Rules For Designing Protocols Using the RFC 5444 Generalized Packet/
Message Format
draft-ietf-manet-rfc5444-usage-02

Abstract

This document updates the generalized MANET packet/message format,
specified in RFC 5444, by providing rules and recommendations for how
protocols can use that packet/message format. In particular, the
mandatory rules prohibit a number of uses of RFC 5444 that have been
suggested in various proposals, and which would have led to
interoperability problems, to the impediment of protocol extension
development, and to an inability to use generic RFC 5444 parsers.

Status of this Memo

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

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

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

This Internet-Draft will expire on September 1, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the
document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal
Table of Contents

1. Introduction .................................................... 3
   1.1. History and Purpose ........................................ 3
   1.2. RFC 5444 Features ........................................... 3
   1.3. Status of This Document .................................... 5
2. Terminology ..................................................... 5
3. Applicability Statement ......................................... 5
4. Information Transmission .......................................... 6
   4.1. Where to Record Information ................................. 6
   4.2. Packets and Messages ....................................... 7
   4.3. Messages, Addresses and Attributes ......................... 9
   4.4. Addresses Require Attributes ................................ 9
   4.5. Information Representation .................................. 11
   4.6. TLVs ......................................................... 12
   4.7. Message Integrity ........................................... 13
5. Structure ........................................................ 13
6. Message Efficiency .............................................. 14
   6.1. Address Block Compression .................................. 14
   6.2. TLVs ......................................................... 15
   6.3. TLV Values .................................................. 16
   6.4. Automation .................................................. 17
7. Security Considerations ......................................... 17
8. IANA Considerations .............................................. 17
9. Acknowledgments .................................................. 17
10. References ...................................................... 18
    10.1. Normative References ....................................... 18
    10.2. Informative References ..................................... 18
Authors’ Addresses .................................................. 19
1. Introduction

[RFC5444] specifies a generalized packet/message format, designed for use by MANET routing protocols.

[RFC5444] was designed following experiences with [RFC3626], which attempted, but did not quite succeed in, providing a packet/message format accommodating for diverse protocol extensions. [RFC5444] was designed as a common building block for use by both proactive and reactive MANET routing protocols.

[RFC5498] mandates the use of this packet/message format, and of the packet multiplexing process described in an Appendix to [RFC5444], by protocols operating over the manet IP protocol and port numbers that were allocated following [RFC5498].

1.1. History and Purpose

Since the publication of [RFC5444] in 2009, several RFCs have been published, including [RFC5497], [RFC6130], [RFC6621], [RFC7181], [RFC7182], [RFC7183], [RFC7188], and [RFC7631], that use the format of [RFC5444]. The ITU-T recommendation [G9903] also uses the format of [RFC5444] for encoding some of its control signals. In developing these specifications, experience with the use of [RFC5444] has been acquired, specifically with respect to how to write specifications using [RFC5444] so as to ensure "forward compatibility" of a protocol with future extensions, to enable the creation of efficient messages, and to enable the use of an efficient and generic parser for all protocols using [RFC5444].

During the same time period, other suggestions have been made to use [RFC5444] in a manner that would inhibit the development of interoperable protocol extensions, would potentially lead to inefficiencies, or would lead to incompatibilities with generic parsers for [RFC5444]. While these uses were not all explicitly prohibited by [RFC5444], they should be strongly discouraged. This document is intended to prohibit such uses, to present experiences from designing protocols using [RFC5444], and to provide these as guidelines (with their rationale) for future protocol designs using [RFC5444].

1.2. RFC 5444 Features

Among the characteristics, and design criteria, of the packet/message format of [RFC5444] are:

- It is designed for carrying MANET routing protocol control signals.
- It defines a packet as a Packet Header with a set of Packet TLVs (Type-Length-Value structures), followed by a set of messages. Each message has a well-defined structure consisting of a Message Header (designed for making processing and forwarding decisions) followed by a set of Message TLVs, and a set of (address, type, value) associations using Address Blocks and their Address Block TLVs. The [RFC5444] packet/message format then enables the use of simple and generic parsing logic for Packet Headers, Message Headers, and message content.

