SICS Technical Report T2002:16
ISSN 1100-3154
ISRN:SICS-T--2002/16-SE

What AMANDA offers

- A comparative case study describing a flexible and decentralised approach for Authorisation Management -

Fredrik Rosenhamer

Master's Thesis*
Department of Computer and Systems Sciences
Stockholm University/Royal Institute of Technology

November 2001

Abstract

In this thesis the term Authorisation Management (AM) refers to a process that begins in the real world when a decision is made concerning the delegation of authorisations. Such a decision is governed by policies. The process ends when the decision has been implemented within some computerised control mechanism in the IT-world. Today most of this process takes place in the real world. The authorisation-decision typically takes the form of a signed piece of paper that somehow is communicated to an administrator. The administrator then implements this decision, made by someone else. Besides enabling the implementation of an authorisation-decision, the process does not add any value to an organisation. It is manual, slow, involves several people and each time a decision is made, the whole process has to be initiated and performed. Further, the decision has to be expressed and implemented in terms of existing models and mechanisms and only the administrator interacts with the computerised control-mechanism in the IT-world. No widely used alternative exists.

In a project named AMANDA (Authorisation Management for Distributed Applications) at the Swedish Institute of Computer Science (SICS) an alternative is being developed. AMANDA offers a mechanism that will allow AM to be decentralised in accordance with the ordinary chain of command. Using a graphical user interface, the decision-maker will implement his decision and it will take effect immediately. AMANDA will be flexible and will closely map and represent real world policies. Assuming the existence of a Public Key Infrastructure, attribute certificates are used to delegate authorisations, if needed in several steps. This thesis examines how AMANDA could simplify and improve AM. The theoretical part of this thesis describes AMANDA and the foundation on which she rests. The empirical part consists of a case study in a specific setting. First, the actual AM-process of today, with respect to a specific application, is modelled and described. Then, the future AM-process using AMANDA is modelled and described. The results indicate that AMANDA would offer a more flexible, precise, fast and secure way of AM in accordance with the operational chain of command. Though not considered in the problem statement, another result is the finding that no approach seems to exist towards modelling and describing AM as a process of it's own. In order to perform the case study, ideas from enterprise modelling has been used to identify and understand the AM-process. Together with the Unified Modelling Language (UML), Enterprise Modelling has also inspired the notation used in the case study.

* This thesis corresponds to 20 weeks of full-time work.

Keywords: Attribute Certificates, Authorisation Management, Authorisation Management process, Authorisation Management for Distributed Applications, AMANDA, Trust Management, real world, IT-world, authorisation, privilege, delegation, management-level power, power, empowerment, object-level permission, permission, enterprise modelling, Keon.

ABBREVIATIONS

Abbreviations commonly used in this thesis are spelled out below. The number within parentheses refers to the page where the abbreviation is first used and the term explained.

AA – Attribute Authority (24)
AC – Attribute Certificate (20)
ACL – Access Control List (1, 15)
ACL – An AMANDA component. (24)
ACM – Access Control Matrix (15)
AM – Authorisation Management (1)
AMANDA – Authorisation Management for Distributed Applications (2)
AP – Application Policy, an AMANDA component (23)
CA – Certification Authority (9)
CRL – Certificate Revocation List (9)
CERT – Certificate database, an AMANDA component. (24)
DAC – Discretionary Access Control (15)
DBC - Database-Credentials (60)
DERCT - Declarative Entity-Relationship with Causation and Time - a language for Enterprise and Information System Modelling (27)
FACT – Fact database, an AMANDA component. (24)
GC - Given-Credentials (60)
Keon – A Public Key Solution provided by RSA Security Inc. (3, 59)
LAN – Local Area Network (34)
LDAP – Lightweight Directory Access Protocol (9)
MAC – Mandatory Access Control (15)
MP - Meta-policy, an AMANDA component (23)
O – The Organisation – the organisation studied in the thesis (5, 33)
ODB – Organisational Database, an AMANDA component (24)
PAC – Privilege Attribute Certificate (32, 35)
PKC – Public Key Certificate (20)
PKI – Public Key Infrastructure (1, 8)
RA – Registration Authority (10)
RBAC – Role Based Access Control (15)
SS – Security Server, a Keon Component (35)
SSL – Secure Socket Layer (10, 62)
SSU – Sales and Service Unit (33)
TCSEC – Trusted Computer System Evaluation Criteria (15)
TM – Trust Management (1, 17)
UC – User-Credentials (60)
UML – Unified Modelling Language (4, 30)
VPN – Virtual Private Network (10)

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

The aim of this chapter is to introduce the field of study, some central terminology and to present the setting in which the empirical part of the thesis was carried out. The problem statement and the purpose of writing this thesis are also stated. The disposition of the rest of the thesis and an introduction of the methodology used close the chapter.

1.1. PROBLEM BACKGROUND

Ever-increasing possibilities enabled by computers and networks continue to change every aspect of life. This fact increases the need of being able to interact trustingly across networks as well as the need to control different types of resources. Public Key Infrastructures (PKI) offers an opportunity to interact trustingly and securely across networks using public key certificates and cryptography. However, traditional security models and mechanisms used to control access to and use of different types of resources, whether in closed or open environments, are still fairly primitive. Possibly a result of inheritance limiting imagination, the authorisations that are possible to express in computerised systems and the way they are implemented and administrated does not meet the needs of the real world.

In a somewhat delimited environment, such as an organisation, it is important that authorisations are administrated, or managed, according to changing real-world policies, organisational structures and chain of commands. In this thesis, Authorisation Management (AM), refers to everything that has to be considered and done in order to enable the use of some resource controlled by a computerised system. A common AM-process functions as follows: A decision-maker, authorised according to real-world policies, delegate authorisations to use resources controlled by a computerised system. The resource may be the system itself or something else such as being allowed physical access to a building. The decision, typically a signed piece of paper, is motivated by policies. The decision then has to be physically forwarded to a designated administrator who will implement the decision by altering an Access Control List (ACL). ACLs are static and the administrator will implement a decision made by someone else. Regardless of what the real-world policies state the authorisation-decision has to be expressed in terms of the existing models and mechanisms in use. This is imprecise. The process of transporting the decision and implementing it is largely a manual process existing only to enable the authorised user to use the system. This is slow and the number of people and the manual routines involved makes the process potentially insecure. Further, the process does not follow the normal operational chain of command.

Therefore, it would be desirable if the Authorisation Management mechanisms of a computerised system could be made expressive and flexible enough as to more thoroughly map and represent real-world policies. It would also be desirable if the empowered and responsible decision-maker himself could implement his decision. That is, if the enforcement mechanism could be decentralised as to map the operational chain of command. Here we by e*mpowerment* mean the right to delegate authorisations, possibly in several steps.

Focusing on open environments with previously unknown users, *Trust Management* (TM) is one approach towards a more flexible and decentralised authorisation-mechanism. The general idea is that each system or application has it's own computerised representation (P) of certain real-world policies. (P) expresses both the set of actions possible to perform within the

system as well as who may perform actions and how the right to perform actions may be delegated. A request to perform an action (r) is accomplished by a reference to (P) and credentials (C). Credentials are actions, as expressed in the policy, which are signed by someone who, according to the same policy, may delegate the right to perform an action. Actions are events such as read or write that are controlled by the system. Credentials are forwarded using some sort of certificates. These certificates bind keys to authorisations unlike public key certificates that bind a key to an identity. In this thesis, certificates that delegate authorisations are referred to as *attribute certificates*.

The signature of an authorised decision-maker may have to be derived through a hierarchy of digital signatures. A certain compliance-checker, based on a general-purpose algorithm, will check whether (r, C, P) complies and approve or reject the request.

Inspired by the ideas of Trust Management the Swedish Institute of Computer Science (SICS) currently performs research in a project named *Authorisation Management for Distributed Applications (AMANDA)*.

This project strives at developing a language for representing real-world policies and an algorithm to be used by the engine making the authorisation decision. The interest lies mainly in representing dynamic policies where the setting of the policies, the authorisation decision as well as its implementation is decentralised and constrained according to prevailing policies, organisational structure, chain of command and other aspects in the real world that need to be considered.

AMANDA addresses how to be able to *delegate* authorisations in several steps. That is, being able to delegate the right to delegate the right, etc, to do something within a system. AMANDA uses attribute certificates to express and communicate decisions to delegate authorisations. However, in AMANDA, the revoking of a certificate somewhere up a certificate chain does not automatically mean that all authorisations motivated by this power are revoked. The default setting is that each privilege is unique. Further, whereas the reasoning about compliance between policies and certificates are an appealing approach towards delegating authorisations, the majority of questions are likely to be of a simple look-up type. Therefore ACLs and other similar solutions are used for looking up the large bulk of usage- and AM-requests that are likely to be made in an organisation. In fact, all presently valid facts with respect to authorisations and issued attribute certificates are kept in different tables. This simplifies revocation as well as control of the system.

The AMANDA research-group has a need to find out if real-world policies can be translated and mapped into the mechanisms being developed and if these mechanisms will function as intended. The organisation described in the empirical part of this thesis has a need to more thoroughly understand how and according to what rules authorisations are administrated today and how this may be improved.

1.2. PROBLEM STATEMENT

How would an AMANDA-based mechanism simplify the full process of AM within an organisation and at the same time improve flexibility and the ability to express and delegate authorisations in close accordance with real world policies?

1.3. OBJECTIVES

The purpose of this thesis is to exemplify and demonstrate the proposed benefits of using an AMANDA-based AM-mechanism. This is done through a case study.

1.4. LIMITATIONS

The case study is performed within the context of a large organisation[1] with an existing PKI provided by Keon[2]. The case study focuses on the full process of Authorisation Management concerning a specific application.

1.5. INTENDED AUDIENCE

The intended reader should have a basic knowledge of computer science with a particular interest in IT-security and Authorisation Management. Readers familiar with Public Key Infrastructures (PKIs) and the concept of trust may choose to skip sections 2.1 and 2.2.

[1] The specific organisation will in this thesis remain unidentified due to confidentiality aspects.
[2] Keon is a registered trademark of RSA Security Inc.

1.6. INTRODUCING THE CONTENTS OF THE THESIS AND PRESENTING THE DISPOSITION

The main contents of this thesis are introduced below and the sequential order in which the chapters are presented is visualised in figure 1. Each chapter and certain sections are introduced with a short disposition.

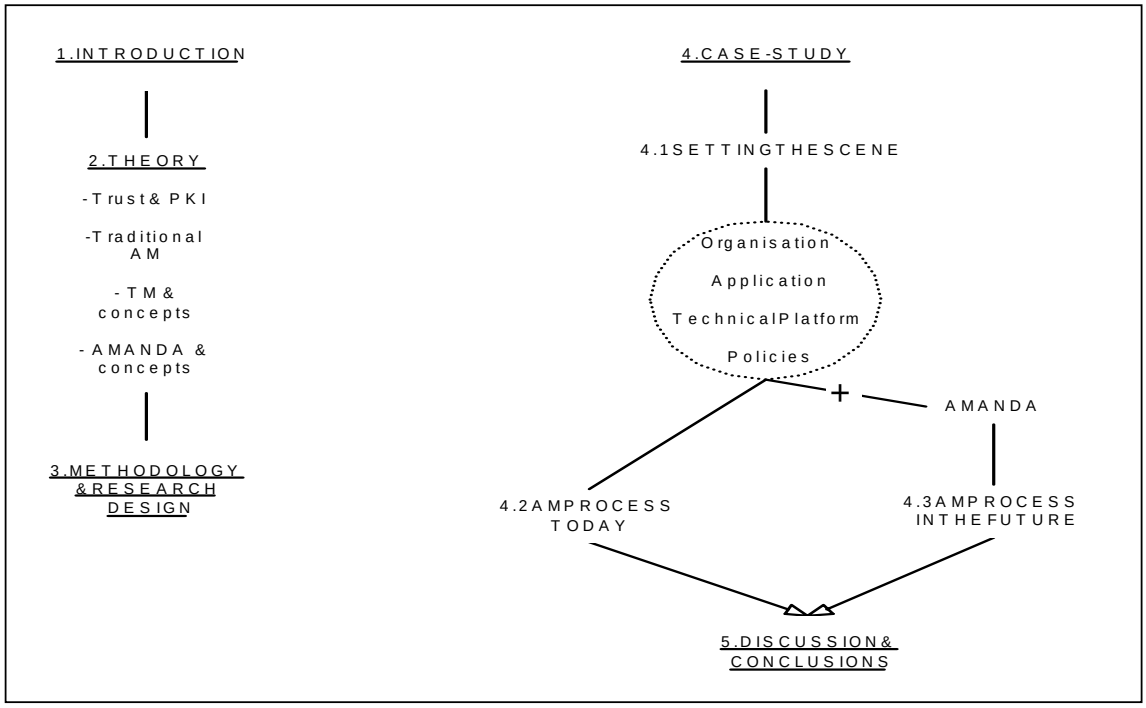

Figure 1: Thesis disposition.

THEORY begins by presenting and discussing the concept of trust. An elementary presentation of the ideas and the functionality of Public Key Infrastructures follow. Next, models and mechanisms commonly used for Authorisation Management, together with associated concepts, are described and discussed. Finally, Trust Management is introduced and described, as is AMANDA. This block is based on theoretical studies and also, concerning Trust Management and in particular AMANDA, to a large extent discussions with Babak Sadighi Firozabadi at SICS.

The aim of the case study has been to identify, understand and describe the full process of AM of both today and in the future. Today, each decision to delegate authorisations triggers an AM-process that starts when a decision concerning an authorisation is requested and ends when a user has been authorised and can make use of his privilege within the computerised system. However, no approach seems to exist towards helping to understand, model and formally structure and describe this AM-process. In order to be able to do this, ideas of Enterprise Modelling have been applied. This has helped to understand the interrelation between interpreting policies, making decisions, communicating them through some manual administrative process and finally implementing them within the technology of a computerised system. A simplified version of the Unified Modelling Language (UML) use-cases, complemented with some notation from Enterprise Modelling, is used to present these

understandings in CASE STUDY. This is described in more detail in METHODOLOGY AND RESEARCH DESIGN, as are other aspects of interest with regard to describing the creation of this thesis.

CASE STUDY is introduced by describing the Organisation (O), the focal application (the Application), the PKI technical platform Keon and the real world policies that govern the full AM-process. These real world policies regulate both the delegation of authorisations as well as the administration and implementation of decisions concerning authorisations. Great effort has been made to identify the policies of interest for the focal application and to analyse and formalise them into terms relevant for AMANDA. O, the Application, Keon PKI and the real world policies are the independent variables of the study. The dependent variables are the AM- and usage-processes. Focus is on changes in these processes when AMANDA is added.

AM PROCESS TODAY presents two use-cases, one describing the full AM-process of today and another describing usage of the Application today. This block is based on studies of written policies and technical documents as well as numerous formal and informal interviews with personnel with the organisation. Interviewees include IT-security-specialists, the system-owner, system-administrators and Keon-specialists.

In AM PROCESS IN THE FUTURE the scene is the same as in AM-PROCESS TODAY but AMANDA has been added. First the components of AMANDA, within the setting of the independent variables, are described. This description is mainly derived from the previous descriptions of O, the Application and the real world policies. Also, some changes have been made within the technical platform of Keon PKI. These changes do not affect the proposed functionality of AMANDA but are still described and considered, as O will be introducing them. Two use-cases present different aspects of the AM-process in the future and one use-case describe the future usage-process. This section is based on AM-PROCESS TODAY and continuous discussions with Babak Sadighi Firozabadi at SICS.

In DISCUSSION AND CONCLUSIONS differences between the use-cases of TODAY and IN THE FUTURE are discussed and conclusions with respect to the problem statement are presented.

In APPENDIX 1 the PKI-solution used by the Organisation, Keon, is described in more detail.

2. THEORY

The aim of this chapter is to present the theoretical foundation for understanding the functioning of AMANDA that is introduced in section 2.5. Section 2.1 presents and discusses the concept of trust. Section 2.2 explains the functioning of Public Key Infrastructures and describes central components. Section 2.3 presents some commonly used models and mechanism for Authorisation Management. The section points out and discusses those limitations in these models and mechanisms that AMANDA intends to solve. Section 2.4 introduces Trust Management, the ideas that AMANDA is inspired by. Section 2.3, 2.4 and 2.5 each ends with an analogy that in non-technical terms describe how authorisations would be managed and controlled with traditional mechanisms, Trust Management and AMANDA respectively.

2.1. TRUST

Trust is, in some sense, always considered within PKIs and TM but it is not always clear what is meant by the term. It seems the term can have different meanings and this may be confusing. This section presents the concept out of some different perspectives and concludes by summing up how trust, in this thesis, relate to computerised systems.

2.1.1. HUMAN TRUST

Understanding and being able to explain the subtle concept of trust as it is established and exists within and between humans is a difficult and complex task. Out of a general psychological, communicational, informational and social perspectives trust creation may be described as consisting of three sub-processes: *Information Acquisition* - resulting in an environmental scanning for relevant information. *Argument Formation* – the aim of this sub-process is to establish a personal argument formation. With personal arguments as input the individual makes a *Trustworthiness Decision* and a certain degree of trust is established in whatever the subject of the trust-creating process was.[3] In its most general form, "…trust can be described as having some sort of *belief* about the outcome or the continuation of a situation, process, or social construction that in a sense involves other human beings".[4] That is, if a situation, process or social construction is recognised by a human, so that an outcome can be predicted with some level of certainty, some degree of trust exists.

2.1.2. TRUST IN ORGANISATIONS

The BATE-model dissects trust in four parts and has been developed to analyse trust in the specific setting of e-commerce. The four parts can be said to represent the information acquisition and argument formation sub-processes described above. *Business Trust* is achieved through e.g. a certain brand name. One aspect of *Administrative Trust* is achieved through knowing one's own and others legal rights and responsibilities. *Technical Trust* is based on the level of confidentiality, integrity and availability that can be achieved.

[3] Sjölin (2000)
[4] Ibid

Experience-based Trust, finally, is built over time by acting consequently and professionally. On a more abstract and general level, *Trust* is defined as the confidence a subject puts in circumstances beyond the subject's immediate control. *A trusted party* then becomes a subject whose knowledge can be verified by a trusting party. *Knowledge* is achieved through verifying circumstances within the immediate control of a subject and a *trusting party*, finally, is a subject who trusts the knowledge of another subject. [5]

Pekka Nikander and Lasse Metso state that all human operations involve trust and that the foundation for trust is some degree of control through social and legal systems. If these systems are compromised, then so is our basic security. Trust in this context, in the real world, is to a large extent implicit and this differs from a computerised system where trust relationships have to be represented explicitly. One way to transfer some of the subtle real-world dimensions of trust are through the use of a certificate-based architecture. Here "…*trust* in a service is a *belief* that the service, when asked to perform an action, will act according to a predefined description". This implies a belief that the service, or operator behind the service, will not attempt to harm the requester, no matter how the action is performed. In the computerised system, trust is always explicitly expressed in relation to a service-provider and to the assumed performance of an action.[6]

2.1.3. TRUST AND AUTHORISATION

Authorisation implies trust in one way or another. That is, if authorisation has been delegated to any other entity than the root, assumed to trust itself, then some degree of trust exists between the entity delegating authorisation and the entity receiving it. And also, often some degree of responsibility and obligation follow with the authorisation.

Routines, policies and rules may to some degree replace human trust. For instance, the decision to authorise an employee may be a matter of following a formal, bureaucratic[7], routine. The responsible decision-maker might not know each individual employee well enough to establish a personal trust in him. However, with the employee in a specific role and with policies governing both the decision-maker and the employee the decision-maker might, on behalf of his organisation, be obliged to authorise the employee to perform certain actions.

If trust already exists between two humans the weak link, when interacting across a computerised network, becomes the technical solution of the system itself. If these two humans decide to trust a third party, guaranteeing that the electronic system will not affect the trusting interaction between the two humans, they can continue to interact trustingly.

These "trust-enabling" routines, rules, policies and technical arrangements must in turn rest on the legal and social infrastructure offered by society. A PKI[8] is claimed to enable humans to interact trustingly over computerised systems and networks. As most computerised systems, they offer some degree of preventative control-mechanism. Not only does the preventative control-mechanism lack the ability to predict and express trust, it also lacks the ability to prevent every possible violation of real world policies and rules. A detective control-

[5] SIG Security (1999)
[6] Nikander & Metso (2000)
[7] The word is used with the meaning "pre-defined", "step-by-step".
[8] PKIs are described below in section 2.2.

mechanism may further deter violations by, should the need arise, enabling responsibility to be claimed after a violation has taken place.[9]

Being subjective, human trust is a grey zone where trust ethically may be jeopardised or violated without conflicting with, or breaking laws or policies. Therefore, the representation of trust in a computerised system can only be a matter of representing what can be explicitly stated based on a judicial or social foundation legitimising some sort of retaliation. It seems that interacting trustingly within computerised systems is a question of what level of control is offered.

