          New Option Review Guidelines and Additional Option Namespace
              <draft-ietf-dhc-option-review-and-namespace-00.txt>

Status of this memo

Abstract

     This document outlines deficiencies that have become evident since
     RFC 2131 and RFC 2132 were published regarding the allocation of
     new option codes, the review of drafts covering these new option
     codes, and the availability of option codes for new parameters. The
     document then presents proposals for correcting these deficiencies.

1. Introduction

     The rapid and wide-spread adoption of DHCPv4 has lead to an
     increasing number of new DHCP option drafts under DHC WG review.
     Experience with the current IANA option code allocation process and
     the DHC WG option draft review process has identified a number of
     deficiencies, namely:

      * We're rapidly going through the remaining option codes, and face

the possibility of exhausting the remaining codes before the
wide-spread adoption of IPv6 is achieved.

* There are no guidelines to help the DHC WG and the DHCP
  community at large gauge the impact of the addition of new
  options. Some options that have been proposed require changes to

the DHCP protocol itself and/or current implementations. Because
the adoption of such options has a greater impact on the DHCP
community than other, simple "payload" options, these options
require more scrutiny by the DHC WG and IESG.

* Since some options could change the DHCP protocol itself, we
  need a method of explicitly communicating the change of DHCP
  versions among implementations. Today, we have no such method.

* There is no provision to preserve compatibility with earlier
  versions of the protocol.

Interoperability testing at Connectathon (1997, 1998, and 1999)
has shown a reduction in the level of interoperability between
implementations. These interoperability problems were found to
be due to confusion among implementors about how certain features
of the protocol should be implemented. Improvement (tightening)
of the general RFC 2131 and RFC 2132 drafts as well as the
tightening of new option specifications (using the guidelines
defined in this document) will help increase the level of
interoperability.

1.1 Conventions Used in the Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY" and "OPTIONAL" in this
document are to be interpreted as described in RFC 2119 [5].

1.2 Terminology

RFC 2132-form options   The RFC 2132 [2] option block form is
                        currently in use by DHCPv4, and is
                        documented in RFC 2131 [1]. This option
                        form is identified by the BOOTP magic
                        cookie {99, 130, 83, 99}.

RFC TBD-form options   A new option block form recommendation
                        to alleviate option code exhaustion
                        (described in section 3.6 below). This

option form will be identified by a TBD
BOOTP magic cookie issued by IANA.

2. Option Review Guidelines

We tackle the option code review problem by defining a set of
option categories based upon upon the impact the adoption of an
option will have on the DHCP community. Option drafts appropriately
categorized aid reviewers by helping them evaluate the draft. Once
the DHC WG and the draft author(s) agree on the category of the
proposed option, that category will be listed explicitly in the
abstract of the option draft.

2.1. General Option Draft Review Guidelines

   A) This option has no relationship to other existing or proposed
      options. It would not require change of existing DHCP client,

      server, or BOOTP relay agent implementations. It would not
      change the version of the DHCP protocol. Its introduction
      would not invalidate previous version(s) of the DHCP protocol.
      Is this option better handled by the Service Location
      Protocol (SLP) [7]? If it provides data which is unrelated
      to network interface configuration, then the option proposer
      SHOULD consider the use of SLP for the delivery of this data,
      if SLP is deployed widely enough to meet her needs.

   B) This option has no relationship to other existing or proposed
      options. It would not require change of existing DHCP
      client, server, or BOOTP relay agent implementations. It
      would not change the version of the DHCP protocol. Its
      introduction would not invalidate previous version(s) of the
      DHCP protocol. It isn't a candidate for SLP. Is this option
      useful to the DHCP community at large (e.g. implementors) or
      is it specific to a particular implementation? If it is the
      latter, then the proposer of the option should consider using
      the encapsulated vendor space available to your
      implementation to encode this information. Note that most
      DHCP implementations only support default option types for
      encapsulated options. (see Section 3.5 below).

