DICE Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2015

S. Kumar
Philips Research
R. Struik
Struik Security Consultancy
March 09, 2015

Transport-layer Multicast Security for Low-Power and Lossy Networks (LLNs)
draft-kumar-dice-multicast-security-00

Abstract

CoAP and 6LoWPAN are fast emerging as the de-facto protocol standards in the area of resource-constrained devices forming Low-power and Lossy Networks (LLNs). Unicast communication in such networks are secured at the transport layer using DTLS, as mandated by CoAP. For various applications, IP multicast-based group communications in such networks provide clear advantages. However, at this point, CoAP does not specify how to secure group communications in an interoperable way. This draft specifies two methods for securing CoAP-based group communication at the transport layer and targets deployment scenarios that may require group authentication, respectively source authentication. The specification leverages the fact that DTLS is already used as the mechanism of choice to secure unicast communications and allows group communications security to be implemented as an extension of DTLS record layer processing, thereby minimizing incremental implementation cost.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

There is an ever growing number of electronic devices, sensors and actuators that have become wireless and Internet connected, thus creating a trend towards the Internet-of-Things (IoT). These connected devices are equipped with communication capability based on CoAP [RFC7252] and 6LoWPAN [RFC6347] standards that enables them to interact with each other as well as with the wider Internet services. However, the devices in such wireless networks are characterized by power constraints (as these are usually battery-operated), have limited computational resources (low CPU clock, small RAM and flash storage) and often, the communication bandwidth is limited and unreliable (e.g., IEEE 802.15.4 radio). Hence, such wireless control networks are also known as Low-power and Lossy Networks (LLNs).

In addition to the usual device-to-device unicast communication that would allow devices to interact with each other, group communication is an important feature in LLNs. It is more effective in LLNs to convey messages to a group of devices without requiring the sender to perform multiple time and energy consuming unicast transmissions to reach each individual group member. For example, in a lighting control system, lighting devices are often grouped according to the layout of the building, and control commands are issued simultaneously to a group of devices. Group communication for LLNs is based on the CoAP sent over IP- multicast [RFC7390].

Currently, CoAP messages are protected using Datagram Transport Layer Security (DTLS) [RFC6347]. However, DTLS is mainly used to secure a connection between two endpoints and it cannot be used to protect multicast group communication. Group communication in LLNs is equally important and should be secured as it is also vulnerable to the usual attacks over the air (eavesdropping, tampering, message forgery, replay, etc.). Although there have been a lot of efforts in IETF to standardize mechanisms to secure multicast communication [RFC3830] [RFC4082] [RFC3740] [RFC4046] [RFC4535], they are not necessarily suitable for LLNs which have much more limited bandwidth and resources. For example, the MIKEY Architecture [RFC3830] is mainly designed to facilitate multimedia distribution, while TESLA [RFC4082] is proposed as a protocol for broadcast authentication of the source with good time synchronization. [RFC3740] and [RFC4046] provide reference architectures for multicast security. [RFC4535] describes Group Secure Association Key Management Protocol (GSAKMP), a security framework for creating and managing cryptographic groups on a network which can be reused for key management in our context with any needed adaptation for LLNs.

An existing IP multicast security protocol could be used after profiling as shown in [I-D.mglt-dice-ipsec-for-application-payload].

However since DTLS is already mandated for CoAP unicast, this would require an additional security protocol to be implemented on constrained devices. This draft describes an alternative transport layer approach by reusing already available functionalities in DTLS. This allows CoAP group communication security to be implemented with minimal additional resources as an extension to the DTLS record layer processing.

1.1. Terminology

This specification uses the following terminology:

- Group Controller: Entity responsible for creating a multicast group and establishing security associations among authorized group members. This entity is also responsible for renewing/ updating multicast group keys and related policies.

- Sender: Entity that sends data to the multicast group. In a 1-to-N multicast group, only a single sender transmits data to the group; in an M-to-N multicast group (where M and N do not necessarily have the same value), M group members are senders.

