Network Working Group
Internet-Draft
Updates: RFC5444 (if approved)
Intended status: Standards Track
Expires: November 29, 2015

T. Clausen
LIX, Ecole Polytechnique
C. Dearlove
BAE Systems AI Labs
U. Herberg
H. Rogge

May 28, 2015

Rules For Designing Protocols Using the RFC5444 Generalized Packet/Message Format
draft-clausen-manet-rfc5444-usage-00

Abstract

This document updates the generalized MANET packet/message format, specified in RFC5444, by providing prescriptive guidelines for how protocols can use that packet/message format. In particular, these mandatory guidelines prohibit a number of uses of RFC5444 that have been suggested in various proposals, and which would have lead to interoperability problems, to impediment of protocol extension development, and to inability to use generic RFC5444 parsers.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

[RFC5444] specifies a generalized packet/message format, designed for use by MANET routing protocols. [RFC5498] mandates the use of this format by protocols operating over the manet IP protocol and port numbers whose allocation it requested.

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 by the MANET working group as a common building block for use by both proactive and reactive MANET routing protocols.

1.1. History and Purpose

Since the publication of [RFC5444] in 2009, several RFCs have been published, including [RFC5497], [RFC6130], [RFC6621], [RFC6622], [RFC7181], [RFC7182], [RFC7183], and [RFC7188], which 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 (i) enable the use of an efficient and generic parser for all protocols using [RFC5444], (ii) ensure "forward compatibility" of a protocol with future extensions, and (iii) enable the creation of efficient messages.

During the same time period, other suggestions have been made to use [RFC5444] in a manner that would lead to incompatibilities with generic RFC 5444 parsers, would inhibit the development of interoperable protocol extensions, or would potentially lead to inefficiencies. 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:

o It is designed for carrying MANET routing protocol control signals.

o It defines a packet as a packet header with a set of packet TLVs, 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 set of message TLVs (Type-Length-Value structures), 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 packets, 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 packet over UDP or directly over IP, the responsibility of this multiplexing process.

o A packet is designed to as 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.

o It supports "internal extensibility" using TLVs; an extension can add information to an existing message type without that information rendering the message un-parseable 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.

o 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 a common interface Identifiers, also, 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 type 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 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, make desirable forms of generic parsers impossible, or make forms of extensions of those protocols impossible, or impedes on the ability to generate efficient messages.

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], specifically 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, make desirable forms of generic parsers impossible, or make forms of extensions of those protocols impossible, or impedes on the ability to generate efficient (small) messages. 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]. #### SEE COMMENTS IN SVN COMMIT MESSAGE AND ON LIST #### 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 considered.

TLV space, although greater than message space, SHOULD also be used

efficiently. The full type of 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 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. [RFC7181] contains a general purpose process for doing that, 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 the following section. 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., HW addresses when the address block is recording IP addresses) MAY be carried as TLV values. Note that a message contains a Message Address Length (MAL) field that can be used to allow carrying alternative message sizes, but only one length of addresses in all address blocks can be used in a single message.

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.

o The owning protocol SHOULD verify each message, it SHOULD allow any extending protocol(s) to also contribute to this.

o 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:

o Outgoing messages MAY be created by 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.

o Outgoing messages are then sent to the [RFC5444] multiplexing process. The owning protocol 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.

o The multiplexing process MAY combine messages from multiple protocols in a packet.

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 message (Message TLV) or any value of any address (Address 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].

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

A protocol MUST NOT assign any meaning to the presence, or absence, of an address, as this would prevent the addition of addresses with other meanings. 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 below). 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.

This example can be taken further. NHDP must also not reject a HELLO message because it contains an unrecognized TLV. This also applies to unrecognized TLV values, where a TLV supports only a limited set of values. For example, the blacklisting described in the previous paragraph 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.

Information may also be added to addresses recognized by the base 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 SMF [RFC6621]) must still process the HELLO message, ignoring the LINK_METRIC TLVs.

This does 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 which is not permitted by [RFC7181]. A receiving OLSRv2-aware implementation of NHDP should 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.

The restrictions on the use of address ordering and an address presence or absence in given address blocks for carrying information are for two reasons. First use of those prevents the approach to information representation described in Section 4.5. Second, it reduces the options available for message optimization described in Section 6.