2.2. PUBLIC KEY INFRASTRUCTURE (PKI)

In this section concepts that are fundamental to understanding the ideas and the functioning of PKIs are briefly introduced. PKIs are not important themselves in this thesis but knowing their purpose and functionality will ease the understanding of the case study where they are assumed.

The components that together enable the creation and management of public and private keys and the associated digital certificates make up a PKI.[10] These components may be defined as: "A set of agreed-upon standards, certification authorities, structures between multiple certification authorities, methods to discover and validate certification paths, operational protocols, management protocols, interoperable tools and supporting legislation".[11] It could also be described as "A Certification Authority, a Registration Authority, Directories and a system for managing certificates".[12]

2.2.1 THE CRYPTOGRAPHIC FUNCTION

When symmetric keys are used in cryptography, the same key is used both to encrypt and decrypt a message. For every two persons wishing to communicate securely a secret key has to be distributed and shared. This means that as many keys are needed, as there are persons to communicate securely with.[13] Key distribution and management becomes a problem, rapidly increasing with the number of users; for every person someone wishes to communicate with a secret key has to be distributed and shared. If Lena wishes to communicate with 100 persons, as many keys need to be securely distributed and Lena needs to keep 100 different keys.

Public key, also called asymmetric, cryptography is characterised by the use of different keys for encrypting than for decrypting. Every user has two keys, one public and one private. The private, also called secret, key is only accessible to the user it belongs to. The public key on the other hand is accessible for anyone wishing to communicate secretly with someone in possession of an asymmetric key-pair. If Ebba wishes to send an encrypted message to Oscar, Ebba could obtain Oscar's public key from a public electronic repository on the Internet. She then encrypts her message using Oscar's public key and sends the message to Oscar who

[9] Sadighi Firozabadi & Sergot (1999). Preventative and detective control-mechanism are described in section 2.3.
[10] RSA Keon (2000b)
[11] Feghhi, Feghhi & Williams (1999)
[12] www.whatis.techtarget.com, June 11, 2001
[13] SIG Security (1998)

decrypts the message using his own secret key. A message encrypted using a public key can only be decrypted using the corresponding private key and vice versa.[14] Commonly available electronic repositories on the Internet are so called LDAP (Lightweight Directory Access Protocol)-servers and X.500-catalogues. According to the X.509-standard an entity is not limited to being a person but may be practically anything such as e.g. an organisation, an account or a site.[15]

2.2.2. PUBLIC-KEY CERTIFICATES

Public-key certificates, commonly referred to as "certificates", bind a public-key to a set of information, or attributes that are unique to and identifies the entity.[16] This binding is asserted by having a trusted Certification Authority sign each certificate.[17] In the X.509-standard some of the attributes, referred to as fields are; a unique serial number, the period of validity and information about the public key bound to the owner of the certificate. Several versions of the X.509-certificate exist and which version is used is also stated in the certificate.[18] X.509 has been extended by adding provision for additional extension fields. These extension field types may be specified in standards or may be defined and registered by any community or organisation.[19]

Certificates may be revoked before expiring for a number of reasons. A private key associated with a certificate may have been compromised, which might occur if, for instance, a smart card is lost. Perhaps the owner of the certificate does not act the way he is obliged to according to agreements with the Registration or Certification Authority. The certificate-issuing authority handles this by signing a "revocation-certificate" that is published in a Certificate Revocation List (CRL). CRLs are similar to repositories but differ in that they, instead of valid certificates, contain lists of revoked certificates. They are public and easily accessible through the use of for instance X.500-repositories or LDAP-servers.[20] Before Ebba uses Oscar's public key she, or rather her software, will look in a CRL to make sure that Oscars' public key certificate has not been revoked.

2.2.3. CERTIFICATION AND REGISTRATION AUTHORITIES

Organisations or vendors called Certification Authorities (CA) issue certificates. A Certification Authority play the role of what is commonly called a *trusted third party*. Briefly, this means that several entities have decided to trust a third common entity as the issuer of certificates. Several factors affect the degree of trust that users of certificates will assign the bindings embodied in the certificate. Such factors are, for instance, the operating policies of the CA, the obligations of the certificate-holder in terms of, for instance, protecting the private key and the warranties and limitations of liability stated by the CA.[21]A Certificate Policy describes what obligations a CA has accepted to fulfil and a Certification Practice Statement describes how these obligations are to be fulfilled.[22]

---

[14] SIG Security (1998)
[15] Chokhani & Ford (1999)
[16] Ibid
[17] Housley, Ford & Polk (1999)
[18] SIG Security (1999)
[19] Housley, Ford & Polk (1999)
[20] SIG Security (1999)
[21] Chokhani & Ford (1999)
[22] SIG Security (1999)

One function closely bound to, but still distinctly separated from, that of CAs, is provided by Registration Authorities (RAs). A RA accepts and registers requests for certificates from entities. Information about the possible future user is collected and the identity of the user is established by some means. A separate RA may handle this process or it may be incorporated with the CA.[23]

2.2.4. PKI STANDARDS

In order for interoperation between different PKIs to be possible and in order to enable multiple applications to interface with a single, consolidated PKI, standards are needed. Three groups of PKI-standards exist as showed in figure 2. The general PKI-standards, X.509, PKIX and PKCS, specifically define the PKI. User-level standards, e.g. the Secure Socket Layer (SSL) and S/MIME, depend on these general standards and applications, such as Virtual Private Networks (VPNs) and e-mail, depend on the user-level standards in that they may require, assume or allow the use of a PKI.[24] For instance: A VPN is based on IPsec and in order to perform secure web-interactions such as online shopping or banking, SSL/TLS is needed. However, both IPsec and SSL/TLS may use any of the general PKI-standards.[25]

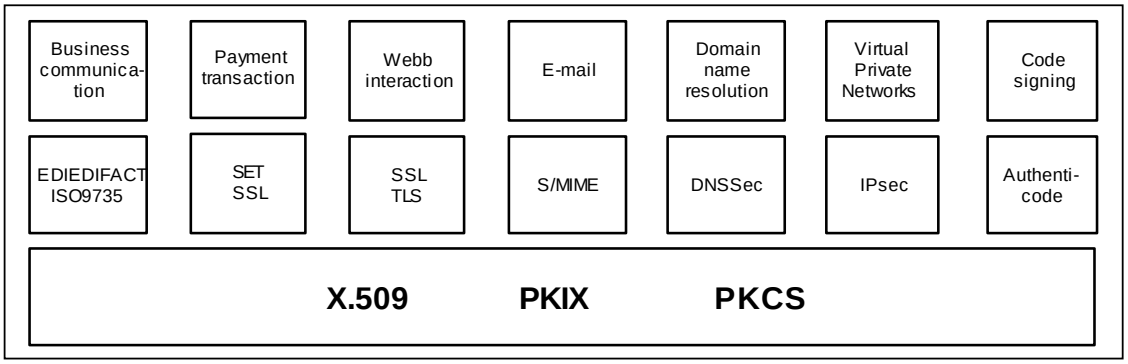

Figure 2: PKI standards.[26]

2.2.5. USING CRYPTOGRAPHY FOR SECURE AND SAFE COMMUNICATION

A typical way for a sender to encrypt and digitally sign a message and for a receiver to decrypt that same message and verify the signature of the sender is described below. Note that both symmetric and asymmetric cryptography is used:

When Ebba has written a message and wishes to encrypt it, her software will generate a symmetric key for this particular occasion. This key is used to encrypt the message. Ebba then takes Oscar's public key that is publicly available in a directory and uses this public key to encrypt the secret key. Next, the unencrypted text is hashed using some algorithm and this hash is encrypted using Ebba's private key. This makes up the digital signature of Ebba. The message may now be sent and consists of three parts: the encrypted message, the encrypted

[23] RSA Keon (2000a)
[24] Ibid
[25] For a more thorough description of this, e.g. SIG Security (1999) or RSA Keon (2000a)
[26] SIG Security (1999)

symmetric key and Ebba's digital signature. Symmetric key encryption is considered to be harder to crack and the encrypted format requires less space than if public key encryption is used. Public key encryption has the advantage that the public keys can be distributed freely enabling anyone to retrieve and use them, thereby eliminating the problem of key-distribution. This is the reason symmetric and public key cryptography is used jointly.

After receiving the message, Oscar, using his software, opens the message and separates the three components. The received encrypted symmetric key is decrypted using his private key. This enables the decryption of the message. Ebba's signature is decrypted using her public key. Hereby the hash is retrieved. Oscar takes the hash-algorithm that may have been sent unencrypted and hashes the recently decrypted text. If the two message-hashes compares correctly, the signature of Ebba has been verified.[27]

2.2.6. VERIFYING THE VALIDITY OF A CERTIFICATE

Oscar retrieves the public key of Ebba from her certificate, found in a repository. If Ebba's public key-certificate has been signed by a trusted third party that Oscar, or rather his own CA, has decided to trust, then the signature of Ebba is considered valid. The public key of the CA that signed Ebba's certificate, using it's private key, has to be retrieved. It might be retrieved from a repository or, as commonly is the case, it is already kept on the browser used.

It is possible for one superior CA to delegate the right to issue certificates to other subordinate CAs. The CAs are then arranged in a hierarchical structure with several CAs below a root-CA. At the leaf-nodes the end users' certificates are found.

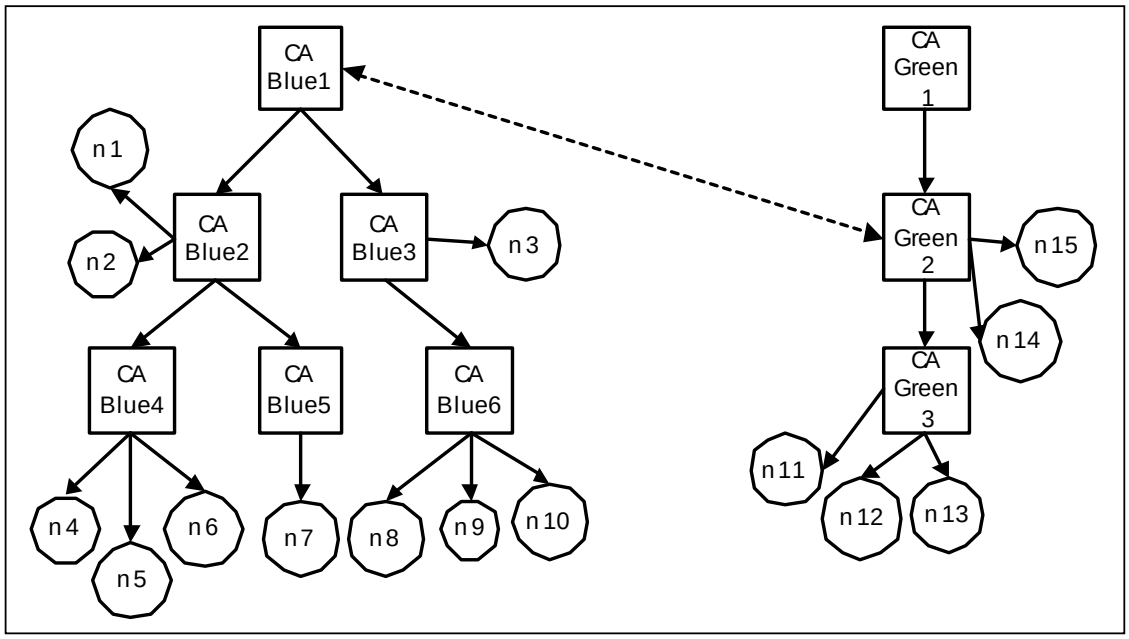

Figure 3: CA-hierarchy and cross-certification.

[27] RSA Keon (2001)

If, in figure 3, node 2 (n2) wants to validate the signature of n10, n2 retrieves the public key certificate of n10. N2 finds that the public key of n10 is signed by CA Blue 6. If n2 does not have the public key of CA Blue 6 listed as a trusted third party then n2 will retrieve the public key of the CA that signed the certificate of CA Blue 6, in this case CA Blue 3. If CA Blue 3 also is unknown n2 will retrieve the public key of the CA that signed the certificate of CA Blue 3, namely that of CA Blue 1. CA Blue 1 is listed as root in the browser of n2 and is therefore trusted. N2 has through *certificate path discovery* identified a path between two arbitrary nodes. Verifying the binding between a node and the corresponding public key based on a certification path is called *certification path validation*.[28]

Each hierarchy in figure 3, e.g. representing different vendors, may be considered separate *islands of trust.* In the blue hierarchy only public key certificates signed by blue CAs are trusted. In the green hierarchy only public key certificates signed by green CAs are trusted. If the root of each hierarchy cross-certifies the other then green will accept public key certificates signed by blue as trusted and vice versa. This is indicated by the dotted line in figure 3. Cross-certification supports large-scale deployments of public key applications such as secure e-mail.[29]

2.3. AUTHORISATION MANAGEMENT

This chapter is introduced by defining some of the terminology central to this thesis. A typical AM-process of today is presented, as is a typical process of using an authorisation. In both cases a PKI is implied. Some traditional models and mechanisms commonly used for AM are presented. Drawbacks with these models and mechanisms as well as the characteristics of a solution to these drawbacks are discussed. The chapter is concluded with an analogy describing how authorisations to enter and use attractions at an amusement park would be administrated and controlled using traditional models and mechanisms.

2.3.1. THE SEMANTICS OF AUTHORISATION

Associated with authorisation are terms such as privilege, right, delegation, obligation and responsibility. The semantics of these and other words differ with context and individuals using them and this may be confusing. In order to manage authorisations using a generic mechanism, as is the case in Trust Management and AMANDA, it is necessary to identify a common terminology to be used in a stringent way when formalising for instance a policy. This terminology is also, of course, useful when describing AM without considering Trust Management or AMANDA in particular.

Privileges possible to manage by a system may be either *object-level permissions* or *management-level powers*, also referred to as *permissions* or *powers* respectively. A permission allows the usage of a resource such as reading or altering a file or executing a program. A power implies the right to declare or delegate permissions or powers.[30]

[28] Feghhi, Feghhi & Williams (1999)
[29] Ibid
[30] Sadighi Firozabadi, Sergot, Bandmann (2001)

Preventative and *detective* control mechanisms aim to ensure that Agents[31] behave according to policies. A preventative mechanism aims at preventing violations of policies from occurring. A detective mechanism will not prevent violations, but will with some likelihood ensure that violations are detected within a reasonable time-period.[32]

In this thesis, *privilege* and *authorisation* are used synonymously, and may imply both power and permission. The meaning of one or the other is implied by the context or by additional details in the text that explicitly state whether a power or a permission is meant. What is permitted may be restricted by policies or policy-based decisions and these restrictions may be enforced through preventative and/or detective control mechanisms.

By *delegation* it is meant the assignment and distribution of privileges according to policies, hence delegation may be of two kinds.[33] The AMANDA-group has focused on the first kind but they are both of interest with respect to matters such as responsibility, trust, and the revocation or altering of a privilege.

- **Delegation in terms of creating new privileges**: The delegatee, that is the receiver of a delegated privilege, is issued a new privilege that is independent from the delegator's privilege. This means that if the delegator's privilege is revoked this does not automatically mean that the privilege of the delegatee also is revoked. It is also possible for someone to have the power to delegate permissions to others without necessarily having that particular permission himself. He might not even be empowered to issue such permission for himself.[34] For example, in the first case, the head of a unit is empowered to delegate permissions to subordinates. If the head of the unit is replaced, everything else being equal, it is likely that the subordinates are expected to keep their permissions. In the second case, one example would be a head of security with the power to delegate permissions to certain computerised mechanisms. This does not mean he himself needs to have the same permissions. It might even, out of the security-perspective, be undesirable. As such this type of delegation serves as a separation-of-duties security-mechanism.

- **Delegation by proxy**: The delegatee does not receive his own privilege but may exercise a privilege that the delegator already has. The delegatee *speaks for* or *acts on behalf* of the delegator and if the privilege of the delegator is revoked then so is, automatically, that of the delegatee.[35]

In this thesis the term *Authorisation Management (AM)* implies the full process of continuously determining, delegating and altering authorisations in a computerised system as well as implementing them. Which authorisations may be delegated and by whom, as well as how this is to be administrated, is regulated through policies. Once authorisation has been delegated and the decision has been implemented the user may perform actions accordingly. The usage of a delegated authorisation is also regulated by policies guiding and governing the behaviour of the user.

The term *policy* in this thesis includes everything that, directly or indirectly, formally guides or governs the delegation, implementation or usage of authorisations. A policy may be a law,

[31] Here the term refers to entities interacting within a computerised system.
[32] Sadighi Firozabadi & Sergot (1999)
[33] Sadighi Firozabadi, Sergot, Bandmann (2001)
[34] Ibid
[35] Ibid

a guideline, a security policy, a mission statement etc. Policies may have been formulated primarily for other reasons than to guide Authorisation Management. Interpretation of one or several policies may motivate the formulation of new policies or the making of decisions. The decisions of interest in this thesis concern the delegation of authorisations.

2.3.2. AUTHORISATION MANAGEMENT TODAY

The AM-process of an organisation varies with the organisation at hand, its policies, routines, computerised systems etc. One particular and detailed example will later be presented in a use-case describing the AM-process of O today. A common AM-process of many organisations today could however be exemplified as follows:

A global corporate strategy is interpreted into a strategic plan for a national subsidiary. This is in turn interpreted and formulated into a tactical plan for a department. Guided by this, the head of the department initialises a project to analyse customer behaviour. The Head of Department requests that all project-members should be authorised read access in an existing customer database. The request is forwarded to whomever, e.g. the system-owner, is assigned the responsibility of deciding whether or not the request complies with policies. If policies and the facts at hand comply, the authorisations may be implemented.

However, this is not done automatically. Typically, AM is to a large extent a matter of manual routines involving several persons, e.g. a system owner, a system administrator, users and a decision-maker, many times separated geographically. The full AM-process does in fact involve two separate sub-processes: In the *real world* policies are interpreted, requests are made and communicated, decisions are made concerning the requests and these decisions are communicated so that they can be enforced. Often both requests and decisions are signed written documents and communicating the decisions implies transporting these documents geographically. The other process takes place in the *IT-world*. An empowered administrator interacts with a computerised system and alters the authorisation-controlling mechanism of the system.

2.3.3. USAGE TODAY

Being able to use an authorisation is the goal of the AM-process. The functions of the authorisation-controlling mechanism vary with systems, models and mechanisms used. One particular and detailed example will later be presented in a use-case describing the usage of authorisation in O today.

Within a PKI-setting it is however common that the task is treated as a combination of authentication and access control.[36] After logging on to the system and being authenticated the user makes a request to use a specific application. Using some mechanism the system will control if the user is in fact authorised to perform the requested action. This could be done through mapping the public key of the user to a unique name and the name to authorisations in an ACL. The authorisations will be transferred to some control-mechanism controlling access to the application and the user will be allowed access accordingly.

[36] Blaze, Feigenbaum, Ionnidis & Keromytis (1999b)

2.3.4. AM- SUPPORTING MODELS AND MECHANISMS OF TODAY

The Trusted Computer System Evaluation Criteria (TCSEC) was created to meet the major security objective of preventing unauthorised access to information even though access to functions also may be regulated. The term access control underlines the fact that authorisation primarily is concerned with access to information or functions. Two types of access controls are specified: Discretionary Access Control (DAC) and Mandatory Access Control (MAC). DAC permits users to grant or deny other users access to objects under their control and is based on the identity of subjects and/or groups. MAC restricts access to information based on the sensitivity of the information contained in the object and the formal clearance of subjects to access differently labelled information.[37]

Role Based Access Control (RBAC) is often used to ease the administration of authorisations. A role can be thought of as associated with one or a set of actions that a user is authorised to perform within the context of an organisation. Roles are group oriented and with each role a set of permissions are associated.[38] Expressiveness may be increased by use of negative permissions, or restrictions, specifying actions a subject is not allowed to perform. Several users may have the same role and one user may have several roles.[39]

Access Control Matrixes (ACMs) are one commonly used way to express the authorisations subjects have been given concerning objects. Each row in an ACM belongs to a specific subject and each column to a specific object. The authorisation a subject has upon an object is expressed in the intersection of a row and a column. Using ACMs authorisations may be implemented through the use of Capabilities or Access Control Lists (ACL). Using capabilities every subject may be given a capability in the form of a non-forgeable token. This capability corresponds to the subject's tuple in the ACM. Capabilities have not been widely used because of the difficulty to get an overview of all subjects permitted to somehow access an object. Another reason is difficulties, if need arises, to revoke an authorisation. More widely used are ACLs where the ACM is implemented as a table, each table representing an object. When a user requests access, the system performs a check in the ACL to see if the subject is authorised to perform the action requested.[40]

2.3.5. DISCUSSING THE MODELS AND MECHANISMS OF TODAY

Deriving authorisations from real-world policies and expressing them in terms of the models and mechanisms described above is *imprecise*. It is not likely that governing policies have been formulated with a security policy in mind and this lack of expressiveness clearly limits the set of actions possible to delegate and control through a computerised security mechanism. It might be that models and authorisation-controlling mechanisms try to map and express some needs of an organisation. The Graham-Denning model, e.g., propose eight primitive protection rights; to create and delete a subject or object and to read, grant, delete or

[37] Ferraiolo & Kuhn (1992)
[38] Ibid
[39] Gollman (1999)
[40] Ibid

transfer access rights[41], but still; existing security-models demand that authorisations are to be expressed and implemented in terms of the models and authorisation-controlling computerised mechanisms at hand. Using the Graham-Denning model, it would be difficult to within a computerised system express and enable: "…the Chief Financial Officer (CFO) has the permission to take legally binding loans, up to US$ million, on behalf of company MiniVans".

