Parameters Index (SPI) [2] identifying a security context from among those available between the DHCP client and server.

Replay Protection
          A 64-bit timestamp (in Network Time Protocol (NTP) [15] format) (see section 7.1).

Authenticator
          (variable length) (See Section 7.2.)

This authentication extension remedies the inability of IPsec (RFC 2402 [11]) to provide for non end-to-end authentication, since authentication is needed even when the client has no IPv6 address with enough scope to reach the DHCP server. The extension can be originated by either the client or server to authenticate the rest of the data in the DHCP message. The default authentication algorithm, which MUST be supported by all clients and servers, is defined in section 7.2.

SPI values 0 through 255 are reserved and, if used, MUST conform to the security context defined by that value in the most recent Assigned Numbers RFC (e.g., STD1 [17]).

6.11. Client Key Selection Extension

A DHCP server may wish to indicate to a prospective client which SPI it must use to authenticate subsequent messages, using the ``Client-Server Authentication Extension''. In such cases, the server includes the ``Client Key Selection Extension'' in its DHCP Advertise message.

```
 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 = 8214           |            Length = 4         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Security Parameters Index (SPI)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Security Parameters Index (SPI) [2] identifies a security context between a pair of nodes among the contexts available in the security association defined between the DHCP client and server.

SPI values 0 through 255 are reserved and, if used, MUST conform to the security context defined by that value as defined in the most recent Assigned Numbers RFC (e.g.,STD1 [17]).

7. Security Considerations

A security protocol is urgently needed for use with DHCP, since otherwise malicious parties could create numerous denial-of-service style attacks based on depleting available server resources or providing corrupted or infected data to unsuspecting clients. The following sections discuss aspects of security relevant for users of the Client-Server Authentication extension 6.10. See also the Security Considerations in the companion specification [4].

7.1. Replay Protection

A 64-bit timestamp, in Network Time Protocol [15](NTP) format, is used to protect against replay of previous authenticated messages by malicious agents. The NTP timestamp value used in the extension

MUST be chosen, and verified, to be larger than values used by the originator in previous Client-Server Authentication extensions. On the other hand, the timestamp value MUST also be chosen (and verified) to be no greater than one year more than the last known value (if any) used by the originator.

7.2. Default Authentication Algorithm

The default authentication algorithm is HMAC [12], using keyed-MD5 [18]. Given a secret key K, and "data" the information to be authenticated, HMAC_result is computed as follows:

1. opad := 0x36363636363636363636363636363636 (128 bits)

2. ipad := 0x5C5C5C5C5C5C5C5C5C5C5C5C5C5C5C5C (128 bits)

3. zero_extended_key := K extended by zeroes to be 128 bits long

4. opadded_key := zero_extended_key XOR opad

5. ipadded_key := zero_extended_key XOR ipad

6. HMAC_result := MD5 (opadded_key , MD5 (ipadded_key, data))

The key K is the shared secret defined by the security association between the client and server and by the SPI value specified in the Authentication Extension. The "data" is the stream of octets in all previous fields in the DHCP message and extensions. The authenticator is the 128-bit value HMAC_result.

8. IANA Considerations

This document MAY be superseded by new documents for DHCP extensions, which will then include the entire current list of valid extensions. This section details the method for specifying new extensions.

Implementation specific use of undefined extensions (all those in the range 86-32767 inclusive) may conflict with other implementations, and registration is required.

The following steps MUST be followed by the author of any new DHCP extension, in order to obtain acceptance of the extension as a part of the DHCP Internet Standard:

1. The author documents the new extension as an Internet Draft.

2. The author submits the Internet Draft for review through the IETF standards process as defined in "Internet Official Protocol Standards" [17]. The new extension will be submitted for eventual acceptance as an Internet Standard.

3. The author requests a number for the new extension from IANA by contacting:

   Internet Assigned Numbers Authority (IANA)
   USC/Information Sciences Institute
   4676 Admiralty Way

   Marina del Rey, California 90292-6695
   or by email as: iana@isi.edu

4. The new extension progresses through the IETF standards process; the new extension will be reviewed by the Dynamic Host Configuration Working Group (if that group still exists), or as an Internet Draft not submitted by an IETF working group.