- Listener: Entity that receives multicast messages when listening to a multicast IP address.

- Security Association (SA): Set of policies and cryptographic keying material that together provide security services to network traffic matching this policy [RFC3740]. Here, a Security Association usually includes the following attributes:

  * selectors, such as source and destination transport addresses;

  * properties, such as identities;

  * cryptographic policy, such as the algorithms, modes, key lifetimes, and key lengths used for authentication or confidentiality;

  * keying material for authentication, encryption and signing.

- Group Security Association: A bundling of security associations (SAs) that together define how a group communicates securely [RFC3740].

- Keying material: Data specified as part of the SA that is necessary to establish and maintain a cryptographic security association, such as keys, key pairs, and IVs [RFC4949].

- Group authentication: Evidence that a received group message originated from some member of this group. This provides assurances that this message was not tampered with by an adversary outside this group, but does not pinpoint who precisely in the group originated this message.

- Source authentication: Evidence that a received group message originated from a specifically identified group member. This provides assurances that this message was not tampered with by any other group member or an adversary outside this group.

1.2. Outline

The remainder of this draft is organized as follows. Section 2 provides use cases for group communications with LLNs. Section 3 provides an overview of transport layer secure multicasting, while Section 4 and Section 5 provide the detailed specifications for securing multicast messaging providing for group authentication and source authentication, respectively.

2. Use Cases

This section introduces some use cases for group communications in LLNs and identifies a set of security requirements for these use cases.

2.1. Group Communication Use Cases

"Group Communication for CoAP" [RFC7390] provides the necessary background for multicast-based CoAP communication in LLNs and the interested reader is encouraged to first read this document to understand the non-security related details. This document also lists a few group communication uses cases with detailed descriptions.

a. Lighting control: Consider a building equipped with 6LoWPAN IP-connected lighting devices, switches, and 6LoWPAN border routers. The devices are organized in groups according to their physical location in the building, e.g., lighting devices and switches in a room or corridor can be configured as a single multicast group. The switches are then used to control the lighting devices in the group by sending on/off/dimming commands to all lighting devices in the group. 6LoWPAN border routers that are connected to an IPv6 network backbone (which is also multicast-enabled) are used to interconnect 6LoWPANs in the building. Consequently, this would also enable logical multicast groups to be formed even if devices in the lighting group may be physically in different subnets (e.g. on wired and wireless networks). Group

communications enables synchronous operation of a group of 6LoWPAN connected lights ensuring that the light preset (e.g. dimming level or color) of a large group of luminaires are changed at the same perceived time. This is especially useful for providing a visual synchronicity of light effects to the user.

b. Integrated Building control: enabling Building Automation and Control Systems to control multiple heating, ventilation and air-conditioning units to pre-defined presets.

c. Software and Firmware updates: software and firmware updates often comprise quite a large amount of data and can, thereby, overload an LLN that is otherwise typically used to communicate only small amounts of data infrequently. Multicasting such updated data to a larger group of devices at once, rather than sending this as unicast messages to each individual device in turn, can significantly reduce the network load and may also decrease the overall time latency for propagating this data to all devices. With updates of relatively large amounts of data, securing the individual messages is important, even if the complete update itself is secured as a whole, since checking individual received data piecemeal for tampering avoids that devices store large amounts of partially corrupted data and may detect tampering hereof only after all data has been received.

d. Parameter and Configuration update: settings of a group of similar devices are updated simultaneously and efficiently. Parameters could be, for example, network load management or network access controls

e. Commissioning of LLN systems: querying information about the devices in the local network and their capabilities, e.g., by a commissioning device.

f. Emergency broadcast: a particular emergency related information (e.g. natural disaster) is relayed to multiple devices.