   C) This option has no relationship to other existing or proposed
      options. It would not require change of existing DHCP
      client, server, or BOOTP relay agent implementations. It
      would not change the version of the DHCP protocol. Its
      introduction would not invalidate previous version(s) of the

DHCP protocol. It is not a candidate for SLP or the vendor's option space. Can this option be effectively encoded in the default option type ([Section 3.5](#) below)? For example, can it be communicated as a single ASCII string? A list of IP addresses? If so, then one of the default formats should be used.

D) This option would have a implicit or explicit relationship between it and other existing options or other proposed options, or the data it represents cannot be carried in a default option type ([Section 3.5](#) below). It MAY change the behavior of existing DHCP client, server, and/or BOOTP relay agent implementations. It would not change the DHCP protocol version. Its introduction would not invalidate previous version(s) of the DHCP protocol.

   Examples of implicit/explicit option relationships:

   Option   Related Options
   ------   ---------------
   Client Class   Encapsulated vendor option(s)
   User Class   Standard option's scope

E) The addition of this option would change Table 3, "Fields and options used by DHCP servers" and/or Table 5, "Fields and options used by DHCP clients" in [RFC 2131](#) [[1](#)], and thus change the DHCP protocol. Pre-existing versions / implementations would continue to interoperate.

F) The addition of this option would invalidate previous versions of the DHCP protocol, preventing client, server, and/or BOOTP relay agents implementing the earlier version(s) from functioning.

   Guidelines   Option Category
   ----------   ---------------
     A   None. Use SLP to register and
                           deliver your information.

     B   None. Use your Vendor-specific
                           option space for your option.

     C   Category One

     D   Category Two

E   Category Three

                F   Category Four

2.2. Category One Options

    Options in this category MUST NOT require changes to the DHCP
    protocol, server, client, or BOOTP relay agent implementations.
    They MAY be either RFC 2132-form or RFC TBD-form options. They MUST
    NOT be dependent on other options being present or absent. Earlier
    versions/implementations of the protocol continue to interoperate
    in the presence of these options. Administrative tools and DHCP
    protocol debugging tools which generically support the default
    option types MAY need to be reconfigured in order to permit
    management of the new option. Options of this type are "payload"
    options, and MUST be of one of the default option types for the
    option block form (RFC 2132 or RFC TBD).

    Although category one options do not require DHCP protocol / DHCP
    implementation interoperability testing, they do require
    interoperability testing between the programs(s) which ultimately
    will consume the information contained within the option value.

    Acceptance criteria:

        Working group/IETF community review: Yes.
        IANA option number registration: Yes.
        Interoperability testing (2 or more implementations)   No.
        DHCP protocol version change: No.

2.3. Category Two Options

    Options in this category MUST NOT require changes to the DHCP
    protocol. They MAY require changes to server, client, relay agent
    implementations, administrative tools, and DHCP protocol debugging
    tools. They MAY depend on the presence or absence of other options,
    as long as those other options are NOT in Table 3 or Table 5 of

    RFC 2131 [1]. Any dependence on other options MUST be made
    explicit in the new options draft. Existing versions /
    implementations of the protocol continue to interoperate in the
    presence of messages containing category two options. Options of
    this type are "affect implementation" options.

An option MUST be designed in such a way as a reply/response from
non-compliant implementations can be easily distinguished from
those of compliant implementations. An option MUST NEVER change the
interpretation of existing options. The option draft author MUST
specify a compliant implementation's behavior if that implement-
ation receives a reply/response from a non-compliant implementation.

Acceptance criteria:

    Working group/IETF community review: Yes.
    IANA option number registration: Yes.
    Interoperability testing (2 or more implementations)   Yes.
    DHCP protocol version change: No.

