CoRE Working Group D. Garcia
Internet-Draft S. Matheu
Intended status: Experimental R. Marin
Expires: June 9, 2017 University of Murcia
December 6, 2016

# Application Layer Security for CoAP using the (D)TLS Record Layer
## draft-garcia-core-app-layer-sec-with-dtls-record-00

Abstract

This document briefly describes an idea to provide Application-Layer Security for CoAP using (D)TLS Record Layer, assuming it is operative in two CoAP endpoints.

Status of This Memo



This Internet-Draft will expire on June 9, 2017.

Copyright Notice

Table of Contents



## 1. Introduction

Secure communications in constrained scenarios is subject of current interest since the restrictions in those kinds of networks motivates rethinking the solutions that up to now have been used in networks that do not suffer from very stringent requirements. (D)TLS [RFC6347][RFC5246] is a standard proposed to secure the communications of CoAP and suitable for end-to-end communications unless a CoAP proxy participates in the communication. To overcome this problem [I-D.ietf-core-object-security] propose Object Security for CoAP (OSCOAP) to allow end-to-end security between two CoAP endpoints in case of a CoAP proxy intermediating between them.

In this document we explore that possibility of providing CoAP security at application layer, assuming a (D)TLS Record Layer is operative (i.e. have the required keys) in both CoAP endpoints. One possibility to "activate" the (D)TLS Record Layer is running (D)TLS handshake over CoAP, as mentioned in CoDTLS [I-D.schmertmann-dice-codtls]. Other (more challenging) options are discussed in [I-D.bhattacharyya-dice-less-on-coap]

### 1.1. Requirements Language

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].

## 2. Application Layer Security for CoAP with (D)TLS Record Protocol

To achieve application layer security using the (D)TLS Record, we assume the (D)TLS [RFC6347] [RFC5246] Record layer is already activated, using a protocol such as CoDTLS [I-D.schmertmann-dice-codtls]. Once we have the (D)TLS Record Layer

active, the next step is to define how the CoAP message will be secured end-to-end using the (D)TLS Record Layer.

The entire CoAP message generated by a CoAP sender will need to arrive to the CoAP recipient, achieving integrity and confidentiality for certain parts of the CoAP message (specific CoAP options and payload) excluding the options CoAP intermediaries (proxies) will need to understand to process the CoAP message correctly. The CoAP header would need to arrive maintaining the semantics and version of the protocol.

## 2.1. CoAP Fields to protect

Here we discuss how the CoAP message is going to be processed to achieve application layer security. How each part of the CoAP message (Header, Options and Payload) is treated and which options are protected an which ones are left unprotected using the (D)TLS Record Layer. Following the procedure specified in OSCOAP [I-D.ietf-core-object-security], we protect all options that are intended to be read by the CoAP recipient.

o CoAP Header: version and code are protected.

o CoAP Options: All the options that can be modified by the proxy are left unprotected. All options that are intended for the the CoAP recipient are protected. There might be the case there an option is both left unprotected for the proxy to process and is also intended for the CoAP recipient to see.

o CoAP Payload: The payload is always protected.

Similarly to OSCON [I-D.ietf-core-object-security], it would be possible to only encrypt the payload of the original CoAP message.

# 3. Processing a CoAP message with the (D)TLS Record

In this section we analyze how the CoAP message is processed and protected using the (D)TLS Record. In Figure 1 we can see how from the Original CoAP Header we obtain the fields that have to be protected (Version and Code) in 2 bytes. We get the Version and the Code. We will have a padding of 6 bits, then the version and code all in 2 bytes.

```
                                                 3
            0 .................................1
Original    +-----+---+-----+------+------------+
CoAP        | Ver | T | TKL | Code | Message ID |
Header      +-----+---+-----+------+------------+

                                 1 1 1 1 1 1
             0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
Protected    +-----------+---+-------------+
CoAP Header  |0 0 0 0 0 0|Ver|    Code     |
(P. Header)  +-----------+---+-------------+
```

