

dice  
Internet-Draft  
Intended status: Informational  
Expires: August 18, 2014

K. Hartke  
Universitaet Bremen TZI  
H. Tschofenig  
ARM Ltd.  
February 14, 2014

A DTLS 1.2 Profile for the Internet of Things  
draft-hartke-dice-profile-03

Abstract

This document defines a DTLS profile that is suitable for Internet of Things applications and is reasonably implementable on many constrained devices.

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 August 18, 2014.

Copyright Notice

Copyright (c) 2014 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 Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. The Communication Model . . . . .	4
3. The Ciphersuite Concept . . . . .	5
4. Pre-Shared Secret Authentication with DTLS . . . . .	6
5. Raw Public Key Use with DTLS . . . . .	8
6. Certificate Use with DTLS . . . . .	10
7. Error Handling . . . . .	11
8. Session Resumption . . . . .	12
9. TLS Compression . . . . .	13
10. Perfect Forward Secrecy . . . . .	13
11. Keep-Alive . . . . .	14
12. Negotiation and Downgrading Attacks . . . . .	14
13. Privacy Considerations . . . . .	14
14. Security Considerations . . . . .	15
15. IANA Considerations . . . . .	15
16. Acknowledgements . . . . .	15
17. References . . . . .	16
17.1. Normative References . . . . .	16
17.2. Informative References . . . . .	17
Authors' Addresses . . . . .	19

## 1. Introduction

This document defines a DTLS 1.2 [RFC6347] profile that offers communication security for Internet of Things (IoT) applications and is reasonably implementable on many constrained devices. It aims to meet the following goals:

- o One-stop shop for implementers through the specification jungle.
- o This document does not alter the DTLS 1.2 specification.
- o This document does not introduce new extensions.
- o This profile aligns with the DTLS security modes of the Constrained Application Protocol (CoAP) [I-D.ietf-core-coap].

DTLS is used to secure a number of applications run over an unreliable datagram transport. CoAP [I-D.ietf-core-coap] is one such protocol and has been designed specifically for use in IoT environments. CoAP can be secured using a number of different ways, also called security modes. These security modes are:

No Security Protection at the Transport Layer: No DTLS is used but instead application layer security functionality is assumed.

Shared Secret-based DTLS Authentication: DTLS supports the use of shared secrets [RFC4279]. This credential is useful if the number of communication relationships between the IoT device and servers is small and for very constrained devices. Shared secret-based authentication mechanisms offer good performance and require a minimum of data to be exchanged.

DTLS Authentication using Asymmetric Credentials: TLS supports client and server authentication using asymmetric credentials. Two approaches for validating these public key are available. First, [I-D.ietf-tls-oob-pubkey] allows raw public keys to be used in TLS without the overhead of certificates. This approach requires out-of-band validation of the public key. Second, the use of X.509 certificates [RFC5280] with TLS is common on the Web today (at least for server-side authentication) and certain IoT environments may also re-use those capabilities. Certificates bind an identifier to the public key signed by a certification authority (CA). A trust anchor store has to be provisioned on the device to indicate what CAs are trusted. Furthermore, the certificate may contain a wealth of other information used to make authorization decisions.

As described in [I-D.ietf-lwig-tls-minimal] an application designer developing an IoT device needs to think about the security threats that need to be mitigated. For many Internet connected devices it is, however, likely that authentication of the device and the server infrastructure will be required. Along with the ability to upload sensor data and to retrieve configuration information the need for integrity and confidentiality protection will arise. While these security services can be provided at different layers in the protocol stack the use of channel security, as offered by DTLS, has been very popular on the Internet and it is likely to be useful for IoT scenarios as well. In case the channel security features offered by DTLS meet the security requirements of your application the remainder of the document might offer useful guidance.

Not every IoT deployment will use CoAP but the discussion regarding choice of credentials and cryptographic algorithms will be very similar. As such, the discussions in this document are applicable beyond the use of the CoAP protocol.

The design of DTLS is intentionally very similar to TLS. Since DTLS operates on top of an unreliable datagram transport a few enhancements to the TLS structure are, however necessary. RFC 6347 explains these differences in great detail. As a short summary, for those familiar with TLS the differences are:

- o An explicit sequence number and an epoch field is included in the TLS Record Layer. Section 4.1 of RFC 6347 explains the processing rules for these two new fields. The value used to compute the MAC is the 64-bit value formed by concatenating the epoch and the sequence number.
- o Stream ciphers must not be used with DTLS. The only stream cipher defined for TLS 1.2 is RC4.
- o The TLS Handshake Protocol has been enhanced to include a stateless cookie exchange for Denial of Service (DoS) resistance. Furthermore, the header has been extended to deal with message loss, reordering, and fragmentation. Retransmission timers have been included to deal with message loss. For DoS protection a new handshake message, the HelloVerifyRequest, was added to DTLS. This handshake message is sent by the server and includes a stateless cookie, which is returned in a ClientHello message back to the server. This type of DoS protection mechanism has also been incorporated into the design of IKEv2. Although the exchange is optional for the server to execute, a client implementation has to be prepared to respond to it.

## 2. The Communication Model

This document describes a profile of DTLS 1.2 and to be useful it has to make assumptions about the envisioned communication architecture. The architecture shown in Figure 1 assumes a uni-cast communication interaction with an IoT device acting as a client and the client interacts with one or multiple servers. Which server to contact is based on pre-configuration onto the client (e.g., as part of the firmware). This configuration information also includes information about the PSK identity and the corresponding secret to be used with that specific server (in case of symmetric credentials). For asymmetric cryptography mutual authentication is assumed in this profile. For raw public keys the public key or the hash of the public key is assumed to be available to both parties. For certificate-based authentication the client may have a trust anchor store pre-populated, which allows the client to perform path validation for the certificate obtained during the handshake with the server. The client also needs to know which certificate or raw public key it has to use with a specific server.

This document only focuses on the description of the DTLS client-side functionality.

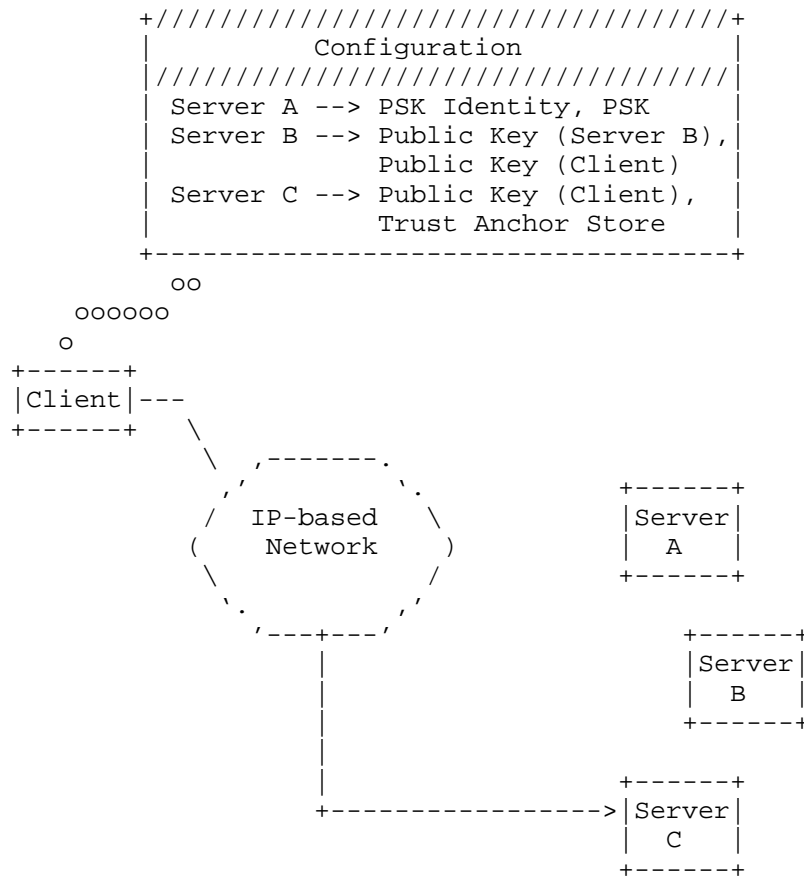


Figure 1: DTLS Profile: Assumed Communication Model.