In a computerised system the security mechanisms monitoring and controlling the usage of authorisations are often *insufficient* and therefore complemented by control mechanisms in the real world. For instance: Some employees may be authorised to read and write a file and others only to read. This is a sort of preventative control mechanism enabling only certain employees to write. However, the permission to write might also be coupled to specific circumstances not possible to express through an ACL. The employee might for instance only be authorised to write during certain hours of the day. If no computerised control mechanism exists in order to enforce such a restriction, it may be that the empowering authority tries to deter the employee from violating the policy e.g. by stating that violations will result in disciplinary actions. This is often combined with a manual monitoring process of reading log files and comparing them to the circumstances at hand when the privilege was being used. It will be difficult to achieve sufficient control if many, perhaps tens of thousands, write actions are performed every hour.

Access-rights such as read, write and execute can be controlled by most operating systems. Procedure-Oriented Access Control is a solution added on top of other authorisation-controlling mechanisms. It can be considered to encapsulate the object, permitting only certain specified accesses through a trusted interface. Procedures, e.g. for adding or deleting a user, are not intended to improve expressiveness but rather to increase security. Even though they do increase security they are fairly complex and slow down the system.[42]

Implementation is also *slow*. A decision communicated via postal services might take days before reaching the systems administrator responsible for implementation. Policies and organisational structures change as well as whom is to be authorised to do what. This causes the set of possible actions to change as well as the circumstances under which they may be performed to change. Each change invokes and requires the full process of Authorisation Management.

It would be useful if a computerised mechanism or mechanisms could go beyond simply enabling the expressing and controlling of primitive access-rights such as read, write and execute. Such a mechanism would have to be generic and be precise enough as to allow a close mapping of any formal representation of real-world policies. Increased expressiveness would also bring about more precise and thorough preventative control. Both directly as it would be possible to explicitly and precisely express restrictions, but also indirectly as being able to precisely express what may be done excludes much of what may not be done.

It would be useful if the full process of Authorisation Management could be so decentralised as to map the hierarchy and chain of command of the organisation it supports. That is; so that each individual decision-maker empowered according to real world policies, not only makes the decision but also implements it within the computerised system. Such a decentralised approach would not only dramatically speed up the AM-process but would also cut out all the

[41] Pfleeger (1996)
[42] Ibid

administration that today must be performed between making the decision and implementing it.

2.3.6. TRADITIONAL USAGE AND AM WITHIN A PKI – AN ANALOGY

A gatekeeper uses traditional models and mechanisms to express and control authorisations at an amusement park. If Mark wants to get into the amusement park he would have to present an ID card to the gatekeeper. The ID card has a unique number together with a photo of Mark. The owner of the amusement park has instructed the gatekeeper to trust ID cards by this particular issuer. Next, the gatekeeper controls the validity of the ID card by calling a special telephone-number in order to find out if it has been revoked. If the ID card is OK he will allow Mark access to the amusement park *(Mark has successfully logged on to the system and has been authenticated*). Once in, Mark decides to ride the roller coaster. He walks over to the ticket-boot and presents his ID card to the clerk sitting there. The clerk has two folders. In the first folder unique ID-card numbers are mapped to names. The second folder contains different lists for different attractions and each list in this folder has look-up tables. Each table consists of one row of names and two columns representing "yes" and "no". From the first folder the clerk will find out that the ID-card number belongs to "Mark". Next, from the second folder, he will take out the list for the roller coaster and look up "Mark". Mark is listed and has the "yes"-column checked. The clerk will issue a ticket and give it to Mark *(Mark has now had his authorisation checked in the ACL and has been given log on information to the particular application).* Mark can now walk over and present the ticket to the gatekeeper at the roller coaster who will allow him to ride it. Each time Mark wants to enjoy an attraction he has to start over at the clerk's.

If Ed, who has a valid ID-card and is already listed in the first folder, wants to be added to the roller coaster list that he is not yet on, he sends a signed letter to the owner of the amusement park and requests this authorisation. *(Ed has now followed formal procedures, according to real world policies, for requesting a privilege).* The owner has certain real world policies that govern him when deciding about whom to authorise to do what in his amusement park. If he decides to grant Ed the privilege of riding the roller coaster he will write a decision, sign it and post it to the supervisor of the clerk. *(The decision-maker has now interpreted policies, made a decision and communicated it to the administrator)* After receiving the decision, the supervisor will collect the roller coaster list and add Ed's name in it. Ed has now been granted the privilege to ride the roller coaster. *(The administrator has altered the ACL)*

2.4. TRUST MANAGEMENT

Trust Management (TM) has very little to do with trust, it simply enables the expressing and delegation of authorisations in a distributed environment using certificates. The ideas, components and functioning of TM are described below. How certificates are used to delegate authorisations may be confusing. Therefore a comparison is made between different approaches. The section closes with analogy where TM is used to control authorisations with respect to the amusement park.

2.4.1. TRUST MANAGEMENT COMPONENTS AND FUNCTIONING

TM is defined as "… a unified approach to specifying and interpreting security policies, credentials, and relationships that allows direct authorisation of security-critical actions".[43] A Trust Management system consists of five basic components: a language for describing actions, a mechanism for identifying principals, a language for specifying application security policies, a language for specifying credentials and a compliance checker.[44]

An **action** is an operation with security consequences that are to be controlled by the system.[45] An action is not limited to observing or altering information and does not have to be predefined in an ACL and mapped to a name derived from a public key certificate. An action might be anything that is possible to regulate through the use of a programming language. This programming language is used to express both the policies and credentials that are used.[46] This may very well be a "traditional" reading, writing or executing a file, but it might also be, for instance, the right to write electronic checks, the right to sign a contract to buy a house or the right do delegate some sort of privilege to someone else.[47]

Principals are entities that can be authorised to perform actions and may also be trusted to issue credentials.[48] Entities may be anything able to make requests for actions in a computerised system.[49] A principal may for instance be a printer, a server, a software process or a person in a particular role. The entities that are of interest in this thesis are humans such as users, decision-makers and administrators.

The Trust Management component referred to as the **security policy** governs what action principals may perform within a particular computerised system upon presenting credentials.[50] Security Policy is the term used to describe the computerised representations of a real world policy. The security policy is part of the IT-world and is expressed in some programming language. Ideally, the policy is derived from and permits actions strictly according to the real world policies that it maps and represents in the IT-world. In this thesis the security policy is, in general terms, referred to simply as "the computerised representation of real world policies". When describing AMANDA in this thesis the equivalence of the TM-security policy is somewhat different. It is divided and referred to as the Meta- and the Application-policies.

The security policy may itself directly authorise certain keys to perform specified actions. However, more typically the security policy will delegate this responsibility to credential issuers that it trusts to have the required domain expertise and relationships with potential principals.[51] In terms used in this thesis, a credential issuer is simply an empowered decision-maker. Issuing a certificate implies the decision-maker electronically creates and digitally signs an attribute certificate that states what authorisations are delegated and to whom.

---

[43] Blaze, Feigenbaum, Ionnidis & Keromytis (1999b)
[44] Blaze, Feigenbaum, Ionnidis & Keromytis (1999a)
[45] Ibid
[46] Ibid
[47] Ellison (1999), Ellison, Frantz, Lampson, Rivest, Thomas & Ylonen, T (May 2001)
[48] Blaze, Feigenbaum, Ionnidis & Keromytis (1999a)
[49] Blaze, Feigenbaum, Ionnidis & Keromytis (1999b)
[50] Blaze, Feigenbaum & Lacy (1996). Blaze, Feigenbaum, Ionnidis & Keromytis (1999a)
[51] Blaze, Feigenbaum, Ionnidis & Keromytis (1999b)

Credentials describe specific delegations of trust and allow principals to delegate authorisation to other principals through use of public keys. These public keys *may* bind authorisations to perform specific tasks directly to the keys,[52] thereby eliminating the need to first extract a name and then look up predefined authorisations in e.g. an ACL, as commonly is the case when using public key certificates.[53] Credentials can be considered as the statement that delegates authorisations. The practical way of actually doing this is by issuing some sort of certificate.

Policies and Credentials are together referred to as *assertions* and they are very much the same. An assertion - a policy or a credential - consists of two parts, the source of authority and a program, or piece of a program, describing the nature of the authority being granted as well as to whom. The same language is used for expressing both policies and credentials. What differ between a policy and a credential are the sources. If the source is the creator, or the root, then it is a policy. If the source is the public key of someone delegating authorisation, then it is a credential.[54]

The **compliance checker** uses a general-purpose, application-independent algorithm for checking proofs of compliance.[55] The compliance checker provides a service to applications for determining how an action requested by principals should be handled, given a policy and one or several credentials".[56] The idea is that a principal requesting an action within a particular application forwards this request (r) to the compliance checker. Together with the request the credentials (C) that the principal calls upon to prove his right to perform the requested action are also forwarded. The security policy (P), finally, is also forwarded to the engine. The engine will use (r, C, P) as input and run them through the algorithm before returning the result to the server or whatever device is controlling access to the application. The compliance checker could respond by approving or rejecting the requested action.[57] It is up to the requesting principal to collect (C) and (P) and to present them to the compliance checker.

A large number of applications may be distributed across one or several networks including the Internet. A large set of principals may make a large set of requests for action. The set of entities may change due to e.g. employee turnover or employees being assigned new roles. The set of possible actions may change e.g. due to changes in the real world policies regulating them.[58] Each change that somehow affects authorisations has to be implemented in the system controlling authorisations. The advantage of being able to express security policies locally becomes clear when considering the number of applications that may be distributed over any network[59], the number of actions desirable to express, and the number of principals requesting authorisations.

[52] Blaze, Feigenbaum, Ionnidis & Keromytis (1999a), (1999b)
[53] Ellison (1999)
[54] Blaze (2001), Blaze, Feigenbaum, Ionnidis & Keromytis (1999b)
[55] Blaze, Feigenbaum, Ionnidis & Keromytis (1999a)
[56] Ibid
[57] Blaze, Feigenbaum, Ionnidis & Keromytis (1999b)
[58] Ibid
[59] Blaze, Feigenbaum, Ionnidis & Keromytis (1999a)

2.4.2. CREDENTIALS, PUBLIC KEY CERTIFICATES, ATTRIBUTE CERTIFICATES AND SPKI-CERTIFICATES.

Differences and relations between credentials and different types of certificates are confusing and difficult to get a hold of. In this section an attempt is made to clarify matters.

In order to meet the needs of the Internet community for TM a certificate structure and operating procedures have been developed. These certificates are called *SPKI-certificates (Simple Public Key Certificates)* and the idea is that they can be used as *credentials.* The main purpose of SPKI-certificates is to bind authorisation directly to a public key. The holder of the corresponding private key may, through use of SPKI-certificates, be authorised to perform actions. An SPKI-certificate contains attributes that may be defined solely by the author of the code that makes use of the SPKI-certificate. However, some basic attributes, such as validity period and the binding of keys to different names are recommended.[60]

Besides binding a public key to a unique identity, a *Public-Key Certificate (PKC)* contains a number of other attributes.[61] X.509 v.3 certificates, for example, contain an extension field that may be used to add attributes that are either predefined standard extensions or defined at will.[62] This extension field could be used to contain authorisation information, this is however considered undesirable for two reasons. First, authorisations have a tendency to change often whereas the coupling of an identity to a public-key normally have a much longer lifetime. With authorisations in the extension field of a PKC, the PKC would have to be revoked and a new one issued each time authorisations change. Second, the entity delegating authority is normally different from the entity issuing PKCs.[63]

Attribute Certificates (ACs) are based on the X.509-standard. The binding of authorisation information to an identity is still considered necessary even though the structure of the ACs do not contain a public key. This is the main difference between the structure of an AC and a PKC. The AC contains authorisation information associated with the holder of the AC. Such information may be role-belongings, group memberships or security clearance. The reason for this is that the RBAC schemes, rather than the identity of an individual, are commonly the criterion on which authorisation decisions are made and controlled. Through the use of extension fields, additional attributes may be associated with the holders. These extensions could be used to delegate authorisations.[64]

The main components of an attribute certificate are:[65]

- *Issuer* (a distinguished name, or a public key)
- *Subject* (a distinguished name, or a pointer to the subject's public key certificate)
- *Validity Interval* (states during what time-periods the attribute certificate is valid)
- *Certificate Serial Number* (a unique ID-number assigned by its issuer)
- *Signature* (The digital signature algorithm used to sign the certificate)
- *Attribute* (The set of attributes associated with the subject).

[60] Ellison (1999)
[61] Chokhani & Ford (1999)
[62] Housley, Ford & Polk (1999)
[63] Farrell & Housley (2001)
[64] Ibid
[65] Sadighi Firozabadi, Sergot, Bandmann (2001)

In summation: *Credentials* are a TM-term that expresses delegation of authorisations although the term may be used in other contexts and have other meanings. *SPKI-certificates* are used to practically express and transport credentials. An authorisation may be bound to the SPKI-certificate without being bound to an identity. SPKI-certificates have been specifically developed to meet the needs of TM.

PKC contain a public key and are a fundamental component in a PKI. A unique identity is associated with each PKC. Typically the lifetime of a PKC is long. *ACs* can contain certain pre-defined authorisation information but may also be used to express and transport delegation of authorisations as defined by a community, e.g. an organisation. The basic structure of an AC does not contain a public key even though an AC must be possible to bind to a PKC.

2.4.3. AUTHORISATION MANAGEMENT THROUGH TRUST MANAGEMENT - AN ANALOGY

TM is used to express and control authorisations in regard to different attractions in an amusement park. No gatekeeper controls access to the amusement park. Instead gatekeepers control access locally. These gatekeepers are fairly stupid and only do what they are told by Mr BigBrain, their boss *(the compliance-checker)*. Mark walks up to the roller coaster gatekeeper and presents a ticket *(a credential)* to him. The ticket states that whoever holds it should be allowed access to ride the roller coaster. The ticket is valid for three rides. Mary has signed the ticket. Mark also presents the gatekeeper with some papers *(the security policy)* that describe in what ways it is possible to use the roller coaster as well as who else may allow others to use it. The gatekeeper takes the ticket and the papers to Mr BigBrain. Mr BigBrain sits down and reads the ticket and the policy. From the ticket he finds out that someone wants to ride the roller coaster and that Mary says it's OK for three rides. From the policy he will find out that one thing possible to do with the roller coaster is to ride it. He will find out that it can be ridden for any number of times and he will also find out that Mary may issue tickets to ride the roller coaster any number of times to anyone she pleases. The contents of the ticket comply with what is stated in the policy. Mr BigBrain first checks the ticket so that only two rides remain. Then Mr BigBrain hands the ticket back to the gatekeeper and tells him to allow the holder of the ticket to ride the roller coaster. The gatekeeper will hand the ticket back to Mark and will also allow him to ride the roller coaster once. When Mark wants to ride the roller coaster again he once again has to present the ticket and the policy to the gatekeeper who will hand it to Mr BigBrain etc.

If Ed wants to ride the roller coaster but does not hold a valid ticket he has to persuade someone, authorised according to the policy, to issue him a ticket. Once he has managed to do this he walks over to the gatekeeper and is taken through the same process as Mark was.

2.5. AMANDA

In this section the aim of the AMANDA-project is presented. A set of components that together make up AMANDA is also presented in more detail. These components are under development. It is therefore important to stress that the exact construction and contents of these components may in the future differ. The purpose is to describe the proposed functioning of AMANDA. The analogy is presented again but this time AMANDA is used to manage and control authorisations.

2.5.1. AUTHORISATION MANAGEMENT FOR ORGANISATIONS

The AMANDA-project aims at developing a language for representing dynamic policies and mechanisms to be used by the engine making the authorisation decisions. Such a language and its mechanisms have to be flexible and adaptive to allow changes in authorisations to be made and to allow for them to occur immediately. It further has to be expressive enough to enable the expressing and implementation of authorisations according to real-world policies and organisational structure. It must be possible to make these changes in a decentralised setting, according to the normal organisational hierarchy and chain of command and without having to rely on certain administrators to enforce the decisions.[66] If the empowered decision-maker also enforces his decisions, then the manual processes of transporting a decision to an administrator for implementation can be eliminated. Not only does this reduce the time-span between the decision and its implementation, security is also improved as a reduced number of individuals involved in the process automatically reduces the number of errors, whether deliberate or not, that are possible to occur.

Access Control Lists have the advantage of being simple and fast when retrieving authorisations for previously known users requesting a predefined action. Everything else being equal, to use the TM-approach would in fact be slow. Every time a request is made it would have to be combined with policy-and credential-assertions. These would have to be run through the engine and the legitimate power of the issuer would have to be verified, possibly through chains of keys. This would have to be performed every time a request for action is made.

It seems reasonable to combine a decentralised and expressive approach towards formulating and managing authorisations with a “look-up” function for fast and easy access. An organisation would benefit from using a mechanism for AM that allows the authorisation-controlling real world policies to be more precisely mapped into the authorisation-controlling mechanisms of a computerised system. It would also benefit if the AM-process could be decentralised as to closely map the ordinary chain of command. In an organisation both the users of powers and permissions will be known in advance and neither will change very often or dramatically. Therefore it would be beneficial if the large bulk of requests to delegate power and to use permissions could be handled fast and easy by a look-up function as those offered by ACLs. The AMANDA-project considers this.

[66] Sadighi Firozabadi , Sergot, Bandmann (2001)

In figure 4[67] an approach within the AMANDA-project is visualised. A prerequisite is the ability to trustingly authenticate any entity somehow able to interact with other entities. Therefore a PKI is implied.

2.5.2. COMPONENTS – UNDRESSING AMANDA

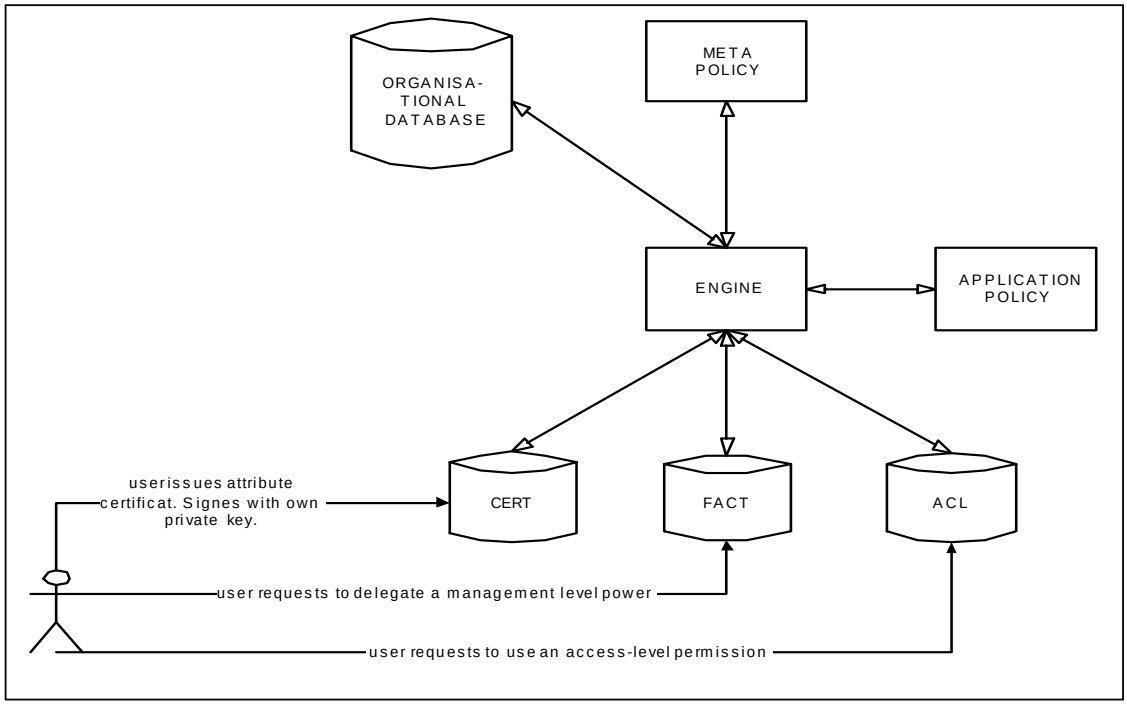

Figure 4: The AMANDA-components.

