ACE Working Group
Internet Draft
Intended status: Informational
Expires: September 2015

S. Gerdes
Universität Bremen TZI
March 09, 2015

Actors in the ACE Architecture
draft-gerdes-ace-actors-03

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

Copyright Notice

Table of Contents

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 hardware 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 entrusted with vast amounts of 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 limitations of the constrained nodes ask for security mechanisms which take the special characteristics of constrained environments into account. Therefore, it is crucial to identify the tasks which must be performed to meet the security requirements in constrained scenarios. 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

Readers are required to be familiar with the terms and concepts defined in [RFC4949]. In addition, this document uses the following terminology:

Resource (R):
: an item of interest, which is represented through an interface. It might contain sensor or actuator values or other information.

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

Actor:
: A logical functional entity that performs one or more tasks. Depending on the tasks an actor must perform, the device that contains the actor 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.

Server (S):
: An entity which hosts and represents a Resource.

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

Resource Overseeing Principal (ROP):
: The principal that is in charge of the resource and controls its access permissions.

Client Overseeing Principal (COP):
: The principal that is in charge of the Client and controls permissions concerning authorized representations of a Resource.

Server Authorization Manager (SAM):
: An entity that prepares and endorses authentication and authorization data for a Server.

Client Authorization Manager (CAM): An entity that prepares and endorses authentication and authorization data for a Client.

Attribute Binding Authority: An entity that is authorized to validate claims about an entity.

2. Problem Statement

The scenario this document addresses can be summarized as follows:

- C wants to access R on a S.
- A priori, C and S do not necessarily know each other and have no security relationship.
- C and / or S are constrained.

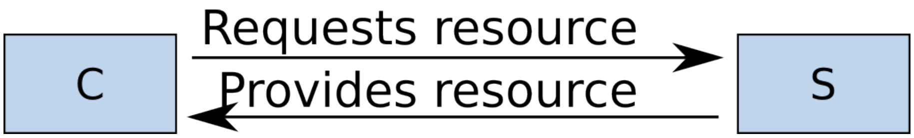

Figure 1: Basic Scenario

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

- Rq0.1: No unauthorized entity has access to (or otherwise gains knowledge of) R.
- Rq0.2: C is exchanging status updates of a resource only with authorized resources. (When C attempts to access R, that access reaches an authorized R).

Rq0.1 requires authorization on the server side while Rq0.2 requires authorization on the client side.

3. Security Objectives

The security objectives that can be addressed by an authorization solution are confidentiality and integrity. Availability cannot be achieved by authorization solutions. However, misconfigured or wrongly designed authorization solutions can result in availability breaches: Users might no longer be able to use data and services as they are supposed to.

Authentication mechanisms can achieve additional security objectives such as non-repudiation and accountability. They are not related to authorization and thus are not in scope of this draft, but still should be considered by Authenticated Authorization solutions. Non-repudiation and accountability may require authentication on device level, if it is necessary to determine which device performed an action. In other cases it may be more important to find out who is responsible for the device's actions.

The importance of a security objective depends on the application the authorization mechanism is used for. [I-D.ietf-ace-usecases] indicates that security objectives differ for the various constrained environment use cases.

In many cases, one participating party might have different security objectives than the other. However, to achieve a security objective, both parties must participate in providing a solution. E.g., if COP requires the integrity of sensor value representations S is hosting, Both C and S need to integrity-protect the transmitted data. Moreover, S needs to protect the access to the sensor representation to prevent unauthorized users to manipulate the sensor values.

4. Authentication and Authorization

Authorization solutions aim at protecting the access to items of interest, e.g. hardware or software resources or data: They enable the principal of such a resource to control who can access it and how.

To determine if an entity is authorized to access a resource, an authentication mechanism is needed. 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.

The security objectives the authorization mechanism aims at can only be achieved if the authentication and the authorization mechanism work together correctly. We use the term *authenticated authorization* to refer to a synthesis of mechanism for authentication and authorization.

If used for authorization, the authenticated attributes must be meaningful for the purpose of the authorization, i.e. the authorization policy grants access permissions based on these attributes. If the authorization policy assigns permissions to an individual entity, the authenticated attributes must be suitable to uniquely identify this entity.