2.2. Security Requirements

"Group Communications for CoAP" [RFC7390] already defines a set of security requirements for group communication in LLNs. We re-iterate and further describe those security requirements in this section with respect to the use cases:

a. Group communication topology: We consider both 1-to-N (one sender with multiple listeners) and M-to-N (multiple senders with multiple listeners) communication topologies. The 1-to-N

communication topology is the simplest group communication scenario that would serve the needs of a typical LLN. For example, in a simple lighting control use case, a switch would be the only entity that is responsible for sending control commands to a group of lighting devices. In more advanced lighting control use cases, an M-to-N communication topology would be required, for example if multiple sensors (presence or day-light) are responsible to trigger events to a group of lighting devices.

b. Group-level data confidentiality: Data confidentiality may be required if privacy sensitive data is exchanged in the group. Confidentiality is only required at the group level since any member of the group can decrypt the message.

c. Group-level authentication and integrity: In certain use cases, it is sufficient to ensure the weaker group-level authentication of group messages. Such cases include M-to-N communication topology where M senders all perform similar roles and where M is of approximately the same size as group size N. Group-level authentication is achieved by a single group key that is known to all group members and used to provide authenticity to the group messages (e.g., using a Message Authentication Code, MAC).

d. Source-level authentication: Certain use-cases require a stronger source-level authentication of group messages. This is especially needed in M-to-N communication topology, where M is closer to 1. Source authentication can be provided using public-key cryptography in which every group message is signed by its sender.

3. Overview of Transport level Secure Multicast

The IETF CoRE WG has selected DTLS [RFC6347] as the default must-implement security protocol for securing CoAP unicast traffic. The goal of this draft is to secure CoAP Group communication with minimal additional overhead on resource-constrained embedded devices. We propose to reuse some of the functionalities of DTLS such that a transport-layer multicast security solution can be implemented by extending the DTLS implementation (that already exists on these devices) with minimal adaptation.

We first give a general overview of how the transport-layer multicast security can be implemented and then provide two variations: one for group authentication and the other for source authentication.

3.1. Setting up Group Security Association

A group controller in an LLN creates a multicast group. The group controller may be hosted by a remote server, or a border router that creates a new group over the network. In some cases, devices may be configured using a commissioning tool that mediates the communication between the devices and the group controller. The controller in the network can be discovered by the devices using various methods defined in [I-D.vanderstok-core-dna] such as DNS-SD [RFC6763] and Resource Directory [I-D.ietf-core-resource-directory]. The group controller communicates with individual device in unicast to add them to the new group. This configuration can be done as unicast CoAP based messages sent over a DTLS secured link. The CoAP resource on the devices that is used to configure the group [RFC7390]can also be used to configure the devices with the Group Security Association (GSA) consisting of keying material, security policies security parameters and ciphersuites. [Note: The details of the CoAP based configuration needs further elaboration in a new revision.]

3.2. Group Communication packets

Senders in the group can authenticate and/or encrypt the CoAP group messages using the keying material in the GSA. The authenticated and/or encrypted message contains a security header which includes a replay counter. This is passed down to the lower layer of the IPv6 protocol stack for transmission to the multicast address as depicted in Figure 1.

```
       +--------+-+---------------------------------------------+-+
       |        | +---------------------------------------------+ |
       |        | |        | +--------------------------------+ | |
       |        | |        | |             | +--------------+ | | |
       |   IP   | |   UDP  | |  Security   | |   multicast  | | | |
       | header | | header | |   Header    | |    message   | | | |
       |        | |        | |             | +--------------+ | | |
       |        | |        | +--------------------------------+ | |
       |        | +---------------------------------------------+ |
       +--------+-+---------------------------------------------+-+
```

Figure 1: Sending a multicast message protected using DTLS Record Layer

The listeners when receiving the message, use the multicast IPv6 destination address and port (i.e., multicast group identifier) to derive the corresponding GSA needed for that group connection. The

received message's authenticity is verified and/or decrypted using the keying material for that connection and the security header.

