EST over secure CoAP (EST-coaps)
draft-vanderstok-ace-coap-est-03

Abstract

Low-resource devices in a Low-power and Lossy Network (LLN) can operate in a mesh network using the IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN) and IEEE 802.15.4 link-layer standards. Enrollment over Secure Transport (EST) [RFC7030] is used for authenticated/authorized endpoint certificate enrollment (and optionally key provisioning) through a Certificate Authority (CA) or Registration Authority (RA). Example low-resource uses cases for EST are: secure bootstrapping and certificate enrollment.

Low-resource devices often use the lightweight Constrained Application Protocol (CoAP) [RFC7252] for message exchanges. This document defines how low-resource devices are expected to use EST over secure CoAP (EST-coaps). 6LoWPAN fragmentation management and extensions to CoAP registries are needed to enable EST-coaps.

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 https://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."
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 June 15, 2018.

Copyright Notice

Copyright (c) 2017 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 (https://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
2. EST operational differences ............................ 4
3. Conformance to RFC7925 profiles ....................... 4
4. Protocol Design and Layering .......................... 5
   4.1. Payload format ....................................... 6
   4.2. Message Bindings ..................................... 6
   4.3. CoAP response codes .................................. 7
   4.4. Message fragmentation ................................ 7
5. Discovery and URI ....................................... 8
6. DTLS Transport Protocol ................................. 10
7. Proxying ................................................. 11
8. Parameters .............................................. 13
9. IANA Considerations ..................................... 13
   9.1. Content-Format registry ............................... 13
   9.2. Resource Type registry ................................ 15
10. Security Considerations ................................. 15
   10.1. proxy considerations ................................ 15
   10.2. EST server considerations ........................... 15
11. Acknowledgements ....................................... 16
12. Change Log ............................................. 16
13. References .............................................. 17
   13.1. Normative References ................................. 17
   13.2. Informative References .............................. 18
Appendix A. EST messages to EST-coaps ..................... 20
   A.1. cacerts ............................................. 21
1. Introduction

Enrollment over Secure Transport (EST) [RFC7030] is used for authenticated/authorized endpoint certificate enrollment (and optionally key provisioning) through a Certificate Authority (CA) or Registration Authority (RA). This functionality is also needed for low resource devices.

IPv6 over Low-power Wireless Personal Area Networks (6LoWPANs) [RFC4944] on IEEE 802.15.4 [IEEE802.15.4] wireless networks is becoming common in many industry application domains such as lighting controls. Although IEEE 802.15.4 defines how security can be enabled between nodes within a single mesh network, it does not specify the provisioning and management of the keys. Therefore, securing a 6LoWPAN network with devices from multiple manufacturers with different provisioning techniques is often tedious and time consuming. An example use case is the application of Bootstrapping of Remote Secure Infrastructures (BRSKI) [I-D.ietf-anima-bootstrapping-keyinfra]. The low resource aspects are detailed for 6tisch in [I-D.ietf-6tisch-minimal-security] and [I-D.ietf-6tisch-dtsecurity-secure-join].

Constrained networks use DTLS [RFC6347], CoAP [RFC7252], and UDP instead of TLS [RFC5246], HTTP [RFC7230] and TCP. EST-coaps replaces the invocations of TLS and HTTP by DTLS and CoAP invocations thus enabling EST for CoAP-based low-resource devices.

Although EST-coaps paves the way for the utilization of EST for constrained devices on constrained networks, some devices will not have enough resources to handle the large payloads that come with EST-coaps. The specification of EST-coaps is intended to ensure that EST works for networks of constrained devices that choose to limit their communications stack to UDP/CoAP. It is up to the network designer to decide which devices execute the EST protocol and which not.

Because the relatively large EST messages cannot be readily transported over constrained (6LoWPAN, LLN) wireless networks, this document specifies the use of CoAP Block-Wise Transfer ("Block") [RFC7959] to fragment EST messages at the application layer.
Support for Observe CoAP options [RFC7641] is out-of-scope for this document. Observe options could be used by the server to notify clients about a change in the cacerts or csr attributes (resources) and might be an area of future work.

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 [RFC2119].

Many of the concepts in this document are taken over from [RFC7030]. Consequently, much text is directly traceable to [RFC7030]. The same document structure is followed to point out the differences and commonalities between EST and EST-coaps.

2. EST operational differences

Only the differences to EST with respect to operational scenarios are described in this section. EST-coaps server differs from EST server as follows:

- Replacement of TLS by DTLS and HTTP by CoAP, resulting in:
  * DTLS-secured CoAP sessions between EST-coaps client and EST-coaps server.

- Only certificate-based client authentication is supported, which results in:
  * The EST-coaps client does not support HTTP Basic authentication (as described in Section 3.2.3 of [RFC7030]).
  * The EST-coaps client does not support authentication at the application layer (as described in Section 3.2.3 of [RFC7030]).

