Janne Poikolainen

AUTHORIZED AUTHENTICATION EVALUATION FRAMEWORK FOR CONSTRAINED ENVIRONMENTS

UNIVERSITY OF JYVÄSKYLÄ
DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS
2016

ABSTRACT

Poikolainen, Janne
Authorized Authentication Evaluation Framework for Constrained Environments
Jyväskylä: University of Jyväskylä, 2016, 95 p.
Information Systems Science, Master's Thesis
Supervisors: Semenov, Alexander & Mazhelis, Oleksiy

The Internet today is growing not only by size, but it is spreading to new areas. New ways to gather more data and control devices are developed in many application areas from smart homes and cities, surrounding environments in cities as well as agricultural settings to industrial settings. This growth is due to miniaturization and the dropping costs. In order to deploy IoT applications in truly pervasive manner the physical size and cost of the devices should remain small. This means especially that in order to keep the cost low some of the device capabilities will be having constraints even when technologies evolve and the price might drop. The compromise is always going to be between narrower deployments with more capable devices and wider deployments with less capable devices.

Wireless communication is in many cases the most economic way and for this reason Wireless Sensor Networks (WSN's) have been used in industrial settings for some time now. The same networking technologies can be used in constrained IoT devices. Many of the current WSN deployments are based on proprietary technologies and do not offer a secure end-to-end communication. Instead they provide the data for the Internet through gateways translating the WSN communication. The communication security is based on settings provided in the time of provisioning the devices.

End-to-end connectivity and security can be realized by using IP-based protocols developed for constrained devices. But dynamic access control for these environments is still more or less an open question. A dynamic authorized authentication mechanism would make the systems even more integratable and easily maintainable. This paper deals with the problem field of conducting dynamic authorized authentication in constrained environments. The main artifact of this study is a framework that identifies both the constraints and security objectives for realizing authorized authentication in constrained environments.

Keywords: Access control, Internet of Things, Constrained Environments, Authorized Authentication

TIIVISTELMÄ

Poikolainen, Janne
Autorisoidun authentikoinnin arviointikehys rajoitettuihin ympäristöihin
Jyväskylä: Jyväskylän yliopisto, 2015, 95 s.
Tietojärjestelmätiede, pro gradu-tutkielma
Ohjaajat: Semenov, Alexander & Mazhelis Oleksiy

Internetin kasvu ei perustu tällä hetkellä vain uusien solmujen määrään, vaan Internet on levittäytymässä aivan uusille alueille. Viimeaikoina erilaiset tavat kerätä tietoa ja ohjata laitteita uusin tavoin ovat yleistyneet esimerkiksi teollisuudessa, kaupunkiympäristöjen melun ja saasteiden seurannassa. Lisäksi käsitteet älykoti tai -kaupunki alkavat olla yleisesti tunnettuja. Nykyinen kasvu näiden teknologioiden käytössä perustuu pitkälti laitteiden koon pienenemiseen ja hintojen laskuun. Jotta Esineiden Internet pystyy kasvamaan merkittävällä tavalla, laitteiden fyysisten kokojen ja hintojen tulisi pysyä matalalla tasolla tai laskea edelleen. Pieni koko ja hinta tarkoittaa kuitenkin usein rajoituksia laitteiden ominaisuuksille. Vaihtoehtoina tuleekin luultavasti aina olemaan rajoittuneempien laitteiden laajempi käyttö tai kyvykkäämpien laitteiden kapeampi käyttöönotto.

Langattomat yhteydet ovat usein edullisin tapa toteuttaa verkko-ominaisuuksia erilaisille laitteille. Tästä syystä langattomia sensoriverkkoja on käytetty teollisuudessa jo pidemmän aikaa. Samat verkkoteknologiat sopivat myös käytettäväksi Esineiden Internetin laitteille. Suuri osa nykyisistä langattomista sensoriverkoista käyttää kuitenkin kaupallisia verkkostandardeja, jotka eivät ole yhteensopivia Internet teknologioiden kanssa. Tästä syystä tämän tyyppisillä järjestelmillä ei saavuteta päästä-päähän yhteyttä Internetissä ja rajoitetussa ympäristössä sijaitsevien laitteiden välille. Tämä tarkoittaa myös sitä, että viestinnän turvaamista ei voida toteuttaa päästä-päähän, vaan viestit puretaan ja suojataan uudelleen, kun ne poistuvat tai tulevat rajoitettuun verkkoon.

Ratkaisuiksi näihin yhteensopivuus ongelmiin on kehitetty IP-pohjaisia protokollia, jotka ovat tarpeeksi kevyitä rajoitetuille laitteille. Yhteyden luomiseen kahden rajoitetun laitteen välille dynaamisesti standardoitua ratkaisua ei kuitenkaan vielä ole. Dynaaminen ratkaisu rajoitettujen laitteden välisen liikenteen turvaamiseen tekisi järjestelmistä entistä paremmin integroitavia ja helpommin ylläpidettäviä. Tämä tutkielma käsitteleekin juuri niitä ongelmia, jotka tulisi ratkaista, jotta rajoitettujen laitteiden dynaamiseen autorisointiin voitaisiin löytää yleisesti hyväksytty menetelmä. Tutkielman artefakti on arviointikehys, joka tunnistaa laitteiden rajoitteet ja turvallisuustavoitteet tällaiselle ratkaisulle.

Asiasanat: Esineiden Internet, Rajoitetut Ympäristöt, Autorisoitu Autentikointi

FIGURES

TABLES

INDEX

1 Introduction

Internet of Things (IoT) is a paradigm describing how objects possessing networking and collaborative abilities have become more ubiquitous and continue to will do so in more pervasive way in the future. This paradigm also predicts that in the near future an increasing amount of information produced for the Internet will not be produced by humans. These visions are based on the continuing development in communication technology and electronics that will not only bring the cost of the technology down, but also bring networking and collaborative abilities to more and more things in our environment. These things include anything from household items to home automation and smart city infrastructure to industrial applications.

Especially the communication between machines often takes place in a more constrained environment than the Internet. In practice a constrained environment can mean constraints on network capacity, processing power or available memory of the things or all of these together. An example of the cause for these constraints is a high packet loss in the networks due to the frequencies used. Due to the constraints these devices are not able to use normal Internet protocols for communication, securing their transmissions or authorization. Constraints that prevent protocol use can come from too big overhead of network packets or too low processing power and memory for using such things as public keys.

In addition to the constraints mentioned the power consumption of these devices should remain low, since many of these devices can be battery powered. The battery consumption should remain low since usually the devices are expected to function for years with out the need for a battery change. The dominating consideration when energy consumption is concerned is the network bandwidth usage. This is due to the fact that radio communications usually consumes a big portion of the devices total energy consumption.

These edges of the future Internet will be constructed of smart objects gathering data from and in some cases also acting in the physical world. These devices only handle very simple tasks, such as provide sensor data on temperature or humidity readings or trigger events such as move an actuator. The most economical way to handle such simple tasks is to use simple devices to keep the cost of the devices and their deployment low. The balance between cost and device abilities means that the devices will always have certain constraints.

Despite the constraints the devices have they still need to be able to function in a secure manner due to privacy concerns. Much of the data collected these devices is potentially sensitive in nature. The devices may be collecting data from everyday life such as home utility consumption. This kind of scenario can infringe the users privacy by allowing and eavesdropper to conclude whether the user is home or not. For this reason it is not enough to secure the data only when it leaves the local network, but an end-to-end solution for securing the communication is needed. (Kothmayr, Schmitt, Hu, Brünig, & Carle, 2013)

Currently most solutions gathering data this way are unable to deliver end-to-end security, due to the networking protocols they use. These protocols are not inter-operable with normal Internet nodes, but use translating gateways to communicate to the Internet. These gateways not only translate the incoming and outgoing network packages, but are also in charge of applying security on the transmissions too. In this kind of setting where the constrained networks protocols are incompatible with the common Internet protocols, end-to-end security can only be achieved within the network.

One solution for this problem is to use IP-based protocols. A protocol stack light enough to be used in constrained environments has been around for some years and described by IETF Request For Comments documents. This stack consist of IPv6 for Low Power and Lossy Networks (6LoWPAN), Constrained Application Protocol (CoAP) and IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL). These protocols enable end-to-end communication between constrained nodes and normal Internet nodes through a border router. This communication can also be secured end-to-end using Datagram Transport Layer Security (DTLS).

End-to-end security ability is not the only thing using IP-protocol stack has to offer compared to other protocols. End-to-end communication ability removes many barriers on how these small devices can be used and integrated in other systems and even the Internet. It provides more seamless integrateability, maintainability and possibility to develop more future proof and evolvable systems. For example better integrateability enables easier way for communication between systems from multiple vendors and areas. This could mean such things as integrating a lighting system of a building with ventilation and heating systems functionalities would be a matter of configuring what data is shared between these systems. Compared to the state of such systems are today, this would mean substantial gains on many areas where siloed systems co-exist side by side but unable to communicate.

As mentioned above using IP-protocol stack and end-to-end security is possible using existing protocols in the constrained environments. But the open question that still remains is how the devices could establish a secure communications channel between them selves with out a previous security context. Currently there is no consensus on a protocol for establishing a secure context between two constrained devices. A consensus is needed in order to provide this the ability to constrained devices in universal way. So a common mechanism for authorized authentication between the devices needs to be decided on and standardized. The major challenge on realizing such a system is how to compose such a protocol within the limits set by environment constraints.

1.1 Motivation

Several protocols have been proposed as a standard for providing authorized authentication in the constrained environments. But at the time of this writing none of them have reached the level of proposed standard. Motivation for this study is to help the process of protocol selection by identifying a set of properties required to realize a solution. An evaluation framework would help the process by combining the constraints the protocols have to submit to and how they effect on security objectives it needs to meet. The major contribution of this study is to provide an overall picture on the problem world of selecting a universally accepted dynamic authorized authentication protocol for constrained environments.

1.2 Objectives and expected results

The objective of this study is to identify the critical features the solution for authorized authentication in constrained environments has to possess and build an evaluation framework that would capture these features. First objective is that this framework is able to identify the constraints. Second objective is to categorize the constraints in order to verify accepted limits for different levels of security possible. The third objective is to define adequate security objectives for a proposed solution. The constraints and security objectives together will form a framework to capture the perquisites for a authorized authentication that can operate under the environment constraints and do so in a secure manner.

After the framework is developed it is applied to two protocols proposed for authorized authentication solution for constrained networks. To gain further knowledge how well one of these protocols is able to meet the environment constraints, it is evaluated further by conducting a simulated experiment. Experiment results are used with protocol definitions to assess the ability of these protocols to cope with environment constraints. Next the security objectives of the framework are tested against the protocol specifications to define if the security objectives match the protocol properties. Based on the analysis between the objectives and protocol properties, the objectives are operationalized against a use case to determine the dependencies between the objectives and use case requirements.

After these steps the framework is evaluated on how well it was able to capture the constraints and how well did the security objectives were able to set basis for different areas of a distributed system.

The expected result of this study is the creation of an artifact, a framework for authorized authentication evaluation. This framework would be able to identify the environment constraints, objectives for secure distributed system and so act as a guideline for protocol selection.

1.3 Research questions

Based on the objectives of this study first the framework needs to identify the constraints posed by the environment. Since the purpose of such a mechanism is security, objectives for a secure system need to be identified also. The identification of constraints and their effect on the mechanism should give answers to the first research question of this study which is:

> RQ 1: What are the prerequisites for establishing authorized authentication mechanism between two devices when one or both have constrained capabilities?

The identifying different constraints answer to the first sub-question:

> RQ 1.1: What kind of constraints do the devices have?

After the constraints are identified it would be helpful to have a taxonomy for the different constraints, which brings us to to the next sub-question:

> RQ 1.2: How the device constraints should be classified?

To understand what the identified constraints mean when choosing a mechanism the third research question needs to be answered:

> RQ 1.3: Which constraints have effect on choosing the mechanism?

After the constraints and their effect on choosing the mechanism are understood, the requirements for a system can be identified. This formed as the second research question of this study:

> RQ 2: What are the requirements for a system supporting authorized authentication between two constrained devices?

Based on the requirements, a system fulfilling them can be described, so there fore RQ 2 has a sub-question:

> RQ 2.1: What kind of system could satisfy these requirements?

1.4 The structure of this study

First the research method for this study is determined and the objectives and expected results are described in further detail in the following chapter. Then then some central concepts are introduced in chapter 3 to help the reader to continue to the more specific subjects. The first of these is the definition of constrained environments in chapter 4, which is expanded with an architecture for authorization in constrained environments in chapter 5. Chapter 6 discusses the security concerns of distributed systems in a general level. This is followed by introducing two protocols proposed as a solution for authorized authentication in constrained environments in chapter 7 and a use case for a system requiring these features is given in chapter 8. Chapters 9 & 10 consist of describing the empirical experiment included in this study and portraying its results. The results for both the literary review conducted in the first part of this study and the experiment are evaluated in chapter 11.

2 Research methods

The field of IT-research is a study of artificial as opposed to natural phenomena. As natural science aims to understand reality, design science attempts to create things that serve human purposes. Since design science is technology-orientated, its products are assessed by value or utility criteria, such as does it work? or is it and improvement? Rather than producing general theoretical knowledge, design science produces and applies knowledge of situations or tasks in order to produce successful artifacts. (March & Smith, 1995)

The design science is fundamentally a problem solving paradigm that has it's roots in engineering. Design Science creates and evaluates IT-artifacts intended to solve identified organizational problems. The artifacts are represented in a structured form such as software, formal logic, rigorous mathematics or informal natural language descriptions. The further evaluation a new artifact can be placed in an organizational contexts, which gives the opportunity to apply empirical and qualitative methods. (Hevner, March, Park, & Ram, 2004)

Both behavioral science and design science paradigms are needed to ensure the relevance and effectiveness of information system research, even though the paradigms have different philosophies. The behavioral science paradigm seeks to find ”what is true” as the design science paradigm seeks to create ”what is effective”. While one can argue that utility relies on truth, the discovery of truth may not provide application to this utility. In this setting design science paradigm can be seen as a proactive agent. It focuses on creating and evaluating artifacts that enable organizations to address important information related tasks. On the other hand behavioral science paradigm is reactive with the respect that it takes technology as a given and focuses on developing theories to explain phenomena related to the acquisition, implementation, management and use of technologies. (Hevner et al., 2004)

Hevner et al. (2004) identify seven guidelines for design science in information systems research. These guidelines are: design as an artifact, problem relevance, design evaluation, research contributions, research rigor, design as a search process and communication of research. These guidelines dictate that knowledge and understanding of a design problem and its possible solutions are acquired by creating an innovative purposeful artifact for a specified problem domain. And to make sure that the artifact has utility value for the specified problem, evaluation of the artifact is very important. The artifact must also be innovative and solve an unsolved problem or solve a known problem more efficiently to contribute novel research information. To meet research rigor guideline the artifact must be rigorously defined, formally represented, coherent and internally consistent. The process of creating and the artifact it self can enable a search process where the problem is processed and an effective solution is found. Finally the result of the research must be communicated effectively to both technical and managerial audience. (Hevner et al., 2004)

2.1 Design Science Research Method

The seven guidelines presented by Hevner et al. (2004) among other works on the area have been refined as a Design Science Research Method (DSRM) proposed by Peffers et al. (2007).

The DSRM framework is aimed to be a commonly accepted and consensus building framework for Design Science research. To accomplish this they based their work on well-accepted elements described in prior research and current thought to determine the appropriate elements for what the DS researchers did or should do. The result of their synthesis was a process model consisting of six activities in a nominal sequence, that are described next. (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007)

Activity 1: Problem identification and motivation. The aim of this activity is to define a specific research problem and justify the value of the solution. The problem definition is used for artifact development that is aimed to provide a solution. Depending on the complexity of the case it may be useful to atomize the problem conceptually to help capturing it's complex features. The solution value justification serves two purposes: it provides motivation and helps to understand the reasoning of the researcher's understanding of the problem. The motivation part of this activity is intended to help both the researcher and the audience to pursue the solution and accept the results. The required resources for this activity include appropriate knowledge of the state of the problem and the importance of it's solution.

Activity 2: Define the objectives for a solution. This activity indicates the objectives of a solution based on the problem identification and knowledge of possible and feasible options. The given objectives can be qualitative or quantitative. Qualitative objectives can be such as in which terms a desirable solution would be better than current ones. Quantitative objectives can describe for example how a new artifact is expected to support solutions to novel problems not yet addressed. The objectives should be inferred rationally from the problem identification. This activity requires knowledge of the state of problems and current solutions, if any exist, also the efficiency of the current solutions.

Activity 3: Design and development. This activity deals with artifact creation. Artifacts can be broadly defined constructs, models, methods or instantiations (Hevner et al., 2004). Conceptually an artifact in design research can be any designed object that embeds the research contribution in it's design. This activity determines the artifact's functionality, architecture and creates the actual artifact. This activity moves from objectives to design and development. The resources required for this transition include knowledge of the theory that can be applied in a solution.

Activity 4: Demonstration. This activity demonstrates the use of an artifact to solve one or more instances of a problem. This activity could involve using the artifact in experimentation, simulation, case study, proof or some other appropriate activity. Resources required for this demonstration include effective knowledge on how an artifact can be used to solve a problem.

Activity 5: Evaluation. During this activity observations and measurements are made to determine how well the artifact supports a solution to the problem. The measurements and other observed results extracted form the use of the artifact are then compared to the objectives of a solution. This requires knowledge of relevant metrics and analysis techniques. Evaluation can take many forms depending on the nature of the problem and artifact. Evaluation can include comparison of the artifacts functionality to the solution objectives, quantitative performance measures, results of satisfaction surveys, client feedback, simulations or quantifiable measures of system performance. Conceptually the evaluation could include any appropriate empirical evidence or logical proof. At the end of this activity the researchers can decide if they want to iterate back to activity 3 to try to improve the effectiveness of the artifact or to continue on to communication. The nature of the research venue may dictate if the iteration is feasible or not.

Activity 6: Communication. In this activity the problem and it's importance, artifact, artifacts utility and novelty, rigor of the artifacts design and the artifacts effectiveness are published. The structure of DSRM process can be used to structure a scholarly research publication. As the nominal structure of and empirical research process (problem definition, literature review, hypothesis, development, data collection, analysis, results, discussion and conclusion) is a common structure for empirical research papers. Communication requires knowledge of the disciplinary culture.

The DSRM process is structured in a nominally sequential order. However it does not expect that researchers would always proceed through the activities in sequential order. In reality the researchers can start at almost any step and move outward. The nominal sequence is based in a problem-centered approach that starts with activity 1. This sequence is natural for research ideas that are resulted from observation of a problem or from suggested future research. To an objective-centered solution the first activity is 2. Objective-centered research can be derived from an industry or research need that can be addressed by artifact creation. A design- and development-centered approach starts with activity 3. It could result from an existing artifact that has not been formally examined as a solution for and explicit problem domain in which it could be used. An artifact could have been used in a different problem or it could have come from another research domain. A client-/context-initiated solution starts from activity 4 and it may be based on observing a working practical solution. This means that the researchers work backward to apply rigor to the process retroactively. This kind of approach could be initiated from a consulting case. (Peffers et al., 2007)

2.2 Requirements engineering

Since the main artifact of this study is a evaluation framework for a software solution running on tightly specified hardware, requirements engineering principals apply on operationalization of the framework security objectives. The framework identifies the security objectives for a solution, which are then operationalized using use case requirements defined in chapter 8.

Requirements engineering can be defined as a coordinated set of activities for exploring, evaluating, documenting, consolidating, revising and adapting the properties of a new or revised system. The goal of a software project is to build a machine that is intended to solve a problem and so improve the world. (Van Lamsweerde, 2009)

When considering the behavior of the new system a decision has to be made on which parts of the world are considered as parts of the problem and therefore need to be analyzed. Pervasive views going into very small details are impractical, so a subset of real-world elements considered relevant is chosen to define the system context. (Haley, Laney, Moffett, & Nuseibeh, 2008)
A process of building a machine needs to investigate the problem world in two versions of the same system: the system-as-is and the system-to-be. These states are the system as it existed before the machine was built and how it should be when the machine is built and operational. The project is initiated because the system-as-is has problems, deficiencies or limitations, which the system-to-be is intended to address based on technology opportunities. The problem world can be divided in three dimensions: why, what and who. (Van Lamsweerde, 2009)

The why dimension aims to identify and make explicit the objectives and reasons for a new version of a system. The objectives need to be identified in regards to the limitations of the system-as-is and the opportunities to be exploited. In order to do so first a through domain knowledge must be acquired and in the basis of the knowledge alternative options and technology options must be evaluated. The objectives of system-to-be should also satisfy the possible conflicting viewpoints, interests or perceptions in the problem world. (Van Lamsweerde, 2009)

The what-dimension identifies the functional services needed to satisfy the objectives identified in the why-dimension. Functional services need to meet constraints and assumptions such as performance, security, usability, interoperability and cost. These constraints and assumptions may be identified from usage scenarios envisioned for the system-to-be or agreed system objectives. (Van Lamsweerde, 2009)

The who-dimension assigns the responsibilities derived from the objectives, services and constraints defined in why- and what-dimensions for the components of the system-to-be. These components include human actors, devices and software. The goal is select the assignments so, that the risk of not achieving system objectives, services or constraints is minimized. (Van Lamsweerde, 2009)

There are two main types of statements involved in requirements engineering: descriptive and prescriptive statements. Descriptive statements state system properties that are true regardless of the system behavior and the properties stated by prescriptive statements are dependent on the system behavior. It is essential to make distinction between descriptive and prescriptive statements since prescriptive statements may be changed or altered and the descriptive statements may not be changed or altered. (Van Lamsweerde, 2009)

Requirements themselves can also be categorized in two groups: functional- and non-functional requirements. Functional requirements address the 'what' aspects described above and refer to the services the software should provide and non-functional requirements define the constraints how the services should be provided. (Van Lamsweerde, 2009) Quality requirements include among others the security attributes which are in central role in this study.