In scenarios where devices are communicating autonomously there is less need to uniquely identify an individual device. For a principal, the fact that a device belongs to a certain company or that it has a specific type (e.g. light bulb) is likely more important than that it has a unique identifier.

Resource and device overseeing principals need to decide about the required level of granularity for the authorization, ranging from *device authorization* over *owner authorization* to *binary authorization* and *unrestricted authorization*. In the first case different access permissions are granted to individual devices while in the second case individual owners are authorized. If binary authorization is used, all authenticated entities have the same access permissions. Unrestricted authorization for an item of interest means that no authorization mechanism is used (not even by authentication) and all entities are able to access the item as they see fit. More fine-grained authorization does not necessarily provide more security. Resource and device overseeing principals need to consider that an entity should only be granted the permissions it really needs to ensure the confidentiality and integrity of resources.

For all cases where an authorization solution is needed (all but Unrestricted Authorization), the authorizing party needs to be able to authenticate the party that is to be authorized. Authentication is therefore required for messages that contain representations of an accessed item. More precisely, the authorizing party needs to make sure that the receiver of a message containing a representation, and the sender of a message containing a representation are authorized to receive and send this message, respectively. To achieve this, the integrity of these messages is required: Authenticity cannot be assured if it is possible for an attacker to modify the message during transmission.

In some cases, only one side (only the client side or only the server side) requires the integrity and / or confidentiality of a resource value. In these cases, principals may decide to use binary authorization which can be achieved by an authentication mechanism or even unrestricted authorization where no authentication mechanism is required. However, as indicated in Section 3, the security objectives of both sides must be considered. The security objectives of one side can often only be achieved with the help of the other side. E.g., if the server requires the confidentiality of a resource representation, the client must make sure that it does not send resource updates to parties other than the server. Therefore, the client must at least use binary authorization.

5. Autonomous Communication

The use cases defined in [I-D.ietf-ace-usecases] demonstrate that constrained devices are often used for scenarios where their principals are not present at the time of the communication. Moreover, these devices often do not have any user interfaces or displays. Even if the principals are present at the time of access, they may not be able to communicate directly with the device. The devices therefore need to be able to communicate autonomously. In some scenarios there is an active user at one endpoint of the communication. Other scenarios ask for true machine to machine (M2M) communication.

To achieve the principals’ security objectives, the devices must be enabled to enforce the security policies of their principals.

6. 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 S or both of them are constrained. Therefore tasks which must be conducted by either C or S must be performable by constrained nodes.

6.1 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:

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

6.2 Security-Related Tasks

The reason for the communication is that C wants S to process some information. S' reaction to C's access request might be processed by C. The reason for using an authorization solution is to validate that the entity that sent the information used for processing is authorized to provide it.

To validate if a sender is authorized to send a received piece of information, the receiver must determine the sender's authorization. Correspondingly, to validate if a receiver is allowed to receive a message, the sender must determine its authorization. This can only be achieved with the help of an authentication mechanism.

6.3 Authentication-Related Tasks

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 entity that is authorized to conduct the attribute binding, the attribute binding authority, checks if an entity actually has the attributes it claims to have and then binds them to a verifier. The binding authority must provide some kind of endorsement information which enables other entities to validate this binding.

Note: The attribute binding can be conducted using either symmetric or asymmetric cryptography.

1. Verifier validation: The entity that wants to authenticate the source of an information checks the attribute-verifier-binding using the endorsement provided by the attribute binding authority.
2. Authentication: The verifier is used for authenticating the source of a data item, i.e. it is checked whether the data item is bound to the verifier. Thus the attributes of the source can be determined.

Step 1 is addressed in Appendix A.2.5. After the first step is conducted, step 2 and step 3 can be performed when needed. They must be performed together and thus are examined together as well. Tasks for step 2 and 3 are Information authenticity (see Appendix A.2.1) and secure communication (see Appendix A.2.3).

6.4 Authorization-Related Tasks

Several steps must be conducted for explicit authorization:

1. Configuration of authorization information: The respective principals (COP and ROP) must configure the authorization information according to their authorization policy. An authorization information must contain one or more permissions and the attribute an entity must have to apply to this authorization.
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 confirmed by applying the request in conjunction with authenticated attributes to the policy provided by the authorization information.
4. Authorization enforcement: According to the result of the authorization validation the access to a resource is granted or denied.