4. Group-level Data Authenticity

This section describes in detail the group level authenticity mechanism for transport level secure group messages. We assume that group membership has been configured by the group controller as described in Section 3.1 , and all member devices in the group have the GSA. The group-level authentication GSA contains the following elements:

```
        CipherSuite
        server write MAC key
        server write encryption key
        server write IV
        SenderID
```

The group controller chooses a CipherSuite for the GSA based on knowledge that all devices in the group support the specific CipherSuite. Based on the CipherSuite selected, one or more of the other elements may be empty. For example, if only using authenticity only ciphersuite, the encryption key and IV is not sent, similarly if an AEAD ciphersuite is used then MAC key is not sent. The SenderID is only sent to devices which are senders in the group

The GSA is used to instantiate the needed elements of the "SecurityParameters" structure (shown below) as defined in [RFC5246] for all devices.

```
struct {
        ConnectionEnd          entity;
        PRFAlgorithm           prf_algorithm;
        BulkCipherAlgorithm    bulk_cipher_algorithm;
        CipherType             cipher_type;
        uint8                  enc_key_length;
        uint8                  block_length;
        uint8                  fixed_iv_length;
        uint8                  record_iv_length;
        MACAlgorithm           mac_algorithm;
        uint8                  mac_length;
        uint8                  mac_key_length;
        CompressionMethod      compression_algorithm;
        opaque                 master_secret[48];
        opaque                 client_random[32];
        opaque                 server_random[32];
} SecurityParameters;
```

a. SecurityParameters.entity is set to ConnectionEnd.server for senders and ConnectionEnd.client for listeners.

b. bulk_cipher_algorithm, cipher_type, enc_key_length, block_length, fixed_iv_length, record_iv_length, mac_algorithm, mac_length, and mac_key_length is set based on the ciphersuite present in the GSA.

c. cipher_type is CipherType.aead (if confidentiality is needed)

d. enc_key_length, block_length are sent in the GSA.

e. prf_algorithm, compression_algorithm, master_secret[48], client_random[32], and server_random[32] are not used and therefore not set.

The current read and write states are then instantiated for all group members based on the keying material in the GSA and according to their roles:

a. Listeners (ConnectionEnd.client) directly use "server write" parameters in the GSA for instantiating the current read state. The write state is not instantiated and not used.

b. Senders (ConnectionEnd.server) first pass each of the "server write" parameters in the GSA through a function output=F(SenderID, input) [Note: F needs to be defined later]. The output is then used for instantiating the current write state.

If a device is both a sender and listener, then the read and write states are both instantiated as above. Additionally each connection state contains the epoch (set to zero) and sequence number which is incremented for each record sent; the first record sent has the sequence number 0.

4.1. Group message format

To reuse the DTLS implementation for group message processing, the DTLS Records are used to send messages in the group.

The following Figure 2 illustrates the structure of the DTLS record layer header, the epoch and seq_number are used to ensure message freshness and to detect message replays.

```
+---------+---------+--------+--------+--------+------------+-------+
| 1 Byte  | 2 Byte  | 2 Byte | 6 Byte | 2 Byte |            |       |
+---------+---------+--------+--------+--------+------------+-------+
| Content | Version | epoch  |  seq_  | Length | Ciphertext |  MAC  |
|   Type  | Ma | Mi |        | number |        |   (Enc)    | (Enc) |
+---------+---------+--------+--------+--------+------------+-------+
```

Figure 2: The DTLS record layer header to transport group-level authenticated messages

The same packet structure is used to transport group messages with the Content Type set to ContentType.application_data, the version set to DTLS version 1.2 (mandated by CoAP) which uses the version {254, 253}, epoch is fixed for all devices by the controller and the seq_number based on current connection state. The seq_number is increased by one whenever the sender sends a new group message in the record. Finally, the message is protected (encrypted and authenticated with a MAC) using the session keys in the "server write" parameters.

