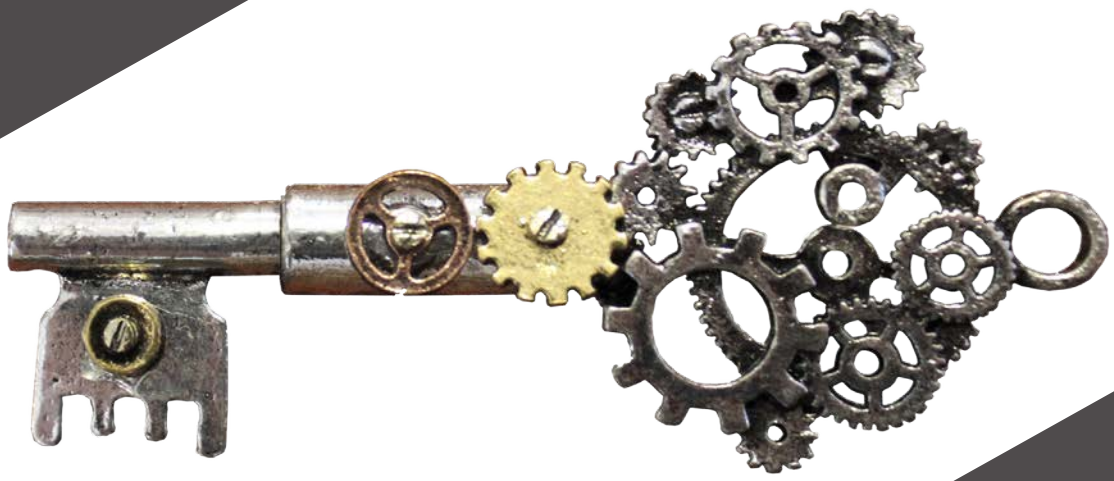
**Puolustusvoimien tutkimuslaitos**
Julkaisuja 8

# Enforcing Role-Based Access Control with Attribute-Based Cryptography in MLS Environments

Mikko Kiviharju



Puolustusvoimien tutkimuslaitoksen julkaisuja 8

# ENFORCING ROLE-BASED ACCESS CONTROL WITH ATTRIBUTE-BASED CRYPTOGRAPHY IN MLS ENVIRONMENTS

(2nd edition)

Mikko Kiviharju

PUOLUSTUSVOIMIEN TUTKIMUSLAITOS
FINNISH DEFENCE RESEARCH AGENCY

RIIHIMÄKI 2017

Kannen layout: Onni Pernu
Kannen kuva: Mikko Kiviharju

ISBN 978-951-25-2883-7 (painettu)
ISBN 978-951-25-2884-4 (verkkojulkaisu)
ISSN 2342-3129 (painettu)
ISSN 2342-3137 (verkkojulkaisu)

Puolustusvoimien tutkimuslaitos
Finnish Defence Research Agency

Juvenes Print
Tampere 2017

# Foreword

This book is the second, extended edition of the work that was submitted as a doctoral dissertation in spring 2016 by the same author. The first edition was written to a slightly different focus in the thesis format, which does not serve the topic itself in the best possible manner. Additionally, the limited number of copies of the first edition and some IPR issues spoke for an extra edition. The focus in this work is not so much on the contributions in the thesis publications, as it is on the main topic of pervasive cryptographic access control (CAC), related cryptographic schemes and their properties.

The main changes and additions between the editions include: shift of focus from thesis publications to the actual concept of pervasive CAC; an additional definitional discussion related to functional encryption and its challenges; a more in-depth view of the meaning of pervasive cryptographic protection of objects and concepts (including, for example, the effect and meaning of blockchains in this context); and more recent simulation results and actual performance numbers from one of the central concepts mentioned here, NATO OLP.

The background work leading to this book and dissertation has been ongoing, with different intensities, for approximately ten years. It is gratifying to watch some initial ideas to mature towards something actually usable, as well as see surprising applications of techniques that were intended for something completely different. These unexpected applications include the idea of attribute-based encryption, originally intended to be identity-based encryption for biometrics, and the use of functional encryption in program obfuscation. It is equally rewarding to see the same principal ideas rise independently from other research or innovations, such as the idea of giving the distributed network almost full responsibility of uncorrupted content availability (e.g. Protected Core Networking and blockchains).

Much of this work would not have been possible without the support from Finnish Defence Forces Technical Research Centre (later Defence Research Agency), and I’d like to express my gratitude towards my closest colleagues inside the Cyber Defence branch of our facility. My special

thanks go also to my dissertation supervisor, professor (emerita) Kaisa Nyberg from Aalto University, and my dissertations reviewers and opponents, professor Reihaneh Safavi-Naini from the University of Calgary, professor Colin Boyd from Technical University of Trondheim and Dr. Konrad Wrona from NATO CI Agency. Without forgetting all the personal support, I am also most grateful to my family and friends.

Riihimäki, January 6th, 2017

Mikko Kiviharju

# Abstract

Access control in computer science defines how different active processes, called subjects, may perform abstract operations on (computing) resources, called objects. General access control enforcement includes a theoretical construct called a reference monitor, which is intended to monitor the access requests between subjects and objects. This dissertation researches the possibilities to replace reference monitors with cryptography, for reasons of implementation-level assurance and distribution of computation. An information security notion called multi-level security (MLS) binds official data confidentiality levels to trustwor- thiness of users such that, for example, users checked ("cleared") for some level should be able to securely access information classified up to their level, inside a system which also contains information classified to a higher level. Traditionally, only cryptography has been considered to have sufficient assurance for large scale MLS environments. However, cryptographic enforcement is rather rigid and limited in some respects. Ideally, cryptographically enforced access control should comply with modern access control management principles such as role-based access control (RBAC). Recent advances in public key infrastructure (PKI), such as attribute-based encryption (ABE) and signatures (ABS), enable more complex policies in access control as well. This dissertation investigates the possibilities to use ABE and ABS in enforcing access control cryptographically, according to modern RBAC principles. The main application we target is publish-subscribe environment for MLS documents. As ABE and ABS represent only one type of PKI authentication architecture, and attributes are elemental for RBAC support, we first research the question, whether the capability to support attributes in general is particular to the authentication architecture represented by ABE, and find that this is not the case. However, due to other benefits of the ABE type, we find that they are still superior to other types. We then present the main assumptions to our application environment and show, how XML-documents can be used to support the access control enforcement cryptographically and nevertheless allow a transition period from conventional PKI to ABE. The actual framework consists of a general model on how to represent different access operations, or permissions, in such a way that they can be cryptographically enforced, as well as XACML reference architecture-based models for implementing confidentiality and integrity policies using ABE and ABS, respectively. We also map different NIST-standardized RBAC-model elements to ABE and ABS functionalities. In the confidentiality enforcement model we note a controversy in the ABE security goal of user collusion prevention with MLS environment requirements, and introduce a scheme to overcome this securely.

**Keywords**: MLS, RBAC, ABAC, ABE, ABS, functional encryption, multi-level security

# Contents

# List of Abbreviations

| | |
|---|---|
| AA | Attribute Authority |
| AACS | Advanced Access Content System |
| ABAC | Attribute-Based Access Control |
| ABE | Attribute-Based Encryption |
| ABGS | Attribute-Based Group Signature |
| ABS | Attribute-Based Signature |
| AC | Access Control |
| ACDT | Advanced Concept and Technology Demonstrator |
| ANSI | American National Standards Institute |
| AR | Administrative Role |
| ARBAC | Administrative Role-Based Access Control |
| BLP | Bell-laPadula (security model) |
| BSW-SIM | Boneh-Sahai-Waters Simulation based security model |
| CA | Certificate Authority |
| CAC | Cryptographic Access Control |
| CAC-S | CAC for data in Storage |
| CAC-ST | CAC for data in Storage or in Transit |
| CAC-T | CAC for data in Transit |
| CBIS | Content-Based Information Security |
| CBPS | Content-Based Publish-Subscribe system |
| CC | Constructive Cryptography |
| CCA | Chosen Ciphertext Attack |
| CFE | Composable Functional Encryption |
| CH | Context Handler |
| CL-PKE | Certificateless Public Key Encryption |
| CMHP | Coalition MLS Hexagon Prototype |
| CP | Ciphertext-Policy |
| CP-ABE | Ciphertext-Policy Attribute-Based Encryption |
| CP-FE | Ciphertext-Policy Functional Encryption |
| CPA | Chosen Plaintext Attack |
| CPR | Content-based Protection and Release |
| CPRESS | CPR Enforcement Separation Service |
| CPRM | Content Protection for Recordable Media |
| CPPM | Content Protection for Pre-recorded Media |
| CRBAC | Cryptographically enforced Role-Based Access Control |
| CRS | Common Reference String |
| DAC | Discretionary Access Control |
| DFS | Delegatable Functional Signature |
| DK | Derived Key (KPD-ABE) |

| | |
|---|---|
| DMA-ABS | Decentralized Multi-Authority Attribute-Based Signature |
| DMA-FE | Decentralized Multi-Authority Functional Encryption |
| DNS | Domain Name Service |
| DOGA | Deny-Or-Grant-Access |
| DRM | Digital Rights Management |
| DSD | Dynamic Separation of Duty |
| EAPK | Encrypted Attribute Private Key (KPD-ABE) |
| EC | Elliptic Curve |
| EHASH | Encrypted agent HASH (KPD-ABE) |
| Exp | Experiment (in cryptographic security notions) |
| FAF | Flexible Authorization Framework |
| FAF ASL | FAF Authorization Specification Logic language |
| FC-PKC | Fuzzy Certificateless Public Key Cryptography |
| FE | Functional Encryption |
| FHE | Fully Homomorphic Encryption |
| FIBE | Fuzzy Identity-Based Encryption |
| FS | Functional Signature(s) |
| GID | Globally unique IDentity |
| GM | Generic group Model |
| HKAS | Hierarchical Key-Assignment Scheme |
| HP-UX | Hewlett-Packard Unix |
| IBBE | Identity-Based Broadcast Encryption |
| IBC | Identity-Based Cryptography |
| IBE | Identity-Based Encryption |
| IBKE | Identity-Based Key Insulated Encryption |
| IBM | International Business Machines |
| IBS | Identity-Based Signature |
| IA | Identity of attributes (ABS survey) |
| ICS | Implicit Certification Schemes |
| ICT | Information and Communication Technology |
| IFC | Information Flow Control |
| INCITS | InterNational Committee for Information Technology Standards |
| IND | Indistinguishability, (a cryptographic security goal) |
| IND-CPA | Indistinguishability under Chosen Plaintext Attack |
| IPAS | Inner-Product Access Structure |
| $I_s$ | Identity of signer (ABS survey) |
| ISR | Instruction Set Randomization |
| K | Private Key (ABS survey) |
| KGC | Key Generation Center |
| KP | Key-Policy |
| KP-ABE | Key-Policy Attribute-Based Encryption |

| | |
|---|---|
| KPD-ABE | Key-Pooling Decentralized Atrtibute-Based Encryption |
| KPF | Key-Processing Facility / Function |
| KSS-IBKI | Kumar-Shalaija-Saxena Identity-Based Key Issuing |
| LW-ABE | Lewko-Waters Attribute-Based Encryption |
| MA | Multi-Authority |
| MAC | Mandatory Access Control |
| MLS | Multi-Level Security / Multi-Level Secure |
| MPR-ABS | Maji-Prabhakaran-Rosulek Attribute-Based Signature |
| MS | Microsoft |
| NATO | North Atlantic Treaty Organization |
| NC | Nick's Class (computational complexity class) |
| NIST | National Institute of Standards and Technology |
| NIWI | Non-Interactive Witness Indistinguishable (proof) |
| NIZK | Non-Interactive Zero Knowledge |
| NIZKAoK | Non-Interactive Zero Knowledge Arguments of Knowledge |
| NM | Non-Monotonic |
| NM-ABS | Non-Monotonic Attribute-Based Signature |
| NM-CP-ABE | Non-Monotonic CP-ABE |
| NM-KP-ABE | Non-Monotonic KP-ABE |
| NNL | Naor-Naor-Lotspiech (a key assignment scheme) |
| NTK | Need-to-Know |
| OA | Object Attributes (in ABAC) |
| OASIS | Organization for Advancement of Structured Information Standards |
| OBJ | Object (in RBAC), Object-subdomain (CRBAC arch.) |
| OLP | Object-Level Protection |
| OPS | Operations (in RBAC) |
| OWE | One-Way Encryption |
| P | Permission (RBAC), Policy (ABS-survey) |
| PA | Permission Assignment (RBAC), Pooling Authority (KPD-ABE) |
| PAP | Policy Administration Point |
| PBS | Policy-Based Signature |
| PDP | Policy Decision Point |
| PE | Predicate Encryption |
| PE-NDS | Predicate Encryption for Non-Disjoint Sets |
| PEP | Policy Enforcement Point |
| PHB | Publisher Hosting Broker |
| PIP | Policy Information Point |
| PKAA | Public-Key Authentication Architecture |
| PKI | Public-Key Infrastructure |

| | |
|---|---|
| PKC | Public-Key Cryptography |
| PoC | Proof-of-Concept |
| PRE | Proxy Re-Encryption |
| PRNG | Pseudo Random Number Generator |
| PS | Policy Store |
| R | Role (in RBAC) |
| R-ABS | Revocable Attribute-Based Signature |
| RBAC | Role-Based Access Control |
| RFC | Request For Comments |
| RH | Role Hierarchy |
| RKH | Role Key Hierarchy |
| RM | Reference Monitor |
| RO(M) | Random Oracle (Model) |
| RSA | Rivest-Shamir-Adleman |
| RTK | Right-to-Know |
| S | Session (RBAC), Signer (ABS survey) |
| SA | Subject Attributes (in ABAC) |
| SAML | Security Assertion Markup Language |
| SHB | Subscriber Hosting Broker |
| SIG | Signature subdomain (CRBAC architecture) |
| Sim | Simulator (in security proofs) |
| SOAP | Simple Object Access Protocol |
| SP | Special Publication (in NIST standards) |
| SPL | Security Policy Language |
| SoD | Separation-of-Duty |
| SQL | Sequential Query Language |
| SRM | Security-Related Metadata |
| SSD | Static Separation of Duty |
| T | Terminal (KPD-ABE) |
| TCB | Trusted Computing Base |
| TCSEC | Trusted Computer System Evaluation Criteria |
| TG | Terminal Group |
| TPM | Trusted Platform Module |
| $T_s$ | Signature Trustee (ABS survey) |
| U | User (RBAC, KPD-ABE) |
| UA | User Assignment |
| UK | United Kingdom |
| USA | United States of America |
| USR | User sub-domain (CRBAC architecture) |
| V | Verifier (ABS survey) |
| VA | Verifiable Attributes |
| VRF | Verification subdomain (CRBAC architecture) |

| | |
|---|---|
| W3C | World Wide Web Consortium |
| WSML | Web Service Modelling Language |
| XACML | eXtensible Access Control Markup Language |
| XML | eXtensible Markup Language |
| XOR | eXclusive OR operation |
| XSD | XML Schema Definition |
| ZVH-RBE | Zhou-Varadharajan-Hitchens Role-Based Encryption |

# List of Symbols

| | |
|---|---|
| $\wedge$ | Logical connective, conjunction |
| $\vee$ | Logical connective, disjunction |
| $\neg$ | Logical connective, negation |
| $\Rightarrow$ | Logical connective, implication |
| $\forall$ | Logical quantifier, universal |
| $\exists$ | Logical quantifier, existential |
| $\cup$ | Set union |
| $\cap$ | Set intersection |
| $\subset$ | (Proper) subset operator |
| $\subseteq$ | Subset operator |
| $\in$ | Set membership |
| $\in_U$ | Set membership, element selected unif. randomly from a set |
| $\notin$ | Negated set membership |
| $\rightarrow$ | Mapping |
| $\leftarrow$ | Assignment |
| $\overset{\$}{\leftarrow}$ | Assignment with sampling from a uniformly random distr. |
| $\times$ | Cartesian product |
| $2^S$ | Power-set of set S |
| $\{0,1\}^*$ | The set of arbitrary-length binary strings |
| $\|S\|$ | Number of elements in set S; impl. size of function |
| $\mathbb{N}$ | Set of natural numbers, {0,1,2,3,…} |
| $\mathbb{Z}_+$ | Set of positive integers {1,2,3,…} |
| $\mathbb{Z}_p$ | A cyclic quotient group of $\mathbb{Z}$ with size $p$ = prime. |
| **NC** | Complexity class **NC** (“Nick’s Class”) |
| $\mathbf{NC^i}$ | Complexity class $\mathbf{NC^i} \subseteq \mathbf{NC}$ |
| $\succcurlyeq$ | Partial order, inheritance relation (RBAC) |
| $\perp$ | Error-symbol (output by a scheme at an error occurrence) |
| $1^n$ | $n$-string of ones (“11…11”) |
| $\bar{v}$ | Vector $v$ |

| | |
|---|---|
| $O(\cdot)$ | The “Big-Oh”-notation |
| $(\cdot)^T$ | Transpose |
| $\log^i n$ | Logarithm of $n$ raised to the power $i$ |
| $e(\cdot)$ | Bilinear map e |
| $\Delta_{i,S}(x)$ | Lagrange coefficient (a type of polynomial) |
| $[\![\cdot]\!]$ | Updateable, shared variable (used to convey status information in, for example, security definitional processes) |
| $\epsilon$ | Empty key or message (in FE) |
| $\oplus$ | Bitwise exclusive OR (XOR) |

# 1. Introduction

## 1.1 Motivation and Problem Statement

The current, tightly connected, highly dynamic and distributed computing environment presents large challenges for information security. Entities with the most sensitive assets and the most challenging protection requirements struggle daily to keep their information systems secure against both inside and outside threats. Typical such examples include government, financial institutions and healthcare.

Early models developed for the most valuable assets in high-risk (computing) environments indeed occurred within the military, from the USA, UK and other Cold War era allied powers in the early 1970s. The theoretical work there resulted in a concept called *multi-level security* (MLS). The level of assurance required for MLS systems to enforce the separation of different security domains was very high: only complete physical and electrical *or* cryptographic separation sufficed. Since then, using cryptography for controlling access to resources has been more or less implicitly studied. Such type of access control enforcement is generally called today *cryptographic access control* or CAC.

Depending on how CAC is used, it can have multiple benefits. If individual information elements are protected independently from each other with cryptography, where only security policies dictate the possible relations between elements, this implies markedly more fine-grained control of resources than is currently possible. For example, if individual data elements in internet communications are separately encrypted, mass surveillance would be required to recover massively large amounts of key material. On the other hand, protecting large data assets on a per-data-item-basis has orders of magnitude more efficient damage control than by simply tagging a whole database to be accessible by anyone with merely a security clearance[1]. Other areas, which benefit from separately protected

[1] One of the systemic flaws in, e.g., the case of U.S. diplomatic cables leak in 2010 [153] was that the exposed database required only a certain type of clearance to be completely accessible.

items, include digital rights management (DRM) and systems handling personally sensitive information, such as healthcare.

On a more general level, cryptography enforces the protection mechanism at the data. This also means that the enforcement method is agnostic to the actual storage media, storage location or transmission status of data, making cryptography the ideal protection mechanism for today's distributed and dynamic information.

Historically, CAC has been focusing on encryption and on distributing the required encryption keys. These types of schemes tend to be both rigid and inefficient in terms of computational cost, key management cost and complexity. They also suffer from the drawbacks that 1) basically every access permission type is translated to the decryption-permission and 2) trusted key management components are required to be central, making peering different security domains difficult.

The main context for this work stems from one of the latest initiatives to enforce MLS with CAC, called *content-based information security* (CBIS). The defining idea in CBIS was to protect individual data elements with cryptography, according to the CAC principles, but with a more fine-grained control than what typically was manifested in other contemporary CAC implementations. The research on CBIS started in the US, and was afterwards conducted in other countries as well, such as Finland [122]. During the Finnish research on CBIS a new class of public key cryptography emerged, called *functional cryptography*, including functional encryption (FE) and functional signatures (FS).