The **META-POLICY (MP)** is the IT-representation of real-world policies that are general to a somehow delimited organisation. The META-POLICY could be thought of as an object-oriented database consisting of nodes that represent the organisational structure. Each node would represent an entity that the organisation owns or controls. A node could represent a role or a function within the organisation and with each node attributes would be associated. Attributes could be powers and/or permissions in relation to other nodes. The powers or permissions would consist of assertions where the ultimate source of authority would be either the META-POLICY itself or someone authorised by the META-POLICY, to sign the attribute-certificate at hand. Of course other nodes of interest to the particular organisation would have to be included, for instance different applications or processes that another node controls or has some other relationship with. Any node, such as “systems-owner”, “CEO”, “administrator”, “economic-transaction monitoring system” or “customers database” would be defined in terms of attributes and relationships with other nodes.

The **APPLICATION POLICY (AP)** functions and serves the same purpose as the meta-policy but differs in that it represents application-specific policies. In it will be stated, in terms of assertions, what actions are possible to perform in the particular application. Further, restrictions and conditions could be stated. An application could be, for instance, a customer’s database and its policy a set of assertions expressing what actions may be performed upon it

[67] Inspired by Sadighi Firozabadi, Sergot, Bandmann (2001) and discussions with Babak Sadighi Firozabadi.

as well as how powers and permissions may be delegated. Definitions of roles or functions and their interrelationships is not necessary here if this has already been done in the meta-policy. The ultimate source of an application policy ought to be defined in the meta-policy.

POLICY, both META and APPLICATION, may also be referred to as a Privilege Management Infrastructure (PMI) and any empowered node as an Attribute Authoritiy (AA).[68] The PMI needs to be dynamic and possible to determine and change locally. Anyone empowered to delegate authorisations, that is, to issue attribute certificates, is considered an AA.

CERT contains all attribute certificates issued for a particular application, including those that are by some reason invalid. As such it functions as a sort of event history database.[69] Anyone empowered by the meta- or application-policy to issue *any* attribute certificates for the particular application, that is, to function as an attribute authority, may issue attribute certificates. All issued attribute certificates are stored in CERT regardless of their compliance with the policies. Each time a new attribute certificate is fed into CERT, a copy is forwarded to the engine that, using algorithms, will reason about its compliance with the policies. If the statement in the attribute certificate complies with the policies it might be complemented with facts from the organisational database. Depending on the type of attribute certificate, one or more of FACT, ACL or ODB will be updated.

The **ORGANISATIONAL DATABASE (ODB)** is a relational database, containing the current instances of nodes and attributes defined in the policies. For instance, if in the META-policy is stated that the organisation has relationships with other organisations, the current sets of trusted public keys of cross-certified CAs are found here. Further, all the public keys of employees, and other nodes defined in the MP, are linked to attributes. Employees, for instance, may be linked to their general roles or functions defined in the meta- or application policy, etc. How and by whom the contents in the organisational database may be changed is of course also delegated through certificates, just like any other authorisation.

The **FACT** database contains all presently valid powers, permissions and restrictions for an application. Everything in FACT is derived from CERT together with the Meta- and Application-policy and the ODB. The engine will take each new certificate and reason about it using the meta- and/or application policy and possibly also the ODB before approving or rejecting what is expressed through the attribute certificate. If approved the statement will, if necessary, be complemented with presently valid facts from the organisational database before being stored in FACT. A new attribute certificate might create a new fact. As soon as a fact no longer is valid it is deleted from FACT. This may be due to a new attribute certificate revoking a power or permission or to its normal expiring.

A continuously updated Access Control List is kept in the **ACL**-database. The ACL is a simple “look-up”-table where the public keys of particular users are mapped to their permissions in the particular application. Everything in ACL can be derived from FACT. However, the ACL only expresses object-level permissions. If an entry in FACT states “Sophia empowers all group-members” then each instance of this statement, with regard to the individual group-members, will be spelled out. For instance, “Ebba is permitted”, “Oscar is permitted”, “Christian is permitted” etc. The engine keeps the ACL updated using CERT and/or FACT and one or several of the other components.

[68] PMI and AA are discussed in Bandmann, Sadighi Firozabadi & Sergot (2001)
[69] Sadighi Firozabadi, Sergot, Bandmann (2001)

As described above, the **ENGINE**, using general-purpose algorithms, reasons about the validity of issued attribute certificates by comparing them to the policies. This functionality is similar to that of the TM-compliance checker. However, instead of producing an authorisation decision itself it extracts information about powers and permissions and puts it in ACL and/or FACT.

Someone wishing to delegate a privilege according to policies has to be allowed to perform this particular action according to FACT. If the action is not supported by FACT, a new attribute certificate will have to be issued, the engine will have to reason about its validity using the policies and possibly complementing the attribute certificate with facts from the organisational database before copying it into FACT. Someone wishing to use an access-level permission addresses such a request, through use of some software, to the ACL.

2.5.3. AUTHORISATION MANAGEMENT USING AMANDA - AN ANALOGY

AMANDA is used to express and control authorisations with regard to different attractions in an amusement park. Before being able to do anything within the system it is necessary to log on. Being authenticated as well as requesting and being permitted to use an object-level permission functions as described in section 2.3.6 with two differences. First, only one list is kept. In this list ID card numbers are directly permitted or not permitted to ride the roller coaster. There is no need to first look up a name. Second, the owner of the amusement park found a need to regulate other uses of the roller coaster than just allowing or not allowing someone to ride it. He found it a good idea to be able to "allow certain certified repairmen to do anything needed when the roller coaster breaks down" and therefore he expressed this in the AP. A column stating this has therefore been added in the ACL and the ID numbers of presently employed and certified repairmen are amongst the subjects in the ACL. If a repairman quits, Mr BigBrain *(the engine)* will be notified and will remove the repairman's permission from the ACL.

Now someone with an ID-card with number 56 decides to delegate the power to delegate the permission to ride the roller coaster to Ed. Number 56 does this by writing and signing a paper saying: "Ed may issue tickets to ride the roller coaster to anyone he pleases". This paper *(an AC)* is sent to Mr BigBrain. Mr BigBrain will read the paper and he will then consult the amusement park policy *(the MP)*. From the policy he will find out that only the owner of the amusement park may delegate the power to delegate permission to ride the roller coaster, and this may only be delegated to employees if they are assigned the role gatekeeper. Next Mr BigBrain contacts the staff-office *(the ODB)*. He will there find out that the owner of the amusement park has an ID card with number 56 and further that Ed is assigned the role of gatekeeper. Mr BigBrain will conclude that the delegation complies with MP and that it therefore is valid. Mr BigBrain will update another list *(FACT)* where all presently valid rights to delegate powers and permissions are listed. In it he will write "Ed may issue tickets to anyone he wants". Ed will issue quite a few tickets each day and they will all be sent to Mr BigBrain. All Mr BigBrain has to do before approving is to look in the FACT-list.

3. METHODOLOGY AND RESEARCH DESIGN

In this section the work with this thesis is described. The systematic approach towards understanding an enterprise, enterprise modelling, is introduced. Ideas from enterprise modelling have been important when identifying, analysing and modelling the AM-process. Further, the Unified Modelling Language (UML) use-cases are described. The use-case notation is the basic notation used to present the results of the case study, both graphically and verbally. Today, using privileges are clearly separated from the AM-process. In the future usage and the AM-process are not as clearly separated. Therefore the usage-process is also considered and described.

I started discussing this thesis with SICS in December of 2000. They were interested in finding out how well their research could improve the Authorisation Management of a large organisation and therefore wanted a description of a real company. Case studies are considered suitable to describe for instance an organisation[70] and this was the approach chosen. Of interest were not only policies governing the delegation and implementation of authorisations. As large an interest was the identification and description of the actual and often manual processes resulting in privileges being implemented.

The organisation was contacted in December of 2000, they found the project interesting and after agreeing to co-operate, a focal application was chosen. The foundational theoretical studies[71] and the majority of the CASE STUDY that describe the situation of today were carried out during March, April and June of 2001. September and October have consisted mainly of modelling and describing the possible future solution.

I have not found any methodology to formally describe and model both policies governing authorisations and the manual and computerised processes of implementing a decision based on such policies. And further, how (and if) such a decision maps to the authorisation-controlling mechanisms of a computerised system. Therefore, in order to be able to identify, understand and describe the full process of AM, the methodology used to perform and present the case studies consists of textual descriptions combined with ideas and notation from enterprise and software (UML) modelling. The aim has been to develop some sort of framework enabling the description of the full process of Authorisation Management in order to be useful for this thesis. Hopefully the interrelation between interpreting policies, making decisions, communicating them through some manual administrative process and finally implementing them within the technology of a computerised system is described in an intuitive way.

The case study itself is opened with a general description of O, its organisation, mission, the technical platform and the Application. This is done to introduce necessary facts and to give a more complete picture of O and the setting in which the AM-process later described takes place. It will also make reading this thesis easier and more fun.

[70] Eisenhardt (1989)
[71] Concerning e.g. PKI, certificates, Trust Management and AMANDA

3.1. CONFIDENTIALITY

The organisation has requested that neither their identity, nor that of the focal application is revealed. Therefore every description that could reveal the true identity of the organisation or the application has been altered. The organisation, the application, the policies as well as the AM- and usage-processes of today have initially been studied and described in real terms. This description has then been altered. The description of the future AM- and usage-processes has fully been performed in these, to the reader, anonymous terms. The focal organisation is referred to as "the Organisation" or "O". The focal application is referred to as "the Application".

3.2. ENTERPRISE MODELLING

Enterprise modelling is used to describe and analyse different aspects of an organisation. Approaches towards enterprise modelling such as TRIAD[72] and DERCT[73] combine a graphical and textual description in order to ease understanding and are at the same time precise.[74] The ideas that I mainly have found helpful in making sense when identifying and describing the AM-process are those of viewing an organisation out of four different *perspectives*, deciding on what to *focus* and noticing that there is a difference between *worlds*.

3.2.1. DIFFERENT WORLDS

What is described in a model varies depending on what world is of interest. The *real world* describes the world as we see and understand what goes on around us. The *information systems world* is a conceptual description of the interrelations and functionality of entities in the information system. Some of the objects represented in the real world may also be represented in the IT-world, on one level of abstraction or another, whereas others only exist in one of the two worlds. For instance, an ACM can be considered to exist in both worlds. It should be remembered that the IT-world is part of the real world.[75] Throughout this thesis I make a distinct separation between the real world and the information systems world, in the thesis referred to as *the IT-world*.

A third world worth mentioning is the *implementation world*. This world describes *how* the functionality modelled in the information systems world is to be implemented. Issues such as system design, algorithms and configuration are described here[76]. These issues are not addressed in this thesis and the implementation world, such as described above, is not considered.

---

[72] A three year joint project with Ericsson, Telia and other companies with the aim of developing a common modelling language to analyse and describe enterprises in terms of general conceptual models.
[73] Declarative Entity-Relationship with Causation and Time - a language for Enterprise and Information System Modelling-
[74] Wohed (1997)
[75] Ibid
[76] In DERCT the term used is the *Subject World*.

Implementations are, however, very much of interest but of a different kind. The implementation of interest for this thesis concern implementing authorisations, decided about in the real world but enforced through some computerised mechanism into the IT-world. This implementation process consists of both manual processes in the real world and computerised mechanisms in the IT-world. This is of fundamental importance to this thesis as the difference between the actual process today and a possible future process is what the case studies cover and stress. The purpose of the AM-process is to implement an AM-decision. Today, before this can be done, a complex manual process has to be gone through. In the future, each decision-maker enforces his own authorisation-decision.

3.2.2. DIFFERENT PERSPECTIVES

Each world may be viewed in four different perspectives: *processes*[77], *concepts*[78], *intentions*[79] and *rules*.

Processes show how something is performed. This perspective is the only, which presents a dynamic picture of the world that is modelled.[80] In this thesis two processes are of main interest to describe, those of Authorisation Management and of Usage.

Concepts show what the enterprise concerns and their relationships.[81] Some concepts of interest in the real world in this thesis are users, administrators, policies and the computerised system. In the IT-world it is Servers, Desktops, Agents and eventually AMANDA.

Intentions express goals and aims more or less vaguely. Their purpose is to guide and govern the individuals engaged in something. An organisation will, for instance, have an explicit mission that serves the purpose of guiding its employees towards a common goal. In this thesis the term *policy* is used instead. Policies are written, governing and guiding documents in the real world. Directly or through derivation and interpretation they express *who* is to be authorised to do *what* and *how* this is to be implemented, administrated, delegated and restricted. Today the IT-world representations of these real world policies are restricted by what is possible to express using the authorisation-controlling mechanisms of the computerised system. In the future the idea is that the authorisation-controlling mechanisms of AMANDA will allow an exact representation of (any) real world policy.

The purpose of *rules* is to express precisely how something is to be performed. They are intentions but broken down into smaller and more exact pieces considering for instance conditions, prerequisites and restrictions. Rules should be possible to express logically and to test, possibly considering conditions, and when tested they are to result in either “true” or “false”.

Each time an authorisation-decision is made it could be considered an instance of trying whether or not a policy complies with the current set of circumstances. This in turn could be considered a rule. A rule may have the form[82]:

---

[77] In TRIAD the term used is *Acting* and in DERCT it is *Behaviour*.
[78] In TRIAD the term used is *Resources*.
[79] In DERCT the term used is *Goals*.
[80] Willars (1993a)
[81] Ibid
[82] Wohed (1993)

WHEN	< something occurs >
IF	< prerequisite is fulfilled? >
THEN	< grant/revoke/alter authorisation! >

Figure 5: A rule.

A rule may state who is to be authorised and under what conditions. It may also state how an administrator is to configure a system component or if and how the administrator is to proceed when a decision-maker requests the implementation of an authorisation.

Rules and intentions are in this thesis not separated unless explicitly expressed; otherwise they are together referred to as *policies*.

All perspectives interrelate and can be described graphically. This is visualised in figure 6.

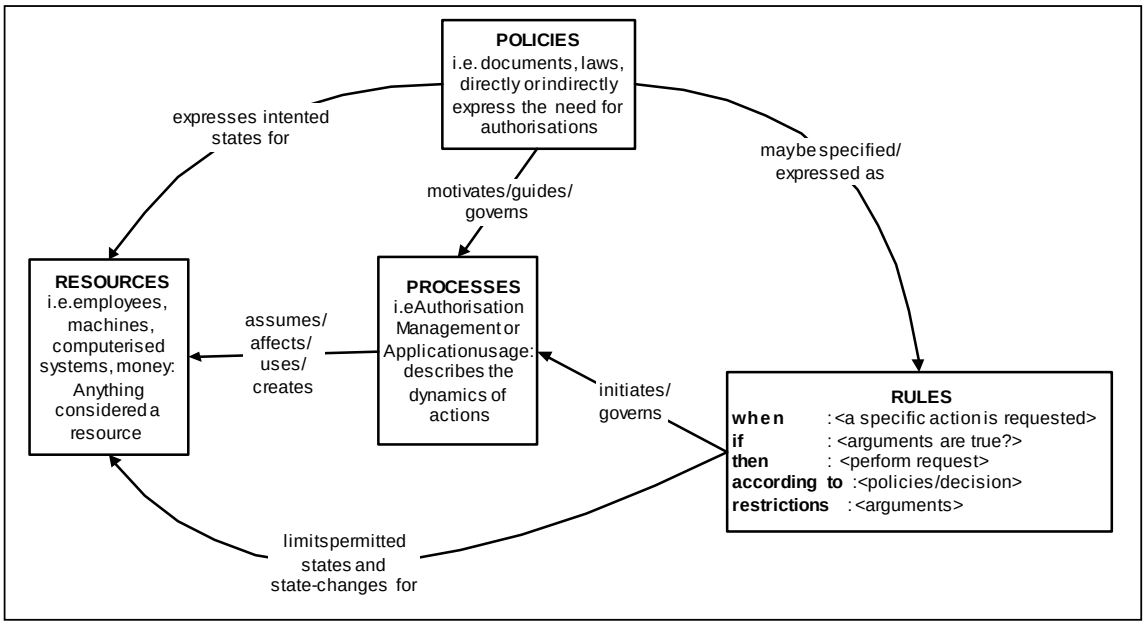

Figure 6: Overview over the interaction between different perspectives.[83]

3.2.3. DIFFERENT FOCUS

What part of the enterprise is in focus, is it a unit, a department or a general process? Is the present day situation being modelled and described or is it a future situation? In this thesis I will initially focus on describing the present-day situation of AM concerning a specific application. Then the focus will change and describe a possible future AM-process concerning the same application.

Considering the real world, all different perspectives and focusing on the AM-process in the organisation today, this could be modelled as in figure 7.

[83] Modified overview from Wohed (1993b)

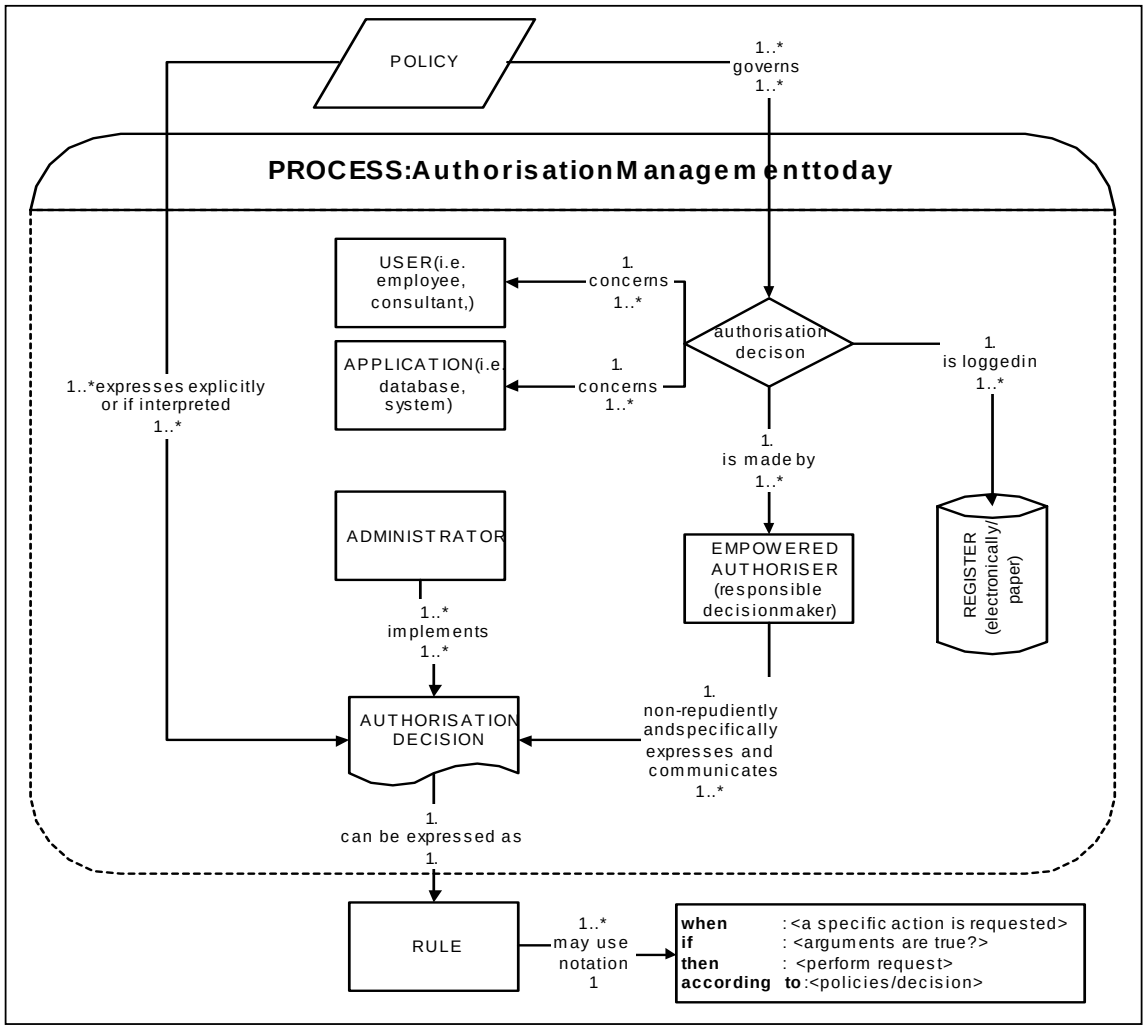

Figure 7: The AM-process of the Organisation today. The notation "1..*" represents a one-to-many relationship. For instance, the expression “1..* governs 1..*” is to be read as; “One or many policies govern one or many authorisation decisions”.