A future version of this document may provide profiles for other communication architectures.

### 3. The Ciphersuite Concept

TLS (and consequently DTLS) introduced the concept of ciphersuites and an IANA registry [IANA-TLS] was created to keep track of the specified suites. A ciphersuite (and the specification that defines it) contains the following information:

- o Authentication and Key Exchange Algorithm (e.g., PSK)
- o Cipher and Key Length (e.g., AES with 128 bit keys)

- o Mode of operation (e.g., CBC)
- o Hash Algorithm for Integrity Protection (e.g., SHA in combination with HMAC)
- o Hash Algorithm for use with the Pseudorandom Function (e.g. HMAC with the SHA-256)
- o Misc information (e.g., length of authentication tags)

The TLS ciphersuite TLS\_PSK\_WITH\_AES\_256\_CBC\_SHA, for example, uses a pre-shared authentication and key exchange algorithm. RFC 4279, which defined this ciphersuite predates publication of TLS 1.2. It uses the Advanced Encryption Standard (AES) encryption algorithm, which is a block cipher. Since the AES algorithm supports different key lengths (such as 128, 192 and 256 bits) this information has to be specified as well and the selected ciphersuite supports 256 bit keys. A block cipher encrypts plaintext in fixed-size blocks and AES operates on fixed block size of 128 bits. For messages exceeding 128 bits, the message is partitioned into 128-bit blocks and the AES cipher is applied to these input blocks with appropriate chaining, which is called mode of operation. In our example, the mode of operation is cipher block chaining (CBC). Since encryption itself does not provide integrity protection a hash function is specified as well, which will be used in concert with the HMAC function. In this case, the Secure Hash Algorithm (SHA).

TLS 1.2 introduced Authenticated Encryption with Associated Data (AEAD) ciphersuites. AEAD is a class of block cipher modes which encrypt (parts of) the message and authenticate the message simultaneously. Examples of such modes include the Counter with CBC-MAC (CCM) mode, and the Galois/Counter Mode (GCM).

TLS 1.2 also replaced the combination of MD5/SHA-1 hash functions in the TLS pseudo random function (PRF) with cipher-suite-specified PRFs. For this reason authors of more recent TLS 1.2 ciphersuite specifications explicitly indicate the MAC algorithm and the hash functions used with the TLS PRF.

#### 4. Pre-Shared Secret Authentication with DTLS

The use of pre-shared secret credentials is one of the most basic techniques for DTLS since it is both computational efficient and bandwidth conserving. Pre-shared secret based authentication was introduced to TLS with RFC 4279 [RFC4279]. The exchange shown in Figure 2 illustrates the DTLS exchange including the cookie exchange. While the server is not required to initiate a cookie exchange with

every handshake, the client is required to implement and to react on it when challenged.

Client		Server
-----		-----
ClientHello	----->	
	<-----	HelloVerifyRequest (contains cookie)
ClientHello (with cookie)	----->	
	<-----	ServerHello *ServerKeyExchange ServerHelloDone
ClientKeyExchange ChangeCipherSpec Finished	----->	
	<-----	ChangeCipherSpec Finished
Application Data	<----->	Application Data

Legend:

\* indicates an optional message payload

Figure 2: DTLS PSK Authentication including the Cookie Exchange.

[RFC4279] does not mandate the use of any particular type of identity. Hence, the TLS client and server clearly have to agree on the identities and keys to be used. The mandated encoding of identities in Section 5.1 of RFC 4279 aims to improve interoperability for those cases where the identity is configured by a person using some management interface. Many IoT devices do, however, not have a user interface and most of their credentials are bound to the device rather than the user. Furthermore, credentials are provisioned into trusted hardware modules or in the firmware by the developers. As such, the encoding considerations are not applicable to this usage environment. For use with this profile the PSK identities MUST NOT assume a structured format (as domain names, Distinguished Names, or IP addresses have) and a bit-by-bit comparison operation can then be used by the server-side infrastructure.

As described in Section 2 clients may have pre-shared keys with several different servers. The client indicates which key it uses by including a "PSK identity" in the ClientKeyExchange message. To help the client in selecting which PSK identity / PSK pair to use, the server can provide a "PSK identity hint" in the ServerKeyExchange message. For IoT environments a simplifying assumption is made that the hint for PSK key selection is based on the domain name of the server. Hence, servers SHOULD NOT send the "PSK identity hint" in the ServerKeyExchange message and client MUST ignore the message.

RFC 4279 requires TLS implementations supporting PSK ciphersuites to support arbitrary PSK identities up to 128 octets in length, and arbitrary PSKs up to 64 octets in length. This is a useful assumption for TLS stacks used in the desktop and mobile environment where management interfaces are used to provision identities and keys. For the IoT environment, however, many devices are not equipped with displays and input devices (e.g., keyboards). Hence, keys are distributed as part of hardware modules or are embedded into the firmware. As such, these restrictions are not applicable to this profile.

Constrained Application Protocol (CoAP) [I-D.ietf-core-coap] currently specifies TLS\_PSK\_WITH\_AES\_128\_CCM\_8 as the mandatory to implement ciphersuite for use with shared secrets. This ciphersuite uses the AES algorithm with 128 bit keys and CCM as the mode of operation. The label "\_8" indicates that an 8-octet authentication tag is used. This ciphersuite makes use of the default TLS 1.2 Pseudorandom Function (PRF), which uses HMAC with the SHA-256 hash function.

## 5. Raw Public Key Use with DTLS

The use of raw public keys with DTLS, as defined in [I-D.ietf-tls-oob-pubkey], is the first entry point into public key cryptography without having to pay the price of certificates and a PKI. The specification re-uses the existing Certificate message to convey the raw public key encoded in the SubjectPublicKeyInfo structure. To indicate support two new TLS extensions had been defined as shown in Figure 3, namely the server\_certificate\_type and the client\_certificate\_type. To operate this mechanism securely it is necessary to authenticate and authorize the public keys out-of-band. This document therefore assumes that a client implementation comes with one or multiple raw public keys of servers, it has to communicate with, pre-provisioned. Additionally, a device will have its own raw public key. To replace, delete, or add raw public key to this list requires a software update, for example using a firmware update.



```

Client                                     Server
-----                                     -----

ClientHello                               ----->
client_certificate_type
server_certificate_type

                                     <----- HelloVerifyRequest

ClientHello                               ----->
client_certificate_type
server_certificate_type

                                     ServerHello
                                     client_certificate_type
                                     server_certificate_type
                                     Certificate
                                     ServerKeyExchange
                                     CertificateRequest
                                     <----- ServerHelloDone

Certificate
ClientKeyExchange
CertificateVerify
[ChangeCipherSpec]
Finished                               ----->

                                     [ChangeCipherSpec]
                                     <----- Finished

```

Figure 3: DTLS Raw Public Key Exchange including the Cookie Exchange.