A packet may include messages from different protocols, such as [RFC6130] and [RFC7181], in a single transmission. This was observed in [RFC3626] to be beneficial, especially in wireless networks where media contention may be significant. [RFC5444] defines a multiplexing process to achieve this that is mandated by [RFC5498] for use on the manet IP port and UDP port. This makes the contents of the Packet Header, which may also contain Packet TLVs, and the transmission of packets over UDP or directly over IP, the responsibility of this multiplexing process.

- Its packets are designed to travel between two neighboring interfaces, which will result in a single decrement/increment of the IPv4 TTL or IPv6 hop limit. The Packet Header and any Packet TLVs should convey information relevant to that link (for example, the Packet Sequence Number can be used to count transmission successes across that link). Packets are not retransmitted, a packet transmission following a successful packet reception may include all, some, or none of the received messages, plus possibly additional messages received in separate packets or generated at that router. Messages may thus travel more than one hop, and are designed to carry end-to-end protocol signals.

- It supports "internal extensibility" using TLVs; an extension can add information to an existing message without that information rendering the message unparseable or unusable by a router that does not support the extension. An extension is typically of the protocol that created the message to be extended, for example [RFC7181] adds information to the HELLO messages created by [RFC6130]. However an extension may also be independent of the protocol, for example [RFC7182] can add ICV ( Integrity Check Value) and timestamp information to any message (or to a packet, thus extending the [RFC5444] multiplexing process).

Information can be added to the message as a whole, such as the [RFC7182] integrity information, or may be associated with specific addresses in the message, such as the MPR selection and link metric information added to HELLO messages by [RFC7181]. An extension may also add addresses to a message.
It uses address aggregation into compact Address Blocks by exploiting commonalities between addresses. In many deployments, addresses (IPv4 and IPv6) used on interfaces share a common prefix that need not be repeated. Using IPv6, several addresses (of the same interface) may have common interface identifiers that need not be repeated.

It sets up common namespaces, formats, and data structures for use by different protocols, where common parsing logic can be used. For example, [RFC5497] defines a generic TLV format for representing time information (such as interval time or validity time).

It contains a minimal Message Header (a maximum of five elements: type, originator, sequence number, hop count and hop limit) that permit decisions whether to locally process a message, or forward a message (thus enabling MANET-wide flooding of a message) without processing the body of the message.

1.3. Status of This Document

This document updates [RFC5444], and is intended for publication as a Proposed Standard (rather than as Informational) because it specifies and mandates constraints on the use of [RFC5444] which, if not followed, makes forms of extensions of those protocols impossible, impedes the ability to generate efficient messages, or makes desirable forms of generic parsers impossible.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This document uses the terminology and notation defined in [RFC5444], in particular the terms "packet", "Packet Header", "message", "Message Header", "address", "Address Block", "TLV" and "TLV Block" are to be interpreted as described therein.

3. Applicability Statement

This document does not specify a protocol, but documents constraints on how to design protocols which are using the generic packet/message format defined in [RFC5444] which, if not followed, makes forms of extensions of those protocols impossible, impedes the ability to
generate efficient (small) messages, or makes desirable forms of
generic parsers impossible. The use of this format is mandated by
[RFC5498] for all protocols running over the manet protocol and port
number, defined therein. Thus, the constraints in this document
apply to all protocols running over the manet protocol and port
number.

4. Information Transmission

Protocols need to transmit information from one instance implementing
the protocol to another.

4.1. Where to Record Information

A protocol has the following choices as to where to put information
for transmission:

- In a TLV to be added to the Packet Header.
- In a message of a type owned by another protocol.
- In a message of a type owned by the protocol.

The first case (a Packet TLV) can only be used when the information
is to be carried one hop. It SHOULD only be used either where the
information relates to the packet as a whole (for example packet
integrity check values and timestamps, as specified in [RFC7182]) or
if the information is of expected wider application than the single
protocol. A protocol can also request that the Packet Header include
Packet Sequence Numbers, but does not control those numbers.

The second case (in a message of a type owned by another protocol) is
only possible if the adding protocol is an extension to the owning
protocol; for example OLSRv2 [RFC7181] is an extension of NHDP
[RFC6130]. While this is not the most common case, protocols SHOULD
be designed to enable this to be possible, and most rules in this
document are to help facilitate that. An extension to [RFC5444],
such as [RFC7182], is considered to be an extension to all protocols
in this regard.