3. Conformance to RFC7925 profiles

This section shows how EST-coaps fits into the profiles of low-resource devices as described in [RFC7925].

EST-coaps can transport certificates and private keys. Private keys can be transported as response to a request to a server-side key generation as described in section 4.4 of [RFC7030].

The mandatory cipher suite for DTLS is TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8 defined in [RFC7251] which is the mandatory-to-implement cipher suite in CoAP. Additionally, the curve
secp256r1 MUST be supported [RFC4492]; this curve is equivalent to the NIST P-256 curve. The hash algorithm is SHA-256. DTLS implementations MUST use the Supported Elliptic Curves and Supported Point Formats Extensions [RFC4492]; the uncompressed point format MUST be supported; [RFC6090] can be used as an implementation method.

The EST-coaps client MUST be configured with an explicit TA database or at least an implicit TA database from its manufacturer. The authentication of the EST-coaps server by the EST-coaps client is based on Certificate authentication in the DTLS handshake. The authentication of the EST-coaps client is based on client certificate in the DTLS handshake. This can either be

- DTLS with a previously issued client certificate (e.g., an existing certificate issued by the EST CA); this could be a common case for simple re-enrollment of clients;
- DTLS with a previously installed certificate (e.g., manufacturer-installed certificate or a certificate issued by some other party);

4. Protocol Design and Layering

EST-coaps uses CoAP to transfer EST messages, aided by Block-Wise Transfer [RFC7959] to transport CoAP messages in blocks thus avoiding (excessive) 6LoWPAN fragmentation of UDP datagrams. The use of "Block" for the transfer of larger EST messages is specified in Section 4.4. The Figure 1 below shows the layered EST-coaps architecture.

```
+------------------------------------------------+
|    EST request/response messages               |
+------------------------------------------------+
|    CoAP for message transfer and signaling     |
+------------------------------------------------+
|    DTLS for transport security                 |
+------------------------------------------------+
|    UDP for transport                           |
+------------------------------------------------+
```

Figure 1: EST-coaps protocol layers

The EST-coaps protocol design follows closely the EST design. The parts supported by EST-coaps are identified by their message types:

- Simple enroll and reenroll, for CA to sign public client-identity key.
CA certificate retrieval, needed to receive the complete set of CA certificates.

CSR Attributes request messages, informs the client of the fields to include in generated CSR.

Server-side key generation messages, to provide a private client-identity key when the client is too restricted or because of lack of an entropy source. [EDNOTE: Encrypting these keys is important. RFC7030 specifies how the private key can be encrypted with CMS using symmetric or asymmetric keys. Mention how symmetric key can be derived for EST server side key generation from the TLS KEM draft.]

4.1. Payload format

The content-format (media type equivalent) of the CoAP message determines which EST message is transported in the CoAP payload. The media types specified in the HTTP Content-Type header (see section 3.2.2 of [RFC7030]) are in EST-coaps specified by the Content-Format Option (12) of CoAP. The combination of URI path-suffix and content-format used for coap MUST map to an allowed combination of path-suffix and media type as defined for EST. The required content-formats for these request and response messages are defined in Section 9. The CoAP response codes are defined in Section 4.3.

EST-coaps is designed for use between low-resource devices using CoAP and hence does not need to send base64-encoded data. Simple binary is more efficient (30% less payload compared to base64) and well supported by CoAP. Therefore, the content formats specification in Section 9 requires the use of binary for all EST-coaps Content-Formats.

4.2. Message Bindings

This section describes the general EST CoAP message characteristics.

It is RECOMMENDED to use CoAP CON messages. This recommendation does not influence the communication efficiency because all EST-coaps messages expect a response.

The Ver, TKL, Token, and Message ID values of the CoAP header are not influenced by EST.

CoAP options are used to convey Uri-Host, Uri-Path, Uri-Port, Content-Format and more in CoAP. The CoAP Options are used to communicate the HTTP fields specified in the EST REST messages.
EST URLs are HTTPS based (https://), in CoAP these will be assumed to be transformed to coaps (coaps://)

Appendix A includes some practical examples of EST messages translated to CoAP.

4.3. CoAP response codes

Section 5.9 of [RFC7252] specifies the mapping of HTTP response codes to CoAP response codes. Every time the HTTP response code 200 is specified in [RFC7030] in response to a GET (POST) request, in EST-coaps the equivalent CoAP response code 2.05 (2.01) MUST be used. Response code HTTP 202 in EST is mapped to CoAP 3.05 as specified in [I-D.hartke-core-pending]. All other HTTP 2xx response codes are not used by EST. For the following HTTP 4xx error codes that may occur: 400, 401, 403, 404, 405, 406, 412, 413, 415; the equivalent CoAP response code for EST-coaps is 4.xx. For the HTTP 5xx error codes: 500, 501, 502, 503, 504 the equivalent CoAP response code is 5.xx.