5. If the new extension fails to gain acceptance as an Internet Standard, the assigned extension number will be returned to IANA for reassignment.

This procedure for defining new extensions will ensure that:

- allocation of new extension numbers is coordinated from a single authority,

- new extensions are reviewed for technical correctness and appropriateness, and

- documentation for new extensions is complete and published.

9. Acknowledgements

The original form of this internet draft was copied directly from RFC1533 [1], written by Steve Alexander and Ralph Droms. Thanks to Mike Carney for his many helpful comments, as well as contributing the design of the Platform Specific Information and Platform Class Identifier. Thanks to Erik Guttman for his helpful suggestions for the Service Location extensions. Thanks to Ralph Droms, Matt Crawford, Thomas Narten, and Erik Nordmark for their careful review as part of the Last Call process.

10. Full Copyright Statement

References

[1] S. Alexander and R. Droms. DHCP Options and BOOTP Vendor Extensions. Request for Comments (Proposed Standard) 1533, Internet Engineering Task Force, October 1993.

[2] R. Atkinson. IP Authentication Header. Request for Comments (Proposed Standard) 1826, Internet Engineering Task Force, August 1995.

[3] T. Berners-Lee, L. Masinter, and M. McCahill. Uniform Resource Locators (URL). Request for Comments (Proposed Standard) 1738, Internet Engineering Task Force, December 1994.

[4] J. Bound, M. Carney, and C. Perkins. DHCP for IPv6. draft-ietf-dhc-dhcpv6-15.txt, May 2000. (work in progress).

[5] S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. Request for Comments (Best Current Practice) 2119, Internet Engineering Task Force, March 1997.

[6] B. Carpenter and Y. Rekhter. Renumbering Needs Work. Request for Comments (Informational) 1900, Internet Engineering Task Force, February 1996.

[7] M. Crawford, C. Huitema, and S. Thomson. DNS Extensions to Support IPv6 Address Aggregation and Renumbering. draft-ietf-ipngwg-dns-lookups-07.txt, 2000. (work in progress).

[8] S. Deering and R. Hinden. Internet Protocol, Version 6 (ipv6) Specification. Request for Comments (Draft Standard) 2460,

Internet Engineering Task Force, December 1998.

[9] E. Guttman, C. Perkins, and J. Kempf. Service Templates and Service: Schemes. Request for Comments (Proposed Standard) 2609, Internet Engineering Task Force, June 1999.

[10] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol, Version 2. Request for Comments (Proposed Standard) 2608, Internet Engineering Task Force, June 1999.

[11] S. Kent and R. Atkinson. IP Authentication Header. Request for Comments (Proposed Standard) 2402, Internet Engineering Task Force, November 1998.

[12] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. Request for Comments (Informational) 2104, Internet Engineering Task Force, February 1997.

[13] D. Mills. Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI. Request for Comments (Informational) 2030, Internet Engineering Task Force, October 1996.

[14] D. L. Mills. Internet time synchronization: The Network Time Protocol. Request for Comments 1129, Internet Engineering Task Force, October 1989.

[15] David L. Mills. Network Time Protocol (version 3) Specification, Implementation. Request for Comments (Draft Standard) 1305, Internet Engineering Task Force, March 1992.

[16] P. V. Mockapetris. Domain names - concepts and facilities. Request for comments (standard), Internet Engineering Task Force, November 1987.

[17] J. Reynolds and R. Braden. Internet Official Protocol Sandards. Request for comments (proposed standard), Internet Engineering Task Force, March 2000.

[18] R. Rivest. The MD5 Message-Digest Algorithm. Request for Comments (Informational) 1321, Internet Engineering Task Force, April 1992.

[19] S. Thomson and T. Narten. IPv6 Stateless Address Autoconfiguration. Request for Comments (Draft Standard) 2462, Internet Engineering Task Force, December 1998.

[20] J. Veizades, E. Guttman, C. Perkins, and S. Kaplan. Service Location Protocol. Request for Comments (Proposed Standard) 2165, Internet Engineering Task Force, June 1997.

[21] P. Vixie, Ed., S. Thomson, Y. Rekhter, and J. Bound. Dynamic Updates in the Domain Name System (DNS UPDATE). Request for Comments (Proposed Standard) 2136, Internet Engineering Task Force, April 1997.

Chair's Addresses

The working group can be contacted via the current chair:

Ralph Droms
Computer Science Department
323 Dana Engineering
Bucknell University
Lewisburg, PA 17837

Phone: (717) 524-1145
EMail: droms@bucknell.edu

Authors' Addresses

Questions about this memo can be directed to:

Jim Bound
Compaq Computer Corporation
Mail Stop: ZK03-3/U14
110 Spitbrook Road
Nashua, NH 03062
USA
Phone: +1-603-884-0400
Email: bound@zk3.dec.com

Mike Carney
Sun Microsystems, Inc
Mail Stop: UMPK17-202
901 San Antonio Road
Palo Alto, CA 94303-4900
USA
Phone: +1-650-786-4171
Email: mwc@eng.sun.com

Charles E. Perkins
Communications Systems Lab
Nokia Research Center
313 Fairchild Drive
Mountain View, California 94043
USA
Phone: +1-650 625-2986
EMail: charliep@iprg.nokia.com
Fax: +1 650 625-2502