ACE Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 14, 2017

G. Selander
Ericsson AB
S. Raza
RISE SICS
M. Vucinic
Inria
M. Furuhed
Nexus
M. Richardson
Sandelman Software Works
March 13, 2017

Enrollment with Application Layer Security
draft-selander-ace-eals-00

Abstract

This document specifies public key certificate enrollment procedures authenticated with application-layer security protocols suitable for Internet of Things deployments. The protocols leverage existing IoT standards including Constrained Application Protocol (CoAP), Concise Binary Object Representation (CBOR) and the CBOR Object Signing and Encryption (COSE) format.

Table of Contents

1. Introduction

Asymmetric cryptography with Public Key Infrastructure (PKI) is a de-facto key exchange and mutual authentication solution in the Internet. Though solutions based on PSK are still state-of-the-art in sensor networks they are not scalable to Internet-connected billions of things. Therefore, most IoT security standards support asymmetric cryptographic protocols. The greatest challenge with asymmetric cryptography and PKI is enrollment, the process of certifying keys. Enrollment is even more challenging in the IoT as things are resource-constrained and traditional enrollment techniques are not compatible with recent IoT security protocols. Without secure enrollment, PKI will not be trustworthy and in turn the

cybersecurity of the entire system will be at stake even though the underlying cryptographic cipher suites are most secure.

Security at the application layer provides an attractive option for protecting Internet of Things (IoT) deployments, in particular in constrained environments [RFC7228] and when using CoAP [RFC7252]; for example where transport layer security is not sufficient [I-D.hartke-core-e2e-security-reqs], or where it is beneficial that the security protocol is independent of lower layers, such as when securing CoAP over mixed transport protocols.

Application layer security protocols suitable for constrained devices are in development, including the secure communication protocol OSCOAP [I-D.ietf-core-object-security]. OSCOAP defines an extension to the Constrained Application Protocol (CoAP) providing encryption, integrity and replay protection end-to-end between CoAP client and server based on a shared secret. The shared secret can be established in different ways e.g. using a trusted third party such as in ACE [I-D.ietf-ace-oauth-authz], or using a key exchange protocol such as EDHOC [I-D.selander-ace-cose-ecdhe]. OSCOAP and EDHOC can leverage other constrained device primitives developed in the IETF: CoAP, CBOR [RFC7049] and COSE [I-D.ietf-cose-msg], and makes only a small additional implementation footprint.

Lately, there has been a discussion in several IETF working groups about certificate enrollment protocols suitable for IoT devices, to support the use case of an IoT device joining a new network domain and establishing credentials valid in this domain. This document describes Enrollment with Application Layer Security (EALS), a certificate enrollment protocol based on CMC [RFC5272] and using OSCOAP as a secure channel. This document also describes how ACE and EDHOC can be used for establishing an authenticated and authorized channel.

This work is inspired by the Enrollment over Secure Transport (EST) protocol [RFC7030], which also is based on CMC, but EALS is secured on application layer instead of on transport layer.

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]. These words may also appear in this document in lowercase, absent their normative meanings.

2. CMC protocol

2.1. Simple Enrollment

This section describes the simple enrollment protocol, which is an embedding of the Simple PKI Request/Response protocol of CMC [RFC5272] in Object Secure CoAP (OSCOAP) [I-D.ietf-core-object-security].

The simple enrollment protocol is a 2-pass protocol between an EALS client (e.g. an IoT device) and an EALS server (a Certification Authority (CA)), see Figure 1. The protocol assumes that both EALS client and EALS server implement CoAP and the Object-Security option of CoAP (OSCOAP).

OSCOAP assumes the existence of a shared secret between an EALS client and server. The shared secret may be obtained by running a key agreement algorithm or by an aid of a trusted third party. Mutual authentication and authorization occurs during this key agreement stage. The simple enrollment protocol may also be run directly with a pre-shared key. In that case, authentication and authorization of EALS client and server is implicit to the shared key protecting the /eals resource.