3.3. THE UNIFIED MODELLING LANGUAGE AND USE-CASES

The Unified Modelling Language (UML) is a notation for modelling systems using object-oriented concepts.[84] An initial step is the use of use-cases. Use-cases are narrative documents that, using a simple graphical notation and textual descriptions, describe a sequence of events an actor needs to take in order to complete a process involving a computerised system. Use-cases illustrate and imply requirements in the stories they tell.[85] They describe how actors interact with a computerised system in order to complete a pre-defined process. The computerised system and the processes in the IT-world are considered a black box. It is

[84] Larman (1998)
[85] Ibid

important to point out that a use-case does not show the only possible sequence of events, it shows one likely sequence of events.

3.4. VISUALISING THE AM- AND USAGE-PROCESSES

In this thesis use-cases are the basis for the graphical and textual description used to describe the AM- and usage-processes of both today and tomorrow. They have however been expanded and do not only show an actor interacting with a black box. The use-case based notation in this thesis show all actors involved in the AM/usage-process as well as policies and system components. The system is not considered a black box, instead the interaction between components in the IT-world are described in the same way as the interaction between actors and the IT-world are shown.

The description is general, not considering technical details in further detail than necessary in order to understand the model. The reasons are three. Technical details are not the focus of this thesis, technical details of AMANDA are being developed and finally, it has turned out that the proposed functionality of some technical aspect, e.g. the cross-certification between PKIs, do not function quite in reality as it is stated.

4. CASE STUDY – INTRODUCTION AND DISPOSITION

The empirical part of this thesis, the case study, describes the AM and usage-processes as they function today on one hand and as they could function in the future, using AMANDA, on the other hand. The purpose is to allow a comparison to be made between the processes of today and those of the future.

The presentation of the case study may be divided in three parts where parts 2 and 3 consist of five use-cases numbered 1 through 5. The contents of the case studies are:

Part 1: Presents and describes the independent variables of this study: the organisation and mission of O, the policies, the Keon PKI[86] and the Application. The purpose is to paint a verbal picture of O and the Application and to present the foundations expected to stay the same tomorrow.

Part 2: Describes the dependent variables, the AM- and usage-processes, of today. Use-case 1 describes a typical processing of a request to grant an employee an object-level permission to use the Application. This is described through policy-interpretation, decision-making, communicating the decision and finally implementing the decision within the computerised authorisation-controlling mechanisms of the technical platform. Use-case 2 describes a typical user-system interaction when a user requests to use and is allowed to use an already implemented object-level permission.

Part 3: AMANDA is added to the independent variables and is described with respect to O, the Application and the policies. Then the dependent variables are once again described. This description is set in the future and the policies are expressed within the computerised authorisation-controlling mechanisms of the technical platform. Further, the management-level power of delegating both further management-level powers and object-level permissions have been decentralised, through technical solutions, to map the regular hierarchical structure and chain of command of O. That is, each empowered decision-maker implements his own decisions directly into the computerised authorisation-controlling mechanism of the technical platform. Use-case 3 describes a possible delegation of a management-level power. Use-case 4 describes a possible delegation of object-level permissions and use-case 5 describes a typical user-system interaction when a user requests to use and is allowed to use an object-level permission.

4.1. SETTING THE SCENE

In this section the hierarchical structure and organisation of O is described. O consists of one operational hierarchical structure and one IT hierarchical structure. The purpose and general functions of the Application as well as some administrative matters are described. The technical platform is described, both as it functions today using Bridge-based operations and as it will function in the future using PAC-based operations.[87] The descriptions of the

[86] Even though details in the Keon PKI also change between *today* and in the *future*, the PKI is still considered an independent variable. The reason is that these changes are independent of AMANDA and they are not part of the scope of this thesis. Further, the changes and their implications are described in detail and are easily separated from the changes brought about by AMANDA.

[87] Bridge- and PAC-based operations are different ways of retrieving user log on information. They will be explained in section 4.1.4.

processes today are based on Bridge-based operations. The descriptions of the processes tomorrow are based on PAC-based operations and, of course, AMANDA. Finally the policies of relevance to the AM and usage of the Application are described in a formal way.

4.1.1. ORGANISATIONAL STRUCTURE

The Organisation, O, is a subsidiary of a global organisation active mainly in Europe. Unless otherwise mentioned all descriptions concern the Swedish subsidiary. O has several thousand employees and provides a wide range of services. Millions of mainly Swedish customers use different services of varying extent every year.

O has a hierarchical structure, which consists of a central command that control 16 departments. Each department covers a certain geographical area. Each department, except those covering the two largest geographical areas, is divided into several sales and service units (SSU). The two largest departments are further divided into 3 sub-departments and the SSUs belong to these sub-departments. Guided by central and locally formulated policies, each sub-unit has a large degree of independence to run their businesses. On the central level a number of functions, roles and groups have been defined. Guided by policies, different units are empowered to, to varying degrees, deviate from or extend these definitions. At any given moment the organisational structure could be thought of as a tree consisting of different nodes, each node with attributes and relations to other nodes.

4.1.2. IT-ADMINISTRATION

The organisation of the national IT-administration does not map the operational structure described above. A central unit (IT-central command) answers directly to the central command. Answering to this IT-central command are six IT-administrative units and responsibility for administrating the computerised systems of the 16 operational departments is divided between the six. Requests to authorise different users different authorisations in different systems have, according to policies, to be written documents signed by the requestor. These requests are forwarded to the system-owner that each system has been assigned. The system-owner will decide whether or not the request complies with the policies and facts at hand. Next, the system-owner will sign a written decision and if approved, it will be forwarded to the IT-administrative unit under which the focal employee belongs. Designated administrators in the IT-administrative unit will then implement the decision by changing the authorisation-controlling mechanisms of the technical platform. In reality this implies altering Access Control Lists. Different administrators are responsible for different applications or systems.

The administration of the Application is temporarily an exception. One central system-owner is responsible for receiving, processing and approving or rejecting requests for authorisations from ALL departments and units requesting authorisations, including those of strategic partners that also, according to policies, are authorised to use the Application. Needless to say, the workload of the system-owner is overwhelming. It is not possible for her to practically check if each individual employee of O and those of strategic partners, for whom authorisation is requested, fulfils the requirements stated by different policies. In reality this functions in such a way that the system-owner of the Application *trusts* that whoever has signed a request also has performed the necessary checks of the compliance between each

individual request and policies. Further, one designated ”root” -administrator implement ALL authorisation-decision concerning the Application.

In the future the intention is to decentralise the administration so that it maps the rest of the IT-administrative organisation and processes.

4.1.3. THE APPLICATION

Knowledge about customer-usage and behaviour-patterns are by O considered to be fundamental in being able to provide high-quality services. Therefore a Data Warehouse (the Application) has been constructed containing different information about customers. This information is extracted from a number of transaction systems and the warehouse is batch-wise updated once a day. Five, also service providing, strategic partners are allowed access to varying amounts of information in the application. The strategic partners are S1, S2, S3, S4, and S5, All together several thousand users are authorised to use the application and they make more than 100.000 action-requests per week.

The Application is one of a large number of different applications in use by the Organisation. A prerequisite in order to be able to use any of these applications is access to the Local Area Network (LAN) of the Organisation. A prerequisite to this is having a user-account and a smart card together with a user name and a password. The smart cards are issued by a specific smart card-provider. Administrators of O set up user accounts. The LAN uses Keon PKI. The main-components of the smart card are a picture of the user together with a chip. After inserting the smart card into a card-reader the user is logged onto the system upon entering a valid user-name and password. Upon logging on to the system the user is, on his screen, presented with a standard-set of icons representing different applications used by O. Some users have additional icons and not all users have access to all applications represented on their screen. This depends on what privileges they have been granted by different decision-makers representing different applications or systems.

An authorised user initiates usage of the system by entering facts in one of five GUIs, each enabling different aspects of customer behaviour to be entered and extracted. One concerns individual customers, one products, one organisations, one geographical regions and one age-groups. The only privilege possible to regulate through the authorisation-controlling mechanisms is object-level permissions. An employee either has permission or not. The Application is constructed in such a way that having a permission always implies both SEARCH *and* READ using any of the 5 GUIs. There are no alternatives.

In reality it is the two strategic partners S1 and S2 that make use of the Application in the same way as the employees of the Organisation. In order for external users to be able to use the Application they need individual user-accounts with O. These accounts are set up by administrators in O. The external users also need a smart card provided by O. A fairly complex manual process, initiated by the strategic partner requesting access to the Application, is performed each time an employee of a strategic partner is to be given the privilege to use the Application. This process involves several instances of signing papers representing decisions and communicating them between persons in different organisations. Of course the employees of O also need user-accounts and smart cards in order to be able use different applications. This is not considered further in this thesis.

4.1.4. KEON-THE TECHNICAL PLATFORM[88]

O uses the Keon PKI. Their strategic partners either do not use PKIs or use PKIs that cannot interact with Keon. Keon consists of three main components, the Desktop, the Agent and the Security Server (SS). A fourth component, the Management Console, is also of interest for the case study as it is used to implement authorisation decisions.

The *Desktop* provides the interface with the user on one hand and the Agent on the other. A user logs on and is authenticated on the Desktop that also provides encrypted communication with Agents. Different *Agents* controls authorisations to different Application Servers and mediates all communication between a specific Application Client on the Desktop and its corresponding Application Server. The Agent also communicates with the SS.

Communication between the different components is either *Bridge-based* or *PAC (Privilege Attribute Certificates)-based operations*. In both cases the Agent mediates all communication between the Desktop and Application Server. O uses bridge-based operations today due to the fact that the Security Servers are old (version 4.5) and do not support PAC-based operations. Within a few years the 4.5-SSs will have been replaced by SS version 5.5 that support PAC-based operations. Below a brief overview of the different operations are given.

Communication between	Bridge-based operations	PAC-based operations
Desktop - Agent	DASP-protocol used for protocol negotiation. ALLTAAS-protocol used for strong authentication between entities. ALLTAK-protocol used for session encryption. All protocols are RSA-specific	Secure Socket Layer (SSL) is used.
Agent - SS	Symmetric key, installed on both entities, used for encryption.	SSL is used to send entries to logging system.
Desktop - SS	All communication is mediated via the Agent.	SSL is used to request and transfer PAC from SS

Figure 8: Communication using Keon Bridge-based vs. PAC-based operations.

Secure communication between Agents and Application Servers is accomplished by having the Application Servers run on the Agent hosts. In both PAC- and Bridge-based operations the privileges[89] are stored in Access Control Lists (ACL) in the SS. Besides the differences listed above, it is the retrieval of user-specific log-in credentials from the ACLs in the SS that differ.

With *Bridge-based operations*, each time a user wishes to use an application he clicks on the corresponding icon on his Desktop. The user's *user-credentials* are, via the Agent, forwarded to the SS where they are mapped to *given-credentials*. The given-credentials in turn are mapped to *database-credentials* that are returned to the Agent who uses them to log the user on to the Application Server before opening a session with the Desktop of the user. Each time the user wishes to access some application a new session has to be set up this same way.

[88] For a more detailed description of Keon, see Appendix A.
[89] In Keon synonymous with *User Access Rights*.

Using *PAC-based operations* the user's request is forwarded directly to the SS. In the SS all database-credentials for all applications a user is authorised to use will be collected in a PAC. PACs are a kind of X.509 v.3 certificate that is used to specify all authorisations a user has been granted within a system. The PAC will be returned to the Desktop where it will be kept. When the user wishes to access an Application the specific database-credentials are retrieved from the PAC by the desktop and forwarded to the Agent who will log the user on to the Application Server and mediate the communication between the Desktop and the Application Server.

PACs ease the workload of the SS and the system in general. The SS does not have to be bothered until it is time to retrieve a new PAC. Of course, one drawback with PACs is that a privilege issued through a PAC cannot be revoked as long as the PAC exists. Therefore the lifetime of a PAC is configurable and is typically set to be short, e.g. a few hours or the length of a normal working day.

Privileges are in the IT-world administrated through altering the ACLs in the SS. Empowered administrators may via their desktops log into their Management Consoles, physically residing on top of the SS, and alter the ACLs. The ACLs in the SS are managed only through the Management Console.

4.1.5. THE POLICIES

Different policies are of interest in the Application perspective. They regulate the delegation of both management-level powers and object-level permissions as well as the work of the IT-administration. Today only object-level permissions are possible to express and regulate in the computerised authorisation-controlling mechanisms of the IT-world. This is done through ACLs and the Management Console. *Policy* set *guides* and governs the delegation of powers and permissions concerning the Application in specific. *Policy B* guides and governs the delegation of powers and permissions in O in general with respect to IT-systems. *Policy C* guides and governs IT-system-ownership within O in general. *Policy* set *D* guides and governs the delegation of powers and permissions concerning Keon.

The policies below are presented in terms of *power* (or *empowerment*), *permission*, *obligation* and *delegation*. Power, permission and delegation are always and without exceptions expressed with regard to the right to *search* AND *read* the database using ANY of the five graphical user interfaces. Where it is written, "Policy signed by:" this indicates the empowering authority.

4.1.5.1 POLICIES GUIDING AND GOVERNING AUTHORISATION

<u>(Policy A1) Strategic mission statement</u>
(Policy signed by: International board of Directors)
Initiated and formulated by the international Board of Directors. This is the root-policy that expresses the intention to offer the same quality of services regardless of where in the world this service is offered.

<u>(Policy A2) Operational interpretation</u>
(Policy signed by: International board of Directors)

Derived from the Strategic Mission Statement. The policy states that in order to fulfil the strategic mission statement a Data Warehouse (the Application) has to be constructed.

Each national subsidiary is *empowered* to *delegate empowerment* to designated strategic partners.
Each national subsidiary is *empowered* to *formulate* policies and restrictions regulating the usage of the Application.
Users may only be *permitted* to use the Application when fulfilling missions.
Each national subsidiary is *empowered* and *obliged* to control and monitor the Application so that usage only takes place in accordance with policies.

(Policy A3) Swedish guidelines for authorisations in the Application
(Policy signed by: Central Command of O)
Derived from the Strategic Mission Statement and the Operational interpretation.

All departments of O are *empowered* to *delegate permissions* to their respective employees.
Designated strategic partners are *empowered* to *delegate permissions*, given a joint mission.

(Policy A4) Interpretation of guidelines
(Policy singed by: Central Command of O)
Derived from the Swedish guidelines for authorisations in the Application.

Employees of S1 and S2 are *permitted* given condition A.
Employees of S3, S4 and S5 are *permitted* given condition B.
Employees of S1, S2 and S3 are *permitted* given condition C.
The Central Command of Sweden is *empowered* to formulate policies and restrictions regulating usage of the Application after consulting organisation I.
All institutions using the Application are *obliged* to act accordingly.

(Policy B) Guidelines
(Policy signed by: Central Command of O)
These are not derived from the application-specific policies presented above but are instead general guidelines for whom and under what circumstances power and permission to IT-systems within O may be delegated:

Permissions may only be *delegated* to role: "employee"[90] that:
- posses knowledge to use the focal IT-system.
- have a need to use the focal IT-system in order to fulfil work-tasks.
- are reliable out of a security perspective.

As soon as any of these requirements are not fulfilled the permission is to be altered or revoked. Previous action taken has to be possible to trace in the transaction log.

Role: "Head of a Department", or someone by him appointed, is *empowered* and *obliged* to *delegate power* and *permission* to sub-ordinate role: "employee" and to employees of other organisations on joint missions with O, unless otherwise regulated through another, superior, policy.

[90] The sentence is to be read as: "Permissions may only be delegated to individuals that are considered to be employees".

A decision to delegate must be documented. Permission may not be implemented unless such a written document exists.

4.1.5.2. POLICIES GUIDING AND GOVERNING ADMINISTRATION

(Policy C) Guidelines concerning development and usage of IT-systems
(Policy signed by: Central Command of O)

Each IT-system used by a department is to have a designated employee with role: "system-owner". Each system used by several departments is furthermore to have a centrally designated employee with role: "system-owner".
Role: "Head of Department" is by default system-owner.
Role: "Head of Department" is *empowered* to *delegate* system-ownership.
Role: "system-owner" is *obliged* to monitor and control usage of a system so that usage is performed in accordance with policies.

(Policy D1) Instructions for usage of the Keon Security Server
(Policy signed by: IT-central command)

Role: "Keon-root-administrator" is *empowered* to perform any actions deemed necessary within Keon SS. Manual routines involving several persons are to be established in order to monitor and control role: "Keon-root-administrator".
Role: "Keon-root-administrator" is *empowered* to *delegate power* to administrative processes to role: "sub-administrator" according to policies.

(Policy D2) Organisation of the Keon-administration
(Policy signed by: IT-central command)
Derived from Instructions for usage of the Keon Security Server.

Role: "Keon-root-administrator" is *empowered* and *obliged* to *delegate permission* to administrative processes to designated roles: "sub-administrator". Designated roles: "sub-administrator" may only implement permissions decided by empowered role: "decision-maker". (Such a decision is in fact, according to the last sentence in policy B above, a paper signed by an authorised decision-maker (authors note))

Not all answers concerning AM are found explicitly through reading the policies, sometimes interpretation is necessary as is the consulting of other policies not presented here. Also, additional policies not identified in this study may exist and regulate different aspects and steps in the AM-process.

4.2. TODAY

In this section two use-cases are presented. Use-case 1 describes the full AM-process of today. Use-case 2 describes how usage of permissions is controlled today.

4.2.1. USE-CASE 1: AUTHORISATION MANAGEMENT TODAY

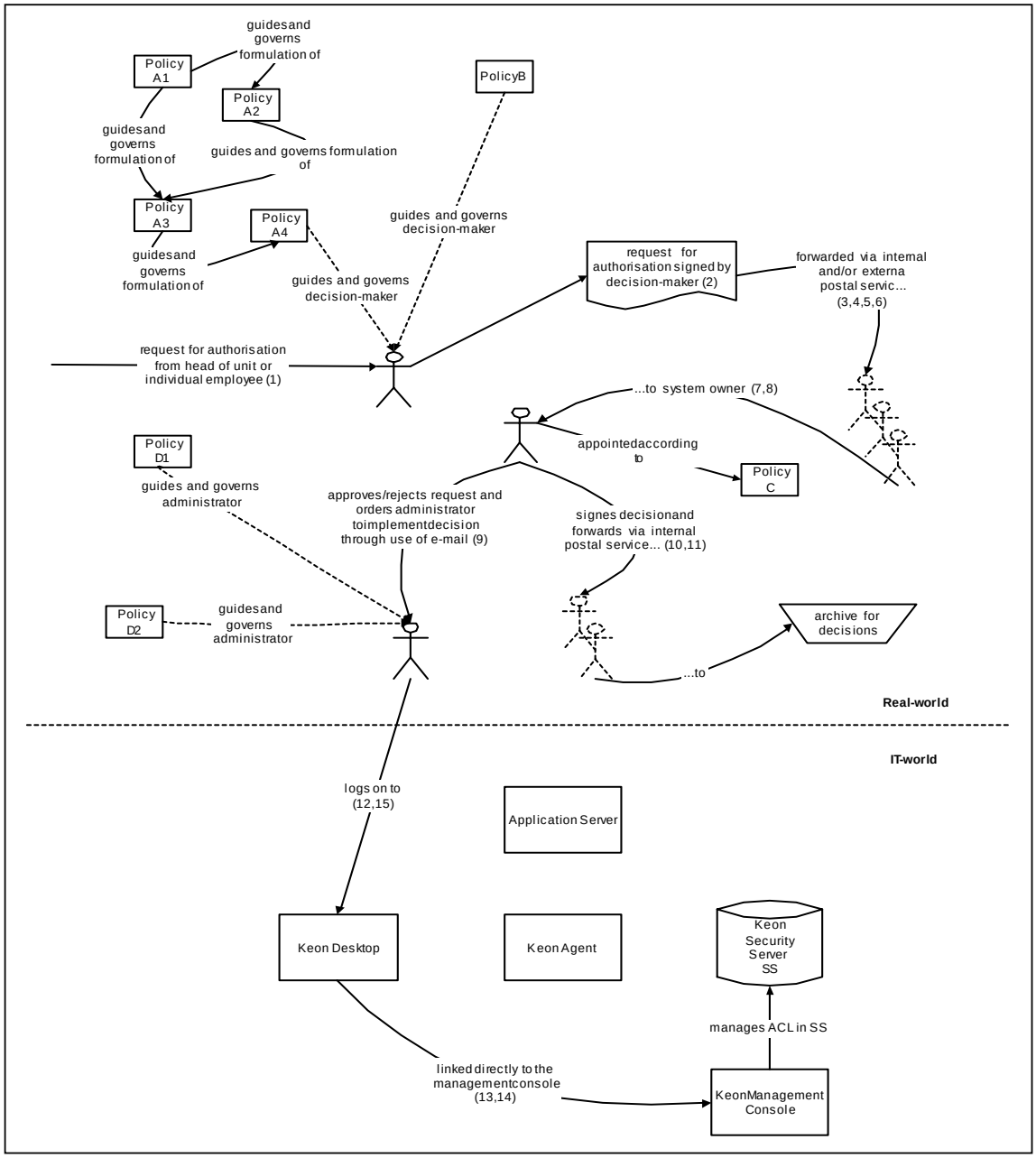

Figure 9: Use-case 1. The AM-process today.