2.4. Category Three Options

Options in this category EXPLICITLY change the DHCP protocol. They
WILL require changes to server, client, and/or relay agent
implementations. They MAY depend on the presence or absence of
other options. Any dependence on other options MUST be made
explicit in the new option draft. The addition of such options
result in changes to Table 3, "Fields and options used by DHCP
servers" and/or Table 5, "Fields and options used by DHCP clients"
in RFC 2131 [1]. Existing versions / implementations of the
protocol continue to interoperate in the presence of traffic
containing category three options. Administrative tools MUST be
changed to support options of this type. DHCP protocol debugging
tools would need to be updated to recognize these options. Options
of this type are known as "affect protocol" options. The acceptance
of a Category Three option results in incrementing the DHCP version
option value.

An option MUST be designed in such a way as a reply/response from
non-compliant implementations can be easily distinguished from
those of compliant implementations. The option draft author MUST
specify a compliant implementation's behavior if that implement-
ation receives a reply/response from a non-compliant
implementation. An option MUST NEVER change the interpretation of
existing options. Category Three option implementations can easily
detect a non-compliant implementation due to the absence of the
DHCP version option or a lower than expected version number.

Acceptance criteria:

    Working group/IETF community review: Yes.
    IANA option number registration: Yes.
    Interoperability testing (2 or more implementations)   Yes.
    DHCP protocol version change: Yes.

2.5. Category Four Options

Options in this category would EXPLICITLY change the DHCP
protocol in a non-backward compatible manner. They would require
changes to ALL DHCP client, server, and/or BOOTP relay agent
implementations. They INVALIDATE one or more of the previous
versions of the BOOTP/DHCP protocol.

Because category four options invalidate previous versions of the
protocol, they are NOT candidates for acceptance. Changes to the
the DHCP protocol MUST BE backward compatible.

3. Specification of DHCP protocol version and DHCP option block form.

We define two RFC 2132-form options. The first is used by a DHCP
client to request a response in a certain option block form. The
second option is used by both DHCP clients and servers to
identify the version of protocol used in the messages generated.

3.1. DHCP Option Block Request Option

This option is used by RFC TBD client implementations to request
RFC TBD-form replies from RFC TBD server implementations. This
option is included in RFC 2132-form DHCPDISCOVERs, INIT-REBOOT
DHCPREQUESTs, and DHCPINFORMs to request RFC TBD-form replies.

This option MUST NEVER be used by DHCP servers in their responses
to DHCP clients.

The DHCP Option Block Request Option is of the RFC 2132-form, and
consists of a single 4 octet string holding the magic cookie of
the desired option block form.

```
code len   magic cookie
+---+---+---------------+
| X | 4 |      ...      |
+---+---+---------------+
```

3.2. DHCP Version Option

This option represents the DHCP version. DHCP messages without
this option are RFC 2131 [1]. The addition of Category Three
option(s) (perhaps grouped together) result in a DHCP version
change. All implementations supporting versions greater than RFC
2131 MUST include the DHCP version option in all messages,
generated by both client or server.

The DHCP Protocol Version Option is of the [RFC 2132]-form, and
consists of an single unsigned 16bit integer. The first Category
Three option accepted by the DHC WG will be assigned version zero
(0).

```
code len    value
+---+---+---------+
| Y | 2 |(0-65535)|
+---+---+---------+
```

DHCP protocol versions MUST be backward compatible with earlier
versions.

3.3 Allocation of options from [RFC 2132] or RFC TBD option name spaces

Allocation of new options from the RFC TBD name space can only
occur if we have two (2) or more RFC TBD option name space
implementations which have demonstrated interoperability. Such
interoperability testing could occur at the next Connectathon
event (held late February/early March every year).

The selection of which name space from which to allocate an option
is left to the DHC WG and IESG, following the process outlined in
Ralph Drom's "Procedure for Defining New DHCP Options" [4].

3.4. [RFC 2132] Option Block Form

For convenience, I have extracted the description of [RFC 2132]-form
options from [2].

```
code len    value
+---+---+-------
|   |   |    ...
+---+---+-------
```

Options of this form consist of three single octet fields: code,
len, and value.