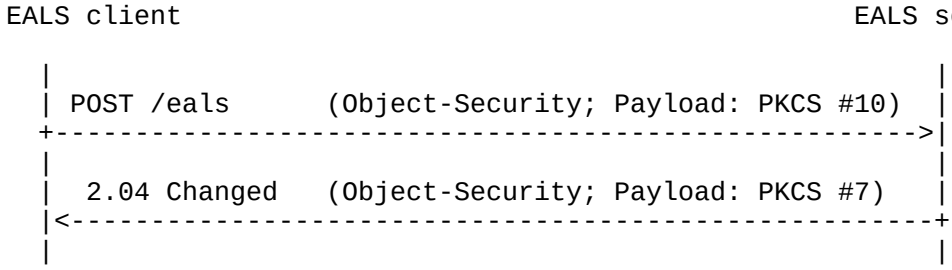

Figure 1: The Simple Enrollment Protocol.

The simple enrollment protocol consists of a CoAP message exchange.

The EALS client sends a CoAP request:

- Method is POST
- Uri-Path is "eals"
- Object-Security option is present

- Payload is the CMC Simple PKI Request [RFC5272] (i.e. a PKCS #10 certification request).

If successful, the EALS server sends a CoAP response:

- Code is 2.04 (Changed)
- Content-Format is "application/pkcs7-mime" (TBD)
- Object-Security option is present
- Payload is a certs-only CMC Simple PKI Response [RFC5272] (i.e the issued certificate)

OSCOAP protects the CoAP message exchange between the endpoints over any transport and via intermediary nodes. The OSCOAP protection requires that a security context is established between client and server. The security context can be derived from a set of Input Parameters (Section 3.3 of [I-D.ietf-core-object-security]), including at least the following:

- Master Secret
- Sender ID
- Recipient ID

where the Master Secret is a uniformly random byte string, and the Sender ID and Recipient ID are byte strings identifying the endpoints. In Section 3 we give examples of how to the OSCOAP input parameters can be established.

The server MUST verify that the Master Secret is associated to the Distinguished Name for which the client is requesting a certificate.

Note 1: The encodings and formats used by CMC may later be updated with other equivalents more adapted to constrained environments.

2.2. Re-enrollment

Re-enrollment and re-keying of clients occurs using the same exchange as during the simple enrollment protocol. Re-enrollment request follows the same format as during the simple enrollment. In case of success, re-enrollment response contains certs-only CMC Simple PKI Response, as in the case of simple enrollment with content-format set to "application/pkcs7-mime".

TBD. Requirements on parsing PKCS messages and X.509 certificates

TBD. Error handling with CoAP error codes

TBD. Server-side key generation

2.3. Full Enrollment

It is straightforward to extend the simple enrollment to the CMC Full PKI Request/Response protocol.

In this case, to authorize the PKCS#10 request to the CA, it is enveloped in a CMC message and signed with a pre-installed device private key and certificate by the device itself.

The public key in the device certificate acts as a unique identifier of the device. By trusting the CA issuing the pre-installed certificate, the enrolment CA can acknowledge the signed request. The trusted factory CA will also ensure the origin of the device.

An alternative to authorize the PKCS#10 request to the CA, is to use a security gateway that can envelope the request in a CMC message using a certificate trusted by the CA.

The details are FFS.

2.4. Compiling Certificate Content

A CA have several means of compiling certificate content during issuance. The subject Distinguished Name (DN) information for the certificate may be based on the content of the request alone.

Alternatively, complementary data can be added to the request by the CA from an external source trusted by the CA, or taken from records of pre-registered information on end-entities that is stored in the CA system and which can be uniquely matched to the data in the request. Due to the constrained device capabilities the amount of subject DN data in a request may be very limited. The method of adding complementary data for the certificate can be a choice of the CA, assuming the source of complementary data can be provided in a trustworthy way.

With the option to add complementary data to a certificate request, the end-entity provided data can be diminished by e.g. submitting only the public key in the PKCS#10 content. The public key can be used to match the device to pre-registered data or for retrieval of subject data from other sources.

3. Establishment of OSCOAP Input Parameters

In this section we present two application layer protocols for establishing OSCOAP input parameters (Section 3.3 of [I-D.ietf-core-object-security]), in particular the OSCOAP master secret.

3.1. EDHOC

EDHOC [I-D.selander-ace-cose-ecdhe] is a key establishment protocol, corresponding to the handshake protocol of TLS, encoded with CBOR and using COSE that may be transported with e.g. CoAP. EDHOC provides mutual authentication of client and server and establishes a shared secret with forward secrecy which may be used as OSCOAP master secret in the CMC protocol (Section 2).

The asymmetric keys authenticated version of EDHOC is described in section 4 of [I-D.selander-ace-cose-ecdhe], a simplified version of the protocol is shown in Figure 2.