1. This use-case begins when an employee, currently in the role of Head of Department and therefore also a decision-maker according to policy B, receives a request for authorisation. The Head of Department could also, according to policy B, have delegated this power to someone else. As it is not possible for the decision-maker himself to judge if the appointees fulfil the prerequisites according to policy B and policy A2 he trusts the requester has performed this check. In practice it is assumed that anyone that can be authorised fulfil the requirements of policy B. He further trusts that the appointees are employees according to policy A4.
2. The decision-maker decides to approve the request and puts his signature on the paper previously signed by the requestor according to policy B. The paper contains a list of names, social security numbers and organisational belongings of the appointees. Telephone-calls and e-mails are not accepted as requests or as a valid decision. According to policy B and D2, both a valid request and a valid decision have to be a signed piece of paper.
3. The decision-maker encloses the decision in an envelope addressed to the system-owner, according to policy C.
4. Personnel from the internal postal service of O collect the enveloped decision, bring it to their office and add a stamp to the envelope.
5. The internal postal service mails the envelope using an external postal service.
6. The envelope is forwarded outside of O and eventually delivered back to O at the destination address of the system owner.
7. The internal postal service delivers the envelope to the mailbox of the system-owner.
8. The system-owner collects the envelope. According to policy C he is obliged to control that usage of the system is performed in accordance with policies. However, in these cases, he trusts that whoever has signed the request has made sure that all appointees fulfil the prerequisites according to policy B.
9. Facts about the appointees are entered into a LAN e-mail-system that is forwarded to the Keon master-root administrator. (policy D1)
10. The request, signed by the decision-maker, is signed by the system-owner (policy B & C).
11. The signed request is forwarded in an open envelope through internal postal service for archiving.
12. The Keon-root administrator receives and reads the LAN e-mail that is interpreted as a decision according to D2.
13. The Keon-root administrator (already authenticated by Keon as he uses the LAN e-mail) logs on to the Management Console using a password and is provided with the interface to the Keon Security Server Master where the ACL of the Application are kept. The Management Console resides directly on top of the Security Server.
14. Using a script the appointees are transferred from the LAN e-mail into the ACL. They have now been granted the privilege to use the Application.
15. The Keon-root administrator logs off from the Management Console.

Summing up use-case 1: The AM-process is, to a large extent, a manual process. The AM-process is separated in two distinct parts, that of the real world and that of the IT-world. The policies exist in the real world where they are interpreted and decisions concerning the delegation of privileges are made based on these policies. These decisions exist physically as a piece of paper stating what privilege to delegate, to whom and under what circumstances. The decision is signed by the empowered decision-maker and then forwarded to the system-owner. The system-owner is obliged to control usage of the Application. This includes

controlling that usage is permitted according to policies. Because of the large number of users and requests for permissions it is impossible for the system-owner to do this. The control enforced by the system-owner at this point is limited to handling large and obvious violations of policies. The system-owner has to trust that the requested decision-makers have performed the necessary control. The request is then forwarded, via local e-mail, to the Keon-master-root administrator who implements the decision by altering ACLs in the Keon Security Server. This implementation is the only part of the whole AM-process that takes place in interaction with and within the IT-world. (In this context I do not consider the usage of e-mail being part of the IT-world). The authorisation-controlling mechanisms are set in the IT-world. Steps 1-11 refer to the part of the AM-process taking place in the real world. Steps 13-14 refer to those that take place in the IT-world and steps 12 and 15 refer to steps that represent interaction between the two worlds.

4.2.2. USE-CASE 2: USAGE TODAY USING BRIDGE-BASED OPERATIONS

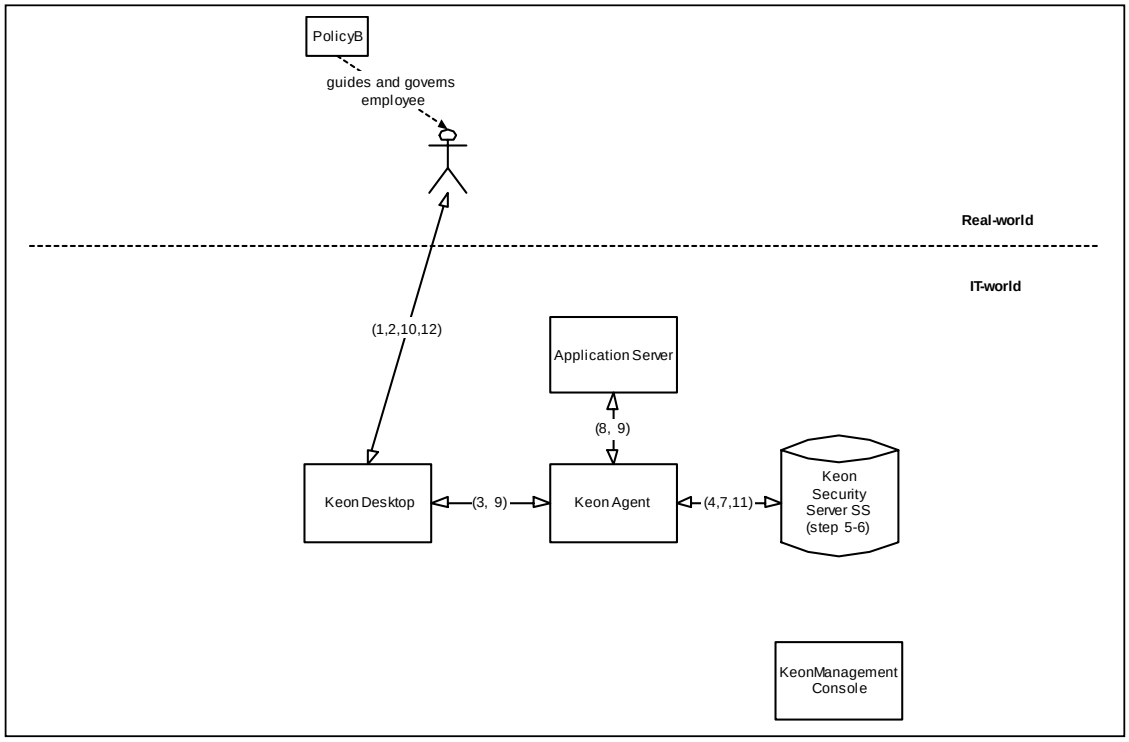

Figure 10: Use-case 2. Using an object-level permission today.

1. This use-case begins when an employee of O wishes to use his privilege to use the Application. Remember that this privilege may only be of one kind, that is, to search and read the Application Database using any of five GUIs. The employee has already logged on to the system using his smart card and password and authentication has taken place by the verification of CA-signatures. On the GUI of the Desktop that the employee uses, a number of icons representing different application clients are presented.
2. On the GUI of the Desktop, the employee clicks on the icon associated with the Application. The fact that the employee has been granted authorisation to use the Application implies that O trusts the employee to act according to policy B. This, as well as the fact that the employee has been issued a smart card and has a user-account, may be

considered preventative control mechanisms the employee has already passed. It is in fact the only preventative control mechanism at hand. Detective control mechanisms may be performed later. For instance, the actions of an employee according to log-files may be compared to policy B and the actual work-tasks, working-hours etc of the employee.

3. The user-credentials of the employee are sent to the Agent-instance associated with the Application.
4. From the Agent the user-credentials are forwarded to the Security Server.
5. In the Security Server the user-credentials are mapped to the corresponding given-credentials associated with the employee.
6. The given-credentials are in turn mapped to the database-credentials, containing data necessary to log the employee on to the Application Server.
7. The database-credentials are returned to the Agent.
8. The Agent uses the database-credentials to log the employee on to the Application Server.
9. The Agent opens a connection between the Application Server of the Application and the Application Client on the desktop in use by the employee.
10. The employee may now use the Application. All communication between the client and the application goes via the Agent.
11. The Agent sends log entries to the SS.
12. The employee logs off from the Application.

Summing up use-case 2: As shown in use-case 1, management-level powers are delegated, expressively through policies or through decisions based on interpretations of policies. Except for the last part, when the administrator interacts with the system in order to alter the ACLs, this is a wholly manual process taking part in the real world.

Object-level permissions are, on the other hand, fully controlled by the system as shown in use-case 2. However, once an employee has been granted such permission there is no way, by means of the computerised authorisation-control mechanisms, to control that the foundation for granting the permission, according to policy B, does not change. The computerised control-mechanism cannot control that the employee himself acts according to policy B. Instead, O expects and *trusts* that the employee will act according to policy B. All the computerised control-mechanism does is grant or reject the request to use the Application. This computerised decision is based on the identity of the requestor. Steps 1, 2, 10 and 12 take place in interaction between the user in the real world and the system in the IT-world. All other steps are performed within the IT-world.

Using Bridge-based operations, as has been the case here, the whole procedure has to be gone through again when the employee wishes to use the Application again.

4.3. THE FUTURE

In this section the future AM- and usage-processes will be described. Initially the components META-POLICY, APPLICATION-POLICY, ORGANISATIONAL DATABASE and CERT of AMANDA will be described with regard to the Organisation, the Application and the policies previously presented. The purpose of doing this is to be able to present use-case 3 that will show how facts are put in the AMANDA-component FACT. This in turn is necessary to be able to present use-case 4 that will show how entries are made in the ACL. These two use-cases will exemplify the future AM-process – the delegation of both management-level powers

and object-level permissions. Use-case 5 will show how a request to use an object-level permission is processed.

As far as Keon is concerned, the Desktop, The Agent and the Application Server function as before. The Security Server and the Management Console have been replaced by AMANDA-components that are considered to be an application of their own. Daemons handle the interaction between the different AMANDA-components and between AMANDA and the Keon-components. PACs are used but instead of being transferred to the desktop from the SS they are transferred from AMANDA.

4.3.1. SETTING THE FUTURE SCENE – DRESSING AMANDA

The different policy-assertions presented below are numbered in order to simplify the description and reading of the use-cases to come. Also, at the end of each assertion is a comment. If the assertion is derived from a real world policy, it is referred to. However, not all assertions presented below represent the real-world policies previously formalised. Much that often is implicit in the real world, such as the definition of an employee or the meaning of "having the necessary knowledge" about something. This must be explicitly expressed and defined in the IT-world. When presented assertions do not represent the previously presented policies of O it is due to this fact. In these cases the assertions have been added to make this section complete. Also, in order to present the use-cases below only a small portion of some of the policies are used. In order for the assertions presented below to be easily understood they have been formalised using normal English.

META-POLICY (MP)

#	Assertion	comment
M1	The Central Command of O empowers MP to be the Meta-Policy of O if three members of the Central Command sign it. Signed by (key1, key2, key3)	This is the root-assertion validating and initiating the MP, given it is signed by three members of the Central Command. The assertion is itself signed by three members of the Central Command, (keys 1,2 and 3), as is the MP in M14.
M2	The Central Command is empowered to delegate Power and Permissions to any application to all heads of Department.	This is derived from the real-world policies A2 and A3.
M3	Anyone who is Head of Department is system-owner over all applications that the particular Department owns.	This is derived from the real-world policy C.
M4	Anyone who is Head of Department is empowered to delegate Power and Permission concerning any applications that the particular Department owns. Delegation may be done to sub-ordinates within the Department.	This is derived from the real-world policy B.
M5	Anyone who is Head of Department is empowered to delegate System-ownership to anyone employed within O.	This is derived from the real-world policy C.
M6	Anyone who is the Chief Technical Officer (CTO) of O is empowered to issue Application Policies together with the System-owner of the particular Application. Applications are initiated only if signed by both the CTO and the particular System-owner.	Not derived from real-world policies. Sets a form for initiating Application Policies. Compare to M1.
M8	Every employee of O has the same basic set of attributes as defined in the EMPLOYEE-ATTRIBUTE-LIST.	Regardless of what role an employee has been assigned within O, everyone has the same basic set of attributes.

M9	The EMPLOYEE-ATTRIBUTE-LIST consists of the following attributes: a *public-key*, a *role-assignment*, a *function*, a *membership of a department*, a *membership of a sub-group*, a *membership of one or more project-groups*, a *security clearing* with regard to a particular object, an *application knowledge* with regard to any application the employee has privileges with, *restrictions* concerning a particular object.	This is a definition of the basic set of attributes and their values that all employees will have. Current instances of employees, and their EMPLOYEE-ATTRIBUTE-LISTS, are found in the Organisational Database (ODB).
M 11	O has 16 Departments	The basic current organisational structure is defined
M 12	Each Department may own one or more Sales and Service sub-groups	Department have at least one sub-ordinate Sales and Service Unit.
M 13	Each Sales and Service sub-group may set up project-groups.	A Sales and Service Unit may them-self create different project-groups
M 14	SIGNED by: key1 AND key2 AND key3	As found out from the ODB, key 1,2 and 3 are members of the Central Command. According to M1 three members of the Central Command have been empowered to issue this policy if they sign it together.

Figure 11: Parts of the meta-policy of O.

APPLICATION POLICY (AP) (for the Application)

#	**Assertion**	**comment**
A1	Having a permission implies the right to search AND read ANY table, or portion of a table.	Not a policy. Expresses how the Application actually functions today.
A2	Having a permission implies the right to use any of the Graphical User Interfaces (GUI) numbered 1, 2, 3, 4 or 5.	No policy, this expresses how the Application actually functions today.
A3	Restriction for delegating a permission: The EMPLOYEE-ATTRIBUTE-LIST attribute "security-clearing" must be at least 2.	Derived from policy B. In order for a computerised system to compare the security clearing to a restriction it somehow has to be explicitly expressed. The security clearing is bound to the employee and a specific subject, e.g. an application.
A4	Restriction for delegating a permission: The EMPLOYEE-ATTRIBUTE-LIST attribute "application-knowledge", for this particular Application, must be at least 2.	Derived from policy B. In order for a computerised system to compare the application-knowledge to a restriction it somehow has to be explicitly expressed. The security clearing is bound to the employee and a specific application.
	SIGNED by: key 3 AND key45	The CTO and the system owner issue this policy together as stated in M6. In the ODB it will be possible to find out that key 3 is the CTO of O and that key45 is the system-owner.

Figure 12: Parts of the application-policy of the Application.

Comment: A1 and A2 exactly express the actual object-level permission of today. It would of course be possible to express this differently, such as: "If the delegatee is an employee of a strategic partner then only GUI 4 and 5 may be used". Real world policy A4 states such limitations. Interpreting policy A4 and mapping it into the AP might be expressed as above. However, strategic partners are not considered in the use-cases and concerning the employees

of O there is no need to be more precise in this matter. Being able to do it is however an important feature of AMANDA.

ORGANISATIONAL DATABASE (ODB)

In this database unique instances of different nodes, defined in the MP, are described. Each node is uniquely identified. In this description the unique identifier is assumed to be a public key. Five nodes (two employees, one department, one Sales and Service Unit and one project-group) and part of their attribute lists are shown to exemplify this. In this example, not all attributes shown are defined in the MP.

Employee
Attribute-list:
Public-key: key60
roles: head of SSU
functions: ()
member of department: A
member of sub-group: SSU big-sales
member of project-group: ()
security clearing: 2 *concerning subject:* (Application)
application knowledge: 1 *concerning application:* (Application)
restrictions: () *concerning subject:* ()

Employee
Attribute-list:
Public-key: key102
role: sales-man
function: ()
member of department: A
member of sub-group: SSU big-sales
member of project-group: profit
security clearing: 3 *concerning subject:* (Application)
application knowledge: 1 *concerning application:* (Application)
restrictions: () *concerning subject:* ()

Department A
Attribute-list:
Head of Department: Key 50
Have employees: key(50-70, 100-700, 10980, 10983, 11021, 12321)
Own SSU: big-sales, mega-big-sales, aggressive, offensive

SSU big-sales
Attribute-list:
Belong to: A
Have employees: key(60, 100-114, 238-311, 314, 398, 402, 404, 10980, 10983, 11021, 12321)
Own project-group: profit

Project-group profit
Attribute-list:
Belong to: big-sales
Expires: Dec 31, 2001
Have members: key(100, 101, 102)

Figure 13: Parts of the Organisational Database of O.

CERT

Now two attribute certificates are issued as stated below. As the issuer signs the AC using the private key of his PKC the attribute certificate is bound to the issuer. What object the certificates concern is implicit as they are forwarded to the CERT-database for a specific object, in this case the Application. Note that certificates A and B represent delegation in three steps. First Key50 delegates a power to further delegate permissions to key60. Then Key60 uses this power and delegates permissions to a group.

#	**Assertion**
A	Key60 is empowered to delegate permission to ALL own members. SIGNED by key50
B	All members of project group profit are permitted. SIGNED by key60

Figure 14: Two attribute certificates delegating power and permissions respectively.

4.3.2. USE-CASE 3: DELEGATING A MANAGEMENT-LEVEL POWER TOMORROW

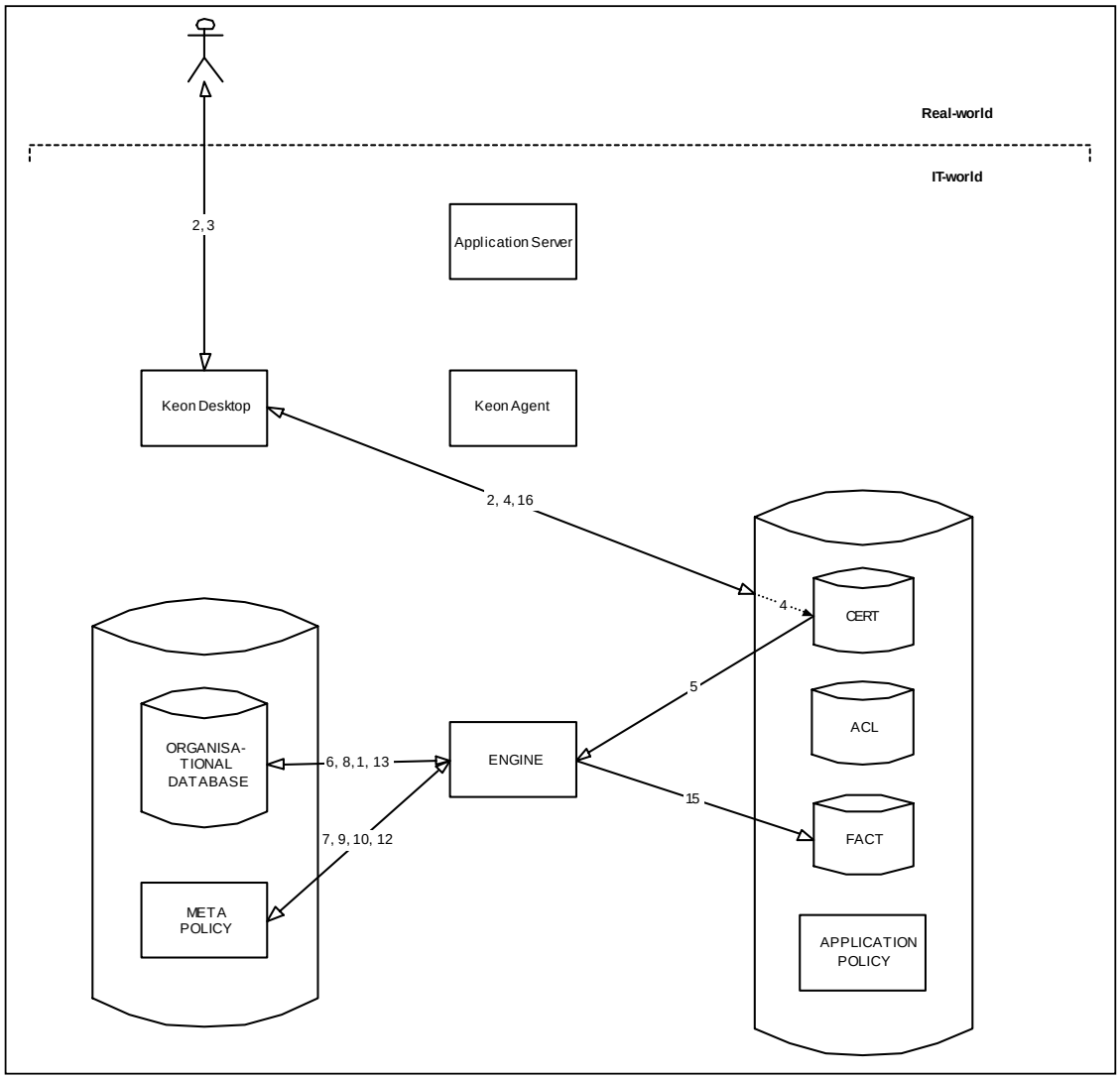

Figure 15: Use-case 3. Delegating a management-level power.