Security requirements can be defined as constraints on the systems functional requirements, instead of themselves being functional requirements. Security requirements are prescriptive requirements like functional requirements, since they provide a specification to achieve desired effect. Security requirements are realized using security objectives. A single security requirement can operationalize one or more security objectives. On the basis of security objectives operationalized to security requirements, satisfaction arguments can be formed to show the system is able to respect the security requirements. (Haley et al., 2008)

2.3 Research process of this study

The DSRM framework is used as the basis for research process of this study. This study has started with a problem-centered approach so the nominal order of DSRM starting from activity 1 applies. The different parts of the research are divided in the six activities as follows:

Activity 1, Problem identification and motivation: The research problem was first identified in the introduction chapter. Motivation for the research was provided in an individual sub chapter 1.1. The initial problem identification and motivation provided in the introduction chapter are supplemented by describing the smart object paradigm and existing legacy and IP-based protocols in chapter 3.

Activity 2, Define the objectives for a solution: The objective for this study is defined in the introduction chapter 1.2. The objective is to construct a framework for evaluating protocols proposed to authorized authentication in constrained environments. The objectives are formed into concrete research questions in chapter 1.3.

When compared to current solutions on the problem this study evaluates, the key property is dynamicity . Current solutions to authorization in the constrained environments are not dynamic in a sense that in most cases the devices are configured when they are commissioned and rarely or never reconfigured

When current solutions similar to the main artifact of this study are considered. A framework that would capture both the constraints and security objectives in this manner does not exist. A literary review was conducted as a part of this study to combine features from previous research, so building a more holistic view of the problem world.

Activity 3, Design and development: The main artifact of this study will be developed based on the literary review conducted in chapters 4, 5 and 6. First part of the framework identifies the constraints and provides classifications for memory and power consumption constraints. This chapter also provides answers to the sub research questions 1.1 and 1.2. The second part of the framework is the security objectives derived from IETF architecture for authorization for constrained environments. This architectures purpose is to describe not only actors and functional requirements, but also some security objectives for designing a authorization solution for constrained environments. The third part deals with common security considerations when building a distributed system. This part identifies the different parts of a secure system and brings more security related objectives to the framework. Figure 1 illustrates the design process and structure of the framework.

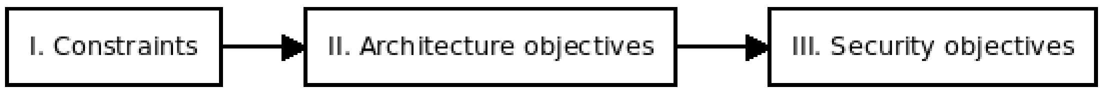


Figure 1: The design process and structure of the main artifact

Activity 4, Demonstration: The use of the artifact is demonstrated by assessing two proposed protocols described in chapter 7. Both protocols are assessed in specification level. In addition one of the protocols is experimented with in a simulated environment. During the simulated experiment described in chapter 9 data is gathered to determine how well this protocol handles certain constraints. The data is presented in chapter 10. After the experiment the framework is applied to the protocols by discussing what kind of solutions they bring to different areas of the framework in chapters 11.1, 11.2 and 11.3. Next the framework security objectives are operationalized to use case requirements and linked to the previous discussions on protocols in chapter 11.4. The experiment completes the answer to research question 1 by answering the remaining sub-question 1.3. Research question 2 and it's sub-question 2.1 is answered by applying the framework to a use case.

Activity 5, Evaluation: The basis for the evaluating the artifact is provided by the demonstration activity, where framework is applied to the protocols and use case. In the evaluation activity is conducted in chapter 11.5 where framework it self is assessed on how well it is able to capture the features of the proposed protocols and provide objectives for the use case requirements.

Activity 6: Communication. The results of this study including the artifact it self will be published as a masters thesis for University of Jyväskylä and it is published in electronic form in jyx.jyu.fi digital archive.

3 Smart object technologies

Smart objects is a good umbrella term for the devices addressed in this study. A technical definition for a smart object is and item equipped with some form of sensor or actuator, microprocessor, communication device and a power source. The first two of the defined traits allow the smart object to interact with the physical world, with the microprocessor the smart object can transform the captured data or control an actuator and it can communicate it's sensor readings or receive commands with the communication device. (Vasseur & Dunkels, 2010)

Smart objects can be used to sense simple physical properties such as light, temperature or air humidity. They can also be used to sense more complex variables like air pollution or when an industrial machine needs service or is about to brake down. Smart objects can also effect the physical world by using different types of actuators. An actuator in this context can mean anything from simple tasks like switching on a small led or as complex as adjusting the heating in a particular part of a building. A single smart object can be very useful, but their real strength comes from their ability to communicate. This enables different functionalities to be combined by smart objects communicating with each other. This could be something like a switch on a door that communicates to other nearby smart objects to turn on the lights, adjust the heating and other functionalities in a house. (Vasseur & Dunkels, 2010)

Another way to define a smart object is based on their behavior. The behavior of a smart object is based on where and what kind of task it is used. A smart object in a container logistics application for example behaves differently than a smart object used to control a smart home functionality. Another important point is that smart object should be designed future proof in some level, since it is impossible to know exactly how they are used in the future. However this does not change the two behavioral properties common to all smart objects: interaction with the physical world and communication. (Vasseur & Dunkels, 2010)

The third definition of smart objects comes from user interaction. Because smart objects have a dual nature as physical and digital entities, they bring forward the fact that Internet of Things cannot be viewed only as a technical system, but it has to be considered as a human centered interactive system. For this reason smart object design has to be expanded beyond hardware and software and include interaction design and social aspects as well. (Kortuem, Kawsar, Fitton, & Sundramoorthy, 2010)

The smart objects are quickly emerging as a technology, never the less there are still are some challenges both node and network levels. At the node level the challenges that have to be addressed are physical size, cost and power consumption. At the network level the challenges come from the scale of nodes in a smart object networks, power consumption and memory constraints. The challenges in the smart object technology it self are standardization and interoperability. As the technology will be produced by many different parties standardization is in a essential role. Interoperability is also essential to integrate smart object devices in the existing IT ecosystem. (Vasseur & Dunkels, 2010)

Historically the origins of smart objects come from the separate strands of development of computing and telephony. Smart objects can be seen as the middle ground between computing and telephony as it borrows features from both. The culture of engineering evolvable systems comes from the computing heritage and the telephony heritage gives the smart objects the principal of connecting disparate systems managed by different organizations. Other areas that have influenced and are related to smart objects are embedded systems, ubiquitous and pervasive computing, mobile telephony, telemetry, wireless sensor networks, mobile computing and computer networking. All the smart object related areas are illustrated in Figure 2 Some of these have industrial background and others have emerged from academic research communities. The relating factors with all the aforementioned areas are that they deal with computationally assisted communication between physical items, wireless communication or involve interaction between the virtual and physical world. (Vasseur & Dunkels, 2010)

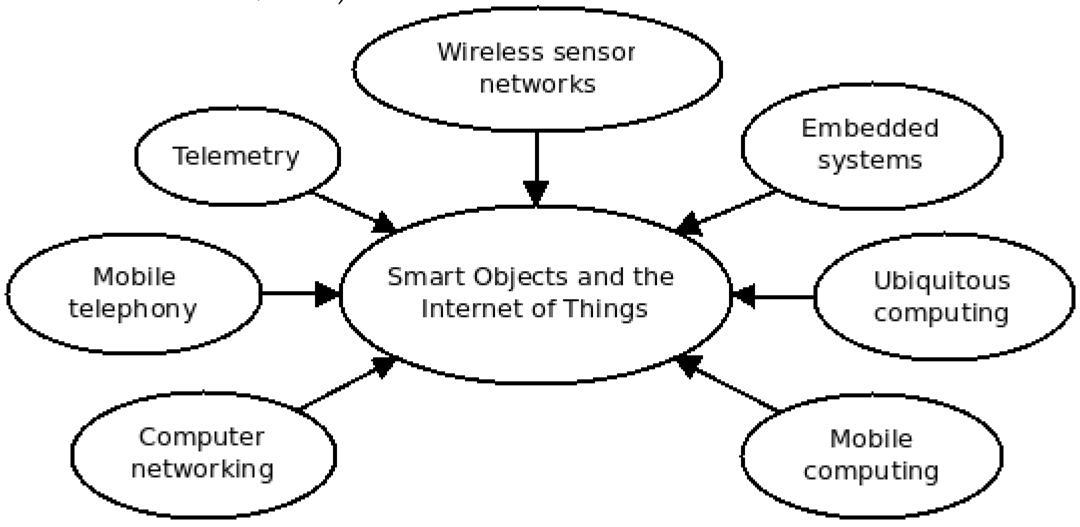


Figure 2: Smart objects and other key technologies (Vasseur & Dunkels, 2010)

3.1 Wireless Sensor networks

The concept of Wireless Sensor Networks (WSN's) is very similar to that of smart objects, with the difference that smart objects are less focused on data gathering. On the other hand WSN's are based on the idea that small wireless sensors are capable to collect and transmit information from the physical environment. WSN's are composed of small sensor nodes that transmit information to a base-station and also help each other to relay the information if the base station is out of reach for some sensors. (Vasseur & Dunkels, 2010)

The research field of WSN's has been very active since the early 2000s. The research community has developed many important mechanisms, algorithms and abstractions targeting the special requirements of small interconnected devices. Such as mechanisms for power-saving, since a typical wireless sensors are battery powered and have a long lifetime requirement. Another important mechanism is the WSN's ability to autonomously configure them selves to a network for transporting sensor readings. (Vasseur & Dunkels, 2010)

The lowering cost of sensor technology has made WSN's applicable in many many scenarios. But today WSN's are characterized by high heterogeneity because they consist of different proprietary and non-proprietary solutions. Closed proprietary systems are connectivity islands with limited communication to the external world through application specific gateways. This wide range of incompatible solutions is delaying a large scale deployment of these technologies and creating a virtual wide sensor network that would be capable to integrate all existing sensor networks. (Mainetti, Patrono, & Vilei, 2011) Next two of these legacy protocols for smart objects are described shortly.

3.2 Legacy protocols for smart objects

3.2.1 ZigBee

ZigBee is a proprietary wireless communication specification based on IEEE 802.15.4 radio link layer and it is owned by the ZigBee alliance. The 802.15.4 standard provides a low bit rate and low duty cycle optimized physical and link layer solution, but sensor and control applications also need a mesh networking layer and a standard syntax for application layer messages. The alliance was formed in 2002 to build these missing standard layers needed to enable a multi vendor mesh network on top of 802.15.4 radio links. (Hersent, Boswarthick, & Elloumi, 2011)

The ZigBee architecture consist of five layers: physical (PHY), medium access control (MAC), network (NWK), application support (APS) and application framework (AF) layers. In addition to the five layers the architecture includes a cross-layer entity called ZigBee Device Object (ZDO). PHY and MAC layers are adopted from IEEE 802.15.4 radio standard and not defined by ZigBee specification. (Vasseur & Dunkels, 2010)

Even though the ZigBee stack layers correspond loosely to the those of the IP Stack, it is still incompatible with the IP architecture. This causes problems if ZigBee networks are deployed together with IP-based services and applications. The only way to communicate between ZigBee network and IP-based services is to use a gateway as an interpreter between the two networks. For this reason and to reduce cost of integrating ZigBee networks with IP networks, the ZigBee Alliance announced in 2009 that ZigBee will start move towards IP-based infrastructure. (Vasseur & Dunkels, 2010)

3.2.2 ZWave

Z-Wave is a proprietary protocol architecture intended for automation in residential and light commercial environments. The architecture is developed by ZenSys and promoted by the Z-Wave Alliance. Z-Wave was developed for reliable transmission of short messages between a control unit and one or more nodes within a network. Z-wave architecture defines it's own physical, MAC, transfer, routing and application layers. There are two types of devices in a Z-Wave network: controllers and slaves. Z-Wave functionality is based on the controllers to poll or send commands to the slaves. The slaves then either reply to the controllers or execute given commands. (Mainetti et al., 2011)

3.3 Lightweight IP-based protocols

The use of IP protocol stack for smart objects has many advantages such as interoperability evolvability and scalability. The interoperability of IP comes from it's initial design, that enabled it to work on top of different link layers. The evolvability is due to the end-to-end principle that IP architecture is based on. But when small constrained devices are concerned it needs to be light enough to meet node level constraints. (Vasseur & Dunkels, 2010) Next the building blocks for an IP-stack intended to constrained environments use is described.

3.3.1.1 6LowPAN

6LoWPAN is a new set of IETF standards for Ipv6 over low-power wireless area networks, that is predicted to be a key technology for Wireless

Embedded Internet. The abbreviation WPAN is inherited from IEEE 802.15.4 standard and it originally stood for wireless personal area network. This term is no longer descriptive for the wide range of applications for 6LoWPAN. A more descriptive term nowadays is low-power wireless area network (LoWPAN). (Shelby & Bormann, 2009)

IPv6 enables smart objects to be connected to other IP-based networks, without intermediate entities like translation gateways or proxies. Since LoWPANs have constraints such as limited packet size among others, the use of IPv6 requires and adaptation layer that performs header compression, fragmentation and address auto-configuration. This adaptation layer between IPv6 and 802.15.4 standard has been defined by IETF 6LoWPAN Working group. 6LoWPAN can be used in applications where embedded devices need to communicate with Internet-based services using open standards that are able to scale across large network infrastructures and have mobility. (Mainetti et al., 2011)

The 6LoWPAN architecture consists of LOWPANs connected to other IP networks via edge routers. The edge routes route traffic in and out the LOWPANs and handle 6LoWPAN compression, NeighborDiscovery and IPv4 connectivity mechanisms for the nodes within the LoWPAN. All LoWPAN nodes are identified by unique IPv6 addresses and are capable of sending and receiving IPv6 packets. The nodes use User Datagram Protocol (UDP) as transport protocol and in most cases support ICMPv6 traffic such as ping. The routing in 6LoWPAN networks can be realized with IPv6 Routing Protocol for Low power and lossy networks (RPL). (Mainetti et al., 2011)

3.3.2 RPL

RPL was specified and developed to achieve a reliable communication and high delivery ratio and at the same time to be energy efficient, so it can run on nodes that have limited energy and memory capabilities. Since many devices in Low Power and Lossy Networks (LLNs) are battery powered it is important to limit the amount of sent control messages in the network. Many routing protocols broadcast control packets at a fixed time interval which wastes energy when the network is in a stable condition. For this reason RPL dynamically adapts the sending rate of routing control messages. Routing messages are rarely generated in a network with stable links and more frequently generated on a network in which the topology changes frequently. (Tsvetkov, 2011)

RPL is based on a distance vector routing and network devices running the protocol are connected in a way that no cycles are present. To achieve this a Destination Orientated Directed Acyclic Graph (DODAG) is built. The graph is routed at a single destination called DODAG root. The graph is constructed using an Objective Function (OF) defining how the routing metrics are computed. The node position relative to the DODAG root is called a rank. The rank of a node increase when they move away from the root and decrease when they move towards the root. The rank of the nodes within a network is then used avoid routing loops. (Tsvetkov, 2011)

RPL allows building a logical routing topology over an existing physical infrastructure, enabling network optimization for different application scenarios and deployments. Optimization can be done by constructing a DODAG that considers expected number of transmissions or battery powered nodes in certain parts of the network. (Tsvetkov, 2011)

3.3.3 CoAP

Constrained Application Protocol (CoAP) is an application layer protocol optimized for resource constrained networks. It consists of a subset of the Hyper Text Transport Protocol functionalities, that have been redesigned for low processing power and energy consumption constraints of small embedded devices. CoAP is built on top of UDP and it uses a fixed length binary header of only 4 bytes followed by compact binary options. (Mainetti et al., 2011) Constrained networks such as 6LoWPAN support the fragmentation of Ipv6 packets to small link layer frames. This causes significant reduction in packet delivery probability. For this reason one of the leading design goals of CoAP has been keeping the message overhead small to limit the need for fragmentation. (Shelby, Hartke, & Bormann, 2014)

CoAP provides a request/response interaction model for application endpoints, supports built-in discovery of services and key web concepts such as URIs and Internet media types. It also meets the special requirements of the constrained environments such as multicast support, low overhead and simplicity. Since CoAP is based on a sub-set of HTTP functionalities it is also easily interfaceable with HTTP. (Shelby et al., 2014)

The differences between HTTP and CoAP interaction models come from the typical machine-to-machine interaction where one single CoAP implementation acts in both client and server roles. CoAP request is similar to HTTP request sent by a client requesting an action using a method code, on a resource identified by URI from the server. The server then responses with a response code and depending on the request also a resource representation may be included. (Shelby et al., 2014)

CoAP deals with the request/response interchanges asynchronously over a datagram orientated transport such as UDP, using a layer of messages that support optional reliability. For this goal CoAP defines four types of messages: confirmable, non-confirmable, acknowledgement and reset. Requests and responses can be carried in confirmable or non-confirmable messages and in addition responses can be carried piggybacked in acknowledgement messages as well. (Shelby et al., 2014)

4 Constrained environments

This chapter gives definitions to what do the terms constrained device and constrained node mean. It also describes and classifies the different constraints and doing so provides the first part for the main artifact of this study.

Constrained devices such as sensors or smart objects with limited CPU, memory and power resources are still able to connect to a network. The network it self can be constrained or challenged, with unreliable or lossy channels, based on wireless technologies with limited bandwidth, dynamic topology and relying on a gateway or proxy to connect to the Internet. (Herberg, Romascanu, Ersue, & Schoenwaelder, 2015) An alternative term for a constrained device when the properties of a network node are in focus is a constrained node. (Keranen, Ersue, & Bormann, 2014)

The need for constrained nodes can be justified as how the Internet of Things could be scaled in the future. The scaling of Internet of Things has two aspects:

- Scaling up of Internet technologies to large number of inexpensive nodes, while
- Scaling down the characteristics of the nodes and networks they form to make the scaling up economically and physically viable solution. This need for scaling down on characteristics leads to “constrained nodes”.

A good way to define the term “constrained node” is to contrast it's characteristics to a more familiar Internet nodes. A constrained node lacks some characteristics that are taken for granted in the case of Internet nodes, due to constraints on available energy and physical constraints such as size and weight. This means that the nodes have tight upper bounds on state buffers, code space and processing cycles. Since both processing and transmitting require energy, the optimization of network bandwidth usage and power consumption used in processing are dominating consideration on all requirements. This is not a rigorous definition, but it clearly sets constrained nodes apart from server systems, personal computers and powerful mobile devices such as smartphones. (Keranen et al., 2014)

The constraints of the nodes can be divided into five subcategories:

1. Maximum code complexity (Read only memory/Flash)
2. Size of state and buffers (Random access memory)
3. Amount of computation ability in a period of time (processing power)
4. Available power and energy
5. Lack of user interface and accessibility during deployment (ability to set keys update software etc.) (Keranen et al., 2014)

The power efficiency demand affects the hardware and software design as well as network architectures and protocol designs of constrained nodes. Because communication consumes power it is crucial that the communication patterns are designed so that they use available resources efficiently. The software design is also limited by the often scarce amount of memory, so the software of constrained nodes not only needs to be power efficient, but must have a small memory footprint. These resource constraints that limit the node level have their effect on the network level also. This leads to demands on network protocol design to minimize the amount of network related information each node has to keep and number of transmissions each node has to make. (Vasseur & Dunkels, 2010)

When constrained nodes form a network, it often leads to constraints on the networks themselves. However the networks can have constraints not related to the nodes. For this reason the terms "constrained networks" and "constrained-node networks" have to be independently distinguished. (Keranen et al., 2014)

The next two chapters give more detailed descriptions on constrained devices and networks. These chapters also provide classifications for memory and power constraints.