4.4. Message fragmentation

DTLS defines fragmentation only for the handshake part and not for secure data exchange (DTLS records). [RFC6347] states "Each DTLS record MUST fit within a single datagram". To avoid using IP fragmentation, which is not supported by 6LoWPAN, invokers of the DTLS record layer MUST size DTLS records so that they fit within any Path MTU estimates obtained from the record layer. In addition, invokers residing on a 6LoWPAN over IEEE 802.15.4 network SHOULD attempt to size CoAP messages such that each DTLS record will fit within one or two IEEE 802.15.4 frames.

That is not always possible. Even though ECC certificates are small in size, they can vary greatly based on signature algorithms, key sizes, and OID fields used. For 256-bit curves, common ECDSA cert sizes are 500-1000 bytes which could fluctuate further based on the algorithms, OIDs, SANs and cert fields. For 384-bit curves, ECDSA certs increase in size and can sometimes reach 1.5KB. Additionally, there are times when the EST cacerts response from the server can include multiple certs that amount to large payloads. Section 4.6 of CoAP [RFC7252] describes the possible payload sizes: "if nothing is known about the size of the headers, good upper bounds are 1152 bytes for the message size and 1024 bytes for the payload size".

Section 4.6 of [RFC7252] also suggests that IPv4 implementations may want to limit themselves to more conservative IPv4 datagram sizes such as 576 bytes. From [RFC0791] follows that the absolute minimum value of the IP MTU for IPv4 is as low as 68 bytes, which would leave only 40 bytes minus security overhead for a UDP payload. Thus, even with ECC certs, EST-coaps messages can still exceed sizes in MTU of
1280 for IPv6 or 60-80 bytes for 6LoWPAN [RFC4919] as explained in section 2 of [RFC7959]. EST-coaps needs to be able to fragment EST messages into multiple DTLS datagrams. Fine-grained fragmentation of EST messages is essential.

To perform fragmentation in CoAP, [RFC7959] specifies the "Block1" option for fragmentation of the request payload and the "Block2" option for fragmentation of the return payload of a CoAP flow.

The BLOCK draft defines SZX in the Block1 and Block2 option fields. These are used to convey the size of the blocks in the requests or responses.

The CoAP client MAY specify the Block1 size and MAY also specify the Block2 size. The CoAP server MAY specify the Block2 size, but not the Block1 size. As explained in Section 1 of [RFC7959]), blockwise transfers SHOULD be used in Confirmable CoAP messages to avoid the exacerbation of lost blocks.

The Size1 response MAY be parsed by the client as a size indication of the Block2 resource in the server response or by the server as a request for a size estimate by the client. Similarly, Size2 option defined in BLOCK should be parsed by the server as an indication of the size of the resource carried in Block1 options and by the client as a maximum size expected in the 4.13 (Request Entity Too Large) response to a request.

Examples of fragmented messages are shown in Appendix C.

5. Discovery and URI

EST-coaps is targeted to low-resource networks with small packets. Saving header space is important and an additional EST-coaps URI is specified that is shorter than the EST URI.

In the context of CoAP, the presence and location of (path to) the management data are discovered by sending a GET request to "/.well-known/core" including a resource type (RT) parameter with the value "ace.est" [RFC6690]. Upon success, the return payload will contain the root resource of the EST resources. It is up to the implementation to choose its root resource; throughout this document the example root resource /est is used. The example below shows the discovery of the presence and location of management data.
REQ: GET /.well-known/core?rt=ace.est

RES: 2.05 Content
</est>; rt="ace.est"

The additional EST-coaps server URIs differ from the EST URI by replacing the scheme https by coaps and by specifying a shorter resource path names:

coaps://www.example.com/est/short-name

The CoAP short URI exists next to the URI defined in [RFC7030].

coaps://www.example.com/.well-known/est/est-name
OR
coaps://www.example.com/.well-known/est/ArbitraryLabel/est-name

Figure 5 in section 3.2.2 of [RFC7030] enumerates the operations and corresponding paths which are supported by EST. Table 1 provides the mapping from the EST URI path to the shorter EST-coaps URI path.

<table>
<thead>
<tr>
<th>EST</th>
<th>EST-coaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>/cacerts</td>
<td>/crts</td>
</tr>
<tr>
<td>/simpleenroll</td>
<td>/sen</td>
</tr>
<tr>
<td>/simplereenroll</td>
<td>/sren</td>
</tr>
<tr>
<td>/csrattrs</td>
<td>/att</td>
</tr>
<tr>
<td>/serverkeygen</td>
<td>/skg</td>
</tr>
</tbody>
</table>

Table 1

When discovering the root path for the EST resources, the server MAY return the full resource paths and the used content types. This is useful when multiple content types are specified for EST-coaps server. For example, the following more complete response is possible.
REQ: GET /.well-known/core?rt=ace.est

RES: 2.05 Content
</est>; rt="ace.est"
</est/crts>; rt="ace.est"; ct=TBD1
</est/sen>; rt="ace.est"; ct=TBD1 TBD4
</est/sren>; rt="ace.est"; ct=TBD1 TBD4
</est/att>; rt="ace.est"; ct=TBD4
</est/skg>; rt="ace.est"; ct=TBD1 TBD4 TBD2

The return of the content-types allows the client to choose the most appropriate one from multiple content types.