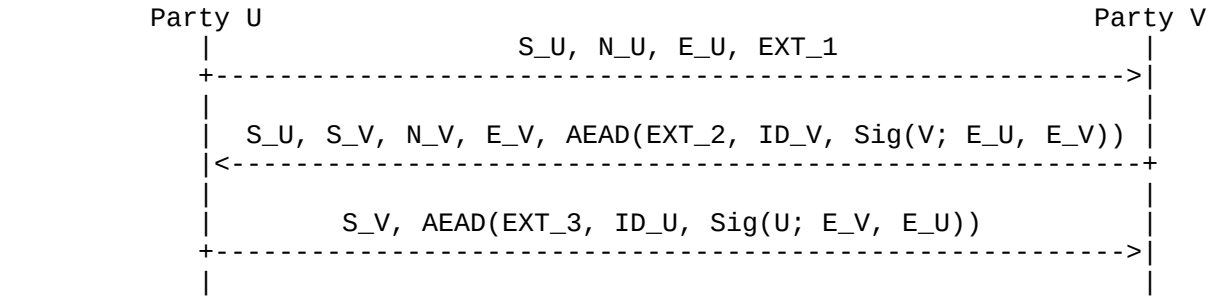

Figure 2: EDHOC with asymmetric key authentication (simplified). S = session identifer, N = nonce, E = ephemeral public key, ID = identifier, and EXT = application defined extension.

The session identifiers S_U and S_V may be used as OSCOAP input parameters Sender ID and Recipient ID of party U, and v.v. as described in Appendix B2 of [I-D.selander-ace-cose-ecdhe].

Figure 3 shows an example of using the EDHOC protocol to establish a mutually authenticated and authorized channel for the simple enrolment protocol. In this case the EALS server is EDHOC client (the mapping with interchanged roles is straightforward and left FFS). This setting has the following properties:

1. The EALS server initiates the EDHOC protocol. This allows the EALS server to orchestrate many concurrent enrollments, and control the associated network load.

2. The EALS client is authenticated first (EDHOC message_2). This allows the EALS server to authenticate the EALS client, and with this information to authorize the EALS client before completing the EDHOC protocol. The EALS server may in this case also relay authorizaton information about the EALS client, such as an ownership voucher, to the client in EDHOC extension EXT_3.

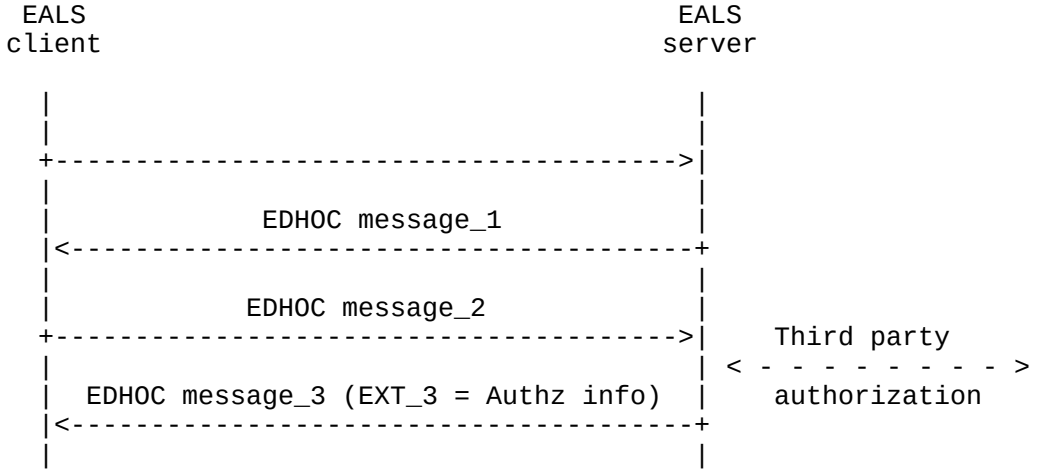

Figure 3: EALS extension of EDHOC.

Appendix B1 of [I-D.selander-ace-cose-ecdhe] shows how to embed EDHOC in a CoAP message exchange, a similar embedding can be applied here.

TBD Detail the protocol

3.2. ACE

The ACE protocol framework [I-D.ietf-ace-oauth-authz] is an adaptation of OAuth 2.0 to IoT deployments. ACE describes different message flows for a Client to get authorized access to a Resource Server (RS) by leveraging an Authorization Server (AS).