4.2. Group message processing

4.2.1. Sending Secure Multicast Messages

Senders in the multicast group when sending a CoAP group message from the application, create the DTLS record payload based on the "server write" parameters. It also manages its own epoch and seq_number in the "server write" connection state; the first record sent has the seq_number 0. After creating the DTLS record, the seq_number is incremented in the "server write" connection state. The DTLS record

is then passed down to UDP and IPv6 layer for transmission on the multicast IPv6 destination address and port.

4.2.2. Receiving Secure Multicast Messages

When a listeners receives a protected multicast message from the sender, it looks up the corresponding "client read" connection state based on the multicast IP destination and port of the connection. This is different from standard DTLS logic in that the current "client read" connection state is bound to the source IP address and port.

Listener devices in a multiple senders multicast group, need to store multiple "client read" connection states for the different senders linked to the Source IP addresses. The keying material is same for all senders however tracking the epoch and the seq_number of the last received packets needs to be kept different for different senders.

The listeners first perform a "server write" keys lookup by using the multicast IPv6 destination address and port of the packet. Then the key is passed through the same F function described above to get specific keys for the particular sender. The listeners decrypt and check the MAC of the message using this key. This guarantees that no one without access to the GSA has spoofed the message. Subsequently, the listeners retrieve the "client read" connection state which contains the last stored epoch and seq_number of the sender, which is used to check the freshness of the message received. The listeners must ensure that the epoch is the same and seq_number in the message received is at least equal to the stored value, otherwise the message is discarded. Alternatively a windowing mechanism can be used to accept genuine out-of-order packets. Once the authenticity and freshness of the message have been checked, the listeners can pass the message to the higher layer protocols. The seq_number in the corresponding "client read" connection state are updated as well.

5. Source-level Data Authenticity

This section describes in detail the source level authenticity mechanism based on public-key cryptography for transport level secure group messages. We assume that group membership has been configured by the group controller as described in Section 3.1 , and all member devices in the group have the GSA.

The source-level authentication GSA contains the following elements:

a. For Senders:

```
        CipherSuite
        {SenderID, SenderID Private Key}
        server write encryption key
        server write IV
```

b. For Listeners:

```
        CipherSuite
        {SenderID1, SenderID1 Public Key}
        {SenderID2, SenderID2 Public Key}
        {SenderID3, SenderID3 Public Key}
        ...
        server write encryption key
        server write IV
```

The group controller chooses a CipherSuite for the GSA for all devices based on knowledge that all devices in the group support the specific CipherSuite. Based on the CipherSuite selected, one or more of the other elements may be empty. For example, if only authenticity is needed then the encryption key and IV is not sent.

The GSA is used to instantiate the needed elements of the "SecurityParameters" structure (shown below) as defined in [RFC5246] for all devices.

```
        struct {
                  ConnectionEnd          entity;
                  PRFAlgorithm           prf_algorithm;
                  BulkCipherAlgorithm    bulk_cipher_algorithm;
                  CipherType             cipher_type;
                  uint8                  enc_key_length;
                  uint8                  block_length;
                  uint8                  fixed_iv_length;
                  uint8                  record_iv_length;
                  MACAlgorithm           mac_algorithm;
                  uint8                  mac_length;
                  uint8                  mac_key_length;
                  CompressionMethod      compression_algorithm;
                  opaque                 master_secret[48];
                  opaque                 client_random[32];
                  opaque                 server_random[32];
        } SecurityParameters;
```

a. SecurityParameters.entity is set to ConnectionEnd.server for senders and ConnectionEnd.client for listeners.

b. bulk_cipher_algorithm, cipher_type, enc_key_length, block_length, fixed_iv_length, and record_iv_length is set based on the ciphersuite present in the GSA.

c. mac_algorithm, mac_length, and mac_key_length are set different to DTLS since we use public key algorithms. The mac here therefore is a public-key based signature.

d. enc_key_length, block_length are sent in the GSA if encryption is needed.

e. prf_algorithm, compression_algorithm, master_secret[48], client_random[32], and server_random[32] are not used and therefore not set.