6. DTLS Transport Protocol

EST-coaps depends on a secure transport mechanism over UDP that can secure (confidentiality, authenticity) the CoAP messages exchanged.

DTLS is one such secure protocol. When "TLS" is referred to in the context of EST, it is understood that in EST-coaps, security is provided using DTLS instead. No other changes are necessary (all provisional modes etc. are the same as for TLS).

CoAP was designed to avoid fragmentation. DTLS is used to secure CoAP messages. However, fragmentation is still possible at the DTLS layer during the DTLS handshake when using ECC ciphersuites. If fragmentation is necessary, "DTLS provides a mechanism for fragmenting a handshake message over a number of records, each of which can be transmitted separately, thus avoiding IP fragmentation" [RFC6347].

CoAP and DTLS can provide proof of identity for EST-coaps clients and server with simple PKI messages conformant to section 3.1 of [RFC5272]. EST-coaps supports the certificate types and Trust Anchors (TA) that are specified for EST in section 3 of [RFC7030].

Channel-binding information for linking proof-of-identity with connection-based proof-of-possession is optional for EST-coaps. When proof-of-possession is desired, a set of actions are required regarding the use of tls-unique, described in section 3.5 in [RFC7030]. The tls-unique information translates to the contents of the first "Finished" message in the TLS handshake between server and client [RFC5929]. The client is then supposed to add this "Finished" message as a ChallengePassword in the attributes section of the PKCS#10 Request Info to prove that the client is indeed in control of the private key at the time of the TLS session when performing a /simpleenroll, for example. In the case of EST-coaps, the same...
operations can be performed during the DTLS handshake. In the event of handshake message fragmentation, the Hash of the handshake messages used in the MAC calculation of the Finished message

\[
\text{PRF(master_secret, finished_label, Hash(handshake_messages)) [0..verify_data_length-1];}
\]

MUST be computed as if each handshake message had been sent as a single fragment [RFC6347].

In a constrained CoAP environment, endpoints can’t afford to establish a DTLS connection for every EST transaction. Authenticating and negotiating DTLS keys requires resources on low-end endpoints and consumes valuable bandwidth. The DTLS connection SHOULD remain open for persistent EST connections. For example, an EST cacerts request that is followed by a simpleenroll request can use the same authenticated DTLS connection. Given that after a successful enrollment, it is more likely that a new EST transaction will take place after a significant amount of time, the DTLS connections SHOULD only be kept alive for EST messages that are relatively close to each other.

7. Proxying

In real-world deployments, the EST server will not always reside within the COAP boundary. The EST-server can exist outside the constrained network in a non-constrained network that supports TLS/HTTP. In such environments EST-coaps is used by the client within the COAP boundary and TLS is used to transport the EST messages outside the CoAP boundary. A proxy entity at the edge is required to operate between the COAP environment and the external HTTP network. The ESTcoaps-to-HTTPS proxy SHOULD terminate EST-coaps downstream and initiate EST connections over TLS upstream.

One possible use-case, shown in one figure below, is expected to be deployed in practice:

- A proxy between any EST-client and EST-server
Table 1 contains the URI mapping between the EST-coaps and EST the proxy SHOULD adhere to. Section 7 of [RFC8075] and Section 4.3 define the mapping between EST-coaps and HTTP response codes, that determines how a proxy translates COAP response codes from/to HTTP status codes. The mapping from Content-Type to media type is defined in Section 9. The conversion from binary to BSD64 needs to be done in the proxy. Conversion is possible because a TLS link exists between EST-coaps-to-HTTP proxy and EST server and a corresponding DTLS linked exists between EST-coaps-to-HTTP proxy and EST client.

Due to fragmentation of large messages into blocks, an EST-coaps-to-HTTP proxy SHOULD reassemble the BLOCKs before translating the binary content to BSD64, and consecutively relay the message upstream into the HTTP environment.

For the discovery of the EST server by the EST client in the coap environment, the EST-coaps-to-HTTP proxy MUST announce itself according to the rules of Section 5. The available functions of the proxies MUST be announced with as many resource paths. The discovery of EST server in the http environment follow the rules specified in [RFC7030].

[EDNOTE: PoP will be addressed here.]