Tasks for step 1 are defined in Appendix A.2.6. Appendix A.2.4 addresses step 2. After step 1 and step 2 are conducted, step 3 and step 4 can be performed when needed. Step 3 and step 4 must be performed together and thus are examined together. Appendix A.2.2 introduces tasks for these steps.

7. Actors

This section describes the various actors in the architecture. An actor consists of a set of tasks and additionally has an security domain (client domain or server domain) and a level (constrained, principal, less-constrained). Tasks are assigned to actors according to their security domain and required level.

Note: Actors are a concept to understand the security requirements for constrained devices. The architecture of an actual solution might differ as long as the security requirements that derive from the relationship between the identified actors are considered. 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.

7.1 Constrained Level Actors

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

C performs the following tasks:

- Negotiate means for secure communication (Task TSecureComm, see Appendix A.2.3).
- Validate that an entity is an authorized source for R (Task TValSourceAuthz, see Appendix A.2.2).
- Securely transmit an access request (Task TSendReq, see Appendix A.1.2).
- Validate that the response to an access request is authentic (Task TAuthnResp, see Appendix A.2.1).
- Process the response to an access request (Task TProcResp, see Appendix A.1.1).

S performs the following tasks:

- Negotiate means for secure communication (Task TSecureComm, see Appendix A.2.3).
- Validate the authenticity of an access request (Task TAuthnReq, see Appendix A.2.1).
- Validate the authorization of the requester to access the requested resource as requested (Task TValAccessAuthZ, see Appendix A.2.2).
- Process an access request (Task TProcReq, see Appendix A.1.1).
- 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 S and not a separate actor. The device on which S is located might contain several resources of different Resource Overseeing Principals. For simplicity of exposition, these resources are described as if they had separate S.

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

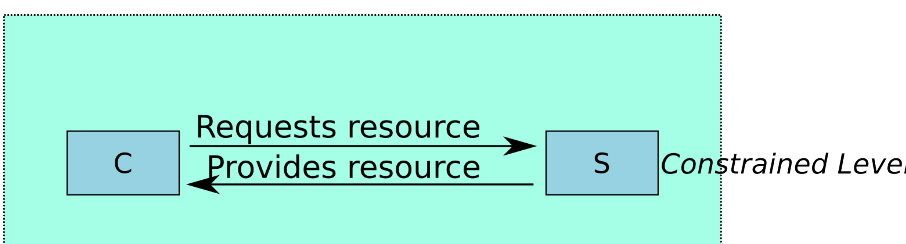

Figure 2: Constrained Level Actors

7.2 Principal Level Actors

Our objective is that C and S are under control of principals in the physical world, the Client Overseeing Principal (COP) and the Resource Overseeing Principal (ROP) respectively. The principals decide about the security policies of their respective devices and belong to the same security domain.

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

COP must fulfill the following task:

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

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

ROP must fulfill the following task:

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

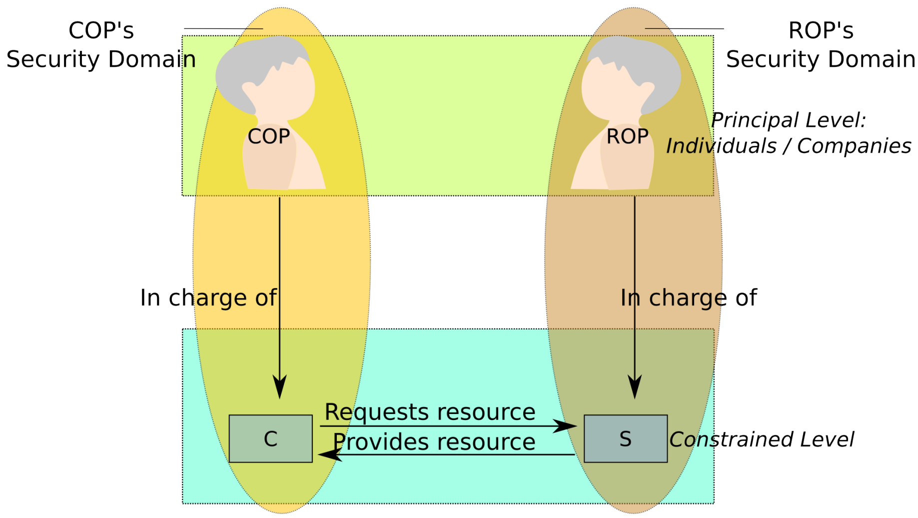

Figure 3: Constrained Level Actors and Principal Level Actors

7.3 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 Client Authorization Manager (CAM) belongs to the same security domain as C and COP. CAM acts on behalf of COP. It assists C in authenticating S and determining if S is an authorized source for R. CAM can do that because for C, CAM is the authority for claims about S.