4.5. Information Representation

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

- o Message: (extended type) -> (length, value)

- o 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:

o 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.

o Input: extended type. Output: list of (address, length, value) triples. As this list length may be significant, the likely 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. 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 values. Such behavior would have the consequences that:

o 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_WILLNG 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.

o 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 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:

- o Packet flags (4 bits, 2 used) that manages the contents of the packet header.

- o Message flags (4 bits, 4 used) that manages the contents of the message header.

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

- o TLV flags (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 introduces new flags in one of the flags fields of a packet, message or Address Block, the following rules MUST be followed:

- o The version number contained in the <version> field of the packet header MUST NOT be 0.

- o 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.

During the development of [RFC5444], and since publication hereof, some proposals have been made to use these RESERVED flags to specify behavior rather than structure, in particular message forwarding. These were, after due consideration, not accepted, for a number of reasons. These 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 spare flags bits at the message level for such use.

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, enables avoiding unnecessary repetition of information - and, consequently, generation of smaller messages.

6.1. Addresses

Addresses in an address block can be compressed, and SHOULD be. While no algorithm for compression is given in [RFC5444], an efficient compression algorithm given a set of addresses, is straightforward to implement.

Compression of addresses is obtained by considering 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 a possible common Head. Note that when, for example, IPv4 addresses have a common Head, their Tail will be empty. For example 10.0.0.1 and 10.0.0.2 would have a 3 octet Head, a 1 octet Mid, and a 0 octet Tail.

Separate address blocks will be compressed separately. One possible use of multiple address blocks (as well as the ability to have more than 255 addresses) is for organizing addresses that compress separately, e.g., from the same interface (same tail) or from the same prefix (common head), or both.

6.2. TLVs

The main opportunities for efficient messages when considering TLVs are 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-valued (associating the same attribute with each address) or multi-valued (associating a separate attribute with each address).

The simplest to implement approach is to use multi-valued 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:

- o 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.

- o 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.

- o Use more than one TLV. Note that this can be efficient when the TLVs thus become single-valued. In a typical case where a LINK_STATUS TLV uses only the values HEARD and SYMMETRIC, with enough addresses, sorted appropriately, two single-valued TLVs can be more efficient than one multi-valued TLV. (When only one value is involved, such as NHDP in a steady state with LINK_STATUS equal to SYMMETRIC in all cases, a single single-valued 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 a single multivalue LINK_STATUS TLV, assigning the third address the value UNSPECIFIED. This approach was specified in [RFC7188], and required for protocols that extend [RFC6130] and [RFC7181]. It is here RECOMMNDED 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) has 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 value

applies in four cases; if these are all zero then the metric value 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 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 while protocol-independent, this cannot be entirely TLV-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 guidelines 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

TBD

10. References

10.1. Normative References

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

[RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized MANET Packet/Message Format", RFC 5444,

February 2009.

10.2. Informative References

[G9903] "ITU-T G.9903: Narrow-band orthogonal frequency division multiplexing power line communication transceivers for G3-PLC networks", May 2013.

[RFC3626] Clausen, T. and P. Jacquet, "The Optimized Link State Routing Protocol", RFC 3626, October 2003.

[RFC5497] Clausen, T. and C. Dearlove, "Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs)", RFC 5497, March 2009.

[RFC5498] Chakeres, I., "IANA Allocations for Mobile Ad Hoc Network (MANET) Protocols", RFC 5498, March 2009.

[RFC6130] Clausen, T., Dean, J., and C. Dearlove, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.

[RFC6621] Macker, J., "Simplified Multicast Forwarding", RFC 6621, May 2012.

[RFC6622] Herberg, U. and T. Clausen, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", RFC 6622, May 2012.

[RFC7181] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", RFC 7181, April 2014.

[RFC7182] Herberg, U., Clausen, T., and C. Dearlove, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", RFC 7182, April 2014.

[RFC7183] Herberg, U., Dearlove, C., and T. Clausen, "Integrity Protection for the Neighborhood Discovery Protocol (NHDP) and Optimized Link State Routing Protocol Version 2 (OLSRv2)", RFC 7183, April 2014.

[RFC7188] Dearlove, C. and T. Clausen, "Optimized Link State Routing Protocol version 2 (OLSRv2) and MANET Neighborhood Discovery Protocol (NHDP) Extension TLVs", RFC 7188, 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

Phone: +44 1245 242194
Email: chris.dearlove@baesystems.com
URI: http://www.baesystems.com/

Ulrich Herberg

Email: ulrich@herberg.name
URI: http://www.herberg.name

Henning Rogge

Email: henning.rogge@fkie.fraunhofer.de