The third case is the normal case for a new protocol. Protocols MUST
be conservative in the number of new message types that they require,
as the total available number of allocatable message types is only
224. Protocol design SHOULD consider whether different functions can
be implemented by differences in TLVs carried in the same Message
Type, rather than using multiple Message Types. If a protocol’s
needs can be covered by use of the second case, then this SHOULD be
TLV space, although greater than message space, SHOULD also be used efficiently. The extended type of a TLV occupies two octets, thus there are many more available TLVs. However, in some cases (currently LINK_METRIC from [RFC7181] and ICV and TIMESTAMP from [RFC7182] in the global TLV space) a full set of 256 TLVs is defined (but not necessarily allocated). Each message also has a block of message specific TLV Types (128 to 233, each with 256 type extensions), these SHOULD be used in preference to the common TLV Types (0 to 127, each with 256 type extensions) when a TLV is message-specific.

A message contains a Message Header and a Message Body; note that the Message TLV Block is considered as part of the latter. The Message Header contains information whose primary purpose is to decide whether to process the message, and whether to forward the message.

A message MUST be recognized by the combination of its type, Originator Address and Message Sequence Number. This allows each protocol to manage its own Message Sequence Numbers, and also allows for the possibility that different Message Types may have greatly differing transmission rates. [RFC7181] contains a general purpose process for managing processing and forwarding decisions, albeit one presented as for use with MPR flooding. (Blind flooding can be handled similarly by assuming that all other routers are MPR selectors; it is not necessary in this case to differentiate between interfaces on which a message is received.)

Most protocol information is thus contained in the Message Body. A model of how such information may be viewed is described in Section 4.3 and Section 4.4. To use that model, addresses (for example of neighboring or otherwise known routers) SHOULD be recorded in Address Blocks, not as data in TLVs. Recording addresses in TLV Value fields both breaks the model of addresses as identities and associated information (attributes) and also inhibits address compression. However in some cases alternative addresses (e.g., hardware addresses when the Address Block is recording IP addresses) MAY be carried as TLV Values. Note that a message contains a Message Address Length field that can be used to allow carrying alternative message sizes, but only one length of addresses can be used in a single message, in all Address Blocks and the Originator Address.

4.2. Packets and Messages

The [RFC5444] multiplexing process has to handle packet reception and message demultiplexing, and message transmission and packet multiplexing.
When a packet arrives, the following steps are required:

- The packet and/or the messages it contains MAY be verified by an extension to the demultiplexer, such as [RFC7182].
- Each message MUST be sent to its owning protocol, which MAY also view the Packet Header, and the source address in the IP datagram that included the packet.
- The owning protocol SHOULD verify each message, it SHOULD allow any extending protocol(s) to also contribute to this.
- The owning protocol MUST process each message, or make an informed decision not to do so. In the former case an owning protocol that permits this MUST allow any extending protocols to process or ignore the message.

Packets are formed for transmission by:

- Outgoing messages are created by their owning protocol, and MAY be modified by any extending protocols if the owning protocol permits this. Messages MAY also be forwarded by their owning protocol. It is RECOMMENDED that messages are not modified in the latter case.
- Outgoing messages are then sent to the [RFC5444] multiplexing process. The owning protocol MUST indicate which interface(s) the messages are to be sent on and their destination address, and MAY request that messages are kept together in a packet; the multiplexing process SHOULD respect this request if possible. A protocol MAY also request that a Packet Sequence Number and/or specified Packet TLVs are included, such requests SHOULD also be respected if possible.
- The multiplexing process SHOULD combine messages from multiple protocols that are sent on the same interface in a packet, provided that in so doing the multiplexing process does not cause an IP packet to exceed the current MTU (Maximum Transmission Unit). (Note that the multiplexing process cannot fragment messages; creating suitable sized messages is the responsibility of the protocol.)
- If requested by a protocol, the multiplexer SHOULD, and otherwise MAY, include a Packet Sequence Number in the packet. Note that, as per the errata to [RFC5444], this Packet Sequence Number MUST be specific to the interface on which the packet is sent.
o An extension to the multiplexing process MAY add TLVs to the
  packet and/or the messages (for example as by [RFC7182]).