In FE and FS, the main idea is to reveal only functions of the protected information, not the whole information. This function may also be associated with access control policies, thus immediately suggesting FE and FS to be natural candidates for CAC. However, the practical use cases for modern role-based access control (RBAC), where complex functionalities are routinely handled, are still far from the rather academic settings of FE. The main problems arise from lack of integration of functionalities and security, and neglected areas of practical role enforcement in cryptography. Typically, highly secure schemes tend also to be inefficient, endowed with simple functionality only, or making unrealistic assumptions of the implementations (ranging from ubiquitous trusted third parties to mathematical constructs with no known instantiations).

Our primary motivation throughout this work has been the question of how can the day-to-day operations of high-assurance access control work and processes be handled cryptographically, finding the greatest level of protection, yet keeping the systems efficient and practical? Thus we remain on a more implementation-oriented level rather than in the abstract.

This work is concerned with the intersection of RBAC, MLS and functional cryptography: we investigate the feasibility and models of enforcing a standard RBAC model in standardized distributed access control architectures using functional cryptography in the MLS setting. We chose environments with assurance requirements fit for MLS due to the CBIS background. FE and FS were also intuitively the most suitable CAC categories, but we also confirmed this with other research later on. More precisely, in the context of ANSI [2] standardized $RBAC_3$, OASIS [3] -standardized Extensible Access Control Markup Language (XACML) frameworks and Extensible Markup Language (XML) documents transmitted in publish-subscribe systems, we research the area from four perspectives, with research questions grouped under them below:

1) Architectural perspective: broadly, how different architectures (such as, publish-subscribe, XACML) are affected by the change from conventional access controls (enforced with reference monitors) to CAC. Research questions here include:

    a) What kind of architecture and architectural elements in XACML and publish-subscribe need to be supported, if access control to MLS-documents is to be enforced with CAC, instead of reference monitors (RM)?

    b) Are the responsibilities of different architectural elements (in publish-subscribe and XACML) the same for CAC as without CAC? If not, what are the main differences?

2) MLS-document management perspective: Due to the background of this work, the different aspect of document management and migration paths and possibilities are relevant. The research questions are:

    a) Are ABE and ABS the only possible choices? Are there other mechanisms to support attributes?

[2] American National Standards Institute
[3] Organization of Advancement of Structured Information Standards

b) Is it possible to support transition from PKI-protected MLS-documents to ABE-/ABS-protected documents with XML?

c) Which MLS-functionalities can be accounted for? Does there arise any new challenges when using ABE/ABS?

d) From document management perspective, what are the major differences in using CAC (instead of using RM)?

3) Modelling perspective, or broadly, how different AC model elements can be realized in different settings. The more detailed research questions include:

a) Can other permission types than just **`read`** and **`write`** be enforced cryptographically and how? Is it equally efficient to support different types?

b) Can the **`read`** (resp. **`write`**) permission be enforced in the publish-subscribe environment, where XACML-architecture and RBAC access control model are the defining factors?

c) Is it possible to implement the standard RBAC-commands with existing ABE- (/resp. ABS-) schemes? If this is not possible for all commands, which of them cannot be expressed with ABE/ABS?

d) Does using CAC (instead of RM) imply any profound access control policy handling changes?

4) Efficiency and security perspective: in order to have usable systems, it is important to check that functionally versatile schemes are not pathologically inefficient or use impractical security models. This perspective is present in all of the research carried in this work, but it is not explicitly mentioned unless some of the schemes under scrutiny are seen to present infeasible security models or inefficient implementation. Typical questions include:

a) Does the security model of some particular scheme allow "normal" dynamics of an ICT system, i.e. multiple instantiation, peering, change of different principals and system attributes (or even the *use* of typical system attributes, such as complex policies) efficiently?

b) Given “normal” system operation, what are the relative processing delay and bandwidth overheads for a scheme? In particular, the overhead should be at most in the same order of magnitude as the parameters of the system *without* the scheme.

At the time of writing this, the RBAC model is slowly being contested by a more modern concept, called attribute-based access control (ABAC). We chose to remain with RBAC instead of ABAC for several reasons:

- There is no widely accepted (in the same sense as for RBAC) formal model for ABAC (as noted in the dissertation of Jin [112]), thus making the mappings we use between RBAC and FE- and FS-schemes ambiguous.
- The wealth of existing practice on access control today is still built on RBAC. To address these concerns it is more fruitful to make a mapping to RBAC rather than to ABAC.
- ABAC and RBAC are actually not that far from each other: 1) $ABAC_{\alpha}$, a formal model for ABAC defined by Jin [112] to be the “core” ABAC, is equally as expressive as RBAC, 2) The current ANSI RBAC-standard, $RBAC_3$, already contains multiple extensions over the original RBAC-concept by Sandhu in 1996 [185], all of them to the ABAC “direction”.
- RBAC can be seen as subset of ABAC. Thus, if it turns out that some features of core RBAC (or the separation between ABAC and RBAC) cannot be supported, it is also an indication that the intuition of having a natural map from ABAC to ABE is not correct and furthermore pinpoints those areas that are supported in ABAC but not with CAC (if implemented with FE and FS).

## 1.2 Changes to the First Edition

This work is intended to be read independently, with no prior knowledge of the first edition or the publications. However, to preserve scientific accountability, we present here the subject-matter changes made afterwards.

We added or modified the following chapters:

- Chapter 2.3.3 about Constructive Cryptography, following the work of Matt and Maurer was added, as it more accurately tackles the many definitional problems within functional encryption
- Chapter 2.4.2 was updated to include a more comprehensive and consolidated picture of functional encryption security definitions, including some of the remaining questions. The aim of the additional definitions was to provide a unified and complete view of what actually is functional cryptography (as the field has been, in the opinion of the author, at the time of writing the first edition, somewhat disintegrated and lacking generally accepted definitions).
- Chapter 2.4.4 was updated for the same reasons and with similar material as Chapter 2.4.2, albeit the field of functional signatures is even more disparate than in functional encryption.
- Chapters 9.1 and 9.2, including results from performance simulations, were added to provide more intuition on the performance of cryptographic access control in conjunction of an existing framework, namely NATO Object-Level Protection (OLP) concept.
- The concept and effect of blockchains to CAC was considered in appropriate chapters (2.2.4, 3.1, 5.2, 6.2, 8.1 and 8.2)

## 1.3 Structure of the Book

The structure of this book is as follows. We first introduce the research questions and the motivation for this research. In Chapter 2 we cover the main concepts used in this work, starting from MLS and ending in FE and FS formal definitions. Chapter 3 presents short history and related work on CAC, with focus on information flow control (IFC) systems designed for official use and theoretical work on CAC for RBAC, called cryptographically enforced RBAC, or CRBAC. Chapter 4 explains the concept of PKAA, its relevance and suitability for CRBAC. This chapter also presents our proof-of-concept scheme [123] to use attributes in another

PKAA than what FE represents. In Chapter 5 we present the document management environment together with some general CAC assumptions and requirements for the environment [120], [121]. Chapter 6 shows our general framework for accomplishing some of the “pervasive” functions, such as extending cryptographic enforcement to additional permission types [121]. Chapter 7 binds some of the ABE schemes and principles to the $RBAC_3$ standard and XACML reference architecture, intended for confidentiality policies [119]. We also introduce a new scheme for an ABE scheme security goal relaxation, required specifically for MLS-systems [117]. Chapter 8 builds the same mapping for $RBAC_3$ and XACML Chapter 7, but for ABS and integrity policies [118]. Chapter 9 is reserved for some specific implementation-related notions, which are easily overlooked in ABE research. Finally, in Chapter 10 we conclude our work.

# 2. Basic Concepts

## 2.1 Multi-Level Security

Multi-level security is an information security concept with a long history[4], which is probably why the only widely accepted definitions are high-level. Very coarsely MLS means the capability for a computing platform to securely enable access to information of different classification levels by users from different clearance levels.

For this work, the robust and flexible MLS capability is the application we are striving for. We are, furthermore, tackling primarily the assurance of the separation feature within MLS, with a minor interest in the related integrity issues, such as integrity of the security labels or integrity of redacted[5] documents. Thus other aspects, such as covert channels or the actual sanitization process or policies are left out-of-scope.

In order to understand the MLS concept and its enforcement requirements in relation to other enforcement requirements, it is necessary to know some background to information security classification systems.

Classifying, or categorizing, information assets according to their value, sensitivity and vulnerability is one of the fundamental tenets of information assurance. If this classification system has a direct legislative basis, it automatically places more stringent enforcement requirements to an information system processing such data. Such *governmental classification systems* differ somewhat, but have at least two things in common:

- Enforcement is mandatory
- Classification types form a (partial or full) order, where types are called *levels*[6]

[4] Started evidently in 1967 with a task force inside US Department of Defense [148].

[5] *Redaction* is a type of sanitization, where certain portions are removed or hidden because of their sensitivity, to allow viewing of the rest (less sensitive parts) of the document.

[6] Different languages tend to overload the term “classification”. To be clear, our use of the word refers to information security classification. A “classification” bestowed upon a person (or an automated process) is called a “clearance”, and environmental classifica-

The general ability for a person to access at least some information on a certain classification level is called the *Right-to-Know* (RTK), and it is always enforced with mandatory controls. RTK always requires a natural person to undergo a vetting process to be *cleared* for the particular level. The ability to access a particular piece of information is tied to the user's responsibilities and granted only if access to the piece of information is needed to perform other duties. This type of ability is called *Need-to-Know* (NTK). NTK may be enforced with mandatory or discretionary controls, depending on the exact system and topic.

Certain special broad topics (i.e. cryptographic information or information related to intelligence activities or nuclear technology) may be considered too sensitive to fall under discretionary controls. In such cases, even though they would conceptually belong under NTK, they are promoted to the RTK category, in which case they are called *compartments*[7], running perpendicular to levels. In formal models this results in a partial order (of level-compartment-list – pairs) described by a lattice instead of a chain for levels with a total order.

The concept of MLS is the result of a long history of computer systems protection theory research driven mainly by US Department of Defense needs [14]. Its use is based on a simple risk-index calculation based on possible user clearances vs. classification levels including different compartments (viewed as a risk matrix). The risk index was an integer in the range 0…7, and it was divided into five risk zones.

The different risk zones indicate the requirements a computing system must fulfill, before it can be considered adequate to handle such a combination of users and information. These zones are called security modes of a system as follows:

---

tions (whether a physical location, network or a device is cleared to process information at a certain level) are expressed with the term "clearance" and a clarifying attribute.

[7] Compartments should not be confused with other labels found in classification visual markings, such as *caveats*, which are basically additional information security policy elements (e.g., distribution restrictions or handling directions) carried with the label. These additional markings do not, however, require special handling from the point of MLS [14]. Compartments are initially an intelligence-derived term – a more general term for vertical separation could be *multilateral* security [14].

- *Dedicated mode*: All users are assumed to have both RTK and NTK to the highest level and all compartments in the system. The computing system is not assumed to perform any restriction of access
- *System high*: All users are assumed to have an RTK to the highest level and all compartments in the system, but not necessarily an NTK for all the information. The system is only assumed to perform discretionary access control
- *Compartmented*: All users are assumed to have an RTK to the highest level but not for all of the compartments in the system and not necessarily an NTK for all the information. The system is assumed to perform mandatory access control with assurance level up to the Trusted Computer System Evaluation Criteria (TCSEC, [201]) assurance level B2.
- *Multilevel, Controlled*: The users' RTK with respect to the information is defined as in *Definition 2.1*, and assurance level is required to be "only" at least TCSEC level B3 [201].
- *Multilevel*: As in *Definition 2.1*, and assurance level is required to be least A1 [201].
- *Multilevel, Unsuitable for computing systems protection only:* The risk index is considered too high to allow purely computing-based controls.

In contrast to formal definitions addressing only some aspects of MLS, we adopt a more general, capability- or requirement-centric view on the definition addressing the original problem of classified information combined to user clearances. Our high-level definition of MLS then is based on some of the official military definitions (TCSEC and others[8]), with explicit statements on access control and separation of capability and an actual system.

*Definition 2.1* (Multi-level security, MLS): Multi-level security is a capability of a computing system processing information of multiple classification levels, where not all users of the system are cleared to the highest level of information processed in the system, with the property that users

[8] For example NATO. Usually the military definitions of MLS are embedded as glossaries or definitions inside non-public documents.

can only access information classified at most to their level of clearance. Systems securely implementing this capability, are called *multi-level secure*.

The above definition does not place any restrictions to the amount of difference between user clearance and data classification, which is why stringent assurance requirements are placed for MLS systems. The MLS principle and the resulting partial order lattice are depicted in Figure 1.

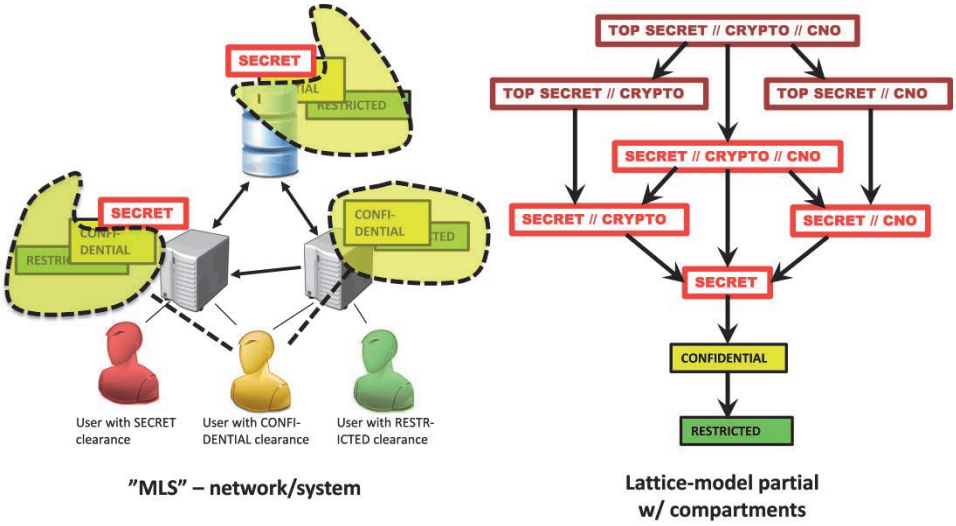



**Figure 1.** MLS principle and a sample of a partial order lattice

The main problem in defining MLS formally is that MLS has been developed both by theoretical and "experimental" (that is, via actual implementations) work, which do not always concur in all of their definitions. Theoretical work include the Bell-LaPadula security policy model [31], considered by many to contain the core requirements of MLS systems. However, other works from noninterference [89] and Harrison-Ruzzo-Ullman-model [105] up to modern RBAC ([185], [126]) have also taken their pick on defining what MLS should actually be.

The Bell-laPadula model states the rules about *information flow*, or which way information may flow between different levels and compartments. Generally, information may flow only from a lower level to higher (called "no read-up, no write-down"). Some exceptions are allowed, for example,

reclassification and the sanitization process[9]. Models expressing the policies in terms of the information flow are called *information flow control* models.

The requirements for an MLS system were originally laid out in the so-called "Orange Book", or TCSEC (US Department of Defense standard 5200.28 [201]). These included:

- Security policy (expressed in terms of the Bell-laPadula model, including the secure and pervasive handling or security labels)
- Accountability (requirements expressed in the assurance levels)
- Assurance (expressed terms of assurance levels, including the requirements for the reference monitors, identification and evaluation of covert channels, requirements for process isolation, etc)

As noted above, the security modes were accompanied by *assurance levels*, or standardized categories of detailed information security requirements a system should implement in order to be considered secure for a certain security mode. The assurance level attached to general MLS is that of TCSEC class A1 (or beyond), which includes, for example, the following properties:

- Formal verification and correctness proofs of the design
- Process isolation
- Formal analysis techniques to identify covert channels
- (from TCSEC class B3:) All of the reference monitor requirements[10] must be satisfied

Even the first systems built along the MLS principles proved to be highly complex to build and maintain (see more discussion, e.g., in the comprehensive book by Anderson [14]). The main problems included reliable

[9] *Sanitization* process involves identifying the higher level elements from the content and removing them, before moving the content to a lower-level environment. For example, when moving content containing SECRET (level II) elements to environment cleared up to RESTRICTED (level IV), all the SECRET and CONFIDENTIAL (level III) elements are first identified and removed. Removing content for this reason is also called *redaction*, and the resulting content is called *redacted*.

[10] Introduced in the access control general concepts

handling of labels, mitigation of covert channels and the difficulty of secure systems composition, if feedback is present.

Moving to the current trend of ubiquitous computing, the original MLS-view is overly simplistic and confidentiality-centric. Especially the distribution of computing has proven to be problematic for the reference monitor concept. However, the high assurance needs for systems handling classified information has not vanished. In environments involving handling of classified information, the main daily challenges revolve around isolation of processes and information of different levels and the systematic labelling for sanitization of information.

Isolation is one way to enforce the Bell-laPadula model IFC. If the environment is grouped together with data into *security domains* identified by the highest label of any data it contains, then if the domains remain isolated (except for carefully defined upload or sanitization mechanisms), the IFC policy remains enforced. Isolation entails that the upload, sanitization and isolation mechanisms have sufficient assurance.

This work is not concerned with the actual upload or sanitization mechanisms (they include techniques such as data diode and pumps), but rather their implications to cryptographic access control. However, the isolation mechanisms are the focus of this work.

Unless a formally verified MLS system with an unpassable reference monitor is in place, the isolation has widely been considered as the only economically viable solution to implement IFC. Isolation mechanisms have traditionally been grouped into three:

1) Physical isolation (requires separate, possibly duplicated physical hardware)
2) Virtualized environments (requires that the virtualization component has sufficient assurance, may not be adequate for full MLS anyway)
3) Cryptographic isolation.

As the security domains' borders (called the security perimeter) has lately been shrinking towards individual data items ([67], [107]), the first option will become infeasible and even the second option seems currently to have extensive overhead (basically requiring at least a separate operating system per each domain). This leaves us with cryptography, which does

enjoy a great abundance of techniques and implementations. We will return to these methods later.

The proper marking of information elements has proven to be problematic mainly for two reasons: marking decisions are traditionally made somewhat arbitrarily by people and not all non-MLS systems connected to (even weakly) MLS-capable systems are equipped to handle or process markings. The attribute-based model that most of the cryptographic schemes presented here use is ideally suited for automating the marking process: markings should be rule-based and dynamic, their final value computed from environment, usage and information attributes rather than being statically assigned by a formal (institutional) authority.

## 2.2 Access Control

### 2.2.1 General Concepts

We explain here the basic (model-independent) concepts in access control on a high level. The exact / formal definitions vary by model. *Access control* in general refers to any method of constraining a set of *subjects* (users, roles, automated processes or other entities) accessing *objects* (usually information in files, databases, documents or parts of them). The *type of access* requested may vary: theoretical work does not generally handle different types separately[11], but in practice tens of differentiable access types exist. The actual ability to access an object is called a *privilege, permission* or an *(access) right*. Usually gaining access is said to be made with an access control *request*, which is then either *granted* or *denied*. Access privileges are also *granted* and *revoked*.

There is a large body of theory researching the access control problem from many facets, from theoretical computer science and computational complexity to cryptography and information security. Our approach concerns the latter ones, and we investigate the theoretical models more thoroughly later.

[11] e.g., the RBAC standard uses general *operations*, which are any executable computer programs

Several practical principles are modelled to some degree of formalism: the *separation of duty* refers to a principle, where the responsibility to perform sensitive operations is distributed over several entities such that they may monitor each other and no single individual can, for example, both order material and authorize the payment for it.

The formal study of access control around different models has started from the 1970's from the access control matrix by Lampson [129], which is still the most general access control model known. The formal study at that time concentrated on the theoretical computation theory aspects, and less focus was given to the practical side of the problem[12].