4.1 Classes of constrained devices

Since a overwhelming variety of Internet-connected devices can be envisioned and even existing today, some kind of classification of constrained devices is needed. Bormann, Ersue & Keränen suggested a three tier classification in their IETF document that reached RFC status in 2014 and has been referred since as a baseline classification. This classification is illustrated in Table 1. They based their classification on distinguishable clusters of commercially available chips and design cores available for constrained devices at the time of writing the document. These boundaries of these classes are expected to move over time, but not as fast as in larger scale of computing. Moore's law tends to be less effective in embedded space and the gains made available by increasing transistor count and density will likely be invested in reduction of cost and power than in increases in computing power. (Keranen et al., 2014)

TABLE 1 Classes of Constrained Devices

Name	Data size (RAM)	Code size (ROM/Flash)
Class 0, C0	< 10 kB	< 100 kB
Class 1, C1	~ 10 kB	~ 100 kB
Class 2, C2	~ 50 kB	~ 250 kB

Class 0 devices are very constrained sensors, with so severe memory and processing constraints that they are unable to communicate directly with the Internet in a secure manner. Class 0 devices need the help of larger devices acting as proxies, gateways or servers to participate in Internet communications. Generally they cannot be secured or managed comprehensively in the traditional sense, but they will likely be preconfigured and will be rarely reconfigured, if at all. (Keranen et al., 2014)

Class 1 devices are quite constrained in code space and processing capabilities. They are not able to employ a full Internet protocol stack and not able to communicate to other nodes using HTTP, Transport Layer Security (TLS), other related security protocols and XML-based data representations. Instead Class 1 devices are capable enough to use a protocols stack designed for constrained nodes including CoAP over UDP and special implementations of Datagram Transport Layer Security (DTLS). This enables them to communicate without the help from a gateway node, so they can be integrated as fully developed peers of an IP-network. But their state memory, code space and often also power expenditure set limits to protocol and application solutions. (Keranen et al., 2014)

Class 2 devices are less constrained and so capable of supporting most of the protocol stacks of normal Internet nodes. However even in this level the devices can often benefit from lightweight and energy-efficient protocol usage and from consuming less bandwidth. The use of protocol stack defined for more constrained devices on Class 2 device leaves more resources available for applications, since they will be using fewer resources for networking. This might also reduce development costs and increase interoperability. Devices significantly beyond minimum level of Class 2 are less demanding on the protocols used, but can still be constrained by a limited energy supply. (Keranen et al., 2014)

4.1.1 Classifications based on energy limitation

As mentioned earlier the available power and energy is also a limiting factor for constrained devices. The power and energy available to a device can differ from kilowatts to microwatts and from unlimited to hundreds of microjoules. Watts determine the sustainable average power available for the device over the time of it is functioning. Joules determine the total electrical energy available before the energy source is exhausted. Devices can be limited both in available energy and available power. Bormann, Ersue & Keränen (2014) describe a four level classification for energy limitations that is illustrated in Table 2. (Keranen et al., 2014)

TABLE 2 Classes of energy limitation

Name	Type of energy limitation	Power source example
E0	Event energy-limited	Event-based harvesting
E1	Period energy-limited	Periodically replaced or recharged battery
E2	Lifetime energy-limited	Non-replaceable primary battery
E9	No limitations to available energy	Mains-powered

Devices classified as E0 have limited amount of energy available for a specific event, such as a button press in an energy-harvesting light switch. E1 classified devices have a energy limitation based on a specific period. Examples of this kind of devices are a solar powered device with limited energy stored for night, device that is manually connected to a charger or a device that needs it's battery replaced in certain intervals. E2 device has an total energy limitation for its usable lifetime and it may be discarded when its non-replaceable primary battery runs out. When no relevant limitations to energy exist the device is classified as E9. (Keranen et al., 2014)

In the case of wireless devices the radio transmissions cause a big portion of the total energy consumption of the device. The parameters of the radio transmissions influence the power consumption during transmission and reception. These parameters include the available spectrum, desired range and the bit rate. The duration and number of transmission and reception including waiting for incoming messages influence the total energy consumption of a device. Depending on the energy source and communication frequency different strategies for power usage and network connectivity may be used. (Keranen et al., 2014)

There are three strategies in the device level for power usage and they can be described as follows.

Always-on: No need for power saving measures, so the device can stay on and connected to the network all the time.

Normally-off: The device sleeps long periods and reconnects to the network when it wakes up. In this strategy the main area of optimization is to minimize the effort needed for the reattachment process and resulting application communications. If the device needs to communicate infrequently, the increase in energy expenditure during reattachment may be acceptable.

Low-power: This strategy is suitable when devices need to operate on small amount of power, but still need to communicate in relatively frequent basis. This strategy requires that low-power solutions are also available in the hardware and and link-layer mechanisms. These devices retain their attachment to the network in some form, despite they may have a relatively short sleep period between transmissions. This strategy minimizes the power usage needed for reestablishing communications. An example of this strategy is duty cycling where components are switched on and off in a regular cycle.

4.2 Constrained networks

Bormann, Ersue & Keränen (2014) define a “constrained network” as a network where some of the characteristics taken for granted with link layers in common user in the Internet are not attainable. These constraints include:

- Low bit rate/throughput including limitations from duty cycling.
- High packet loss and packet loss variability, that causes low delivery rate.
- Highly asymmetric link characteristics.
- Using larger packets causes high packet loss due to link-layer fragmentation.
- Limits on reachability of nodes over time, since devices may power off and be able to communicate for brief periods of time.
- Lack of or severe constraints on advanced services such as IP-multicast.

The term constrained network is used when at least some of the nodes in the network have some of these characteristics. The reasons behind the constraints may be one or several of the following:

- Network cost constraints
- Node constraints, this concerns constrained-node networks.
- Physical constraints, such as power constraints, environmental constraints, media constraints etc.
- Regulatory constraints, such as limits on spectrum availability and radiated power in a region of the world or industry such as explosion safety.
- Technology constraints, such as heritage lower-speed technologies still operational.

4.2.1 Constrained-node network

Constrained-Node Network is a network composed of significant part of constrained nodes, which give the network constrained characteristics, so a constrained-node network is always a constrained network. It may also have other constraints in addition of consisting of constrained nodes. (Keranen et al., 2014)

4.2.2 Summary

In summary constrained environments can have constraints from two main sources: the devices them selves and the from the network they use. The constraints described in this chapter are gathered in Table 3. These constraints are the first building block for the main artifact of this study.

TABLE 3 Constraint summary

Constraint	Description
Maximum code complexity	ROM/Flash constraints limit the size of algorithms needed for computing complex tasks such as asymmetric keys. The minimum amount of ROM for secure communication is ~100 kB. This constraint is described by the three classes of devices based on available ROM and RAM. By this classification the amount of ROM and RAM need to be considered together (see next constraint on this table).
Size of state and buffers	RAM constraints limit the size of keys and other state buffers needed to hold in the working memory. The minimum amount or RAM for secure communication is considered to be ~10 kB.
Processing power	Amount of computation ability in a period of time effects to the latency of complex calculations and may render too complex calculations demanded from the constrained nodes non applicable.
Available power and energy	Transmitting is identified as the dominating factor on energy consumption. For this reason the network transmissions required from energy constrained nodes should be minimized. The level of power saving measures is dictated by the available energy.
Network constraints	Throughput, delivery rate and node reachability are the considerations that need to be taken into account. Since network constraints can also effect the security mechanisms used.
Lack of user interfaces	Sets limits to accessibility during deployment and ability to set keys and update software during the device lifetime.
Physical constraints	Size and weight of the device.
Cost	The cost of a constrained distributed system is also a constraint to consider when building a business case around an application.

This chapter also describes two ways the for classifying the device constraints. The first classification is made based on available memory and second based on power and energy limitations. These classifications can be found from Tables 1 and 2.

The device constraints pose several limits to choosing an authorized authentication mechanism design. First the memory constraints limit the size of used algorithms and encryption keys. Second the processing power also limits the applicability of more complex algorithms. And third the demand for energy efficiency brings a need for minimizing the number of transmissions.

In addition to three groups of device constraints, the network constraints and lack of user interfaces should also be considered. Network constraints can cause larger latency for requests and other network related challenges that can effect the authentication mechanism. The lack of user interfaces needs to be taken into consideration when the constrained devices are deployed. For example all necessary keying material to initial secure connections have to be set when the devices are deployed.

The physical size and cost of the devices are very much dependent on the other device properties. For example a larger on board battery capacity is in many cases the key factor to the devices physical size and cost is heavily effected by processor and memory capacity.

5 Architecture for authorization in constrained environments

This chapter is based on IETF Authentication and Authorization for Constrained Environments (ACE) working group architecture. It provides objectives on specific features that need to be taken into account when a authorized authentication mechanism is designed to be used in a constrained environment. The objectives presented in this chapter provide the second part for the main artifact of this study.

Since constrained nodes have limitations they may not be able to perform all necessary tasks required by the complex security mechanisms. As a solution to this problem it is proposed that more demanding tasks are assigned to other less constrained entities to achieve the required security level in constrained scenarios. The functionalities need to be grouped by their demands on the platform i.e. can they be assigned to a constrained level device or a less-constrained level device. In this chapter the architecture for authentication and authorization in constrained environments in this manner is described and the needed elements and relationships of between functionalities are identified. (Gerdes, Seitz, Selander, & Bormann, 2015a)

First some assumptions for this architecture needs to be made in a general level. The primary aim of the architecture is to control and protect a resource-based interaction between two potentially constrained endpoints. One endpoint is not limited to hosting the functionalities belonging to just one actor. Client is determined as an endpoint that requests an access to a resource that is hosted on another endpoint called the resource server. These two endpoints may not have a security setup readily available, so one has to be established. (Gerdes et al., 2015a)

5.1 Actors and their tasks

Actors in this architecture should be considered as a concept to understand the security requirements for constrained devices. Actors are not synonymous to devices, but a single device or piece of software can include the functionalities of several actors. The actors consist of a set of tasks they perform. Actors are grouped to three levels: principal, less-constrained and constrained levels. Another defining factor is the actors security domain that can be different between actors. For example if the client and resource reside in different security domains. (Gerdes et al., 2015a)

5.1.1 Constrained level actors

First lets look into the constrained level actors client and resource server and define their tasks. Both client and resource server need to have an ability to communicate in a secure way and validate each others authorizations to perform the tasks needed for the transaction. The client has to validate that the resource server is an authorized endpoint for the resource it wants to access. On the other end resource server needs to validate that the client requesting a resource has an authorization to do so.(Gerdes et al., 2015a)

5.1.2 Less-constrained level actors

Less-Constrained level actors task is to assist the constrained level actors by relieving them from computationally intensive and memory demanding tasks such as managing keys for numerous endpoints. On the Client side this actor is called Client Authorization Server (CAS) and on the Resource Server side actor is called Authorization Server (AS). These less-constrained actors belong to the same security domain and function under the same principal as their constrained counterparts. They act on behalf of their principals and as authorities for claims about the constrained level actors residing in the other security domain. (Gerdes et al., 2015a)

The Client Authorization Servers function is to authenticate the Resource Server and determine if it is an authorized server for the Resource in question. The tasks to achieve this are to validate that entity attributes, mediate the authorization information between Requesting Party and the Client and handle the negotiation process needed for secure communication in behalf of the Client. (Gerdes et al., 2015a)

The Authorization Servers main function is to authenticate a Client on behalf of the Resource Server and determine the Clients permissions to the different Resources. The sub tasks are similar with the client side: validate the entity attributes, mediate authorization information between Resource Owner and Resource Server and handle the secure communication establishing process. (Gerdes et al., 2015a)

Many use cases for constrained devices portray a scenario where principals are not present at the time of communication, can't communicate directly or prefer the device to communicate autonomously for some other reason. In such cases the principal requires an agent to maintain the policies set for the endpoint interaction it governs. Namely Authorization Server must be able to act on behalf of the Resource owner handling access request on behalf of

Resource Server and Client Authorization server making resource request and handling their responses on behalf of the Client. (Gerdes et al., 2015a)

5.1.3 The principal level actors

On the principal level the client and resource server are controlled by an individual or a company. The principal controlling the client is called Requesting Party (RqP) and resource server and the actual resource are controlled by Resource Owner (RO). Requesting Party and Resource Owner are in charge of specifying security policies to the constrained level actors and therefore by definition Client and Requesting Party belong to the same security domain as do Resource Server and Resource Owner. Requesting Party configures C for authorization information for allowed sources for a resource and the Resource Owner configures the Resource Server for authorization information for accessing a Resource. (Gerdes et al., 2015a)

As said the principal level actors make the authorization decisions and specify them to the less-constrained level actors by encapsulating them in to security policies. These policies are then enforced by the constrained level actors. Security objectives in the principal level that are valid for any scenario can be divided into two types on basis that they concern the Client or Resource Server side. The first concerns the resource server side: all entities that gain access or knowledge of a Resource have to be authorized by the Resource Owner. The Client side security objective is: Client conducts exchanges, including requesting data or accepting a response only from resources authorized by the Requesting Party. (Gerdes et al., 2015a)

The architecture described above and it's three levels are illustrated in Figure 3. Note that the vertical arrows in this figure are not representative of information flows but illustrate exerted control or provided support. Nor do the arrows indicate a necessary connection during communication. As described previously the principal level actors are not always present and in such cases the authorization servers act autonomously.

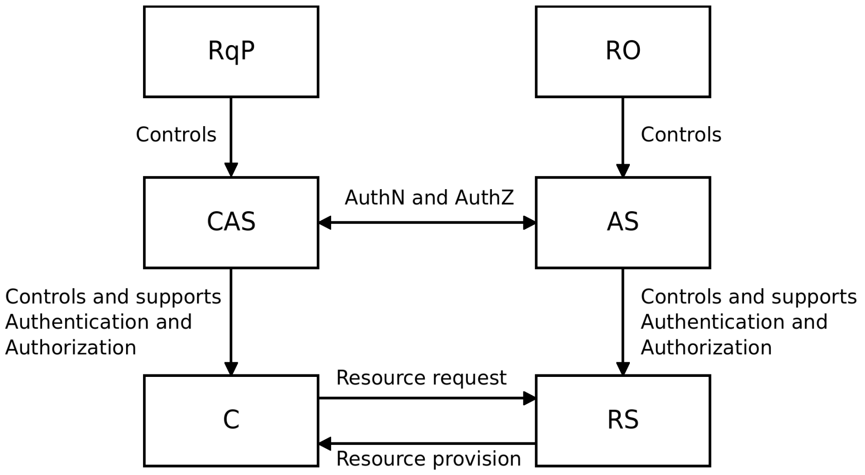

RqP
RO
Controls
Controls
CAS
AuthN and AuthZ
AS
Controls and supports Authentication and Authorization
Controls and supports Authentication and Authorization
C
Resource request
RS
Resource provision

Figure 3: Overall architecture (Gerdes et al., 2015a)

5.1.4 Possible role combinations

Elements described here are purely architectural. Depending on implementation and the device capabilities, several elements can reside in a single device or even a single piece of software. For example if Client or Resource Server are located in a powerful enough device their functionalities can be combined with their Authorization Servers. (Gerdes et al., 2015a)

Another good example of combining the functionalities of two actors is a situation where the Client and Resource Server have the same principal i.e. Requesting Party is the same individual or company as Resource Owner. In this kind of situation Client and server side authorization servers can be combined as one entity if desired. (Gerdes et al., 2015a)

5.2 Information flows

When reviewing the information flows the architecture focuses on we need to take note of the fact that the message flow may pass unprotected paths. The interaction between principal level actors is not a concern since existing mechanisms can be employed. The critical points in the message flow are the messages between constrained nodes and constrained nodes and their less constrained counterparts. These messages can after all contain permissions, client and server attributes, conditions on resources, not to mention keys and credentials. This control information may also be rather asymmetric in the client and server side. (Gerdes et al., 2015a)

The architecture assumes that the necessary credentials are provided for the control information flows between constrained and their less-constrained counterparts and it needs to be part of a solution. The problem statement for authorization in constrained environments is derived from the information flows and can be summarized by three major points:

1. Less-constrained nodes control the interaction between potentially constrained nodes on behalf of their principals.
2. The interaction needs to be secured between endpoints in end-to-end manner, including scenarios with intermediary nodes, including necessary key establishment.
3. Transferring control information needs to be secured in an end-to-end manner, including scenarios with intermediary nodes. This may require employment of pre-established keying material. (Gerdes et al., 2015a)

The first potential information flow is a push sequence initiated by Client, first acquiring credentials using CAS and presenting them to RS is presented in Figure 4.

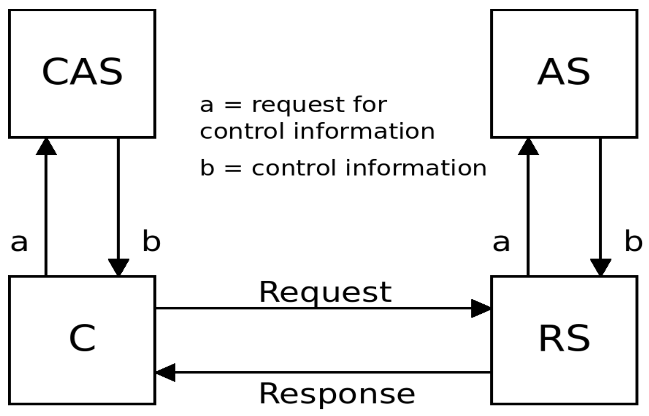


Figure 4: Information flow (Gerdes et al., 2015a)

But the architecture does not imply that this would be the only possibility. It states *"Authorization information is transferred from AS to RS using Agent, Push or Pull mechanisms" (Gerdes et al., 2015a)*. This implies that another potential information flows would be possible. Such as a pull sequence. Again initiated by C, but using RS as intermediary for authorization with AS, which then communicates to CAS.

5.3 Summary

The architecture is intended to be used as a framework for designing authorized authentication mechanisms for constrained environments. It does not pose tight restrictions on such things as authorization sequence or in which actual device the different elements of the mechanism reside. The architecture can so be seen as a set of objectives on how to realize such mechanism. Table 4 summarizes the objectives gathered from the architecture. These objectives are the second building block for the main artifact of this study.

TABLE 4 Architecture objective summary

Objective	**Description**
Delegation of demanding tasks	Demanding tasks are assigned to other less constrained entities. Functionalities should be grouped by their demands on the platform to determine if they should be assigned to a constrained level device or a less-constrained level device.
Validation of actors	• The client has to validate that the resource server is an authorized endpoint for the resource it wants to access. • Resource server needs to validate that the client requesting a resource has an authorization to do so. • The Client Authorization Server needs to validate that the Resource Server is an authorized server for the Resource. • The Authorization Servers needs to validate the Client on behalf of the Resource Server and determine the Clients permissions to the Resource.
Autonomous functionality	• Less-constrained nodes control the interaction between potentially constrained nodes on behalf of their principals. • Authorization Server must be able to act on behalf of the Resource owner handling access request on behalf of Resource Server. • Client Authorization server making resource request and handling their responses on behalf of the Client.
End-to-end security	The interaction needs to be secured between endpoints in end-to-end manner, including scenarios with intermediary nodes, including necessary key establishment and control information.

6 Security concerns

This chapter deals with common security concerns for a distributed system that need to be achieved even in constrained environments. It also discusses the different options on choosing an authentication sequence and authentication method. This chapter provides the third part for the main artifact of this study.

Constrained devices are designed to be small, inexpensive and easily integratable. By this definition they are meant to be used in various environments. They can be in control of important functions and have access to large amount of valuable data. In this kind of scenario the need to protect from unauthorized access is self-evident, but there are other scenarios to take into consideration. Even gathering seemingly innocuous data and information of functions can lead to insights of a system that can be used to gain some level of control. Another scenario could be that the devices themselves are not the primary target, but they are used as an intrusion point to infiltrate the network and used to attack other more valuable devices. Because constrained devices have limited capabilities they can be the weakest link in a network and hence an attractive target. (Selander, Mani, Kumar, Seitz, & Gerdes, 2015)

First objectives for a secure and reliable conduct for a distributed system has to be defined. Pfleeger & Pfleeger (2002) address the security goals of a computer-related system with three aspects. These aspects are confidentiality, integrity and availability described in Table 5.

TABLE 5 Security objectives for a computer-related system

Objective	**Description**
Confidentiality	Ensuring that assets are accessed only by authorized parties. Access includes not only reading privileges, but also viewing, printing and knowledge of the existence of an asset.
Integrity	Only authorized parties have the ability to modify the assets. Modification includes writing, changing, changing status, deleting and creation.
Availability	All authorized parties have access to the assets, at the time they need it.

These three objectives are considered as hallmarks of solid security. Their first appearance in literature is found as early as 1972 in James P. Anderson's essay in computer security (Anderson, 1972). These properties are referenced as the C-I-A triad or the security triad. (Pfleeger, Pfleeger, & Margulies, 2015) These properties define how resources should be secured.

Nguyena, Laurentb & Oualhaa (2015) propose five objectives in their definition for security objectives of IoT: confidentiality, integrity, authentication, authorization and freshness. These properties are centered on data shared between devices and they address message security (confidentiality, integrity, authentication and freshness) and access rights (authorization). (Nguyen, Laurent, & Oualha, 2015) These objectives are summarized in Table 6.

TABLE 6 Security objectives for IoT

Objective	**Description**
Confidentiality	Exchanged messages in the IoT may need to be protected. An attacker should not gain knowledge about the messages exchanged between a sensor node and any other Internet entity.
Integrity	The alteration of messages should be detected by the receiver.
Authentication	The receiver should be able also to verify the origin of the exchanged messages.
Authorization	IoT devices should be able to verify whether certain entities are authorized to access their measured data. At the network layer, only authorized devices should be able to access the IoT network. Unauthorized devices should not be able to route their messages over the IoT devices, because it may deplete their energy.
Freshness	This property ensures that no older messages are replayed. This is important to secure the communication channel against replay attacks.

Both taxonomies can be seen congruent by content, but the grouping of the objectives take different perspectives. The second taxonomy adds more IoT specific features dealing with more message centric issues. Such as alteration detection and origin verification for the messages. It also acknowledges the possibility of replay attacks as a separate property. These issues can be dealt with securing the communication between devices and enforcing authorized authentication. Next chapters describe different methods for communication and resource security.