A proxy SHOULD authenticate the client downstream and it should be authenticated by the EST server or CA upstream. The Registration Authority (RA) is necessary to (re-)create the secure connection from DTLS to TLS and vice versa. A trust relationship needs to be pre-established between the proxy and the EST servers to be able to proxy these connections on behalf of various clients.

[EDNOTE: To add more details about trust relations in this section.]
8. Parameters

[EDNOTE: This section to be populated. It will address transmission parameters described in sections 4.7 and 4.8 of the CoAP draft. EST does not impose any unique parameters that affect the CoAP parameters in Table 2 and 3 in the CoAP draft but the ones in CoAP could be affecting EST. For example, the processing delay of CAs could be less then 2s, but in this case they should send a CoAP ACK every 2s while processing.]

9. IANA Considerations

9.1. Content-Format registry

Additions to the sub-registry "CoAP Content-Formats", within the "CoRE Parameters" registry are needed for the below media types. These can be registered either in the Expert Review range (0-255) or IETF Review range (256-9999).

1. *
   * application/pkcs7-mime
   * Type name: application
   * Subtype name: pkcs7-mime
   * ID: TBD1
   * Required parameters: None
   * Optional parameters: None
   * Encoding considerations: binary
   * Security considerations: As defined in this specification
   * Published specification: [RFC5751]
   * Applications that use this media type: EST

2. *
   * application/pkcs8
   * Type name: application
   * Subtype name: pkcs8
3.

* application/csrattrs
* Type name: application
* Subtype name: csrattrs
* ID: TBD3

* Required parameters: None
* Optional parameters: None
* Encoding considerations: binary
* Security considerations: As defined in this specification
* Published specification: [RFC7030]

* Applications that use this media type: EST

4.

* application/pkcs10
* Type name: application
* Subtype name: pkcs10
* ID: TBD4

* Required parameters: None
9.2. Resource Type registry

Additions to the sub-registry "CoAP Resource Type", within the "CoRE Parameters" registry are needed for a new resource type.

- rt="ace.est" needs registration with IANA.

10. Security Considerations

10.1. proxy considerations

The proxy proposed in Section 7 must be deployed with great care, and only when the recommended connections are impossible.

[EDNOTE: To add more details about trust relations through proxies in this section.]

10.2. EST server considerations

The security considerations of section 6 of [RFC7030] are only partially valid for the purposes of this document. As HTTP Basic Authentication is not supported, the considerations expressed for using passwords do not apply.

Given that the client has only limited resources and may not be able to generate sufficiently random keys to encrypt its identity, it is possible that the client uses server generated private/public keys to encrypt its certificate. The transport of these keys is inherently risky. A full probability analysis MUST be done to establish whether server side key generation enhances or decreases the probability of identity stealing.

When a client uses the Implicit TA database for certificate validation, the client cannot verify that the implicit data base can act as an RA. It is RECOMMENDED that such clients include "Linking Identity and POP Information" Section 6 in requests (to prevent such requests from being forwarded to a real EST server by a man in the middle). It is RECOMMENDED that the Implicit Trust Anchor database
used for EST server authentication be carefully managed to reduce the chance of a third-party CA with poor certification practices from being trusted. Disabling the Implicit Trust Anchor database after successfully receiving the Distribution of CA certificates response (Section 4.1.3 of [RFC7030]) limits any vulnerability to the first DTLS exchange.

In accordance with [RFC7030], TLS cipher suites that include "_EXPORT_" and "_DES_" in their names MUST NOT be used. More information about recommendations of TLS and DTLS are included in [RFC7525].

As described in CMC, Section 6.7 of [RFC5272], "For keys that can be used as signature keys, signing the certification request with the private key serves as a POP on that key pair". The inclusion of tls-unique in the certification request links the proof-of-possession to the TLS proof-of-identity. This implies but does not prove that the authenticated client currently has access to the private key.

Regarding the CSR attributes that the CA may list for inclusion in an enrollment request, an adversary could exclude attributes that a server may want, include attributes that a server may not want, and render meaningless other attributes that a server may want. The CA is expected to be able to enforce policies to recover from improper CSR requests.

Interpreters of ASN.1 structures should be aware of the use of invalid ASN.1 length fields and should take appropriate measures to guard against buffer overflows, stack overruns in particular, and malicious content in general.

11. Acknowledgements

The authors are very grateful to Klaus Hartke for his detailed explanations on the use of Block with DTLS. The authors would like to thank Esko Dijk and Michael Verschoor for the valuable discussions that helped in shaping the solution. They would also like to thank Peter Panburana from Cisco for his feedback on technical details of the solution. Constructive comments were received from Eliot Lear, Jim Schaad, Hannes Tschofenig, and Julien Vermillard.

