

ACE Working Group
Internet-Draft
Intended status: Informational
Expires: January 5, 2015

S. Gerdes
Universitaet Bremen TZI
July 04, 2014

Actors in the ACE Architecture
draft-gerdes-ace-actors-01

Abstract

Constrained nodes are small devices which are limited in terms of processing power, memory, non-volatile storage and transmission capacity. Due to these constraints, commonly used security protocols are not easily applicable. Nevertheless, an authentication and authorization solution is needed to ensure the security of these devices.

Due to the limitations of the constrained nodes it is especially important to develop a light-weight security solution which is adjusted to the relevant security objectives of each participating party in this environment. Necessary security measures must be identified and applied where needed.

In this document, the required security related tasks are identified as guidance for the development of authentication and authorization solutions for constrained environments. Based on the tasks, an architecture is developed to represent the relationships between the logical functional entities involved.

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 January 5, 2015.

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	3
2. Problem Statement	4
3. Tasks	4
3.1. Basic Scenario Tasks	5
3.2. Authentication-Related Tasks	5
3.3. Authorization-Related Tasks	6
4. Actors	6
4.1. Constrained Level Actors	6
4.2. Principal Level Actors	7
4.3. Less-Constrained Level Actors	8
5. Protocol Requirements	10
5.1. Constrained Level Protocols	10
5.1.1. Cross Level Support Protocols	10
5.2. Less-Constrained Level Protocols	11
6. IANA Considerations	11
7. Security Considerations	11
8. Acknowledgments	11
9. References	11
9.1. Normative References	11
9.2. Informative References	11
Appendix A. List of Tasks	12
A.1. Basic Scenario	13
A.1.1. Processing Information	13
A.1.2. Sending Information	14
A.2. Security-Related Tasks	16
A.2.1. Information Authenticity	16
A.2.2. Authorization Validation	17
A.2.3. Transmission Security	19
A.2.4. Obtain Authorization information	19
A.2.5. Attribute Binding	20

A.2.6. Configuration of Authorization Information	22
Author's Address	23

1. Introduction

Constrained nodes are small devices with limited abilities which in many cases are made to fulfill a single simple task. They have limited system resources such as processing power, memory, non-volatile storage and transmission capacity and additionally in most cases do not have user interfaces and displays. Due to these constraints, commonly used security protocols are not always easily applicable.

Constrained nodes are expected to be integrated in all aspects of everyday life and thus will be trusted with a lot of personal data. Without appropriate security mechanisms attackers might gain control over things relevant to our lives. Authentication and authorization mechanisms are therefore prerequisites for a secure Internet of Things.

The Authentication and Authorization in Constrained Environments (ACE) Working Group aims at defining a solution for authenticated and authorized access to resources. To achieve this, it is necessary to develop a deep understanding of the problem to be solved. An essential part of this is to identify the tasks which must be performed to meet the security requirements in this scenario. Moreover, these tasks need to be assigned to logical functional entities which perform the tasks: the actors in the architecture. Thus, relations between the actors and requirements for protocols can be identified.

In this document, the required security related tasks are identified as guidance for the development of authentication and authorization solutions for constrained environments. Based on the tasks, an architecture is developed to represent the relationships between the logical functional entities involved.

1.1. Terminology

This document uses the following terminology:

Resource: an item of interest. It might contain sensor or actuator values or other information. The author had resources in the sense of RFC7231 [RFC7231] in mind, but for the considerations in this document the kind of representation of the item is not relevant.

Constrained node: a constrained device in the sense of [RFC7228].

Actor: A logical functional entity within a device that performs one or more tasks. Depending on the tasks, the device may need to have certain system resources available. Multiple actors may share, i.e. be present within, a device or even a piece of software.

Resource Server (RS): An entity which hosts a Resource.

Client (C): An entity which attempts to access a resource on a Resource Server.

Resource Owner (RO): The principal that owns the resource and controls its access permissions.

Client Owner (CO): The principal that owns the Client and controls permissions concerning authorized sources for R.

2. Problem Statement

The scenario the ACE Working Group addresses can be summarized as follows:

- o A Client (C) wants to access a Resource (R) on a Resource Server (RS).
- o A priori, C and RS do not necessarily know each other and have no security relationship.
- o C and / or RS are constrained.

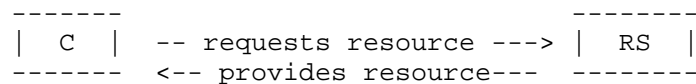


Figure 1: Basic Scenario

There are some security requirements for this scenario including one or more of:

- o Rq0.1: No unauthorized entity has access to (or otherwise gains knowledge of) R.
- o Rq0.2: When C attempts to access R, that access reaches the proper R.

3. Tasks

This section gives an overview of the tasks which must be performed in the given scenario (see Section 2) to meet the security requirements.

As described in the problem statement, either C or RS or both of them are constrained. Therefore tasks which must be conducted by either C or RS must be performable by constrained nodes.

3.1. Basic Scenario Tasks

This document does not assume a specific solution. We assume however, that at least the following information is exchanged between the client and the server:

- o C transmits to RS which resource it requests to access, the kind of action it wants to perform on the resource and the parameters needed for the action.
- o RS transmits to C the result of the attempted access.

3.2. Authentication-Related Tasks

According to the Internet Security Glossary [RFC4949], authentication is "the process of verifying a claim that a system entity or system resource has a certain attribute value." Examples for attribute values are the ID of a device, the type of the device or the name of its owner. Authentication attributes might be (but not necessarily are) suitable to uniquely identify an individual entity.

Several steps must be conducted for authenticating certain attributes of an entity and validating the authenticity of an information:

1. Attribute binding: The attribute that shall be verifiable must be bound to a verifier, e.g. a key. To achieve this, an attribute binding authority has to check if the entity in possession of a certain verifier really possesses the attributes it claims to have. The authority must provide some kind of endorsement information which enables other entities to validate the binding.
2. Authentication: The entity which wants to use the verifier for authenticating an entity checks the attribute-verifier-binding using the endorsement of the claim validation authority and uses the verifier for authenticating an entity or the source of an information.

Step 1 is addressed in Appendix A.2.5. Two types of tasks were defined for step 2: Information authenticity (see Appendix A.2.1) and secure communication (see Appendix A.2.3).

3.3. Authorization-Related Tasks

Several steps must be conducted for authorization:

1. Configuration of authorization information: The owner must configure the authorization information.
2. Obtaining authorization information: Authorization information must be made available to the entity which enforces the authorization.
3. Authorization validation: The authorization of an entity with certain attributes must be checked by mapping the attributes (which must be validated by authentication) to the authorization information.

Tasks for step 1 are defined in Appendix A.2.6. Appendix A.2.4 addresses step 2. Appendix A.2.2 introduces tasks for step 3.

4. Actors

This section describes the various actors in the architecture. An actor is identified by the tasks it has to fulfill. Several actors might share a single device or even be combined in a single piece of software. Interfaces between actors may be realized as protocols or be internal to such a piece of software.

The concept of actors is used to assign the tasks defined in Appendix A to logical functional entities.

4.1. Constrained Level Actors

As described in the problem statement (see Section 2), either C or RS or both of them may be located on a constrained node. We therefore define that C and RS must be able to perform their tasks even if they are located on a constrained node. Thus, C and RS are considered to be Constrained Level Actors.

C performs the following tasks:

- o Negotiate means for secure communication (Task TSecureComm, see Appendix A.2.3).
- o Validate that an entity is an authorized source for R (Task TValSourceAuthz, see Appendix A.2.2).
- o Securely transmit an access request (Task TSendReq, see Appendix A.1.2).

- o Validate that the response to an access request is authentic (Task TAuthnResp, see Appendix A.2.1).
- o Process the response to an access request (Task TProcResp, see Appendix A.1.1).

RS performs the following tasks:

- o Negotiate means for secure communication (Task TSecureComm, see Appendix A.2.3).
- o Validate the authenticity of an access request (Task TAuthnReq, see Appendix A.2.1).
- o Validate the authorization of the requester to access the requested resource as requested (Task TValAccessAuthZ, see Appendix A.2.2).
- o Process an access request (Task TProcReq, see Appendix A.1.1).
- o Securely transmit a response to an access request (Task TSendResp, see Appendix A.1.2).

R is an item of interest such as a sensor or actuator value. R is considered to be part of RS and not a separate actor. The device on which RS is located might contain several resources of different resource owners. For simplicity of exposition, these resources are described as if they had separate RS.

As C and RS do not necessarily know each other they might belong to different security domains.

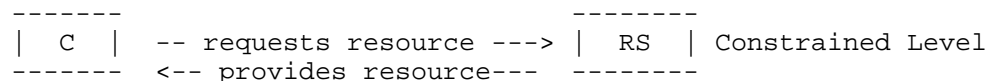


Figure 2: Constrained Level Actors

4.2. Principal Level Actors

Our objective is that C and RS are under control of principals in the physical world, the Client Owner (CO) and the Resource Owner (RO) respectively. The owners decide about the security policies of their respective devices and belong to the same security domain.

CO is in charge of C, i.e. CO specifies security policies for C, e.g. with whom C is allowed to communicate. By definition, C and CO belong to the same security domain.

CO must fulfill the following task:

- o Configure for C authorization information for sources for R (Task TConfigSourceAuthz, see Appendix A.2.6).

RO is in charge of R and RS. RO specifies authorization policies for R and decides with whom RS is allowed to communicate. By definition, R, RS and RO belong to the same security domain.

RO must fulfill the following task:

- o Configure for RS authorization information for accessing R (Task TConfigAccessAuthz, see Appendix A.2.6).

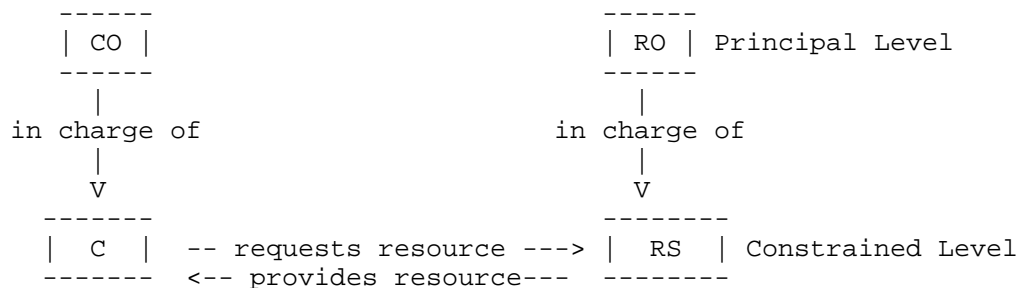


Figure 3: Constrained Level Actors and Principal Level Actors

4.3. Less-Constrained Level Actors

Constrained level actors can only fulfill a limited number of tasks and may not have network connectivity all the time. To relieve them from having to manage keys for numerous devices and conducting computationally intensive tasks, another complexity level for actors is introduced. An actor on the less-constrained level belongs to the same security domain as its respective constrained level actor. They also have the same principal.

The Authentication Manager (AM) belongs to the same security domain as C and CO. AM acts on behalf of CO. It assists C in authenticating RS and determining if RS an authorized source for R. AM can do that because for C, AM is the authority for claims about RS.

AM performs the following tasks:

- o Validate on the client side that an entity has certain attributes (Task TValSourceAttr, see Appendix A.2.5).
- o Obtain authorization information about an entity from C's owner and provide it to C. (Task TObtainSourceAuthz, see Appendix A.2.4).
- o Negotiate means for secure communication to communicate with C (Task TSecureComm, see Appendix A.2.3).

The Authorization Server (AS) belongs to the same security domain as R, RS and RO. AS acts on behalf of RO. It supports RS by authenticating C and determining C's permissions on R. AS can do that because for RS, AS is the authority for claims about C.

AS performs the following tasks:

- o Validate on the server side that an entity has certain attributes (Task TValReqAttr, see Appendix A.2.5).
- o Obtain authorization information about an entity from RS' owner and provide it to RS (Task TObtainAccessAuthz, see Appendix A.2.4).
- o Negotiate means for secure communication to communicate with RS (Task TSecureComm, see Appendix A.2.3).

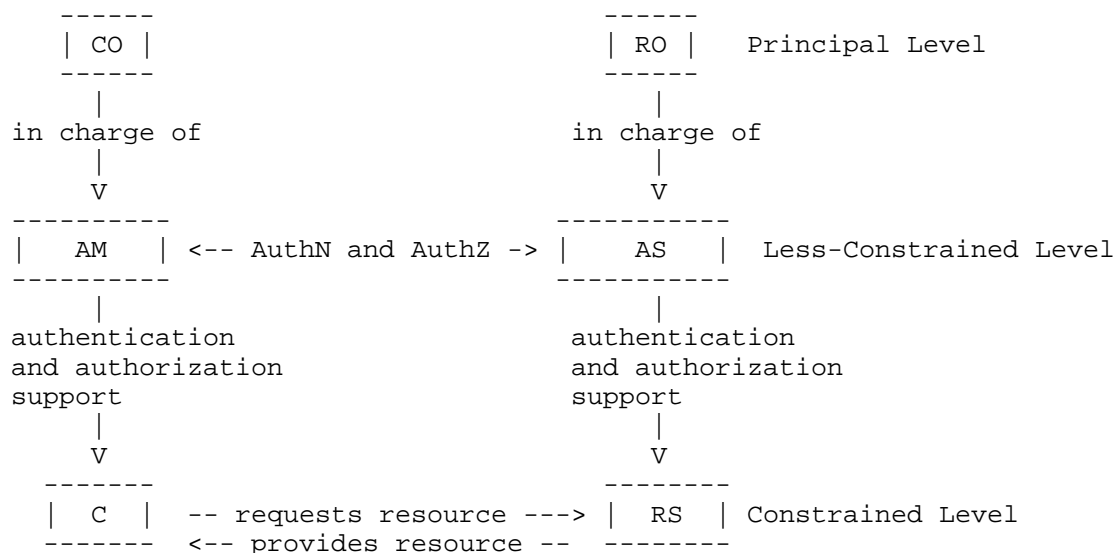


Figure 4: Overview of all Complexity Levels

For more detailed graphics please consult the PDF version.

5. Protocol Requirements

Devices on the less-constrained level potentially are more powerful than constrained level devices in terms of processing power, memory, non-volatile storage. This results in different requirements for the protocols used on these levels.

5.1. Constrained Level Protocols

A protocol is considered to be on the constrained level if it is used between the actors C and RS which are considered to be constrained (see Section 4.1). C and RS might not belong to the same security domain. Therefore, constrained level protocols are required to work between different security domains.

Commonly used Internet protocols can not in every case be applied to constrained environments. In some cases, tweaking and profiling is required. In other cases it is beneficial to define new protocols which were designed with the special characteristics of constrained environments in mind.

On the constrained level, protocols must be used which address the specific requirements of constrained environments. The Constrained Application Protocol (CoAP) [RFC7252] should be used as transfer protocol if possible. CoAP defines a security binding to Datagram Transport Layer Security Protocol (DTLS) [RFC6347]. Thus, DTLS should be used for channel security.

Constrained devices have only limited storage space and thus cannot store large numbers of keys. This is especially important because constrained networks are expected to consist of thousands of nodes. Protocols on the constrained level should keep this limitation in mind.

5.1.1. Cross Level Support Protocols

Protocols which operate between a constrained device on one side and the corresponding less constrained device on the other are considered to be (cross level) support protocols. Protocols used between C and AM or RS and AS are therefore support protocols.

Support protocols must consider the limitations of their constrained endpoint and therefore belong to the constrained level protocols.