The code value is a single octet (0-255). 0 (pad) and 255 (end)
are the only codes that do not follow the code, len, value form.
Len is a single octet (0-255) describing the length of the value
field. Value contains the data payload.

3.5. [RFC 2132]-form Default Option Types

1) Single variable-length ASCII string.

2) Single variable-length OCTET string.

3) 1-X Multiple of NUMBERs (1, 2, 4, 8 octets each). May consist of
   a LIST where the list entries are defined as multiple NUMBERs.

      Example: "A variable-length list of 2 32bit NUMBERs
                representing the offset and length of a
                block of data"

4) 1-X Multiple of IPv4 addresses. May consist of a LIST where the
   list entries consist of multiple IP addresses.

      Example: "A list of static routes (destination, router), ..."

If an implementation encounters RFC 2132-form options which it does
not understand, it will simply skip over the option (using that
option's len field) and continue option processing. This behavior
ensures that RFC 2131 compliant implementations safely ignore new
options (e.g. the DHCP version option). Current DHCP

implementations SHOULD ignore any non-RFC 2132-form option block.

3.6. Proposed RFC TBD Option Block Form

We propose the creation of the following option block form
described below, which greatly increases the option space and size
of individual options. We feel that the adoption of this option
block form will resolve the approaching RFC 2132 standard option
code availability problem.

```
     Code      Len            Data
  +---------+---------+------------------
  | uint16_t| uint16_t|                   ...
  +---------+---------+------------------
```

Code value is a network-order double-octet unsigned value (0-
65535). 0 (pad) and 65535 (end) are the only codes that do not
follow the code, len, value form. Len is a network-order
double-octet unsigned value (0-65535). Data contains Len octets.

The first 255 RFC TBD option codes are preassigned for mapping
RFC 2132 options into RFC TBD option space. Because the size limit
of options within the RFC TBD option space is much higher than
that of options within the RFC 2132 option space, RFC TBD

implementations MUST use care in responding to RFC 2132 clients. If
the data for a mapped RFC 2132 option would exceed the 253 length
restriction of RFC 2132 clients, the RFC TBD implementation MUST
drop the option from the reply, and SHOULD increment an error count
or log a message regarding this event. Note that the creation of
multiple instances of RFC 2132 options in order to provide all of
the data within the configured RFC TBD option is not workable, since
by definition options within the RFC 2132 (and RFC TBD) option space
are position-independent, and thus deterministic ordering of data
is not possible.

3.7. RFC TBD Default Option Types

    All option types as listed in Section 3.5, and:

        1) UNICODE string.

        2) IPv6 addresses(s). Multiple of 16 octets. Encoded within an
           implementation's configuration table using the text
           representation as defined by Section 2.2 of RFC 2373 [6].

        3) Encapsulated RFC TBD options. For example, New User class,
           Vendor class-style options. Encapsulated options follow same
           code, len, value form of RFC TBD options. Overall length
           specifies limit of encapsulation.

        4) Mixed-type options. These options are made up of combinations
           of 3.5/3.7 typed fields. Variable-length typed fields are
           preceded by a two octet unsigned integer length field. Mixed
           type options are useful for encoding combinations of data
           (structures).

        Example: The following option contains an NVT ASCII string
        followed by a boolean, followed by a 32bit unsigned integer.
           Code   Len   Data
        +-------+-------+-------+------------+---+--------------+
        | 771 | 17 | 13 |Hello, World!| 1 | 4123782 |
        +-------+-------+-------+------------+---+--------------+

        5) BOOLEAN. A single octet value which is explicitly TRUE if
           non-zero, explicitly FALSE if zero (0). Len is always one.

4. Behavior of RFC 2132 and RFC TBD Client and Server Implementations

4.1 Client Behavior

A DHCP client in the INIT state prefers DHCP replies matching the
DHCP option block form (RFC 2132 or RFC TBD) and DHCP protocol
version of its DHCPDISCOVER. Option block form is preferred over
DHCP protocol version.