The later decades have shown that the simplifications and high level of abstraction used in the early models lead either to long deployment time of systems or sloppy implementations on some level of abstraction or at some stage of system life-cycle such that security breaches arise. The models themselves have evolved to mandatory access control (MAC), RBAC and the latest model of ABAC, each with myriad intermediate models and finer extensions.

In this chapter we introduce the more formal access control concepts we use later. These include

- RBAC and its main standard, $RBAC_3$ by ANSI and International Committee for Information Technology Standards (INCITS) [15], and the XACML reference architecture [177],
- ABAC and its relation to RBAC,
- Cryptographic Access Control.

Access control *policies* communicate in a formal or semi-formal manner the intent and purpose of access control in specific systems, or sets of systems. Policies need to be *implemented* or *enforced* with some mechanism in order to be effective: a system with perfectly aligned policies but without any enforcement is a system without any access control.

Access control enforcement mechanisms are modelled with a concept called a *reference monitor* (RM) [13]. Reference monitors are abstract elements, which are assumed to have the following properties [13]:

[12] Interestingly, straightforward implementations of the access control matrix were investigated and found to be either undecidable [38] or not expressive enough [37].

- *Unpassability*: all access control requests are channelled through the RM, and it is not possible to gain access to objects without passing through the RM. If the RM is not operational, no access to any object is available, regardless of the user privileges.
- *Tamperproofness*: the RM cannot be modified without either alerting the system administration or shutting down the RM.
- *Verifiability*: the RM implementation can be verified formally to implement exactly the specified functionality, within a reasonable time frame.

The RM needs to be implemented by a *security kernel*, consisting of hardware and software (e.g., selected operating system kernel functions). In addition to granting or denying access to objects, based on the access control policy, RM is also expected to leave an audit trail, for later scrutiny and inspection. The RM concept is illustrated below in Figure 2.

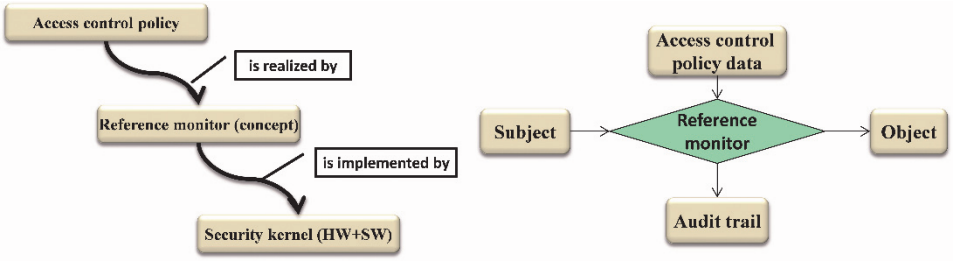



**Figure 2.** The Reference Monitor Concept

### 2.2.2 Role-Based Access Control

RBAC was formally presented by Sandhu *et al.* in 2000 [187] and standardized by ANSI [15]. It is an access control model, which decouples the user (subject) relations to protected objects via a *role*. Roles represent, for example, a real-life employee's or automatic agents' functionality, and the RBAC model eases the administrational tasks considerably compared to access control lists and lattice-based models for military MAC systems.

In this work we use the RBAC model and terminology described in the RBAC-standard by ANSI and INCITS [15] (currently the consolidated version referred to as $RBAC_3$), and restrict ourselves to the administrative commands of the standard in the Core RBAC for brevity. We do, howev-

er, consider Hierarchical and Constrained RBAC as for the structure and functionality they provide to the whole, but not their dynamics (e.g., adding inheritance relations or modifying Dynamic Separation of Duty (DSD) sets). Note that we include the concept of an administrative role from Administrative RBAC (ARBAC97, [186]). The RBAC elements are described in Figure 3 (adapted from the ANSI standard by unifying the notation between concepts, sets and mappings).

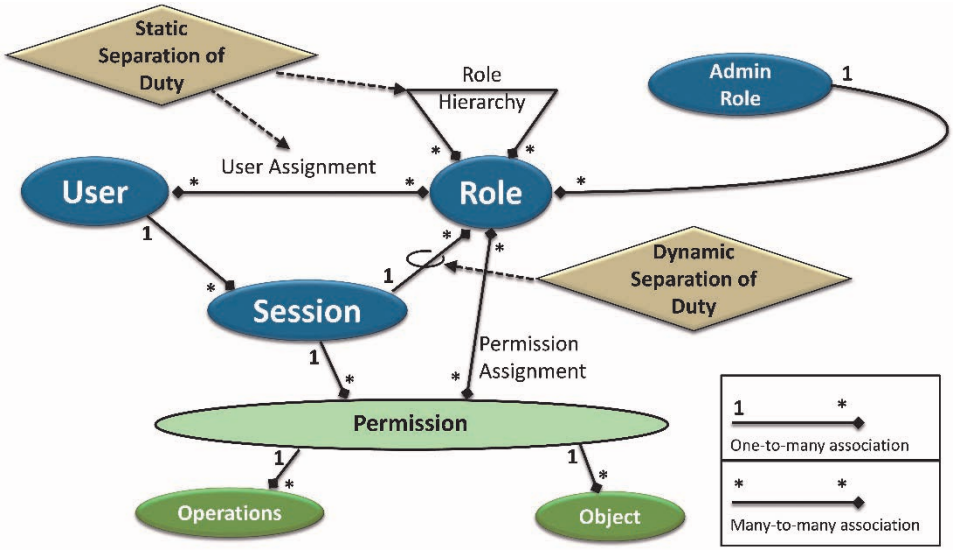



**Figure 3.** Constrained hierarchical (A)RBAC elements and relations

The $RBAC_3$ elements have the following, informal purpose:

- Set-based elements:
  - *Object* (a set OBJ) describes entity requiring protection (usually containing or receiving information or having exhaustible system resources)
  - *Operations* (a set OPS) represent some functionality performed to the object (e.g., reading or writing)
  - *Permission* (a set P) is the privilege to perform an operation on an object
  - *User* (a set U) represents an entity with a need for a permission

- *Role* (a set R) describes a function within the system, such as "Clerk". Only roles may have permissions.
- *Session* (a set S) describes the exercising of a role assignment, e.g., logging in as a "Power User" (called *activating* a role).
- *Role Hierarchy* (a set RH) describes role inheritance: if role $r_1$ inherits $r_2$, then $r_1$ has at least the same privileges as $r_2$. It also implies that the UA-set for $r_1$ is a superset of that of $r_2$.
- *Administrative role* (a set AR) describes a role with the sole function of adding and removing normal roles, their hierarchies and role assignments; appears in the ARBAC-model only.

- Mapping-based elements
  - *Permission Assignment* (a relation PA) relates a permission to a role.
  - *User Assignment* (a relation UA) relates a user to a role.
  - *Object Mapping* (a mapping OBJ.M) maps a permission to a set of objects
  - *Operation Mapping* (a mapping OPS.M) maps a permission to a set of operations
  - *Session-User Mapping* (a mapping SU.M) maps a session to a user
  - *Session-Role Mapping* (a mapping SR.M) maps a session to a set of roles
  - *Session-Permission Mapping* (a mapping SP.M) maps a session to a set of permissions such that it is possible to gather the permissions available to a user via active sessions.
- Restrictors:
  - *Static Separation of Duty* (a set SSD) places constraints on which set of roles a user may simultaneously be assigned to.
  - *Dynamic Separation of Duty* (a set DSD) places constraints on which set of roles a user may simultaneously activate.

More formally, using the set identifications above, we have the following definitions (we present core RBAC from the ANSI standard [15], ch.

5.1.1, role hierarchies from ch. 5.2.1 and constrained RBAC from ch. 5.4 and 5.5 separately for clarity):

*Definition 2.2* (Core RBAC): Given sets *OBJ, OPS, P, U, R* and $\mathbb{S}$, we define the following relations and mappings:

- $UA \subseteq U \times R$, with a specific mapping
  - $UA.M: R \rightarrow 2^U$, such that if $U' \subseteq U, r \in R$ and $U' = UA.M(r)$, then $(\forall u \in U'): \langle u, r \rangle \in UA$
- $PA \subseteq P \times R$
- $OBJ.M: P \rightarrow 2^{OBJ}$ (the range of the mapping being the power set of objects)
- $OPS.M: P \rightarrow 2^{OPS}$
- $SU.M: S \rightarrow U$
- $SR.M: S \rightarrow 2^R$, such that if $R' \subseteq R, s \in S$ and $R' = SR.M(s)$, then
  - $(\forall r \in R'): \langle SU.M(s), r \rangle \in UA$
- $SP.M: S \rightarrow 2^P$

The elements are collectively called the *Core RBAC component*.

A direct implication of the UA relation is that a set of users cannot collectively create additional user assignment for themselves consisting of artificial users. This is only possible via the role manager. This property should also be enforced in CAC (called collusion prevention in FE).

Role inheritance is used to structure roles more closely to organizational hierarchy. Inheritance can be defined in two types: generalized and limited. The limitation concerns multiple inheritance (i.e. a derived role may inherit only from one role) only.

*Definition 2.3* (Role Hierarchies): A partial order $RH \subseteq R \times R$, denoted $\succcurlyeq$, is also called an *inheritance relation* if, for the mappings *AU.M* and *AP.M* described below and $r_1, r_2 \in R$ it holds that $r_1 \succcurlyeq r_2 \Rightarrow AP.M(r_2) \subseteq AP.M(r_1) \wedge AU.M(r_1) \subseteq AU.M(r_2)$. The mappings are defined as:

- Authorized users in the presence of role hierarchy: $AU.M: R \rightarrow 2^U$, such that if $U' \subseteq U, r_1, r_2 \in R, r_1 \succcurlyeq r_2$, and $U' = AU.M(r_2)$, then

$(\forall u \in U'): \langle u, r_1 \rangle \in UA$. This means those users that are assigned to this role and all its inherited roles.

- Authorized permissions in the presence of role hierarchy: $AP.M: R \rightarrow 2^P$ , such that if $P' \subseteq P, r_1, r_2 \in R, r_1 \succcurlyeq r_2$ , and $P' = AP.M(r_2)$, then $(\forall p \in P'): \langle p, r_1 \rangle \in PA$. This means those permissions that are assigned to this role and all its inherited roles.

If additionally it holds, that $\forall(r, r_1, r_2 \in R): (r \succcurlyeq r_1) \wedge (r \succcurlyeq r_2) \Rightarrow r_1 = r_2$, the relation *RH* is said to be *limited*, otherwise it is called *generalized*.

The RBAC standard also involves the specification on how to handle separation of duty. This can be done either statically or dynamically, and these are jointly called constrained RBAC. SSD in the RBAC standard model refers to constraints placed on the *UA*-relation, whereas the dynamic version (DSD) refers to constraints placed on the activated roles, or the *SR.M*-relation. Constrained RBAC is defined more formally below.

<u>*Definition 2.4*</u> (Constrained RBAC): A *constrained RBAC* model consists of the core RBAC, possibly the role inheritance and the following relations:

- $SSD \subseteq 2^R \times \mathbb{N}$ in the presence of *RH* is a collection of pairs $\langle rs, n \rangle$, where $n \geq 2$ and $\forall(\langle rs, n \rangle \in SSD) \wedge \forall(t \subseteq rs): |t| \geq n \Rightarrow \bigcap_{r \in t} AU.M(r) = \emptyset$.
- $DSD \subseteq 2^R \times \mathbb{N}$ is a collection of pairs $\langle rs, n \rangle$, where $n \geq 2$ and $\forall(rs \in 2^R, n \in \mathbb{N})$ the inclusion $\langle rs, n \rangle \in DSD$ implies necessarily all of the following:
  - $n \geq 2$
  - $|rs| \geq n$
  - $\forall(s \in S, rs, rs' \in 2^R, n \in \mathbb{N}): [\langle rs, n \rangle \in DSD \wedge rs' \subseteq rs \wedge rs' \subseteq SR.M(s)] \Rightarrow |rs'| < n$

If role hierarchy is not used, *AU.M* above is replaced with *UA.M*. Informally for SSD, no user should be assigned to no more than *n*-1 of the (sensitive) roles listed in *rs*. In some models, *n* is all cases restricted to *exactly* two, but this is considered overly restrictive in the ANSI standard [15]. In DSD, the definition basically means that for the sensitive roles listed in *rs*, no more than *n* may be activated at the same time.

In some sense, DSD is based more on a flavor of the least privilege-principle called *timely revocation of trust*, which is difficult and error-prone to implement without the DSD-concept in the constrained RBAC-model.

According to the ANSI standard, compliance is achieved by implementing the core set of the functional specification [15]. Such an implementation will need to realize the Core RBAC commands defined by the standard. These commands can be divided into:

- Administrative commands
- Supporting system functions
- Review functions
- Advanced review functions

In order to investigate the feasibility of CAC enforcement in general, it suffices to concentrate on the administrative commands and supporting system functions only. We further present only the informal descriptions of each command for brevity, as the formal descriptions are most often straightforward and do not add extra rigor to the informal descriptions. The commands and their functionality are given in Table 1 below.

**Table 1.** $RBAC_3$ commands and their functionality

| **Administrative commands** | |
|---|---|
| ***Command*** | ***Functionality*** |
| `AddUser()` | Adds a new user to the system, without any sessions. Duplicity should be checked at the time of issuing this command. Adds an element to the set U. |
| `DeleteUser()` | User is deleted, and removed from those data structures it still resides in. Whether or not to terminate the sessions owned by the user (or to wait for them to finish gracefully) is an implementation issue. Removes an element from the set U. |
| `AddRole()` | Adds a new role to the systems, without any sessions or user assignments. Duplicity should be checked. Adds an element to the set R |
| `DeleteRole()` | Removes a role from the system and those data structures it still remains in. The users of the role are left unaffected, |

| | |
|---|---|
| | but whether or not to terminate the sessions owned by the role (or to wait for them to finish gracefully) is an implementation issue. removes an elements from the set R. |
| `AssignUser()` | User is assigned to a role (e relation in UA is created). |
| `DeassignUser()` | Deletes the user assignment from a role. It is implementation dependent, whether to terminate or inactivate the possible open sessions or to wait for them to exit gracefully. The options dictate, whether the user loses authorization immediately or not. Removes the relation from UA. |
| `GrantPermission()` | Adds a permission to the relation PA. |
| `RevokePermission()` | Removes the permission from the relation PA. |
| **Supporting System Functions** | |
| ***Command*** | ***Functionality*** |
| `CreateSession()` | Creates a new session (to the set S), with at least the owning user defined. The initial set of (active) roles to that session may also be empty. |
| `DeleteSession()` | Removes an existing session (from the set S). Also removes the mappings from the active roles and owning user. |
| `AddActiveRole()` | Adds an active role to a session (a mapping SR.M). Usual checks for the user, role, ownership of the session and valid user assignment apply. |
| `DropActiveRole()` | Removes a mapping of type SR.M. Usual checks for the ownership of the user apply. |
| `CheckAccess()` | Checks whether the current relations enable the user of a given session to perform a given operation on a given object. Access is possible to be enabled iff the requested permission is assigned to an active role in one of the users sessions. Note that *enabling* access is not a *guarantee* of the access, as other restrictions may apply. |

### 2.2.3 Attribute-Based Access Control

Attribute-based access control, or ABAC, refers to the principle of basing access control decisions on subject, object and environment attributes [80]. However, there is no consensus (nor even a widely accepted) on the formal model for ABAC, nor a clear picture of the scope of ABAC de-

tailed functionality or mechanisms. Currently one of the most authoritative definitions of the concept can probably be found in the National Institute of Standards and Technology (NIST) SP 800-162 [80]:

*"[ABAC is] an access control method, where subject requests to perform operations on objects are granted or denied based on assigned attributes of the subject, assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions."*

As a high-level concept ABAC is the most suitable counterpart for attribute-based cryptography and reference implementation architectures such as XACML. However, due to the lack of consensus on what exactly ABAC should entail, it was deemed too immature to be considered as the model to work within the context of this book. The reason we include a short review here is solely to give a benchmark of how far from, or close to the future ABAC we currently are in this work, as we consider some RBAC extensions and XACML reference architecture, which is inherently ABAC-based itself.

ABAC has arisen from the need to tie real-life constraints and requirements more directly to the actual access control model: although RBAC successfully models many needs in a traditional enterprise, modern data-driven applications and highly complex enterprise ICT system present ever more fine-grained access control requirements, with ever more dynamic environments. In highly dynamic environments there may commonly be situations which are naturally expressible by attribute-based language and model, but need a formal translation and role structure instantiation in order to be enforced with RBAC. This latter process may become prohibitively expensive both in terms of process and software complexity and time.

Commonly, subjects and objects are given certain attributes stating their purpose, sensitivity, and meta-data, and policies on how to handle the objects depend on those attributes (including environment attributes). If these attributes can directly be employed in the access control model, it presents a significant benefit for the enterprise user.

Different versions for a formal model for ABAC abound, but they tend to be somewhat application-centric. A more general formal approach, which

still tries to embrace as many existing models as possible, can be found in the ABAC dissertation by Jin [112].

As ABAC considers many of the extensions (e.g., Rule-based controls [8] and Task-based controls [157]) of the core RBAC, ABAC is strictly more general than RBAC. Some of the specific extensions that can be considered part of ABAC, but outside RBAC, are listed as follows [112]:

- Extended constraints on role activation, e.g., activation order, prerequisite roles and contextual factors.
- Extended concept for a role, meaning that a role is endowed with parameters (essentially attributes)
- Role – permission relationship changes, including task-based RBAC [157], which adds an additional entity between a role and permission
- Embedding additional organizational structures in the model, such as organizations, teams, groups etc.,
- Adding context information to most of the element in the RBAC model.
- Extended permission structure, mostly related to privacy (adding e.g., purpose, contextual conditions and obligations to the <OBJ, OPS> - tuple).

Some elements of RBAC that we consider here (such as DSD and role hierarchy), are sometimes viewed as “extensions” to RBAC from the ABAC perspective [112].

Although ABAC can be seen as one of the more versatile access control models among the widely acknowledged access control concepts, there are some extensions to RBAC not currently considered to be in the general ABAC framework [112], but exist in others [170]:

- Mutable attributes, e.g., usage control (where resource is allowed to be accessed only a limited number of times)
- Continuous enforcement, meaning that access control is enforced in other points of time in addition to the moment of access control decision.

The most general ABAC formal model given by Jin [112] is called ABAC$_\beta$. This is depicted in Figure 4.

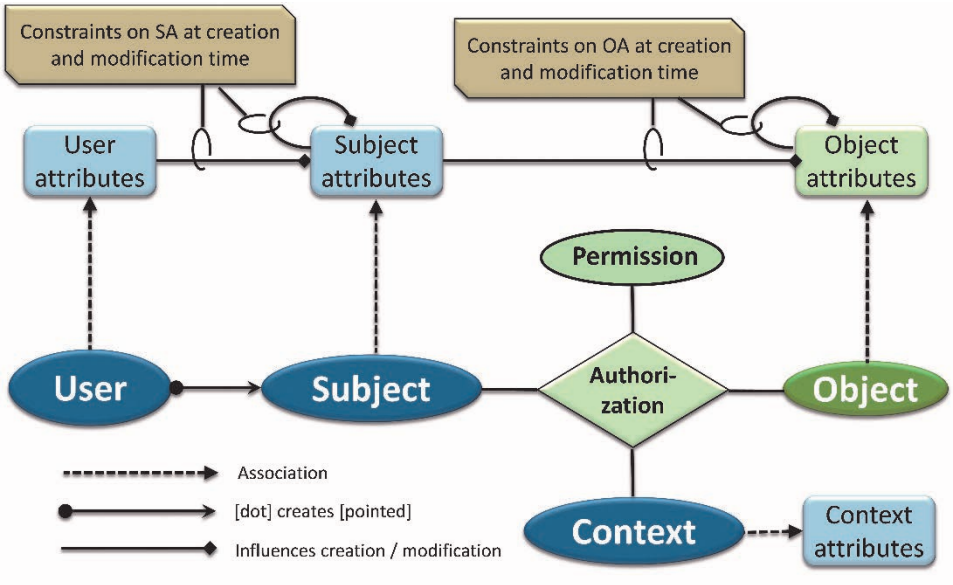



**Figure 4.** ABAC$_\beta$ elements and relations, according to Jin [112]

The ABAC$_\beta$ is somewhat more distributed than RBAC as a model since it does not include roles (which are a central point of RBAC). Instead the authorization function necessarily collects much of the functionality. Moreover, the ABAC$_\beta$-model separates Users from Subjects, referring to actual human users as *users*, and the processes the (human) user creates to the system to perform tasks as *subjects*. For the purposes of ABAC$_\beta$, the subjects cannot create additional subjects, as they are assumed to inherit their rights from the user anyway (thus making it equivalent to having the user create those additional subjects). Objects may also be created by subjects (but not necessarily by the same subjects that access them later, though).