6.1 Communication security

There are alternatives on how communication security can be achieved. These alternatives include session based security at the transport layer, object security at the application layer or hybrid solutions that utilize both session-based and object security. Session-based security offers security, integrity and confidentiality protection for the whole application layer exchange. But it may not be able to provide end-to-end security over multiple hops. It also requires usage of a handshake protocol that has high memory and power demands that can be too expensive for constrained devices. Object security is a more flexible communication model, but it's problem is that it can not provide confidentiality for the message headers. Hybrid solutions can be built so that key exchange is handled with secure data objects, but session-based security is used for resource access. (Gerdes et al., 2015a)

In addition to attacks focusing on monitoring and altering the data transit constrained devices are specially vulnerable for Denial of Service (DoS) attacks. These attacks can result not only temporary failure in a service, but also more permanent failures, for example draining the battery by repeated flooding with connection attempts. For these reasons authorization solutions should take into account such things as power usage, number and size of authorization messages needed, protocol code size and memory requirements. (Selander et al., 2015)

There are several proposed techniques for securing IP-based constrained network traffic that utilize 6LoWPAN and CoAP. CoAP standard defines a binding for Datagram Transport Layer Security (DTLS), but this solution is based on a setting up the security information during the provisioning phase. (Shelby et al., 2014)

DTLS is a common Internet protocol, so it does not consider the needs of constrained devices by design and use a heavyweight, protocol like this in a constrained environment propose certain problems. One of them is the 127 byte Maximum Transmission Unit (MTU) of 802.15.4 standard which 6LowPAN stack uses as data link and transmission layer. For a constrained device sending and receiving messages is much more expensive than local processing, so for this reason it is proposed that 6LoWPAN compression is used also for DTLS headers and messages to minimize number of messages needed. The 6LoWPAN standard defines header compression and fragmentation schemes to overcome the small MTU. For compression the standard defines an IP Header Compression (IPHC) for the IP-header and also a Next Header Compression (NHC) for the IP extension headers and the UDP header. These compression mechanisms can be used to compress security protocols as well. (Raza, Trabalza, & Voigt, 2012)

DTLS can be applied in an end-to-end manner. An end-to-end security protocol can provide security regardless of the underlying network infrastructure. DTLS protocol is placed between transport and application layers, so it does not rely on infrastructure provider to support the security mechanism. The security establishment is left to the two communicating applications. DTLS assumes an unreliable transport, but it is still possible to use DTLS over a reliable transport such as TCP. (Kothmayr et al., 2013)

6.2 Authorized authentication

The terms authentication and authorization together define access rights and "who can do what" for given object. Authentication addresses the who part of the equation and authorization binds the who and permission to the object in question, so defining if permission is granted or denied. Authentication is in most cases done using credentials. Types of credentials can vary from long -lived generic authorizations and some more specific credentials directed to authorize specific actions. In some cases satisfactory level of authenticated authorization can be achieved by static authorization and there may be no need to change the credentials after manufacturing or deployment. But many use cases demand for more flexible and fine-grained access control policies. (Gerdes et al., 2015a)

There are also privacy considerations to take in to account. Since in many cases the purpose of constrained devices is to collect data from or affect their surroundings in the physical world. Collected data can often be associated with individuals. Privacy protection and a guarantee is needed that only authorized entities with prior consent from resource owner are allowed to access the data and trigger actions. (Selander et al., 2015)

These requirements for authorized authentication can be divided into smaller pieces to better understand what is needed to achieve this goal. So next authentication and authorization and their prerequisites are dealt with.

The key part of authentication is identity. So what is the definition of identity and since we are in the digital realm, more precise term we are looking for is digital identity. Both ITU (ITU-T, 2009a) and IETF (Shirey, 2007) documentation define similar characteristics for the term. They both define identity to be a set of characteristics presented as attribute values, that are sufficient to positively identify an entity within context. An entity in both definitions is left as an ambiguous term that can be applied to a user or other system entity such as a device or a piece of software.

Authorization is defined in the IETF Internet Security Glossary (Shirey, 2007) as "An approval that is granted to a system entity to access a system resource" or "A process for granting approval to a system entity to access a system resource". ITU-T-800 (ITU-T, 1991) definition is quite similar it states that authorization is: "The granting of rights, which includes the granting of access based on access rights".

There are plethora of different architectures for handling access control with different mechanisms to handle authentication and authorization. Few of them are described in the next chapters.

6.2.1 Identity based access control

The digital identity is always based on finite number of attributes and should not considered as a holistic identity of an entity. Holistic identity is a theoretical issue with indefinite number of attributes and it should be considered as an theoretical issue. (ITU-T, 2009a)

We are always dealing with a subset of attributes of an entity when digital identity is being authenticated. In many cases a full set of attributes is not even necessary to achieve sufficient level of identification. The necessary level of identification information always dependent on the context. But in any case sufficient identity information to uniquely recognize an entity within a context to the extent that is necessary for the relevant applications is needed. So even generic attributes, such as location or liaison to an organization can be enough to identify an entity to adequate extent. (Chadwick, 2009)

This brings us to the question how identities are handled in information systems. The process of handling identity attributes is called identity management. ITU defines identity management as a set of functions and capabilities such as administration, management and maintenance, discovery, communication exchanges, correlation and binding, policy enforcement, authentication and assertions. These functions and capabilities are the mechanisms used in assurance of identity information and the identity of an entity and so enabling security applications. (ITU-T, 2009a)

In the system level there are different ways to deal with identity management. In a centralized system an entity presents its identifier and some form of authentication such as password or authentication token. The authentication acts as a proof that the entity is entitled to be known by its identifier. If the system accepts this authentication it associates the entity with the given identifier, including all attributes linked to this identifier and grants the entity access to the system. In a distributed system each local system has its own set of identifiers, so an accessing entity needs to have credentials to every local system in order to access them. (Chadwick, 2009)

Single sign on (SSO) is there fore desirable solution for fighting complexity in distributed systems. First designs to achieve SSO were based on public key infrastructure (PKI) that allocated each user a globally unique identifier. This identifier would be known by all local systems in a distributed system and used to grant access. The weakness of this kind of system was that everyone had to know everyone else's globally unique identifier and this raised obvious security concerns. (Chadwick, 2009)

The next breakthrough was when it was realized that there was no need to use global identifiers. Instead the identifiers could remain local to the system that allocated them. This was achieved using a federation between local and remote systems for authorization. In federated systems authorization is granted based on identity attributes, rather than globally unique identifiers. The service provider can verify an entity based on identity attributes provided by trusted authoritative sources. These systems are known as federated identity management systems. (Chadwick, 2009)

A federation is defined in ITU document X.1250 (ITU-T, 2009b) as ”An association of users, service providers and identity providers”. This means that a group of service providers have formed an association and trust between themselves and are willing to exchange messages. Further more if these messages are also used to exchange authentication and authorization credentials between systems. So allowing access to another system using credentials verified by another federated system, we have federated identity management (FIM). FIM enables SSO, but it also brings other benefits, such as offloading the load of managing user attributes and easier scalability. (Chadwick, 2009)

But even though FIM systems give the ability to SSO and simplify things compared to locally administrated access control, it still has its shortcomings. These problems originate from the fact that only the identity is federated and the authorization of resources is left remote system.

6.2.2 Authorization base access control

In an Identity Based Access Control (IBAC) described in the previous chapter an entity request a resource from a service and includes a proof of identity with the request. The service that was called submits this information to its policy engine to determine the authorizations, so the service bases its access decision on this authorization not the provided identity. (Karp, 2006)

In an Authorization Based Access Control the only thing that changes is that the local policy engine used to determine the authorizations before a resource request to the service. The calling entity then presents the appropriate authorizations to the service along with the resource request. This way the only function left to the remote system is to verify that the authorizations presented haven't been forged. (Karp, 2006)

Authorization based access control has many advantages and the major one is that each domain only needs handle the information about its own principals. The user entities only authenticate to their own domains, which makes system updates and authorization formats simpler and makes the systems more scalable. This architecture also makes the systems more secure and private, since the systems only needs to interpret the contents of the authorizations and no organizational information leaks between domains. System manageability becomes easier because authorization delegation is much simpler between entities. (Karp, 2006)

6.2.3 Capability based access control

Traditional access control models that are based on authorization lists or group authorizations have big challenges when they are used in large distributed systems. This problem comes from the requirement of a consistent definition of complex access policies. To overcome this inherent complexity of traditional access control models, some alternative approaches have been studied. One of these approaches is Capability Based Access Control (CapBAC). (Hernández-Ramos, Jara, Marın, & Skarmeta, 2013)

A capability based authorization was first proposed to address the dynamicity and scalability issues in an environment where services are independently made available by service providers. The proposal was made by Skinner (Skinner, 2009) as a part of Digital Ecosystem (DE) environment promoted by SUN. DE environment provides facilities for searching services by their characteristics and accessing them based on the needs of the user. The proposal intended to add basic services specific mechanism for the user to gain an access token to a service the user needs, but has never used before. DE environment does not assume environment wide access control mechanisms or federated identity management to be used. The environment description implies that each service is potentially autonomous and users needs could be highly dynamic, so a more permanent link was not necessarily needed. (Gusmeroli, Piccione, & Rotondi, 2013)

A capability is a communicable token of authority that is unforgeable and associated to a set of access rights. The name is devised from the capability the token gives to a process for interaction with an object in predetermined way. In this model an entity has to present an authorization capability, by proving it has the ownership of a token to the service provider for gaining access to resources. Compared to access control list models where the service provider verifies the authorization rights to each requested resource for each requester. This approach brings benefits in the form of simplified delegating of access rights, capability revocation and information granularity. Delegation support means that a subject can grant rights to another subject and also grant the rights for further delegation. Delegations can concern the whole set of rights granted to the original subject or only part of them. Capability revocation can be done by properly authorized subjects at any time. Information granularity means that a capability can specify the level of granted rights dynamically and there fore refine the level data access granted by a capability token. (Gusmeroli et al., 2013)

In capability based systems access control policies and authorization procedure simpler from the service point of view. Since the entity that wishes to access a resource sends a token together with the request, the receiving entity already knows the right level of permissions the requester has been granted. (Hernández-Ramos et al., 2013)

There are also some challenges facing capability based authorization. First is the large amount of capabilities needed to issue and requesting them when a resource is accessed. The second is to standardize the use of capability tokens for cross-domain use. The first problem can be mitigated by setting up services for granting capability policies, that can also generate on the fly access for properly identified and authorized users. (Gusmeroli et al., 2013)

6.3 Authentication message sequence models

There are three message sequence models to consider when a secure communication link needs to be set between a client and a service providing a resource. These sequence models are described next. The models are derived from AAA Security Framework specification (Farrell et al., 2000), but for continuity the names used for the actors are from ACE WG architecture.

Agent sequence: The acts as an agent between the Client and Resource Server. The Authorization Server receives a request form the Client and forwards the authorization and configuration to the Resource server. In this model the Authorization Server after receiving a request authenticates and applies authorization policies for the client and for the particular service it has requested. (Farrell et al., 2000) Agent sequence is presented in Figure 5.

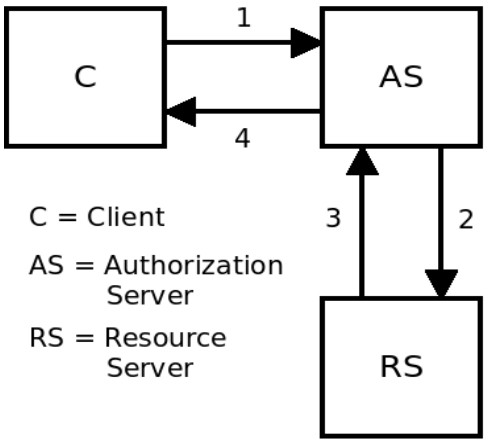


Figure 5: Agent sequence (Farrell et al., 2000)

Pull sequence: The Client sends a request to the Resource Server, which forwards it to the Authorization Server. Authorization server returns an response to the Resource Server after validating the Client and request. If the request is accepted the Resource Server is ready to provide the requested service. (Farrell et al., 2000) Pull sequence is presented in Figure 6.

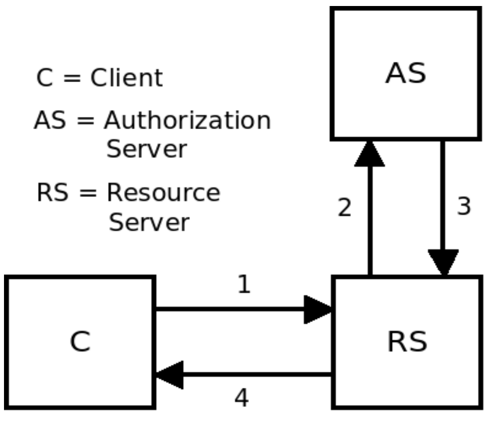


Figure 6: Pull sequence (Farrell et al., 2000)

Push sequence: The Client requests a certificate or a ticket verifying it has authorization to access the service. After obtaining the ticket the Client makes a request for a service and includes the ticket with the request. After receiving the ticket the Resource Server is ready to provide the requested service. (Farrell et al., 2000) Push sequence is presented in Figure 7.

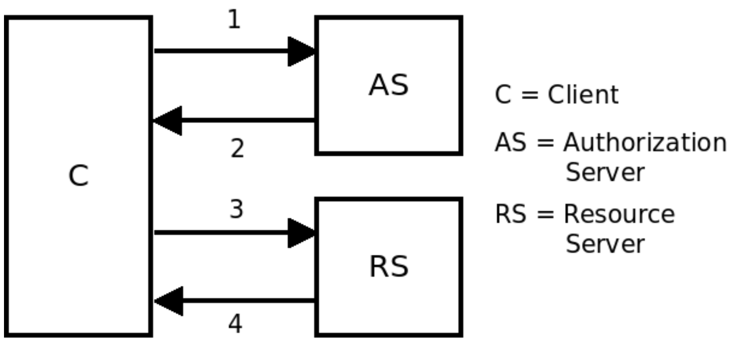


Figure 7: Push sequence (Farrell et al., 2000)

The same message sequences are possible in a simple scenario where all the actors are part of the same security domain, as well as in roaming scenarios. The scenarios involving different security domains are a little bit more complicated, but the main prerequisite is that the client has to have a relationship with an Authorization Server able to verify its authorization for the service.

6.4 Summary

The security aspects for a distributed systems need to be considered even when the system includes constrained devices. Constrained environments also have some specific security issues. Such as they are specially vulnerable for denial of service and replay attacks on the counts of limited processing and networking capabilities and also because such attacks can shorten the lifetime of battery powered devices considerably. The main security objectives identified in this chapter are gathered to Table 7. These security objectives are the third building block for the main artifact of this study.

TABLE 7 Security objective summary

Objective	**Description**
Resource confidentiality	The access for the resources should only be possible to authorized parties. Access includes reading and viewing, but also actions requested from the node in question. Confidentiality of a resource could be extended to so far as only authorized parties should even have knowledge of the resource.
Resource integrity	Only authorized parties should have the ability to modify the resources. Modification includes writing, changing, changing status, deleting and creation.
Resource availability	All authorized parties have access to the resources, when needed.
Message confidentiality	The exchanged messages need to be protected. A possibly malicious third party should not gain knowledge of the message contents or any other information about the messages exchanged between nodes or between nodes and entities in the Internet.
Message integrity	The alteration of messages should be detected by the receiver.
Message authentication	The receiver should be able to verify the origin of the received messages.
Message freshness and authorization	The replaying of old messages should be prevented to secure the communication channel against replay attack. At the network layer, only authorized devices should be able to access the IoT network. Unauthorized devices should not be able to route their messages over the IoT devices, because it may deplete their energy
Access control method	Identity management for systems including constrained devices should consider the access control method to be used. This kind of distributed systems will benefit from federated identity and authorization management.
Message sequence for authentication	Message sequence used between the constrained actors and their less constrained counterparts can have effect on the level of identifying the actors.

First three objectives derived from the C-I-A triad concern the authorization issues of resources. These objectives set the requirements for authorized resource access that form resource security.

The next four objectives describe the different secure communication. This includes confidentiality, integrity and authenticity of the messages which are generic requirements for any type of distributed system. The message freshness and authorization considers the vulnerability to replay and flooding attacks, which is have a larger potential effect in constrained environments.

Access control method and authentication message sequence are architectural decisions that need to be considered when building a authorized authentication mechanism for constrained environments. For example list based access control in large distributed system will raise the complexity and inflict too much strain on the actors enforcing it. For this reason using some of the federated approaches to access control is advisable, specially when access control is enforced by a constrained device.

Message sequence for authentication objective adds more definition to the validation of all the actors objective presented in architecture based objectives. The architecture based objective identifies the need to identify the actors based on the functions they need to accomplish. As selected message sequence effects such things as the ability of actors to verify each others identities and the demand for connectivity of different actors. These features are different between sequences. For example if a push sequence is used and Client and Server have their own authorization servers. In this kind of authentication sequence the Client negotiates its authorization trough authorization server in its own security domain. This means the authorization server in charge of the resource has no direct way to authenticate the Client directly. The difference with the other two sequences described is, that in both agent and pull sequences the Client credentials are available at least in some level to the authorization server in charge.

7 Proposed protocols

This chapter describes two protocols that have been proposed for authorized authentication in constrained environments in past ACE Working Group proceedings. Other protocols have been proposed also, such as OAuth 2.0 which is currently on the standards track (Erdtman, Seitz, Wahlstroem, Selander, & Tschofenig, 2016). These two protocols were selected since they have different approaches to the problem and so provide basis for evaluating the evaluation framework.

7.1 DCAF

Delegated CoAP Authentication and Authorization Framework (DCAF) relieves the constrained devices form the load of handling public key based authentication and authorization. These resource consuming tasks are delegated to authorization servers that handle these tasks behalf of their constrained counterparts. DCAF is designed to be used with CoAP application protocol and DTLS is used to secure transmission of authorization information between constrained devices enabling end-to-end security. (Gerdes, Bergmann, & Bormann, 2014)

7.1.1 DCAF objectives

The objectives of DCAF are twofold, consisting of communication scenario and authorization related tasks. The communication scenario describes the basic interaction of devices and concerns only the basic actors. Authorization related tasks describe the security related requirements set for the message exchange. Basic functional scenario for DCAF is a communication between a Client (C) and Resource Server (RS) is:

1. C wants to access a Resource (R) on a RS.
2. C and RS do not necessarily have an existing security relationship.
3. C and/or RS are Level 1 Constrained devices.

4. Unauthorized entities should not gain knowledge or access to R
5. Confidentiality and integrity of R are needed and authenticity of messages concerning R must be assured. (Gerdes et al., 2014)

Authentication and authorization in the context of DCAF include several attributes to validate. The authorization related tasks include:

1. Attribute binding: a verifiable attribute must be bound to a verifier such as a key. This handled by an attribute binding authority, checking that the attributes of the entity possessing a verifier match the ones it claims to have. If the attributes are correct the authority provides endorsement information to be used by other entities to validate the binding.
2. Verifier validation: when an entity needs to authenticate another entity it checks the attribute-verifier binding using an endorsement from the claim validation authority.
3. Authentication: An entity or the source of information must be authenticated using the verifier.

After authentication the following tasks for authorization must be performed:

4. Configuration of authorization information: the owner needs to configure the authorization information.
5. Obtaining authorization information: the authorization information needs to be made available to the entity enforcing the authorization.
6. Authorization validation: Entity attributes validated by authentication need to be mapped to the authorization information.
7. Authorization enforcement: the results of the authorization validation determine is the access to a resource granted or denied. (Gerdes et al., 2014)

7.1.2 Architecture

DCAF architecture matches the ACE Working Group architecture elements. The tasks identified in the previous chapter are divided to three levels based on the capabilities of the entities. These levels are as in ACE WG architecture: constrained-, principal- and less-constrained level.