The Token Introspection flow (Section 7 of [I-D.ietf-ace-oauth-authz]) allows an RS to access authorization information relating to a client provided Access Token. If the access token is valid, the RS obtains information about the access rights and a symmetric key used by the client, and also a Client

Token containing the same shared key protected for the legitimate client (Section 7.4 of [I-D.ietf-ace-oauth-authz], Figure 4).

This message flow assumes that the Client and AS, as well as the RS and AS, has or can establish a mutually authenticated secure channel such that:

- The AS can encrypt the symmetric key for the Client in the Client Token, and the Client can verify the Client Token is issued by the AS;
- The RS and AS can exchange encrypted, integrity and replay protected introspection messages. In this case, the establishment of the secure channel can take place immediately before introspection, triggered by the RS receiveing the Access Token.

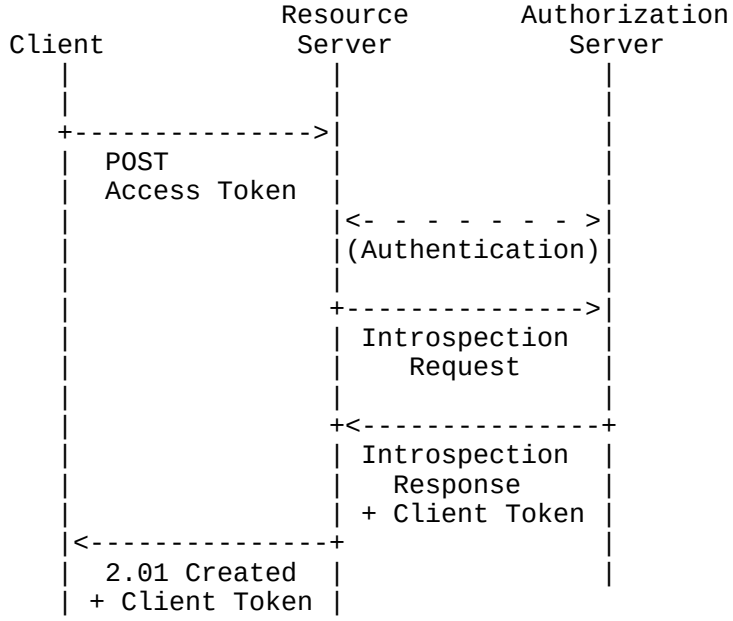

Figure 4: ACE Token Introspection with Client Token.

By mapping the EALS client and server to the ACE client and resource server, respectively, this application of ACE enables the authorization of EALS client and establishment of a shared key, which can be used as master secret with OSCOAP in the CMC protocol (Section 2). In this case, the access token contains access rights to /eals, but is not (yet) bound to a particular resource server.

The access token could be pre-provisioned to the client, e.g. during manufacture. Information about binding to resource server comes with the introspection response.

Section 2 of [I-D.seitz-ace-oscoap-profile] defines additional common header parameters for COSE_Key structure that are used to carry OSCOAP input parameters Sender and Recipient ID. The OSCOAP master secret is transported as part of the symmetric COSE_Key object. This document uses the same construct: COSE_Key object with OSCOAP input parameters present is transported as part of the Introspection Response and in the Client Token.

For the benefit of the client authorizing the enrollment, this document defines an additional common parameter for the Client Token called Voucher, extending the definition in Section 7.4 of [I-D.ietf-ace-oauth-authz]:

voucher
: OPTIONAL. Contains authorization information about the server, e.g. ownership voucher. The encoding is TBD.

Parameter name	CBOR Key	Major Type
voucher	TBD	2 (byte string)

Figure 5: CBOR mapping of parameters extending the client token.

Additionally, the certificate attributes presented by the Client in the enrolment request (Section 2) may be carried in the Client Token. The encoding is TBD.

4. Application to 6TiSCH

One candidate embedding of EALS into a bootstrapping architecture is as described in [I-D.ietf-6tisch-minimal-security]. The new device (a.k.a. Pledge) requests to be admitted into the network managed by the Join Registrar/Coordinator. The Pledge maps to an EALS/CoAP client, and the Join Registrar/Coordinator maps to an EALS/CoAP server.