5.2. Less-Constrained Level Protocols

A protocol is considered to be on the less-constrained level if it is used between the actors AM and AS. AM and AS might belong to different security domains.

On the less-constrained level, HTTP [RFC7230] and Transport Layer Security (TLS) [RFC5246] can be used alongside or instead of CoAP and DTLS. Moreover, existing security solutions for authentication and authorization such as the Web Authorization Protocol (OAuth) [RFC6749] and Kerberos [RFC4120] can likely be used without modifications and there are no limitations for the use of a Public Key Infrastructure (PKI).

6. IANA Considerations

None

7. Security Considerations

This document discusses security requirements for the ACE architecture.

8. Acknowledgments

The author would like to thank Carsten Bormann, Olaf Bergmann and Klaus Hartke for their valuable input and feedback.

9. References

9.1. Normative References

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, May 2014.

9.2. Informative References

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

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

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

- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.
- [RFC7230] Fielding, R. and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, June 2014.
- [RFC7231] Fielding, R. and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, June 2014.
- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.

Appendix A. List of Tasks

This section defines the tasks which must be performed in the given scenario (see Section 2) starting from communication related tasks and then deriving the required security-related tasks. An overview of the tasks can be found in Section 3.

A task has the following structure:

- o The name of the task which has the form TXXX
- o One or more Requirements (if applicable) of the form RqXXX
- o One or more Preconditions (if applicable) of the form PreXXX
- o One or more Postconditions (if applicable) of the form PostXXX

Requirements have to be met while performing the task. They derive directly from the scenario (see Section 2) or from the security requirements defined for the scenario. Preconditions have to be fulfilled before conducting the task. Postconditions are the results of the completed task.

We start our analysis with the processing tasks and define which preconditions need to be fulfilled before these tasks can be conducted. We then determine which tasks therefore need to be performed first (have postconditions which match the respective preconditions).

Note: Regarding the communication, C and RS are defined as entities each having their set of attributes and a verifier which is bound to these attributes. Attributes are not necessarily usable to identify

an individual C or RS. Several entities might have the same attributes.

A.1. Basic Scenario

The intended result of the interaction between C and RS is that C has successfully accessed R. C gets to know that its access request was successful by receiving the answer from RS.

The transmission of information from C to RS comprises two parts: sending the information on one side and receiving and processing it on the other. Security has to be considered at each of these steps.

A.1.1. Processing Information

The purpose of the communication between C and RS is C's intent to access R. To achieve this, RS must process the information about the requested access and C must process the information in the response to a requested access. The request and the response might both contain resource values.

The confidentiality and integrity of R require that only authorized entities are able to access R (see Rq0.1). Therefore, C and RS must check that the information is authentic and that the source of the information is authorized to provide it, before the information can be processed. C must validate that RS is an authorized source for R. RS must validate that C is authorized to access R as requested.

If proxies are used, it depends on the type of proxy how they are integrated into the communication and what kind of security relationships need to be established. A future version of this document will provide more details on this topic. At this point we assume that C and RS might receive the information either from RS or C directly or from a proxy which is authorized to speak for the respective communication partner.

- o Task TProcResp: Process the response to an access request.
Description: C processes the response to an access request according to the reason for requesting the resource in the first place. The response might include resource values or information about the results of a request.
Requirements:
 - * RqProcResp.1: Is performed by C (derives from the problem statement).
 - * RqProcResp.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).Preconditions:

- * PreProcResp.1: A response to an access request was sent (see Appendix A.1.2).
 - * PreProcResp.2 (required for Rq0.2): C validated that the response to an access request is authentic, i.e. it stems from the entity requested in TSendReq (see Appendix A.1.2), i.e. RS or an entity which is authorized to speak for RS (see Appendix A.2.1).
 - * PreProcResp.3 (required for Rq0.2): C validated that RS or the entity which is authorized to speak for RS is an authorized source for R (see Appendix A.2.2).
 - Postcondition:
 - * PostProcResp.1: C processed the response.
- o Task TProcReq: Process an access request.
- Description: RS either performs an action on the resource according to the information in the request, or determines the reason for not performing an action.
- Requirements:
- * RqProcReq.1: Is performed by RS.
 - * RqProcReq.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).
- Preconditions:
- * PreProcReq.1: An access request was sent (see Appendix A.1.2).
 - * PreProcReq.2 (needed for Rq0.1): RS validated that the request is authentic, i.e. it stems from C or an entity which is authorized to speak for C and is fresh. (see Appendix A.2.1).
 - * PreProcReq.3 (needed for Rq0.1): RS validated the authorization of C or the entity which is authorized to speak for C to access the resource as requested (see Appendix A.2.2).
- Postconditions:
- * PostProcReq.1: The access request was processed (fulfills PreSendResp.1, see Appendix A.1.2).

Note: The preconditions PreProcReq.2 and PreProcReq.3 must be conducted together. RS must assure that the response is bound to a verifier, the verifier is bound to certain attributes and the authorization information refers to these attributes.

A.1.2. Sending Information

The information needed for processing has to be transmitted at some point. C has to transmit to RS which resource it wants to access with which actions and parameters. RS has to transmit to C the result of the request. The request and the response might both contain resource values. To fulfill Rq0.1, the confidentiality and integrity of the transmitted data has to be assured.

If proxies are used, it depends on the type of proxy how they need to be handled. A future version of this document will provide more details on this topic. At this point we assume that C and RS might transmit the message either to RS and C directly or to a proxy which is authorized to speak for the respective communication partner.

- o Task TSendReq: Securely transmit an access request.
Description: C wants to access a resource R hosted by the resource server RS. To achieve this, it has to transmit some information to RS such as the resource to be accessed, the action to be performed on the resource and, if a writing access is requested, the value to write. C might send the request directly to RS or to an entity which is authorized to speak for RS. C assures that the request reaches the proper R. C binds the request to C's verifier to ensure the integrity of the message. C uses means to assure that no unauthorized entity is able to access the information in the request.
Requirements:
 - * RqSendReq.1: Is performed by C (derives from problem statement).
 - * RqSendReq.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).
 - * RqSendReq.3: As the request might contain resource values, the confidentiality and integrity of the request must be ensured during transmission. Only authorized parties must be able to read or modify the request (derives from Rq0.1).Preconditions:
 - * PreSendReq.1: Validate that the receiver is an authorized source for R (see Appendix A.2.2).
 - * PreSendReq.2: To assure that the request reaches the proper RS, that no unauthorized party is able to access the request, and that the information in the request is bound to C's verifier it is necessary to negotiate means for secure communication with RS (see Appendix A.2.3).Postconditions:
 - * PostSendReq.1: The request was sent securely to RS (necessary for Rq0.1) (fulfills PreProcReq.1, see Appendix A.1.1).

Note: The preconditions PreSendReq.1 and PreSendReq.2 must be conducted together. C must assure that the request reaches an entity with certain attributes and that the authorization information refers to these attributes.

- o Task TSendResp: Securely transmit a response to an access request.
Description: RS sends a response to an access request to inform C about the result of the request. RS must assure that response reaches the requesting C. RS might send the response to C or to an entity which is authorized to speak for C. The response might

contain resource values. RS binds the request to RS's verifier to ensure the integrity of the message. RS uses means to assure that no unauthorized entity is able to access the information in the response.

Requirements:

- * RqSendResp.1: Is performed by RS (derives from the problem statement).
- * RqSendResp.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).
- * RqSendResp.3: As the response might contain resource values, the confidentiality and integrity of the response must be ensured during transmission. Only authorized parties must be able to read or modify the response (derives from Rq0.1).

Preconditions:

- * PreSendResp.1: An access request was processed (see Appendix A.1.1).
- * PreSendResp.2: If information about R is transmitted, validate that the receiver is authorized to access R (see Appendix A.2.2).
- * PreSendResp.3: RS must assure that the response reaches the requesting C, no unauthorized party is able to access the response and the information in the response is bound to RS' verifier: Means for secure communication were negotiated (see Appendix A.2.3).

Postconditions:

- * PostSendResp.1: A response to an access request was sent (fulfills PreProcResp.1, see Appendix A.1.1).

A.2. Security-Related Tasks

A.2.1. Information Authenticity

This section addresses information authentication, i.e. using the verifier to validate the source of an information. Information authentication must be conducted before processing received information. C must validate that a response to an access request is fresh, really stems from the queried RS (or an entity which is authorized to speak for RS) and was not modified during transmission. RS must validate that the information in the access request is fresh, really stems from C (or an entity which is authorized to speak for C) and was not modified during transmission.

The entity which processes the information must be the entity which is validating the source of the information.

C and RS must assure that the authenticated source of the information is authorized to provide the information.

- o Task TAuthnResp: Validate that the response to an access request is authentic.
Description: C checks if the response to an access request stems from an entity in possession of the respective verifier and is fresh. Thus, C validates that the response stems from the queried RS or an entity which is authorized to speak for RS.
Requirements:
 - * RqAuthnResp.1: Must be performed by C.
 - * RqAuthnResp.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).Preconditions:
 - * PreAuthnResp.1: Means for secure communication were negotiated (see Appendix A.2.3).Postconditions:
 - * PostAuthnResp.1: C knows that the response came from RS (fulfills PreProcResp.2, see Appendix A.1.1).
- o Task TAuthnReq: Validate the authenticity of a request.
Description: RS checks if the request stems from an entity in possession of the respective verifier and is fresh. Thus, RS validates that the request stems from C or an entity which is authorized to speak for C.
Requirements:
 - * RqAuthnReq.1: Must be performed by RS.
 - * RqAuthnReq.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).Preconditions:
 - * PreAuthnReq.1: Means for secure communication were negotiated (see Appendix A.2.3).Postconditions:
 - * PostAuthnReq.1: RS knows that the request is authentic (fulfills PreProcReq.2, see Appendix A.1.1).

A.2.2. Authorization Validation

This section addresses the validation of the authorization of an entity. The entity which processes the information must validate that the source of the information is authorized to provide it. The processing entity has to verify that the source of the information has certain attributes which authorize it to provide the information: C must validate that RS (or the entity which speaks for RS) is in possession of attributes which are necessary for being an authorized source for R. RS must validate that C (or the entity which speaks for C) has attributes which are necessary for a permission to access R as requested.

- o Task TValSourceAuthz: Validate that an entity is an authorized source for R.
Description: C checks if according to CO's authorization policy and the authentication endorsement provided by the attribute binding authority, RS (or an entity which speaks for RS) is authorized to be a source for R. RS assures that the entity's verifier is bound to certain attributes and the authorization information refers to these attributes.
Requirements:
 - * RqValSourceAuthz.1: Is performed by C
 - * RqValSourceAuthz.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).Preconditions:
 - * PreValSourceAuthz.1: Authorization information about the entity are available. Requires obtaining authorization information about the entity from C's owner (see Appendix A.2.4).
 - * PreValSourceAuthz.2: Means to validate that the entity has certain attributes which are relevant for the authorization: Requires validation of claims about RS (see Appendix A.2.5).Postconditions:
 - * PostValSourceAuthz.1: The entity which performs the task knows that an entity is an authorized source for R (fulfills PreProcResp.3, see Appendix A.1.1 and PreSendReq.1, see Appendix A.1.2).
- o Task TValAccessAuthZ: Validate the authorization of the requester to access the requested resource as requested.
Description: R's owner configures which clients are authorized to perform which action on R. RS has to check if according to RO's authorization policy and the authentication endorsement provided by the attribute binding authority, C (or an entity which speaks for C) is authorized to access R as requested. RS assures that requester's verifier is bound to certain attributes and the authorization information refers to these attributes.
Requirements:
 - * RqValAccessAuthz.1: Is performed by RS
 - * RqValAccessAuthz.2: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).Preconditions:
 - * PreValAccessAuthz.1: Authorization information about the entity are available. Requires obtaining authorization information about the entity from RS's owner (see Appendix A.2.4).
 - * PreValAccessAuthz.2: Means to validate that the entity has certain attributes which are relevant for the authorization: Requires validation of claims about C or the entity which speaks for C (see Appendix A.2.5).

Postconditions:

- * PostValAccessAuthz.1: The entity which performs the task knows that an entity is authorized to access R with the requested action (fulfills PreProcReq.3, see Appendix A.1.1).

A.2.3. Transmission Security

To ensure the confidentiality and integrity of information during transmission means for secure communication have to be negotiated between the communicating parties.

- o Task TSecureComm: Negotiate means for secure communication.
Description: To ensure the confidentiality and integrity of transmitted information, means for secure communication have to be negotiated. Channel security as well as object security solutions are possible. Details depend on the used solution and are not in the scope of this document.

Requirements:

- * RqSecureComm.1: Must be performable by a constrained device (derives from the problem statement: C and / or RS are constrained).

Preconditions:

- * PreSecureComm.1: Sender and receiver must be able to validate that the entity in possession of a certain verifier has the claimed attributes. (see Appendix A.2.5).

Postconditions:

- * PostSecureComm.1: C and RS can communicate securely: The integrity and confidentiality of information is ensured during transmission. The sending entity can use means to assure that the information reaches the intended receiver so that no unauthorized party is able to access the information. The sending entity can bind the information to the entity's verifier (fulfills PreSendResp.3 and PreSendReq.2, see Appendix A.1.2 as well as PreAuthnResp.1 and PreAuthnReq.1, see Appendix A.2.1).

A.2.4. Obtain Authorization information

As described in Section 3.3, the authorization of an entity requires several steps. The authorization information must be configured by the owner and provided to the enforcing entity.

- o Task TObtainSourceAuthz: Obtain authorization information about an entity from C's owner.

Description: C's owner defines authorized sources for R. The authorization information must be made available to C to enable it to enforce CO's authorization information. To facilitate the configuration for the owner this device should have a user interface. The authorization information has to be made available to C in a secure way.

Requirements:

- * RqObtainSourceAuthz.1: Must be performed by an entity which belongs to C's security domain.
- * RqObtainSourceAuthz.2: Must be performed by an entity which is authorized to speak for C's owner concerning authorized sources for R.
- * RqObtainSourceAuthz.3: Should be performed by a device which can provide some sort of user interface to facilitate the configuration of authorization information for C's owner.

Preconditions:

- * PreObtainSourceAuthz.1: C's owner configured authorized sources for R (see Appendix A.2.6).

Postconditions:

- * PostObtainSourceAuthz.1: C obtained RS' authorization to be a source for R (fulfills PreValSourceAuthz.1, see Appendix A.2.2).

- o Task TObtainAccessAuthz: Obtain authorization information about an entity from RS' owner.

Description: RS' owner defines if and how C is authorized to access R. The authorization information must be made available to RS to enable it to enforce RO's authorization policies. To facilitate the configuration for the owner this device should have a user interface. The authorization information has to be made available to RS in a secure way.

Requirements:

- * RqObtainAccessAuthz.1: Must be performed by an entity which belongs to R's security domain.
- * RqObtainAccessAuthz.2: Must be performed by an entity which is authorized to speak for R's owner concerning authorization of access to R.
- * RqObtainAccessAuthz.3: Should be performed by a device which can provide some sort of user interface to facilitate the configuration of authorization information for R's owner.

Preconditions:

- * PreObtainAccessAuthz.1: R's owner configured authorization information for the access to R (see Appendix A.2.6).

Postconditions:

- * PostObtainAccessAuthz.1: RS obtained C's authorization for accessing R (fulfills PreValAccessAuthz.1, see Appendix A.2.2).

A.2.5. Attribute Binding

As described in Section 3.2, several steps must be conducted for authentication. This section addresses the binding of attributes to a verifier.

For authentication it is necessary to validate if an entity has certain attributes. An example for such an attribute in the physical world is the name of a person or her age. In constrained environments, attributes might be the name of the owner or the type of device. Authorizations are bound to such attributes.