The constrained level entities should be relieved from all complicated task if possible. These tasks are delegated to more powerful entities on the upper levels. For security reasons some tasks have to be performed in the constrained level. These tasks are: authentication of received information and enforcing authorization including validation. These tasks are the same ones described in previous chapter. The authentication tasks in question are: verifier validation and authentication (2 & 3). And authorization tasks: authorization validation and enforcement (6 & 7). (Gerdes et al., 2014)

The naming of constrained level actors is a little bit different than the naming in ACE WG documents that adopted OAuth naming conventions. Client (C) remains the same, but the Resource Server is called simply Server (S) in DCAF specification. (Gerdes, Bergmann, & Bormann, 2015c)

The principal level actor naming in DCAF is also a bit different. The client side principals is called the Client Overseeing Principal (COP) instead of

Requesting party and the server side principal is called the Resource Overseeing Principal (ROP) instead of Resource Owner. The responsibilities and specification of these actors are the same regardless of the naming. (Gerdes et al., 2015c) The principals are in charge of authorization configuration in behalf of their constrained counterparts. Namely configuration of authorization information, as defined in the previous chapter (4). (Gerdes et al., 2014)

The differences in less constrained level actor naming in DCAF specification compared to ACE WG concern both actors. Client Authorization Server is named Client Authentication Manager (CAM) and Authorization Server is named Server Authorization Manager (SAM). (Gerdes et al., 2015c)

Devices in the less constrained level are at least C2 level devices such as smartphones or laptops. They must also provide user interfaces for configuration of authorization information etc. But their main tasks and the ones concerning the authentication and authorization process are attribute binding (1) and obtaining authorization information (5). The latter means that they preprocess information from their owners, to make it usable for the constrained devices. (Gerdes et al., 2014)

DCAF makes the same assumptions on the interaction and placement of the actors as the ACE WG architecture. CAM belongs to the same security domain with the C and COP and SAM belongs to the same domain as R, S and ROP. (Gerdes et al., 2014)

7.1.3 Protocol

As mentioned the key property of DCAF is to relieve constrained nodes from resource intensive tasks such as validating certificate chains or parsing large data structures. This is achieved by offloading these tasks to more powerful nodes within the same security domain. The security objectives dictate that a secure channel between constrained and less-constrained nodes is required. DCAF does not force any specific means of communication security, but draws on the properties of DTLS since they are well understood in constrained environments. So the described approach assumes that communication is secured with DTLS when at least one of the endpoints is constrained. (Gerdes et al., 2014)

As described in the previous chapters the C and S have their less constrained level devices CAM and SAM. These entities share a symmetric key within their respective security domains and use them to establish the initial secure channel needed. DCAF only assumes that this trust relationship is established and gives no specification how the key provisioning mechanism for this relationship should work. These secure channels are used to provide dynamic authenticated authorization mechanism for the constrained devices. This means the tasks is appointed to the less-constrained devices. More specifically providing attribute binding information (1) and authorization information (5). (Gerdes et al., 2014)

Protocol flow begins when C needs to access a specific R served by S. C needs to requests an access ticket and to do this uses it's own designated CAM.

CAM relays the request to one of the Server Authorization Manager in charge of S. SAM determines the authorization level by the authorization policies configured by Resource Overseeing Principal. If C is allowed to access the resource, SAM generates a DTLS pre-shared key (PSK) for the communication between C and S and wraps it into an access ticket. The ticket may also contain detailed access permissions in a way that SAM and S can interpret them. This ticket is then relayed to C via CAM and after presenting the ticket to S, C and S can establish a secure communication channel. (Gerdes et al., 2014)

If C does not have SAM information on the S it has two options. C can send an initial unauthorized resource request to S. S denies the request and sends the address of the SAM in charge of it back to C. The other option is to look up the desired resource in a resource directory, that lists server resources. Once the address of SAM is known C can use CAM to send a request for authorization to the S in question. But CAM has to approve the request first. If it does CAM and SAM authenticate each other and their authorization to act on behalf of their constrained devices. After the authorization between the less-constrained devices SAM can do the authentication and ticketing procedure for C. An illustration of DCAF protocol flow is presented in Figure 6. (Gerdes et al., 2014)

DCAF access tokens consist of two parts: the face and verifier. The verifier holds the DTLS PSK for C and S communication, so it must be transmitted over a secure channel. The ticket face is generated for S and used as a PSK-identity of C in the ClientKeyExchange message during the DTLS handshake between C and S. The PSK-identity contains sufficient information for S to authorize C and validate that the ticket face in particular the authorization information and timestamp were generated by SAM. This eliminates the need for additional message exchange. A ticket may also contain a lifetime, so S keeps the ticket face as long it is valid. (Gerdes et al., 2014)

The mutual authentication between C and S is based on the ticket face and verifier. S authentication of C is based on C proving that it is in possession of the ticket verifier, namely the DTLS PSK generated by SAM. C can determine S's authenticity because only S is able to derive the DTLS PSK from the PSK-identity field of the ticket face. Only SAM and S can generate the same DTLS PSK based on the PSK-identity and to ensure this they use an keyed-hash message authentication code with a shared key. For resource saving reasons the hash function used is the one used in the cipher suite for their DTLS connection and for agility the function is signaled within the ticket face.(Gerdes et al., 2014)

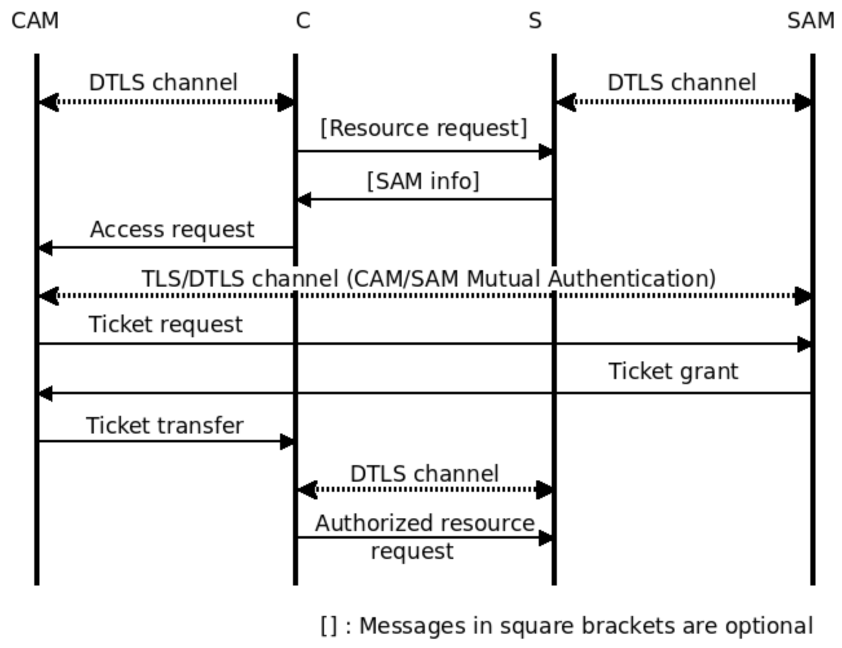


Figure 8: DCAF authentication steps (Gerdes et al., 2015c)

7.2 ABFAB

Application Bridging for Federated Access Beyond Web (ABFAB) has it's roots on a project called Moonshot that had goals for extending the benefits and developing single unifying technology for federated identities for non web based services. ABFAB is based on widely used security mechanisms and protocols such as Authentication, Authorization and Accounting (AAA) framework, Extensible Authentication Protocol (EAP), Generic Security Services Application Program Interface (GSS-API), Remote Authentication Dial-In User Service (RADIUS) and Security Assertion Markup Language (SAML). It uses EAP for end user authentication and keying material derivation like many other implementations of AAA authentication processes. For high level authentication it uses SAML since it is also a widely deployed standard for expressing end user's identity information. RADIUS is used, even though any AAA protocol could be used. Again this is because of it's wide deployment and high acceptance. (Perez-Mendez, Pereniguez-Garcia, Marin-Lopez, Lopez-Millan, & Howlett, 2014)

The benefits of federated access management are single or simplified sign-on (SSO), data minimization, less need of user participation and flexible provisioning. (Tschofenig, Hartman, Lear, Schaad, & Howlett, 2014)

SSO provides the ability for an Internet service to delegate access management responsibilities to an organization that already has a long-term relationship with the Client. This is also desirable from the Relying Party's point of view, because in many cases it does not want these responsibilities. The benefits for the Client is that it requires fewer credentials. (Tschofenig et al., 2014)

Data minimization is due to the fact that the often the Relying Party does not need to know the identity of a Client to reach an access management decision. In most cases it is adequate to only gain knowledge of specific attributes about the Client, such as that the Client is affiliated with an organization or has a certain role or entitlement. In some cases it might be adequate if the Relying Party only knows a pseudonym of the Client. (Tschofenig et al., 2014)

The need for Client interaction is reduced by the fact just mentioned that the identification is based on the Client attributes held by the Identity Provider. Prior to releasing the attributes to the RP, the IdP checks configurations and policies to determine if the attributes should be released. In this decision point there is no need for direct Client participation. (Tschofenig et al., 2014)

Provisioning of the attributes is flexible and in the cases the Relying Party needs to know more about a Client than an affiliation or pseudonym, some federated access management Identity Providers have the ability to provide the information on request or unsolicited. (Tschofenig et al., 2014)

7.2.1 ABFAB Objectives

The main focus of ABFAB is a functionality on non-web-based environment and protocols where HTTP is not used. Despite the trend to layer protocols on top of HTTP, there are still numerous protocols that do not use HTTP -base transport. Many of such protocol lack a native authentication and authorization framework. (Tschofenig et al., 2014)

Other ABFAB design objectives are:

- All parties taking part to a transaction are authenticated, but not necessarily identified and the Client is authorized for resource specific access.
- Authentication is independent from the application protocol used. This allows for multiple authentication methods with minimal changes to the application.
- The architecture does not imply sharing of long term private keys between Client and Relying Party.
- System is scalable to large number of identity providers, relying parties and clients.
- System is based on existing standards and components and operational practices. (Tschofenig et al., 2014)

7.2.2 Architecture

The ABFAB architecture consist of three main entities: Client, Relying Party (RP) and Identity Provider (IdP). Client represents an entity that needs to access a service hosted by RP. RP acts as a service provider in control for the service the client wishes to access. IdP is the entity in charge of verifying the clients credentials and distribution of authorization information to the RP. (Perez-Mendez et al., 2014)

In addition of elements the architecture describes how these elements are interconnected and the protocols used in these connections. An application protocol provides the service for the eventual communication between the Client and RP, but ABFAB architecture does not assume any particular protocol. As a part of the access control procedure a GSS-API security context must be established between the Client and RP, in which the Client acts as GSS Initiator and the RP as GSS Acceptor. The GSS context authenticates the Client and provides the Client identity information used for authorization purposes to the RP. (Perez-Mendez et al., 2014)

The GSS-API handles the high level abstraction of the access control process, but the actual authentication process is carried out by executing EAP. The EAP authentication process is a part of the GSS context establishment and it enables a federated operation as it involves IdP in the process. The client acts as EAP peer, the IdP as an EAP server and RP as EAP authenticator. EAP packets are transported between the Client and RP GSS-API Mechanism for the Extensible Authentication Protocol (GSSEAP) defined by ABFAB WG specially for this purpose. GSSEAP specifies the means for transporting EAP transport on GSS tokens and using key material exported by the EAP method in GSS security services. (Perez-Mendez et al., 2014)

Finally ABFAB specifies that RADIUS is used as an AAA protocol between RP and IdP. This provides the federation substrate, in other words implementing trust relationships to form a federation. The AAA protocol has two functions: it conveys EAP packets between RP and IdP and transports Client's identity attributes from IdP to RP. This information is represented by SAML statements. (Perez-Mendez et al., 2014)

7.2.3 Protocol

An ABFAB workflow example of a client attempting to connect to a server to access data or perform a some type of transaction has the following prerequisites and steps. An illustration of ABFAB protocol flow is illustrated in Figure 7. (Tschofenig et al., 2014)

1. Client Configuration: The IdP has pre-configured the client with a Network Access Identifier (NAI). The client is also configured with the necessary keys, certificates, passwords or other information needed to run the EAP protocols between it and IdP.

2. Authentication mechanism selection: The Client is configured to use GSS-EAP as a GSS-API mechanism for authentication and authorization.

3. Client provides an NAI to RP: The Client starts an application protocol for transport to the RP and begins the GSS-EAP authentication. RP sends an EAP request message in response nested in the GSS-EAP protocol asking for the Client's name. The Client sends an EAP response with an NAI name form containing at the minimum the realm portion of it's full NAI.

4. Discovery of federated IdP: The RP determines which IdP to contact based on policy and the realm portion of the Client NAI. For this it uses pre-configured information or a federation proxy.

5. Request from Relying Party to IdP: When the RP finds out the IdP or and agent of the IdP in question, it sends a RADIUS access request to the IdP, including an encapsulated EAP response from the Client. In this point the RP probably has no idea of the clients identity. RP sends it's identity to the IdP in AAA attributes and it may include a SAML Attribute Request in a AAA attribute. The AAA network checks that the identity RP claims is valid.

6. IdP begins EAP with the client: IdP sends an EAP message to the Client with an EAP method to be used. In this stage IdP should accept a realm only request to protect the client's name.

7. The EAP protocol is run: Several EAP messages are passed between the client and the IdP. The content and number of the messages depend on the selected EAP method. If the IdP is not able to authenticate the client, it sends an EAP failure message to the RP. A part of the EAP protocol is that the Client sends a channel bindings message to the IdP. In this message the Client identifies the RP it is attempting to authenticate. The IdP checks between the channel binding data received from the Client and RP using the AAA protocol. If the bindings do not match this also constitutes a failure and IdP sends an EAP failure message to the RP.

8. Successful EAP Authentications: The IdP and Client have mutually authenticated each other and hold two cryptographic keys. A Master Session Key (MSK) and an Extended MSK (EMSK). The Client has also a level of assurance about the RP's identity. Based on the channel binding data and the naming information check from the AAA framework conducted by IdP .

9. Local IdP Policy Check: The IdP checks the local authorization policy to determine if the given transaction or service is permitted between the Client and RP. If the action is permitted IdP releases the attributes to RP, if not it sends an EAP failure message to the RP and Client.

10. IdP provides the RP with the MSK: IdP sends a positive result EAP message and EAP MSK to the RP. Optionally it sends a set of Client AAA attributes as one or more SAML assertions.

11. RP Processes Results: After receiving the result from the IdP, RP should have enough information to either grant or refuse the resource access request. Depending on the authorization level it may have specific authorization identity information concerning the Client. If the RP needs additional attributes from the IdP, it can make a new SAML request to the IdP. The RP will apply the results in an application specific way.

12. RP returns results to the client: When the RP has received a response it informs the Client of the result. If all the checks has gone well all parties are authenticated. The Client can complete the authentication of the RP by using the EAP MSK and proceed with appropriate authorization level.

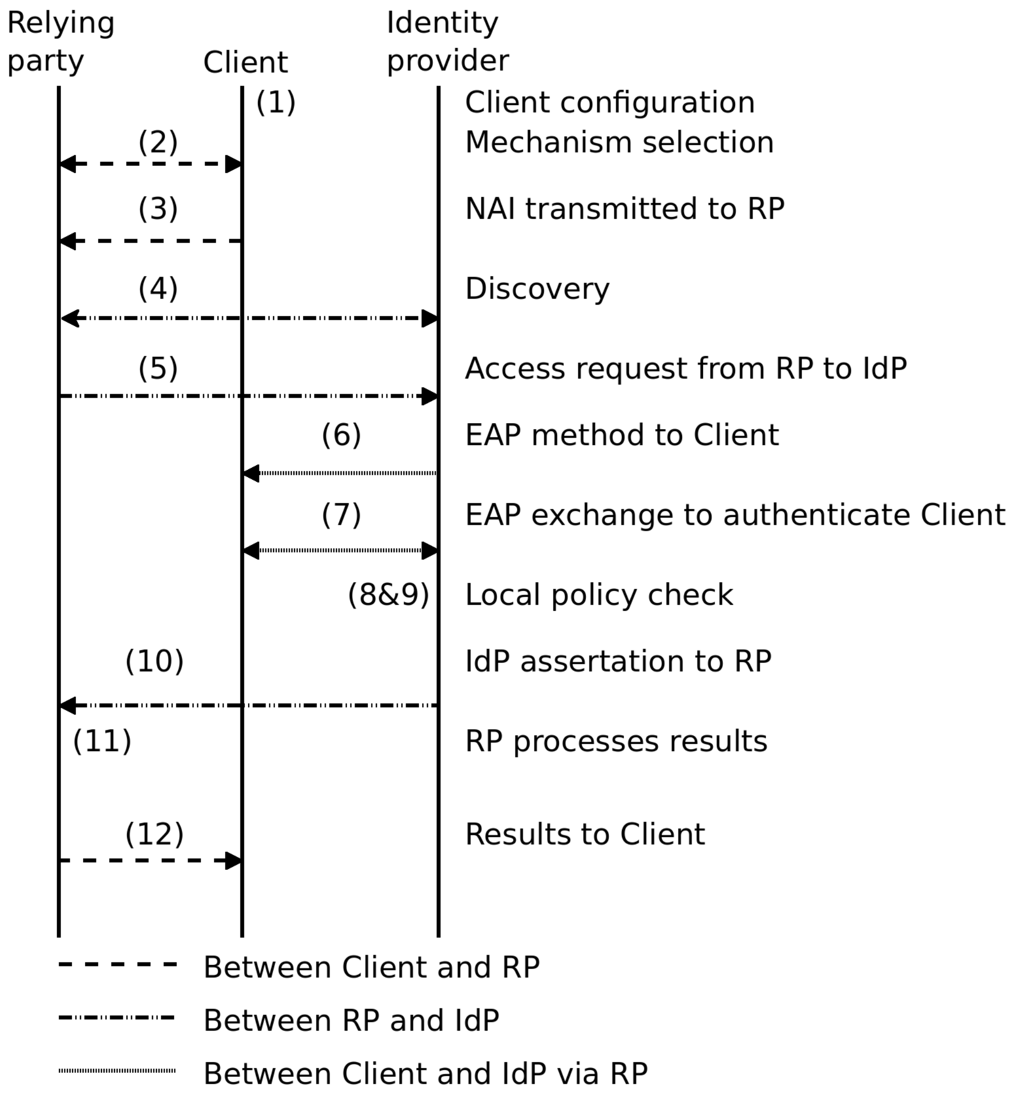


Figure 9: ABFAB authentication steps (Tschofenig et al., 2014)

8 Use case

IETF has specified several use cases to identify and address problems on authentication and authorization in constrained environments. The application areas vary from logistics and smart metering to health monitoring and entertainment. These use cases assume that communication architecture between the devices is Representational State Transfer (REST) based and although most conclusions are generic it is assumed that Constrained Application Protocol (CoAP) is used. REST architecture means that a device acts as a server, offering resources and these resources can be accessed by clients. Resources accessed can be sensor data or control of actuators. In some cases the communication happens through intermediaries such as gateways or proxies and the processes also can happen without human intervention (M2M communication). (Selander et al., 2015)

Each use case presents a general description of an application environment and one or more specific use cases, as well as a list of relevant authorization problems to the application area. Definitions are needed for many communication issues such as how do the devices find each other or which device initiates the conversation. These requirements vary between use cases and there can be a need for multiple communication schemes. (Selander et al., 2015)

For this work a logistics use case was selected, dealing with container monitoring. The reason for selecting this use case over others is that it brings many different authorization scenarios, since it deals with a mobile system in various transport modes.

The motivation for this use case arises from the need of storing perishable goods in a constant temperature and adequate ventilation. There are also multiple stakeholders interested in the real-time information on the state of the goods. For example a transporter needs to prioritize the handling goods by their expiration date and vendors need to fulfill their delivery obligations. Wireless sensor systems make this kind of continuous tracking of environmental variables possible. (Selander et al., 2015)

This use case describes a logistics chain for bananas grown in south America that are transported for the German market. Bananas are shipped to

Germany and picked up by trucks that deliver them to the fruit vendors ripening facility. After ripening a supermarket chain buys the bananas and transports them to multiple stores by trucks. The fruit vendor equips the banana boxes with sensors as a quality management measure. The bananas are monitored consistently through the whole chain and system records abnormal fluctuations in sensor values. The temperature and ventilation information sensors are also used to control the climate control system of the transport container and ripening facility. In the ripening facility the sensors monitor the ripeness of the bananas, so that ripe bananas can be identified and sold before they spoil. (Selander et al., 2015)

The second functionality in the banana box sensors is identifying and locating the goods meant for a specific customer during loading and unloading. The use case also gives additional parameters on logistics data confidentiality. It defines that personnel is only allowed to see data for the time of loading or unloading and information about the state of the goods needs to be confidential in these situations. Another issue concerning data confidentiality and ownership is that after the ripe bananas are sold to the supermarket chain, the ownership of the sensors is transferred to the buyer. (Selander et al., 2015)

Networking problems in this use case include environmental problem concerning the radio connection between the nodes and also a Internet connectivity problem. The radio connection problem is due to high water content of the bananas, that can cause problems on direct communication between nodes and messages need to be forwarder over multiple hops. The Internet connectivity problem during the journey may be solved using relay stations owned by the transport company. (Selander et al., 2015)

From previous user story eleven authorization problems summarized:

1. Different authorizations need to be granted to the resources of the banana box sensors owned by the fruit vendor and container systems owned by the container owner.
2. Fruit vendor requires the integrity and authenticity for the sensor data to ensure the quality of the product and climate control functionality.
3. The container owner requirement for sensor data integrity and authenticity for climate control input.
4. Fruit vendor requirement for sensor data confidentiality for state of the goods and location to protect against attacks to the data by a competitors.
5. Different protection needed by the fruit vendor to protect sensor data and logistics data residing in the same endpoint.
6. Data integrity and authenticity requirement of the fruit vendor and logistics personnel for locating the goods and ensuring that the goods are treated and delivered correctly.
7. Authorization process has to work even if the fruit vendor and logistics personnel are at the time of access unable to take part on the process.

8. Temporary access grants need to be issued in order to avoid giving out permanent access to parties who may no longer be involved with the process. This concerns all actors.
9. Security objectives need to be intact even if messages are forwarded over multiple hops. This concerns all actors.
10. Authorization policies have to be enforced even if the devices do not have access to the Internet at the time.
11. Need to revoke authorization on a malfunctioning sensor by fruit vendors and container owners.

These problems can be derived to security requirements for an authorized authentication mechanism. The requirements are presented in Table 8. Each requirement is based on one or more problems. The corresponding problems for every requirement are shown in brackets after the requirement name.