DHCP clients SHOULD always indicate the size of response they
can accept using the DHCP MTU option.

4.1.1 INIT/INFORM states

A client in one of these states pauses to collect replies (DHCP
of any type/version or BOOTP replies) for a tunable polling
interval (default: 5 seconds). Retransmissions use the algorithm
outlined in Section 4.1 of RFC 2131 [1].

Existing RFC 2132-form clients require no change in their behavior
as specified in RFC 2131 [1].

RFC 2132-form clients who support a later version of the DHCP
protocol include the DHCP version option identifying the version of
DHCP protocol they support. They prefer replies matching their
protocol version over older versions of DHCP and BOOTP replies. If
a RFC 2132-form client selects a reply from a down-rev DHCP protocol
server (or BOOTP server), then that client MAY update an error count
or log this event.

A RFC TBD-form client in one of these states forms the appropriate
RFC 2132-form message (DHCPDISCOVER, DHCPINFORM) and includes the
DHCP Option Block Request Option (set to the RFC TBD magic cookie)
and the DHCP version option if it supports a version greater than
that specified in RFC 2131, and sends this message in the manner
appropriate for the message type. A RFC TBD-form client
prefers RFC TBD-form responses over RFC 2132-form responses. Within
responses of a certain option form, responses which match the
client's DHCP protocol version are preferred over those responses
that are down-rev. If no DHCP responses are received, the client
MAY choose a BOOTP response if such a response is available.

In the case of an INIT state client, the DHCPREQUEST used to accept
the DHCPOFFER is formed using the option block form and DHCP
protocol version of the selected DHCPOFFER.

For the duration of the lease, the client MUST form request messages
using the option block form and DHCP protocol version of the

original DHCPACK (INIT/INFORM) states when verifying existing
configuration (INIT-REBOOT DHCPREQUEST), requesting lease
extensions (DHCPREQUEST), declining an IP address (DHCPDECLINE),
or releasing an IP address (DHCPRELEASE).

In summary, a RFC TBD-form DHCP client will prefer RFC TBD-form
replies over RFC 2132-form replies. Within replies which match the
selected option block form, a client will prefer replies which match
its DHCP protocol version. All subsequent traffic is done using the
option block form and DHCP protocol version of the accepted
DHCPOFFER/DHCPACK.

4.2 Server Behavior

4.2.1 Option Block Format-independent Server Behavior

A server which receives a BOOTP magic cookie it does not understand
SHOULD respond with a standard BOOTP reply ONLY if it is explicitly
configured to do so. Note that RFC TBD-form requests will appear to
RFC 2131 servers as BOOTP requests with an unrecognized magic
cookie. In such environments, the RFC 2131 server's ability to
respond to BOOTP clients SHOULD be restricted, and RFC TBD servers
SHOULD be configured to respond to BOOTP clients.

If a DHCPDISCOVER/DHCPINFORM is received which contains a DHCP
protocol version option, a server MUST:

   a) If the DHCP protocol version is down-rev of that supported by
      the server, the server will format all messages in that
      earlier version. If the DHCP server does not support the
      earlier version of the protocol, it MUST drop the request,
      and MAY update an error counter or log the event.

   b) If the DHCP protocol version is greater than that supported
      by the server, it MAY either drop the request or respond to
      the client using messages of the DHCP protocol version
      supported by the server. This behavior SHOULD be configurable
      by the administrator. The server SHOULD choose to delay
      responding to the up-rev client for a configurable number of
      seconds, in order to give potential higher-rev servers a
      chance to reply.

   c) If the DHCP protocol version matches that of the server, then
      the server responds immediately (if possible) to the client
      with the appropriately formatted reply.