The possession of attributes must be verifiable. For that purpose, attributes must be bound to a verifier. An example for a verifier in the physical world is a passport. In constrained environments, a verifier will likely be the knowledge of a secret.

At some point, an authority has to check if an entity in possession of the verifier really possesses the claimed attributes. In the physical world, government agencies check your name and age before they give you a passport.

The entity that validates the claims has to provide some kind of seal to make its endorsement verifiable for other entities and thus bind the attributes to the verifier. In the physical world passports are stamped by the issuing government agencies (and must only be provided by government agencies anyway).

- o Task TValSourceAttr: Validate on the client side that an entity has certain attributes.
Description: The claim that an entity has certain attributes has to be checked and made available for C in a secure way. The validating party states that an entity in possession of a certain key has certain attributes and provides C with means to validate this endorsement.
Requirements:
 - * RqValSourceAttr.1: Must be performed by an entity which belongs to C's security domain and is an authority for claims about RS.
 - * RqValSourceAttr.2: The executing entity must have the means to fulfill the task (e.g. enough storage space, computational power, a user interface to facilitate the configuration of authentication policies).Postconditions:
 - * PostValSourceAttr.1: Means for authenticating (validating the attribute-verifier-binding of) other entities were given to C in form of a verifiable endorsement (fulfills PreValSourceAuthz.2, see Appendix A.2.2 and PreSecureComm.1, see Appendix A.2.3).
- o Task TValReqAttr: Validate on the server side that an entity has certain attributes.

Description: The claim that an entity has certain attributes has to be checked and made available for RS in a secure way. The validating party states that an entity in possession of a certain key has certain attributes and provides RS with means to validate this endorsement.

Requirements:

- * RqValReqAttr.1: Must be performed by an entity which belongs to RS' security domain and is an authority for claims about C.
- * RqValReqAttr.2: The executing entity must have the means to fulfill the task (e.g. enough storage space, computational power, a user interface to facilitate the configuration of authentication policies).

Postconditions:

- * PostValReqAttr.1: Means for authenticating (validating the attribute-verifier-binding of) other entities were given to RS in form of a verifiable endorsement (fulfills PreValSourceAuthz.2, see Appendix A.2.2 and PreSecureComm.1, see Appendix A.2.3).

A.2.6. Configuration of Authorization Information

As stated in Section 3.3, several steps have to be conducted for authorization. This section is about the configuration of authorization information.

The owner of a device or resource wants to be in control of her device and her data. For that purpose, she has to configure authorization information. C's owner might want to configure which attributes an entity must possess to be a source for R. R's owner might want to configure which attributes are required for accessing R with a certain action.

- o Task TConfigSourceAuthz: Configure for C authorization information for sources for R.
Description: C's owner has to define authorized sources for R.
Requirements:
 - * RqConfigSourceAuthz.1: Must be provided by C's owner.Postconditions:
 - * PostConfigSourceAuthz.1: The authorization information are available to a device which performs TObtainSourceAuthz (fulfills PreObtainSourceAuthz.1 see Appendix A.2.4).
- o Task TConfigAccessAuthz: Configure for RS authorization information for accessing R.
Description: R's owner has to configure if and how an entity with certain attributes is allowed to access R.
Requirements:
 - * RqConfigAccessAuthz.1: Must be provided by R's owner.Postconditions:

* PostConfigAccessAuthz.1: The authorization information are available to the device which performs TObtainAccessAuthz (fulfills PreObtainAccessAuthz.1, see Appendix A.2.4).

Author's Address

Stefanie Gerdes
Universitaet Bremen TZI
Postfach 330440
Bremen D-28359
Germany

Phone: +49-421-218-63906
Email: gerdes@tzi.org

CoRE Working Group
Internet-Draft
Intended Status: Informational
Expires: August 18, 2014

L. Seitz
SICS Swedish ICT
G. Selander
Ericsson

February 14, 2014

Design Considerations for Security Protocols in Constrained Environments
draft-seitz-ace-design-considerations-00

Abstract

Considerable effort has been spent on securing existing Internet standard authentication and authorization protocols such as TLS, Kerberos, and OAuth, among others. It would save a lot of effort if these protocols could be profiled to be feasible for constrained environments, with some easily obtainable security considerations.

However, these protocols were typically not designed with constrained environments in mind, so profiling of an existing protocol may result in a far from optimal solution. Moreover they are not necessarily complying with their original design objectives outside their intended domain of application.

This document examines the impact of typical characteristics of security protocols (e.g. cryptographic calculations, number and size of protocol messages) in a constrained environment. The goal is to provide decision support when different resource usage optimizations are possible in the adaptation of a security protocol for this setting.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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".

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright and License 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	4
1.1 Terminology	4
2. Background	4
2.1. Device assumptions	5
2.2 Relevant Factors	5
2.3 Security protocols in constrained environments	6
3. Protocol design considerations	7
3.1 Straightforward optimizations	7
3.1.1 Smaller messages	7
3.1.2 Fewer messages	7
3.1.3 Less computations	8
3.1.4 Reduce RAM usage	8
3.1.5 Reduce code size	8
3.2 Trade-offs	9
3.2.1 Fewer vs smaller messages	9
3.2.2 Crypto vs message exchange	9
3.2.3 Transmitting vs receiving messages	9
3.2.4 Distributing costs over deployment life time	10
3.2.5 Outsourcing heavy computations	10
3.2.6 DoS mitigation and anti-spoofing in the Internet	10
3.2.7 Outsourcing key management	11
3.2.8 Verifying authorization	11

4. Security Considerations	11
5. IANA Considerations	12
6. Acknowledgements	12
7. References	13
7.1 Normative References	13
7.2 Informative References	13
Authors' Addresses	14

1. Introduction

When adapting security protocols for constrained nodes, one has to take into account the various resource limitations. While it might be tempting to optimize the usage of a certain resource (e.g. minimizing RAM consumption), such an approach might produce a less-than-optimal overall solution, compared to a more holistic approach that leverages the combined effect of different optimization possibilities.

The goal of this document is to summarize some characteristics of security protocols and weigh their impact against each other in order to allow effective trade-offs when adapting existing protocols to a constrained setting. While there is some overlap with the scope of the Lightweight Implementation Guidance WG, this document is aimed more at security protocol profiling and design than actual implementation decisions that are the main focus of LWIG.

1.1 Terminology

Certain security-related terms are to be understood in the sense defined in [RFC4949]. These terms include, but are not limited to, "authentication", "authorization", "confidentiality", "encryption", "data integrity", "message authentication code", and "verify".

Terminology for constrained environments is defined in [I-D.ietf-lwig-terminology] e.g. "constrained device".

2. Background

We are assuming a multi-party protocol setting with at least the following parties

- a) a resource server hosting resources
- b) a client seeking access to some resource, and
- c) an authorization server acting Trusted Third Party (TTP) for key distribution and access control handling.

The resource server and/or the client is assumed to be constrained, but the authorization server is not.

The authorization server can provide authentication and authorization means (e.g. cryptographic keys, access control information, certificates) for the other parties.

There are various authentications and authorizations taking place in this multi-party protocol. For example, the client and

authorization server mutually authenticates and the client is being authorized by the authorization server. The resource server needs authenticate information provided by the authorization server, based on a previously established relationship (e.g. shared symmetric keys). Finally when the client communicates with the resource server, the client's authorization needs to be verified, which might include authentication of the client.

Note: Security protocols designed to handle authentication and authorization between two mutually unknown less-constrained peers are not necessarily adapted to the current setting, where optimizations can be made by relying on an relatively unconstrained TTP.

2.1. Device assumptions

Devices may be constrained in different ways, as described in the LWIG terminology document [I-D.ietf-lwig-terminology]. This work is targeting class 1 devices, but may be applicable even the most constrained class of devices (C0) if supported by relevant proxy functionality. Class 2 devices probably do not need any special considerations, since they can mostly support the same protocols as unconstrained devices.

A device for which these considerations apply could e.g. run the following protocol stack, potentially supported by a proxy:

- o The application layer protocol is CoAP [I-D.ietf-core-coap], using UDP at the transport layer.
- o CoAP will be running on top of DTLS [RFC6347].
- o IPv6 [RFC4291] is assumed to be the Internet layer protocol on top of the adaptation layer 6LoWPAN [RFC4944].
- o IEEE 802.15.4 [IEEE802] is assumed as the Link layer protocol for wireless communication. We assume that a large proportion of the target devices will communicate over wireless channels.

2.2 Relevant Factors

From the LWIG terminology draft [I-D.ietf-lwig-terminology] we can list the following resources that need to be considered in general:

- o RAM memory (required state and buffers for running protocols)
- o Flash/ROM memory (required libraries and code complexity)
- o Computational power (required processing speed)
- o Electrical energy (battery consumption, if not mains-powered)
- o User interface and physical accessibility (for performing manual operations directly on the device)
- o Network (bit rate, loss rate, dynamic topology, fragmentation,

lack of advanced services)

The consumers of these resources in the case of security protocols can be summarized as follows:

- o Cryptographic algorithms
 - based on symmetric cryptography
 - based on asymmetric cryptography
 - (orthogonal) implemented by a co-processor (e.g. AES, SHA, ECC)
- o Composing/parsing protocol messages (e.g. Base64 en/decoding, JSON, ASN.1, CBOR)
- o Sending/receiving protocol messages
- o Listening, while waiting to receive protocol messages

2.3 Security protocols in constrained environments

One of the potential advantages with extending basic Internet Protocols to constrained nodes is that other standardized protocols can be applied too.

In particular in the case of security protocols, there is a considerable effort spent to eliminate flaws and weaknesses that could otherwise be exploited for attacking the system. It would save a lot of effort if it was possible to profile these protocols for running efficiently in a constrained environment while maintaining their security properties.

However, the profiling of a protocol may result in a far from optimal solution. For example assume that a constrained profile of a security protocol is made by reducing the message sizes. Such a protocol may still be badly suited for constrained devices e.g. because the number of round trips is what makes the latency high, and reducing that would essentially change the security properties of the protocol.

Moreover, as many of these protocols were not designed for a constrained environment, they are not necessarily complying with their original design objectives outside their intended domain of application. Even security objectives that applied to the Internet may be violated: e.g. a DoS mitigation measure that is based on a processing commitment by a client (a "puzzle", see e.g. [RFC5201]) may be inappropriate if the server is much more constrained than the client.

This memo is intended to support the adaptation of an existing security protocol for a constrained environment by providing some

considerations on resource consumption. Furthermore this memo documents the assumptions that were made as a basis for these considerations.

3. Protocol design considerations

3.1 Straightforward optimizations

This section lists some potential targets for resource optimizations.

3.1.1 Smaller messages

Reducing message size will reduce composing/parsing and sending/receiving costs which is favorably impacting energy consumption and latency. Some specific considerations:

- o Smaller than CoAP payload size (1024 bytes) avoids fragmentation at the application layer.
- o Smaller than the maximum MAC-layer frame size (e.g. 127 bytes for IEEE 802.15.4) avoids fragmentation at the link layer.
- o The largest messages are potentially those containing certificates or authorization tokens, so reducing their size significantly will have a large impact.

3.1.2 Fewer messages

Removing message exchanges or round trips have potentially large impact on energy consumption and latency.

Reducing the number of messages in a given security protocol is in general not possible without changing the essential security properties of the protocol. Experiments by Google with TLS false start [I-D.bmoeller-tls-falsestart] and TLS snap start [I-D.agl-tls-snapstart] illustrate the difficulty of trying to reduce the number of messages in an established security protocol.

Challenge-response based authentication protocols may potentially be replaced with other protocols with alternative measures to ensure freshness, such as time or sequence numbers. Such an approach would require fewer message passes, but ensuring freshness can be problematic, since some constrained devices may not be able to reliably measure time.

On the other hand, there are long lifetime battery powered IEEE 802.15.4e devices implementing Time Slotted Channel Hopping (TSCH) which has good time synchronization properties, since that is required for communication.

3.1.3 Less computations

One way of reducing the complexity of required computations is to reduce the number of public key operations used during normal operations, e.g. by keeping existing sessions alive, or generating session resumption state on a less constrained device. The drawback in this case is that either more RAM or more sending and receiving of messages are needed.

An alternative is to replace public key operations with symmetric key operations. Significant reductions in resource consumption can be achieved by using symmetric cryptography instead of asymmetric cryptography, since asymmetric cryptography generally requires larger libraries (e.g. BigInteger, elliptic curves), and consumes more RAM, processing power and energy than symmetric algorithms.

However, it is not always possible to make this replacement as some of the properties of asymmetric cryptography, such as non-repudiation of signatures, and non-confidential key distribution do not apply for symmetric keys. It may require a change in trust model, where a TTP is assumed e.g. for key management.

3.1.4 Reduce RAM usage

Reducing the usage of RAM memory can be achieved by reducing the size of variable state information required by a protocol. Different security protocols and -modes have different requirements in this respect. Optimizations may potentially be done by profiling certain options of the protocol to predefined, default values.

Another possibility is to simplify parsing and processing of protocol messages, leading to smaller libraries that need to be loaded into memory. Further the size of the protocol messages, e.g. certificates and authorization tokens, directly affects the size of the buffers that need to be allocated for receiving and sending them, so keeping them small also helps.

3.1.5 Reduce code size

The overall size of the code is influenced mainly by the size of the libraries needed for cryptography and parsing messages (ASN.1, JSON, XML). In general asymmetric cryptography requires larger libraries (e.g. BigInteger, Elliptic curves) than symmetric cryptography. Minimal libraries for parsing ASN.1 and JSON are roughly comparable in size (around 6 kB) while even minimal XML parsers generally have a significantly larger size.

3.2 Trade-offs

This section looks at the more difficult question how to weigh different optimizations against each other. We emphasize in this section the potential role of the authorization server as an enabler for some of the optimizations.

3.2.1 Fewer vs smaller messages

When comparing reduction of message size versus sending fewer messages in total, if one takes into account the overhead of setting up a bearer, it is more efficient to send longer messages than shorter messages. Considering fragmentation it is better to send messages shorter than the fragmentation limit. Therefore optimal message size seems to be just below the fragmentation limit. Note that fragmentation carries an additional performance penalty in excess of just adding the overhead of sending several fragments, since fragmenting a message increases the risk that a fragment is lost and that the message as a whole needs to be retransmitted.

3.2.2 Crypto vs message exchange

It is known that in wireless constrained devices, the energy consumption for sending and receiving messages is high, and significantly higher than symmetric crypto operations [Margil0impact] and [Meu08engery]. Hence if it is possible to send fewer messages at the cost of delegating some symmetric crypto to the constrained device, such a trade off is favorable. The potential drawback is increased latency and code size. The latter could probably be avoided by reusing existing symmetric algorithms that are needed anyway.

Results from [Meu08engery] indicate that energy consumption for public key operations is on par or greater than message exchange for a particular security protocol. However, the efficiency of processing is increasing: The processing power follows Moore's law (up to point) and depends on the manufacturing technology while the transmission/reception power is based on laws of physics laws that don't change with manufacturing. So processing will be more and more energy efficient (up to a point) while the transmission/reception remains almost stable in terms of energy efficiency.

3.2.3 Transmitting vs receiving messages

Results comparing energy consumption of transmitting versus receiving messages seem contradictory. While [Margil0impact] indicates that receiving a message is much cheaper in energy consumption, than sending, [Meu08engery] seems to suggest that both costs are roughly

on par.

An important point from [Meu08engery] is that one should consider the cost of listening for the next message in a protocol, while the other party is performing some computations. It is not obvious how much impact smart listening techniques such as Low Power Listening (LPL) or X-MAC [Bue06xmac] have.