This use-case will describe the processing of certificate (A) stating: “Mr Pyle is empowered to delegate permission to ALL own members. SIGNED by key50”.

1. This use-case starts when the present Head of Department A decides to delegate the management-level power to further delegate object-level permissions. He decides to delegate this power to the head of one of the Sales and Service units belonging to A. The Head of Department A has this right according to Policy C. He is already logged on to the system and is authenticated.
2. By clicking on the administration icon for the Application, he downloads the graphical user interface (GUI) designed to delegate privileges within the Application.
3. By clicking and dragging in the GUI he states that he wants his sub-ordinate head of Sales and Service Unit “big-sales”, Mr Pyle (key60), to be able to authorise his own personnel to use the Application.
4. When acknowledging his intention to the system by clicking on an “OK” in the GUI, an attribute certificate,[91] digitally signed with the current private key of the Head of Department A (key50), is created and sent to CERT.
5. CERT forwards the new certificate to the ENGINE.
6. The engine will communicate with the Relational database ODB and retrieve the data at this moment known needed, that is the attribute list of key50 and of Mr Pyle.
7. The engine will establish connections with MP.
8. From the attribute-list of key50 the engine will find out that this key currently has the role of Head of Department A.
9. From the MP the engine will find out that anyone who is Head of Department is empowered to delegate both power and permission to sub-ordinate employees within his own department (M4). Backtracking the chain of delegation from (M4) to the root (M1) through (M2) validates the authority of this empowerment.
10. From (M8) the engine concludes that everyone employed by O has the same basic set of attributes.
11. From the attribute-list of Mr Pyle (key60) the engine will find out that Mr Pyle is head of sales and service-unit “big-sales”.
12. From (M12) the engine finds out that a department of O has at least one sub-ordinate SSU.
13. From ODB, that has to be contacted again, the engine will find out that one of Department A’s sales and service-units are called “big-sales”.

The engine now has gathered all the knowledge necessary in order to be able to reason about the compliance of certificate A with the policies.

14. The engine will reason and conclude that certificate (A) complies with MP and AP. That is, key50 (Head of Department) is empowered to delegate power to further delegate permissions to key60, Mr Pyle.
15. The engine enters the following facts in FACT:

[91] For instance using the X.509 v3-extension field to host the assertion or assertions.

FACT

#	Fact
(F1)	key50 has power to empower subordinate employees that belong to Department A
(F2)	key50 has power to permit subordinate employees that belong to Department A
(F3)	key60 has power to permit subordinate employees that belong to big-sales

Figure 16: Facts concerning management-level powers, derived from AC A.

16. The daemon handling the communication between the different AMANDA-components will send a confirmation to the Keon Desktop.

Summing up use-case 3: The MP and AP closely map the real world policies they represent. For every question the engine needs to ask in order to be able to reason about the compliance of any certificate with policies the answers has to be explicitly expressed. Such questions that may arise are "If key50 has signed the certificate, who does key50 belong to and is this entity empowered to issue such an attribute certificate?" or "What is "big-sales" and how does it relate to key50?" Such questions are not answered by the real world policies. In order to answer these questions, as we have seen, more has to be formalised and expressed within the authorisation-controlling mechanism of the system than just the application-specific policies. Depending on where the system-boundaries are drawn this can include identifying and formalising a great deal of details of an organisation. Succeeding in doing this would also, as an indirect result, bring about a more exact preventative control mechanism. It does not really matter if someone issuing an attribute certificate is up to date with the latest changes in facts or policies. The engine will reason about it. If the statement in the attribute certificate complies with policies and facts a power or a permission will be delegated, revoked or altered. Else nothing will happen. However, it is unlikely that decision-makers will just issue ACs randomly. They will of course know the policies of their organisation. What is important is that a much more precise and flexible control-mechanism has been incorporated within the computerised system and that the manual processes are reduced. Only steps 2 and 3 takes place in interaction between the user in the real world and AMANDA in the IT-world. All the other steps take place in the IT-world.

Every answer the engine finds, concerning the delegation of privileges, to questions it needs to ask in order to reason about attribute certificates, is put in FACT. FACT contains continuously updated and therefore valid statements concerning a certain object.

4.3.3 USE-CASE 4: DELEGATING OBJECT-LEVEL PERMISSION TOMORROW

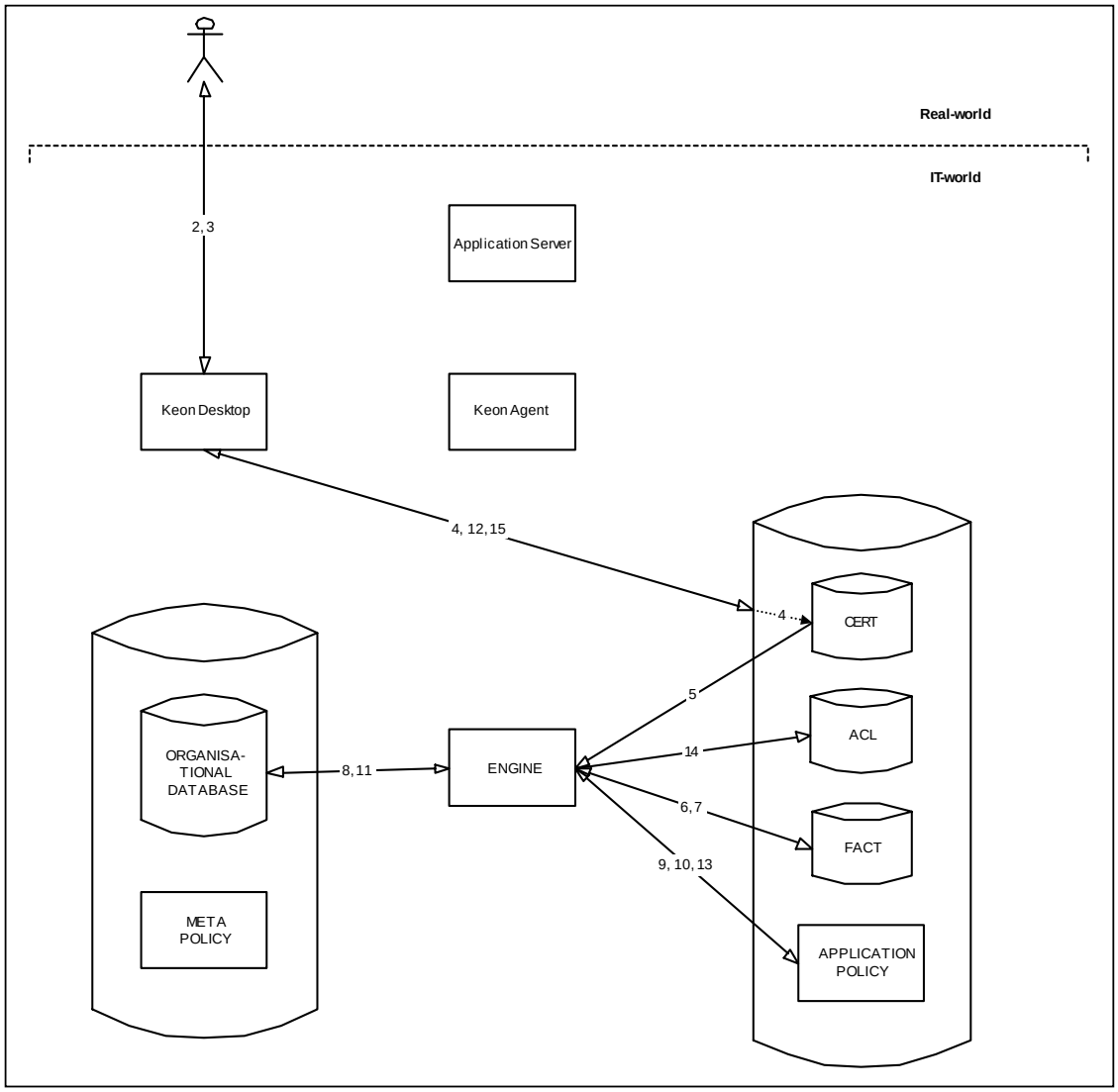

Figure 17: Use-case 4. Delegating object-level permissions.

This use-case will describe the processing of certificate (B) stating: “All members of project group “Profit” are permitted. SIGNED by key60”

1. This use-case begins when Mr Pyle, head of Sales and Service Unit “Big-Sales” of Department A decides to authorise all sub-ordinate employees in the project-group “Profit” to use the Application.
2. By clicking on the administration icon for the Application, he downloads the graphical interface designed to delegate privileges within the Application.
3. By clicking and dragging in the GUI he states that he wants to authorise all members of the project-group “Profit” to use the Application.

4. When acknowledging his intention to the system by clicking on an "OK", an attribute certificate, digitally signed with the public key of Mr Pyle's (key60) is created and sent to CERT.
5. CERT forwards the new certificate to the ENGINE.
6. The Engine connects to FACT to see if the answers to questions that needs to be found can be found here or if any other components have to be queried.
7. From FACT the engine will find out that key60 may delegate object-level permissions to members of Sales and Service-unit "Big Sales" (F3).
8. The engine will communicate with the Relational database ODB and retrieve the data at this moment known needed, that is the fact that the project-group "Profit" belongs to SSU "Big Sales" and the keys and employee-attribute-lists of the members of "Profit" (key100, key101, key102)
9. The engine will establish a connection with AP to find out what is stated in the local policy.
10. From the AP the engine will find out that both the employee-attribute-list-attributes "security clearing" and "application-knowledge" "must be at least 2" (A3 and A4).
11. From the employee-attribute-lists of key100, key101 and key102 the engine will find out that these standards are met except for key102 who only has an application-knowledge of 1.
12. The daemon handling the communication between the different AMANDA-components will send an error-message concerning the inferior application-knowledge of "Profit-member with key102" to the Keon Desktop. The requested permission concerning key102 will not be granted.
13. From the AP the engine will further find out that a permission to use the Application implies the right to both search and read and also the right to use all 5 GUIs.
14. The engine will enter the new values in the ACL:

ACL

Key	search	read	GUI1	GUI2	GUI3	GUI4	GUI5
Key101	Y	Y	Y	Y	Y	Y	Y
key102	Y	Y	Y	Y	Y	Y	Y

Figure 18: An ACL expressing object-level permissions.

In figure 18 the "Y" indicates a "yes, key 101 is allowed to search, read, use GUI1, use GUI2, use GUI3, use GUI4, use GUI5".

The engine will enter new facts in FACT:

#	Fact
(F4)	Any employee to be permitted must have a security clearing of minimum 2.
(F5)	Any employee to be permitted must have an application-knowledge of minimum 2.

Figure 19: Facts derived from AP.

The daemon handling the communication between the different AMANDA-components will send a confirmation to the Keon Desktop concerning the new permissions concerning "Profit"-members with keys100 and 101.

Summing up use-case 4: When reasoning about the compliance of this attribute certificate the engine initially consulted FACT to find out if any answers could be found there. The engine

did also find that the request in the attribute certificate B was valid and there was no need to proceed to querying the policies in this case. When proceeding to update the ACL however, new questions were raised calling for answers not found in FACT and therefore the AP and the ODB had to be consulted. The resulting object-level permissions were put in the ACL so that requests for permissions easily and quickly can be looked up. Also, as an indirect result of reasoning about the compliance of certificate B, new facts were found and FACT was updated. For each additional fact in FACT the need to reason about policies decreases. Each time the engine has to reason about certificates questions have to be raised and answers found. A large portion of the requests to delegate powers and permissions are likely to be the same, e.g. the Head of Department has delegated the right to delegate permissions to use the Application to the head of a unit with a large employee turnover. Each time the head of this unit need to revoke and grant privileges a certificate is issued, all the engine has to do is go to FACT and there find out that the head of unit may do this. As reasoning about the compliance of certificates will not be necessary, the workload of the system will be eased. FACT is continuously kept updated and includes statements concerning the presently valid delegation chains and their restrictions.

4.3.4. USE-CASE 5: USAGE TOMORROW USING PAC-BASED OPERATIONS AND AMANDA

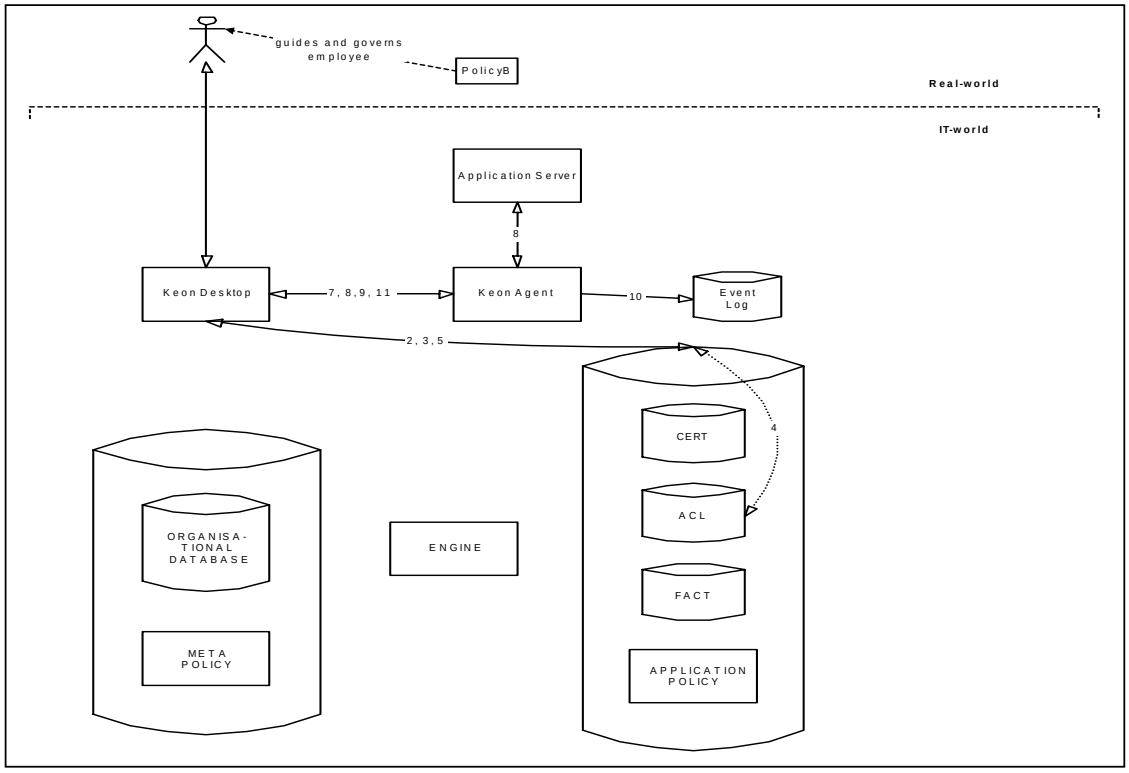

Figure 20: Use-case 5. Future usage.

1. This use-case begins when Mrs Fine, an employee of O and member of the project-group "Profit" in SSU "Big-sales", wishes to make use of her privilege to use the Application. Mrs Fine logs on to the system using her smart card and a password. Authentication is

performed and a number of icons, representing different application clients, are presented on the GUI of the user.
2. An SSL connection is negotiated and set up between the Desktop and AMANDA.
3. Using the SSL connection the Desktop requests a PAC from AMANDA.
4. AMANDA checks all ACLs in the database. For all object-level permissions to different applications that Mrs Fine has, the corresponding log-on information is retrieved. (in the SS these log-in information is referred to as database-credentials) All log-on information is contained within a PAC that is issued.
5. The PAC is sent to the Desktop of Mrs Fine and this SSL-connection is closed.
6. On the GUI of the Desktop, Mrs Fine clicks on the icon associated with the Application.
7. A new SSL-connection is negotiated and set up between the Desktop and the Agent.
8. Mrs Fine's log-on information for the Application is forwarded to the Agent that uses them to log Mrs Fine on to the Application.
9. Mrs Fine may now use the Application to search and read, using any of the 5 GUIs. For the remainder of the session the SSL-connection is kept and all communication between the Client and the Application is mediated through the Agent.
10. An SSL-connection between the Agent and AMANDA is also set up and is used to forward logs to the Event Logging System in the AMANDA.
11. Mrs Fine disconnects from the Application and the SSL-connection between the Desktop and the Agent is closed.
12. Mrs Fine logs off from the Application.

Summing up use-case 5: Besides differences due to using PAC-based operations instead of Bridge-based operations, this use-case does not differ from the events in use-case 2. Object-level permissions are put in ACLs to allow fast and easy answers when called upon and it is not desired for this to deviate from the traditional ACL in use-case 2.

Also, the default meaning of a permission is the same as in use-case 2. With the possibility to express other permissions than traditional ones, like read, write and execute, it would of course be possible to do so. This has been commented above under "Application Policy (AP)."

The permissions that will be requested will be known in advance and expressed in the ACL. The fact that simple look-up tables are used for the large bulk of questions likely to be asked speeds up the system.

5. DISCUSSION AND CONCLUSION

In this chapter the results from the comparison in the case study, between the AM-process of today and that of the future, are discussed and conclusions presented. The chapter also criticises the thesis out of some perspectives and suggests areas of study that needs to be addressed further.

The comparison between the actual AM-process of today and the theoretical solution offered by AMANDA is abstract. However, assuming that AMANDA will function as proposed the AM-process would clearly be simplified. The AM-process today, to a large extent, consists of manual real world routines. These manual routines are sub-processes in the overall AM-process and their only purpose is to enable the implementation of an authorisation-decision in the computerised control-mechanism. The AM-process of today does not add any value to the Organisation. It seems to exist simply because there are no alternatives. Using AMANDA these slow manual routines will be eliminated.

The AM-process is considered to start when a decision is made to delegate an authorisation and ends when that decision has been implemented. Today the AM-process normally takes several days. Eliminating the manual routines would speed up the AM-process to practically real-time. At the same time the decision is made it is also implemented into the computerised system.

The decision to delegate an authorisation is today already decentralised according to the chain of command of O. Implementation of the decision is however centralised to certain administrators. The administrators belong to O but the structure and chain of command of the IT organisation differs from the operational structure and chain of command. According to real world policies the system-owner is supposed to control the decision's compliance with policies before implementing it. In reality this is not the case. Even though the system-owners possess the proper domain knowledge to perform the control, they do not have the time needed. If the decision-maker implemented his own decision the AM-process would be truly decentralised. But not only would both the decision and the implementation map the operational chain of command, it would also simplify control and be easier to monitor. CERT keeps all ACs that have been issued and this simplifies detective control. Further, being able to more precisely express both the type of privileges and how they may be delegated serves as a preventative control mechanism. Also, the system-owner could be relieved of his control-mission and the administrators could do other things than implement the decisions of others.

The real world policies of today state that the privileges delegated to the employees of strategic partners differ from those delegated to employees of O. Today it is not possible to control this through the computerised control-mechanisms and therefore an object-level permission always is the same. Being able to express several different permissions in the computerised control-mechanism would enable the differentiated usage stated by the real-world policies.

Every time a user wants to use the system it would be possible for the user to present, or refer to, an AC that delegate the privilege called upon. However, then the engine would have to reason about the compliance of the certificate with policies and presently valid facts every time. In a large organisation the workload of the engine would be great and it is likely the

large bulk of requests will be predictable. Most will concern object-level permissions such as searching and reading the Application. By adopting the ACL look-up approach for answering the large bulk of questions the workload of the system is eased and the process is speeded up. A similar advantage, but with respect to the predicted large bulk of requests to use management-powers, is achieved by use of FACT. Continuously updated FACTs and ACLs also enable control and easy overview over privileges.