Figure 1: Protected Header Fields

This "denatured" CoAP header is concatenated with the CoAP options to be protected and the Payload (if any) of the Original CoAP message (see Figure 2). This array of bytes is sent to the (D)TLS Record Layer to be processed. The resulting information is a protected array of bytes with a (D)TLS Record Header which will process it generating the (D)TLS Record (adding the (D)TLS Record Header). After that, we add the (D)TLS record to the payload of the message to be sent, that will contain the original CoAP Header and only the options available for the proxies. The Protected CoAP message is formed by the Original CoAP Message Header, the Options that are not considered to be protected with this mechanism, then the marker 0xFF and finally the Payload that contains the (D)TLS Record.

```
   Original CoAP Message

          +------------------------------------------+
          |Header| Options |FF|      Payload        |
          +------------------------------------------+
                           |
    CoAP Content           |
     to protect            v

          +------------------+-----------------+
          | P.   |Options to be|  |             |
          |Header| Protected   |FF|  Payload    |
          +------------------------------------+
                           |
                           v
                  ------------------------
                 (   (D)TLS Record Layer  )
                  ------------------------
                           |
                           v
     DTLS Record
          +-------------+------------------------+
          |(D)TLS Record|    CoAP Content*       |
          |   Header    |     to protect         |
          +-------------+------------------------+
                           |
                           |
      Protected            v
      CoAP Message

          +--------+-----------+----+-----------------+
          |        |Unprotected|    |                 |
          | Header |  Options  | FF |  (D)TLS Record  |
          +--------+-----------+----+-----------------+

   * (Ciphered and Integrity protected)
```

Figure 2: Processing CoAP message

Upon reception, the CoAP recipient will get the CoAP Payload of the Protected Message and send it to the (D)TLS Record Layer to obtain the Protected Header and the list of options within the (D)TLS Record. With this information, once it is verified correctly, the CoAP recipient constructs the Original CoAP Message.

## 4. Bootstrapping the (D)TLS Record Layer for Application Security

To enable this solution, the (D)TLS Record Layer in both CoAP endpoints must have a connection to process this information. One alternative is running (D)TLS over CoAP as specified in [I-D.schmertmann-dice-codtls]. However, we consider that it would be possible to define we define a separation between the (D)TLS Handshake and the (D)TLS Record Layer with an interface to be standarized. The (D)TLS Handshake is used to negotiate the parameters to establish a Security Association (SA) in (D)TLS Record Layer. With this interface, we argue that this SA can be set by the (D)TLS Handshake or any other Key Management Protocol (KMP), as we show in Figure 3. This would be similar to the separation in the IPsec architecture [RFC4301], where IKEv2 [RFC7296] is just one of the possible Key Management Protocol to establish IPsec SAs. In fact, IPsec defines a standard API (PFKEY_v2 [RFC2367])) for this purpose.

An example of this separation has been proposed in [I-D.bhattacharyya-dice-less-on-coap]. Another way to benefit from this separation could be that one of the CoAP endpoint has a token (e.g. Kerberos ticket, and ACE token, etc...), with the key material and information (cryptographic keys, algorithms) to start the (D)TLS Record Layer, just presenting the ticket, without the need of running the (D)TLS handshake.

```
+-------------------+-------+-------+--------+
|                   |       |       |        |
| (D)TLS Handshake  | KMP 1 | KMP 2 | KMP i  |
|                   |       |       |        |
+-------------------+-------+-------+--------+
|                                            |
|       (D)TLS Record Layer Interface        |
|                                            |
+--------------------------------------------+
|                                            |
|                                            |
|           (D)TLS Record Layer              |
|                                            |
|                                            |
+--------------------------------------------+
```

Figure 3: (D)TLS Record Layer Interface

## 5. Acknowledgments

This work has been possible partially by the ARMOUR project (FP7-ARMOUR-644852 EU Project) and the Spanish National Project CICYT EDISON (TIN2014-52099-R) granted by the Ministry of Economy and Competitiveness of Spain (including ERDF support).

## 6. Normative References

[I-D.bhattacharyya-dice-less-on-coap]
Bhattacharyya, A., Bandyopadhyay, S., Ukil, A., Bose, T., and A. Pal, "Lightweight Establishment of Secure Session (LESS) on CoAP", draft-bhattacharyya-dice-less-on-coap-00 (work in progress), April 2015.

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security of CoAP (OSCOAP)", draft-ietf-core-object-security-00 (work in progress), October 2016.

[I-D.schmertmann-dice-codtls]
Schmertmann, L., Hartke, K., and C. Bormann, "CoDTLS: DTLS handshakes over CoAP", draft-schmertmann-dice-codtls-01 (work in progress), August 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2367] McDonald, D., Metz, C., and B. Phan, "PF_KEY Key Management API, Version 2", RFC 2367, DOI 10.17487/RFC2367, July 1998, <http://www.rfc-editor.org/info/rfc2367>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <http://www.rfc-editor.org/info/rfc4301>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

## Authors' Addresses

Dan Garcia Carrillo
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia  30100
Spain

Phone: +34 868 88 78 82
Email: dan.garcia@um.es

Sara Nieves Matheu Garcia
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia  30100
Spain

Phone: +34 868 88 78 82
Email: saranieves.matheu@um.es

Rafa Marin-Lopez
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia  30100
Spain

Phone: +34 868 88 85 01
Email: rafa@um.es