Our conclusion on this issue is that it warrants further investigation in order to determine whether it should influence protocol design and profiling or not.

3.2.4 Distributing costs over deployment life time

Provisioning (e.g. access control lists) has a cost which potentially may be amortized over the lifetime of a deployment. Security protocol establishment (e.g. DTLS handshake) may similarly have a high cost that but can be acceptable, if the established session can be used for a long time. The drawback is that storage or RAM memory is consumed to save the state of the provisioned data or the established protocol.

3.2.5 Outsourcing heavy computations

A method of saving computational effort is to outsource computations to a less constrained TTP e.g. authorization decisions and policy management to the authorization server. Note however that this may be changing the trust model of the original protocol, and if the constrained device needs the result of the outsourced computation, this information must be transported in a secure way which in turn incurs a non-negligible cost.

3.2.6 DoS mitigation and anti-spoofing in the Internet

As we have seen it is important in a wireless constrained environment to restrict the number of messages sent and received in a protocol.

Some Internet security protocols include DoS mitigation or anti-spoofing mechanisms such as cookies (cf. [RFC6347]) or puzzles (cf. [RFC5201]) which adds message size and/or round trips. These mechanisms were in general not designed for a constrained environment and may potentially make the protocol unnecessarily heavy without efficiently providing the desired effect.

In fact the existence of a TTP allows for more efficient mechanisms, e.g. that a client first commits or proves source address to the authorization server which can assert such properties in an authorization token verified by a constrained server.

3.2.7 Outsourcing key management

Securing communication between two mutually unknown less-constrained peers has a high cost in terms of additional round trips, e.g. to protect against requests from spoofed initiators, DoS mitigation, challenge response protocols etc. In addition, both parties are often contributing to the generation of key material, which requires exchange of data used in key generation. These costs are a consequence of the trust model and is clearly not adapted to the current setting, where optimizations can be made by relying on an relatively unconstrained authorization server.

In addition to providing authorization decisions, the authorization server may support authentication and authorization between resource server and client by e.g.

- o providing symmetric keys to support authentication (cf. Kerberos).
- o providing protected assertions containing statements about client and server, including public key certificates.

3.2.8 Verifying authorization

As noted above, it is desirable to verify authorization of a request as early as possible in a protocol, to reduce unnecessary message exchanges and processing. However, if that involves verifying a digital signature, then the operation is in itself heavily resource consuming and would preferably only take place after it is known that the request is authorized. This is obviously a "catch 22" and there are various options to attempt to design around this.

In the present case, where we assume a TTP with a previously established relationship - say a shared symmetric key - with the resource server, the legitimacy of the request may e.g. be indicated with a Message Authentication Code instead of a digital signature over an authorization decision.

Authentication of client and server may still require verification of digital signature if public keys are used. However, as noted above, the authorization server may also support key distribution and provide symmetric keys for authentication (cf. Kerberos).

4. Security Considerations

This memo deals with design considerations for security protocols, including security trade-offs that can be made to save resources, some of which will come at the cost of weakening security.

Since a security protocol itself consume resources, one factor that needs to be taken into consideration is the possibility for attackers to use these very security protocols in order to mount a denial of service attack.

Each profiled or modified security protocol must bear its own security considerations. Protocol designers need to carefully evaluate the feasibility of stronger (and thus more resource consuming) security against the risks incurred by a weaker security that is more easy to implement or execute on a constrained node.

5. IANA Considerations

This document has no actions for IANA.

6. Acknowledgements

The authors would like to thank Sumit Shingal and Vlasios Tsiatsis for contributing to the discussion and giving helpful comments.

7. References

7.1 Normative References

- [I-D.ietf-core-coap]
Shelby, Z., Hartke, K., Bormann, C., and B. Frank,
"Constrained Application Protocol (CoAP)", draft-ietf-core-coap-18 (work in progress), June 2013.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.
- [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", FYI 36, RFC 4949, August 2007.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

7.2 Informative References

- [I-D.ietf-lwig-terminology]
Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained Node Networks", draft-ietf-lwig-terminology-07 (work in progress), February 2014.
- [IEEE802]
IEEE Computer Society, "IEEE Standard for Local and metropolitan area networks Part 15.4: Low-Rate Wireless Personal
- [I-D.bmoeller-tls-falsestart]
Langley, A., Modadugu, N., and B. Moeller, "Transport Layer Security (TLS) False Start", draft-bmoeller-tls-falsestart-00 (expired draft), June 2010.
- [I-D.agl-tls-snapstart]
Langley, A., "Transport Layer Security (TLS) Snap Start", draft-agl-tls-snapstart-00 (expired draft), June 2010.
- [Meu08engery]
Meulenaer, G., Gosset, F., Standaert, F., and L. Vandendorpe, "On the Energy Cost of Communication and

Cryptography in Wireless Sensor Networks", proceedings of the IEEE International Conference on Wireless and Mobile Computing, 2008.

[Margil0impact]

Margi, C., Oliveira, B., Sousa, G., Simplicio, M., Barreto, P., Carvalho, T., Naeslund, M., and R. Gold, "Impact of Operating Systemes on Wireless Sensor Networks (Security) Applications and Testbeds", proceedings of the 19th International Conference on Computer Communications and Networks (ICCCN), 2010.

[Bue06xmac]

Buettner, M., Yee, G., Anderson, E., and R. Han, "X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks", proceedings of SenSys'06, 2006

[RFC5201] Moskowitz, R., Nikander, P., Jokela, P., Ed., and T. Henderson, "Host Identity Protocol", RFC 5201, April 2008.

Authors' Addresses

Ludwig Seitz
SICS Swedish ICT AB
Scheelevagen 17
22370 Lund
SWEDEN
EMail: ludwig@sics.se

Goeran Selander
Ericsson
Farogatan 6
16480 Kista
SWEDEN
EMail: goran.selander@ericsson.com

ACE Working Group
Internet-Draft
Intended Status: Informational
Expires: January 4, 2015

L. Seitz
SICS Swedish ICT
G. Selander
Ericsson

July 3, 2014

Problem Description for Authorization in Constrained Environments
draft-seitz-ace-problem-description-01

Abstract

We present a problem description for authentication and authorization in constrained-node networks, i.e. networks where some devices have severe constraints on memory, processing, power and communication bandwidth.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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".

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright and License 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	3
2. Background	3
3. Problem Description	5
3.1. Authorization	6
3.2. Authentication	7
3.3. Communication Security	7
3.4. Cryptographic Keys	8
4. Assumptions and Requirements	9
4.1 Architecture	9
4.2 Constrained Devices	10
4.3 Authorization	11
4.4 Authorization information	11
4.5 Access to authorization information	12
4.6 Resource access	12
4.7 Keys and cipher suites	13
4.8 Communication security paradigm	13
4.9 Network considerations	13
4.10 Legacy considerations	13
4.11 Open issues	14
5. Security Considerations	14
5.1 Physical attacks on sensor and actuator networks	14
5.2 Time measurements	15
6. IANA Considerations	16
7. Acknowledgements	16
8. References	16
8.1 Normative References	16
8.2 Informative References	17
Authors' Addresses	18

1. Introduction

Authorization is the process of deciding what an entity ought to be allowed to do. This memo is about properties of security protocols to enable explicit and dynamic authorization of clients to access a resource at a server, in particular in constrained environments when the client and/or server are constrained nodes.

Relevant use cases are provided in [I-D.seitz-ace-usecases], which also lists some requirements derived from the use cases. In this memo we present a more specific problem description for authentication and authorization in constrained RESTful environments together with a more detailed set of assumptions and requirements (cf. section 4).

1.1 Terminology

Certain security-related terms are to be understood in the sense defined in [RFC4949]. These terms include, but are not limited to, "authentication", "authorization", "confidentiality", "(data) integrity", "message authentication code", and "verify".

RESTful terms including "resource", "representation", etc. are to be understood as used in HTTP [RFC7231] and CoAP [RFC7252].

Terminology for constrained environments including "constrained device", "constrained-node network", "class 1", etc. are defined in [RFC7228].

"Explicit" authorization is used here to describe the ability to specify in some detail which entity has access to what and under what conditions, as opposed to "implicit" authorization where an entity is either allowed to access everything or nothing.

"Dynamic" authorization means that the access control policies and the parameters on which they are evaluated may change during normal operations, as opposed to "static" authorization meaning that access control policies cannot be changed during normal operations and may require some special procedure such as out-of-band provision.

2. Background

We assume a client-server setting, where a client wishes to access some resource hosted by a server. Such resources may e.g. be sensor data, configuration data, or actuator settings. Thus access to a resource could be by different methods, some of which change the state of the resource. In this memo, we consider the REST setting i.e. GET, POST, PUT and DELETE, and application protocols in scope

are HTTP [RFC7231] and CoAP [RFC7252].

We assume that the roles of client and server are not fixed, i.e. a node which is client could very well be server in some other context and vice-versa. Further we assume that in some cases, clients are not previously known to servers, thus we cannot assume that the server has access control policies specific to that client when the client initiates communication.

Finally we also assume that in a significant number of cases, the server and/or the client are too constrained to handle the authorization policies and related configuration on their own. Many authorization solutions involve a centralized, trusted third party, supporting the client and/or resource server. A trusted third party provides a more scalable way to centrally manage authorization policies, in order to ensure consistent authorization decisions. The physical separation of policy decision and policy enforcement is an established principle in policy based management, e.g. [RFC2748].

Borrowing from OAuth 2.0 [RFC6749] terminology we name the entities: client (C), resource server (RS), authorization server (AS - the third party), and resource owner (RO). RO does not need to be active in an constrained device access control setting, so interactions with the RO are out of scope for this memo. In the target setting RS is typically constrained, C may be constrained, whereas AS is not assumed to be constrained.

Since RS is constrained, we assume that it needs to offload authorization policy management and/or authorization decision making to AS. This means that some authorization information needs to be transferred from AS to RS. This information may for example be specific access control decisions such as "client C has the right to access this URI with this RESTful method, this payload value, under these local conditions", "client C has the right to access these URIs" or more indirect information "client C is in this access group". In the latter it is assumed that RS knows what rights are associated to a particular access group.

Protecting information carried between AS and RS, requires some a priori established cryptographic keys. How those keys are established is out of scope for this problem description. However, cryptographic keys that are used to protect information between AS and C are in scope: The reason being that dynamic access control is one of the use cases to be supported, and this may involve granting access to a client previously unknown to the server. An RS may have multiple trusted ASs corresponding to resources of different ROs, in which case it requires a key for each AS. This is a straightforward extension and is not further elaborated in this memo.

AS may for example be implemented as a cloud service, in a home server, or in a smartphone. C and RS may or may not have connectivity to AS (e.g. because AS is switched off), or may only have intermittent connectivity, where a connection at the time of access request cannot be guaranteed. Another reason for intermittent connectivity may be that constant connectivity is not affordable (e.g. due to limited battery power, or a sensor mobility business case for which cellular connectivity cost too much or is not available). Obviously, in order for a client request to reach RS there must be connectivity between C and RS, but that could be a short range technology such as Bluetooth, ZigBee, NFC, etc. Furthermore, if there is not sufficient authorization information about C in RS, and neither C nor RS can access AS, access requests will be denied. Therefore we assume that either C or RS can access AS at some point in time, prior to the client's request.

As a summary, there are potentially a number of information flows that needs to be secured:

- a. The transfer of authorization information from AS to RS
- b. The transfer of cryptographic keys or credentials from AS to RS and C, respectively
- c. The access request/response procedure between C and RS

3. Problem Description

From the background described in the previous section, we see the following problems that need to be solved in order to achieve explicit and dynamic authorization:

o Authorization

RS must have access to authorization information.

Given that the relevant information has been provided to RS, it must be able to handle an access request from C (match request against authorization information, grant or deny the request, and in the case of grant perform what is requested).

o Authentication

Some property of C needs to be authenticated to bind authorization information to it.

RS needs to establish the authenticity of authorization

information, and that it comes from a trusted AS.

Finally some property of RS needs to be authenticated to C, so that C can verify that it is communicating with the intended RS.

- o Communication Security

Communication security is needed to protect the integrity, and sometimes the secrecy of information flows between the parties. This includes the flow of authentication and authorization information, but also the actual request and response upon which access control is performed.

- o Key establishment

C and RS need to establish cryptographic keys in order to set up secure communications

Clearly, these problems are interconnected and need to take into account the involved constrained devices.

3.1. Authorization

The core problem we are trying to solve is authorization.

- o AS needs to transfer authorization information to RS. This can be done with or without involvement of C. In the case of C involvement there are three different message sequences: Agent, Pull or Push [RFC2904].
 - (i) In the agent sequence, C submits its request to AS and AS contacts RS to execute the request on C's behalf.
 - (ii) When using the pull sequence, C contacts RS and RS pulls authorization information directly from AS as a reaction to C's request (as e.g. in RADIUS [RFC2865]).
 - (iii) In the push sequence, C is used as intermediary between AS and RS, and authorization information is transferred in the form of some token (as e.g. in OAuth [RFC6749]).
- o What does the transferred authorization information contain and how should it be formatted/encoded? This must be efficient for constrained devices, considering size of authorization information and parser complexity.
- o How does RS verify the authenticity of the authorization information? There is a trade-off between processing complexity

and deployment complexity in using digital signatures with asymmetric keys or message authentication codes with symmetric keys.

- o How does RS enforce authorization decisions of AS? Does the authorization information it obtained from AS require additional policy evaluation (e.g. matching against local access control lists, evaluating local conditions)? What kind of "policy evaluation" can we assume a constrained device to be capable of?
- o Finally, as is indicated in the previous bullet, for a particular authorization decision there may be different kinds of authorization information needed, and these pieces of information may be transferred to RS at different times and in different ways prior to or during the client request. For example, local access control lists for particular access groups may be pushed from AS to RS without involvement of C at regular intervals, whereas an assertion of group membership (client attribute) of a particular C can be pushed involving C as in (iii) above.

3.2. Authentication

The following problems need to be addressed, when considering authentication:

- o RS needs to authenticate some property of C, in order to bind it to the relevant authorization information. This could e.g. be a digital signature or a message authentication codes performed by C where a corresponding cryptographic key is contained in the authorization information.
- o In many use cases C wants to authenticate RS, in order to ensure that it is interacting with the right resources.
- o AS needs to authenticate its communication partner (either C or RS), in order to ensure it serves the correct device.
- o Since AS has a trust relation to both C and RS, it could also provide them with the means of mutual authentication (similar to a Kerberos [RFC4120] server). This would make it possible for RS to authenticate previously unknown clients.

3.3. Communication Security

There are different alternatives to provide communication security.

- o One is session-based security at transport layer such as DTLS [RFC6347], which offers security, including integrity and confidentiality protection, for the whole application layer exchange. One cost of DTLS is the handshake protocol, which may be expensive for constrained devices especially in terms of memory and power consumption for message transmissions.
- o An alternative is data object security at application layer, e.g. using JWE [I-D.ietf-jose-json-web-encryption]. Secure objects can be stored or cached in network nodes and provide security for a more flexible communication model such as publish/subscribe (compare e.g. CoRE Mirror Server [I-D.vial-core-mirror-proxy]). However, data object security only may not provide confidentiality for the message headers. For example, information such as the RESTful method, the host address, and the resource URI may be revealed.
- o A solution to the overall authorization problem may be based on session-based security only, data object security only or a hybrid. An example of a hybrid is where authorization information and cryptographic keys are provided by AS in the format of secure data objects, but where the resource access is protected by session-based security. (For secure objects containing authorization information, compare e.g. OAuth 2.0 MAC Tokens [I-D.ietf-oauth-v2-http-mac].)
- o A hybrid solution may also be useful to support a flexible trust model, e.g. a resource representation wrapped end-to-end in JWE sent over DTLS hop-by-hop in a case where an intermediary is allowed to read the header but not the payload.
- o A detailed analysis how different use cases benefit from different communication security paradigms is beyond the scope of this memo. Current Internet standards support both approaches, and this should be possible to leverage also in constrained environments.