CAM performs the following tasks:

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

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

SAM performs the following tasks:

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

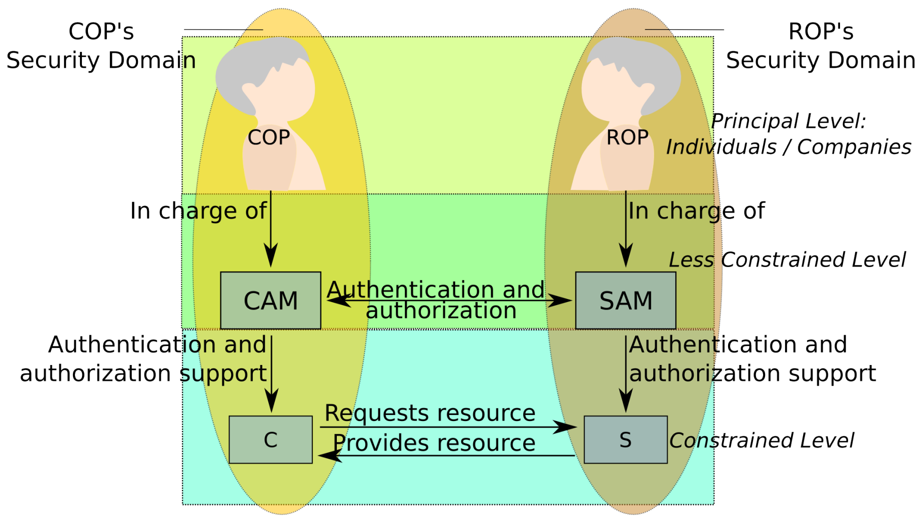

Figure 4: Overview of all Complexity Levels

8. 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.

8.1 Constrained Level Protocols

A protocol is considered to be on the constrained level if it is used between the actors C and S which are considered to be constrained (see Section 7.1). C and S might not belong to the same security domain. Therefore, constrained level protocols are required to work between different security domains.

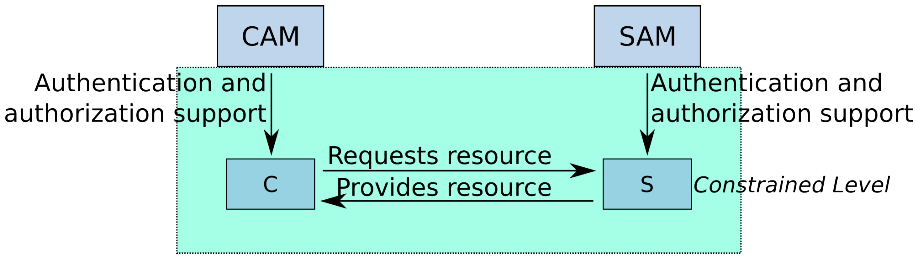

Figure 5: Constrained Level Tasks

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.

8.1.1 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 CAM or S and SAM are therefore support protocols.

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

8.2 Less-Constrained Level Protocols

A protocol is considered to be on the less-constrained level if it is used between the actors CAM and SAM. CAM and SAM 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).