The RBAC role-functionality is distributed into the attributes in ABAC. In ABAC$_\beta$, each element (users, subjects, objects, context and even attributes themselves) is assumed to have attributes[13]. Assigning values to these attributes may depend on the attributes of the creator of the element,

[13] Attributes describing attributes are called *meta-attributes*.

both at the attributes' creation and at the attributes' modification time. This dependency is formalized through the use of *constraints*.

### 2.2.4 Cryptographic Access Control

Cryptographic Access Control (CAC) is in its loosest sense any form of access control that uses cryptography for some access control model elements or functions. In this sense, many versatile CAC-systems capable of very fine-grained access control have existed for some time. However, due to the integration possibilities to modern AC models offered by functional cryptography, we would like to promote a more defined and content-centric view of CAC. While in search of a content-centric CAC-definition, we found that (as is the case with ABAC) there does not seem to be any definitions of what CAC actually is (even high-level definitions that would enjoy a wide acceptance), let alone a formal definition with an abstract model. In order to study CAC properly, there is then need for at least a high-level definition.

We follow the postulation in many papers ([104], [77], [65]) about CAC and view CAC as an *enforcement paradigm*, which aims to replace the reference monitor in as many access controls points as possible. According to this interpretation the following types of schemes do not qualify as CAC in our work:

- The use of cryptography solely in virtualized access tokens (such as Kerberos) is not considered to be CAC, as tokens only contribute to the authenticity of access control claims.
- Any scheme that leaves the primary protected objects to the responsibility of a reference monitor.

The foremost purpose of cryptography even before access control concepts has been to protect information confidentiality while it has been transmitted and after the digitalization became possible, also during other phases of information processing. In that sense, CAC can be considered as old as cryptography itself. However, in the explicit connection to access control, cryptography was used only later. Thus systems that are meant to simply encrypt transmissions or large data archives because of other level requirements are not viewed here as CAC systems: since the focus is access control, we restrict ourselves to abstract *information ele-*

*ments that are independent of their processing media and type or their transmission or storage status*. Thus, for example, traffic encryptor devices, (full) disk encryption software and encrypted databases[14] are not considered to be CAC here.

Due to the lack of existing definitions, we formulate here a high-level definition (in line with the ABAC high-level definition by NIST) for the purposes of this work. Additionally, as our goal is not to define any formal CAC models, we also refrain from making any formal definitions.

*Definition 2.5* (Cryptographic Access Control, CAC): CAC is an access control enforcement method, which involves cryptographic transformations of *application-level information objects*, for the purpose of preventing unauthorized operations on the objects. Information objects are assumed to be independent of their presentation, storage, processing and transmission types, states and technologies.

In *Definition 2.5* we do not exclude key-management of environment-dependent information objects outside CAC, if the key management involves cryptographic transformations and the keys themselves form environment-independent information objects. In this case, the key management objects would be protected with CAC. However, our focus in this work assumes that the content itself (or the lowest level of the data-metadata hierarchy) is also environment-independent information object, protected with CAC methods. It should also be emphasized that the cryptographic transformations are intended for the *purpose* of preventing unauthorized operations, but not *directly responsible* for the prevention. Thus, for example cryptographic signatures would not prevent modifications as such, but could provide sufficient information for active processes to discard corrupted data.

Examples of more formal definitions of CAC in a more restricted setting include CRBAC [77], where Ferrara, Fuchsbauer and Warinschi focus on modelling RBAC as a multi-party computation game, and defining security guarantees on the correct enforcement of the policy.

[14] It would be justifiable to include database encryption systems as CAC systems as well, as they represent a large application area inside the MLS and cloud computing domains. However, we point out that databases agnostic to client content format still fall under the CAC paradigm. Only those solutions that, roughly speaking, encrypt the database rows and tables instead of the data inside them are not considered.

In most CAC settings the focus is on encrypting the actual content. However, our motive is to extend cryptographic support to other access control model elements than merely the actual access check. This was partly addressed in [121] as “pervasiveness” of CAC, which we detail more in the *Definition 2.6*.

A scheme or implementation A addressing at least one requirement in *Definition 2.6* is called (cryptographically) *more pervasive* than scheme (or implementation) B, if the measures used in A are larger than in B for at least one requirement and at least equal for other requirements.

In order to enable pervasive CAC in the metadata dimension, additional solutions are required (examples in [120]).

When the security of different CAC schemes is discussed, the focus tends to be on how well the individual schemes can protect the confidentiality or integrity of the content. This is, however, not the whole picture. In conventional RM-based access control assurance is measured with the trust that the security kernel correctly implements the requirements. This trust is, however, measured with “categorized heuristics”, or levels which basically gather mostly qualitative evidence that an implementation will behave correctly (the levels in TCSEC [201] and derivatives, like Common Criteria [60]). The only exceptions to this in the TCSEC and Common Criteria are the highest assurance levels, where formal proofs are required, indicating a leap from qualitative measures to mathematically verified security (within the protection profile used).

*Definition 2.6* (CAC pervasiveness): A CAC scheme or implementation is called *pervasive*, if it uses cryptographic techniques to enforce or encode access control model elements listed as described in Table 2.

**Table 2.** The requirements for definition of pervasive CAC

| **Requirement** | **Provided Extension** | **Measure type** (used for estimating the compliance with the requirement) |
|---|---|---|
| The enforcement or access control policy affects directly the cryptographic transformation used (e.g., is encoded in the keying material or in the ciphertext) | Extends cryptographic support from the content to the *binding* of the content to the AC policy | The level of flexibility the current scheme or implementation allows: how complex policies are able to be encoded, whether the policies can be encoded in the key, ciphertext or both, and the efficiency of the encoding scheme |
| Access control model operations require cryptographic transformations, for example user-role association (e.g., via cryptographic role-binding [219]) or permission revocation (for example, re-encryption [181]) | Extends cryptographic enforcement from the access check to other AC model operations, such as user-role mapping and its dynamics | The number of AC-model operations supported, with more weight given to the model's core operations |
| Object or content metadata, including key material, can itself be considered as objects requiring protection. | Extends cryptographic support from objects themselves to their metadata, including security management data. | The number of metadata types considered to be requiring cryptographic protection in the implementation. |
| Different permission types are supported with cryptographic enforcement | Permissions outside the basic `read` and `write` types | The number of cryptographically supported permission types |
| Security guarantees concern complete policy enforcement and can be measured in cryptographic terms | Extends the security model scope from content to the AC policy enforcement | By considering the completeness of the security model in relation to the operation of the whole system |

The "categorized heuristics" - type of assurance metric traditionally used in information assurance is not compatible with the use of "provable security" definitions used in cryptography. One approach to achieve compatibility was given in CRBAC ("Cryptographic RBAC") by Ferrara *et al.* [77]. In the CRBAC by Ferrara *et al.* the security goal was shifted from content confidentiality to correct enforcement of a given policy, and due to the modelling of RBAC as a state-transition system, multi-party computation frameworks could be applied to it and security formulated in

cryptographic terms. In the end, the assurance of the correct policy enforcement could be reduced to the security of the underlying (functional encryption) scheme.

CAC is the main tool for access control investigated in this work. It is essential to have a good grasp of the current status of CAC schemes and implementations in order to understand the need for more fine-grained, pervasive, flexible, efficient and yet secure enough schemes to implement large-scale systems. We will make a short review of related work in CAC separately, in Chapter 3.

Although CAC is an enforcement paradigm, it is not completely independent of the underlying access control model. These issues are investigated more in depth later on, but especially with respect to ABAC, two features of the model stand out: high-assurance environmental attribute references and the speed of authorization decision process.

- ABAC uses environmental attributes to make authorization decisions. However, verifying environmental attributes in general is challenging with CAC, as cryptographic elements are very often independent and, in fact, unaware of the environment, which leads to low assurance of the truthness of the stated environmental attribute. This usually leads to delegating the trusted environmental reference to some specialized element, such as Trusted Computing Base modules. In some cases, however, environmental attribute references can be inferred from data-level. Examples include blockchains, which enforce a trusted time reference inferred from a massively distributed and cryptographically verified chain of events. Thus environmental attributes may also be plausibly enforceable with cryptography.
- CAC principles may also provide solutions for apparent discrepancies within ABAC: as the ABAC authorization decision is intended to be, by its very nature, quite an automatic and swift process, this is not always as secure as desired (automated systems are then entrusted to possibly make drastic changes to security postures). In this case, the ability of CAC to make high-integrity (and thus high-assurance) decisions in the form of blockchain smart contracts [198], may be of help to consolidate ABAC-style automated decision making in larger scope.

### 2.2.5 Reference Architecture

Access control models leave the actual implementation details and even the architecture out of scope. Recommendations for specifying what kind of elements are required, and what their interrelations are in order to achieve a successfully working implementation, tend to be application-specific. Of the standardized architectures one of the most popular and widely adopted is the OASIS XACML standard's reference architecture [177].

XACML, or the Extensible Access Control Markup Language, is an OASIS-developed markup language for fine-grained authorization management based on the ABAC concept. The XACML standard as a whole contains:

- A declarative access control policy language implemented in XML
- A reference architecture
- A language for a request-response protocol to be used when requesting services and transmitting information on application level within the reference architecture. XACML does not define actual protocol or transport mechanisms [154]. Instead, other mechanisms are used, commonly Security Assertion Markup Language (SAML) [155] and Simple Object Access Protocol (SOAP, [210], [211], [212], [213]).

The XACML reference architecture is shown in Figure 5. It consists of the architectural elements, and *data / control flow*. Control flow represents, which operations are performed, and at which stage or order.

The architecture main elements and their functionality are as follows:

- *Policy Enforcement Point* (PEP): the embodiment of the RM, and the interface towards the application. The implementation-dependent PEP makes the actual access check (as per the decision by PDP) and either grants or revokes the access.
- *Policy Decision Point* (PDP) makes the actual decision based on policy and other information.
- *Policy Administration Point* (PAP) is where the policies are given and modified from.

- *Policy Information Point* (PIP) extracts relevant information, such as attributes, from different sources and provides them for the PDP. PIP is also implementation-dependent.
- *Context Handler (CH),* which acts as the central translator and orchestrator in collecting information from attribute sources and passing them forward.
- *Obligations Service* is responsible for executing possible obligations (actions that PEP should take in addition of granting or denying access, e.g., logging unauthorized access attempts) forwarded from the PDP. This service typically consists of multiple components not directly in the scope of XACML.

The XACML messages, specified in the protocol language and used in the reference architecture, are either query/response-pairs, or assertions:

- Queries and responses are meant to transmit information about attributes, policies and authorization decisions, and they always occur in pairs.
- Assertions (actually from SAML) are statements regarding different security information. Statements attest to authentication already performed, an existing authorization decision, or that subject has certain attributes.
- The reference architecture follows the ABAC concept, but it has also profiles for RBAC (with extensions). We are not aware of any formal access control models behind the architecture[15].
- Like most architectures and models, XACML assumes implicitly the reference monitor enforcement type. One of our goals in this work is to verify (or disprove) that XACML is general enough to be enforced with CAC.

[15] OASIS has defined its own extensions for RBAC [25], which define extended constraints for role activation based on rule sets.

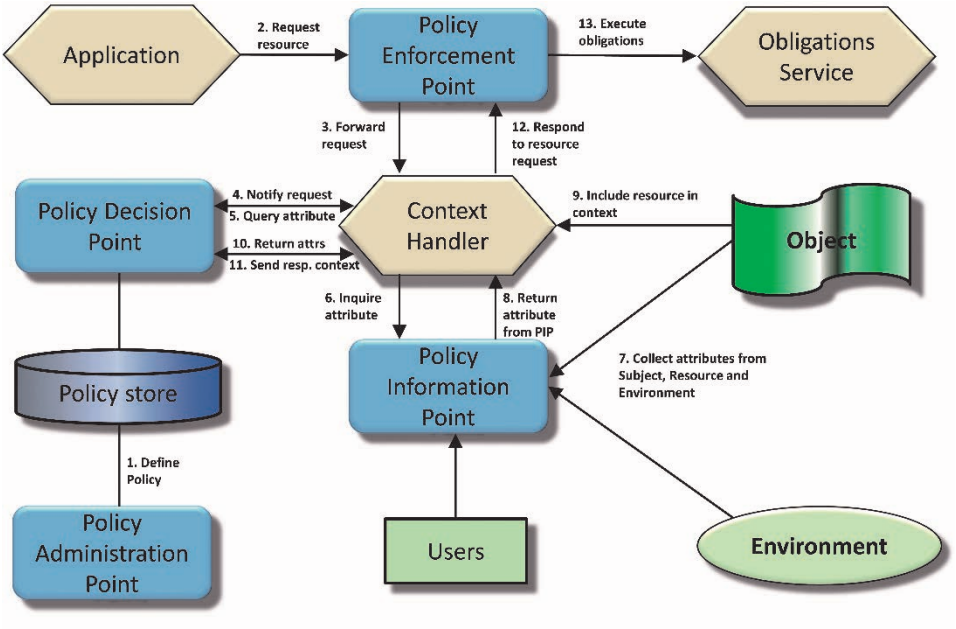



**Figure 5.** XACML reference architecture, according to OASIS [177], adapted to access control terminology

## 2.3 Security Models and Building Blocks

### 2.3.1 Mathematical Logic

The schemes studied in this book make formal and informal use of different branches of mathematical logic for expressing scheme versatility, and the terminology varies. Here we review some of the most commonly used terms in mathematical logic, which we then use consistently. More exact and thorough discussion can be found e.g., from computational complexity textbooks [168], [167].

**Propositional logic**, or propositional calculus, is the study of propositions and their truth values, formally expressed as a formal language with variables, operators, inference rules and axioms. The set of *operators* (omega set) is not fixed, but a very common use of the omega set includes at least logical connectives. The set of *variables* (alpha set) contains symbols with which it is possible to associate actual propositions. For a more formal definition of propositional logic, see e.g., the definitions in the work

of Papadimitriou [168]. The term “propositional logic” can be used interchangeably with “zero-order logic”.

**Nth order logic**. In the common everyday use “mathematical logic” most often refers to first-order logic, which is extensively used to describe and formalize different formal systems. However, functional encryption schemes are not yet at the level where arbitrary first-order logic predicates could be encoded into the systems. Thus we separate the use of terminology and different “orders” of logic.

- N=0. The same as N=1, without quantifiers (universal ∀ and existential ∃) and predicates; zero-order logic is isomorphic to propositional logic, if axioms are considered as well.
- N=1. First order logic is a formal system consisting of syntax and interpretation of the formal language, based on the domain. From the terminology point of view, only the syntax is relevant. Logical formulas consist of
    - *variables*, as in propositional logic,
    - *object constants*, type 0 objects representing some domain-specific, immutable constants,
    - *function constants*, assignments of terms to other terms
    - *logical connectives*, e.g., AND, OR and NOT
    - *atomic formulas*, domain specific predicate not involving any logical connectives
    - *terms*, which are either variables, object or function constants
    - *atoms*, terms or atomic formulas (domain-specific predicates)
    - *literals*, negative or positive atoms

Almost all of the FE- and ABE-schemes use predicates in their descriptions without actually resorting to first-order logic otherwise. In this case, the predicates should be interpreted as templates expressible with logical connectives, constants and variables only. Thus general, natural language predicates, which are possible in first-order logic, are not within the scope of the FE-schemes.

**Logical circuits**. The process of determining truth values of predicates consumes time proportional to the *complexity* of the predicate. Formal complexity theory has modelled the evaluation of logical circuits into computational complexity classes. These classes are used in functional cryptography, and we define them here.

We use a Boolean circuit $B(n)$ with $n$ inputs (bits), and gates described by $s \in \{\vee, \wedge, \neg\} \cup \{\texttt{input, output}\}$, i.e., the logical connectives and input- and output-gates. The Boolean circuits are intended to compute scalar-valued Boolean functions, restricting the number of output-gates to one. For a more formal treatise, see the discussion in [167].We call the depth of a logical circuit $B(n)$, denoted by $\mathbf{Depth_b}(B(n))$, the maximal length path leading from an input gate to the output gate.

The complexity classes for logical circuits stem from research in parallel computation. Definitions here are modified (simplified) from those in [169]. A circuit family $F = \{B(n): n \in \mathbb{N}\}$ is said to be polynomial-time uniform, if there is a deterministic and polynomial-time Turing machine $M$ such that: $\forall(n \in \mathbb{N}): M(1^n) = B(n)$.

*Definition 2.7* (Complexity class $\mathbf{NC^i}$): If a decisional problem is solvable by a polynomial-time uniform circuit family $F = \{B(n): n \in \mathbb{N}\}$, such that $\mathbf{Depth_b}(B(n)) = O(\log^i n)$ for all $n$, we say that the decisional problem belongs to complexity class $\mathbf{NC^i}$.

*Definition 2.8* (Complexity class **NC**): The complexity class **NC** is defined as follows:

$$\mathbf{NC} = \bigcup_i \mathbf{NC^i}$$

Large and important categories of ordinary problems fall in the class $\mathbf{NC^1}$, such as logical connectives, basic mathematical operators (addition, multiplication, exponentiation and logarithm) and comparisons [166]. Examples of problems not known to be in $\mathbf{NC^1}$ (but which are in $\mathbf{NC^2}$) are, for example, graph reachability and matrix determinant [166].

Currently there exist functional cryptography schemes, which are able to encode predicates from the full class **NC** (shorthand for "Nick's Class"), which in turn are all expressible by zeroth order logic. No constructions using first (or higher) order logic are known. Some of the special cases of functional cryptography (notably attribute-base cryptography), which use

specific constructions to optimize bandwidth or processing performance, are capable of encoding predicates from the class $\mathbf{NC}^1$ only.

### 2.3.2 Cryptographic Security Notions

An important goal of cryptography is to provide integrity- and confidentiality-related security services. These in turn comprise of a huge variety of more detailed goals. In this chapter we will cover those security notions that arise commonly throughout the schemes we create or investigate.

Although this book does not strive to be a security-theoretical work, it is still of importance to understand whether certain theoretical advances in functionality of the schemes have reasonable security assumptions. This means, in particular, that the security models should allow dynamic and scalable implementations while not being completely insecure outside their constructed universe. To achieve this, we need to be aware of the concepts and some of the more common notions used in modelling security within the functional cryptography scene.

#### *Trust and Honesty in Cryptography*

The concept of *trust* in the computer security field is rather abstract and not defined exactly or formally. However, we repeatedly use terms relating actors called principals (users, processes or components) to different levels of trust. We describe these terms below.

*Trusted* entities (actors, communications channels, keys, etc.) in a security model are usually considered to be both external to the security model and secure. Security means that as far as the model is concerned, a trusted entity behaves exactly as the model defines, produces perfect randomness and never leaks information. This behaviour also makes trusted entities external to the model, in the sense that it is not necessary to consider its internal operation, only the services and interfaces it provides, such as secure computation, key storage or key material transport.