3.4. Cryptographic Keys

With respect to cryptographic keys, we see the following problems that need to be addressed:

- o Symmetric vs Asymmetric Keys

Do we want to support solutions based on asymmetric keys or symmetric keys, or both? The question applies both to protection of resource access and to protection of

authentication and authorization information.

There are classes of devices that can easily perform symmetric cryptography, but consume considerably more time/battery for asymmetric operations. On the other hand asymmetric cryptography has benefits e.g. in terms of deployment.

- o Key Establishment

How are the corresponding cryptographic keys established? Considering section 3.1 there must be a binding between these keys and the authorization information, at least in the sense that AS must be able to specify a unique client identifier which RS can verify (using an associated key).

One of the use cases of [I-D.seitz-ace-usecases] describes spontaneous change of access policies - e.g. giving a hitherto unknown client the right to temporarily unlock your house door. In this case C is not previously known to RS and a key must be provisioned by AS.

- o Revocation and Expiration

How are keys replaced and how is a key that has been compromised revoked in a manner that reaches all affected parties, also keeping in mind scenarios with intermittent connectivity?

4. Assumptions and Requirements

In this section we list a set of candidate assumptions and requirements to make the problem description in the previous sections more concise and precise.

4.1 Architecture

The architecture consists of at least the following types of nodes:

- o RS hosting resources, and responding to access requests
- o C requesting access to resources
- o AS supporting the access request/response procedure by providing authorization information to RS.
 - AS may also provide other services such as authenticating C on behalf of RS, or providing cryptographic keys or

credentials to C and/or RS to secure the request/response procedure.

- o The architecture may contain intermediary nodes between any pair of C, RS and AS, such as e.g. translation/reverse proxies in the CoRE architecture. The solution shall not unduly restrict the use of intermediaries.

4.2 Constrained Devices

- o C and/or RS may be constrained in terms of power, processing, communication bandwidth, memory and storage space, and moreover
 - unable to manage complex authorization policies
 - unable to manage a large number of secure connections
 - without user interface
 - without constant network connectivity
 - unable to precisely measure time
 - required to save on wireless communication due to high power consumption
- o C and RS may be class 1 (potentially with large effort) or more powerful devices.
- o AS is not a constrained device.
- o All devices can process symmetric cryptography without incurring an excessive performance penalty.
 - We assume the use of a standardized symmetric key algorithm, such as AES.
 - Except for the most constrained devices we assume the use of a standardized cryptographic hash function such as SHA-256.
- o Public key cryptography requires additional resources (e.g. RAM, ROM, power).
- o A DTLS handshake with public key cryptography involves significant computation, communication, and memory overheads in the context of constrained devices.

- The RAM requirements of DTLS handshake with public key cryptography may be prohibitive for constrained devices.
- Certificate-based DTLS handshake requires extensive resources e.g. in terms of ROM.
- o The solution shall support a simple scheme for expiring authentication and authorization information on devices which are unable to measure time (cf. section 5.2).

4.3 Authorization

- o The authorization decision may be based on credentials presented by C, resource, RESTful method and local context in RS at the time of the request.
- o The authorization decision may be taken either by AS or RS.
- o The authorization decision is enforced by RS.
 - RS needs to have access to authorization information in order to verify that C is allowed to access the resource as requested.
 - RS needs to make sure that it provides resource access only to authorized clients.
- o Apart from authorization for access to a resource, authorization may also be required for access to information about a resource.
- o The solution may be able to support authorizing the delegation of access rights.

4.4 Authorization information

- o Authorization information is information that allows RS to verify that a requesting C is authorized.
- o Authorization information includes self-contained information such as authorization decisions or client capability lists which allows RS to directly match against a request.
- o Authorization information includes also such information that RS may need to combine, in order to verify that a requesting C is authorized, including client attributes and authorization policies (e.g. access control lists) based on client attributes.

4.5 Access to authorization information

- o Authorization information may a priori be transferred directly between AS and RS, or via C; using Agent, Push or Pull mechanisms [RFC2904].
- o RS shall authenticate that the authorization information is coming from AS.
- o The authorization information may also be encrypted end-to-end between AS and RS.
- o RS may not be able to communicate with AS at the time of the request from C.
- o RS may store or cache authorization information.
- o Authorization information may be pre-configured in RS.
- o Authorization information stored or cached in RS shall be possible to change. The change of such information shall be subject to authorization.
- o Authorization policies stored on RS may be handled as a resource, i.e. information located at a particular URI, accessed with RESTful methods, and the access being subject to the same authorization mechanics. AS may have special privileges when requesting access to the authorization policy resources on RS.
- o There may be mechanisms for C to look up the AS which provides authorization information about a particular resource.

4.6 Resource access

- o Resources are accessed in a RESTful manner using GET, PUT, POST, DELETE.
- o By default, the resource request shall be integrity protected and may be encrypted end-to-end from C to RS. It shall be possible for RS to detect a replayed request. (DTLS supports this.)
- o By default, the response to a request shall be integrity protected and may be encrypted end-to-end from RS to C. (DTLS supports this.)
- o By default, C shall be able to verify that the response to a request comes from the intended RS. (DTLS supports this.)

- o There may be resources whose access need not be protected (e.g. for discovery of the responsible AS).

4.7 Keys and cipher suites

- o AS and RS have established cryptographic keys. Either AS and RS share a secret key or each have the other's public key.
- o The access request/response may be protected with symmetric and/or asymmetric keys.
- o The transfer of authorization information is protected with symmetric and/or asymmetric keys.
- o There may be a mechanism for C to look up the supported cipher suites of a RS.

4.8 Communication security paradigm

- o The solution shall support session based security and/or data object security.

4.9 Network considerations

- o The solution shall prevent network overload due to avoidable communication with AS.
- o The solution shall prevent network overload by compact authorization information representation.
- o The solution shall optimize the case where authorization information does not change often.
- o The solution where possible shall support an efficient mechanism for providing authorization information to multiple RSs, for example when multiple entities need to be configured or change state.

4.10 Legacy considerations

- o The solution shall work with existing infrastructure.
- o The solution shall support authorization of access to legacy devices.

4.11 Open issues

The section lists some known open issues

- o What is the level of functionality that can be achieved with class 1 devices
 - with off-the-shelf software?
 - using heroic efforts?
- o Is authorization for network access in scope?
- o Should the model of draft-gerdes-ace-actors [I-D.gerdes-ace-actors] (in particular the Authorization Manager) be included in the default architecture?
- o Should the requirements include cross-domain authorization?

5. Security Considerations

The entire document is about security. Security considerations applicable to authentication and authorization in RESTful environments are provided in e.g. OAuth 2.0 [RFC6749].

In this section we focus on specific security aspects related to authorization in constrained-node networks.

5.1 Physical attacks on sensor and actuator networks

The focus of this work is on constrained-node networks consisting of connected sensors and actuators. The main function of such devices is to interact with the physical world by gathering information or performing an action. We now discuss attacks performed with physical access to such devices.

The main threats to sensors and actuator networks are:

- o Unauthorized access to data to and from sensors and actuators, including eavesdropping and manipulation of data.
- o Denial-of-service making the sensor/actuator unable to perform its intended task correctly.

A number of attacks can be made with physical access to a device including probing attacks, timing attacks, power attacks, etc. However, with physical access to a sensor or actuator device it is possible to directly perform attacks equivalent of eavesdropping,

manipulating data or denial of service. For example:

- o Instead of eavesdropping the sensor data or attacking the authorization system to gain access to the data, the attacker could make its own measurements on the physical object.
- o Instead of manipulating the sensor data the attacker could change the physical object which the sensor is measuring, thereby changing the payload data which is being sent.
- o Instead of manipulating data for an actuator or attacking the authorization system, the attacker could perform an unauthorized action directly on the physical object.
- o A denial-of-service attack could be performed physically on the object or device.

All these attacks are possible by having physical access to the device, since the assets are related to the physical world. Moreover, this kind of attacks are in many cases straightforward (requires no special competence or tools, low cost given physical access, etc.)

As a conclusion, if an attacker has physical access to a sensor or actuator device, then much of the security functionality elaborated in this draft is not effective to protect the asset during the physical attack.

Since it does not make sense to design a solution for a situation that cannot be protected against we assume there is no need to protect assets which are exposed during a physical attack. In other words, either an attacker does not have physical access to the sensor or actuator device, or if it has, the attack shall only have effect during the period of physical attack.

5.2 Time measurements

Measuring time with certain accuracy is important to achieve certain security properties, for example to determine whether a public key certificate, access token or some other assertion is valid.

Dynamic authorization in itself requires the ability to handle expiry or revocation of authorization decisions or to distinguish new authorization decisions from old.

For certain categories of devices we can assume that there is an internal clock which is sufficiently accurate to handle the time

measurement requirements. If RS can connect directly to AS it could get updated in terms of time as well as revocation information.

If RS continuously measures time but can't connect to AS or other trusted source, time drift may have to be accepted and it may not be able to manage revocation. However, it may still be able to handle short lived access rights within some margins, by measuring the time since arrival of authorization information or request.

Some categories of devices in scope may be unable measure time with any accuracy (e.g. because of sleep cycles). This category of devices is not suitable for the use cases which require measuring validity of assertions and authorizations in terms of absolute time.

However there are simpler schemes to grant certain temporary access requests in a secure manner. For example, one-time authorization grants based on some freshness maintained between AS and RS such as sequence numbers or nonces. For the convenience of the reader we now outline a very simplistic scheme. AS may keep a counter for each RS, step the counter for each time it generates new authorization information and include the counter in the authorization information. RS accepts as fresh authorization information with a higher counter compared to highest previously received counter value. If the authorization information is fresh, RS grants the associated access request and replaces the old counter value with the new. The security considerations of this scheme is out of scope for this memo.

6. IANA Considerations

This document has no actions for IANA.

7. Acknowledgements

The authors would like to thank Carsten Bormann, Stefanie Gerdes, Sandeep Kumar, John Mattson, Corinna Schmitt, Mohit Sethi, Hannes Tschofenig, Vlasios Tsiatsis and Erik Wahlstroem for contributing to the discussion, giving helpful input and commenting on the 00-version. The authors would also like to acknowledge input provided by draft-gerdes-ace-actors[I-D.gerdes-ace-actors] and Hummen et al [HUM14delegation].

8. References

8.1 Normative References

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI

36, RFC 4949, August 2007.

- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC7231] Fielding, R., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, June 2014.
- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.
- [RFC7252] Shelby, Z., Hartke K., and C. Bohrmann, "The Constrained Application Protocol (CoAP)", RFC7252, June 2014

8.2 Informative References

- [I-D.seitz-ace-usecases]
Seitz, L., Gerdes, S., and Selander, G., "ACE use cases", draft-seitz-ace-usecases (work in progress), February 2014.
- [I-D.ietf-jose-json-web-encryption]
Jones, M., Hildebrand, J., "JSON Web Encryption (JWE)", draft-ietf-jose-json-web-encryption (work in progress), April 2014.
- [I-D.vial-core-mirror-proxy]
Vial, M., "CoRE Mirror Server", draft-vial-core-mirror-proxy (expired), July 2012.
- [I-D.ietf-oauth-v2-http-mac]
Richter, J., Mills, W., Tschofenig, H. (Ed.), and P. Hunt, "OAuth 2.0 Message Authentication Code (MAC) Tokens", draft-ietf-oauth-v2-http-mac (work in progress), January 2014.
- [I-D.gerdes-ace-actors]
Gerdes, S., "Actors in the ACE Architecture", draft-gerdes-ace-actors-00 (work in progress), May 2014.
- [HUM14delegation] Hummen, R., Shafagh, H., Raza, S., Voigt, T., Wehrle, K., "Delegation-based Authentication and Authorization for the IP-based Internet of Things", 11th IEEE International Conference on Sensing, Communication, and Networking (SECON'14), June 30 - July 3, 2014.
- [RFC2748] Durham, D., Ed., Boyle, J., Cohen, R., Herzog, S., Rajan,

R., and A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000.

- [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.
- [RFC2904] Vollbrecht, J., Calhoun, P., Farrell, S., Gommans, L., Gross, G., de Bruijn, B., de Laat, C., Holdrege, M., and D. Spence, "AAA Authorization Framework", RFC 2904, August 2000.
- [RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.
- [RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, May 2014.

Authors' Addresses

Ludwig Seitz
SICS Swedish ICT AB
Scheelevagen 17
22370 Lund
SWEDEN
EMail: ludwig@sics.se

Goeran Selander
Ericsson
Farogatan 6
16480 Kista
SWEDEN
EMail: goran.selander@ericsson.com

ACE Working Group
Internet-Draft
Intended status: Informational
Expires: January 3, 2015

L. Seitz, Ed.
SICS Swedish ICT AB
S. Gerdes, Ed.
Universitaet Bremen TZI
G. Selander
Ericsson
M. Mani
Itron
S. Kumar
Philips Research
July 02, 2014

ACE use cases
draft-seitz-ace-usecases-01

Abstract

This document presents use cases for authentication and access control in scenarios involving constrained RESTful devices. Where specific details are relevant, it is assumed that the devices use CoAP as communication protocol, however most conclusions apply generally.

A number of security requirements are derived from the use cases, which are intended as a guideline for developing a comprehensive authentication and access control approach for this class of scenarios.

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 January 3, 2015.

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	3
2. Use Cases	4
2.1. Container monitoring	4
2.1.1. Bananas for Munich	4
2.1.2. Requirements	5
2.2. Home Automation	5
2.2.1. Remotely letting in a visitor	6
2.2.2. Requirements	6
2.3. Personal Health Monitoring	7
2.3.1. John and the heart rate monitor	7
2.3.2. Requirements	8
2.4. Building Automation	9
2.4.1. Device Lifecycle	9
2.4.2. Requirements	11
2.5. Smart Metering	12
2.5.1. Drive-by metering	12
2.5.2. Meshed Topology	13
2.5.3. Customer Direct Access to the metering Data	13
2.5.4. Requirements	13
3. Consolidated Requirements From The Use Cases	14
3.1. General Security Requirements	14
3.2. Authentication Requirements	15
3.3. Access Control Requirements	15
4. Security Considerations	17
5. Acknowledgments	18
6. IANA Considerations	19
7. Informative References	20
Authors' Addresses	21

1. Introduction

This document presents use cases in an attempt to analyze the authentication and access control requirements in an Internet of Things setting. This setting features constrained devices [RFC7228] communicating over the Internet.

Some of these devices may have very low capacity in terms of memory and processing power, and may additionally be limited by the fact that they run on battery power.

These devices offer resources such as sensor data and actuators, which are accessed by clients, sometimes without human intervention (M2M). In some situations the communication will happen through intermediaries (e.g. gateways, proxies).

Where specific detail is necessary it is assumed that the devices communicate using the CoAP protocol [RFC7252], although most conclusions are generic.

1.1. Terminology

Resource Server (RS): The constrained device which hosts resources the Client wants to access.

Client (C): A device which wants to access a resource on the Resource Server.

This could also be a constrained device.

Resource Owner (RO): The subject who owns the resource and controls its access permissions.