The ciphersuite for use with this credential type is TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM\_8 [I-D.mcgreg-tls-aes-ccm-ecc]. This elliptic curve cryptography (ECC) based AES-CCM TLS ciphersuite uses the Elliptic Curve Diffie Hellman (ECDHE) as the key establishment mechanism and an Elliptic Curve Digital Signature Algorithm (ECDSA) for authentication. This ciphersuite make use of the AEAD capability in DTLS 1.2 and utilizes an eight-octet authentication tag. Based on the Diffie-Hellman it provides perfect forward secrecy (PFS). More details about the PFS can be found in Section 10.

RFC 6090 [RFC6090] provides valuable information for implementing Elliptic Curve Cryptography algorithms.

Since many IoT devices will either have limited ways to log error or no ability at all, any error will lead to implementations attempting to re-try the exchange.

QUESTION: [I-D.sheffer-tls-bcp] recommends a different ciphersuite, namely TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 [RFC5289] or alternatively TLS\_DHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (with a 2048-bit or 1024 DH parameters as second and third priority, respectively). Is TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM\_8 a good choice?

## 6. Certificate Use with DTLS

The use of mutual certificate-based authentication is shown in Figure 4. Note that the figure also makes use of the cached info extension, which is indicated by the TLS extension (cached\_information) and the changed content in the exchanged certificates. Caching certificate chains allows the client to reduce the communication overhead significantly since otherwise the server would provide the end entity certificate, and the certificate chain. Because certificate validation requires that root keys be distributed independently, the self-signed certificate that specifies the root certificate authority is omitted from the chain. Client implementations MUST be provisioned with a trust anchor store that contains the root certificates. The use of the Trust Anchor Management Protocol (TAMP) [RFC5934] is, however, not envisioned. Instead IoT devices using this profile MUST rely a software update mechanism to provision these trust anchors.

When DTLS is used to secure CoAP messages then the server provided certificates MUST contain the fully qualified DNS domain name or "FQDN". The coaps URI scheme is described in Section 6.2 of [I-D.ietf-core-coap]. This FQDN is stored in the SubjectAltName or in the CN, as explained in Section 9.1.3.3 of [I-D.ietf-core-coap], and used by the client to match it against the FQDN used during the look-up process, as described in RFC 6125 [RFC6125]. For the profile in this specification does not assume dynamic discovery of local servers.

For client certificates the identifier used in the SubjectAltName or in the CN MUST be an EUI-64 [EUI64], as mandated in Section 9.1.3.3 of [I-D.ietf-core-coap].

For certificate revocation neither the Online Certificate Status Protocol (OCSP) nor Certificate Revocation Lists (CRLs) are used. Instead, this profile relies on a software update mechanism. While multiple OCSP stapling [RFC6961] has recently been introduced as a mechanism to piggyback OCSP request/responses inside the DTLS/TLS handshake to avoid the cost of a separate protocol handshake further

investigations are needed to determine its suitability for the IoT environment.

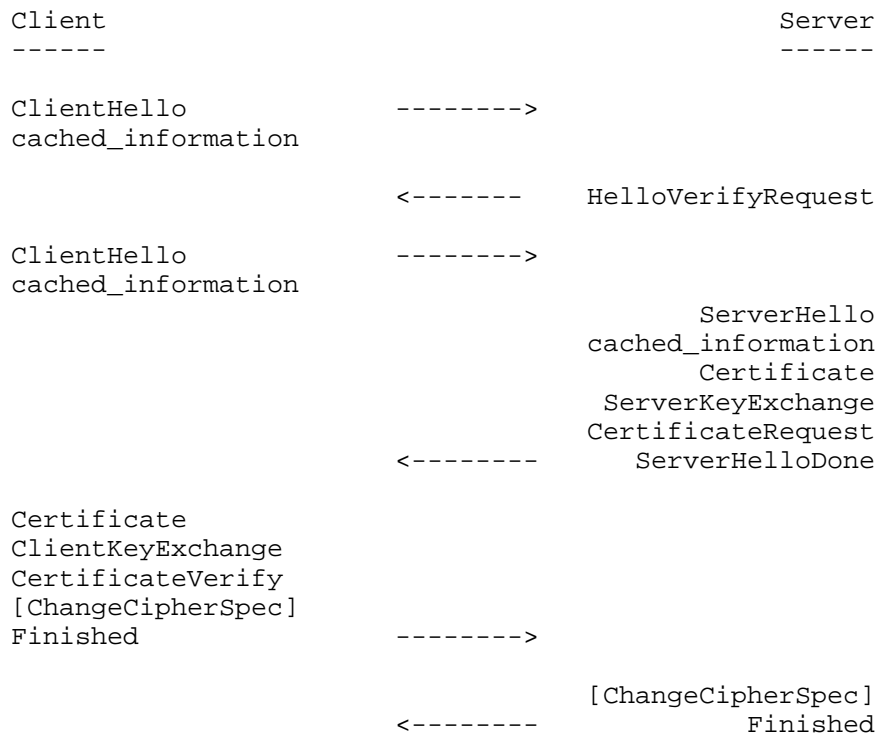


Figure 4: DTLS Mutual Certificate-based Authentication.

Regarding the ciphersuite choice the discussion in Section 5 applies. Further details about X.509 certificates can be found in Section 9.1.3.3 of [I-D.ietf-core-coap].

QUESTION: What restrictions regarding the depth of the certificate chain should be made? Is one level enough?

## 7. Error Handling

DTLS uses the Alert protocol to convey error messages and specifies a longer list of errors. However, not all error messages defined in the TLS specification are applicable to this profile. All error messages marked as RESERVED are only supported for backwards compatibility with SSL and are therefore not applicable to this profile. Those include `decryption_failed_RESERVED`,

no\_certificate\_RESERVE, and export\_restriction\_RESERVED. A number of the error messages are applicable only for certificate-based authentication ciphersuites. Hence, for PSK and raw public key use the following error messages are not applicable: bad\_certificate, unsupported\_certificate, certificate\_revoked, certificate\_expired, certificate\_unknown, unknown\_ca, and access\_denied.

Since this profile does not make use of compression at the TLS layer the decompression\_failure error message is not applicable either.

RFC 4279 introduced a new alert message unknown\_psk\_identity for PSK ciphersuites. As stated in Section 2 of RFC 4279 the decryption\_error error message may also be used instead. For this profile the TLS server MUST return the decryption\_error error message instead of the unknown\_psk\_identity.

Furthermore, the following errors should not occur based on the description in this specification:

protocol\_version: This document only focuses on one version of the DTLS protocol.

insufficient\_security: This error message indicates that the server requires ciphers to be more secure. This document does, however, specify the only acceptable ciphersuites and client implementations must support them.

user\_canceled: The IoT devices in focus of this specification are assumed to be unattended.

## 8. Session Resumption

Session resumption is a feature of DTLS that allows a client to continue with an earlier established session state. The resulting exchange is shown in Figure 5. In addition, the server may choose not to do a cookie exchange when a session is resumed. Still, clients have to be prepared to do a cookie exchange with every handshake.

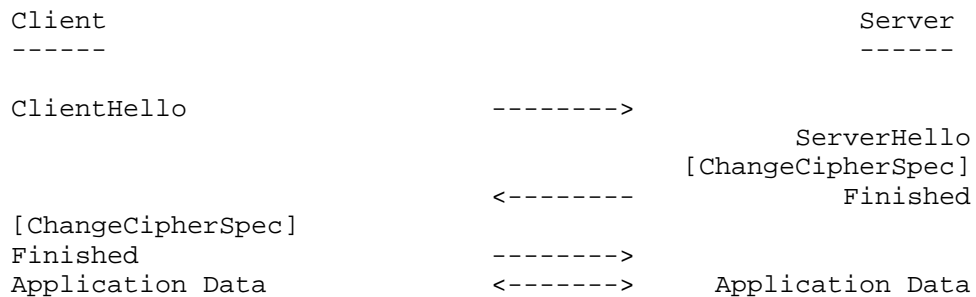


Figure 5: DTLS Session Resumption.