4.2.2 RFC TBD-form Server Behavior

A RFC TBD-form server can identify a RFC TBD-form client by the
presence of the DHCP Option Block Request Option with the RFC TBD
magic cookie in the option value field in RFC 2132-form requests

or RFC TBD-form requests identified by the RFC TBD-form magic
cookie. For the following scenarios, assume a RFC TBD-form server
has free resources available for allocation.

   a) If a RFC TBD-form response is requested, it MUST respond with
      a RFC TBD-form response if it chooses to respond at all.

   b) If a RFC 2132-form response is requested (no DHCP Option Block
      Request Option is present) and the RFC TBD-form server is
      explicitly configured to respond to RFC 2132-form clients, it
      MUST respond with a RFC 2132-form response if it chooses to
      respond at all.

   c) If a request is received which contains a DHCP Option Block
      Request Option with a value it does not recognize, the server
      MUST either drop the request or respond with a RFC 2132-form
      response. Its behavior in this situation SHOULD be an
      administrator-configured option.

   A DHCP server MUST NOT use the DHCP Option Block Request Option in
   the messages it generates for DHCP clients.

5. Open Issues

   The author would appreciate comments/feedback regarding these issues.

    * Parameter request option in RFC 2132-form requests.
       A RFC 2132-form DISCOVER/REQUEST can only request options from
       the RFC 2132 name space. One possible solution is for a
       RFC TBD-form implementation to generate a RFC TBD-form DISCOVER
       or REQUEST containing the full parameter request list if it
       receives RFC TBD-form replies. The downside of this approach is
       the extra DHCP traffic. I propose that RFC TBD-form implementors
       include a policy mechanism which turns on pure RFC TBD-form and
       thus disables the RFC 2132-form request method. Such behavior
       might be controlled by a RFC 2132-form option, for example.

    * RFC TBD-form requests look like BOOTP requests with an
       unrecognized magic cookie. Disabling BOOTP compatibility in
       RFC 2132 servers and turning it on in RFC TBD servers (which
       can appropriately recognize the clients) seems like a reasonable
       approach.

6. Security Considerations

Not Applicable.

7. Acknowledgements

The author would like to gratefully acknowledge the active
participation of the following DHCP future panel members:
Ralph Droms, Kester Fong, Pratik Gupta, Barr Hibbs, Kim Kinnear,
Ted Lemon, Nathan Lane, and Glenn Waters. The author would also
like to thank Thomas Narten for his review comments.

8. Copyright

Copyright (C) The Internet Society (1999). All Rights Reserved.

9. References

[1] Droms, R. "Dynamic Host Configuration Protocol", RFC 2131,
Bucknell University, March 1997.
<ftp://ds.internic.net/rfc/rfc2131.txt>

[2] Alexander, S. and Droms, R., "DHCP Options and BOOTP
Vendor Extension", RFC 2132, March 1997.
<ftp://ds.internic.net/rfc/rfc2132.txt>

[3] Prindeville, P. "BOOTP Vendor Information Extensions", RFC 1048,
McGill University, February 1988.
<ftp://ds.internic.net/rfc/rfc1048.txt>

[4] Droms, R. "Procedure for Defining New DHCP Options",
INTERNET-DRAFT, August 1998
<ftp://www.ietf.org/internet-drafts/draft-ietf-dhc-new-options-02.txt>

[5] Bradner, "Key words for use in RFCs to Indicate Requirement
Levels", RFC 2119, Harvard University, March 1997.
<ftp://ds.internic.net/rfc/rfc2119.txt>

[6] Hinden R. and Deering, S., "IP Version 6 Addressing
Architecture", RFC 2373, Nokia and Cisco Systems, July 1998.
<ftp://ds.internic.net/rfc/rfc2373.txt>

[7] Guttman E., Perkins C., Veizades J., and Day M., "Service
Location Protocol, Version 2", April 1999.
<ftp://www.ietf.org/internet-drafts/draft-ietf-svrloc-protocol-v2-16.txt>

10. Author's Address

Michael Carney
Sun Microsystems, Inc.
One Network Drive
Burlington, MA 01803

Phone: (781) 442-0469
Fax: (781) 442-1677
Email: Michael.Carney@sun.com

11. Expiration

This document will expire on December 31, 1999.