12. Change Log

-03:

removed all motivation to and dependence on BRKI

Supports full EST, except password support
discovery limited to EST functions
./well-known/est is alternative path to short coap path
proxy discussion is simplified to one case

-02:
binary instead of CBOR binary in mime types.
supported content types are discoverable.
DTLS POP text improved.
First version of Security considerations section written.
First version of Proxying section written.
Various text improvements.

-01:
Merging of draft-vanderstok-ace-coap-est-00 and draft-pritikin-coap-bootstrap-01
URI and discovery are modified
More text about 6tisch bootstrap including EDHOC and OSCOAP
mapping to DICE IoT profiles
adapted to BRISKI progress

13. References

13.1. Normative References

[I-D.hartke-core-pending]
Stok, P. and K. Hartke, "The 'Pending' Response Code for
the Constrained Application Protocol (CoAP)", draft-
hartke-core-pending-01 (work in progress), August 2017.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,


13.2. Informative References

[I-D.ietf-6tisch-minimal-security]

[I-D.ietf-anima-bootstrapping-keyinfra]

[ieee802.15.4]
Institute of Electrical and Electronics Engineers, "IEEE Standard 802.15.4-2006", 2006.


Appendix A. EST messages to EST-coaps

This section takes all examples from Appendix A of [RFC7030], changes the payload from Base64 to binary and replaces the http headers by their CoAP equivalents.

The corresponding CoAP headers are only shown in Appendix A.1. Creating CoAP headers are assumed to be generally known.

[EDNOTE: The payloads of the examples need to be re-generated with appropriate tools and example certificates.]
A.1.  cacerts

In EST-coaps, a coaps cacerts IPv4 message can be:

GET coaps://[192.0.2.1:8085]/est/crts

The corresponding CoAP header fields are shown below. The use of
block and DTLS are worked out in Appendix C.

Ver = 1
T = 0 (CON)
Code = 0x01 (0.01 is GET)
Options
  Option1 (Uri-Host)
    Option Delta = 0x3  (option nr = 3)
    Option Length = 0x9
    Option Value = 192.0.2.1
  Option2 (Uri-Port)
    Option Delta = 0x4  (option nr = 4+3=7)
    Option Length = 0x4
    Option Value = 8085
  Option3 (Uri-Path)
    Option Delta = 0x4   (option nr = 7+4= 11)
    Option Length = 0x9
    Option Value = /est/crts
Payload = [Empty]

A 2.05 Content response with a cert in EST-coaps will then be:

2.05 Content (Content-Format: application/pkcs7-mime)
   {payload}

with CoAP fields

Ver = 1
T = 2 (ACK)
Code = 0x45 (2.05 Content)
Options
  Option1 (Content-Format)
    Option Delta = 0xC  (option nr = 12)
    Option Length = 0x2
    Option Value = TBD1 (defined in this document)

Payload = 3023906092a6206734107028c2a3023260201013100300b06092a6206734107018
c0c3020bb302063c2010202020900a61e75193b7acc0d06092a62067341010500301b3119301706355040313106573744578616d706c654341204f774f301e170d3133035039333333333315a170d31343035303930333333333315a301b3119301706
A.2. csrattrs

In the following valid /csrattrs exchange, the EST-coaps client authenticates itself with a certificate issued by the connected CA.

The initial DTLS handshake is identical to the enrollment example. The IPv6 CoAP GET request looks like:

REQ:
GET coaps://[2001:db8::2:1]:61616/est/att

A 2.05 Content response contains attributes which are relevant for the authenticated client. In this example, the EST-coaps server returns two attributes that the client can ignore when they are unknown to him:

A.3. enroll / reenroll

[EDNOTE: We might need a new Option for the Retry-After response message. We might need a new Option for the WWW-Authenticate response.]

During the Enroll/Reenroll exchange, the EST-coaps client uses a CSR (PKCS#10) request in the POST request payload.
After verification of the certificate by the server, a 2.05 Content response with the issued certificate will be returned.

POST [2001:db8::2:1]:61616/est/sen
(Content-Format: application/pkcs10)

30208530206d020100301f311d301b0603550403131464656d6f737465703420313336383134313335323020620673410101050003204f0030204a022041005d9f4dffd3cf599f20646a958436777856095b355c35b8e34726dd37645231347975b4c09b9c6d75d408311307a81f7ade7f5d241f7d5be85620c5d4438bb421cf21672f2ccfc363c64ea2618a62f50365767369d630e6a968772247d862af079f9ac7af6f694cfdaf5b84c42087dc062d462190c525813f210a036a737b4f30d8891f4b7559f7b27252453146332d51c937557716cace62f452512c3a4447ad311502048133e0100018701f06092a6206734109731123102b72724369722f372b45597535034300d06092a6206734101050003204100441b0177a3a65501487735a8ad5d3827a4eaab876013920e2afcd87aa81733c7c0335b9e7e1bf47a7cd5176e7cc6be22ae03498588d5f2e3b143f2b1a6175ec544e8e7625af6b836fd44168942e55a9c6606f69075d6d3475410729aa6d80a0fbb984caf7b84b5b3e4545f1f901785ada007060cad6db26a592d4a7bda7d586b681109621707110340755155cddc75481e27b2b5d53a8593f7e25100a6f7605085dab4fc7e0731f0e7fe305073971362d5157e92e6bc2e3edbcadb40