Clients **MUST** implement session resumption to improve the performance of the handshake (in terms of reduced number of message exchanges, lower computational overhead, and less bandwidth conserved).

Since the communication model described in Section 2 does not assume that the server is constrained. RFC 5077 [RFC5077] describing TLS session resumption without server-side state is not utilized by this profile.

## 9. TLS Compression

[I-D.sheffer-tls-bcp] recommends to always disable DTLS-level compression due to attacks. For IoT applications compression at the DTLS is not needed since application layer protocols are highly optimized and the compression algorithms at the DTLS layer increase code size and complexity. Hence, for use with this profile compression at the DTLS layer **MUST NOT** be implemented by the DTLS client.

## 10. Perfect Forward Secrecy

Perfect forward secrecy is designed to prevent the compromise of a long-term secret key from affecting the confidentiality of past conversations. The PSK ciphersuite recommended in the CoAP specification [I-D.ietf-core-coap] does not offer this property. [I-D.sheffer-tls-bcp] on the other hand recommends using ciphersuites offering this security property.

QUESTION: Should the PSK ciphersuite offer PFS?

## 11. Keep-Alive

RFC 6520 [RFC6520] defines a heartbeat mechanism to test whether the other peer is still alive. The same mechanism can also be used to perform path MTU discovery.

QUESTION: Do IoT deployments make use of this extension?

## 12. Negotiation and Downgrading Attacks

CoAP demands version 1.2 of DTLS to be used and the earlier version of DTLS is not supported. As such, there is no risk of downgrading to an older version of DTLS. The work described in [I-D.bmoeller-tls-downgrade-scsv] is therefore also not applicable to this environment since there is no legacy server infrastructure to worry about.

QUESTION: Should we say something for non-CoAP use of DTLS?

To prevent the TLS renegotiation attack [RFC5746] clients MUST respond to server-initiated renegotiation attempts with an Alert message (no\_renegotiation) and clients MUST NOT initiate them. TLS and DTLS allows a client and a server who already have a TLS connection to negotiate new parameters, generate new keys, etc by initiating a TLS handshake using a ClientHello message. Renegotiation happens in the existing TLS connection, with the new handshake packets being encrypted along with application data.

## 13. Privacy Considerations

The DTLS handshake exchange conveys various identifiers, which can be observed by an on-path eavesdropper. For example, the DTLS PSK exchange reveals the PSK identity, the supported extensions, the session id, algorithm parameters, etc. When session resumption is used then individual TLS sessions can be correlated by an on-path adversary. With many IoT deployments it is likely that keying material and their identifiers are persistent over a longer period of time due to the cost of updating software on these devices.

User participation with many IoT deployments poses a challenge since many of the IoT devices operate unattended, even though they will initially be enabled by a human. The ability to control data sharing and to configure preference will have to be provided at a system level rather than at the level of a DTLS profile, which is the scope of this document. Quite naturally, the use of DTLS with mutual authentication will allow a TLS server to collect authentication information about the IoT device (potentially over a long period of time). While this strong form of authentication will prevent mis-

attribution it also allows strong identification. This device-related data collection (e.g., sensor recordings) will be associated with other data to be truly useful and this extra data might include personal data about the owner of the device or data about the environment it senses. Consequently, the data stored on the server-side will be vulnerable to stored data compromise. For the communication between the client and the server this specification prevents eavesdroppers to gain access to the communication content. While the PSK-based ciphersuite does not provide PFS the asymmetric version does. No explicit techniques, such as extra padding, have been provided to make traffic analysis more difficult.

#### 14. Security Considerations

This entire document is about security.

The TLS protocol requires random numbers to be available during the protocol run. For example, during the ClientHello and the ServerHello exchange the client and the server exchange random numbers. Also, the use of the Diffie Hellman exchange requires random numbers during the key pair generation. Special care has to be paid when generating random numbers in embedded systems as many entropy sources available on desktop operating systems or mobile devices might be missing, as described in [Heninger]. Consequently, if not enough time is given during system start time to fill the entropy pool then the output might be predictable and repeatable, for example leading to the same keys generated again and again. Guidelines and requirements for random number generation can be found in RFC 4086 [RFC4086].

We would also like to point out that designing a software update mechanism into an IoT system is crucial to ensure that both functionality can be enhanced and that potential vulnerabilities can be fixed. This software update mechanism is also useful for changing configuration information, for example, trust anchors and other keying related information.

#### 15. IANA Considerations

This document includes no request to IANA.

#### 16. Acknowledgements

Thanks to Rene Hummen, Sye Loong Keoh, Sandeep Kumar, Eric Rescorla, Zach Shelby, and Sean Turner for helpful comments and discussions that have shaped the document.

## 17. References

### 17.1. Normative References

- [EUI64] "GUIDELINES FOR 64-BIT GLOBAL IDENTIFIER (EUI-64) REGISTRATION AUTHORITY", April 2010, <<http://standards.ieee.org/regauth/oui/tutorials/EUI64.html>>.
- [I-D.ietf-core-coap] Shelby, Z., Hartke, K., and C. Bormann, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-18 (work in progress), June 2013.
- [I-D.ietf-tls-cached-info] Santesson, S. and H. Tschofenig, "Transport Layer Security (TLS) Cached Information Extension", draft-ietf-tls-cached-info-15 (work in progress), October 2013.
- [I-D.ietf-tls-oob-pubkey] Wouters, P., Tschofenig, H., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", draft-ietf-tls-oob-pubkey-11 (work in progress), January 2014.
- [I-D.mcgreew-tls-aes-ccm-ecc] McGrew, D., Bailey, D., Campagna, M., and R. Dugal, "AES-CCM ECC Cipher Suites for TLS", draft-mcgreew-tls-aes-ccm-ecc-08 (work in progress), February 2014.
- [RFC4279] Eronen, P. and H. Tschofenig, "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, December 2005.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [RFC5746] Rescorla, E., Ray, M., Dispensa, S., and N. Oskov, "Transport Layer Security (TLS) Renegotiation Indication Extension", RFC 5746, February 2010.
- [RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.



- [RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC6520] Seggellmann, R., Tuexen, M., and M. Williams, "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension", RFC 6520, February 2012.

## 17.2. Informative References

- [Heninger]  
Heninger, N., Durumeric, Z., Wustrow, E., and A. Halderman, "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices", 21st USENIX Security Symposium, <https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/heninger>, 2012.
- [I-D.bmoeller-tls-downgrade-scsv]  
Moeller, B. and A. Langley, "TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks", draft-bmoeller-tls-downgrade-scsv-01 (work in progress), November 2013.
- [I-D.campagna-suitee]  
Campagna, M., "A Cryptographic Suite for Embedded Systems (SuiteE)", draft-campagna-suitee-04 (work in progress), October 2012.
- [I-D.cooper-ietf-privacy-requirements]  
Cooper, A., Farrell, S., and S. Turner, "Privacy Requirements for IETF Protocols", draft-cooper-ietf-privacy-requirements-01 (work in progress), October 2013.
- [I-D.greevenbosch-tls-ocsp-lite]  
Greevenbosch, B., "OCSP-lite - Revocation of raw public keys", draft-greevenbosch-tls-ocsp-lite-01 (work in progress), June 2013.
- [I-D.gutmann-tls-encrypt-then-mac]  
Gutmann, P., "Encrypt-then-MAC for TLS and DTLS", draft-gutmann-tls-encrypt-then-mac-05 (work in progress), December 2013.