*Principals* (entities, users, agents) are the players encrypting, signing, verifying and decrypting content, sending and receiving messages. They can be natural persons, computer devices, processes, resources or organizations, among other things. In security models they can be divided into three “honesty” levels:

- *Honest* principals "play by the book", that is, follow the protocol or the scheme description exactly and do not perform any activities besides that, including storing private information computed by them. All trusted principals are considered honest, and trusted to handle private/secret keys, perform decryption and sign material. However, not all honest principals are trusted (to perform all tasks, as they may not have been assigned the adequate resources, and are considered to be internal to the model).
- *Semi-honest* principals follow the protocol or scheme description, but secretly store all intermediate computational results computed by them or directly visible to them, and later try to infer extra information about the computation inputs and outputs [130]. *Semi-trusted* principal, on the other hand, is a scheme-dependent concept, where some confidence is expected with respect to the correct scheme operations. The concept is used, e.g., in signature schemes (such as the scheme by Dong *et al.* [73]) for solving disputes; and in proxy re-encryption schemes (such as the PRE-scheme by Shao *et al.* [195]) to perform the re-encryption. In these views a semi-trusted principal is trusted to perform its task as specified by the scheme/protocol, but it is not trusted to preserve privacy or confidentiality, and thus not allowed, e.g., to escrow private keys. Some views attribute even malicious properties to a semi-trusted party, such as the ability to act maliciously on its own, but not corrupt other principals.
- *Malicious* principals can be expected to do anything, that is, deviate arbitrarily from the protocol / scheme description, including active modification of content and attacks against availability services.

*Game- and simulation based security*

There are several approaches to formalizing cryptographic security notions. Cryptographic primitives have usually rather closely defined usage cases, and thus the formalizations used with primitives tend to model the adversary in a manner that is not the most generic. In contrast, cryptographic protocols need to operate in very complex scenarios, and their formalizations thus have to be more generic than with primitives. Functional cryptography, the topic discussed in this work, appears to be – at

least security-wise - on the borderline between primitives and protocols. In this case we will need to consider security notions from both the frameworks intended for primitives and protocols likewise.

*A security game* is formally a two-party protocol between a malicious adversary $\mathcal{A}$ and an honest challenger $\mathcal{C}$. The protocol is sequential with respect to the parties (parties to do not act simultaneously) and is defined based on the rules of the actual scheme under scrutiny: For $\mathcal{A}$ an assumed "interface" to the scheme is defined, but otherwise no assumptions about $\mathcal{A}$'s behaviour are made. $\mathcal{C}$ is assumed to act innocuously according to the scheme and specific model rules.

*Definition 2.9* (Security game): A security game $\mathcal{G}$ is a alternating two-party protocol between a challenger algorithm $\mathcal{C}$ and an adversarial algorithm $\mathcal{A}$, which always terminates and outputs a value $y \in \{$**abort, SUCCESS, FAIL**$\}$, described as follows:

$\mathcal{G}$:
$\bar{x} \leftarrow \mathcal{D}$
$\bar{z} \leftarrow \mathcal{G}_{atk}(\bar{x})$
$y \leftarrow P(\bar{z})$

$\mathcal{G}_{atk}(\bar{x})$
$\bar{z} \leftarrow \mathcal{A}(\bar{x})$
$\bar{z} \leftarrow \mathcal{C}(\bar{x}, \bar{z})$
$\bar{z} \leftarrow \mathcal{A}(\bar{x}, \bar{z})$
$\bar{z} \leftarrow \mathcal{C}(\bar{x}, \bar{z})$
...
$output\ \bar{z}$

In the description, the main game $\mathcal{G}$ acts as a framework for the actual protocol $\mathcal{G}_{atk}$, taking care of the desired input (vector) distribution $\mathcal{D}$ and a possible predicate $P$ evaluated on the (observable) outcome of the game. The result will be **SUCCESS** (or **FAIL**) if the predicate returns **TRUE** (or **FALSE**, respectively). It should be noted that $\mathcal{A}$ can abort the protocol whenever it wishes, as can $\mathcal{C}$, if the scheme internal rules so dictate. In this case, $y$ will be set to **abort**. Usually a probability measure is associated with a security game, estimating the probability for $\mathcal{G}$ outputting **SUCCESS**, given the input distribution $\mathcal{D}$. The output vector $\bar{z}$ also acts as a temporary storage between different instances of $\mathcal{A}$ and $\mathcal{C}$.

Game-based security models describe:

- what the attacker is assumed to be able to do, and how much information he/she is allowed to access (attack model);

- what kind of information the attacker is able to extract from the system (adversarial success), and
- how a cryptographic scheme is assumed to be able to resist these attacks (rigorous statements of attacker resources or set of assumptions in complexity).

When cryptographic protocols are considered, formulating the security goals itself is a non-trivial task, as sometimes the goals may be too strict to allow any kind of scheme, when on other times goals are so loose as to actually admit circumventing possible implementations. In this case, it is intuitive to consider a type of ideal world, where no better circumstances for a protocol can be envisioned. The aim, then, is to show that a protocol in consideration provides as good a protection as if the participants would operate in the ideal world.

In order to show the operational equivalence of the protocol to an ideal world, a game modelling the tolerated adversarial behaviour is typically constructed, and then the adversary is placed in two experiments: one uses the real protocol / scheme, and the other one a simulator that only pretends to be the real world (actually wrapping the real adversary to an ideal world adversary). If the adversary can in all cases be shown to pay only negligible attention to the difference in the world views, the protocol / scheme is said to be *simulation secure*.

There are multiple details and variations in formal definitions of simulation-based security [53], [90], [91], [92], [93]. As in game-based security, the exact definitions are very much scheme-dependent. We follow here the principle based on the formalization by Laur [130], outlined in Figure 6.

#### *Standard model and ideally randomized models*

In provable security, the standard model refers to proof scenarios, where no idealized random functions or mathematical constructions without implementation specifics exist; adversaries have limited time and limited computational power. In essence, proofs in the standard model rely only on well-specified complexity assumptions and use oracles only to model real-life behavior not considered to be intrinsic to the model itself.

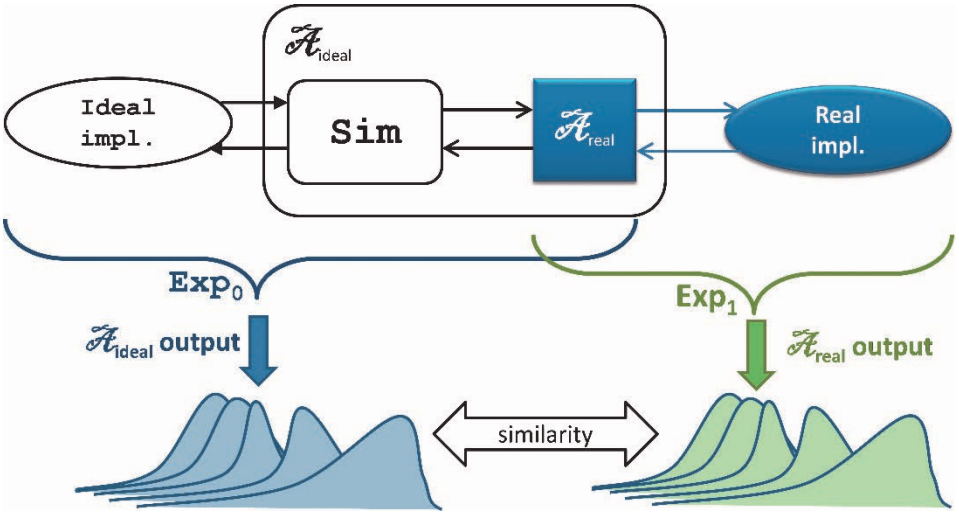



**Figure 6.** The simulation-based security concept

In contrast, there are a multitude of models that assume the existence of certain oracles, providing, for example, abstractions, idealized random functions or efficient functionalities not known to exist. In implementations, schemes that utilize infeasible abstractions should be avoided.

Typical models with separate ideal random functions or high-level abstractions include:

- *Random Oracle* (RO) model, introduced by Bellare and Rogaway [33], which makes use of publicly accessible random functions, called oracles. Important benefits of this model are that schemes are both straightforward to prove secure and to instantiate with secure hash functions, and the resulting schemes are usually quite efficient. Drawbacks include the reliance on secure implementations of the random oracle, and whether the separation between secure schemes in RO-model and standard model includes any "natural" schemes. Random oracles are naturally both *observable* and *programmable*, meaning that in security-games the challenger is able to intercept all the adversary calls for a random function and additionally insert hard problems into the oracle calls / responses. In a *non-programmable* RO-model the simulator / challenger is not allowed to affect the oracle outputs in a meaningful way (to the simulator). This can be achieved, e.g., by allowing additional accesses to the simulator outputs for control [153]. Also

*non-observable* ROMs are known [12]. For our purposes, the conventional RO-model suffices.

- *Multilinear maps* (*Definition 2.10*) can also be viewed as belonging to this category, as it assumes the existence of (efficient) structures with some unconventional properties within algebraic group theory. No efficient multilinear maps are known at the time of writing.
- *Common Reference String Model* (CRS), also called the reference string model or public parameters model [54], assumes that all of the scheme players have access to trusted (or at least authenticated) setup parameters, such as algorithms and their parameters. Most implementations and standards assume CRS, although some publicized cases (e.g., the Dual Elliptic Curve Pseudo Random-Number Generator incident) show that such trust is sometimes misplaced. More importantly, security proofs are valid as the average over all possible setups parameters, whereas in practice only few of them will ever be used [130]. CRS is a fundamental element of NIZK, which in turn appears in some FS and ABS schemes.
- *Other models*, such as the Ideal Cipher Model (ICM, originally due to Shannon [194]) and Generic Group Model (GM) exist as well, but they are not crucial for this work. ICM is similar to the RO-model, but replaces the random function with a random permutation (or ideal cipher). ICM has recently (Coron *et al.* in 2014 [62]) been proven equivalent to the ROM, in the sense of computational indifferentiability[16]. GM [196] assumes the existence of abstract algebraic groups, which do not allow any other operations than checking the membership, performing the group operation and finding the inverse of a group element. Any actions using instantiation-specific structures are not allowed.

All of the models mentioned here, appear in functional cryptography. It is important not to overlook the idealization model used (if any) as it may have important consequences on efficiency and implementation possibili-

[16] An indistinguishability notion based on simulation-based security from Maurer *et al.* [144]

ties in general. This is especially the case in the oracle-schemes, where sometimes rather advanced properties are postulated given the existence of certain oracles, e.g., multilinear maps.

*Linear Maps and Groups*

The notions of bilinear and multilinear maps are extensively used in functional cryptography schemes. We use the definition from the paper of Boneh and Silverberg [48].

*Definition 2.10* (Multilinear maps): For $\kappa + 1$ cyclic groups $\mathbb{G}_1, \dots, \mathbb{G}_\kappa, \mathbb{G}_T$, of the same prime order $p$, a map $e: \mathbb{G}_1 \times \dots \times \mathbb{G}_\kappa \to \mathbb{G}_T$ is called $\kappa$-multilinear, if:

- For $\forall \left( (i, g_i, \alpha): (i \in [1, \kappa]) \wedge (g_i \in \mathbb{G}_i) \wedge \left(\alpha \in \mathbb{Z}_p\right) \right)$ the following holds: $e(g_1, \dots, \alpha g_i, \dots, g_\kappa) = \alpha e(g_1, \dots, g_i, \dots, g_\kappa)$
- $e(\cdot)$ is *non-degenerate*, meaning that if the elements $g_i$ above are all generators of their groups, then $e(g_1, \dots, g_\kappa) \in \mathbb{G}_T$ is also a generator of $\mathbb{G}_T$.

*Definition 2.11* (Bilinear maps): Bilinear maps are multilinear maps with $\kappa$=2.

The algebraic groups supporting bilinear maps are called *bilinear groups* [46]. More formally, if for a group $\mathbb{G}$ there is a suitable group $\mathbb{G}_T$ and a bilinear map $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, $\mathbb{G}$ is called a bilinear group.

Multilinear maps appear in many cryptographic schemes that use functional encryption for general circuits, ranging from witness encryption to program obfuscation [58]. However, it seems to be an elusive goal to find secure constructions to actually implement multi-linear maps: all the three known multilinear map instantiations with integers and lattices ([82], [86], [63]) have subsequently been completely broken (efficient algorithms have been given to recover all the secret parameters of the schemes [58], [61]).

### 2.3.3 Constructive Cryptography

Constructive cryptography (CC) is a concept introduced by Maurer et al. in 2011 (e.g. [142], [143] and [145]).CC aims to add more abstraction levels to cryptographic notions and theorems, such that cryptographic

scheme designers would not necessarily have to deal with higher level issues, such as composability, nor with implementation (or even instantiation-) level details. In a sense, CC is a more hierarchical retake on the notions of simulation-based security and universal composability (UC, more discussion e.g. in [128]) in more of a system-theory setting. The CC authors claim to solve some evident definitional discrepancies in functional encryption with their approach, including those of FE security definitions, making the concept of separate interest in this work.

In the CC model, as in UC, cryptographic protocol or scheme components are thought to be constructed independently (including their security), and a full scheme can then be built securely based on a separate composition theorem ([142]). This implies, however, that the components are built enabling possible composition, or defined in a certain way.

We give here a brief overview of the elements of CC following Matt and Maurer's definitional framework [141]; a more formal discussion can be found e.g. in Maurer's work [142]. CC is concerned with objects called *systems*, which may interact with each other only via *interfaces*. Larger systems can be composed of smaller ones by connecting their interfaces. Three types of systems are distinguished:

- *Resources*, which have a finite set of interfaces
- *Converters*, which have only two interfaces, inner and outer. Inner interfaces can only be connected to resources, making it at new resource: Given a converter $\alpha$, a resource $\mathbf{R}$, and an interface $I \in \mathcal{I}$ (where $\mathcal{I}$ is th finte set of interfaces) of $\mathbf{R}$, connecting $\alpha$ to $I$ yields a new resource $\alpha^I\mathbf{R}$, with $\alpha$'s outer interface as the new interface replacing $I$. A special *blocking converter* is denoted $\perp^I$, which blocks all interactions via $I$. Several converters connected to different interfaces are demoted simply by concatenation: $\alpha^I\beta^E\mathbf{R}$, for converters α and β, and $I, E \in \mathcal{I}$.
- *Distinguishers*, whose purpose is to distinguish between two resources with $n$ interfaces. Distinguishers themselves have $n$+1 interfaces, $n$ of which are connected to an ordinary resource, and one interface outputs a Boolean value. If a distinguisher $\mathbf{D}$ is connected to a resource $\mathbf{R}$, then the probability that $\mathbf{D}$ outputs a Boolean "1" is denoted P($\mathbf{DR}$=1). The success of the distinguisher is measured by the usual definition of an advantage $\mathbf{D}$ has in outputting different Boolean values for different resources, denoted $\Delta^D(\mathbf{R}_1, \mathbf{R}_2)$ for resources $\mathbf{R}_1$ and $\mathbf{R}_2$. If the advantage is negligible

for all efficient $\mathbf{D}$, $\mathbf{R}_1$ and $\mathbf{R}_2$ are said to be computationally indistinguishable, or $\mathbf{R}_1 \approx \mathbf{R}_2$.

CC defines furthermore *compositions*. These can be either *sequential* or *parallel*: sequential compositions are defined naturally by connecting different converter's inner interface to other converter's outer interface, denoted[17] e.g. $\alpha(\beta\cdot\mathbf{R})$; parallel compositions $[\mathbf{R}_1,\ldots,\mathbf{R}_n]$ are resources, where each $I \in \mathcal{I}$ enables accessing the corresponding interface within each $\mathbf{R}_i$ as a sub-interface. The parallel composition of converters is only defined in conjunction with a matching set of resources: $[\alpha_1, \ldots, \alpha_n]^I[\mathbf{R}_1, \ldots, \mathbf{R}_n] \stackrel{\text{def}}{=} [\alpha_1^I\mathbf{R}_1, \ldots, \alpha_n^I\mathbf{R}_n]$ . Note that compositions cannot form branches starting from converters: the actual resource needs to support more interfaces naturally for this to happen[18].

The constructive cryptography concept embraces protocol security and simulation-based security embedding them in the secure composition rules, called *resource construction*. The general case of resources with an arbitrary number of interfaces corresponding to an arbitrary number of honest and malicious principals is lengthy (reader is referred to [143] for a detailed definition) and we need here to consider only the case with three interfaces, one of which corresponds to a principal that may be malicious.

The CC resource construction defines:

- *Protocols* as converter-tuples, denoted $\pi = \langle\pi_1, \ldots, \pi_k\rangle$ with the purpose of constructing a new resource from existing one(s), to achieve some previously unavailable functionality.
- *Simulators* as efficient converters, denoted $\sigma_I$ for $I \in \mathcal{I}$, that provide sub-interfaces from the existing resources to a distinguisher.
- *Construction rule*, which for our case states that for $\mathcal{I} = \{A, B, M\}$ ($M$ potentially malicious), two resources $\mathbf{R}$ and $\mathbf{S}$ with interfaces from $\mathcal{I}$, a protocol $\pi = \langle\pi_A, \pi_B, \pi_M\rangle$:
    - $\pi$ *constructs* $\mathbf{S}$ from $\mathbf{R}$ (denoted $\mathbf{R} \overset{\pi}{\Rightarrow} \mathbf{S}$ ) if there is an efficient $\sigma_M$ such, that:

[17] The parenthesis around $\beta\mathbf{R}$ are actually required, since merely concatenating converter symbols signifies the (not necessarily ordered) use of multiple interfaces in parallel.
[18] Thus, such crypto schemes that offer multiple outputs, need to be modelled as a resource with multiple interfaces and converters rather than just one single converter.

- $\pi_A \pi_B \pi_M \mathbf{R} \approx \mathbf{S} \wedge \pi_A \pi_B \mathbf{R} \approx \sigma_M \mathbf{S}$

The first construction rule condition (correctness) states that (for all practical purposes) the constructed resource behaves exactly like the postulated resource should. The second condition (security) states that whatever $M$ can learn from the protocol and the original resource, it might as well learn from the new (idealized) resource via the simulator. Maurer and Renner prove that there can be constructed a relation that can be used to implement both sequential and parallel compositions securely [143], i.e. if the original systems or schemes are secure (and follow the CC paradigm), the resulting systems / schemes are also secure, provided that a suitable simulator exists.

*Constructive Cryptography Special Resources*

Typically, application-level databases, repositories and communication channels are modelled as resources, and different cryptographic elements as converters, whereas distinguishers are used to prove the security of compositions.

The authors of CC have defined some specific resources that are used rather often and which have more general use. These include:

- *Authenticated channels* between two honest principals A and B and one passive malicious principal E ($\mathbf{AUT}_{A,B}$). $\mathbf{AUT}_{A,B}$ has three interfaces: A is for inputting arbitrary length messages, which is then output to both E and B. E is not assumed to be able to modify the messages.
- *Secure channels* ($\mathbf{SEC}_{A,B}$), which are mostly the same as $\mathbf{AUT}_{A,B}$, except for the difference that E is given only the length $|m|$ of the message *m*. It can be shown that $\mathbf{SEC}_{A,B}$ can be constructed from $\mathbf{AUT}_{A,B}$ via some public-key cryptographic converters, using CC construction theorems.
- *Repositories with access control* ($\mathbf{REP}_F$) model cloud storage-type applications, where one interface (A) is intended for honest users to input (encrypted) data into the repository, another one for possibly partly corrupted users (E) to retrieve functions in class *F* of the plaintext data from the repository, and finally an administrative interface (C) to manage, which principals are allowed to use

which functions at interface E. The data is referenced with special addressing elements called handles (e.g. URIs).

- *Public repositories without access control* ($\mathbf{PREP}_X$) is a special case of $\mathbf{REP}_F$ such that $F$ contains merely the identity function $\mathbf{id}_X$ (with range $X$), and is thus always authorized for anyone to access any data.

Due to their importance to FE, we formalize $\mathbf{REP}_F$ and $\mathbf{PREP}_X$ below, adapting notation from [141].