RET:
2.05 Content (Content-Format: application/pkcs7-mime)
3020f806092a62067341070283293020e50201013100300b06092a62067341070a1830b3020c730206300d06092a6206734101050003204100441b0177a3a65501487735a8ad5d3827a4eaab876013920e2afcd87aa81733c7c0335b9e7e1bf47a7cd5176e7cc6be22ae03498588d5f2e3b143f2b1a6175ec544e8e7625af6b836fd44168942e55a9c6606f69075d6d3475410729aa6d80a0fbb984caf7b84b5b3e4545f1f901785ada007060cad6db26a592d4a7bda7d586b681109621707110340755155cddc75481e27b2b5d53a8593f7e25100a6f7605085dab4fc7e0731f0e7fe305073971362d5157e92e6bc2e3edbcadb40

[EDNOTE: If POP is used, make sure tls-unique in the CSR is a valid HMAC output. ]

A.4. serverkeygen

During this valid /serverkeygen exchange, the EST-coaps client authenticates itself using the certificate provided by the connected CA.

[EDNOTE: the client incudes a CSR with a public key that the server should ignore, so we need a content-format here. ]

[EDNote: If POP is used, make sure tls-unique in the CSR is a valid HMAC output. ]

The initial DTLS handshake is identical to the enrollment example. The CoAP GET request looks like:

POST coaps://[192.0.2.1:8085]/est/skg
302081302069020100305b313e303c0603550403313357365727665724b657947656e2072657120627920636c69656e74696e206465666f6f73657373696f6e20616e646c6572203133363831343139353119301706035040513105049443a5769646765742043
3ae3a130302062300d06092a6206734101050003204f0030204a22040100446af03f7f2766b23776c333cd20f9d1a7a6ee36d01499b6f075d1e38a57e98
e9c197f5175228454b7f169523e26e52e4a974c6a2e34edf80b33f5f473db
chf76116bb0e4d3e04a965128a84f763e1c1862a256e04657f7c271cf1f04e47
31fad53c22b21a1e513b69fa0d187314ac39445949a488059329390e78c7659621
63d61327a53f4eaa7721d2b1343c7362b37da502717ccf2475653c7a3860c5f4
0612a5dbd63379d4755264eb6327e3a3263b14962858b585e57e2f6b2377519b0
2030100018701f06092a6206734109073112131064667341586d4a6e6a6f6b427
44476762300d06092a6206734101050500032044010472d11007e5a2b2c2023d47a
6d71046c307701d8eb9e47272713378390b4ee321462a3db54579f5a514f6f
4050a497f428189b63655d03a194ef729f101743e5d03fbc6ae1e84846813d0a
f9288724381909188c851fa9a5059802eb64449f2a3c9e441353d136768da27ff
4f277651d6766a7e51931bobf56135a2230891fd184960e1313e7ala9139ed19
28196867079a456cd2266c7b754a45151b71b939e381be333efa61580fe5d25bf
4823d2b2d6a98445b46305c10637e20856611

RET:

2.05 Content (Content-Format: application/pkcs8)
30213e020100300d06092a6206734101050004212830212402010022041003
c0bc2748f2003e3e8eaa1f5f746f2a71e83f5854129c6ff8e646de0e056153274
d0c195dd9cff3112aa41774ab655c3d56359c3b3df05524962ed84e7e30a1
1bfb14e7e0693d901702b4cd39e6d40325356152b213c8b535851e681a7074c
0c6d2b60e7c32fc0336b28e743ebad4e5921074d47195d3c05e43c527526e92d5
45e562578d2d4b5f219bffe9d3eef0227264a26764371a1f99257216647df6704
efec5adb54dab24231844eb59587579500e637d6682310a146b7d3e1083010
01022041004eb3f78b7791d6377f23117c17844531c81111fbbf800028216263
915565bc7c3f3f643b537a2ce69140a31c22550fa97e5132c61b74166b8626704
260620333050f510096e6770f5880e7ec15dc0ca6ce2b5f187e2325d1a4ab705
ad00471f73bf2f779127bf5c535e0ce6a343b502722f2f397a26126e0af606b5aa7
Without the DecryptKeyIdentifier attribute, the response has no additional encryption beyond DTLS. [EDNOTE: Add comment about deriving symmetric keys by using the TLS KEM draft.]

The response contains first a preamble that can be ignored. The EST-coaps server can use the preamble to include additional explanations, like ownership or support information.
Appendix B. Encoding for server side key generation

Server side key generation for coap can be implemented efficiently using multipart encoding

[EDNOTE: text to be written.]