4.3. Messages, Addresses and Attributes

The information in a Message Body, including Message TLVs and Address
Block TLVs, can be considered to consist of:

  o Attributes of the message, each attribute consisting of an
    extended type, a length, and a value (of that length).

  o A set of addresses, carried in one or more Address Blocks.

  o Attributes of each address, each attribute consisting of an
    extended type, a length, and a value (of that length).

Attributes are carried in TLVs. For Message TLVs the mapping from
TLV to attribute is one to one. For Address Block TLVs the mapping
from TLV to attribute is one to many, one TLV can carry attributes
for multiple addresses, but only one attribute per address.
Attributes for different addresses may be the same or different.

A TLV extended type MAY be (and this is RECOMMENDED whenever
possible) defined so that there may only be one TLV of that extended
type associated with the packet (Packet TLV), message (Message TLV),
or any value of any address (Address Block TLV). Note that an
address may appear more than once in a message, but the restriction
on associating TLVs with addresses covers all copies of that address.
It is RECOMMENDED that addresses are not repeated in a message.

4.4. Addresses Require Attributes

It is not mandatory in [RFC5444] to associate an address with
attributes using Address Block TLVs. Information about an address
could thus, in principle, be carried using:

  o The simple presence of an address.

  o The ordering of addresses in an Address Block.

  o The use of different meanings for different Address Blocks.

This specification, however, requires that those methods of carrying
information MUST NOT be used for any protocol using [RFC5444].
Information about the meaning of an address MUST only be carried
using Address Block TLVs.

In addition, rules for the extensibility of OLSRv2 and NHDP are
described in [RFC7188]. This specification extends their applicability to other uses of [RFC5444].

These rules are:

- A protocol MUST NOT assign any meaning to the presence or absence of an address, to the ordering of addresses in an Address Block, or to the division of addresses among Address Blocks.

- A protocol MUST NOT reject a message based on the inclusion of a TLV of an unrecognized type. The protocol MUST ignore any such TLVs when processing the message. The protocol MUST NOT remove or change any such TLVs if the message is to be forwarded unchanged.

- A protocol MUST NOT reject a message based on the inclusion of an unrecognized Value in a TLV of a recognized type. The protocol MUST ignore any such Values when processing the message, but MUST NOT ignore recognized Values in the such a TLV. The protocol MUST NOT remove or change any such TLVs if the message is to be forwarded unchanged.

- Similar restrictions to the two preceding points apply to the packet multiplexing process, which also MUST NOT reject a packet based on an unrecognized message; although it will reject any such messages, it MUST deliver any other messages in the packet to their owning protocols.

The following points indicate the reasons for these rules, based on considerations of extensibility and efficiency.

Assigning a meaning to the presence, absence or location, of an address would reduce the extensibility of the protocol, prevent the approach to information representation described in Section 4.5, and reduce the options available for message optimization described in Section 6.

For example, consider NHDP’s HELLO messages [RFC6130]. The basic function of a HELLO message is to indicate that an address is of a neighbor, using the LINK_STATUS and OTHER_NEIGHB TLVs. An extension to NHDP might decide to use the HELLO message to report that, for example, an address is one that could be used for a specialized purpose, but not for normal NHDP-based purposes. Such an example already exists (but within the basic specification, rather than as an extension) in the use of LOST Values in the LINK_STATUS and OTHER_NEIGHB TLVs to report that an address is of a router known not to be a neighbor. A future example might be to list an address to be added to a "blacklist" of addresses not to be used. This would be indicated by a new TLV (or a new Value of an existing TLV, see
An unmodified extension to NHDP would ignore such addresses, as required, as it does not support that specialized purpose. If NHDP had been designed so that just the presence of an address indicated a neighbor, that extension would not have been possible.

Rejecting a message because it contains an unrecognized TLV type, or an unrecognized TLV Value, reduces the extensibility of the protocol.

For example, OLSRv2 [RFC7181] is, among other things, an extension to NHDP. It adds information to addresses in an NHDP HELLO message using a LINK_METRIC TLV. A non-OLSRv2 implementation of NHDP, for example to support Simplified Multicast Flooding (SMF) [RFC6621], must still process the HELLO message, ignoring the LINK_METRIC TLVs.

Also, the blacklisting described in the example above could be signaled not with a new TLV, but with a new Value of a LINK_STATUS or OTHER_NEIGHB TLV (requiring an IANA allocation as described in [RFC7188]), as is already done in the LOST case.