When a pledge first joins a constrained network, it typically does not have IPv6 connectivity to reach the Join Registrar/Coordinator. For that reason, pledge communicates with the Join Proxy, a one hop neighbor of the pledge. Join Proxy statelessly relays the exchanges between the pledge and the Join Registrar/Coordinator.

As in the model of [I-D.ietf-6tisch-minimal-security], the Join Proxy plays the role of a CoAP proxy. Default CoAP proxy, however, keeps state information in order to relay the response back to the originating client, in this case the pledge. To mitigate Denial of Service attacks at the Join Proxy, [I-D.ietf-6tisch-minimal-security] mandates the use of a new CoAP option, Stateless-Proxy option, that allows the Join Proxy to operate without keeping per-client state.

The use of EDHOC as described in Section 3.1 enables mutual authentication and authorization of Pledge and Join Registrar/ Coordinator, and supports the use of the Stateless-Proxy option in order to provide the CoAP Proxy functionality described in this section.

5. Application to BRSKI

Another application of EALS is to the BRSKI [I-D.ietf-anima-bootstrapping-keyinfra] problem statement. BRSKI specifies an automated bootstrapping of a remote secure key infrastructure (BRSKI) using vendor installed X.509 certificate, in combination with a vendor authorized service on the Internet. BRSKI is referencing Enrolment over Secure Transport (EST) [RFC7030] to enable zero-touch joining of a device in a network domain. The same approach can be applied using EDHOC instead of EST, as is outlined in this document.

The audit/ownership vouchers specified in [I-D.ietf-anima-bootstrapping-keyinfra] are carried as part of EDHOC application-defined extensions, as described in Section 3.1. Nonces of the EDHOC protocol can be used for freshness also of the authorization step.

The limitations of applicability to energy-constrained devices due to credential size applies also to this document, and further work is needed to specify certificate formats relevant to constrained devices. Having said that, one rationale for this document is a more optimized message exchange, and potentially also code footprint, which is favorable in low-power deployments.

6. Security Considerations

7. Privacy Considerations

8. IANA Considerations

9. Acknowledgments

The authors wants to thank the participants of the 6tisch security design team for discussions and input contributing to this document.

10. References

10.1. Normative References

[I-D.ietf-ace-oauth-authz]
Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments (ACE)", draft-ietf-ace-oauth-authz-05 (work in progress), February 2017.

[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-01 (work in progress), December 2016.

[I-D.ietf-cose-msg]
Schaad, J., "CBOR Object Signing and Encryption (COSE)", draft-ietf-cose-msg-24 (work in progress), November 2016.

[I-D.selander-ace-cose-ecdhe]
Selander, G., Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", draft-selander-ace-cose-ecdhe-04 (work in progress), October 2016.

[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>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <http://www.rfc-editor.org/info/rfc7049>.

[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>.

10.2. Informative References

[I-D.hartke-core-e2e-security-reqs]
Selander, G., Palombini, F., and K. Hartke, "Requirements for CoAP End-To-End Security", draft-hartke-core-e2e-security-reqs-02 (work in progress), January 2017.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., and K. Pister, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-01 (work in progress), February 2017.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-04 (work in progress), October 2016.

[I-D.seitz-ace-oscoap-profile]
Seitz, L. and F. Palombini, "OSCOAP profile of ACE", draft-seitz-ace-oscoap-profile-01 (work in progress), October 2016.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, DOI 10.17487/RFC5272, June 2008, <http://www.rfc-editor.org/info/rfc5272>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <http://www.rfc-editor.org/info/rfc7030>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <http://www.rfc-editor.org/info/rfc7228>.

Appendix A. Examples

Authors' Addresses

Goeran Selander
Ericsson AB
Farogatan 6
Kista SE-16480 Stockholm
Sweden

Email: goran.selander@ericsson.com

Shahid Raza
RISE SICS
Isafjordsgatan 22
Kista SE-16440 Stockholm
Sweden

Email: shahid.raza@ri.se

Malisa Vucinic
Inria
2 Rue Simone Iff
Paris 75012
France

Email: malisa.vucinic@inria.fr

Martin Furuhed
Nexus
Telefonv. 26
Stockholm SE-12626
Sweden

Email: martin.furuhed@nexusgroup.com

Michael Richardson
Sandelman Software Works
470 Dawson Avenue
Ottawa, ON K1Z5V7
Canada

Email: mcr+ietf@sandelman.ca