2. Use Cases

This section lists use cases involving constrained devices with security requirements. Each use case first presents a general description of the application area, then one or more specific use cases, and finally the resulting requirements. We assume that basic communication security requirements apply for all of these scenarios.

2.1. Container monitoring

The ability of sensors to communicate environmental data wirelessly opens up new application areas. The use of such sensor systems makes it possible to continuously track and transmit specific characteristics such as temperature, humidity and gas content during the transportation and storage of goods.

The proper handling of the sensors in this scenario is not easy to accomplish. They have to be associated to the appropriate pallet of the respective container. Moreover, the goods and the corresponding sensors belong to specific customers.

During the shipment to their destination the goods often pass stops where they are transloaded to other means of transportation, e.g. from ship transport to road transport.

The transportation and storage of perishable goods is especially challenging since they have to be stored at a constant temperature and with proper ventilation. Additionally, it is very important for the vendors to be informed about irregularities in the temperature and ventilation of fruits to avoid the delivery of decomposed fruits to their customers. The need for a constant monitoring of perishable goods has led to projects such as The Intelligent Container (<http://www.intelligentcontainer.com>).

2.1.1. Bananas for Munich

A fruit vendor grows bananas in Costa Rica for the German market. It instructs a transport company to deliver the goods via ship to Rotterdam where they are picked up by trucks and transported to a ripening facility. A Munich supermarket chain buys ripened bananas from the fruit vendor and transports them with their own company trucks.

The fruit vendor's quality management wants to assure the quality of their products and thus equips the banana boxes with sensors. The state of the goods is monitored consistently during shipment and ripening and abnormal sensor values are recorded. Additionally, the sensor values are used to control the climate within the cargo

containers.

The personnel that transloads the goods must be able to locate the goods meant for a specific customer. However the fruit vendor does not want to disclose sensor information pertaining to the condition of the goods to other companies.

When the goods arrive at the supermarket in Munich, the supermarket conducts its own quality check. If no anomalies occurred during the transport, the bananas are admitted for sale.

2.1.2. Requirements

- o U1.1 The fruit vendor must be able to allow the transport company and the delivery service to access the position data on the monitoring devices. Other state information must not be accessible.
- o U1.2 The climate regulation system in the containers must be able to access the monitoring devices' state information to regulate the climate accordingly, without manual intervention of the resource owner.
- o U1.3 The fruit vendor must be able to allow the fruit vendor's quality management to access the recorded state information on the monitoring devices.
- o U1.4 Since the fruit vendor does not want other companies to be able to read sensor information, there should be some access control for the monitoring devices' state information.

2.2. Home Automation

Automation of the home has the potential to become a big future market for the Internet of Things. A home automation system connects electrical devices in a house to the Internet and thus makes them accessible and manageable remotely. Such devices might control for example heating, ventilation, lighting, home entertainment or home security.

Such a system needs to accommodate a number of regular users (inhabitants, close friends, cleaning personnel) as well as a heterogeneous group of dynamically varying users (visitors, repairmen, delivery men).

The security required by the systems in a automated home varies, however it is clear that the security system controlling e.g. the door-locks and alarms needs to be at least as secure as for a

comparable unautomated home.

As the users are not typically trained in security (or even computer use), the configuration must use secure default settings, and the interface must be well adapted to novice users.

2.2.1. Remotely letting in a visitor

Jane is the owner of a flat with automated connected door-locks and alarm system, that allow her to remotely control them through a web interface or mobile application. To allow for centralized management of both locks and the alarm system, they need to be able to communicate with both the web interface and the mobile application using a standardized, secure protocol.

Jane has invited her acquaintance Jeffrey over for dinner, but is stuck in traffic and can not arrive in time, while Jeffrey who uses the subway will arrive punctually. Jane calls Jeffrey and offers him to let him in remotely, so he can make himself comfortable while waiting.

Jeffrey downloads an application that lets him communicate with Jane's door-lock and alarm system. Then Jane sets permissions for Jeffrey that allow him to open the door, and shut down the alarm when he arrives.

2.2.2. Requirements

- o U2.1 Jane needs to be able to spontaneously provision authentication means to Jeffrey.
- o U2.2 Jane must be able to spontaneously change the access control policies.
- o U2.3 Jane needs to be able to apply different rights for different users.
- o U2.4 Jane must be able to apply context-based conditions (presence, time) to authorizations, and the devices (door-lock or alarm) need to be able to verify these conditions.
- o U2.5 The security mechanisms of the door-lock and the alarm in Jane's home need to be able to communicate with different control devices (e.g. Jeffrey's mobile phone).
- o U2.6 The access control configuration of Jane's home needs to be secure by default.

- o U2.7 It must be easy for Jane to edit the access control policies for her home, even remotely and easy for Jeffrey to get access with correct authorization.

2.3. Personal Health Monitoring

The use of wearable health monitoring technology is expected to grow strongly, as a multitude of novel devices are developed and marketed. The need for open industry standards to ensure interoperability between products has lead to initiatives such as Continua Alliance (continuaalliance.org) and Personal Connected Health Alliance (pchalliance.org). Personal health devices are typically battery driven, and located physically on the user. They monitor some bodily function, such as e.g. temperature, blood pressure, or pulse. They are connected to the Internet through an intermediary base-station, using wireless technologies. Through this connection they report the monitored data to some entity, which may either be the user herself, or some medical personnel in charge of the user.

Medical data has always been considered as very sensitive, and therefore requires good protection against unauthorized disclosure. A frequent, conflicting requirement is the capability for medical personnel to gain emergency access, even if no specific access rights exist. As a result, the importance of secure audit logs increases in such scenarios.

Since the users are not typically trained in security (or even computer use), the configuration must use secure default settings, and the interface must be well adapted to novice users. Parts of the system must operate with minimal maintenance. Especially frequent changes of battery are unacceptable.

2.3.1. John and the heart rate monitor

John has a heart condition, that can result in sudden cardiac arrests. He therefore uses a device called HeartGuard that monitors his heart rate and his position. In case of a cardiac arrest it automatically sends an alarm to an emergency service, transmitting John's current location. The HeartGuard also broadcasts emergency information in the neighborhood to notify doctors or people with certain skills who have been enrolled in an emergency program, e.g. people who got training in heart and lung rescue. For doctors, medical information or diagnosis can be provided with the notification to improve immediate treatment.

The device includes some smart logic, with which it identifies its owner John and allows him to configure the device's settings, including access control.

This prevents situation where someone else wearing that device can act as the owner and mess up the access control and security settings.

John can configure additional persons that get notified in an emergency, for example his daughter Jill. Furthermore the device stores data on John's heart rate, which can later be accessed by a physician to assess the condition of John's heart.

However John is a rather private person, and is worried that Jill might use HeartGuard to monitor his location while there is no emergency. Furthermore he doesn't want his health insurance to get access to the HeartGuard data, or even to the fact that he is wearing a HeartGuard, since they might refuse to renew his insurance if they decided he was too big a risk for them.

NOTE: Monitoring of some state parameter (e.g. an alarm button) and the position of a person also fits well into an elderly care service. This is particularly useful for people suffering from dementia, where the relatives or caregivers need to be notified of the whereabouts of the person under certain conditions. In this case it is not the patient that decides about access.

2.3.2. Requirements

- o U3.1 John must be able to pre-configure access rights to the position data for persons or groups, in the context of an emergency.
- o U3.2 John must be able to selectively allow different persons or groups to access the heart rate data.
- o U3.3 John must be able to block access to specific persons in an otherwise allowed group (e.g. doctors in an emergency), if he mistrusts them.
- o U3.4 The security measures must consider the battery lifetime of the devices and should consume as little energy as possible.
- o U3.5 The device must have secure access control settings by default.
- o U3.6 The device's access control settings must be easy to configure for an authorized, non-technical user.
- o U3.7 Security mechanisms on medical devices must not provide opportunities for denial of service attacks on the device.

2.4. Building Automation

Buildings for commercial use such as shopping malls or office buildings nowadays are equipped increasingly with semi-automatic components to enhance the overall living quality and to save energy where possible. This includes for example heating, ventilation and air condition (HVAC) as well as illumination and security systems such as fire alarms.

Different areas of these buildings are often exclusively leased to different companies. However they also share some of the common areas of the building.

Accordingly, a company must be able to control the light and HVAC system of its own part of the building and must not have access to control rooms that belong to other companies.

Some parts of the building automation system such as entrance illumination and fire alarm systems are controlled either by all parties together or by a service company.

2.4.1. Device Lifecycle

2.4.1.1. Installation and Commissioning

A building is hired out to different companies for office space. This building features various automated systems, such as a fire alarm system, which is triggered by several smoke detectors which are spread out across the building. It also has automated HVAC, lighting and physical access control systems.

A vacant area of the building has been recently leased to company A. Before moving into its new office, Company A wishes to replace the lighting with a more energy efficient and a better light quality luminaries. They hire an installation and commissioning company C to redo the illumination. Company C is instructed to integrate the new lighting devices, which may be from multiple manufacturers, into the existing lighting infrastructure of the building which includes presence sensors, switches, controllers etc.

Company C gets the necessary authorization from the service company to interact with the existing Building and Lighting Management System (BLMS).

To prevent disturbance to other occupants of the building, Company C is provided authorization to perform the commissioning only during non-office hours and only to modify configuration on devices belonging to the domain of Company A's space. After installation (wiring) of the new lighting devices, the commissioner adds the devices into the company A's lighting domain.

Once the devices are in the correct domain, the commissioner authorizes the interaction rules between the new lighting devices and existing devices like presence sensors. For this, the commissioner creates the authorization rules on the BLMS which define which lights form a group and which sensors /switches/controllers are allowed to control which groups. These authorization rules may be context based like time of the day (office or non-office hours) or location of the handheld lighting controller etc.

2.4.1.2. Operational

Company A's staff move into the newly furnished office space. Most lighting is controlled by presence sensors which control the lighting of specific group of lights based on the authorization rules in the BLMS. Additionally employees are allowed to manually override the lighting brightness and color in their office by using the switches or handheld controllers. Such changes are allowed only if the authorization rules exist in the BLMS. For example lighting in the corridors may not be manually adjustable.

At the end of the day, lighting is dimmed down or switched off if no occupancy is detected even if manually overridden during the day.

On a later date company B also moves into the same building, and shares some of the common spaces with company A. On a really hot day James who works for company A turns on the air condition in his office. Lucy who works for company B wants to make tea using an electric kettle. After she turned it on she goes outside to talk to a colleague until the water is boiling. Unfortunately, her kettle has a malfunction which causes overheating and results in a smoldering fire of the kettle's plastic case.

Due to the smoke coming from the kettle the fire alarm is triggered. Alarm sirens throughout the building are notified and alert the staff of both companies. Additionally, the ventilation system of the whole building is closed off to prevent the smoke from spreading and to withdraw oxygen from the fire. The smoke cannot get into James' office although he turned on his air condition because the fire alarm overrides the manual setting.

The fire department is notified of the fire automatically and arrives within a short time. After inspecting the damage and extinguishing the smoldering fire a fire fighter resets the fire alarm because only the fire department is authorized to do that.

2.4.1.3. Maintenance

Company A's staff are annoyed that the lights switch off too often in their rooms if they work silently in front of their computer. Company A notifies the commissioning Company C about the issue and asks them to increase the delay before lights switch off.

Company C again gets the necessary authorization from the service company to interact with the BLMS. The commissioner's tool gets the necessary authorization from BMLS to send a configuration change to all lighting devices in Company A's offices to increase their delay before they switch off.

2.4.1.4. Decommissioning

Company A has noticed that the handheld controllers are often misplaced and hard to find when needed. So most of the time staff use the existing wall switches for manual control. Company A decides it would be better to completely remove handheld controllers and asks Company C to decommission them from the lighting system.

Company C again gets the necessary authorization from the service company to interact with the BLMS. The commissioner now deletes any rules that allowed handheld controllers authorization to control the lighting. Additionally the commissioner instructs the BLMS to push these new rules to prevent cached rules at the end devices from being used.

2.4.2. Requirements

- o U4.1. A user with sufficient authorization to a device should be able to transfer the device to a different authorization server.
- o U4.2. Authorization rules may be context-based.
- o U4.3. Devices can access resources on other devices only if a rule exists in the authorization server (default deny).
- o U4.4 Devices can be authorized to control individual devices using unicast or multiple devices using multicast.
- o U4.5. Devices may cache authorization rules locally.
- o U4.6. Subsystems under different operational domains must be able to interoperate with each other if the domain owners agree.
- o U4.7. A user with sufficient authorization to a device should be able to remove the device from an authorization server.

- o U4.8. Authorization server may have a mechanism to override locally cached rules at devices.
- o U4.9. Revocation of security credentials should be possible.

2.5. Smart Metering

Automated measuring of customer consumption is an established technology for electricity, water, and gas providers. Increasingly these systems also feature networking capability to allow for remote management. Such systems are in use for commercial, industrial and residential customers and require a certain level of security, in order to avoid economic loss to the providers, vulnerability of the distribution system, as well as disruption of services for the customers.

The smart metering equipment for gas and water solutions is battery driven and communication should be used sparingly due to battery consumption. Therefore the types of meters sleep most of the time, and only wake up every minute/hour to check for incoming instructions. Furthermore they wake up a few times a day (based on their configuration) to upload their measured metering data.

Different networking topologies exist for smart metering solutions. Based on environment, regulatory rules and expected cost, one or a mixture of these topologies may be deployed to collect the metering information. Drive-By metering is one of the most current solutions deployed for collection of gas and water meters.

2.5.1. Drive-by metering

A company offers smart metering infrastructures and related services to various providers. Among these is a water provider, who in turn supplies several residential complexes in a city. The smart meters are installed in the end customer's homes to measure water consumption and thus generate billing data for the provider. The meters do so by sending data to a base station. Several base stations are installed around the city to collect the metering data. However in the denser urban areas, the base stations would have to be installed very close to the meters. This would require a high number of base stations and expose this more expensive equipment to manipulation or sabotage. The company has therefore chosen another approach, which is to drive around with a mobile base-station and let the meters connect to that in regular intervals in order to gather metering data.

2.5.2. Meshed Topology

In another deployment, the water meters are installed in a building that already has power meters installed, the latter are mains powered, and are therefore not subject to the same power saving restrictions. The water meters can therefore use the power meters as proxies, in order to achieve better connectivity. This requires the security measures on the water meters to work through intermediaries.

2.5.3. Customer Direct Access to the metering Data

The provider also wishes to offer its customer limited access to some of the data on the metering devices, in order to allow them to check and optimize their consumption. However the provider expects the company to implement measures to prevent tampering with the data relevant for billing.

2.5.4. Requirements

- o U5.1 If security information can be recovered by a physical attack on a meter, this information must not be usable in an attack on other parts of the metering infrastructure.
- o U5.2 The meters must be able to perform fine-grained access control on the metering data and on the configuration while being offline.
- o U5.3 Authentication and access control must function without online connection to a back-end server.
- o U5.4 Since there are many smart meters deployed and reaching them is difficult, authentication and access control policy updates must not depend on directly (or worse manually) provisioning these updates to individual meters.
- o U5.5 The authentication and access control measures must cope with the presence of intermediary proxies between the Resource Servers and the Client.

3. Consolidated Requirements From The Use Cases

This section consolidates the requirements derived from the use cases above. Note that not every single requirement applies to every Resource Server, however protocols should allow for all of these requirements to be fulfilled.

3.1. General Security Requirements

The following requirements refer to general security measures that are affected by the design of authentication and access control protocols.