The creation of Multi-Topology OLSRv2 (MT-OLSRv2) [RFC7722], as an extension to OLSRv2 that can interoperate with unextended instances of OLSRv2, would not have been possible without these restrictions, which were applied to NHDP and OLSRv2 by [RFC7181].

These restrictions do not, however, mean that added information is completely ignored for purposes of the base protocol. Suppose that a faulty implementation of OLSRv2 (including NHDP) creates a HELLO message that assigns two different values of the same link metric to an address, something that is not permitted by [RFC7181]. A receiving OLSRv2-aware implementation of NHDP MUST reject such a message, even though a receiving OLSRv2-unaware implementation of NHDP will process it. This is because the OLSRv2-aware implementation has access to additional information, that the HELLO message is definitely invalid, and the message is best ignored, as it is unknown what other errors it may contain.

4.5. Information Representation

A message (excluding the Message Header) can thus be represented by two, possibly multivalued, maps:

- **Message:** `(extended type) -> (length, value)`
- **Address:** `(address, extended type) -> (length, value)`

These maps (plus a representation of the Message Header) can be the basis for a generic representation of information in a message. Such maps can be created by parsing the message, or can be constructed.
using the protocol rules for creating a message, and later converted into the octet form of the message specified in [RFC5444].

While of course any implementation of software that represents software in the above form can specify an application programming interface (API) for that software, such an interface is not proposed here. First, a full API would be programming language specific. Second, even within the above framework, there are alternative approaches to such an interface. For example, and for illustrative purposes only, for the address mapping:

- Input: address and extended type. Output: list of (length, value) pairs. Note that for most extended types it will be known in advance that this list will have length zero or one. The list of addresses that can be used as inputs with non-empty output would need to be provided as a separate output.

- Input: extended type. Output: list of (address, length, value) triples. As this list length may be significant, a possible output will be of one or two iterators that will allow iterating through that list. (One iterator that can detect the end of list, or a pair of iterators specifying a range.)

Additional differences in the interface may relate to, for example, the ordering of output lists.

4.6. TLVs

Within a message, the attributes are represented by TLVs. Particularly for Address Block TLVs, different TLVs may represent the same information. For example, using the LINK_STATUS TLV defined in [RFC6130], if some addresses have Value SYMMETRIC and some have Value HEARD, arranged in that order, then this information can be represented using two single value TLVs or one multivalue TLV. The latter can be used even if the addresses are not so ordered.

A protocol MAY use any representation of information using TLVs that convey the required information. A protocol SHOULD use an efficient representation, but this is a quality of implementation issue. A protocol MUST recognize any permitted representation of the information; even if it chooses to (for example) only use multivalue TLVs, it MUST recognize single value TLVs (and vice versa).

A protocol defining new TLVs MUST respect the naming and organizational rules in [RFC7631]. It SHOULD follow the guidance in [RFC7188], except where those requirements are ones that MUST be followed as required by this specification (or when extending [RFC6130] or [RFC7181], when these MUST also be followed).
4.7. Message Integrity

In addition to not rejecting a message due to unknown TLVs or TLV Values, a protocol MUST NOT fail to forward a message (by whatever means of message forwarding are appropriate to that protocol) due to the presence of such TLVs or TLV Values, and MUST NOT remove such TLVs or TLV Values. Such behavior would have the consequences that:

- It might disrupt the operation of an extension of which it is unaware. Note that it is the responsibility of a protocol extension to handle interoperation with unextended instances of the protocol. For example OLSRv2 [RFC7181] adds an MPR_WILLING TLV to HELLO messages (created by NHDP, [RFC6130], of which it is in part an extension) to recognize this case (and for other reasons). If an incompatible protocol extension were defined, it would be the responsibility of network management to ensure that incompatible routers were not both present in the MANET; this case is NOT RECOMMENDED.

- It would prevent the operation of end to end message authentication using [RFC7182], or any similar mechanism. The use of immutable (apart from hop count and/or hop limit) messages by a protocol is strongly RECOMMENDED for that reason.

5. Structure

The elements defined in [RFC5444] have structures that are managed by a number of flags fields:

- Packet flags field (4 bits, 2 used) that manages the contents of the Packet Header.

- Message flags field (4 bits, 4 used) that manages the contents of the Message Header.

- Address Block flags field (8 bits, 4 used) that manages the contents of an Address Block.