TABLE 8 Use case security requirements

Requirement	**Description**
Integrity & authenticity of sensor data (2,3,6)	Quality of monitored recordings and identification data need to be attained, in order to serve climate control and communication to logistics applications.
Confidentiality of sensor data (4)	Transmitting data on product quality and location needs to be secured against third parties, that might benefit from this information.
Authorization by resource and requesting party basis (1,5)	Several different resources can reside in a single endpoint and different authorizations need to be granted for different parties. For this reason authorization needs to be configured on resource and requesting party basis.
Autonomous authorization (7,10)	Resource server has to be able to enforce authorization even when resource owners are not able to intervene or there is no Internet connectivity at the time of authorization. In addition the connectivity problems caused by high water content of the cargo may call for autonomous capabilities from individual devices.
Temporary access permissions (8,11)	Mechanism for temporary permissions and/or revoking permissions is needed. In order to avoid situations where personnel no longer involved with the process have access to the system or when resource servers having malfunctioning sensors need their access to be revoked.
End-to-end security (9)	Security objectives need to be achieved, even if messages between endpoints are forwarded over multiple hops.

The empirical experiment is an implementation of this use case. It portrays a scenario where a Client from another security domain such as logistics company wants to gain access to temperature data from the banana boxes owned by the fruit vendor.

9 Experiment design

The empirical part of this study is and experiment conducted using DCAF protocol implementation. The objective of this experiment is to gain further knowledge on how well DCAF protocol functions in a constrained environment and the effect it has on the power consumption of the devices. The results of the experiment are used when the framework objectives are evaluated in chapter 11.1. Since these protocols are in the development phase this was the only implementation of such a protocol implementation to be found at the time of conducting the study. The experiment is setup so, that the constrained actors are simulated and less constrained actors are running as separate processes in the host machine. The actors communicate using IPv6 networking through a border router connecting the simulated actors and the actors running on the host machine. Next the different parts of the experiment are described in greater detail.

9.1 Contiki

The simulation environment used in this study is Cooja, which simulates devices running Contiki operating system.

Contiki is an operating system developed for constrained environments. It has been ported to a number of micro-controller architectures including Texas Instruments MSP430 and the Atmel AVR, which both are very popular and inexpensive (Dunkels, Grönvall, & Voigt, 2004).

Contiki is based on an event driven kernel, to reduce the size of the system. On top of the event driven kernel Contiki uses an application library that provides preemptive multi-threading. The library is optionally linked with applications requiring multi-threading features. This design was adopted to avoid the usage difficulties and problems of handling long running computations with state-machine driven programming of an event-driven system. (Dunkels et al., 2004)

A Contiki system consist of two parts: a core and loaded programs. The core contains the kernel, base services and some parts of the language run-time and support libraries. Shared functionalities are implemented as shared libraries and run as services. The programs can be loading and unloading individually at runtime and services can be updated or replaced individually. This gives a Contiki system a flexible structure. (Dunkels et al., 2004)

9.2 Cooja

Due to the distributed nature of sensor networks, developing applications for them is often times a difficult and time consuming task. This is caused by a longer compile-run-debug cycle, since compiled programs need to be transferred onto a set of sensor nodes for debugging and testing. This process can be simplified by using a simulator which enables the development of algorithms, system behavior study and interaction observations in a controlled environment. (Osterlind, Dunkels, Eriksson, Finne, & Voigt, 2006)

Cooja is a Java-based simulator for simulating networks of devices running the Contiki operating system. In a network simulation each node can be of different type, not only by running different software, but also by simulated hardware. Cooja is flexible as far as any many parts of the simulator can be easily replaced or extended with additional functionality. For example simulated radio medium, node hardware or plug-ins for input/output can be extended. (Osterlind et al., 2006)

Contiki programs can be executed in two different ways in Cooja: native code running directly on the host CPU or running in an instruction level emulator (Osterlind et al., 2006). Examples of these emulators are MSPSim for TI MSP430 (Eriksson et al., 2009) and Avrora for the Atmel AVR platforms (Titzer, Lee, & Palsberg, 2005). Non Contiki nodes can also be simulated. These include nodes implemented in Java and nodes running another operating system. All these approaches have their own advantages and disadvantages. Java-nodes do not run deployable code, but they enable faster simulations and are useful for tasks such as distributed algorithm development. Using emulated nodes a set of more fine-grained execution details can be captured, compared to Java or native code based nodes. Native code simulations run deployable code, but are more efficient than emulated node simulations. Using deployable code makes it possible to simulate and deploy the same code without any modifications. This minimizes the time needed to move from simulations to hardware deployments. (Osterlind et al., 2006)

There are also advantages in combining different abstraction levels in one simulation. Good example of this is a large simulated networks where only few nodes need to be simulated at the hardware level and others can be implemented at Java level. The advantage of this approach is the faster simulation comparing to using all emulated nodes, but still receiving the fine-grained execution details from the emulated nodes. (Osterlind et al., 2006)

Hardware peripherals of the simulated nodes are called interfaces. Interfaces enable the simulator to detect and trigger events such as incoming radio traffic. They also represent node properties such as positions in the environment that the actual node is not aware of. Interactions between the simulation and simulated nodes are done through plugins, such as simulation control that enables a user to start or pause the simulation. Both interfaces and plugins can be used to add custom functionality to simulations. (Osterlind et al., 2006)

For user interaction Cooja has a Graphical User Interface (GUI), which allows the placement of sensor nodes, modifying sensor inputs and disturbing network communication by increasing the noise level among other things (Kugler, Nordhus, & Eskofier, 2013). The GUI also presents information on different aspects of the simulation such as radio traffic, reception areas and mote logs. Cooja can also provide estimates on mote energy consumption (Kugler et al., 2013). In this study the power consumption estimates are gathered by Powertrace.

9.3 Powertrace

Powertrace is a run-time power profiling mechanism based on power state tracking to estimate power consumption of nodes running Contiki. It breaks down the power consumption to energy capsules and aggregates them to higher-level activities. It allows inspecting both energy behavior in node-level and network-level. (Dunkels, Eriksson, Finne, & Tsiftes, 2011)

Powertrace has a linear power model, which means that instantaneous power is estimated as the sum of all active power states. Component energy is derived from the time that the system spends in each power state. Components in this case are such as the CPU, radio transceiver, on board Flash and sensors. Examples of the power states are CPU in active mode or sleep mode and radio in listening mode or transmission mode. Device drivers are instrumented to record a time stamp when a component enters a new state and when it leaves the state, time difference is calculated and added to the corresponding capsule. (Dunkels et al., 2011)

The power state tracking is done entirely in software, so no additional hardware is needed. The advantages of a software-based approach are that it is not affected by environmental factors, such as temperature and humidity that affect the energy consumption of a physical system. It also yields the same result on different batches of the same hardware, which is not always true when hardware-based method is used. (Dunkels et al., 2011)

9.4 DCAF setup

As mentioned the DCAF experiment consists of constrained devices running Contiki that are simulated in Cooja environment and two processes running on the host machine. The processes running on the host machine represent the less constrained actors Server Authorization Manager (SAM) and Client Authorization Manager (CAM). The constrained nodes are Client (C) and Resource Server (RS).

9.4.1 Constrained actors

Client and Server are compiled as emulated Wismote-devices and added to the simulation. The RPL Border Router is also compiled to the same platform. The actors are positioned in the simulation so, that they do not have a network path between them, because they are out of each others radio range. So all the communication between the Client and Server is relayed by the RPL Border Router. This setup was chosen so that the energy consumption of the actors can be compared more accurately. The constrained network is displayed in Figure 10. Cooja network view does not have an option to show a name of the node and node type can only be displayed by color. But in this picture node number 1 is the RPL Border Router, node number 2 is the Resource Server and node number 3 is the Client. The green area around the Server node represents its radio range and shows that only the Border Router is within the Servers radio range.

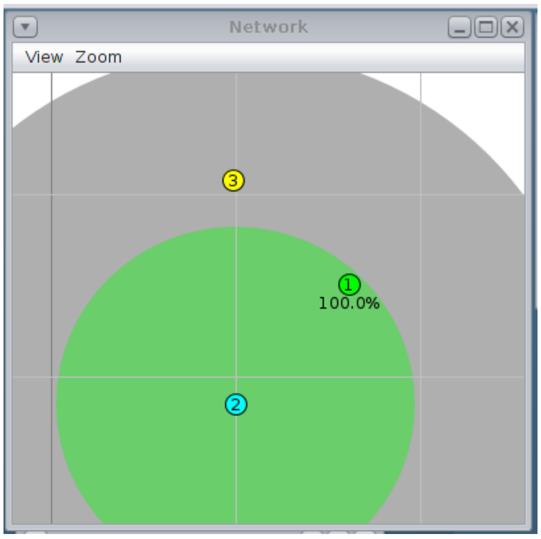


Figure 10: Client, Server and the Border router shown in Cooja Network

When the simulation is started the RPL Border Router starts to form the network. The IPv6 addresses the nodes get is consistent with the node numbers. Node number 1 address is aaaa::200:0:0:1, node number 2 address is aaaa::200:0:0:2 and so on. To enable communication to an external network first the Border Router needs to have an open server serial socket configured. When the simulation is running a bridge between the RPL network and the external network can be created with the Tunslip utility provided in Contiki.

9.4.2 Less constrained actors

The less constrained actors are run from the command line of the host machine. CAM has nothing to configure. SAM can be configured with JSON files in the program folder or through a web-interface. To make a working DCAF authorization the less constrained actors need to have a set of CA-Certificates. When these Certificates are created CAM can be added to the SAM list of authorized CAM's. Individual CAM's are identified by SHA1-fingerprints provided by their certificates.

New servers can be added from the SAM web-interface, by giving a host name or IP address, secret and one or more resources and methods they support (GET, POST, PUT, DELETE). To establish a connection between the SAM and a RS, the RS has to be commissioned by SAM. This can be done from the web-interface also. In order to establish this connection there needs to be a mutual secret configured in the adding phase and the same secret also needs to be configured for RS. After commissioning SAM can be used to create access rules for the RS.

A rule is created by selecting a CAM that the rule is given, expiration date and one or more resources with their allowed methods. In addition special conditions for each rule can be given. When a rule is in place CAM can negotiate an access ticket from SAM and relay it to one of it's Clients. When a ticket is granted it can be viewed from a list in the SAM web-interface where a list of granted and revoked tickets can be found. A granted ticket can be revoked from the list view of granted tickets.

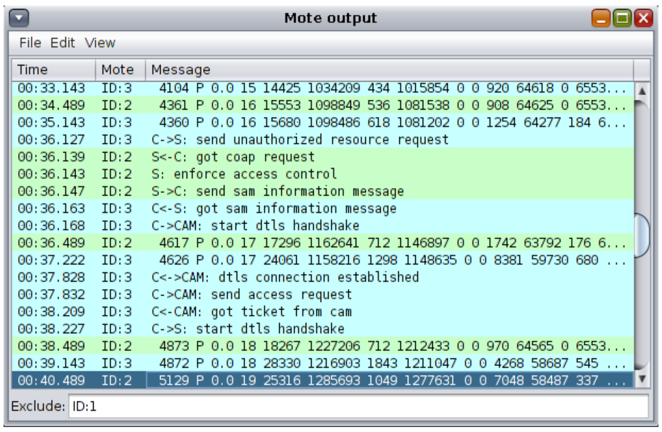
The less constrained actors communicate with their constrained counterparts through the RPL Border Router and Tunslip. Both are configured to aaaa::1 IPv6 address. CAM listens to port 5684 and SAM listens to port 8080.

9.4.3 Running the experiment

Running the experiment begins with starting the Cooja simulation. When the simulation is started the RPL Border Router starts to form the network inside the simulation. Next Tunslip is started to form a bridge to the external network, namely to the host machine. When Tunslip output shows that it has formed the bridge to the simulation for the less constrained actors, SAM and CAM can be started in separate terminal windows.

The DCAF procedure is initiated by the Client by a 30 second timer. This timer is in place so that the RPL Border Router has enough time to form the network and there is also enough time to start Tunslip, SAM and CAM processes. The first message sent by Client is an unauthorized resource request to the Resource Server. The Resource Server enforces access control and replies to this message with an unauthorized resource request code and includes SAM information with the message. After the Client receives the SAM information it starts DTLS handshake with CAM to establish a DTLS connection. When the secure connection is established, the Client can send an access ticket request to CAM requesting access to the RS. CAM receives the message and forms a DTLS connection with SAM and forwards the ticket request. SAM then checks it's settings if a rule for the requested resource is set for the CAM requesting it. If a rule is set SAM sends an access ticket to CAM which relays it to the Client. Now the Client has an access ticket to the resource it wants to access in the RS. The Client starts a DTLS handshake with the Resource Server to form a secure connection between the two. After the handshake is done the Client can send authorized resource requests for the resource as long as the access ticket is valid.

Cooja provides lots of information on the running simulation in for of logs and time lines. In this experiment the most important log is the Mote output log shown in Figure 11. This log is important because it does not only show the debug messages from the motes and network log generated by the RPL Border Router, but it also logs the Powertrace messages used for power estimation. This log view has an exclude filter which helps to filter out messages to help analyzing the messages.



```
Time       Mote  Message
00:33.143  ID:3   4104 P 0.0 15 14425 1034209 434 1015854 0 0 920 64618 0 6553...
00:34.489  ID:2   4361 P 0.0 16 15553 1098849 536 1081538 0 0 908 64625 0 6553...
00:35.143  ID:3   4360 P 0.0 16 15680 1098486 618 1081202 0 0 1254 64277 184 6...
00:36.127  ID:3  C->S: send unauthorized resource request
00:36.139  ID:2  S<-C: got coap request
00:36.143  ID:2  S: enforce access control
00:36.147  ID:2  S->C: send sam information message
00:36.163  ID:3  C<-S: got sam information message
00:36.168  ID:3  C->CAM: start dtls handshake
00:36.489  ID:2   4617 P 0.0 17 17296 1162641 712 1146897 0 0 1742 63792 176 6...
00:37.222  ID:3   4626 P 0.0 17 24061 1158216 1298 1148635 0 0 8381 59730 680 ...
00:37.828  ID:3  C<->CAM: dtls connection established
00:37.832  ID:3  C->CAM: send access request
00:38.209  ID:3  C<-CAM: got ticket from cam
00:38.227  ID:3  C->S: start dtls handshake
00:38.489  ID:2   4873 P 0.0 18 18267 1227206 712 1212433 0 0 970 64565 0 6553...
00:39.143  ID:3   4872 P 0.0 18 28330 1216903 1843 1211047 0 0 4268 58687 545 ...
00:40.489  ID:2   5129 P 0.0 19 25316 1285693 1049 1277631 0 0 7048 58487 337 ...
Exclude: ID:1
```

Figure 11: Cooja mote output log

The Powertrace messages can be gathered form the log and processed in another application. The message contain values for power consumption both aggregated during the whole simulation and per cycle. Table 9 explains the structure of this example message:

4626 P 0.0 17 24061 1158216 1298 1148635 0 0 8381 59730 680 67432 0 0 (radio 97.-7240% / 100.00% tx 0.10% / 0.99% listen 97.-7251% / 99.00%)

TABLE 9 Powertrace parameters

Value	Parameter	Description
4626	clock_time	Clock time
0.0	rimeaddr	Rime address
17	seqno	Sequence number
24061	all_cpu	Accumulated CPU energy consumption
1158216	all_lpm	Accumulated Low Power Mode energy consumption
1298	all_transmit	Accumulated transmission energy consumption
1148635	all_listen	Accumulated listen energy consumption
0	all_idle_transmit	Accumulated idle transmission energy consumption
0	all_idle_listen	Accumulated idle listen energy consumption
8381	cpu	Latest cycle CPU energy consumption.
59730	lpm	Latest cycle Low Power Mode energy consumption
680	transmit	Latest cycle transmission energy consumption
67432	listen	Latest cycle listen energy consumption
0	idle_transmit	Latest cycle idle transmission energy consumption
0	idle_listen	Latest cycle listen energy consumption

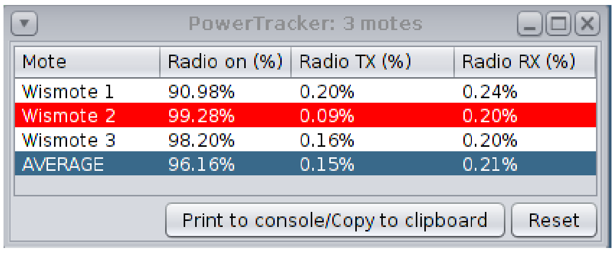
The values the Powertrace prints out are not actual power usage values, but a sum of all power states as explained in the chapter about Powertrace. The actual power consumption needs to be calculated from these values, because it is dependent on the hardware of the device. The power consumption in milliwatts can be calculated with the following formula (Han, Cao, Alinia, & Crespi, 2015):

$$Power\,consumption = \frac{Powertrace\,value * Current * Voltage}{RTIMERSECOND * Runtime}$$

Where power trace value is the value from the Powertrace log. Current is the current consumption of the component, CPU, CPU in Low Power Mode, Transmit (TX) or Receive (RX). These values can be obtained from the device datasheets, which in the case of Wismote are MSP430 (Texas Instruments, 2014) and CC2520 (Texas Instruments, 2007). The values of Currents are 2,2 mA and 0.00169 mA, 33.6 mA and 18.5 mA for CPU, LPM, TX and RS respectively.

Voltage value in the formula is the supply voltage of the system. RTIMERSECOND value is the number of ticks per second in the system, which for Wismote is 32768 and Runtime is the seconds between log events which in this experiment is 2 seconds.

Total radio usage can also followed with the PowerTracker tool provided by Cooja shown in Figure 12.



Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Wismote 1	90.98%	0.20%	0.24%
Wismote 2	99.28%	0.09%	0.20%
Wismote 3	98.20%	0.16%	0.20%
AVERAGE	96.16%	0.15%	0.21%

Figure 12: Cooja PowerTracker tool

10 Results

This chapter presents the results extracted from the experiment. The first chapter presents the time consumption of different parts of DCAF authorization sequence. The data was gathered from mote outputs after each simulation was run. The second chapter deals with power consumption data gathered from Powertrace outputs, which was then calculated to actual power consumption readings in mW. The results are evaluated more throughly and their positioning in the evaluation framework is discussed in chapter 11.

10.1 Time consumption data

Time data was gathered from the logs that are written as the Client and Server initiate different parts of the DCAF authorization sequence. To analyze how much time each part consumes, the total time of the authorization sequence was divided in smaller parts and average times for these parts were calculated. The parts include: unauthorized request flight time, Client-CAM DTLS handshake, ticket request flight time, Client-Server DTLS handshake, authorized request flight time and the total time used in DCAF authorization sequence. Average times for different parts and also minimum and maximum times are presented in Table 10.

The table shows that the most time consuming part of the authorization sequence is the DTLS handshake between the Client and Server. Time spent on this sequence also had the biggest variance. The time taken on the DTLS handshake is caused by the hash function the Server needs to run to validate the access ticket provided by the Client. But although this part of the mechanism takes time it saves energy, because no further transmissions is needed to establish the DTLS connection.

Other parts the variance was smaller, including the DTLS handshake between Client and CAM. Request response sequences between the constrained actors gave totally congruent values.

TABLE 10 Time elapsed in key functions

Sequence	**Average time (sec)**	**Min (sec)**	**Max (sec)**
Unauthorized request flight time	0,036	0,036	0,036
Client-CAM DTLS handshake	1,682	1,612	1,726
Ticket request flight time	0,336	0,256	0,389
Client-Server DTLS handshake	5,362	4,558	8,572
Authorized request flight time	0,051	0,051	0,051
DCAF authorization sequence total	7,502	6,623	10,576

10.2 Power consumption data

The power consumption of the constrained nodes was logged as described in the previous chapter. The readings were then converted to milliwatts to analyze power consumption on different stages of the experiment. Client and Server were analyzed separately. A graphs of power consumption of the Client and Server are visualized in Figures 13 and 14 respectively.