- o Protect the Resource Server against denial of service (U3.7)
 - * Minimize the number of protocol steps that an attacker can induce a Resource Server to perform without proper authentication and access control.
 - * Note well that for constrained devices this includes attacks that aim to drain the battery of the target.
- o Authentication and access control measures must work when traffic from the Client to the Resource Server goes through intermediary nodes. (U5.5)

Rationale: In many deployments, there will be gateways, proxies, firewalls etc. between a Client and a Resource Server. This means that e.g. DTLS [RFC6347] client authentication can not be used to authenticate the Client.

- o Minimize resource usage for authentication and access control on the constrained device(s) (U3.4)
 - * Minimize battery usage
 - + Minimize message exchanges required by security measures
 - + Minimize the size of authentication and access control data that is transmitted
 - + Minimize the size of code required and reuse existing code libraries
 - + Minimize memory and stack usage on the devices

- o Require secure default settings (U1.4, U2.6, U3.5, U4.3)

Rationale: Many attacks exploit insecure default settings, and experience shows that default settings are frequently left unchanged by the end users. Therefore the security protocols for constrained devices should require secure modes of use by default.

- o Interoperability (U1.1, U2.5, U4.6)

Rationale: Resource Owners may interact with Clients from various manufacturers and vice-versa. For the overall system to function correctly the authentication and access control mechanisms need to work consistently. This is best achieved by standardization.

- o Usability (U2.7, U3.6)

- * Keep response times reasonable
- * Make authentication and access control transparent for human users where possible
- * Make the administration of authentication and access control as simple as possible

3.2. Authentication Requirements

- o Standardized provisioning of authentication means to Clients and Resource Servers (U2.1, U4.1, U4.7)
 - * Allow for remote provisioning as an option
- o Enable remote revocation of authentication means (U4.9, U5.4)

3.3. Access Control Requirements

- o Enforce the access control policies of the Resource Owner (all use cases)
 - * Provision of access control policies set by the Resource Owner to the Policy Decision Point [RFC2904] (which may be on the Resource Server or on another trusted entity).
 - * Apply the access control policies to incoming requests (this may be done by the Resource Server or by another trusted entity).

- o Allow for different rights to the same resource for different requesting entities (U1.1, U1.2, U2.3, U3.1, U3.2, U3.3, U5.2)

Rationale: In some cases different types of users require different access rights, as opposed to all-or-nothing access control.

- o Allow for fine-grained access control (U1.1, U1.2, U3.1, U3.2, U5.2) Resource Servers can host several resources, and a resource (e.g. an actuator) can have different settings. In some cases access rights need to be different at this level of granularity.
- o Support access control on multicast requests to several Resource Servers (U4.4)
- o Access control must work when the Resource Server has intermittent connectivity (U4.5)
- o The Resource Server should be able to evaluate context-based permissions (U2.4, U3.1, U4.2)

Access may depend on local conditions e.g. access to health data in an emergency. The Policy Decision Point must be able to take such conditions into account.

- o Enable policy updates without re-provisioning individual devices (U2.2, U4.7, U4.8, U5.4)

Rationale: Clients can change rapidly and re-provisioning might be prohibitively expensive.

- o Do not require manual intervention of the Resource Owner in the access control process (U1.2, U3.1, U5.4).

Rationale: Manually approving access requests, while being a common solution in web access control, does not scale well in an M2M scenario.

- o Enable revocation of authorizations, also considering locally cached authorization information (U4.9)

4. Security Considerations

This document lists security requirements for constrained devices, motivated by specific use cases. Therefore the whole document deals with security considerations.

5. Acknowledgments

The authors would like to thank Olaf Bergmann, Sumit Singhal, John Mattson, Mohit Sethi, Carsten Bormann, Corinna Schmitt, Hannes Tschofenig, and Erik Wahlstroem for reviewing and/or contributing to the document. Also, thanks to Markus Becker, Thomas Poetsch and Koojana Kuladinithi for their input on the container monitoring use case."

6. IANA Considerations

This document has no IANA actions.

7. Informative References

- [RFC2904] Vollbrecht, J., Calhoun, P., Farrell, S., Gommans, L., Gross, G., de Bruijn, B., de Laat, C., Holdrege, M., and D. Spence, "AAA Authorization Framework", RFC 2904, August 2000.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, May 2014.
- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.

Authors' Addresses

Ludwig Seitz (editor)
SICS Swedish ICT AB
Scheelevaegen 17
Lund 223 70
Sweden

Email: ludwig@sics.se

Stefanie Gerdes (editor)
Universitaet Bremen TZI
Postfach 330440
Bremen 28359
Germany

Phone: +49-421-218-63906
Email: gerdes@tzi.org

Goeran Selander
Ericsson
Faroegatan 6
Kista 164 80
Sweden

Email: goran.selander@ericsson.com

Mehdi Mani
Itron
52, rue Camille Desmoulins
Issy-les-Moulineaux 92130
France

Email: Mehdi.Mani@itron.com

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

Email: sandeep.kumar@philips.com

Network Working Group
Internet-Draft
Intended status: Best Current Practice
Expires: August 18, 2014

H. Tschofenig
ARM Ltd.
February 14, 2014

Authentication and Authorization for Constrained Environments (ACE):
Overview of Existing Security Protocols
draft-tschofenig-ace-overview-00.txt

Abstract

This document surveys existing three party authentication and authorization protocols for use with Internet of Things use cases. The discussed protocol frameworks are Kerberos, OAuth, ABFAB, and the certificate model. The aim is to understand whether any of the available standardized security protocols are re-usable for constrained environments. A future version of this document will provide a more detailed analysis against the requirements.

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.

1. Introduction

[I-D.seitz-ace-usecases] introduces a number of use cases that require device-to-device authentication whereby both devices may be constrained. [I-D.ietf-lwig-terminology] discusses the different types of constraints of these devices.

This document aims to raise the high-level question about the possible re-use of existing three party authentication and key exchange protocols for use in IoT environments. This version of the document does not aim to map requirements derived from the use cases against these protocols. Such a detailed analysis is premature at this point when use case descriptions are still in flux.

The starting assumption for the architectures in this document is that a device (a client) wants to access some resource (referred as service in this document). It unfortunately does not have any relationship with the server offering that service. Figure 1 shows the scenario graphically.

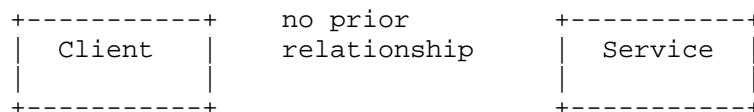


Figure 1: Two Party Scenario.

Imagine that the client is a light-switch and the service is a light-bulb.

Today, companies solve this case by using a pairing protocol (at the link layer typically) where the two devices execute a special imprinting/pairing protocol to establish an initial key by using out-of-band (OOB) channel. This OOB channel can come in many forms:

- o Using an alternative communication channel, such as a USB stick, Ethernet cable
- o Human involvement by comparing hashed keys, entering passkeys, scanning QR codes
- o Second wireless connectivity (e.g., infrared)

- o Proximity-based information

The pairing is a suitable approach where wireless communication replaces a wired communication technology previously used. For example, a headset connected to a music player using a wired connection is replaced with the wireless version. Not all use cases do, however, allow users to pair their device with other devices upfront. Consider an enterprise with electronic door locks. It is hard to imagine an employee who has to pair their digital key with every door in the building first before they use the system.

Requiring every device to pair with every other device upfront is often inconvenient or not feasible. Hence, this document does not explore pairing solutions further. To offer an improved user experience with better scalability properties a device might either share credentials with some trusted third party. There are various ways how credentials can be shared with these trusted third parties. For example, credentials may be provisioned during the manufacturing process or devices may have been paired with the trusted third party (in case the trusted third party is local to the user). In fact, today it is very common for IoT devices to have at least credentials pre-provisioned for use with the vendor / manufacturer of the device to allow software updates to be provided securely.

Thus, we move to a model where the device (client) shares some credentials with a trusted third party. This trusted third party does not need to be a server on the Internet; ideally it could also be operated locally within someones' home, within an enterprise, or within a factory.

This three party architecture is shown in Figure 2.

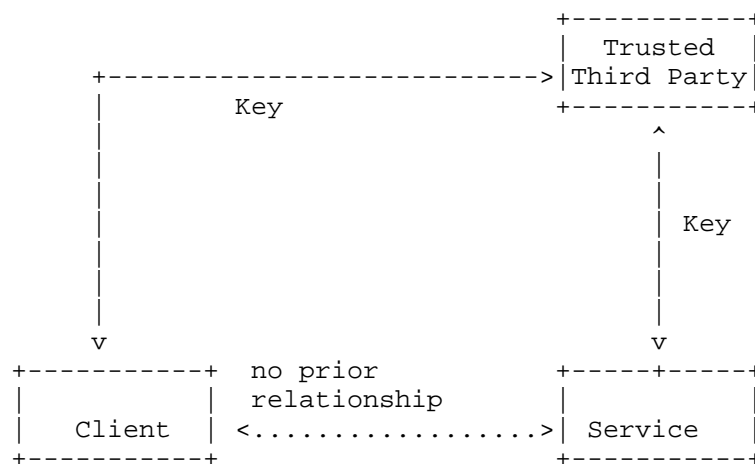


Figure 2: Three Party Scenario.

This three party architecture and messaging pattern has been explored with prior IETF work and this document lists the most relevant efforts (on a high level).

The goal of the communication exchange is that the client has been authorized to access the service, and is able to secure the exchange of information. The client and the service may, optionally, possess keying material for future use of the service with the benefit of better performance for future interactions.

Note: This document does not aim to cover the use cases in their entirety. First, we assume that the security protocol interaction for link layer authentication is outside the scope. The focus of this document is on the application layer interactions when accessing services. Second, this document does not survey access control policy languages and mechanisms for managing these access control policies. These policies are important since many of the systems described below only provide an answer to the question 'Who is the holder of this key?' and standards for answering the question 'Can this key be used for this purpose?' (authorization) are often realized in a proprietary way.

While Figure 1 shows three parties the protocols described in Section 2 have been generalized to four or even multi-party scenario. The result is shown in Figure 2.

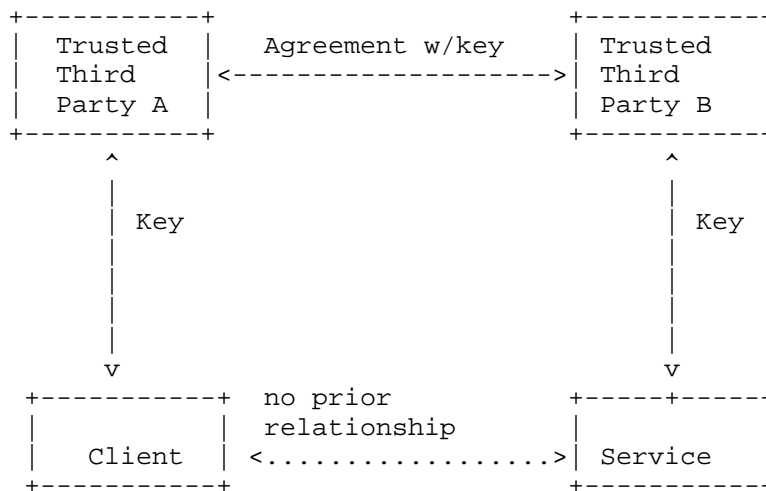


Figure 3: Generalization of Three Party Scenario.

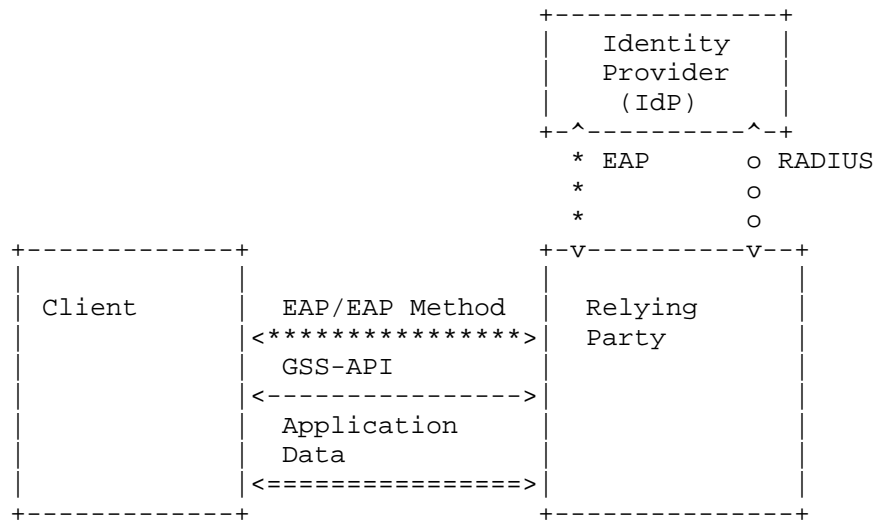
2. Three Party Security Frameworks

This section introduces four authentication and authorization frameworks standardized in the IETF. The description is intentionally kept at a high level and a reader is encouraged to consult the referenced documents for details and various options these protocols offer. The terminology with each of these protocols is lightly different and appropriate mappings have been applied.

To demonstrate the level of maturity of these frameworks availability of products, source code, and deployment experience is mentioned for each of these frameworks. Note, however, that this experience does not imply suitability for use with the IoT environment.

2.1. Application Bridging for Federated Access Beyond Web Architecture

This section describes the Application Bridging for Federated Access Beyond Web architecture [I-D.ietf-abfab-arch], which builds on the Authentication, Authorization and Accounting (AAA) framework. The AAA framework re-uses the Extensible Authentication Protocol (EAP) [RFC5247] and EAP methods for the authentication protocol capabilities. A detailed description of the AAA keying framework can be found in [RFC5247].



Terminology Mapping:

- The term 'Relying Party' corresponds to the 'service'.
- The term 'Identity Provider' corresponds to the 'trusted third party'.

Figure 4: ABFAB Architecture.

With the message exchange shown in Figure 4 the client wants to obtain access to a service and starts interacting with that service. Since no prior relationship between the client and the service is assumed the EAP message exchanges is relayed by the service and the EAP server component of the IdP. Between the client and the service these EAP payloads are encapsulated within the GSS-API. After a successful authentication and authorization session keys are delivered from the IdP to the service and can then be used to secure the application layer data exchange between the client and the service.

While the use of EAP and the AAA architecture has mostly found use for network access authentication the work on ABFAB applies this architecture to application layer services.

Pros:

- o Re-uses existing protocols: RADIUS, GSS-API, EAP, EAP methods.

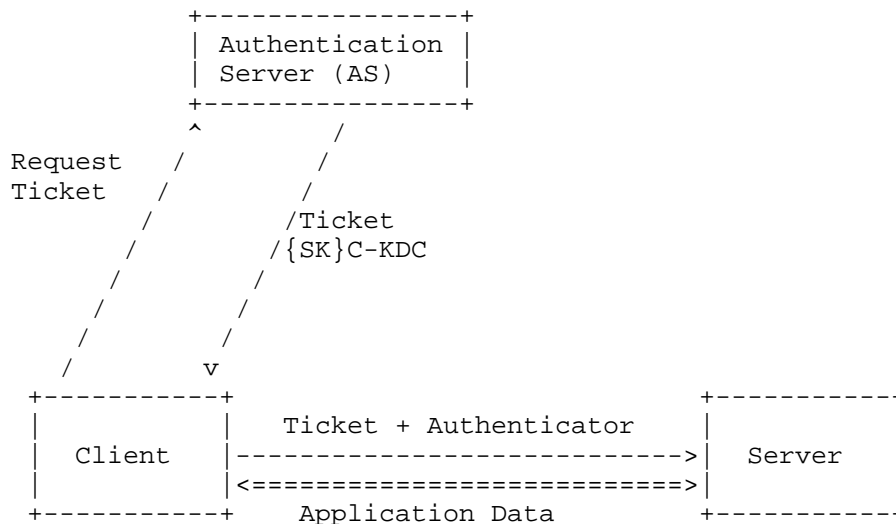
- o Security properties of the AAA / EAP framework well studied and large deployments of the AAA framework exist.
- o Products and open source code exists for EAP, EAP methods, RADIUS, and the GSS-API. The extensions needed for ABFAB also have been implemented but they are less mature compared to the EAP/AAA deployment.
- o Large range of EAP methods available offering all possible authentication and key exchange protocols for authentication between the client and the AAA server. These mechanisms have been deployed and are in widespread use today. While many EAP methods have been standardized only a few are in widespread use in non-IoT environments. However, there are many (open source) implementations available such that further experience concerning IoT suitability can be gathered.
- o IoT devices might use the AAA/EAP architecture for network access authentication (e.g., WLAN-based, IEEE 802.15.4-based ZigBee-IP deployments).
- o The AAA framework also supports authentication in a federated environment.
- o Authorization information is conveyed within RADIUS (and potentially in SAML assertions, as envisioned by ABFAB).