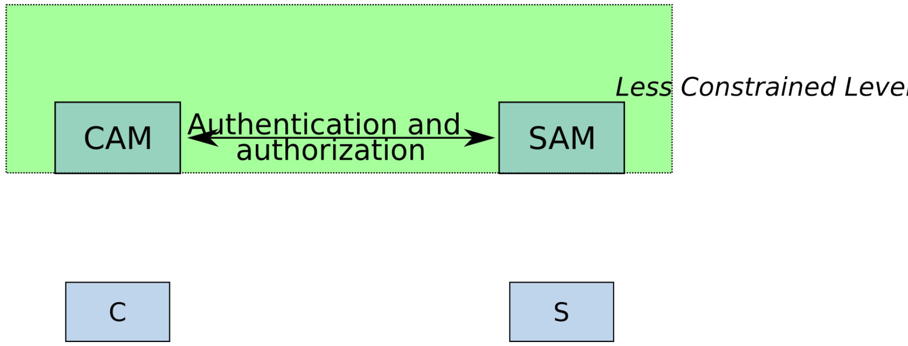

Figure 6: Less-constrained Level Tasks

9. IANA Considerations

None

10. Security Considerations

This document discusses security requirements for the ACE architecture.

11. Acknowledgments

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

12. References

12.1 Normative References

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

12.2 Informative References

[I-D.ietf-ace-usecases] Seitz, L., Gerdes, S., Selander, G., Mani, M., and S. Kumar, "ACE use cases", Internet-Draft draft-ietf-ace-usecases-02 (work in progress), February 2015.

[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.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.

Author's Address

Stefanie Gerdes
Universität Bremen TZI
Postfach 330440
Bremen, D-28359
Germany
Phone: +49-421-218-63906
EMail: gerdes@tzi.org

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 6.

A task has the following structure:

- The name of the task which has the form TXXX
- One or more Requirements (if applicable) of the form RqXXX
- One or more Preconditions (if applicable) of the form PreXXX
- 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 S 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 S. Several entities might have the same attributes.

A.1 Basic Scenario

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

The transmission of information from C to S 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 S is C's intent to access R. To achieve this, S 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 S 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 S is an authorized source for R. S 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 S might receive the information either from S or C directly or from a proxy which is authorized to speak for the respective communication partner.

- 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 S 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. S or an entity which is authorized to speak for S (see Appendix A.2.1).
* PreProcResp.3 (required for Rq0.2): C validated that S or the entity which is authorized to speak for S is an authorized source for R (see Appendix A.2.2).
Postcondition:
* PostProcResp.1: C processed the response.

- Task TProcReq: Process an access request.
  Description: S 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 S.
  * RqProcReq.2: Must be performable by a constrained device (derives from the problem statement: C and / or S are constrained).
  Preconditions:
  * PreProcReq.1: An access request was sent (see Appendix A.1.2).
  * PreProcReq.2 (needed for Rq0.1): S 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): S 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. S 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 S which resource it wants to access with which actions and parameters. S 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 S might transmit the message either to S and C directly or to a proxy which is authorized to speak for the respective communication partner.

- Task TSendReq: Securely transmit an access request.
  Description: C wants to access a resource R hosted by the resource server S. To achieve this, it has to transmit some information to S 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 S or to an entity which is authorized to speak for S. 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 S 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 S, 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 S (see Appendix A.2.3).
Postconditions:
* PostSendReq.1: The request was sent securely to S (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.

- Task TSendResp: Securely transmit a response to an access request.
Description: S sends a response to an access request to inform C about the result of the request. S must assure that response reaches the requesting C. S might send the response to C or to an entity which is authorized to speak for C. The response might contain resource values. S binds the request to S's verifier to ensure the integrity of the message. S uses means to assure that no unauthorized entity is able to access the information in the response.
Requirements:
* RqSendResp.1: Is performed by S (derives from the problem statement).
* RqSendResp.2: Must be performable by a constrained device (derives from the problem statement: C and / or S 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: S 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 S' 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 S (or an entity which is authorized to speak for S) and was not modified during transmission. S 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 S must assure that the authenticated source of the information is authorized to provide the information.

- 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 S or an entity which is authorized to speak for S.
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 S 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 S (fulfills PreProcResp.2, see Appendix A.1.1).

- Task TAuthnReq: Validate the authenticity of a request.
  Description: S checks if the request stems from an entity in possession of the respective verifier and is fresh. Thus, S validates that the request stems from C or an entity which is authorized to speak for C.
  Requirements:
  * RqAuthnReq.1: Must be performed by S.
  * RqAuthnReq.2: Must be performable by a constrained device (derives from the problem statement: C and / or S are constrained).
  Preconditions:
  * PreAuthnReq.1: Means for secure communication were negotiated (see Appendix A.2.3).
  Postconditions:
  * PostAuthnReq.1: S 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 S (or the entity which speaks for S) is in possession of attributes which are necessary for being an authorized source for R. S must validate that C (or the entity which speaks for C) has attributes which are necessary for a permission to access R as requested.