The current read and write states are then instantiated for all group members based on the keying material in the GSA and according to their roles:

a. Listeners (ConnectionEnd.client) use the SenderID's public keys for instantiating the current read state for different senders. If confidentiality is needed, the "server write" parameters in the GSA is also added to the current read state. The write state is not instantiated and not used.

b. Senders (ConnectionEnd.server) use the SenderID private key to instantiate the current write state. If confidentiality is used, first pass each of the "server write" parameters in the GSA through a function output=F(SenderID, input) [Note: F needs to be defined later.].

If a device is both a sender and listener, then the read and write states are both instantiated as above. Additionally each connection state contains the epoch (set to zero) and sequence number which is incremented for each record sent; the first record sent has the sequence number 0.

5.1. Group message format

To reuse the DTLS implementation for group message processing, the DTLS Records are used to send messages in the group with minor changes.

The following Figure 2 illustrates the structure of the DTLS record layer header, the epoch and seq_number are used to ensure message freshness and to detect message replays.

```
+---------+---------+--------+--------+--------+------------+-------+
| 1 Byte  | 2 Byte  | 2 Byte | 6 Byte | 2 Byte |            |       |
+----------------------------------------------------------------+--+
| Content | Version | epoch  |  seq_  | Length | Ciphertext | Sig-  |
|   Type  | Ma + Mi |        | number |        |  (if enc)  | nature|
+---------+----+----+--------+--------+--------+------------+-------+
```

Figure 3: The DTLS record layer header to transport source-level authenticity messages

The same packet structure is used to transport group messages with the Content Type set to ContentType.application_data, the version set to DTLS version 1.2 (mandated by CoAP) which uses the version {254, 253}, epoch is fixed for all devices by the controller and the seq_number based on current connection state. The seq_number is increased by one whenever the sender sends a new group message in the record. Finally, the message is signed using the SenderID private key and added to the signature field. If confidentiality is needed then data is encrypted using the session keys in the "server write" parameters.

5.2. Group message processing

5.2.1. Sending Secure Multicast Messages

Senders in the multicast group when sending a CoAP group message from the application, create the DTLS record payload based on the current write connection state, i.e. the data (along with the headers) is signed using the private key and if confidentiality is needed, the server write parameters are used to encrypt the data. It also manages its own epoch and seq_number in the current write connection state; the first record sent has the seq_number 0. After creating the DTLS record, the seq_number is incremented in the "server write" connection state. The DTLS record is then passed down to UDP and IPv6 layer for transmission on the multicast IPv6 destination address and port.

5.2.2. Receiving Secure Multicast Messages

When a listeners receives a protected multicast message from the sender, it looks up the corresponding "client read" connection state based on the source address, source port, multicast IP destination and destination port of the connection. This is different from standard DTLS logic in that the current "client read" connection state is bound only to the source IP address and source port.

Listener devices in a multiple senders multicast group, need to store multiple "client read" connection states for the different senders linked to the source IP addresses. Each of these connection states contain the public key of the corresponding sender. Further tracking the epoch and the seq_number of the last received packets needs to be kept different for different senders.

The listeners first perform a public key lookup by retrieving the "client read" connection state using the source address, source port, multicast IPv6 destination address and port of the packet. The listeners verifies the signature of the message using this public key. This guarantees that no one without access to the GSA for the specific device has spoofed the message. Subsequently, the listeners retrieve from the "client read" connection state the last stored epoch and seq_number of the sender, which is used to check the freshness of the message received. The listeners must ensure that the epoch is the same and seq_number in the message received is higher than the stored value, otherwise the message is discarded. Alternatively a windowing mechanism can be used to accept genuine out-of-order packets. Once the authenticity and freshness of the message have been checked, the listeners can pass the message to the higher layer protocols. The the seq_number in the corresponding "client read" connection state are updated as well.

6. Security Considerations

Some of the security issues that should be taken into consideration are discussed below.