- [I-D.hummen-dtls-extended-session-resumption]  
Hummen, R., Gilger, J., and H. Shafagh, "Extended DTLS Session Resumption for Constrained Network Environments", draft-hummen-dtls-extended-session-resumption-01 (work in progress), October 2013.
- [I-D.ietf-lwig-guidance]  
Bormann, C., "Guidance for Light-Weight Implementations of the Internet Protocol Suite", draft-ietf-lwig-guidance-03 (work in progress), February 2013.
- [I-D.ietf-lwig-terminology]  
Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained Node Networks", draft-ietf-lwig-terminology-06 (work in progress), December 2013.
- [I-D.ietf-lwig-tls-minimal]  
Kumar, S., Keoh, S., and H. Tschofenig, "A Hitchhiker's Guide to the (Datagram) Transport Layer Security Protocol for Smart Objects and Constrained Node Networks", draft-ietf-lwig-tls-minimal-00 (work in progress), September 2013.
- [I-D.ietf-tls-applayerprotoneg]  
Friedl, S., Popov, A., Langley, A., and S. Emile, "Transport Layer Security (TLS) Application Layer Protocol Negotiation Extension", draft-ietf-tls-applayerprotoneg-04 (work in progress), January 2014.
- [I-D.pettersen-tls-version-rollback-removal]  
Pettersen, Y., "Managing and removing automatic version rollback in TLS Clients", draft-pettersen-tls-version-rollback-removal-02 (work in progress), August 2013.
- [I-D.sheffer-tls-bcp]  
Sheffer, Y. and R. Holz, "Recommendations for Secure Use of TLS and DTLS", draft-sheffer-tls-bcp-01 (work in progress), September 2013.
- [IANA-TLS]  
IANA, "TLS Cipher Suite Registry", <http://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml#tls-parameters-4>, 2014.
- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

- [RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.
- [RFC4492] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, May 2006.
- [RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, January 2008.
- [RFC5289] Rescorla, E., "TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)", RFC 5289, August 2008.
- [RFC5934] Housley, R., Ashmore, S., and C. Wallace, "Trust Anchor Management Protocol (TAMP)", RFC 5934, August 2010.
- [RFC6090] McGrew, D., Igoe, K., and M. Salter, "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090, February 2011.
- [RFC6961] Pettersen, Y., "The Transport Layer Security (TLS) Multiple Certificate Status Request Extension", RFC 6961, June 2013.
- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, July 2013.

#### Authors' Addresses

Klaus Hartke  
Universitaet Bremen TZI  
Postfach 330440  
Bremen D-28359  
Germany

Phone: +49-421-218-63905  
Email: hartke@tzi.org

Hannes Tschofenig  
ARM Ltd.  
110 Fulbourn Rd  
Cambridge CB1 9NJ  
Great Britain

Email: [Hannes.tschofenig@gmx.net](mailto:Hannes.tschofenig@gmx.net)  
URI: <http://www.tschofenig.priv.at>

DICE Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: August 18, 2014

S. Keoh  
University of Glasgow Singapore  
S. Kumar, Ed.  
O. Garcia-Morchon  
E. Dijk  
Philips Research  
A. Rahman  
InterDigital  
February 14, 2014

DTLS-based Multicast Security for Low-Power and Lossy Networks (LLNs)  
draft-keoh-dice-multicast-security-05

Abstract

The CoAP and 6LoWPAN standards are fast emerging as key protocols in the area of resource-constrained devices. Such IP-based systems are foreseen to be used for building and lighting control systems where wireless devices interconnect with each other, forming low-power and lossy networks (LLNs). Both multicast and its security are key needs in these networks. This draft presents a method for securing IPv6 multicast communication in LLNs based on the DTLS which is already supported for unicast communication for CoAP devices. This draft deals with the adaptation of the DTLS record layer to protect multicast group communication, assuming that all group members already have the group security association parameters in their possession. The adapted DTLS record layer provides message confidentiality, integrity and replay protection to group messages using the group keying material before sending the message via IPv6 multicast to the group.

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 August 18, 2014.

## Copyright Notice

Copyright (c) 2014 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 Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
1.1. Terminology . . . . .	4
1.2. Outline . . . . .	5
2. Use Cases and Requirements . . . . .	5
2.1. Group Communication Use Cases . . . . .	5
2.2. Security Requirements . . . . .	6
3. Overview of DTLS-based Secure Multicast . . . . .	8
3.1. IP Multicast . . . . .	9
3.2. Securing Multicast in LLNs . . . . .	10
4. Multicast Data Security . . . . .	11
4.1. SecurityParameter derivation . . . . .	11
4.2. Record layer adaptation . . . . .	12
4.3. Sending Secure Multicast Messages . . . . .	14
4.4. Receiving Secure Multicast Messages . . . . .	14
4.5. Proxy Operation . . . . .	15
5. IANA Considerations . . . . .	15
6. Security Considerations . . . . .	16
6.1. Group level security . . . . .	16
6.2. Late joiners . . . . .	16
6.3. Uniqueness of SenderIDs . . . . .	17
6.4. Reduced sequence number space . . . . .	17
7. Acknowledgements . . . . .	17
8. References . . . . .	17
8.1. Normative References . . . . .	17
8.2. Informative References . . . . .	18
Appendix A. Change Log . . . . .	20
Authors' Addresses . . . . .	20

## 1. Introduction

There is an increased use of wireless control networks in environmental monitoring, industrial automation, lighting controls and building management systems. This is mainly driven by the fact that the independence from physical control wires allows for freedom of placement, portability and for reducing the cost of installation as less cable placement and drilling are required. Consequently, there is an ever growing number of electronic devices, sensors and actuators that have become Internet connected, thus creating a trend towards the Internet-of-Things (IoT). These connected devices are equipped with communication capability that enables them to interact with each other as well as with the wider Internet services. However, the devices in such wireless control 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 allow devices to directly 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 building and lighting control system, the heating, ventilation, air-conditioning and 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 Constrained Application Protocol (CoAP) [I-D.ietf-core-coap] sent over IP- multicast [I-D.ietf-core-groupcomm].

Currently, CoAP messages are protected using Datagram Transport Layer Security (DTLS) [RFC6347]. However, DTLS is currently 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). There have been a lot of previous efforts in IETF to standardize mechanisms to secure multicast communication such as [RFC3830], [RFC4082], [RFC3740], [RFC4046], and [RFC4535]. However, these approaches 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 and not for protecting the confidentiality of multicast messages. [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.

This draft describes an approach to use DTLS as mandated in CoAP unicast to also support multicast security. We will assume that all devices in the group already have a group security association parameters based on a key management mechanism which is outside the scope of this draft. This draft focuses primarily on the adaptation of the DTLS record layer to protect multicast messages to be sent to the group, and thus providing confidentiality, integrity and replay protection to the CoAP group messages.

Lastly, even though this draft is written from the perspective of securing CoAP based group communication, it is important to note that DTLS is a powerful and flexible security protocol. Thus use of DTLS-based multicast for application layer protocols other than CoAP are possible as long as they follow the approach outlined in this draft.

### 1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This specification uses the following terminology:

- o Group Controller: The entity that is responsible for creating a multicast group and establishing security associations among authorized group members. It is also responsible for renewing/ updating the multicast group keys.
- o Sender: The Sender is an entity that sends data to the multicast group. In a 1-to-N multicast group only a single sender is authorized to transmit data to the group. In an M-to-N multicast group (where M and N are not necessarily the same value), M group members are authorized to be senders.
- o Listener: The entity that receives multicast messages when listening to a multicast IP address.
- o Security Association (SA): A set of policy and cryptographic keys that provide security services to network traffic that matches



that policy [RFC3740]. A Security Association usually contains 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.
- o Group Security Association (GSA): A bundling of security associations (SAs) that together define how a group communicates securely. [RFC3740]
  - o Keying material: Data that is specified as part of the SA which is needed to establish and maintain a cryptographic security association, such as keys, key pairs, and IVs [RFC4949].