*Definition 2.12* (Access-controlled repository in CFE): Given a set $F$ of *access functions* with a domain-set $X \neq \emptyset$, some special function $f_0 \in F$, (an initially empty) set $R \subseteq F$ of *authorized* access functions, a *handle* universe $H$ and a map $M: H \to X \cup \{\perp\}$, where $\perp \notin X$, the *access-controlled repository* $\mathbf{REP}_F$ is a three-interface resource, with the resources described as below:

1) Initial settings:
   - $R = \{f_0\}$
   - $(\forall h \in H): M(h) = \perp$
2) Interface A (honest user):
   - **Input** $x \in X$
   - $h \leftarrow$ **getHandle()**
   - $M(h) \leftarrow x$
   - **Output** $h$ (at A)
3) Interface E (potentially malicious user, or a group of them):
   - **Input** $\langle f, h \rangle \in F \times H$
   - **if** $(f \in R) \wedge (M(h) \neq \perp)$ **then**
     - **Output** $f\big(M(h)\big)$ (at E)
4) Interface C (trusted admin):
   - **Input** $f \in F$
   - $R \leftarrow R \cup \{f\}$
   - **Output** $f$ (at C)

The function **getHandle()** returns a newly allocated or reused handle (address, pointer, URI or equivalent), but its internal working is outside the scope for CC.

*Definition 2.13* (Public repository in CFE): Given an access-controlled repository $\mathbf{REP}_P$ with respect to functionality $P = \{f_0\}$, where $f_0$ is the

identity-function $id_X: X \to X$, we define the public repository $\mathbf{PREP}_X$ with domain $X$ the same as $\mathbf{REP}_P$, omitting the identity-funtion-subscript.

It can be shown, that the $\mathbf{REP}_F$ can be constructed from $\mathbf{PREP}_X$ with the help of a CFE-secure FE-scheme [141].

## 2.4 Functional Cryptography

Functional cryptography refers here both to functional encryption (FE) and functional signatures (FS). Although we are primarily interested in one of their special cases, called attribute-based cryptography, the functionalities we require can be found in various types of schemes, which do not necessarily fall strictly under the attribute-based realm. We thus take a more formal look on what FE and FS actually are, and how they differ from ABE and ABS.

### 2.4.1 Functional Encryption

The term "functional" stems from a talk given by Waters in 2008 [180] and it addresses the various new IBE-based cryptographic schemes attempting to solve the cryptographically enforced access control problem (then basically ABE, PE[19] and IBE). Formal definitions were given by Boneh *et al.* in 2010 (published in a peer-reviewed forum in 2011 [47]). Other types of definitions include O'Neill's framework [161] and the composable FE by Matt and Maurer [141].

The formal definition by Boneh *et al.* [47] uses plaintext or messages (space *M*) together with cryptographic keys. In our case, we are limited to using Boolean-valued functions or predicates in the evaluation of the function. We thus modify the original definition to the direction of schemes that specifically employ predicates[20]. When predicates are used, additional information (e.g., access control policies or metadata) are annexed to messages, and these are then called indices (space *I*). The cryp-

[19] Predicate Encryption

[20] There are also schemes under FE that employ a more general decision criteria for *P* instead of predicates, such as Waters' FE [205], which uses deterministic finite automata (i.e. any context-free language) and the ABE by Gorbunov *et al.* [94], which extends *P* to the class **NC**.

tographic keys may also be endowed with policies or metadata. A predicate space (*P*) defined over the key- and index-space determines whether the function is possible to be evaluated in the first place.

Formally, let

$$f: K \times (I \times M) \rightarrow \{0,1\}^*$$

be a function selected from a set *F* defined over a key space *K*, index space *I* and message space *M*, with output as an arbitrary (but polynomially dependent on the input) length bitstring[21]. We also define the predicate space *P* as

$$P: K \times I \rightarrow \{0,1\}$$

*P* is thus a polynomial-time computable predicate over *K* and *I*. It is recommended to carefully distinguish between the usage of *P* and *f per scheme*, as notation and emphasis varies[22]. Typically, *f* is used to modify the available plaintext, and *P* is used to encode an access control policy to the (whole) plaintext.

*Definition 2.14* (Functional Encryption, FE): Given a security parameter $\lambda$, a functional encryption scheme FE is a four-tuple of algorithms $\langle \mathbf{Setup}, \mathbf{KeyGen}, \mathbf{Encrypt}, \mathbf{Decrypt} \rangle$, such that

- $\langle pk, msk \rangle \leftarrow \mathbf{Setup}(1^\lambda)$
- $sk_f \leftarrow \mathbf{KeyGen}(msk, k)$
- $c \leftarrow \mathbf{Encrypt}(pk, (i, m))$
- $y \leftarrow \mathbf{Decrypt}(sk_f, c)$
- All of the algorithms run in polynomial time
- **Setup**(), **KeyGen**() and **Encrypt**() are probabilistic and **Decrypt**() is deterministic.

where

- *msk* is the system-wide master secret key
- *pk* is the system-wide public key

[21] In this work we further assume that all $f \in F$ are deterministic and *K*, *I* an *M* are scalar universes (even though bit-strings could be seen as vectors).

[22] This may stem from the fact that many of the schemes have been designed for databases or massive data archives, and the definition's "plaintext" may refer to a larger encrypted dataset (of which different portions are encrypted differently, and referred to as "messages"). Thus a function on the "plaintext" may become a predicate on the "message" and vice versa.

- $k \in K$ is the set of asymmetric key components used for encryption, dependent on *f*. These are typically public.
- $sk_f$ is the set of private (or secret) key components corresponding to *k*, used for decryption, also called *evaluation token* in some texts [161].
- $m \in M$ is the message (in practical systems usually the symmetric block cipher key used to encrypt the actual content) to be encrypted
- $i \in I$ is the index used (containing metadata and/or policies)
- *c* is the resulting ciphertext when employing **Encrypt**()
- *y* is the result of the decryption operation, which depends on the evaluation of the function *f* and the predicate *P* as:

$$y = \begin{cases} f\big(k, (i, m)\big), & P(k, i) = 1 \\ \perp, & P(k, i) = 0 \end{cases}$$

For the security definitions used by Boneh *et al.* [47], a special "empty" key $\epsilon \in K$ is required. The empty key has the properties:

- $\forall (i \in I): P(\epsilon, i) = 1$
- $f\big(\epsilon, (i, m)\big)$ has some scheme-specific value representing the minimum information leakage by, e.g., observing the ciphertext in-transit.

The public key typically consists of public system-wide parameters (e.g., the group generator used) only. As nearly all FE-schemes are IBE-derivatives, the per-user (or per-attribute or per-predicate) "public key", can be trivially derived from the description of the user (attribute or predicate) itself, given the system-wide *pk*.

Informally, functional encryption enables the encryptor (and implicitly the key management) select functions and keys / key-components such that a user with the secret key (components) can only compute the specified function over $m \in M$. The function type is determined by the scheme, and parametrized by $i \in I$ and $k \in K$.

When the intended usage of FE is access control, the emphasis is on the expressive power of *P*. However, in database queries over encrypted data, private information retrieval or cloud applications the usage may emphasize *f* more. In the former cases, *f* merely returns the whole of *m* (optionally also *i*), but in the latter, *P* is not used, and *f* may return only a predi-

cate or a partial value of the message. It should be noted, however, that the schemes used in these cases are very different[23] and produce functionality only for *either P* or *f*, and we are not aware of any schemes that allow general functionality for *both P* and *f*.

Important subclasses of FE in our case are predicate encryption (PE) and ABE. Historically, PE and ABE were introduced independently, and only afterwards generalized as FE. In terms of FE, PE and ABE can be expressed as denoted in *Definitions 2.15* and *2.16*. *Definition 2.15* is derived directly from the definitional framework of Boneh, Sahai and Waters [47], but *Definition 2.16* is intended to capture the key- and ciphertext-policy versions of ABE more formally than in other definitions in literature, translated into FE notation.

*Definition 2.15* (Predicate Encryption, PE): Predicate encryption is FE, where *P* is non-trivial and:

- $\forall(k \in K, k \neq \epsilon)$: $f\big(k,(i,m)\big) = m$
- $f\big(\epsilon,(i,m)\big) = \begin{cases} \mathbf{len}(m), & \text{hidden index PE} \\ \langle i, \mathbf{len}(m)\rangle, & \text{public-index PE} \end{cases}$

*Definition 2.16* (Attribute-based encryption, ABE): Attribute-based encryption is public-index PE with the following additional restrictions:

- Define $\Phi$ as the set of all polynomial-sized Boolean formulas $\phi(\bar{z})$, where $\bar{z} = (z_1, \dots, z_n)^T$ and $z_i \in \{0,1\}$
- Define $\mathrm{Z} = \{0,1\}^n$
- $K = \mathrm{Z} \times \Phi$, and any $\bar{z}_s \in \mathrm{Z}$ is called a *subjective attribute*, and any $\phi_o \in \Phi$ is called an *objective policy*
- $I = \Phi \times \mathrm{Z}$ (basically *K* and *I* are isomorphic universes), and any $\bar{z}_o \in \mathrm{Z}$ is called an *objective attribute*, and any $\phi_s \in \Phi$ is called a *subjective policy*
- *P* is defined as:

$$P\big((z_s \in \mathrm{Z}, \phi_o \in \Phi)\backslash\{\epsilon\}, (\phi_s \in \Phi, z_o \in \mathrm{Z})\big) = \begin{cases} 1, \text{if } \phi_s(\bar{z}_s) = \phi_o(\bar{z}_o) = 1 \\ 0, \quad \text{otherwise} \end{cases}$$

[23] A very typical setting in schemes specializing in the functionality of *f* is a concept called private-key encryption. In private-key encryption, the large dataset is encrypted with a secret or private key, and the *f* is then evaluated using so-called "tokens" or specific asymmetric (private) key material. This usage, however, is not known to easily translate to attributes and access control.

(i.e. objective and subjective attributes must match to the objective and subjective policies, respectively)

In the ABE literature, this more general view of ABE is referred to as *dual-policy* ABE (DP-ABE) [21]. Additionally, if $\Phi = \emptyset$ for $K$ and $\mathrm{Z} = \emptyset$ for $I$, the term *ciphertext-policy* ABE (CP-ABE) is used. If the roles are reversed: $\Phi = \emptyset$ for $I$ and $\mathrm{Z} = \emptyset$ for $K$, we call the system *key-policy* ABE (KP-ABE).

The reason for naming of CP- and KP-ABE is the location of the actual access control policy, either in the ciphertext or in the key. This categorization carries over to functional encryption and even to signature schemes (indeed, not many dual-policy schemes are known in general).

### 2.4.2 Security Notions for Functional Encryption

The formal security notions for FE have been under scrutiny since 2011 (a version of simulation-based security, named BSW-SIM here [47]). However, as Matt and Maurer point out [141], the gap between application-level security and technical scheme-level security is particularly striking in FE. The actual formal definitions are more tightly bound to the actual application in mind, with a number of seemingly innocuous details to choose from. This results easily in security definitions that are either too weak or too strong [141]: too weak in the sense that trivially insecure functions can be proven secure (happens, when the definition is tied too closely to a specific application); and too strong in the sense that no scheme can satisfy them (in case the definition is too universal). One reason for this could be that FE is on the borderline between applications, protocols and primitives.

Borrowing from the realm of protocol security, where universal composability is used to prove protocol components "application-ready", Matt and Maurer propose to use their notion of constructive cryptography (CC). We use here their security definition [141], since the resources and construction used there closely correspond to our setting, and the definition is very close[24] to the original simulation-based security definitions

[24] In fact the CFE-security definition in [141] *implies* the BSW-SIM-definition in [47]. Even though BSW-SIM already has impossibility results, CFE is defined in a more constrained setting.

[47]. The security is called *composable functional encryption* security (CFE).

The CFE-model achieves possible instantiations in the RO-model only. FE-security definitions using the simulation-paradigm in the standard model seem problematic (evidently too strong) even for public-index schemes [47], [4] and even for various relaxations in the simulation concept. We thus work with the RO-model here.

The essential part of the CFE security is the abstract notion of the application (expressed in constructive cryptography terms), which the definition is tied to. The application used is that of an access controlled data repository, where users can retrieve (functions of) data by using special handles, assuming the users are approved to evaluate a particular function on the data. This is modelled in CC as a separately defined repository-resource, noted $\mathbf{REP}_F$, defined in chapter 2.3.3.

Using $\mathbf{REP}_F$, the CFE-security can be defined such that it is equivalent with a property of an FE-scheme that securely constructs $\mathbf{REP}_F$ from a public repository $\mathbf{PREP}_C$, where $C$ is the domain of Encrypt()-algorithm of the FE-scheme. This construction is represented in the Figure 7.

The construction uses three converters (marked $\pi = \langle \pi_A, \pi_E, \pi_C \rangle$) in parallel with three special resources: $\mathbf{PREP}_C$, $\mathbf{AUT}_{C,A}$ and $\mathbf{SEC}_{C,E}$, or in CC-notation:

$$\left[\mathbf{PREP}_C, \mathbf{AUT}_{C,A}, \mathbf{SEC}_{C,E}\right] \overset{\pi}{\Rightarrow} \mathbf{REP}_F$$

The FE-scheme is used at $\pi_C$ to create and distribute public parameters for the scheme and also distribute the tokens for authorized functions for E. At $\pi_A$ the scheme is used to encrypt user A's sensitive data, and at $\pi_C$ to evaluate authorized functions for E.

The actual CFE-security definition is simulation-based and does not require formalism from the repository definition, which is why we do not replicate it here. An interested reader is referred to the Matt and Maurer's work [141] for more information.

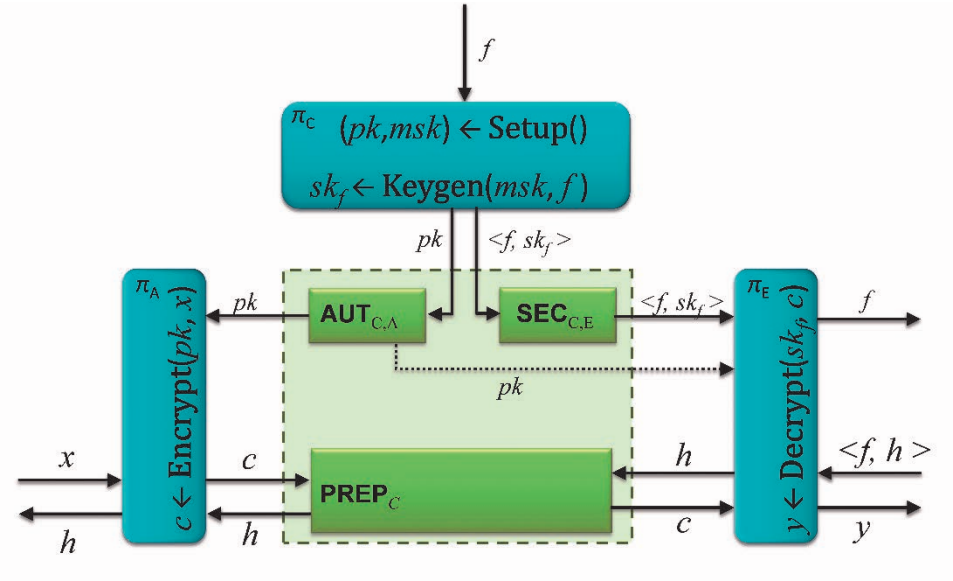



**Figure 7.** Constructing **REP**$_F$ from **PREP**$_C$ using a CFE-secure FE-scheme [141]

With the use case defined, we can now define the actual security notion for CFE:

*Definition 2.17* (CFE-sim security): Let $\mathcal{E}$ be an FE-scheme as defined in *Definition 2.14*, with domain *X*. Also, let $Adv = \langle Adv_1, Adv_2 \rangle$ be a pair of efficient and probabilistic (oracle and normal, resp.) adversarial algorithms, and $Sim = \langle Sim_1, Sim_o, Sim_2 \rangle$ a triple of efficient and probabilistic simulator algorithms, and define two experiments $\mathbf{CFE\text{-}Exp}^{Real}_{\mathcal{E},Adv}$ and $\mathbf{CFE\text{-}Exp}^{Ideal}_{\mathcal{E},Adv,Sim}$ as in Figure 8, with the following additional notation:

- $A^{B(\cdot)}(x)$: *A* gets *x* as input, and may query the oracle *B* with a query *q* (is answered with *B*(*q*)).
- $A(x)[\![s]\!]$:computes a tuple $\langle y, s \rangle = A(x, s)$, but returns only *y*, keeping *s* as an internal state variable.
- $\mathcal{O}(f, x_1, \ldots, x_l)[\![s]\!] \stackrel{\text{def}}{=} Sim_o\big(f, f(x_1), \ldots, f(x_l)\big)[\![s]\!]$
- $\Delta^{\mathbf{D}}\big(\mathbf{CFE\text{-}Exp}^{Real}_{\mathcal{E},Adv}, \mathbf{CFE\text{-}Exp}^{Ideal}_{\mathcal{E},Adv,Sim}\big)$: the advantage for (an efficient) distinguisher **D** in distinguishing the outputs between the real an ideal experiments, taken over the security parameter $\lambda$.

If $(\forall Adv = \langle Adv_1, Adv_2 \rangle, \mathbf{D}): \exists Sim = \langle Sim_1, Sim_o, Sim_2 \rangle$ such that the advantage for **D** is negligible, the scheme $\mathcal{E}$ is called *Composable Functional Encryption (CFE)-secure*.

The notation and variable names follow that of FE *Definition 2.14*, with the additional variables as described below:

- $l$: internal indexing variable
- $\bar{\tau}$: internal state variable for $Adv$, incorporating all the internal relevant information, such as queries made by $Adv_1$ (equals the *Message*-algorithm in [47]) and $Adv_2$, as well as the actual "verbose" output of $Adv_2$ (denoted $\alpha$ in [47]).
- $s$ in an internal state variable for $Sim$.
- $x_l$ and $c_l$ are the plaintext- and ciphertext-messages (respectively) generated by the adversarial algorithms
- $T$: a Boolean value signifying, if the adversary is finished with the experiment. Note that all additional output from $Adv_2$ is gathered to the state-variable $\bar{\tau}$.

**$\text{CFE-Exp}^{Real}_{\mathcal{E},Adv}$**

- $\langle pk, msk\rangle \leftarrow \mathcal{E}.\text{Setup}(1^\lambda)$
- $\langle l, \bar{\tau}\rangle \leftarrow \langle 0, \bar{0}\rangle$
- **do**
  - $l \leftarrow l + 1$
  - $x_l \leftarrow Adv_1^{\mathcal{E}.\text{Keygen}(msk,\cdot)}(pk)[\![\bar{\tau}]\!]$
  - $c_l \leftarrow \mathcal{E}.\text{Encrypt}(pk, x_l)$
  - $T \leftarrow Adv_2(c_l)[\![\bar{\tau}]\!]$
- **while** ($T$ = **FALSE**)
- **return** $\bar{\tau}$

**$\text{CFE-Exp}^{Ideal}_{\mathcal{E},Adv,Sim}$**

- $\langle pk, msk\rangle \leftarrow Sim_1(1^\lambda)$
- $\langle l, \bar{\tau}\rangle \leftarrow \langle 0, \bar{0}\rangle$
- **do**
  - $l \leftarrow l + 1$
  - $x_l \leftarrow Adv_1^{\mathcal{O}(\cdot, x_1, \ldots, x_{l-1})[\![s]\!]}(pk)[\![\bar{\tau}]\!]$
  - $\langle f_1, \ldots, f_q\rangle \leftarrow$ all queries by $Adv_1$ so far
  - $c_l \leftarrow Sim_2\left(f_0(x_l), \ldots, f_q(x_l)\right)[\![s]\!]$
  - $T \leftarrow Adv_2(c_l)[\![\bar{\tau}]\!]$
- **while** ($T$ = **FALSE**)

**return** $\bar{\tau}$

**Figure 8.** CFE-security definition experiments [141]

Compared to the original, non-adaptive definition [47], and an adaptive definition by Gorbunov, Vaikuntanathan and Wee [95], CFE-security has the following differences (mostly to simplify the proofs):

- For simplification:
  - All the internal relevant information is assumed to be encoded in $\bar{\tau}$ (actual state information, queries made by $Adv_1$ / *Message* and $Adv_2$, the actual output of $Adv_2$)

- $Adv_2$ is not given oracle access to $\mathcal{E}.\mathbf{Keygen}(msk,\cdot)$, as $Adv_2$ can delegate this task to $Adv_1$ via $\bar{\tau}$ in the next iteration of the loop

- For a stronger model:
  - Removing oracle access to $Adv_2$ from $Sim_2$ (and thus the ability from the simulator to play the adversary against itself)
  - Allowing $Sim_1$ to fake the public keys instead of calling $\mathcal{E}.\mathbf{Setup}(1^\lambda)$, giving the adversary more chances to make a non-negligible distinction between the world-views. This idea was used in the FE non-adaptive definition [47] and contested in a paper by Barbosa and Farshim [27], since using a system based on trapdoor one-way permutations would give the simulator an advantage of knowing the trapdoor at $Sim_1(1^\lambda)$. However, as noted by Matt and Maurer [141], this knowledge is not relevant to the actual system, and only possibly visible to external systems *unless* hidden by the converters used in the CC-framework.