- Task TValSourceAuthz: Validate that an entity is an authorized source for R.
  Description: C checks if according to COP's authorization policy and the authentication endorsement provided by the attribute binding authority, S (or an entity which speaks for S) is authorized to be a source for R. S 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 S are constrained).
  Preconditions:
  * PreValSourceAuthz.1: Authorization information about the entity is available. Requires obtaining authorization information about the entity from C's principal (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 S (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).
- Task TValAccessAuthZ: Validate the authorization of the requester to access the requested resource as requested.
  Description: R's principal configures which clients are authorized to perform which action on R. S has to check if according to ROP'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. S assures that requester's verifier is bound to certain attributes and the authorization information refers to these attributes.
  Requirements:
  * RqValAccessAuthz.1: Is performed by S
  * RqValAccessAuthz.2: Must be performable by a constrained device (derives from the problem statement: C and / or S are constrained).
  Preconditions:

* PreValAccessAuthz.1: Authorization information about the entity are available. Requires obtaining authorization information about the entity from S's principal (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.

- 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 S 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 S 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 6.4, the authorization of an entity requires several steps. The authorization information must be configured by the principal and provided to the enforcing entity.

- Task TObtainSourceAuthz: Obtain authorization information about an entity from C's principal.
Description: C's principal defines authorized sources for R. The authorization information must be made available to C to enable it to enforce COP's authorization information. To facilitate the configuration for the principal 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 is authorized by C's principal.
* RqObtainSourceAuthz.2: Must be performed by an entity which is authorized to speak for C's principal 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 principal.
Preconditions:
* PreObtainSourceAuthz.1: C's principal configured authorized sources for R (see Appendix A.2.6).
Postconditions:
* PostObtainSourceAuthz.1: C obtained S' authorization to be a source for R (fulfills PreValSourceAuthz.1, see Appendix A.2.2).
- Task TObtainAccessAuthz: Obtain authorization information about an entity from S' principal.
Description: S' principal defines if and how C is authorized to access R. The authorization information must be made available to S to enable it to enforce ROP's authorization policies. To facilitate the configuration for the principal this device should have a user interface. The authorization information has to be made available to S in a secure way.

Requirements:
* RqObtainAccessAuthz.1: Must be performed by an entity which is authorized by R's principal.
* RqObtainAccessAuthz.2: Must be performed by an entity which is authorized to speak for R's principal 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 principal.
Preconditions:
* PreObtainAccessAuthz.1: R's principal configured authorization information for the access to R (see Appendix A.2.6).
Postconditions:
* PostObtainAccessAuthz.1: S obtained C's authorization for accessing R (fulfills PreValAccessAuthz.1, see Appendix A.2.2).

A.2.5 Attribute Binding

As described in Section 6.3, 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).

- 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 is authorized by C's principal to validate claims about S.
  * RqValSourceAttr.3: 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).
- 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 S in a secure way. The validating party states that an entity in possession of a certain key has certain attributes and provides S with means to validate this endorsement.
  Requirements:
  * RqValReqAttr.1: Must be performed by an entity which is authorized by S' principal to validate 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 S 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 6.4, several steps have to be conducted for authorization. This section is about the configuration of authorization information.

The principal 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 principal might want to configure which attributes an entity must have to be allowed to represent R. R's principal might want to configure which attributes are required for accessing R with a certain action.

- Task TConfigSourceAuthz: Configure for C authorization information for sources for R.
  Description: C's principal has to define authorized sources for R.
  Requirements:
  * RqConfigSourceAuthz.1: Must be provided by C's principal.
  Postconditions:
  * PostConfigSourceAuthz.1: The authorization information are available to a device which performs TObtainSourceAuthz (fulfills PreObtainSourceAuthz.1 see Appendix A.2.4).
- Task TConfigAccessAuthz: Configure for S authorization information for accessing R.
  Description: R's principal 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 principal.
  Postconditions:
  * PostConfigAccessAuthz.1: The authorization information are available to the device which performs TObtainAccessAuthz (fulfills PreObtainAccessAuthz.1, see Appendix A.2.4).