## 1.2. Outline

This draft is structured as follows: Section 2 motivates the proposed solution with group communication use cases in LLNs and derives a set of requirements. Section 3 provides an overview of the proposed DTLS-based multicast security assuming that all devices in the group already have a group security association parameters in their possession. In Section 4, we describe the details of the adaptation of DTLS record layer for confidentiality and integrity protection of the multicast messages. Section 6 presents the security considerations.

## 2. Use Cases and Requirements

This section defines the use cases for group communication in LLNs and specifies a set of security requirements for these use cases.

### 2.1. Group Communication Use Cases

The "Group Communication for CoAP" draft [I-D.ietf-core-groupcomm] 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 multicast group communication use cases with detailed descriptions and some are listed here briefly:

- a. Lighting control: enabling synchronous operation of a group of 6LoWPAN [RFC4944] [RFC6282] connected lights in a room/floor/building. This ensures that the light preset like on/off/dim-level of a large group of luminaries are changed at the same time, hence providing a visual synchronicity of light effects to the user.
- b. Parameter update: configuration settings of a group of similar devices are updated simultaneously and efficiently.
- c. Device and Service discovery: information about the devices in the local network and their capabilities can be queried and requested using multicast, e.g. by a commissioning device. The responses are sent back in unicast.

Elaborating on one of the main use cases that this document addresses, 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/floor 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 multicast groups to be formed across different physical subnets (which may be individually protected with L2 security). In such a multicast group, group messages can traverse from one physical subnet to another physical subnet through a IPv6 backbone which may not be protected. Additionally, other non-lighting devices (like window blind controls) may share the physical subnet for networking.

## 2.2. Security Requirements

The "Miscellaneous CoAP Group Communication Topics" draft [I-D.dijk-core-groupcomm-misc] 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. Multicast 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 the simple lighting control use case, the switch is the only entity that is responsible for sending control commands

to a group of lighting devices. In more advanced lighting control use cases, a N-to-M 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. Multicast group size: The security solutions should support the typical group sizes that "Group Communication for CoAP" draft [I-D.ietf-core-groupcomm] intends to support. Group size is the combination of the number of Senders and Listeners in a group with possible overlap (a Sender can also be a Listener but need not be always). In LLN use cases mentioned in the document, the number of Senders (normally the controlling devices) is much smaller than the number of Listeners (the controlled devices). A security solution that supports 1 to 50 Senders would cover the group sizes required for most use cases that are relevant for this document. The total number of group devices must be in the range of 2 to 100 devices. Groups larger than these should be divided into smaller independent multicast groups such as grouping lights of a building per floor.
- c. Establishment of a GSA: A secure mechanism must be used to distribute keying materials, multicast security policies and security parameters to members of a multicast group. A GSA must be established by the group controller (which manages the multicast group) among the group members. The 6LoWPAN border router, a device in the 6LoWPAN, or a remote server outside the 6LoWPAN could play the role of the group controller. However, GSA establishment is outside the scope of this draft, and it is anticipated that an activity in IETF dedicated to the design of a generic key management scheme for the LLN will include this feature preferably based on [RFC3740], [RFC4046] and [RFC4535].
- d. Multicast data confidentiality: Multicast message should be encrypted, as some control commands when sent in the clear could pose unforeseen privacy risks to the users of the system.
- e. Multicast data replay protection: It must not be possible to replay a multicast message as this would disrupt the operation of the group communication.
- f. Multicast data group authentication and integrity: It is essential to ensure that a multicast message originated from a member of the group and that messages have not been tampered with by attackers who are not members. The multicast group key which is known to all group members is used to provide authenticity to the multicast messages (e.g., using a Message Authentication Code, MAC). This assumes that all other group members are trusted not to tamper with the multicast message.

- g. Multicast data security ciphersuite: All group members must use the same ciphersuite to protect the authenticity, integrity and confidentiality of multicast messages. The ciphersuite is part of the GSA. Typically authenticity is more important than confidentiality in LLNs. Therefore the proposed multicast data security protocol must support at least ciphersuites with MAC only (NULL encryption) and AEAD [RFC5116] ciphersuites. Other ciphersuites that are defined for data record security in DTLS should also be preferably supported.
- h. Multicast data source authentication: Source authenticity is required if the group members are assumed to be untrusted and can tamper with the multicast messages. This can happen if nodes of the group can be easily compromised. Source authenticity helps to minimize the risk of any node compromise leading to the compromise of the whole multicast group. Source authenticity can be typically provided using public-key cryptography in which every multicast message is signed by the sender. Alternatively, a lightweight broadcast authentication, i.e., TESLA [RFC4082] can be deployed, however it requires devices in the multicast group to have a trusted clock and have the ability to loosely synchronize their clocks with the sender. Source authenticity mechanisms should be preferably defined at the application layer. The transport layer group level security can provide an additional layer of security for the source authenticity mechanism against DoS attacks. However, even with source authenticity the risk still remains that compromise of a sender can still compromise the whole group.
- i. Forward security: Devices that leave the group should not have access to any future GSAs. This ensures that a past member device cannot continue to decrypt confidential data that is sent in the group. It also ensures that this device cannot send encrypted and/or integrity protected data after it leaves the group. The GSA update mechanism has to be defined as part of the key management scheme.
- j. Backward confidentiality: A new device joining the group should not have access to any old GSAs. This ensures that a new member device cannot decrypt data sent before it joins the group. The key management scheme should ensure that the GSA is updated to ensure backward confidentiality.

### 3. Overview of DTLS-based Secure Multicast

The goal of this draft is to secure CoAP Group communication over 6LoWPAN networks, by extending the use of the DTLS security protocol to allow for the use of DTLS record layer with minimal adaptation.

The IETF CORE WG has selected DTLS [RFC6347] as the default must-implement security protocol for securing CoAP, therefore it is desirable that DTLS be extended to facilitate CoAP-based group communication. Reusing DTLS for different purposes while guaranteeing the required security properties can avoid the need to implement multiple security protocols and this is especially beneficial when the target deployment consists of resource-constrained embedded devices. This section first describes group communication based on IP multicast, and subsequently sketches a solution for securing group communication using DTLS.

### 3.1. IP Multicast

Devices in the LLN are categorized into two roles, (1) sender and (2) listener. Any node in the LLN may have one of these roles, or both roles. The application(s) running on a device basically determine these roles by the function calls they execute on the IP stack of the device.

In principle, a sender or listener does not require any prior access procedures or authentication to send or listen to a multicast message [RFC5374]. A sender to an IPv6 multicast group sets the destination of the packet to an IPv6 address that has been allocated for IPv6 multicast. A device becomes a listener by "joining" to the specific IPv6 multicast group by registering with a network routing device, signaling its intent to receive packets sent to that particular IPv6 multicast group. Figure 1 depicts a 1-to-N multicast communication and the roles of the nodes. Any device can in principle decide to listen to any IPv6 multicast address. This also means applications on the other devices do not know, or do not get notified, when new listeners join the LLN. More details on the IPv6 multicast and CoAP group communication can be found in [I-D.ietf-core-groupcomm]. This draft does not intend to modify any of the underlying group communication or multicast routing protocols.