- TLV flags field (8 bits, 5 used) that manages the contents of a TLV.

Note that all of these flags are structural, they specify which elements are present or absent, or field lengths, or whether a field has one or multiple values in it.

In the current version of [RFC5444], indicated by version number 0 in the <version> field of the Packet Header, unused bits in these flags
fields "are RESERVED and SHOULD each be cleared ('0') on transmission and SHOULD be ignored on reception".

If a specification updating [RFC5444] introduces new flags in one of the flags fields of a packet, message or Address Block, the following rules MUST be followed:

- The version number contained in the <version> field of the Packet Header MUST NOT be 0.
- The new flag(s) MUST indicate the structure of the corresponding packet, message, Address Block or TLV, and MUST NOT be used to indicate any other semantics, such as message forwarding behavior.

An update that would be incompatible with the current specification of [RFC5444] SHOULD NOT be created unless there is a pressing reason for it that cannot be satisfied using the current specification (e.g., by use of a suitable Message TLV).

During the development of [RFC5444], and since publication thereof, some proposals have been made to use these RESERVED flags to specify behavior rather than structure, in particular message forwarding. These proposals were, after due consideration, not accepted, for a number of reasons. These reasons include that message forwarding, in particular, is protocol-specific; for example [RFC7181] forwards messages using its MPR (Multi-Point Relay) mechanism, rather than a "blind" flooding mechanism. (The later addition of a 4 bit Message Address Length field later left no unused message flags bits, but other fields still have unused bits.)

6. Message Efficiency

The ability to organize addresses into different, or the same, Address Blocks, as well as to change the order of addresses within an Address Block, and the flexibility of the TLV specification, enables avoiding unnecessary repetition of information, and consequently can generate smaller messages. No algorithms for address organization or compression or for TLV usage are given in [RFC5444], any algorithms that leave the information content unchanged MAY be used.

6.1. Address Block Compression

Addresses in an Address Block can be compressed, and SHOULD be.

Compression of addresses in an Address Block considers addresses to consist of a Head, a Mid, and a Tail, where all addresses in an Address Block have the same Head and Tail, but different Mids. An
additional compression is possible when the Tail consists of all zero-valued octets. Expected use cases are IPv4 and IPv6 addresses from within the same prefix and which therefore have a common Head, IPv4 subnets with a common zero-valued Tail, and IPv6 addresses with a common Tail representing an interface identifier as well as having a possible common Head. Note that when, for example, IPv4 addresses have a common Head, their Tail will usually be empty. For example 192.0.2.1 and 192.0.2.2 would, for greatest efficiency, have a 3 octet Head, a 1 octet Mid, and a 0 octet Tail.

Putting addresses into a message efficiently also has to include:

- The split of the addresses into Address Blocks.
- The order of the addresses within the Address Blocks.

This split and/or ordering is for efficiency only, it does not provide any information. The split of the addresses affects both the address compression and the TLV efficiency (see Section 6.2), the order of the addresses within an Address Block affects only the TLV efficiency. However using more Address Blocks than is needed can increase the message size due to the overhead of each Address Block and the following TLV Block, and/or if additional TLVs are now required.

The order of addresses can be as simple as sorting the addresses, but if many addresses have the same TLV Types attached, it might be more useful to put these addresses together, either within the same Address Block as other addresses, or in a separate Address Block. A separate address block might also improve address compression, for example if more than one address form is used (such as from independent subnets). An example of the possible use of address ordering is a HELLO message from [RFC6130] which MAY be generated with local interface addresses first and neighbor addresses later. These MAY be in separate Address Blocks.

### 6.2. TLVs

The main opportunities for efficient messages when considering TLVs are in Address Block TLVs, rather than Message TLVs.

An Address Block TLV provides attributes for one address or a contiguous (as stored in the Address Block) set of addresses (with a special case for when this is all addresses in an Address Block). When associated with more than one address, a TLV may be single value (associating the same attribute with each address) or multivalue (associating a separate attribute with each address).
The simplest to implement approach is to use multivalue TLVs that cover all affected addresses. However unless care is taken to order addresses appropriately, these affected addresses may not all be contiguous. Approaches to this are to:

- Reorder the addresses. It is, for example, possible (though not straightforward) to order all addresses in HELLO message as specified in [RFC6130] so that all TLVs used only cover contiguous addresses. This is even possible if the MPR TLV specified in OLSRv2 [RFC7181] is added; but it is not possible, in general, if the LINK_METRIC TLV is also added.