Cons:

- o The initial authentication and authorization exchange requires real-time interaction between the AAA server and the service.
- o Deployments have so far used this architecture mainly for network access and for specific applications (VoIP) only. Experience with other applications, as ABFAB envisions, is rather limited.
- o ABFAB architecture uses layering of EAP within the GSS-API, which adds additional overhead. A binding for the transport of EAP payloads in CoAP, for example, does not exist.
- o No unified authorization policy language has been defined for the AAA/EAP architecture. Instead, RADIUS attributes carry information about access control decisions.

2.2. Kerberos

Kerberos [RFC4120] is authentication system for distributed environments that has enjoyed deployment for more than three decades. The security properties have been extensively studied and various implementations exist.



Terminology Mapping:

- The term AS corresponds to the 'trusted third party.'
- The term Server corresponds to the 'service'.

Figure 5: Kerberos.

The Kerberos exchange shown in Figure 5 illustrates a client who wants to get access to a server. It first has to interact with the Authentication Server (AS) to request a ticket. In response, the AS provides a ticket, which is a data structure encrypted with a key known only between the server and the AS. This ticket includes information about the client, a session key (SK) for later use, and various other security relevant information elements. The client also obtains the session key encrypted with a key it shares with the AS.

When a service access is required then the client interacts with the server and presents the ticket along with an Authenticator. The Authenticator demonstrates that the client was able to decrypt the session key with the key it shares with the AS and that it was able

to apply this key to compute a keyed message digest over several fields, including a time-stamp, when accessing the service. The time-stamp avoids replay attacks.

Pros:

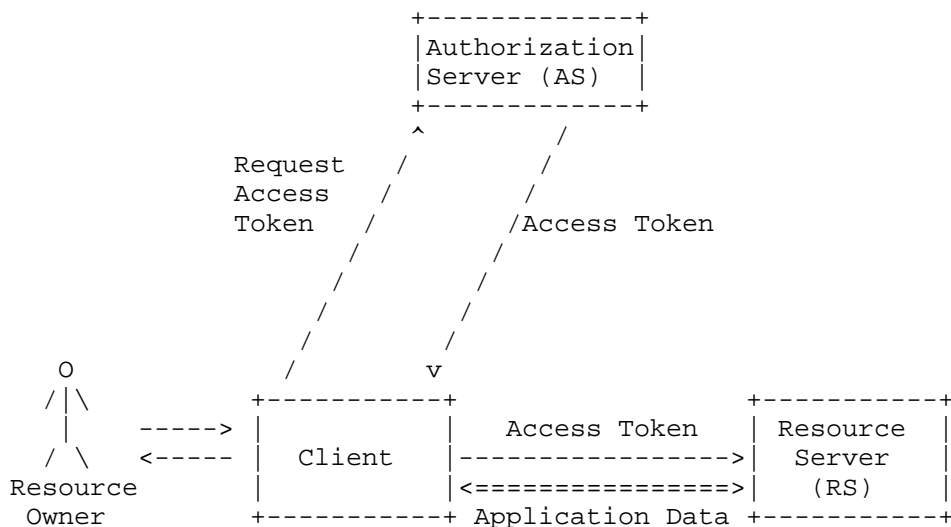
- o Re-uses existing protocol: Kerberos
- o Security properties well studied and large deployments exist.
- o Products and open source service exist.
- o Most parts of Kerberos, particularly the ticket concept, are designed with symmetric key cryptography, which improves performance. The Kerberos ticket is consequently fairly small and uses a binary encoding.
- o Kerberos also supports cross-realm authentication for scalable deployments.
- o Kerberos also specifies a UDP-based transport.
- o The message exchanges between the client and the service can be tailored to the need of the application.

Cons:

- o Each ticket is only usable for a single service (intentionally). As such, new tickets have to be requested whenever the client wants to access a new service or when the ticket expired.
- o For the authentication between the client and the KDC a limited number of authentication protocols have been specified.
- o Kerberos uses ASN.1 for encoding of the ticket and various messages. This may increase implementation complexity but the binary encoding is more efficient than other encodings, like XML or JSON.
- o No standardized access control policy has been standardized for inclusion inside a ticket. Proprietary policies are, however, used in real-world deployments.
- o A CoAP binding for the KRB_PRIV and the KRB_SAFE message exchanges used to secure application data between the client and the service have not been defined.

2.3. OAuth

The OAuth protocol is a recent development for the Web, which re-uses the Kerberos interaction pattern with influences from the Web / mobile app space. It initially aimed to solve the problem of delegated access to protected resources where websites asked users to share their long-term password. Over time OAuth has been used in other use cases that require delegated access.



Terminology Mapping:

- The term AS corresponds to the 'trusted third party'.
- The term RS corresponds to the 'service'.

Figure 6: Simplified OAuth Architecture.

Figure 6 shows the high-level OAuth message exchange. The canonical OAuth example allows a web user (resource owner) to grant a printing service (client) access to her private photos stored at a photo sharing service (resource server), without sharing her username and password. Instead, she authenticates directly with the authorization server which issues the printing service delegation-specific credentials.

Pros:

- o Re-uses existing protocols: OAuth Core [RFC6749], OAuth Bearer Token [RFC6750] OAuth MAC Token [I-D.ietf-oauth-v2-http-mac]/ HOTK [I-D.tschofenig-oauth-hotk], JWT [I-D.ietf-oauth-json-web-token].
- o Large deployments in the Web environment exist, which use the OAuth Bearer Token.
- o Products and open source service exist.
- o OAuth is flexible with regard to the used cryptography. A standardized format for the access token has been described with the JSON Web Token (JWT). For security protection of the JWT various specifications from the JOSE working group are available.
- o The message exchanges between the client and the service can be tailored to the need of the application. Bindings are available for HTTPS and SASL [I-D.ietf-kitten-sasl-oauth].
- o With regard to the offered security mechanism the interaction between the client and the resource server gives several choices: The OAuth Bearer Token requires a TLS exchange between the client and the resource server. The MAC Token specification is conceptually similar to Kerberos; a version based on asymmetric cryptography exists as well (see HOTK).

Cons:

- o For an environment with more than one authorization server or where the authorization server is located in a different domain than the resource server the standardization work is still in progress. Efforts have mostly be done in Kantara with the User-Managed-Access (UMA) working group.
- o A binding for CoAP does not exist for the client to authorization server nor for the client to resource server.
- o The OAuth architecture does not standardize the authentication procedure of the resource owner to the authorization server itself. This is a common approach for the Web environment where a number of different authentication protocols are in use in the browser. As such, the protocol works with any authentication mechanism.

2.4. Certificate Model

Prior work on the Public Key Infrastructure, certificate formats, certificate extensions, and various certificate management protocols can be re-used in the IoT context. With respect to the use cases

described in [I-D.seitz-ace-usecases] certificates may be short-lived and might need to contain attributes (which may be used for making access control decisions) rather than purely relying on the identity of users and their devices.

For the purpose of dynamic provisioning short-lived certificates, this document envisions to re-use a subset of the functionality offered by protocols like the Certificate Management over CMS (CMC) [RFC5272], the Certificate Management Protocol (CMP) [RFC4210], the Simple Certificate Enrollment Protocol [I-D.nourse-scep], Certification Request Syntax Standard - PKCS#10 [RFC2315] (with TLS or with PKCS#7 [RFC2986]). While these protocols offer slightly different features, on a high-level they all fulfill the same function. Note that the management of trust anchors may be provided by a different protocol, such as Trust Anchor Management Protocol (TAMP) [RFC5934].

Of course, certificates do not necessarily need to be short-lived and could even be provisioned during the manufacturing process and never changed during the lifetime of the device. The drawback of such an approach is, however, that mechanisms for certificate revocation have to be provided. Furthermore, privacy concerns might arise since the same client certificate content will be shown to every service rather than information that is only relevant for a specific purpose.

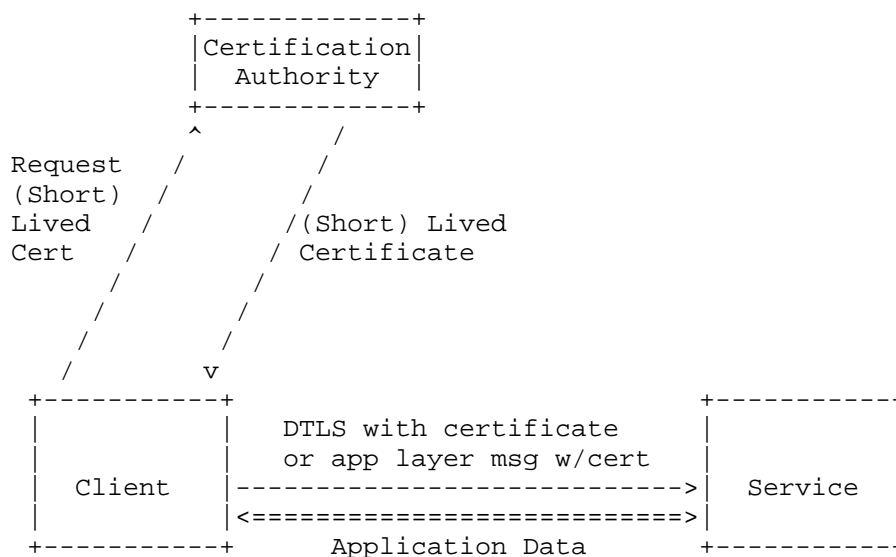


Figure 7: Certificate Model.

Pros:

- o Re-uses existing protocols: DTLS (or application protocol), CMP/CMC/PKCS#10/SCEP, specifications (certificate format - RFC 6818 [RFC6818]), and concepts (PKI).
- o Large deployments on the Web and with enterprise system exist.
- o Products and open source code exists.
- o The certificate format offers flexibility in terms of content. New extensions have been defined over time.
- o Certificates can be used with DTLS without any additional modifications. Certificates can also used with application security mechanisms.
- o Authorization information may be placed in an extension of a public key certificate or in a separate attribute certificate [RFC3281]. Earlier work on KeyNote [RFC2704] could be re-used as it provides a more flexible authorization policy language.
- o A single certificate can be used with a number of different services.
- o Various PKI management protocols have been defined and they offer some flexibility. The properties vary on the specific use cases.

Cons:

- o The certificate format and the PKI management protocols use ASN.1.
- o No UDP or CoAP transport is defined for CMC/CMP/SCEP. For PKCS#10 no transport is defined at all.
- o The public key infrastructure only focuses on asymmetric cryptography. A separate body of work is available for provisioning symmetric keys (like one-time-keys), such as the Portable Symmetric Key Container (PSKC) [RFC6030] and Dynamic Symmetric Key Provisioning Protocol (DSKPP) ([RFC6063]).
- o Protocols for certificate enrollment are in use but many deployments use their own strategy for distributing certificates (typically long-lived) to their users.
- o Asymmetric cryptography is computationally more expensive than symmetric cryptography but offers additional security benefits.

3. Conclusion

Several existing protocols can be used to meet the use cases outlined in [I-D.seitz-ace-usecases]. Each technology presented here offers a number of possibilities for profiling to make them work on for constrained devices. Despite the range of available security protocols, the use cases suggest that there is a need to profile and to extend those in order to make them fit for the constrained environment.

The right choice of authentication and authorization protocol will heavily depend on the envisioned usage environment.

It is, however, also worth noting that several aspects that are not discussed in this document although they appear as requirements in the use case document, namely

- o a language for describing access control policies,
- o the encoding of these policies, and
- o the container for associating these policies with keying material.

4. Security Considerations

This entire document is about security.

5. IANA Considerations

This document does not require any actions by IANA.

6. Acknowledgements

The author would like to thank Stefanie Gerdes for her review comments.

7. Informative References

[I-D.ietf-abfab-arch]

Howlett, J., Hartman, S., Tschofenig, H., Lear, E., and J. Schaad, "Application Bridging for Federated Access Beyond Web (ABFAB) Architecture", draft-ietf-abfab-arch-10 (work in progress), December 2013.

[I-D.ietf-kitten-sasl-oauth]

Mills, W., Showalter, T., and H. Tschofenig, "A set of SASL Mechanisms for OAuth", draft-ietf-kitten-sasl-oauth-12 (work in progress), December 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-oauth-json-web-token]
Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", draft-ietf-oauth-json-web-token-15 (work in progress), January 2014.
- [I-D.ietf-oauth-v2-http-mac]
Richer, J., Mills, W., Tschofenig, H., and P. Hunt, "OAuth 2.0 Message Authentication Code (MAC) Tokens", draft-ietf-oauth-v2-http-mac-05 (work in progress), January 2014.
- [I-D.nourse-scep]
Pritikin, M., Nourse, A., and J. Vilhuber, "Simple Certificate Enrollment Protocol", draft-nourse-scep-23 (work in progress), September 2011.
- [I-D.seitz-ace-usecases]
Seitz, L., Gerdes, S., and G. Selander, "ACE use cases", draft-seitz-ace-usecases-00 (work in progress), February 2014.
- [I-D.tschofenig-oauth-hotk]
Bradley, J., Hunt, P., Nadalin, A., and H. Tschofenig, "The OAuth 2.0 Authorization Framework: Holder-of-the-Key Token Usage", draft-tschofenig-oauth-hotk-03 (work in progress), January 2014.
- [RFC2315] Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", RFC 2315, March 1998.
- [RFC2704] Blaze, M., Feigenbaum, J., Ioannidis, J., and A. Keromytis, "The KeyNote Trust-Management System Version 2", RFC 2704, September 1999.
- [RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.
- [RFC3281] Farrell, S. and R. Housley, "An Internet Attribute Certificate Profile for Authorization", RFC 3281, April 2002.

- [RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.
- [RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, September 2005.
- [RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, August 2008.
- [RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, June 2008.
- [RFC5934] Housley, R., Ashmore, S., and C. Wallace, "Trust Anchor Management Protocol (TAMP)", RFC 5934, August 2010.
- [RFC6030] Hoyer, P., Pei, M., and S. Machani, "Portable Symmetric Key Container (PSKC)", RFC 6030, October 2010.
- [RFC6063] Doherty, A., Pei, M., Machani, S., and M. Nystrom, "Dynamic Symmetric Key Provisioning Protocol (DSKPP)", RFC 6063, December 2010.
- [RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.
- [RFC6750] Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, October 2012.
- [RFC6818] Yee, P., "Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 6818, January 2013.

Author's Address

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

Email: Hannes.tschofenig@gmx.net
URI: <http://www.tschofenig.priv.at>