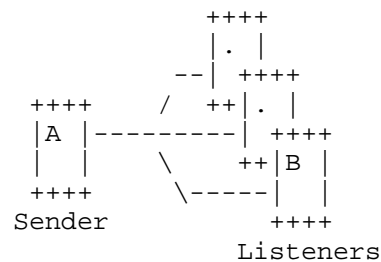


Figure 1: The roles of nodes in a 1-to-N multicast communication topology

### 3.2. Securing Multicast in LLNs

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 to add them to the new group. Additionally it distributes the GSA consisting of keying material, security policies security parameters and ciphersuites using a standardized key management for LLN which is outside the scope of this draft. Additional ciphersuites may need to be defined to convey the bulk cipher algorithm, MAC algorithm and key lengths within the key management protocol. We provide two examples of ciphersuites (based on the security requirements) that could be defined as part of a future key management mechanism:

```

Ciphersuite MTS_WITH_AES_128_CCM_8 = {TBD1, TBD2}
Ciphersuite MTS_WITH_NULL_SHA256   = {TBD3, TBD4}

```

Ciphersuite MTS\_WITH\_AES\_128\_CCM\_8 is used to provide confidentiality, integrity and authenticity to the multicast messages where the encryption algorithm is AES [FIPS.197.2001], key length is 128-bit, and the authentication function is CCM [RFC6655] with a Message Authentication Code (MAC) length of 8 octets. Similar to [RFC4785], the ciphersuite MTS\_WITH\_NULL\_SHA is used when confidentiality of multicast messages is not required, it only provides integrity and authenticity protection to the multicast message. When this ciphersuite is used, the message is not encrypted but the MAC must be included in which it is computed using a HMAC

[RFC2104] that is based on Secure Hash Function SHA256 [FIPS.180-2.2002]. Depending on the future needs, other ciphersuites with different cipher algorithms and MAC length may be supported.

Senders in the group can encrypt and authenticate the CoAP group messages from the application using the keying material into the DTLS record. The authenticated encrypted message is passed down to the lower layer of the IPv6 protocol stack for transmission to the multicast address as depicted in Figure 2. The listeners when receiving the message, use the multicast IPv6 destination address and port (i.e., Multicast identifier) to look up the GSA needed for that group connection. The received message is then decrypted and the authenticity is verified using the keying material for that connection.

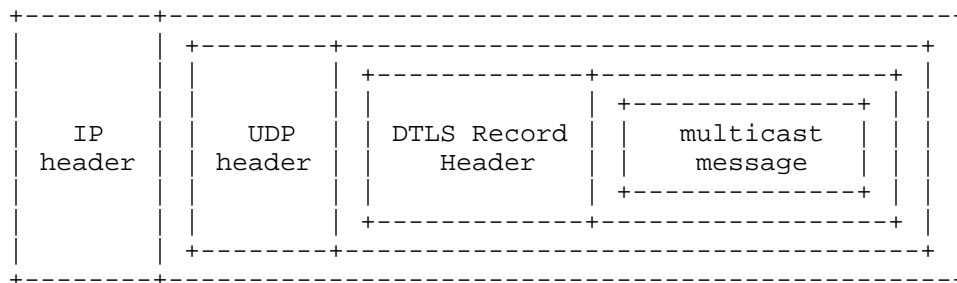


Figure 2: Sending a multicast message protected using DTLS Record Layer

#### 4. Multicast Data Security

This section describes in detail the use of DTLS record layer to secure multicast messages. This assumes that group membership has been configured by the group controller, and all member devices in the group have the GSA.

#### 4.1. SecurityParameter derivation

The GSA is used to derive the same "SecurityParameters" structure as defined in [RFC5246] for all devices.

The `SecurityParameters.ConnectionEnd` should be set to "server" for senders and "client" for listeners. The current read and write states can be derived from `SecurityParameters` by generating the six keying materials:

```

client write MAC key
server write MAC key
client write encryption key
server write encryption key
client write IV
server write IV

```

This requires that the `client_random` and `server_random` within the `SecurityParameters` are also set to the same value for all devices as part of the GSA to derive the same keying material for all devices in the group with the PRF function defined in Section 6.3 of [RFC5246] . Alternatively, the GSA could directly include the above six keying material when being configured in all group devices.

The current read and write states are instantiated for all group members based on the keying material and according to their roles: senders use "server write" parameters for the write state and listeners use "server write" parameters for the read state. Additionally each connection state contains the sequence number which is incremented for each record sent; the first record sent has the sequence number 0.

#### 4.2. Record layer adaptation

In this section, we describe in detail the adaptation of the DTLS Record layer to enable multiple senders in the group to securely send information using a common group key, while preserving the confidentiality, integrity and freshness of the messages.

The following Figure 3 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 Type	Version Ma   Mi	epoch	seq_ number	Length	Ciphertext (Enc)	MAC (Enc)

Figure 3: The DTLS record layer header and optionally encrypted payload and MAC

The `epoch` is fixed by the DTLS handshake and the `seq_number` is initialized to 0. The `seq_number` is increased by one whenever a



sender sends a new record message. This is the mechanism of DTLS to detect message replay. Finally, the message is protected (encrypted and authenticated with a MAC) using the session keys in the "server write" parameters.

One of the problems with supporting multiple senders is that, the `seq_number` used by senders need to be synchronized to avoid their reuse, otherwise packets sent by different senders may get discarded as replayed packets. Further, the bigger problem is using a single key in a multiple sender scenario leads to nonce reuse in AEAD cipher suites like AES-CCM [RFC6655] and AES-GCM [RFC5288] as defined in DTLS. Nonce reuse can completely break the security of these cipher suites.

According to the AES-CCM for TLS, Section 3 [RFC6655], the CCMNonce is a combination of a salt value and the sequence number.

```
struct {
    opaque salt[4];
    opaque nonce_explicit[8];
} CCMNonce;
```

The salt is the "client write IV" (when the client is sending) or the "server write IV" (when the server is sending) as defined in the "SecurityParameters". Further [RFC6655] requires that the value of the `nonce_explicit` MUST be distinct for each distinct invocation of the CCM encrypt function for any fixed key. When the `nonce_explicit` is equal to the sequence number of the TLS packets, the CCMNonce has the structure as below:

```
struct {
    uint32 client_write_IV; // low order 32-bits
    uint64 seq_num;         // TLS sequence number
} CCMClientNonce.

struct {
    uint32 server_write_IV; // low order 32-bits
    uint64 seq_num;         // TLS sequence number
} CCMServerNonce.
```

In DTLS, the 64-bit sequence number is the 16-bit epoch concatenated with the 48-bit `seq_number`. Therefore to prevent that the CCMNonce is reused, either all senders need to synchronize or separate non-overlapping sequence number spaces need to be created for each sender. Synchronization between senders is especially hard in LLN and therefore we go for the second approach of separating the

sequence number spaces by embedding a unique sender identifier in the sequence number as suggested in [RFC5288].

Thus in addition to configuring each device in the group with the GSA, the controller needs to assign a unique SenderID to each device which has the sender role in the group. The size of the SenderID is 1-octet based on the requirement for the supported group size mentioned in Section 2.2. The list of SenderIDs are then distributed to all the group members by the controller.

The existing DTLS record layer header is adapted such that the 6-octet seq\_number field is split into a 1-octet SenderID field and a 5-octet "truncated" trunc\_seq\_number field. Figure 4 illustrates the adapted DTLS record layer header.