- Allow the TLV to span over addresses that do not need the corresponding attribute, using a Value that indicates no information, see Section 6.3.

- Use more than one TLV. Note that this can be efficient when the TLVs thus become single value TLVs. In a typical case where a LINK_STATUS TLV uses only the Values HEARD and SYMMETRIC, with enough addresses, sorted appropriately, two single value TLVs can be more efficient than one multivalue TLV. (When only one Value is involved, such as NHDP in a steady state with LINK_STATUS equal to SYMMETRIC in all cases, one single value TLV SHOULD always be used.)

6.3. TLV Values

If, for example, an Address Block contains five addresses, the first two and the last two requiring Values assigned using a LINK_STATUS TLV, but the third does not, then this can be indicated using two TLVs. It is however more efficient to do this with one multivalue LINK_STATUS TLV, assigning the third address the Value UNSPECIFIED. In general, use of UNSPECIFIED Values allows use of fewer TLVs and thus often an efficiency gain; however a long run of consecutive UNSPECIFIED Values (more than the overhead of a TLV) may make more TLVs more efficient.

This approach was specified in [RFC7188], and required for protocols that extend [RFC6130] and [RFC7181]. It is here RECOMMENDED that this approach is followed when defining any Address Block TLV that may be used by a protocol using [RFC5444].

It might be argued that this is not necessary in the example above, because the addresses can be reordered. However ordering addresses in such a way for all possible TLVs is not, in general, possible.

As indicated, the LINK_STATUS TLV, and some other TLVs that take single octet Values (per address), have a Value UNSPECIFIED defined,
as the Value 255, in [RFC7188]. A similar approach (and a similar Value) is RECOMMENDED in any similar cases. Some other TLVs may need a different approach. As noted in [RFC7188], but implicitly permissible before then, the LINK_METRIC TLV has two octet Values whose first four bits are flags indicating whether the metric applies in four cases; if these are all zero then the metric does not apply in this case, which is thus the equivalent of an UNSPECIFIED Value.

6.4. Automation

There is scope for creating a protocol-independent optimizer for [RFC5444] messages that performs appropriate address re-organization (ordering and Address Block separation) and TLV changes (of number, single- or multi-valuedness and use of UNSPECIFIED Values) to create more compact messages. The possible gain depends on the efficiency of the original message creation, and the specific details of the message. Note that this process cannot be TLV Type independent, for example a LINK_METRIC TLV has a more complicated Value structure than a LINK_STATUS TLV does if using UNSPECIFIED Values.

7. Security Considerations

This document does not specify a protocol, but provides rules and recommendations for how to design protocols using [RFC5444]. This document does not introduce any new security considerations; protocols designed according to these rules and recommendations are subject to the security considerations detailed in [RFC5444]. In particular the applicability of the security framework for [RFC5444] specified in [RFC7182] is unchanged.

8. IANA Considerations

This document has no actions for IANA.

9. Acknowledgments

The authors thank Cedric Adjih (INRIA) and Justin Dean (NRL) for their contributions as authors of RFC 5444.
10. References

10.1. Normative References


10.2. Informative References


[RFC7188] Dearlove, C. and T. Clausen, "Optimized Link State Routing
Protocol version 2 (OLSRv2) and MANET Neighborhood Discovery Protocol (NHDP) Extension TLVs", RFC 7183, April 2014.


Authors’ Addresses

Thomas Clausen
LIX, Ecole Polytechnique
91128 Palaiseau Cedex,
France

Phone: +33-6-6058-9349
Email: T.Clausen@computer.org
URI: http://www.thomasclausen.org

Christopher Dearlove
BAE Systems Applied Intelligence Laboratories
West Hanningfield Road
Great Baddow, Chelmsford
United Kingdom

Email: chris.dearlove@baesystems.com
URI: http://www.baesystems.com

Ulrich Herberg

Email: ulrich@herberg.name
URI: http://www.herberg.name
Henning Rogge
Fraunhofer FKIE
Fraunhofer Strasse 20
53343 Wachtberg
Germany

Email: henning.rogge@fkie.fraunhofer.de