Appendix C. EST-coaps Block message examples

This section provides a detailed example of the messages using DTLS and BLOCK option Block2. The minimum PMTU is 1280 bytes, which is the example value assumed for the DTLS datagram size. The example block length is taken as 64 which gives an SZX value of 2.

The following is an example of a valid /cacerts exchange over DTLS. The content length of the cacerts response in appendix A.1 of [RFC7030] is 4246 bytes using base64. This leads to a length of 2509 bytes in binary. The CoAP message adds around 10 bytes, the DTLS record 29 bytes. To avoid IP fragmentation, the CoAP block option is used and an MTU of 127 is assumed to stay within one IEEE 802.15.4 packet. To stay below the MTU of 127, the payload is split in 39 packets with a payload of 64 bytes each, followed by a packet of 13 bytes. The client sends an IPv6 packet containing the UDP datagram with the DTLS record that encapsulates the CoAP Request 40 times. The server returns an IPv6 packet containing the UDP datagram with the DTLS record that encapsulates the CoAP response. The CoAP request-response exchange with block option is shown below. Block option is shown in a decomposed way indicating the kind of Block option (2 in this case because used in the response) followed by a colon, and then the block number (NUM), the more bit (M = 0 means last block), and block size exponent (2**(SZX+4)) separated by slashes. The Length 64 is used with SZX= 2 to avoid IP fragmentation. The CoAP Request is sent with confirmable (CON) option and the content format of the Response is /application/cacerts.

GET [192.0.2.1:8085]/est/crts -->
   <-- (2:0/1/39) 2.05 Content
GET URI (2:1/1/39) -->
   <-- (2:1/1/39) 2.05 Content
   |
GET URI (2:65/1/39) -->
   <-- (2:65/0/39) 2.05 Content

For further detailing the CoAP headers of the first two blocks are written out.
The header of the first GET looks like:

Ver = 1
T = 0 (CON)
Code = 0x01 (0.1 GET)
Options
  Option1 (Uri-Host)
    Option Delta = 0x3  (option nr = 3)
    Option Length = 0x9
    Option Value = 192.0.2.1
  Option2 (Uri-Port)
    Option Delta = 0x4  (option nr = 3+4=7)
    Option Length = 0x4
    Option Value = 8085
  Option3 (Uri-Path)
    Option Delta = 0x4    (option nr = 7+4=11)
    Option Length = 0x9
    Option Value = /est/crts
Payload = [Empty]

The header of the first response looks like:

Ver = 1
T = 2 (ACK)
Code = 0x45 (2.05 Content.)
Options
  Option1 (Content-Format)
    Option Delta = 0xC  (option 12)
    Option Length = 0x2
    Option Value = TBD1
  Option2 (Block2)
    Option Delta = 0xB  (option 23 = 12 + 11)
    Option Length = 0x1
    Option Value = 0x0A (block number = 0, M=1, SZX=2)
Payload =
30233906092a6206734107028c2a3023260201013100300b06092a6206734107018
c0c3020bb302063c20102020900a61e75193b7acc0d06092a6206734101

The second Block2:
Ver = 1
T = 2 (means ACK)
Code = 0x45 (2.05 Content.)
Options
  Option 1 (Content-Format)
    Option Delta = 0xC   (option 12)
    Option Length = 0x2
    Option Value = TBD1
  Option 2 (Block2)
    Option Delta = 0xB  (option 23 = 12 + 11)
    Option Length = 0x1
    Option Value = 0x1A (block number = 1, M=1, SZX=2)
Payload =
05050030
1b31193017060355504031310657374578616d706cc654341204f774f301e170d313
3303530393033353333315a170d3134303530393033353333315a

The 40th and final Block2:
Ver = 1
T = 2 (means ACK)
Code = 0x21
Options
  Option 1 (Content-Format)
    Option Delta = 0xC   (option 12)
    Option Length = 0x2
    Option Value = TBD1
  Option 2 (Block2)
    Option Delta = 0xB  (option 23 = 12 + 11)
    Option Length = 0x2
    Option Value = 0x272 (block number = 39, M=0, SZX=2)
Payload = 73a30d0c006343116f58403100

Authors’ Addresses

Peter van der Stok
Consultant

Email: consultancy@vanderstok.org

Panos Kampanakis
Cisco Systems

Email: pkampana@cisco.com
Sandeep S. Kumar
Philips Lighting Research
High Tech Campus 7
Eindhoven 5656 AE
NL

Email: ietf@sandeep.de

Michael C. Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/

Martin Furuhed
Nexus Group

Email: martin.furuhed@nexusgroup.com

Shahid Raza
RISE SICS
Isafjordsgatan 22
Kista, Stockholm 16440
SE

Email: shahid@sics.se