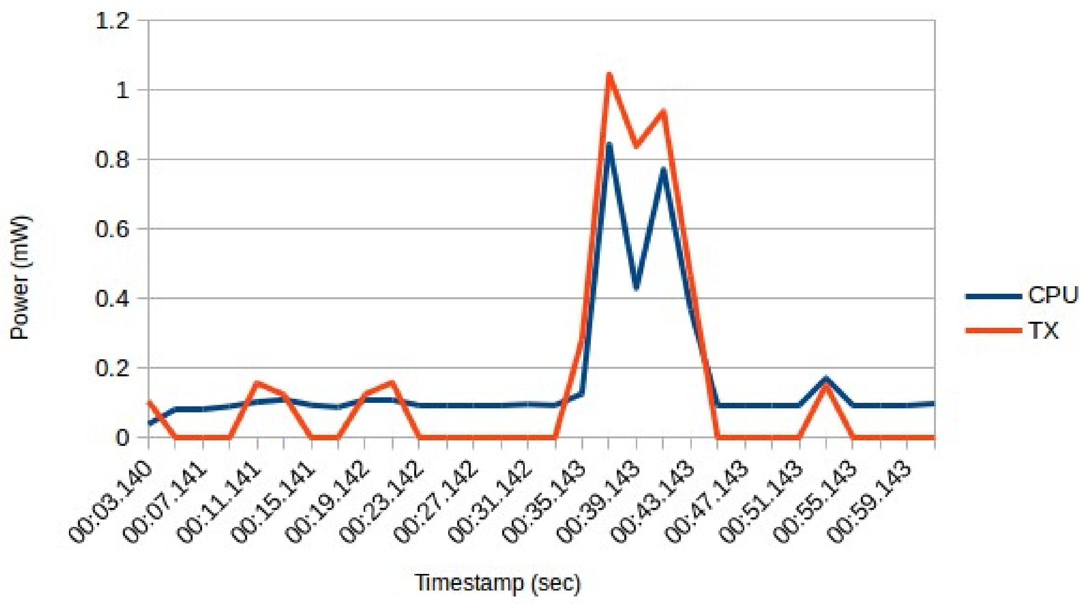


Figure 13: Client power consumption

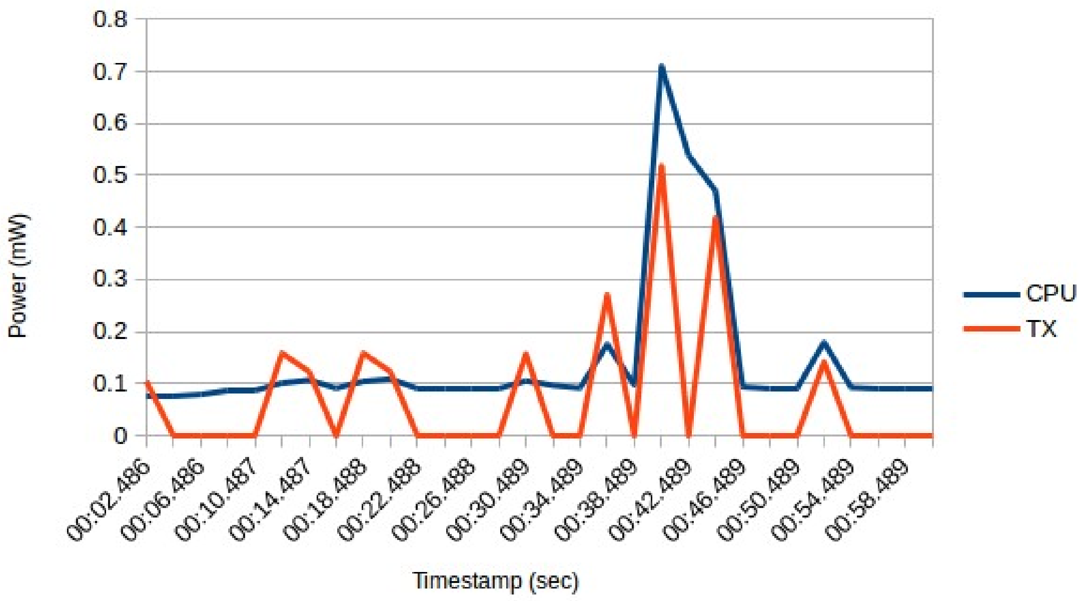


Figure 14: Server power consumption

Since the nodes in the experimental setup do not use duty cycling, in other words the radio is always on, the listening of radio transmissions (RX) values remained constant. Since RX values would not bring any additional information so the values were ignored in the analysis. The graphs and tables display CPU and Transmission (TX) values, where the CPU value consists of both active and Low Power Mode CPU values.

Both graphs clearly show the power consumption peaks starting at approximately 30 second mark. This peak is the authorization sequence initiated by the Client. Transmissions seen before the sequence are ICMP traffic when the RPL Border Router forms the network and the nodes negotiate paths within the network. The smaller peak after the authorization sequence is a authorized resource request from the Client, that the Server replies with an appropriate resource response.

The graphs clearly show that the Client peak power usage on transmissions is substantially higher compared to the Server. This is due to bigger number of transmissions the Client has to do during the authorization sequence. The authorization sequence from the Servers point of view consists of two request-response sequences and a DTLS handshake. The first request is unauthorized and so denied and SAM information is sent to the Client. During the second resource-response sequence the Client has an access ticket. So it can initiate DTLS handshake with the Server and after the secure connection is established it can send an authorized resource request. The Client on the other hand consumes much more energy on transmissions during the authorization sequence, because it also needs to negotiate a secure connection with CAM and send an access ticket request, in addition to communication with the Server. This amounts to two times higher Client peak power usage on transmissions compared to the Server.

The CPU power usage peak values between the Client and the Server do not differ as much as the transmission values. But when the graphs are compared, they show that the Server has a higher peak on CPU usage than in transmissions. During the DCAF authorization sequence the Client negotiates two secure DTLS connections when the Server only needs to negotiate one. This suggests that the DTLS handshake it self does not amount to very large CPU power usage when the keys are provided for the constrained actor.

The power consumption was further analyzed by separating different events and calculating how much energy was consumed during these events. The events taken into further analysis were network negotiation sequence, authorization sequence and the authorized request-response sequence. These values were then compared to the power consumption when the system was in idle state and the total power consumption of the experiment.

The experiment was conducted five times and averages derived from the values gathered were used as approximations of power usage in each sequence. The power consumption values for each sequence were gathered from the aggregated power consumption values for each component separately. The start and end of each state was determined form the node log events and the closest aggregate value from the end of the sequence was subtracted from the closest aggregate value from the start of the sequence. After gathering the data on all experiments, the average values were collected together. This information is provided in Table 11 for the Client and Table 12 for the server.

TABLE 11 Client power consumption in different sequences

Sequence	**CPU (mW)**	**TX (mW)**	**CPU+TX (mW)**
Authorization	2,4831	3,3546	5,8377
Network negotiation	0,1453	0,1046	0,2499
Authorized request response	0,1757	0,1507	0,3265
Idle (2 seconds)	0,0980	0	0,0980
Total	4,9714	4,4565	9,4279

TABLE 12 Server power consumption in different sequences

Sequence	**CPU (mW)**	**TX (mW)**	**CPU+TX (mW)**
Authorization	2,0521	1,1819	3,2340
Network negotiation	0,2495	0,1046	0,3541
Authorized request response	0,1845	0,1418	0,3263
Idle (2 seconds)	0,0974	0	0,0974
Total	4,4590	2,1718	6,6307

This data supports the findings made from the graph data. The Client uses more than two times more power to transmissions during the DCAF authorization cycle compared to the Server. The CPU values on the other hand do not differ so much between the two based on this data as the graph also suggested. On the idle and authorized request-response sequences the total CPU+TX power usage between the Client and the Server hardly differs at all.

From this data an estimate can be calculated for how many times this sequence could be conducted if the nodes would be powered by batteries. The battery life can be calculated with a formula (Han et al., 2015):

$$Battery\ life = \frac{Battery\ capacity\ x\ Current}{Power\ usage}$$

Where battery capacity is the mAh of the battery used, for example a typical alkaline AA battery this value is 2500 mAh. Current is the current drawn from the battery and power usage is the power usage of an event in mW. If two AA batteries are used which both have an output of 1,5 Volts they conveniently supply the current needed. If this formula is applied to the power usage of the CPU and TX during a Clients DCAF authorization sequence, which is 5,84 mW this formula would provide the number of authorization sequences possible:

$$\frac{2500\,mAh\ x\ 1{,}5V * 2}{5{,}84} = 1285$$

For the Server this formula produces the value 2319 sequences.

11 Evaluation

In this chapter the framework objectives constructed in chapters 4, 5, & 6 are applied to the results of the empirical part and protocol specifications in chapter 7. This will identify protocol features that can provide solutions to the constraints and security objectives identified by the framework. To make the following the different parts of this chapter easier, an overview of all framework components is displayed in Table 13.

TABLE 13 Framework overview

Constraints	Maximum code complexity
	Size of state and buffers
	Processing power
	Available power and energy
	Network constraints
	Lack of user interfaces
	Physical constraints
	Cost
Architecture based security objectives	Delegation of demanding tasks
	Validation of actors
	Autonomous functionality
	End-to-end security
Security objectives	Resource confidentiality
	Resource integrity
	Resource availability
	Message confidentiality
	Message integrity
	Message authentication
	Message freshness and authorization
	Access control architecture
	Message sequence for three party authentication

The results of the experiment described in the previous chapter are mapped into the framework in the next chapter 11.1. The following chapters 11.2 discuss how the architecture based objectives are met. The third chapter 11.3 assesses the security properties of the protocols by the framework security objectives. The fourth chapter 11.4 identifies the dependencies between use case requirements framework objectives. And finally chapter 11.5 evaluates the framework it self.

11.1 Constraints

This chapter discusses the constraints identified by the framework and how well they match the implications of constraints observed during the experiment.

11.1.1 Memory constraints

Based on the experiment and documentation DCAF it can handle the the memory constraints of a Class 1 device in both ROM and RAM. In practice this means that the Contiki operating system and the needed libraries needed for CoAP and securing the transmissions with DTLS fit in ~100kB of ROM available. The available RAM also seems to be adequate to run DCAF authentication sequence. But considering the memory footprint of the needed libraries and operating system they can not be supported with a Class 0 device. This supports the minimum classification identified in the framework.

11.1.2 Processing power

Processing power demand of DCAF authorization sequence can be assessed based on the logged times for different tasks in the experiment results. When the log was inspected the most time consuming part of the whole sequence was the DTLS handshake between Client and Resource Server. The handshake was responsible for more than two thirds of the total time consumption of the entire sequence. This gives direction on the latency increase that can be expected in a mechanism where constrained devices are required to handle more complex calculations. This design decision can be justified with a lower power consumption compared to the alternatives, such as requesting the Authorization Manager to verify the credentials. The effects of increased latency depend on the application and they should be considered when decisions are made between lower latency of tasks versus lower power consumption.

11.1.3 Available power and energy

This brings us to the power consumption during the experiment, that can be used to estimate the available power and energy needed to conduct the DCAF authorization sequence. The first thing the results show is that the Client energy consumption on transmissions is significantly larger compared to the Resource Server. On Average the Client power consumption on transmitting was three times the value recorded from the Resource Server during the authorization sequence.

CPU power consumption readings between the Client and Resource Server did not have such a dramatic difference. The CPU power consumption difference between the actors was under twenty percent. What is noticeable the Client CPU power consumption was higher than the Servers, even though the Server conducted the calculation based on the access ticket during the Client-Server handshake.

Both actors show the highest peaks on both CPU and transmission power consumption during the authorization sequence. The Client being the more active party in this sequence seems to have larger power demand on both CPU and transmissions. The Client after all has to perform a DTLS handshake with the Client Authorization Manager and send a ticket request, in addition to the handshake and messages sent and negotiated with the Server. This brings further implications to the notion that increased computation does not have dramatic effect on power consumption.

The Clients larger power consumption on transmissions reflects why the minimizing of transmissions is an important objective. It also shows what kind of gains can be achieved by optimizing the number of transmissions over computation required from the constrained actors. This was identified in the framework under available power and energy objectives.

Since only DCAF was experimented with only speculations can be made on ABFAB authorization sequence power consumption. Based on the specification ABFAB does delegate more tasks to the Identity Provider, but the sequence includes more transmissions between the constrained actors and the Identity Provider. The total number of messages also vary depending on such things as the EAP-method used. This would suggest that ABFAB would have a larger energy footprint on the constrained devices than DCAF.

11.1.4 Network, interface, physical and cost constraints

Network constraints did not play a significant role in the experiment. Duty cycling and higher packet loss situations were not tested, because the experimental setup was not prepared to handle these situations. The lack of user interfaces, physical constraints and cost could not be assessed in a simulation based experiment. Settings and changes to the constrained actors could be easily made by compiling new versions in the simulation and physical constraints and cost are both zero in a simulation. In a real world deployment of a constrained environment these factors can play a significant role.

11.2 Architecture related security objectives

In this chapter the framework architecture related security objectives are evaluated by comparing them to the protocol specifications of DCAF and ABFAB. The delegation of demanding tasks to less constrained devices, as it is the key architecture objective, is discussed first.

11.2.1 Delegation of demanding tasks

Both protocols do delegate the more demanding tasks to less constrained actors. In DCAF this means the authorization managers CAM and SAM and in ABFAB the identity provider. The protocol architectures and mechanisms of the protocols different. DCAF delegates the key negotiation to CAM and SAM, but the access control is enforced by the Resource Server. Where as ABFAB delegates the whole access decision making process to the Identity Provider that returns the results to the Relying party. From this point of view ABFAB delegates more tasks to the less constrained actor. But on the other hand ABFAB requires more transmissions from the constrained actors, so it shifts the demand from processing to power consumption. But since no experiment on ABFAB was conducted this can not be verified.

11.2.2 Validation of actors

When DCAF protocol is used the the Client and Client Authorization Manager are able to verify that the Resource server in the level that it is associated with SAM. This is because only SAM and Resource server can produce the same Pre-shared key based from the ticket-face information.

How ever the validation needed by the Server Authorization Manager and Resource Server about the Client differs between the two protocols. The differences between the protocols are due to the different message sequences they use. This matter is discussed in more detail in the following chapter dealing with the three party authentication sequence.

11.2.3 Autonomous functionality

The third objective concerning autonomous functionality is yet another aspect that separates the two protocols. The authentication and authorization of ABFAB requires real-time interaction between the AAA-server and the Service. So the scenario where the Relying Party is unconnected at the time of authorization is not possible. The Relying Party has to have a connection to the Identity Provider at the time of authorization. AAA-servers are normally used in server environments, in which remote systems can connect over the Internet. For this reason it is assumed that the Relying Party has to have an Internet Connection and connection to a local network is not enough. On the other hand since in a push sequence type of authorization the Relying Party is handling the connections to the Identity Provider This takes some load off the Client, since it only needs to be able to connect to the Relying Party, and not directly to the Identity Provider.

DCAF can tolerate temporary network unavailability situations, since the Resource Server can enforce access control even if it would not have connection to the Server Authorization Manager at the time of communication. Client can provide all the necessary information after Client Authorization Manager has negotiated a DCAF ticket on its behalf from the Server Authorization Manager. The only problem regarding Resource Server connectivity is the ability to revoke tickets. If the Resource Server does not have connection to SAM, tickets can not be revoked. DCAF Client on the other hand needs a connection to Client Authorization Manager which in turn needs to have connection to the Server Authorization Manager at the time when authorization sequence is executed.

11.2.4 End-to-end security

The objective for secure end-to-end communication more specific to constrained environments. It realizes that the constrained devices need to communicate with entities that can reside outside their own network and identifies the need for securing these connections. When legacy protocols are used the gateway is responsible for interpreting a message coming from a constrained node, translating it to Internet package and transmitting it further. In this kind on system the security features also have to applied at the gateway for all outgoing and incoming messages. For this reason protocols using translating-gateway breaks the end-to-end security objective.

When using a IP-based communication is, that this translation phase is not needed, since the protocols are compatible. This enables that the transmission can be secured from endpoint-to-endpoint, even if one resides in a constrained node network and the other is a common Internet node. As an example DTLS it is not dependent on any specific network infrastructure and can be applied in an end-to-end manner. This feature is one of the strengths of using IP-protocols in constrained environments. The ability to provide end-to-end security can be seen as a side product of compatible protocols between constrained nodes and even node in the Internet. This does not only enable this security feature, but also possibilities build more integratable and evolvable systems. Other matters of message security are discussed in the next chapter.

11.3 Security objectives

This chapter evaluates how well the framework security objectives are enable to capture the different properties of the two proposed protocols. Most of these properties can be assessed based on the specifications so also ABFAB can also be discussed.

11.3.1 Resource security

The security objectives concerning resource confidentiality, integrity and availability are realized through the authorization procedures. DCAF Resource Servers preserve the resource confidentiality and integrity by enforcing access control to their resources. The access rules can be set for all RESTful methods individually. This said confidentiality of a resource is based on enforcing the rules for GET method and integrity is realized by enforcing the rules for POST, PUT and DELETE. Availability of resources for all authorized parties is also realized through enforcing the access rules, that enable all parties access to the resources they possess a valid access ticket. The confidentiality and integrity are enforced with the possibility to a define lifetime and possibility to revoke granted tickets. This also simplifies the access management.

ABFAB provides the same basic features on resource confidentiality, integrity and availability by enforcing access control. But since ABFAB is a used in many different environments and its roots are in larger systems, the specification does not go to details on constrained environment implementations, such as the resolution of access control in RESTful environment.

11.3.2 Message security

Message security objectives confidentiality, integrity and authentication are dependent on the encryption of messages. When message confidentiality concerning the message contents is achieved in the transport level, it is reasonably hard for third parties to gain knowledge of it. This is the case with DCAF which uses DTLS to encrypt messages between actors. The only unencrypted message in the sequence is the optional unauthorized resource request and response messages between Client and Resource Server. This message is optional, so if SAM identity is already known this message is not used. The response message contains the URI of SAM in charge of the Resource Server. This information is not critical in the wrong hands since SAM is a less constrained device capable of using necessary methods to secure it self. Even using a transport layer security does not provide total confidentiality of messages in a wireless environment. While message contents can be secured a third party can still gather information about the transmissions even if encryption holds. Eavesdropping on a wireless network can always reveal such things as number of transmissions and other activity.

Message integrity in this context means the receivers ability to detect message alteration is a side product of using transport layer security. If a message is altered during transmission by a third party, the receiver can detect this when decrypting the message. Provided of course the third party does not have the encryption key for the transmission.

Message origin can be authenticated in similar way. When two parties negotiate a DTLS connection between each other they exchange encryption keys to encrypt the transmissions. Message comes in through a DTLS encrypted transmission the origin can therefore only be from the other party which the connection has been formed.

One possibility for achieving message freshness mentioned in the framework is with DTLS replay detection. Replay detection is an optional feature of DTLS, since packet duplication is not always malicious. The technique used is the same one as in IPSec, based on maintaining a bitmap window of received records. This is used to silently discard all records previously received or too old to fit in the window. (Rescorla & Modadugu, 2012)

ABFAB does not specify or restrict the usage of any specific application protocol or specific transport security. So the same examples of how to achieve message confidentiality, integrity, authenticity and freshness apply to ABFAB.

11.3.3 Access control architecture

When access control architecture is concerned DCAF uses a capability based access control model in the constrained level. This has several advantages compared to other approaches, such as the Resource Server has no need to store client privilege information. Those kind of approaches would work only on very small deployments, due to limited storage capacity for user information. The architecture does not limit on combining actors if they reside within a same domain. Similarly the architecture does not mention limitations on combining the authorization managers to other systems. So they could be integrated to other systems as long as they perform the specified tasks.

On the constrained level ABFAB uses identity based access control. Since ABFAB is based on AAA-framework it assumes framework services are present. But if an existing AAA-infrastructure exists, an ABFAB implementation of constrained devices can be integrated seamlessly. This solution would realize a system wide uniform authorization management, where access control configuration for all parts can be done in a centralized way.

11.3.4 Message sequence for three party authentication

As mentioned in the previous chapter validation of the Client between the protocols bares difference. DCAF has a disadvantage inherent to message sequence it uses so the protocol is based on trust between the two authorization managers. This means that the Server Authorization Manager has no direct means to validate the origins of a resource request. Sever Authorization Manager can only trust the Client Authorization Manager for not giving out or disclosing authorization tickets that are used as session keys. (Gerdes et al., 2015c)

ABFAB has different set of features relating to this same problem. ABFAB architecture it is able to authenticate any of the actors, including the client. This is due to the push sequence used in the authentication messaging. So the Relying Party can authenticate the Client in an adequate level that depends on the policies set for it and the Identity Provider. This is based on the trust based on the federation substrate mechanism.

11.4 Use case security requirements

In this chapter the frameworks security objectives are operationalized by use case requirements presented in chapter 8. Most of the use case requirements operationalize more than one framework objectives. Few of the framework objectives also have dependencies to multiple requirements. The only two objectives that did not have clear dependency to any of the use case requirements were message freshness and access control architecture. Table 14 presents the dependencies found between use case requirements and framework objectives.

TABLE 14 Dependencies between use case requirements and framework objectives

Requirement	**Framework objective**
Integrity & authenticity of sensor data	Resource integrity
	Message integrity
	Message authentication
Confidentiality of sensor data	Message confidentiality
Authorization by resource and requesting party basis	Validation of actors
	Message sequence for three party authentication
	Resource confidentiality
	Resource integrity
	Resource availability
Autonomous authorization	Delegation of demanding tasks
	Autonomous functionality
Temporary access permissions	Resource confidentiality
	Resource integrity
End-to-end security	End-to-end security

Based on these dependencies a system that would satisfy these requirements can be envisioned. The main categories of properties this kind of system needs to possess in the highest level are resource and message security.

The rest of this chapter discusses these dependencies in more detail and ways to meet the requirements.

11.4.1 Integrity & authenticity of sensor data

The first functional requirement extracted form the use case requiring integrity and authenticity of sensor data operationalize objectives from both resource and message security. This is due to the fact that quality of monitored readings can be effected by both tampering the resources them selves or messages exchanged between the server and client. This requirement operationalizes integrity objectives for both resources and messages messages. In addition it operationalizes authentication objective for messages. As described in the previous chapter resource integrity can be achieved by enforcing access control, in this case the access rights to change the resources. Message integrity is achieved through using a secure channel for communication between the server and client requesting the data. So to put this in the use case context the quality of monitored readings and identification data served for the climate control and logistics applications can be achieved through enforcing the access control and using a secure communications channel.