Different simulation-based security notions are undoubtedly strong, but it is difficult to come by efficient implementations fulfilling the most stringent security requirements. However, the setting for MLS does not require the most general type of expressions from the FE functions. Instead, the public-index PE (especially ABE) schemes suffice.

One striking feature in the most general of FE setting is that the conventional public-key notion of indistinguishability does not, in general, imply semantic security [47], [161]. This may be due to the extra degree of freedom introduced in the possibility to select the function *f* somewhat arbitrarily, which is not accounted for in the traditional indistinguishability (IND) games[25]. On the other hand, it has been shown (by Boneh, *et al.* [47]) that for ABE the game-based IND-security implies both semantic security and simulation-based security, at least in the RO-model in the BSW-SIM-definition. Intuitively, this is because in ABE, *f* on the message is merely the identity function, and thus does not leak any more information than the message itself. As to what extent the equivalence holds

[25] The selection of the function itself also leaks information, which forces the IND-based definitions to accept only specific classes of functions. Basically, encoding the functions in the secret key gives a "semantic attacker" an additional oracle to some functions, if viewed only in the IND-setting. See the paper by O'Neill [161] for more discussion.

as *f* is generalized, is explored more in in the definitional framework by O'Neill [161] and the NM-ABE by Ostrovsky *et al.* [47]. It is not known, whether the implication for ABE carries over from BSW-SIM to CFE or other flavors of simulation-based-security[26].

Moving toward ABE and the security notions particular to ABE, we need to define IND-level security as well. We start with a "pure FE" based definition from the work De Caro et al. [68][27], and expand it to a definition more in the ABE notation.

*Definition 2.18* (FE-IND security): Let $\mathcal{FE}$ be an FE-scheme as defined in *Definition 2.14*, with domain *X*. Also, let $Adv = \langle Adv_1, Adv_2 \rangle$ be a pair of efficient and probabilistic oracle adversarial algorithms. Then $\mathcal{FE}$ is called $(q_1, l, q_2)$-IND-secure, if the distinguishing advantage for *Adv*, as defined below, in the game in Figure 9 is negligible. Here $q_1$ and $q_2$ are the number of oracle (key-) queries made by $Adv_1$ and $Adv_2$, respectively, and *l* is the maximum number of challenge messages allowed by the model.

The distinguishing advantage for *Adv* (essentially a distinguisher) is defined similarly as in *Definition 2.17*:

$$\Delta^{\mathbf{D}}\left(\mathcal{G}^{IND}_{\mathcal{FE},Adv}\right) = Pr(b' = b) - {}^{1}/_{2}$$

$\mathcal{G}^{IND}_{\mathcal{FE},Adv}$

- $\langle pk, msk \rangle \leftarrow \mathcal{FE}.\mathbf{Setup}(1^{\lambda})$
- $\bar{\tau} \leftarrow \bar{0}$
- $\langle \bar{x}_0, \bar{x}_1 \rangle \leftarrow Adv_1^{\mathcal{FE}.\mathbf{Keygen}(msk,\cdot)}(pk)[\![\bar{\tau}]\!]$
- $b \xleftarrow{\$} \{0,1\}$
- $\bar{c}_b \leftarrow \mathcal{FE}.\mathbf{Encrypt}(pk, \bar{x}_b)$
- $b' \leftarrow Adv_2^{\mathcal{FE}.\mathbf{Keygen}(msk,\cdot)}(pk, \bar{x}_b)[\![\bar{\tau}]\!]$
- **return** $b'$

**Figure 9.** FE-IND security definition, adapted from De Caro et al. [68]

[26] This is very likely, due to the IND-security proof of Boneh *et al.* [47] being agnostic to the adaptiveness of the adversary, and the closeness of BSW-SIM and CFE-definitions. We will not, however, present a rigorous analysis here. Independently, it has been shown [68] that any IND-secure FE-scheme can be transformed into a similar SIM-secure FE-scheme with only a linear loss in bandwidth efficiency.

[27] There are multiple similar definitions. This one was selected as the base of adaptation due to its more general theoretic background and notational simplicity.

A restriction is placed for the challenge messages such that $\forall(i \in \{1, \dots, l\})$: $f(k, x_0[i]) = f(k, x_1[i])$ for all $k$ queried by $Adv_1$ or $Adv_2$and all messages $x_0[i], x_1[i]$ selected by $Adv_1$, to prevent trivial distinguishing[28].

The notation and variable names follow those of FE *Definition 2.14*, and CFE-security (*Definition 2.17*) with additional variables as described below:

- $\bar{x}_0, \bar{x}_1 \in X_n^l$, where $X_n^l$is the space of *n*-bit (plaintext) messages forming a vector of length *l* (each member being an *l*-vector of *n*-bit strings)
- $\bar{c}_b \in C_l$, where $C_l$ is the space of *l*-length ciphertext vectors
- $b$ an $b$' are bit-valued variables, the first of which is selected randomly by the challenger from a uniform distribution.

*Definition 2.18* is an adaptive version of the IND-game. If $Adv_2$ does not have any oracle access, the game is called *non-adaptive*. Typical schemes proven secure in the game-based model (for FE or other schemes for that matter) are (*poly*, 1, *poly*)-IND-secure, meaning that only one challenge message is allowed, and the number of oracle queries is limitied only polynomially. However, constructing such schemes in the general FE-setting, and most importantly in the simulation paradigm (or combined with the standard model) seems tricky, which is why different limiting factors are present.

The general FE IND-game captures also ABE IND-games. However, due to the generality of the FE notion, this is not immediately obvious:

- The ABE public-key components are embedded in the FE's function (*f*) parameter *k* and used in forming $sk_f$.
- ABE private-key components are included inside $sk_f$, since the evaluation (and implicitly, decryption) of the function over the message cannot be done without $sk_f$. Here *f* should be understood as being *parametrized* by the policy predicate *P* such that even if *f* in PE and ABE is the identity-function, different policies and us-

[28] Indeed, if $\exists(k \in K, i \in \{1, \dots, l\})$: $f(k, x_0[i]) \neq f(k, x_1[i])$ such that *k* has already been queried, *Adv* can just check if FE.**Decrypt**$(sk_k, c_b[i]) = f(k, x_0[i])$ to check the match for *b*=0.

ers employ different parametrizations and thus, ultimately, also different instances of $sk_f$.

- The ABE public (subjective) policy- and (objective) attribute-components lie in the index, which is considered to be part of the message in FE.
- With the above remarks, the FE-IND-game's restriction becomes valid for *all queried policies*: if there exists any key that allows evaluating the identity function for both messages, the FE-IND restriction requires that $m_0 = m_1$, leaving nothing to distinguish. Thus we end up with the typical ABE IND-game restriction that no queried policies (or their trivial derivatives) must be present at the challenge. Furthermore, as the index is part of the message, the challenge *policy* is implicitly announced together with the actual "payload" messages (using the FE-IND-game).

*Definition 2.19* (ABE-IND security): Let $\mathcal{ABE}$ be an ABE-scheme as defined in *Definition 2.16.* Also, let $Adv = \langle Adv_1, Adv_2 \rangle$ be a pair of efficient and probabilistic oracle adversarial algorithms. Then $\mathcal{ABE}$ is called $(q_1, q_2)$-IND-secure, if the distinguishing advantage for *Adv*, as defined for FE-IND-security (*Definition 2.18*), in the game in Figure 10 is negligible. Here $q_1$ and $q_2$ are the number of oracle (key-) queries made by $Adv_1$ and $Adv_2$, respectively.

We reiterate the notation in Figure 10 for convenience:

- $A^{B(\cdot)}(x)$: *A* gets *x* as input, and may query the oracle *B* with a query $q$ (is answered with $B(q)$).
- $A(x)[\![s]\!]$: computes a tuple $\langle y, s \rangle = A(x, s)$, but returns only *y*, keeping *s* as an internal state variable.
- $\bar{\tau}$: internal state variable for *Adv*, incorporating all the internal relevant information, such as queries made by $Adv_1$ and $Adv_2$, as well as the actual "verbose" output of $Adv_2$
- $x_l$ and $c_l$ are the plaintext- and ciphertext-messages (respectively) generated by the adversarial algorithms
- $\bar{z}_s, \phi_o, \bar{z}_o^*, \phi_s^*$ and $pk$ follow the ABE notation in *Definition 2.16.*

$\mathcal{G}^{IND}_{\mathcal{ABE},Adv}$

- $\langle pk, msk\rangle \leftarrow \mathcal{ABE}.\mathbf{Setup}(1^{\lambda})$
- $\bar{\tau} \leftarrow \bar{0}$
- $\langle x_0, x_1, \bar{z}_o^*, \phi_s^*\rangle \leftarrow Adv_1^{\mathcal{ABE}.\mathbf{Keygen}(msk,\cdot)}(pk, \bar{z}_s, \phi_o)[\![\bar{\tau}]\!]$
- $b \xleftarrow{\$} \{0,1\}$
- $c_b \leftarrow \mathcal{ABE}.\mathbf{Encrypt}(pk, x_b, \bar{z}_o^*, \phi_s^*)$
- $b' \leftarrow Adv_2^{\mathcal{ABE}.\mathbf{Keygen}(msk,\cdot)}(pk, x_b, \bar{z}_s, \phi_o)[\![\bar{\tau}]\!]$, such that $\phi_o(\bar{z}_o^*) = 0 \wedge \phi_s^*(\bar{z}_s) = 0$
- **return** *b'*

**Figure 10.** ABE-IND security definition.

Additionally we note the following:

- In public-index schemes and IND-level security, one- and many-message security concepts are equivalent [95], and we work with one-message security for simplicity. Thus, in comparison to *Definition 2.18*, we were able to omit the parameter *l*, to replace $\bar{x}_b$ with $x_b$, $\bar{c}_b$ with $c_b$ and the universes $X_n^l$ and $C_l$ with $X_n$and $C$, respectively. In a typical scheme, $q_1$ and $q_2$ are bounded only polynomially.
- To better express the ABE details, we incorporated ABE **Keygen**() and **Encrypt**()-parameters as defined in DP-ABE [21], and also separated the challenge access policies and attributes from the challenge messages.
- $Adv_2$ may not query such private keys for such subjective attributes and objective policies that the model trivially breaks, i.e. the queried subjective attributes may not fulfill the challenge subjective policy formula, and the queried objective policy may not be such that the challenge objective attributes would fulfill its policy formula.
- The model covers both ciphertext- and key-policy versions of ABE (as does the *Definition 2.16*).

This definition addresses so-called "full" security or (due to the overuse of the word) adaptive, non-selective security. If $Adv_2$ is not given any oracle accesses, the security definition becomes *non-adaptive*. Furthermore, if $Adv_1$ may not decide on the challenge attribute-set and access policy, but instead the adversary is forced to declare them before **Setup**(),

the model is called *selective(-set)* security. In case selective security is used, $Adv_1$ is subject to the same restrictions as $Adv_2$, w.r.t the trivially satisfying policies and attributes.

### 2.4.3 Functional Signatures

The terminology w.r.t digital signatures according to the "functional (encryption)"-paradigm is not as consolidated as it is with FE. Although there have been schemes dubbed "functional signatures", they may not necessarily encompass other types of signature schemes with similar functionality as widely as in FE. However, we adopt here the term *functional signatures* (FS) as the main concept.

In order to accommodate ABS-schemes under the more general FS, we split the FS message- space into index- and message-space (much like in *Definition 2.14* for FE). As in *Definition 2.14*, we denote additional metadata or policies annexed to the actual content with the index-set.

In contrast to FE, all known FS- and pure ABS-schemes do not allow a separate public per-policy key-space (space *K* in *Definition 2.14*). Using such a space would basically allow the verifier to set a policy or the signing policy to be visible, which is against the FS security goals (not against ABS security goals, however).

With these additional details, and using a notation analogous to FE, we follow the definition by Boyle *et al.* [51]. Formally, let

$$f \in F: (I \times M) \rightarrow \{0,1\}^*$$

be a function defined over an index space *I* and message space *M* (handled as one universe in FS definitions) with output as an arbitrary (but polynomially dependent on the input) length bit-string[29]. We furthermore write desc(*f*) to denote a *description* of *f* in order to separate it from *evaluation* of *f*, and $|f|$ to denote the *size* of function in (some) implementation and in some metric (e.g., the number of logic gates). We also define the predicate space *P* as

[29] Generalizations to $f \in F$ being a function exist as well: Policy-based signatures [32] extend the domain of $f$ to include a universe of witnesses, and $f$ to be any policy-based language in **NP** requiring replacing $\pi \in P$ with a more general relation *R*. For access control purposes, however, predicates suffice.

$$P: I \rightarrow \{0,1\}$$

Thus *P* is a polynomial-time computable predicate over *I*. *P* is used later in defining the ABS paradigm in the FS framework.

*Definition 2.20* (Functional Signatures, FS): Given a security parameter $\lambda$, a functional signature scheme FS is a four-tuple of probabilistic polynomial-time algorithms $\langle \mathbf{Setup}, \mathbf{KeyGen}, \mathbf{Sign}, \mathbf{Verify} \rangle$, such that

- $\langle mvk, msk \rangle \leftarrow \mathbf{Setup}(1^{\lambda})$
- $sk_f \leftarrow \mathbf{KeyGen}(msk, \text{desc}(f))$
- $\langle f(i,m), \sigma \rangle \leftarrow \mathbf{Sign}\left(mvk, \text{desc}(f), sk_f, (i,m)\right)$
- $b \leftarrow \mathbf{Verify}(mvk, f(i,m), \sigma)$

where

- *msk* is the system-wide master secret key; the holder of *msk* is called the *signature trustee* (since the trustee can forge any signature)
- *mvk* is the system-wide master verification key; depending on the actual scheme *mvk* may or may not include system-wide global public parameters
- $sk_f$ is the *signing key*, the set of private (or secret) key components corresponding to *f*, used for signing. The holder of $sk_f$ is generally a different entity (or process) than the signature trustee. With ABS, the $sk_f$ manager is usually called an *attribute authority*.
- $m \in M$ is the message (in practical systems usually a cryptographic hash of the actual content) to be signed
- $i \in I$ is the index used (containing metadata and/or policies)
- $\sigma$ is the resulting signature or tag (possibly itself consisting of multiple components), when employing **Sign**()
- $b \in \{0,1\}$ is a Boolean value indicating whether the verification succeeded (σ was *accepted*; *b*=1) or failed (σ was *rejected*; *b*=0).

Functional signatures (and signatures in general) need to fulfill *correctness*: honestly generated signatures are accepted if and only if verified with correct parameters.

The term “attribute” was first associated with group signatures in [115] on 2007. The first pure ABS-schemes supporting threshold gates and finally arbitrary length monotone access structures arose in 2009 from a work by Shahandashti and Safavi-Naini [193]. They also appeared in 2008 in a work by Maji *et al.* [140] that was peer-reviewed in 2011 [139]. ABS can be thought of as an FS-scheme with some restrictions. Note that we adopt the notation from *Definition 2.20*, especially for the description of the predicate itself, as it is not always clear, whether the description, functionality or “presence” (for example, the **C**-programming language “function pointer”) of the predicate is used.

*Definition 2.21* (Attribute-based Signatures, ABS): Attribute-based signatures are functional signatures with the following additional definitions:

- $ABS.f(i,m) = \begin{cases} \langle i,m \rangle, \pi(i) = 1 \\ \perp, \quad \pi(i) = 0 \end{cases}$
- $ABS.\sigma = \langle \sigma, \mathrm{desc}(\pi) \rangle$

The predicate $\pi \in P$ is called the *claim predicate* (terminology from the ABS by Maji *et al.* [139]).

In comparison to ABS-schemes in the literature, we use in our definitions:

- predicates instead of access structures (the mapping is efficient and one-to-one, but usually left out, for simplicity, from ABS definitions [139])
- one setup function instead of multiple, and as a consequence, we do not divide *msk* into decentralized components (ABS definitions sometimes account for multiple independent administrative attribute-issuing authorities, which definition-wise can be seen as an extension of the basic functionality)

A major difference between ABS and FS is that ABS schemes do not generally require policy privacy: the FS-verifier can only be certain that the signature was generated according to *some* generator-approved policy, whereas in ABS the verifier also receives the policy itself (embedded in σ, in *Definition 2.21*). Indeed, most ABS-schemes cannot even perform verification without at least some guess of the signing policy. Thus we do not specify where the policy needs to be described and where it needs to be evaluated.

Functional signatures can also be extended in various ways. One significant (in our case) way to do this is *delegation*. Delegatable functional signatures (DFS, [24]) are such an extension, and thus also form a more general type of FS. DFS allow some malleability in signatures in order to comply with situations, where some portions of the data are modified on behalf of the secret key holder (such as redaction or outsourced computation). DFS could also be used to split the signing policy into a private and publically verifiable portion.

### 2.4.4 Security notions for Functional Signatures

The basic security goal of all signature schemes is *unforgeability*: only legitimate signer(s) should be able to produce signature elements, which verify correctly under applicable parameters. However, as the number of signers, signature elements and verification parameter options are increased, the possible functionalities and thus also possible security goals are various.

Even the basic security notion of unforgeability is more complex for FS, since secret keys have functions as their parameter. Most notably two secret keys with different functions signing the same message into two unlinkable signatures may sound like a candidate for strong existential unforgeability, while they are actually just two different private keys (albeit possibly under single ownership).

The development towards functional signatures has gone via group-, ring-, mesh- and attribute-based signatures. While many of the former signature types are not of concern in this work, it is, however, instructive to review, which of the security goals have been incorporated into functional signatures.

Group signatures, even in the attribute-based setting (ABGS, [115]), typically involve the *anonymity of the signer* as their most important security goal after unforgeability. This goal is preserved in more or less strict form in all of the FS predecessors (and even in some successors). However, the original group signatures also call for *traceability*, i.e. the ability for a special entity, the group manager, to expose an individual signer in case of dispute. Serious group signature schemes also cover (under suitably defined anonymity):

- *Unlinkability*: infeasibility to decide whether two signatures were signed with the same key or not
- *Exculpability*: infeasibility to create valid signatures for a non-participating group member.

Later signature types, namely ring- and mesh signatures, dropped traceability from their security goals in order to provide for a better protection to the actual signer [50], [179]. Mesh signatures already consider attributes, but do not consider user collusion nor have attribute or policy privacy as a concern.

The attribute-based signature schemes include *collusion prevention* and *attribute privacy* to the list of security goals, whereas policy privacy is not a concern (rather a requirement in order to build working schemes).