Viewing everything from interpreting policies to implementing decisions as one AM-process may seem natural when discussing models and mechanism for AM. Today, however, AM is discussed in terms of existing models and mechanisms and focus is on the IT-world. This is remarkable as only a small part of the full AM-process today concerns the IT-world. Most concern the real world and routines that have to be performed there. Considering this, it is not surprising to find that no unified approach towards analysing, modelling, understanding and describing the full AM-process exists. Developing an approach towards modelling the full AM-process is not part of the problem statement of this thesis. It was however necessary to do this before being able to reason about the benefits of AMANDA through a "today" vs. "in the future" case study.

Combining Enterprise Modelling and UML in order to be able to analyse, model, understand and describe the AM-process has been an accessible approach towards modelling the AM-process.

Modelling the functionality of AMANDA as proposes in this thesis indicates that more of the real world has to be formalised and expressed in the computerised system than just the AP-policy derived from real-world policies. Not all that needs to be expressed can be found in real world policies. For instance, policy B states that a user must "posses knowledge to use the focal IT-system". In order for an expressive and dynamic authorisation-controlling mechanism to consider this, knowledge has to be formalised and expressed. In the case study the organisational structure, different nodes and relations between these nodes are expressed in general terms such as roles, functions and units. This enables a great deal of flexibility but also increases the complexity of the system.

How generic and flexible a computerised mechanism for AM will be is related to the complexity of the actual instance of the mechanism. The engine will reason about the contents of an AC. The more complex the expression is in the AC the more complex will the process of checking compliance with policies and presently valid facts be. Answers to all questions the engine will ask must be represented within the IT-world. Even if a computerised mechanism can represent this, a very demanding task will be identifying, modelling and formalising all relevant aspects in the real world. As for now no approach exists towards doing this. Of course, the complexity of all this will depend on the complexity of what can be represented in the IT-world.

5.1. SELF-CRITICISM

Organisations differ in many ways but the models and mechanisms used for authorisation management are basically the same. No radical and widely used alternatives seem to exist. This indicates that the AM- and usage-processes of other organisations, in other settings are similar, and that they too would benefit from using an AMANDA-mechanism. However, only one application and organisation, in a specific setting, has been examined. The use-cases

represent only a few processes. Use-cases describing other processes, perhaps in other organisations and settings, would be different. This must be kept in mind.

Even though the drawbacks identified through the case study and the solutions offered by AMANDA are intuitive, the approach has been deductive, I started out with the proposed benefits of AMANDA in mind. Modelling and describing the AM-and usage-processes in a more inductive way, without having an opinion on what to look for and find, the identified drawbacks and benefits of both existing models and AMANDA might have been different.

5.2. FUTURE WORK

In AMANDA computerised authorisation-controlling mechanisms are to represent real world policies. In order for this to function different aspects of the real world have to be considered, modelled and formalised in terms relevant for AMANDA. In this thesis enterprise modelling and UML have been combined and used in order to do this. Being able to describe real world policies and aspects has to be possible to do in terms common to the real world and the computerised mechanism. Finding an approach to model and map the real world into the IT-world is an important issue that have to be further addressed.

If somewhat computer-knowledgeable employees in an organisation are to implement their own decisions directly into a computerised system, their interface with the computerised system has to be easy to understand and use. The development of a suitable user-friendly human-computer interface is an issue that has to be given a great deal of consideration.

Some sort of interface also has to be developed in order to set and alter the MP and the AP.

In theory PKIs can be cross-certified in order to inter-operate. However, the case study has shown that, at least in one case, this is not true in reality. Being able to interact trustingly over large, open and closed, networks implies the need of PKIs. It would be interesting to study how and why PKI-solutions offered by different vendors cannot inter-operate and more interestingly, how they can be made to.

6. REFERENCES

Sadighi Firozabadi, B., Sergot, M., Bandmann, O., (2001) ”Using authority certificates to create management structures”. To be published in *Proceedings of Security Protocols – 9th International Workshop Cambridge, April 2001. Lecture Notes in Computer Science.* Springer.

Blaze, M., Feigenbaum, J., Lacy, J. (1996) ”Decentralised trust management”. *Proceedings of the 17th Symposium on Security and Privacy.* Pages 164-173, IEEE Computer Society Press, Los Alamitos.

Blaze, M., Feigenbaum, J., Ionnidis, J., Keromytis, A. (1999a) *The KeyNote Trust Management System Version 2*. Request for Comments: 2704. In January 2001 available online: http://www.ietf.org/rfc/rfc2704.txt?number=2704

Blaze, M., Feigenbaum, J., Ionnidis, J., Keromytis, A. (1999b) ”The Role of Trust Management in Distributed Systems Security”. *Secure Internet Programming: Security Issues for Mobile and Distributed Objects.* Springer-Verlag.

Blaze, M. (2001) ”Trust Management, Compliance Checking & Security Policy” Power point slides. Invited talk in Bristol, UK. In April 2001 available online: http://www.crypto.com/talks/blaze-trust.pdf

Chokhani, S., Ford, W. (March 1999) *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework.* Request for Comments: 2527. In January 2001 available online: http://www.ietf.org/rfc/rfc2527.txt?number=2527

Eisenhardt, Kathleen M. (1989) ”Building theories from case study research”. *Academy of Management Review,* Volume 14, No 4, 1989, s. 532-550.

Ellison, C., (September 1999) *SPKI Requirements*. Request for Comments: 2692. In March 2001 available online: http://www.ietf.org/rfc/rfc2692.txt?number=2692

Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Ylonen, T. (May 2001) *SPKI certificate theory*. Request for Comments: 2693. In March 2001 available online: http://www.ietf.org/rfc/rfc2693.txt?number=2693

Farrell, S., Housley R. (June 2001) *An Internet Attribute Certificate Profile for Authroization <draft-ietf-pkix-ac509prof-06.6>*. Internet-draft. In June 2001 available online: http://search.ietf.org/internet-drafts/draft-ietf-pkix-ac509prof-09.txt

Feghhi, Feghhi, Williams (1999) *Digital Certificates – Applied Internet Security.* Addison Wesley Longman, Inc.

Ferraiolo, D., Kuhn, R. (1992) ”Role-Based Access Control”. *Proceedings of the 15th national Computer Security Conference, 1992.* In June 2001 available online: http://hissa.ncsl.nist.gov/rbac/paper/rbac1.html

Gollmann, D. (1999) *Computer Security.* John Wiley & Sons Ltd.

Housley, R., Ford, W., Polk, W. (January 1999) *Internet X.509 Public Key Infrastructure Certificate and CRL Profile.* Request for Comments: 2459. In January 2001 available online: http://www.ietf.org/rfc/rfc2459.txt?number=2459

Larman, C. (1998) *Applying UML and Patterns. An introduction to Object-Oriented Analysis and Design*. Prentice-Hall.

Nikander, P., Metso, L. (September 2000) ”Policy and Trust in Open Multi-Operator Networks”. *Proceedings of IFIP Smartnet' 2000*. Kluver Academic Publishers

Pfleeger, Charles, P. (1996) *Security in Computing*. Second edition. Prentice Hall.

RSA Keon. (2000c) ”Introduction to RSA Keon Agents” RSA manual. RSA Security Inc.

Sadighi Firozabadi, B., Sergot B. (1999) ”Power and Permission in Security Systems” *Proceedings of Security Protocols – 7th International Workshop Cambridge, UK*. Pages 48-53. *Lecture Notes in Computer Science 1796.* Springer.

SIG Security (1999) *SIG Security 1998. Säkerhetsarkitekturer.* Studentlitteratur, Lund.

SIG Security (2000) *SIG Security 1999. Elektronisk affärssamverkan mellan företag.* Studentlitteratur, Lund.

Sjölin, M. (2000) *The Concept of Trust.* Master's Thesis, number 00-63. Department of Computer and Systems Science. Stockholm University.

Willars, H. (1993a) *Modelleringsledarens bashandledning.* TRIAD-report number 10:2. All TRIAD-reports published by former SISU, today called Swedish Institute of Computer Science (SICS)

Willars, H. (1993b) *Modelleringsväskan.* TRIAD-report number 10:6. All TRIAD-reports published by former SISU, today called Swedish Institute of Computer Science (SICS)

Wohed, R., Öhlund, S-E. (1993) *Regelmodellering i praktiken.* TRIAD-report number 10:9. All TRIAD-reports published by former SISU, today called Swedish Institute of Computer Science (SICS)

Wohed, R. (1997) *A Language for Enterprise and Information System Modelling*. Akademitryck AB, Edsbruk.

ONLINE

http://www.whatis.techtarget.com

RSA Keon. (2000a) ”RSA Understanding Public Key Infrastructure (PKI)”. Published online: http://www.rsa.com/products/keon/index.html Technology White Paper. Available December 20, 2000.

RSA Keon. (2000b) ”RSA Keon Advanced PKI”. Published online: http://www.rsa.com/products/keon/index.html Solution White Paper. December 24, 2000.

RSA Keon. (2001) ” RSA Laboratories' Frequently Asked Questions About Today's Cryptography, Version 4.1” Published online: http://www.rsa.com/rsalabs/faq/ February 2001.

APPENDIX 1

KEON[i]

Keon is a vendor-specific PKI solution provided by RSA Security. It consists of several components that together make up or may be used within a PKI. The components are modularly designed. The design is intended to allow interoperability with standard-based PKI components provided by other vendors. The functionality offered varies depending on what modules are used and their configuration.

COMPONENTS

The components of main interest for AM and usage of the Application are the Desktop, the agent, the security server, the management console and aspects associated with these components. The components and their relations are shown in figure 1. Appendix 1 and are described more in detail below.

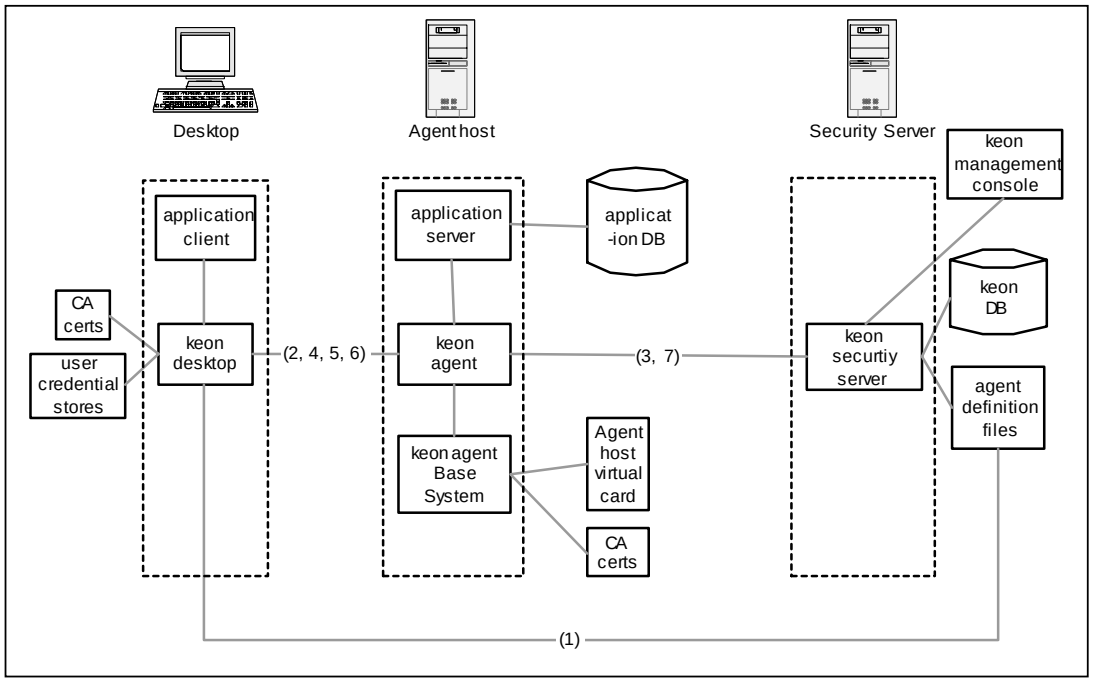

Figure 1. APPENDIX 1: Keon components and component interaction.[ii]

[i] The component description is based on white papers, technical specifications, administrators guides issued by RSA Security and interviews.
[ii] RSA Keon (2000c)

APPLICATION CLIENT

One or several Application clients run on the Desktop and must use TCP/IP to communicate with the Application server. If the user is authorised to log on to several accounts on the Application server a *role name* has to be specified by the user. The Agent will forward this information to the SS where the necessary Application-specific log on credentials will be returned.

KEON DESKTOP

The Desktop provides the interface through which a user logs on and is authenticated and provides line encryption with Agents. That is, an Application-level Virtual Private Network is established with an Agent using an encrypted and authenticated channel based on standard PKI protocols. A standard Microsoft or Netscape web browser is needed to communicate with the SS to retrieve user certificates via an Agent. On each Desktop different CA certificates must be present so that the signature of the user certificate stored in the user's credential store can be verified. A CA certificate is also needed to verify Agent Host certificates. Using the Desktop Communication Protection module the Desktop intercepts communication, using fixed ports, from an Application Client to an Application Server and will provide line encryption and strong user authentication with Agents.

APPLICATION SERVER

The Application server runs on the Agent host. To avoid the Agent being bypassed, the Application server should be configured to only accept connections from the machine where the Application Server is running, that is, the local host. With this configuration, there can be no secondary connections from the server back to the Application. The user is authenticated to the Application Server by presenting credentials that the Desktop, via the Agent, previously has retrieved from the SS or from a PAC cached on the Desktop being used. The Agent may intercept the log on procedure if the Agent knows the protocol used between the Application server and the Application client, or if it is possible for the Agent to extract the username and password from the communication between the two.

THE CREDENTIAL STORE

Each user has a Credential Store, contained on a Virtual or real Smart Card and signed by a trusted CA. The smart card is used to log the user onto the Desktop where the Desktop performs strong authentication of the user by challenging the digital certificate presented from the Smart Card of the user. The certificate contains information such as public and private keys of the user, validity period and possibly other information depending on who issued the credential store. What is important is that it contains a digital certificate that uniquely identifies the user. This certificate has a signature that can be verified as originating from a CA considered a trusted third party within the PKI. Or, for that matter, between PKIs if they have been cross certified.

CREDENTIALS

The basic operation of the Agent is to authenticate the user and then retrieve the credentials needed to log the user on to the Application at hand. Three sets of credentials are used*: user-credentials* (UC), *given-credentials* (GC) and *DB-credentials* (DBC). UC contain user-specific information originating from a user's credential store and are forwarded by the Desktop to an Agent and from the Agent to the SS. GC and DB are stored in the SS.

For each UC that can be entered there exists a corresponding DBC in the SS. For each GC there exists a corresponding DBC. The DBCs are the credentials returned to the Agent and by the Agent forwarded to the Application server that use them to log the user on to the Application. The credentials needed to log a user on to an Application vary depending on the behaviour of the Application server.

When a user logs on to an Application, one of three user-credential scenario may occur: *One-to-one*: a user is assigned one set of credentials such as username and password. *Many-to-one*: several users share the same set of credentials. *One-to-many*: A user is assigned several sets of credentials. In the first two scenarios the Agent, acting on behalf of a user, only has access to one set of credentials for a given access method.

For each scenario a mechanism must exist by which the user can indicate which set of credentials are to be used for a given session. This is achieved by establishing different roles for users. The user logs in using credentials specific for a user role. These user role credentials are then used to find the corresponding credentials in the SS. This is called *user role mapping*. For instance: The user forwards his UC (through the Desktop and the Agent). If the user can take on different roles in the Application, such as "user" or "administrator", the user also has to specify in which role he wishes to act. The UC are then matched to the specified user role, the DB-credentials are retrieved and returned to the Agent etc. A set of credentials has values, e.g. in Keon SS 4.5 the set of credentials can comprise of up to six values.

KEON AGENT

The Agent controls user rights to access Application Servers by retrieving credentials from the SS and logging the user on to the Application Server. This can be done using Privilege Attribute Certificates (PAC) in which all credentials of a user's, for different applications, are retrieved at the same time or it may be done once each time the user logs on to a new Application Server. The Agent further negotiates encryption and authentication protocols with the Desktop and also sends audit logs to the SS. Several Agent instances may reside on the same host where each Agent mediates or filters all communication between a specific Application Client, residing on the Desktop, and its associated Application Server. Each Agent is configured or uniquely developed to suit a specific Application. When an Agent is configured, the port to which the Application Server listens and the port to which the Agent listens are specified. To separate several instances of the same Agent they are assigned unique names called *access method (names).* An access method defined on an Agent instance must also be defined in the SS. This is done to grant users access to the Application Server protected by an Agent with the same access method.

Each Agent installed has a set of Agent Definition Files (ADF) associated to it. An ADF define the Agent service name together with a definition of the set of credentials that the Agent needs to log a user on to different Application Servers (for instance username and password). These credentials are called *user access rights* and their characteristics as well as the set of credentials can vary from Agent to Agent. A *service name* is assigned each Agent implementation. The Agent service name comprises a symbolic name used to identify the Application protected by the Agent and a version number. The Agent service name is used to install the ADF in the SS as the Agent Server module.

KEON AGENT BASE SYSTEM

Residing on each Agent host is also an Agent Base System. Several Agents residing on the same host use the same Base System. The Base System contains dynamically linked libraries, configuration parameters, data and various programs.

When the Agent host is defined in the SS, the SS creates an Agent host Virtual Card that is imported to a file and installed in the Base System. Agents residing on the same host use a common virtual card when authenticating to Desktops.

The Agent host Virtual Card is protected by a password, a symmetric key called the node key, installed in both the Base System on the Agent host and on the SS. When operations are Bridge-based, the node key is also used to encrypt communication between the Agent host and the SS.

The Base System is also used to specify the host name and port number of the SS and the location of the Event Logging System Server. Trusted CA certificates are installed in the Base System and during Bridge-based operations each Agent instance use these to verify the signature of user certificates when users connect to the Agent.

KEON SECURITY SERVER. BRIDGE, - AND PAC-BASED OPERATIONS

Each user must be registered in the SS together with the access control information defining user authorisations. The SS corresponds with Agents using PAC-based or Bridge-based operations. The SS also manages the central audit log and provides certificate revocation information. The Keon SS is administrated through use of specific Management consoles handled by designated administrators.

A *PAC-based operation* uses Privilege Attribute Certificates based on the Security Socket Layer (SSL) protocol. A PAC is a form of X.509 v3 certificate. It is similar to a digital certificate like the one stored on a smart card. Unlike this certificate, however, the PAC specifies the user's rights within the issuing system. A PAC may be issued by, or an already existing PAC downloaded from, the SS to the Desktop where it is kept for the remainder of the time the user is logged in. Another option is for the Desktop, once a user has downloaded a PAC, to cache the PAC so that it will be there the next time the same user logs on to the same Desktop. When the user wishes to access an Application the necessary authorisation information is retrieved from the PAC and forwarded to the Agent by the Desktop. This eliminates the need to check authorisations through the SS every time an action is requested from a new Application. All necessary information about a user is downloaded to the Desktop

upon login. The lifetime of a PAC is configurable and it is typically short, such as a few hours or the duration of a normal working day.

Using PAC-based operations the communication between the SS and the Desktop is not mediated by the Agent. Instead the Secure Socket Layer (SSL) is used to forward PACs directly from the Security Server to the Desktop (1).[iii] When a user requests to use an Application, SSL is set up and used between the Desktop and the Agent to forward log in information to the Agent who will use it to log on to the Application Server (2). The Agent also uses SSL to send entries to the Event Logging System in the SS (3).

In a *Bridge-based operation* the Agent retrieves the users Application specific credentials from the SS every time a new action is requested from an Application server. In Bridge-based operations the DASP-protocol is used for protocol negotiation between the Desktop and the Agent (4). Between the Desktop and the Agent the ALLTAK-protocol is used for session encryption (5) and the ALLTAAS-protocol is used for strong authentication (6). Communication between the Agent and the SS is encrypted using the symmetric node key that has been installed on both the Agent and the SS (7).

MANAGEMENT CONSOLE

With Keon SS authorisations are managed through a management console. In Keon SS versions 4.5 this basically functions in such a way that the administrator, through the Management console, specifies what administrative action he wishes to perform. Different services are assigned different ports on the SS and basically different forms are coupled to each port. The requested form is filled out or altered by the administrator before being returned to and implemented in the SS.

In newer versions of the SS, such as version 5.5, the management console is a Java Application that can run on the same machine as the SS or from a remote Desktop. Java classes provided by RSA Security are used to manage objects in the SS. Designated administrators uses the Management Console for database maintenance tasks as adding and modifying user records and access rights information as well as for managing the list of trusted certificates. The administrator authenticates to the SS without the intermediate layer provided by an Agent. The communication between the management console and the SS is secured by using SSL.

[iii] Numbers within parenthesis refer to numbers in the picture ”Keon components”