11.4.2 Confidentiality of sensor data

The description of the second functional requirement "Confidentiality of sensor data" focuses purely on securing the transmitted data, so it operationalizes only one message security objective, which is message confidentiality. As discussed in the previous chapter message confidentiality is also a matter of using a secure channel for communication. The solution for this requirement can be formulated: data transmissions containing quality and location information can be secured by using a secure communications channel for transmissions.

11.4.3 Authorization by resource and requesting party basis

Authorization by resource and requesting party basis is a matter of operationalization of all objectives concerning resource security. Namely confidentiality, integrity and availability of resources. But this requirement also operationalize two more objectives concerning actor validation. The architecture based objectives define the baseline objective for validation of actors, which is then made more specific by the objective concerning the message sequence used.

This issue is discussed in the previous chapter in the context of the protocols. The requirement it self has two dimensions. First it acknowledges that one node can serve more than one resources and different resources are required to have their unique set of access rights. This rules out the possibility of using a node wide global access control. How ever this requirement does not impose the need for explicit access control, where the rights to a resource are determined in method level, such as read and write privileges. The second dimension is the authentication dimension. A requesting party can be realized as a specific user or a group of users. Where the single user can be authenticated in different levels of identity attributes. In any case this requirement can be fulfilled by resource security objectives namely enforcing access control. So the solution for this requirement is: authorization by resource and requesting party basis can be achieved by enforcing fine-grained access control.

11.4.4 Autonomous authorization

Autonomous authorization requirement in the use case requires the system has the ability to enforce authorization also in such situations where there is no Internet connectivity or the resource owners cannot take part in the process.

This requirement has a strong dependency on autonomous functionality objective. But it also has a dependency on delegation of demanding tasks objective, which can influence this requirement through the selected delegation approach. This is due to larger demand on connectivity when more task are delegated away from the constrained actors. For these reasons this requirement operationalizes both of these objectives.

This matter can also be divided into two parts. The system needs to be able to function without intervention from the resource owners and it has to be able to function without Internet connection. In order to achieve the first part the system has needs to support adequate level of configurability, to be able to full fill this responsibility with out the need for actions from the resource owner. For the second part of being able to function with limited connectivity one solution is to use capability based authentication, where the resource server can enforce access control even when it does not have connectivity to it less constrained counterparts.

11.4.5 Temporary access permissions

This requirement operationalizes the resource confidentiality and integrity objectives. These objectives were also operationalized by previous requirements concerning resource security.

A solution for this requirement dictates a need for a way to set a lifetimes for a permissions and the ability from access control to enforce them. The matter of revoking granted tickets requires a similar mechanism. Enforcing these features have some differences between authorization methods in constrained environments. These differences can be caused by such things as limited connectivity.

Life time of an access permission is pretty straight forward, since the server has knowledge of this starting from the initial authorization negotiated. Even with limited connectivity the enforcing party has knowledge when the authorization is no longer valid.

The revoking of a once granted access permission in a limited connectivity situation on the other hand can pose a problem. An example of this would be a situation, where a resource server has limited connectivity and capability based authentication is used. A client possesses a access ticket for a resource, which is then revoked. The problem arises when the information of the revocation needs to be transmitted to the resource server. The server has has no connection present for its less constrained counterpart and therefore no knowledge that the access ticket has been revoked. The solution for this requirement in any case fall down to configuring and enforcing access control in the level of detail needed.

11.4.6 End-to-end security

End-to-end security has a strong dependency on the framework objective of the same name, which it there fore operationalizes.

End-to-end security requirement expand the requirement for sensor data confidentiality to state that transmission should be secured from endpoint-to-endpoint. This matter was discussed earlier as a part of architecture objectives and it can be solved by using a security protocol able to crossing the necessary borders on negotiating a secure communications channel. When constrained environments are concerned the borders can include different transport mediums and underlying network protocols. This rules out systems that use translating gateways since they break the end-to-end security when the other endpoint resides outside of their own network. The solution fulfilling this objective to use transport level security able to use any underlying network such as DTLS.

11.5 Evaluation of the main artifact

Based on the evaluations done previously in this chapter the framework seemed to be able capture the different features needed for an authorized authentication mechanism to be used in constrained environments. All three levels, namely constraints, architecture objectives and security objectives served the a purpose of bringing unique features to strengthen the framework.

The constraints level identified the different constraints and provided taxonomies on memory and power constraints. The memory taxonomy enables the identification of minimum ability the devices need for safe conduct. It also shows the level where normal Internet protocols can be used. The constraint level inspection identified and verified several effects the constraints can cause. These effect included such things as latency due to low processing ability or rising total power consumption due to large number of transmissions. These were tested with an experiment, which showed processor usage and transmitting have effects on device power consumption and latency. Transmitting being the dominant factor on power consumption and processing on increased latency.

The architecture level objectives enabled the identification of several differences in the protocol functionalities, such as difference in delegation strategies, ability to identify all the actors and autonomous functionality. These differences bring decision points on which property to optimize over another. These decisions are such as optimizing power consumption over processing required from the constrained nodes or possible latency gains over autonomous functionality. In addition to these the the architecture level identifies what kind of gains on security can be achieved by end-to-end security.

The security objective level identified two main areas for realizing a secure distributed system. These areas are resource and message security, both having sub areas. The framework was enable to identify these areas as objectives and therefore the protocols could be evaluated based on the mechanisms they provide to each objective.

When the framework security objectives derived from the architecture and security concerns were operationalized as requirements of a logistics use case, dependencies for most objectives could be found. Requirements concerning resource and message security had dependencies on objectives identified at the security concerns level. Some of the requirements having dependencies on security level were supplemented by objectives from the architecture level. But there were two requirements that operationalized only architecture level objectives. These were the requirements for autonomous authorization and end-to-end security. These requirements are more constrained environment specific so they could not be identified by the common security objectives for distributed systems. Overall the framework provided one or more objectives for every requirement in the use case.

12 Conclusions

This study was conducted using the Design Science Research Method which is essentially a problem solving paradigm consisting of six activities. In the first activity the problem was identified as a lack of dynamic authorized authentication mechanism enable to function in constrained environments.

Based on the problem identification activity the objective for this study was set to create an artifact, an authorized authentication evaluation framework for constrained environments. The framework was to be developed to identify the environment constraints, prerequisites for a secure distributed system and how these two can coexist in a single system.

The framework development activity already gave answers to the first two sub-research questions: *RQ 1.1 What kind of constraints do the devices have?* and *RQ 1.2 How the device constraints should be classified?*

In the demonstration activity the framework was used to evaluate the properties of two protocols. This provided answers to the rest of the research questions: *RQ 1.3 Which constraints have effect on choosing the mechanism?* and first of the main research questions: *RQ 1:What are the prerequisites for establishing authorized authentication mechanism between two devices when one or both have constrained capabilities?* Then framework security objectives were operationalized to use case requirements to provide answers to the second main research question: *RQ 2: What are the requirements for a system supporting authorized authentication between two constrained devices?* and its sub question: *RQ 2.1 What kind of system could satisfy these requirements?*

With all the research questions answered the evaluation activity concentrated on evaluating the framework on its ability to capture the properties it was designed to capture. The final activity of the process is communicating the results which is this document.

To justify the value of the this research terminology from requirements engineering can be used, even though the main artifact is not a machine per se. The problem world could be described as follows. The first state of the problem world system-as-is in this case is a situation where no framework for evaluating dynamic authorized authentication protocols exists. The system-to-be state is the situation where a such a framework exists. The why-dimension limitations on identifying constraints and security objectives and their dependencies is harder to do with system-as-is. The opportunity to be exploited to make this process easier. The what-dimension identifies a framework for evaluating authorized authentication solutions as a service needed to satisfy the objectives identified in the why-dimension. The who-dimension assigns the responsibilities defined in the why- and what-dimensions to the artifact.

Evaluation of the main artifact in the previous chapter showed that it was able to identify perquisites to realize authorized authentication in constrained environments at least when this use case and these two protocols are evaluated. A good subject to a future research would be to add more use cases and proposed protocols to determine if the resolution of the framework would be adequate to evaluate them.

The requirements which in this case were the solutions ability to meet the constraints posed by the environment constraints were identified by the first level of the framework that specified the areas of constraints to be evaluated. The evaluation was based on protocol specifications and an experiment where DCAF authorization sequence was studied. The experiment showed that the constraints identified in the framework, namely, power consumption and processing power had the effects predicted. The results showed that the number of transmissions the constrained node has to perform, increases its power usage substantially. The processing required from a constrained node did not show similar effect on node power consumption.

This is a crucial decision point when a mechanism for authorization is chosen. This decision is between delegating more tasks away from constrained devices and minimizing the transmissions needed. From the power consumption point of view it can sometimes be wise to optimize to the direction of less transmissions over processing assigned to the constrained actors. But if the power consumption is not an issue in a particular system, the decision could be the opposite. This could be the case if the optimizing need to be done in the direction of fast operation. Such as a user application where long response times are not accepted, since people tend to get anxious when a action takes more than a few seconds. Other constraints identified in the framework could not be evaluated in this extent based on the experiment or the protocols being reviewed.

After all this said about the power footprint of the authorization sequences, based on the experiment the power usage can still be considered reasonable. Comparing the power consumption readings with the capacity of an ordinary battery showed that DCAF authorization sequence could be run more than thousand times before the batteries run out. Since an authorization sequences are not that frequent in normal situations this suggests that an acceptable battery life can be achieved.

The first stage of evaluating the security related objectives identified by the framework was applying them to the specifications of the two proposed protocols DCAF and ABFAB. This stage showed that the framework objectives seemed to capture protocol features quite well. The objectives for the main aspects of security which are securing resources and transmissions had good enough resolution to identify the technologies needed for a solution. The framework was also enable to differentiate the protocols on several features and there fore supply answers on questions about the effect different approaches on authorized authentication can cause. A good example of these difference is the effect a delegation strategy has to autonomous functionality. When more tasks are delegated to the less constrained actors it can mean decrease in ability to autonomous functionality. The architecture based requirements only specified that the other actors need to be able to function without the supervision from the principal level actors. The ability of a constrained actor to function with limited connectivity to its less constrained counterpart can in some cases be advantageous in its own right. One such situation can be envisioned based on the logistics use case. If a resource server does not have connection to it's authorization manager, due to radio signals being dampened by the high water content cargo. If the system uses capability based authorization, this server would still be able to validate tickets and enforce access control in this extent.

When the frameworks security related objectives were operationalized to a use case the primary security objectives turned out to be the objectives demanding for resource and transmission security. These objectives were operationalize by the four out of six use case requirements defining the need for confidentiality, integrity and availability to resources and confidentiality, integrity and authenticity to the messages. These requirements can be seen as some what essential, since they define the base line requirements for secure conduct.

The higher-level objectives for resource security were complemented by operationalizing more fine-grained objectives concerning validation level of actors and end-to-end security. These objectives are the security objectives derived from the ACE architecture document, that describe more constrained environment specific security objectives. Only two out of six requirements were operationalized by these objectives alone. Both of these requirements can be seen more or less constrained environment specific, since they specified the need for autonomous authorization and end-to-end security capabilities. These kind of objectives get less attention in Internet systems since connectivity and protecting transmissions are pretty much taken for granted.

This use case showed that most of the requirements for authorized authentication in constrained environments can be achieved by common security objectives for distributed systems. The security requirements them selves do not change when moving from a larger less constrained system to a more constricted one. The major difference for the requirements comes from how to realize these requirements with constrained resources. For this a set of more finer-grained objectives requirements can be a big help. But as the results of this study have shown even a battery powered device with Class 1 memory constraints, can fulfill the security objectives described by the framework reasonably well.

REFERENCES

Anderson, J. P. (1972). *Computer Security Technology Planning Study. Volume 2*. DTIC Document. Retrieved from http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=AD0772806

Chadwick, D. W. (2009). Federated identity management. In *Foundations of security analysis and design V* (pp. 96–120). Springer. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-03829-7_3

Dunkels, A., Eriksson, J., Finne, N., & Tsiftes, N. (2011). Powertrace: Network-level power profiling for low-power wireless networks. Retrieved from http://soda.swedish-ict.se/4112/

Dunkels, A., Grönvall, B., & Voigt, T. (2004). Contiki-a lightweight and flexible operating system for tiny networked sensors. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on* (pp. 455–462). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1367266

Erdtman, S., Seitz, L., Wahlstroem, E., Selander, G., & Tschofenig, H. (2016, October 25). Authorization for the Internet of Things using OAuth 2.0. Retrieved March 28, 2016, from https://tools.ietf.org/html/draft-ietf-ace-oauth-authz-01

Eriksson, J., Österlind, F., Finne, N., Tsiftes, N., Dunkels, A., Voigt, T., … Marrón, P. J. (2009). COOJA/MSPSim: interoperability testing for wireless sensor networks. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques* (p. 27). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). Retrieved from http://dl.acm.org/citation.cfm?id=1537650

Farrell, S., Holdrege, M., Laat, C. T. A. M. de, Calhoun, P. R., Gommans, L., Vollbrecht, J. R., … Spence, D. W. (2000). AAA Authorization Framework. Retrieved from https://tools.ietf.org/html/rfc2904

Gerdes, S., Bergmann, O., & Bormann, C. (2015c). Delegated CoAP Authentication and Authorization Framework (DCAF). Retrieved January 23, 2016, from https://tools.ietf.org/html/draft-gerdes-ace-dcaf-authorize-04

Gerdes, Bergmann, O., & Bormann, C. (2014). Delegated Authenticated Authorization for Constrained Environments. In *2014 IEEE 22nd*

International Conference on Network Protocols (ICNP) (pp. 654–659). http://doi.org/10.1109/ICNP.2014.104

Gerdes, Seitz, L., Selander, G., & Bormann, C. (2015a). An architecture for authorization in constrained environments. Retrieved from https://tools.ietf.org/html/draft-ietf-ace-actors-02

Gusmeroli, S., Piccione, S., & Rotondi, D. (2013). A capability-based security approach to manage access control in the Internet of Things. *Mathematical and Computer Modelling, 58*(5–6), 1189–1205. http://doi.org/10.1016/j.mcm.2013.02.006

Haley, C. B., Laney, R., Moffett, J. D., & Nuseibeh, B. (2008). Security requirements engineering: A framework for representation and analysis. *Software Engineering, IEEE Transactions on, 34*(1), 133–153.

Han, S. N., Cao, Q. H., Alinia, B., & Crespi, N. (2015). Design, Implementation, and Evaluation of 6LoWPAN for Home and Building Automation in the Internet of Things. Retrieved from https://www.researchgate.net/profile/Son_Han/publication/285236710_Design_Implementation_and_Evaluation_of_6LoWPAN_for_Home_and_Building_Automation_in_the_Internet_of_Things/links/565cf2db08aefe619b254d0d.pdf

Herberg, U., Romascanu, D., Ersue, M., & Schoenwaelder, J. (2015, May). Management of Networks with Constrained Devices: Problem Statement and Requirements. Retrieved January 26, 2016, from http://tools.ietf.org/html/rfc7547

Hernández-Ramos, J. L., Jara, A. J., Marın, L., & Skarmeta, A. F. (2013). Distributed capability-based access control for the internet of things. *Journal of Internet Services and Information Security (JISIS), 3*(3/4), 1–16.

Hersent, O., Boswarthick, D., & Elloumi, O. (2011). *The Internet of Things: Key applications and protocols*. John Wiley & Sons.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly, 28*(1), 75–105.

ITU-T. (2009a). *Baseline capabilities for enhanced global identity management and interoperabilityITU-T X.1250*. International Telecommunication Union. Retrieved from http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=9456

ITU-T. (2009b). *NGN identity management framework Y.2720*. International Telecommunication Union. Retrieved from http://www.itu.int/ITU-T/recommendations/rec.aspx?id=9574&lang=en

ITU-T. (1991). *Security Architecture For Open Systems Interconnection For CCITT Applications, Recommendation X.800*. International

Telecommunication Union. Retrieved from http://www.itu.int/ITU-T/recommendations/rec.aspx?id=3102&lang=en

Karp, A. H. (2006). Authorization-based access control for the services oriented architecture. In *Creating, Connecting and Collaborating through Computing, 2006. C5'06. The Fourth International Conference on* (pp. 160–167). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4019390

Keranen, A., Ersue, M., & Bormann, C. (2014, May). Terminology for Constrained-Node Networks. Retrieved January 23, 2016, from http://tools.ietf.org/html/rfc7228

Kortuem, G., Kawsar, F., Fitton, D., & Sundramoorthy, V. (2010). Smart objects as building blocks for the Internet of things. *IEEE Internet Computing, 14*(1), 44–51. http://doi.org/10.1109/MIC.2009.143

Kothmayr, T., Schmitt, C., Hu, W., Brünig, M., & Carle, G. (2013). DTLS based security and two-way authentication for the Internet of Things. *Ad Hoc Networks, 11*(8), 2710–2723.

Kugler, P., Nordhus, P., & Eskofier, B. (2013). Shimmer, Cooja and Contiki: A new toolset for the simulation of on-node signal processing algorithms. In *Body Sensor Networks (BSN), 2013 IEEE International Conference on* (pp. 1–6). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6575497

Mainetti, L., Patrono, L., & Vilei, A. (2011). Evolution of wireless sensor networks towards the Internet of Things: A survey. In *2011 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)* (pp. 1–6).

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems, 15*(4), 251–266.

Nguyen, K. T., Laurent, M., & Oualha, N. (2015). Survey on secure communication protocols for the Internet of Things. *Ad Hoc Networks*. Retrieved from http://www.sciencedirect.com/science/article/pii/S1570870515000141

Osterlind, F., Dunkels, A., Eriksson, J., Finne, N., & Voigt, T. (2006). Cross-level sensor network simulation with cooja. In *Local computer networks, proceedings 2006 31st IEEE conference on* (pp. 641–648). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4116633

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems, 24*(3), 45–77.

Perez-Mendez, A., Pereniguez-Garcia, F., Marin-Lopez, R., Lopez-Millan, G., & Howlett, J. (2014). Identity Federations Beyond the Web: A Survey. *IEEE Communications Surveys Tutorials*, *16*(4), 2125–2141. http://doi.org/10.1109/COMST.2014.2323430

Pfleeger, C. P., Pfleeger, S. L., & Margulies, J. (2015). *Security in Computing (5th Edition)* (5th ed.). Upper Saddle River, NJ, USA: Prentice Hall Press.

Raza, S., Trabalza, D., & Voigt, T. (2012). 6LoWPAN compressed DTLS for CoAP. In *Distributed Computing in Sensor Systems (DCOSS), 2012 IEEE 8th International Conference on* (pp. 287–289). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6227754

Rescorla, E., & Modadugu, N. (2012). Datagram transport layer security version 1.2. Retrieved from http://tools.ietf.org/html/rfc6347.txt

Selander, G., Mani, M., Kumar, S., Seitz, L., & Gerdes, S. (2015, b). Use Cases for Authentication and Authorization in Constrained Environments. Retrieved from https://tools.ietf.org/html/draft-ietf-ace-usecases-10

Shelby, Z., & Bormann, C. (2009). *6LoWPAN: The wireless embedded Internet* (Vol. 43). John Wiley & Sons.

Shelby, Z., Hartke, K., & Bormann, C. (2014). The constrained application protocol (CoAP). Retrieved from http://tools.ietf.org/html/rfc7252%23section-5.5.2

Shirey, R. (2007). Internet Security Glossary, Version 2. Retrieved January 23, 2016, from https://tools.ietf.org/html/rfc4949

Skinner, G. D. (2009). Cyber security management of access controls in digital ecosystems and distributed environments | NOVA. The University of Newcastle's Digital Repository, Cyber security management of access controls in digital ecosystems and distributed environments. Retrieved from http://novaprd-lb.newcastle.edu.au:8080/vital/access/manager/Repository/uon:9103?f0=sm_subject%3A%22access+controls%22

Texas Instruments. (2007). CC2520 2.4 Ghz IEEE 802.15.4 / ZigBee RF Transceiver Datasheet. Retrieved May 19, 2016, from http://www.ti.com/product/CC2520

Texas Instruments. (2014). MSP430F5437 and MSP430F5x/6x Mixed-Signal Microcontrollers Datasheet. Retrieved May 19, 2016, from http://www.ti.com/product/MSP430F5437

Titzer, B. L., Lee, D. K., & Palsberg, J. (2005). Avrora: Scalable sensor network simulation with precise timing. In *Proceedings of the 4th international symposium on Information processing in sensor networks* (p. 67). IEEE Press. Retrieved from http://dl.acm.org/citation.cfm?id=1147768

Tschofenig, Hartman, S., Lear, E., Schaad, J., & Howlett, J. (2014). Application Bridging for Federated Access Beyond Web (ABFAB) Architecture. Retrieved February 9, 2016, from https://tools.ietf.org/html/draft-ietf-abfab-arch-13

Tsvetkov, T. (2011). RPL: IPv6 Routing Protocol for Low Power and Lossy Networks. *Sensor Nodes–Operation, Network and Application (SN)*, *59*, 2.

Van Lamsweerde, A. (2009). *Requirements engineering: from system goals to UML models to software specifications*. Retrieved from http://dial.uclouvain.be/handle/boreal:78351

Vasseur, J.-P., & Dunkels, A. (2010). *Interconnecting Smart Objects with IP: The Next Internet*. Morgan Kaufmann.