6.1. Late joiners

Listeners who are late joiners to a multicast group, do not know the current epoch and seq_number being used by different senders. When they receive a packet from a sender with a random seq_number in it, it is impossible for the listener to verify if the packet is fresh and has not been replayed by an attacker. To overcome this late joiner security issue, the group controller which can act as a listener in the multicast group can maintain the epoch and seq_number of each sender. When late joiners send a request to the group

controller to join the multicast group, the group controller can send the list of epoch and seq_numbers to the late joiner.

7. Acknowledgements

8. References

8.1. Normative References

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", RFC 5116, January 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5288] Salowey, J., Choudhury, A., and D. McGrew, "AES Galois Counter Mode (GCM) Cipher Suites for TLS", RFC 5288, August 2008.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

[RFC6655] McGrew, D. and D. Bailey, "AES-CCM Cipher Suites for Transport Layer Security (TLS)", RFC 6655, July 2012.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.

[RFC7390] Rahman, A. and E. Dijk, "Group Communication for the Constrained Application Protocol (CoAP)", RFC 7390, October 2014.

8.2. Informative References

[FIPS.180-2.2002] National Institute of Standards and Technology, "Secure Hash Standard", FIPS PUB 180-2, August 2002, <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>.

[FIPS.197.2001] National Institute of Standards and Technology, "Advanced Encryption Standard (AES)", FIPS PUB 197, November 2001, <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.

[I-D.dijk-core-groupcomm-misc]
Dijk, E. and A. Rahman, "Miscellaneous CoAP Group Communication Topics", draft-dijk-core-groupcomm-misc-06 (work in progress), June 2014.

[I-D.ietf-core-resource-directory]
Shelby, Z. and C. Bormann, "CoRE Resource Directory", draft-ietf-core-resource-directory-02 (work in progress), November 2014.

[I-D.mcgrew-aero]
McGrew, D. and J. Foley, "Authenticated Encryption with Replay prOtection (AERO)", draft-mcgrew-aero-01 (work in progress), February 2014.

[I-D.mglt-dice-ipsec-for-application-payload]
Migault, D. and C. Bormann, "IPsec/ESP for Application Payload", draft-mglt-dice-ipsec-for-application-payload-00 (work in progress), July 2014.

[I-D.vanderstok-core-dna]
Stok, P., Lynn, K., and A. Brandt, "CoRE Discovery, Naming, and Addressing", draft-vanderstok-core-dna-02 (work in progress), July 2012.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC3740] Hardjono, T. and B. Weis, "The Multicast Group Security Architecture", RFC 3740, March 2004.

[RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, August 2004.

[RFC4046] Baugher, M., Canetti, R., Dondeti, L., and F. Lindholm, "Multicast Security (MSEC) Group Key Management Architecture", RFC 4046, April 2005.

[RFC4082] Perrig, A., Song, D., Canetti, R., Tygar, J., and B. Briscoe, "Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction", RFC 4082, June 2005.

[RFC4535] Harney, H., Meth, U., Colegrove, A., and G. Gross, "GSAKMP: Group Secure Association Key Management Protocol", RFC 4535, June 2006.

[RFC4785] Blumenthal, U. and P. Goel, "Pre-Shared Key (PSK) Ciphersuites with NULL Encryption for Transport Layer Security (TLS)", RFC 4785, January 2007.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", RFC 4949, August 2007.

[RFC5374] Weis, B., Gross, G., and D. Ignjatic, "Multicast Extensions to the Security Architecture for the Internet Protocol", RFC 5374, November 2008.

[RFC6282] Hui, J. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, September 2011.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, February 2013.

Authors' Addresses

Sandeep S. Kumar
Philips Research
High Tech Campus 34
Eindhoven 5656 AE
The Netherlands

Email: ietf.author@sandeep-kumar.org

Rene Struik
Struik Security Consultancy

Email: rstruik.ext@gmail.com