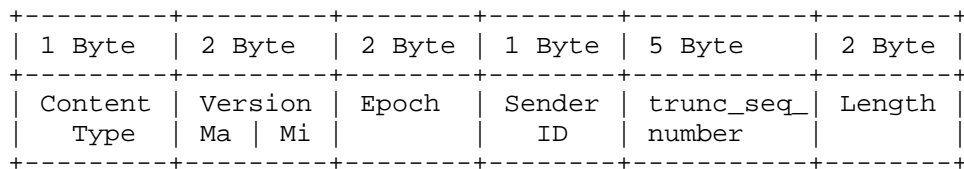


Figure 4: The adapted DTLS record layer header

#### 4.3. Sending Secure Multicast Messages

Senders in the multicast group when sending a CoAP group message from the application, create the adapted DTLS record payload based on the "server write" parameters. Each sender in the group uses its own unique SenderID in the DTLS record layer header. It also manages its own epoch and trunc\_seq\_number in the "server write" connection state; the first record sent has the trunc\_seq\_number 0. After creating the DTLS record, the trunc\_seq\_number is incremented in the "server write" connection state. The adapted DTLS record is then passed down to UDP and IPv6 layer for transmission on the multicast IPv6 destination address and port.

#### 4.4. 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 packet. This is fundamentally 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 SenderIDs. The keying material is same for all senders however the epoch and the trunc\_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. By knowing the keys, the listeners decrypt and check the MAC of the message. This guarantees that no one outside the group has spoofed the SenderID, as it is protected by the MAC. Subsequently, by authenticating the SenderID field, the listeners retrieve the "client read" connection state which contains the last stored epoch and trunc\_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 trunc\_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 epoch and the trunc\_seq\_number in the corresponding "client read" connection state are updated as well.

#### 4.5. Proxy Operation

CoAP allows a client to designate a (forward) proxy to process its CoAP request for both unicast and multicast scenarios as described in Section 2.10 of [I-D.ietf-core-groupcomm]. In this case, the proxy (and not the client) appears as the originating point to the destination server for the CoAP request.

As mentioned in Section 11.2 of [I-D.ietf-core-coap], proxies are by their nature men-in-the-middle and break DTLS protection of CoAP message exchanges. Therefore, in a DTLS-based multicast scenario involving a proxy, a two-step approach is required. First, the client will send a unicast DTLS request to the proxy. The proxy will then receive and decrypt the unicast message. The proxy will then take the contents of the received message and create a new multicast message and secure it using DTLS-based multicast before sending it out to the group. For this approach to work properly, the client needs to be able to designate the proxy as an authorized sender. The mechanism for this authorization is outside the scope of this draft.

#### 5. IANA Considerations

This memo includes no request to IANA.

## 6. Security Considerations

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

### 6.1. Group level security

This proposal uses a single group key to protect communication within the group. This requires that all group members are trusted, for e.g. they do not forge messages as a different sender in the group. In many usecase, the devices in a group belong to a common authority and are configured by a commissioner. In a professional lighting scenario, the roles of the senders and listeners are configured by the lighting commissioner and devices follow those roles.

The use of the protocol should take into consideration the risk of compromise of a group device in a deployment scenario. Therefore the group size should be limited to 100 devices unless additional source authenticity mechanisms are implemented at the application layer. Further, the damage due to a compromised key can be limited by increasing the frequency of rekeying based on the unique unicast key-pair shared by each device with the controller. Additionally the risk of compromise is reduced when deployments are in physically secured locations, like lighting inside office buildings.

### 6.2. Late joiners

Listeners who are late joiners to a multicast group, do not know the current epoch and trunc\_seq\_number being used by different senders. When they receive a packet from a sender with a random trunc\_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, we can use the techniques similar to AERO [I-D.mcgregre-aero] where the late joining listener on receiving the first packet from a particular sender, initialize its last seen epoch and trunc\_seq\_number in the "client read" state for that sender, however does not pass this packet to the application layer and instead drops it. This provides a reference point to identify if future packets are fresher than the last seen packet. Alternatively, the group controller which can act as a listener in the multicast group can maintain the epoch and trunc\_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 trunc\_seq\_numbers as part of the GSA.

### 6.3. Uniqueness of SenderIDs

It is important that SenderIDs are unique to maintain the security properties of the DTLS record layer messages. However in the event that two or more senders are configured with the same SenderID, a mechanism needs to be present to avoid a security weakness and recover from the situation. One such mechanism is that all senders of the multicast group are also listeners. This allows a sender which receives a packet from a different device with its own SenderID in the DTLS header to become aware of a clash. Once aware, the sender can inform the controller on a secure channel about the clash along with the source IP address. The controller can then provide a different SenderID to either device or both.

### 6.4. Reduced sequence number space

The DTLS record layer `seq_number` is truncated from 6 octets to 5 octets. This reduction of the `seq_number` space should be taken into account to ensure that epoch is incremented before the `trunc_seq_number` wraps over. The sender or the controller can increase the epoch number by sending a `ChangeCipherSpec` message whenever the `trunc_seq_number` has been exhausted. This should be done as part of the key management mechanism which is not defined in this draft.

## 7. Acknowledgements

The authors greatly acknowledge discussion, comments and feedback from Dee Denteneer, Peter van der Stok, Zach Shelby and Michael StJohns. Additionally thank David McGrew for suggesting options for recovering from a SenderID clash, and John Foley for the extensive review and pointing us to the AERO draft. We also appreciate prototyping and implementation efforts by Pedro Moreno Sanchez who worked as an intern at Philips Research.

## 8. References

### 8.1. Normative References

- [I-D.ietf-core-coap]  
Shelby, Z., Hartke, K., and C. Bormann, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-18 (work in progress), June 2013.
- [I-D.ietf-core-groupcomm]  
Rahman, A. and E. Dijk, "Group Communication for CoAP", draft-ietf-core-groupcomm-18 (work in progress), December 2013.

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

## 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-05 (work in progress), December 2013.
- [I-D.ietf-core-resource-directory]  
Shelby, Z., Bormann, C., and S. Krco, "CoRE Resource Directory", draft-ietf-core-resource-directory-01 (work in progress), December 2013.
- [I-D.mcgrew-aero]  
McGrew, D. and J. Foley, "Authenticated Encryption with Replay prOtectiOn (AERO)", draft-mcgrew-aero-00 (work in progress), October 2013.

- [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.

#### Appendix A. Change Log

(To be removed by RFC editor before publication.)

Changes from keoh-03 to keoh-04:

- o Added description of Proxy operation in a DTLS-based multicast scenario in Section 4.5 (Proxy Operation).
- o Corrected text in Section 2.2 (Security Requirements), item "h", to indicate that multicast source authentication is not specified in this version of the draft.
- o Clarified that draft is written primarily for securing of CoAP based group communication, but that other protocols may also be supported if they have similar characteristics. See Section 1 (Introduction).
- o Ran IETF spell checker and ID-Nits tools and corrected various issues throughout the document.
- o Various editorial updates.

Changes from keoh-04 to keoh-05:

- o In section 2.1, removed the firmware upgrade usecase and clarified the commissioning use case. The lighting use-case expanded with shared and multiple subnets issues.
- o In Section 2.2, (b) reduced the group size to 100; (h) clarified data source authenticity
- o Added new Section 6.1 (Group level security) in security considerations to make clear the risks of the single group key.

#### Authors' Addresses

Sye Loong Keoh  
University of Glasgow Singapore  
Republic PolyTechnic, 9 Woodlands Ave 9  
Singapore 838964  
SG

Email: SyeLoong.Keoh@glasgow.ac.uk



Sandeep S. Kumar (editor)  
Philips Research  
High Tech Campus 34  
Eindhoven 5656 AE  
NL

Email: sandeep.kumar@philips.com

Oscar Garcia-Morchon  
Philips Research  
High Tech Campus 34  
Eindhoven 5656 AE  
NL

Email: oscar.garcia@philips.com

Esko Dijk  
Philips Research  
High Tech Campus 34  
Eindhoven 5656 AE  
NL

Email: esko.dijk@philips.com

Akbar Rahman  
InterDigital  
1000 Sherbrooke Street West  
Montreal H3A 3G4  
CA

Email: akbar.rahman@interdigital.com