- Collusion prevention means the same thing as in FE: two or more users with secret keys to different capabilities should not be able to attest for a property or a message with any higher confidence or trust than is given to any of them individually
- Attribute privacy refers to the inability of the verifier to tell, which attributes the signer satisfies, only that they fulfill the policy communicated.

Functional signatures use the concepts of *function privacy* (denoted IND for indistinguishability w.r.t policies) and extend unforgeability to functions on the message. These also imply:

- Collusion prevention: the FS unforgeability definition (*Definition 2.23*) entails that the adversary, after querying many function-secret keys $sk_{f,i}$ and messages $m_j$ cannot produce signatures for any function-secret key and message not queried. Collusion, on the other hand, corresponds to a case of constructing a signature on some message (against security policy) with a key $sk_g = g(sk_{f,1}, sk_{f,2})$, where $sk_{f,1}$and $sk_{f,2}$ are keys issued by the signature trustee, while $sk_g$is not. This corresponds to a case in the unforgeability model, where the adversary has queried $sk_{f,1}$and $sk_{f,2}$ and tries to forge a signature with $sk_g$.
- Policy and attribute privacy: since policies, even if expressed only with formulas over attributes, are still functions (as expressed in the definitions). In some models [32], unforgeability is defined via

additional (non-interactive zero-knowledge, NIZK[30]) notions that require the signature trustee, together with another entity tasked for the purpose, to be able to extract the policy with their private-key material. This is, in a sense, weakening to the policy privacy requirement.

- Unlinkability is implied by the function privacy: the ability to identify, whether two (different) signatures are created under a same key or not can be trivially used to check if a signature created by the IND-adversary was created by the same key as the adversary.

Functional signatures have the additional ability to control the message space, which are authorized for entities to sign in the first place. In the basic FS this space is implicitly restricted to the range of the function defined / used. In policy-based signatures (PBS, [32]), the allowed messages are explicitly listed in the language description.

Following the conventions set in the chapter for FE security notions, we start from the more general definitions, applying the simulation-based definition for unforgeability in PBS [32].

*Definition 2.22* (Simulatability for FS): Let $\mathcal{FS}$ be an FS-scheme as defined in *Definition 2.20*. Also, let $Adv = \langle Adv_1, Adv_2 \rangle$ be a pair of efficient and probabilistic oracle adversarial algorithms, and $Sim = \langle Sim_1, Sim_o, Sim_2 \rangle$ a triple of efficient and probabilistic simulator algorithms, and define two experiments $\mathbf{Exp}^{Real}_{\mathcal{FS},Adv}$ and $\mathbf{Exp}^{Ideal}_{\mathcal{FS},Adv,Sim}$ as in Figure 11, with notation as in *Definition 2.17*:

- $A^{B(\cdot)}(x)$: $A$ gets $x$ as input, and may query the oracle $B$ with a query $q$ (is answered with $B(q)$).
- $A(x)[\![s]\!]$:computes a tuple $\langle y, s \rangle = A(x, s)$, but returns only $y$, keeping $s$ as an internal state variable.
- $\mathcal{O}(f, x_1, \ldots, x_l)[\![s]\!] \stackrel{\text{def}}{=} Sim_o^{\mathcal{FS}.\mathbf{KeyGen}(msk,\cdot)}\big(f, f(x_1), \ldots, f(x_l)\big)[\![s]\!]$
- $\Delta^{\mathbf{D}}\big(\mathbf{Exp}^{Real}_{\mathcal{FS},Adv}, \mathbf{Exp}^{Ideal}_{\mathcal{FS},Adv,Sim}\big)$: the advantage for (an efficient) distinguisher $\mathbf{D}$ in distinguishing the outputs between the real an ideal experiments, taken over the security parameter $\lambda$.

If $(\forall Adv = \langle Adv_1, Adv_2 \rangle, \mathbf{D}) : \exists Sim = \langle Sim_1, Sim_o, Sim_2 \rangle$ such that the advantage for $\mathbf{D}$ is negligible, the scheme $\mathcal{FS}$ is called simulatable.

[30] Extractability for signatures of knowledge

Other notation and variable names follow that of CFE simulatability definition (*Definition 2.17*), re-listed here for convenience:

- $l$: internal indexing variable
- $\bar{\tau}$: internal state variable for $Adv$, incorporating all the internal relevant information, such as queries made by $Adv_1$ and $Adv_2$, as well as the actual "verbose" output of $Adv_2$.
- $s$ in an internal state variable for $Sim$.
- $x_l$ is a message produced by $Adv_1$. $x_l$ contains here also the index $i$ as well as the actual "payload" $m$. Note that $x_l$ may also be empty.
- $sk_{f_l}$ is a secret key corresponding to $\mathrm{desc}(f_l)$, queried by $Adv_1$ from the $\mathcal{FS}$.KeyGen()-oracle. $Adv_1$ may also decide to output an empty $sk_{f_l}$.
- $\sigma_l$ is the signature, including evaluation of $f(x_l)$, generated by $Adv_2$. he adversarial algorithms
- $T$: a Boolean value signifying, if the adversary is finished with the experiment. Note that all additional output from $Adv_2$ is gathered to the state-variable $\bar{\tau}$.

---

$\mathbf{Exp}^{Real}_{\mathcal{FS},Adv}$

---

- $\langle mvk, msk\rangle \leftarrow \mathcal{FS}.\mathbf{Setup}(1^\lambda)$
- $\langle l, \bar{\tau}\rangle \leftarrow \langle 0, \bar{0}\rangle$
- **do**
  - $l \leftarrow l + 1$
  - $x_l, sk_{f_l} \leftarrow Adv_1^{\mathcal{FS}.\mathbf{Keygen}(msk,\cdot)}(\mathrm{desc}(f_l))[\![\bar{\tau}]\!]$
  - $\sigma_l, T \leftarrow Adv_2^{\mathcal{FS}.\mathbf{Sign}(sk_{f_l},\cdot)}(x_l, \mathrm{desc}(f_l))[\![\bar{\tau}]\!]$
- **while** ($T$ = **FALSE**)
- **return** $\bar{\tau}$

---

$\mathbf{Exp}^{Ideal}_{\mathcal{FS},Adv,Sim}$

---

- $\langle mvk, msk\rangle \leftarrow Sim_1(1^\lambda)$
- $\langle l, \bar{\tau}\rangle \leftarrow \langle 0, \bar{0}\rangle$
- **do**
  - $l \leftarrow l + 1$
  - $x_l, sk_{f_l} \leftarrow Adv_1^{\mathcal{O}(\cdot,x_1,\ldots,x_{l-1})[\![s]\!]}(\mathrm{desc}(f_l))[\![\bar{\tau}]\!]$
  - $\langle \mathrm{desc}(f_1), \ldots, \mathrm{desc}(f_q)\rangle \leftarrow$ all queries by $Adv_1$ so far

---

- $\sigma_l, T \leftarrow Adv_2^{Sim_2(f_0(x_l),\ldots,f_q(x_l))[\![s]\!]}(x_l, \mathrm{desc}(f_l))[\![\bar{\tau}]\!]$
- **while** ($T$ = **FALSE**)

**return** $\bar{\tau}$

**Figure 11.** FS simulatability definition experiments [32]

Moving from simulation-based definition to game-based is straightforward, as the definition in PBS [32] is technically the same as in the FS by Boyle *et al*. [51], only adding simulators in place of the oracle calls for the ideal world experiment. Thus the game-based unforgeability game is exactly the same as the simulation-based security real-world experiment, except for the return value and additional restrictions for the oracle calls. This is formalized as a game $\mathcal{G}^{UF}_{\mathcal{FS},Adv}$ that operates exactly as $\mathbf{Exp}^{Real}_{\mathcal{FS},Adv}$, except for the following:

- The experiment(/game) will return $\langle x^+ = f^*(x^*), \sigma^* \rangle$, for signature tag $\sigma^*$, some function $f^*$ and some message $x^*$ such that:
  - $\forall f_i$ queried from the $\mathcal{FS}$.**Keygen**()-oracle in $\mathbf{Exp}^{Real}_{\mathcal{FS},Adv}$: $\nexists x: x^+ = f_i(x)$
  - $\forall (f_i, x_k)$ queried from the $\mathcal{FS}$.**Sign**()-oracle in $\mathbf{Exp}^{Real}_{\mathcal{FS},Adv}$: $x^+ \neq f_i(x_k)$
- The oracle model requires that identical calls to the oracles with identical parameters are answered identically, i.e. deterministically instead of probabilistically (oracle is required to keep track of the calls given to it).

We can then formulate the definition for unforgeability:

*Definition 2.23* (Unforgeability for FS): Let $\mathcal{FS}$ be an FS-scheme as defined in *Definition 2.20*. Also, let $Adv = \langle Adv_1, Adv_2 \rangle$ be a two-tuple of efficient and probabilistic oracle adversarial algorithms. Then $\mathcal{FS}$ is said to be *(weakly existentially) unforgeable against chosen-message attack*, if the advantage for *Adv*, as defined in the game $\mathcal{G}^{UF}_{\mathcal{FS},Adv}$ is negligible.

The forging advantage for $Adv$ is defined as:

$$\Delta^{\mathbf{F}}\left(\mathcal{G}^{UF}_{\mathcal{FS},Adv}\right) = Pr(\mathcal{FS}.\mathbf{Verify}(mvk, x^+, \sigma^*) = 1)$$

For FS-schemes, unforgeability needs to be accompanied by function privacy, defined below.

*Definition 2.24* (Function privacy for FS): Let $\mathcal{FS}$ be an FS-scheme as defined in *Definition 2.20*. Also, let $Adv = \langle Adv_0, Adv_1, Adv_2, Adv_3 \rangle$ be a 4-tuple of efficient and probabilistic adversarial algorithms. Then $\mathcal{FS}$ is said to have *function privacy* (alternatively, *indistinguishability* w.r.t policies), if the distinguishing advantage for *Adv*, as defined below in the game in Figure 12 is negligible.

The distinguishing advantage for *Adv* (essentially a distinguisher) is defined as follows:

$$\Delta^{\mathbf{D}}\left(\mathcal{G}^{IND}_{\mathcal{FS},Adv}\right) = Pr(b' = b) - {}^{1}/_{2}$$

$\mathcal{G}^{IND}_{\mathcal{FS},Adv}$

- $\langle mvk, msk \rangle \leftarrow \mathcal{FS}.\mathbf{Setup}(1^{\lambda})$
- $\bar{\tau} = \bar{0}$
- $f_0 \leftarrow Adv_0(mvk, msk)[\![\bar{\tau}]\!]$
- $sk_{f_0} = \mathcal{FS}.\mathbf{Keygen}(msk, \mathrm{desc}(f_0))$
- $f_1 \leftarrow Adv_1(mvk, msk)[\![\bar{\tau}]\!]$, where $|f_0| = |f_1|$
- $sk_{f_1} = \mathcal{FS}.\mathbf{Keygen}(msk, \mathrm{desc}(f_1))$
- $\langle x_0, x_1 \rangle \leftarrow Adv_2(\cdot)[\![\bar{\tau}]\!]$, where $|x_0| = |x_1| \wedge f_0(x_0) = f_1(x_1)$
- $b \overset{\$}{\leftarrow} \{0,1\}$
- $\sigma_b \leftarrow \mathcal{FS}.\mathbf{Sign}(mvk, desc(f_b), sk_{f_b}, x_b)$
- $b' \leftarrow Adv_3(\sigma_b)[\![\bar{\tau}]\!]$
- **return** $b'$

**Figure 12.** Function privacy security definition for FS [51]

The notation in Figure 12 follows the conventions set earlier in this chapter. Especially the signature $\sigma_b$ is assumed to contain $f_b(x_b)$ as well; and $x_b$ is expected to contain both the index and the message. The restrictions for $Adv_1$ and $Adv_2$ prevent trivial distinguishing of the signature. The latter restriction for $Adv_2$ may seem overly strong, but as per *Definition 2.20* the signature actually contains the function of the message, this is inevitable. We furthermore assume the adversary encodes all available information into its internal state variable $\bar{\tau}$, such that the challenger does not need to repeat the previous inputs to the adversarial algorithm.

The function privacy definition by Boyle *et al*. [51] and the indistinguishability definition in PBS [32] differ only slightly. In addition to their difference in how they describe the policy, the difference lies in whether the adversary needs to announce the policies and messages beforehand (PBS) or allow the adversary to explore the first secret key before querying the second (the basic FS by Boyle *et al*), and additionally whether the adversary is allowed to generate one (PBS) or two (basic FS) messages. We chose to adapt our definition from the basic FS, as it gives (at least seemingly) a little more freedom to the adversary.

Bellare and Fuchsbauer showed [32] that the simulation-based security definition implies function privacy (indistinguishability), and also unforgeability, if the PBS notion of extractability was satisfied. Note, however, that the extractability is used in the proof solely to extract the policy efficiently from the message, signature and witness. Thus, if the function definition is in itself efficiently decidable (as we assume for FS), the notion of simulatability alone seems sufficient to imply unforgeability.

ABS security goals include attribute privacy and unforgeability. The argument from FS applies to ABS as well w.r.t unforgeability extended to the selection of the attributes, meaning that the ABS unforgeability definition also implies collusion prevention.

The UF-game for ABS is nearly consistent over literature: many schemes ([11], [139], [159], [193]) present practically the same security model. The model is the same as for FS par minor changes. Due to the generalizations used in FS for ABS (e.g. embedding the index (attributes) and predicates (policy formulas) inside the message and function / signature, respectively) we present the game here in full and with ABS terminology for clarity. Note specifically that ABS **Keygen**()-function requires attributes (index) instead of the policy formula. In this case the function descripton given as an argument to the FS **Keygen**()-function should be interpreted to be expressed in a set of attributes that satisfy the (implicitly) given policy description.

In the ABS context, producing a valid challenge signature for a message and a policy that has been queried from the oracle that is *different* from the signature returned by the oracle, is not considered a forgery (rather a violation of attribute privacy).

Attribute privacy is a property of "true" ABS: it is defined as a security goal in the works of Maji *et al.* [139], Okamoto and Takashima [159], and Anada *et al.* [11], but not in earlier works. From these three, Okamoto and Takashima [159] consider only singleton attributes in the privacy game, whereas Maji *et al.* [139] and Anada *et al.* [11] generalize this to attribute sets.

*Definition 2.25* (Unforgeability for ABS): Let $\mathcal{ABS}$ be an ABS-scheme as defined in *Definition 2.21*. Also, let $Adv = \langle Adv_1, Adv_2, Adv_3 \rangle$ be a triple of efficient and probabilistic oracle adversarial algorithms. Then $\mathcal{ABS}$ is said to be *(weakly existentially) unforgeable against chosen-message attack*, if the advantage for *Adv*, as defined in the game $\mathcal{G}^{UF}_{\mathcal{ABS},Adv}$ in Figure 13 is negligible.

The forging advantage for $Adv$ is defined as:

$$\Delta^{\mathbf{F}}\left(\mathcal{G}^{UF}_{\mathcal{ABS},Adv}\right) = Pr\left(\mathcal{ABS}.\mathbf{Verify}\left(mvk, \langle i^*, x^* \rangle, \sigma^*, \mathrm{desc}(\pi^*)\right) = 1\right)$$

The notation in Figure 13 follows those in previous definitions and ABS definitional notation. In particular

- $P$ is the predicate (policy formula) space and $I$ the index space (attributes) as defined in the ABS definition
- $r$ is the game-internal indexing variable (chosen for notational convenience to be different from *l*)

$\mathcal{G}^{UF}_{\mathcal{ABS},Adv}$

- $\langle mvk, msk \rangle \leftarrow \mathcal{ABS}.\mathbf{Setup}\left(1^{\lambda}\right)$
- $\langle r, \bar{\tau} \rangle \leftarrow \langle 0, \bar{0} \rangle$
- **do**
  - $r \leftarrow r + 1$
  - $i_r \in I, \pi_r \in P \leftarrow Adv_1(\cdot)[\![\bar{\tau}]\!]$
  - $\langle i_r, x_r \rangle, sk_{i_r} \leftarrow Adv_2^{\mathcal{ABS}.\mathbf{Keygen}(msk,\cdot)}(i_r)[\![\bar{\tau}]\!]$, such that $\pi_r(i_r) = 1$
  - $\langle \sigma_r, \mathrm{desc}(\pi_r) \rangle \leftarrow Adv_3^{\mathcal{ABS}.\mathbf{Sign}\left(sk_{i_r},\cdot\right)}(\mathrm{desc}(\pi_r), \langle i_r, x_r \rangle)[\![\bar{\tau}]\!]$
- **while** ($T$ = **FALSE**)
- **return** $\langle \langle i^*, x^* \rangle, \pi^*, \sigma^* \rangle$, s.t. $\langle \langle i^*, x^* \rangle, \pi^* \rangle$ has not been queried from the $\mathcal{ABS}.\mathbf{Sign}()$-oracle.

**Figure 13.** ABS unforgeability, adapted from ABS by Maji *et al.* [139]

*Definition 2.26* (Attribute privacy for ABS): Let $\mathcal{ABS}$ be an ABS-scheme as defined in *Definition 2.21*. Also, let $Adv = \langle Adv_0, Adv_1, Adv_2, Adv_3 \rangle$ be a four-tuple of efficient and probabilistic adversarial algorithms. Then $\mathcal{ABS}$ is said to have *attribute privacy* if the distinguishing advantage for $Adv$ as defined below, in the game in Figure 14 is negligible.

The distinguishing advantage for $Adv$ is defined as follows:

$$\Delta^{\mathbf{D}}\left(\mathcal{G}^{IND}_{\mathcal{ABS},Adv}\right) = Pr(b' = b) - {}^{1}/_{2}$$

$\mathcal{G}^{IND}_{\mathcal{ABS},Adv}$

- $\langle mvk, msk \rangle \leftarrow \mathcal{ABS}.\mathbf{Setup}(1^{\lambda})$
- $\bar{\tau} = \bar{0}$
- $\pi, i_0 \leftarrow Adv_0(mvk, msk)[\![\bar{\tau}]\!]$, where $i_0 \in I$, and $\pi(i_0) = 1$
- $sk_{i_0} = \mathcal{ABS}.\mathbf{Keygen}(msk, i_0)$
- $i_1 \leftarrow Adv_1(mvk, msk)[\![\bar{\tau}]\!]$, where $\pi(i_1) = 1$
- $sk_{i_1} = \mathcal{ABS}.\mathbf{Keygen}(msk, i_1)$
- $x \leftarrow Adv_2(\cdot)[\![\bar{\tau}]\!]$
- $b \overset{\$}{\leftarrow} \{0,1\}$
- $\langle \langle i_b, m \rangle, \sigma_b \rangle \leftarrow \mathcal{ABS}.\mathbf{Sign}(mvk, desc(\pi_b), sk_{i_b}, x)$
- $b' \leftarrow Adv_3(\sigma_b)[\![\bar{\tau}]\!]$
- **return** $b'$

**Figure 14.** Attr. privacy definition for ABS, adapted from two ABS schemes [139], [11]

The notation in the figure above follows that of previous ABS definitions and the UF-ecurity game.

In the ABS by *Anada et al.* [11] the authors also accept as challenge signatures such attribute sets that do *not* satisfy the chosen policy formula. While this is a stronger notion of security, it is *too* strong in our case, since for those attribute sets that do not satisfy the policy, the whole signature is irrelevant in any case.

In contrast to FS function privacy, only one policy and message are selected in the attribute privacy game. This is due to the fact that ABS is not designed to hide the actual message to be signed, nor the policy (and these can be seen readily from the output of the signature algorithm).

We have further generalized the ABS security games from those by Maji *et al.* in the sense that we have not defined whether **Keygen**() is executed at the signature trustee, or whether **Keygen**() for different attribute sets are executed under the same attribute authority. These choices induce only minor changes to the security games and neither weaken nor strengthen the actual model.

Speaking of FS-schemes in general, unlike for FE, we are not aware of any results showing implications or equivalence from ABS (or other public policy signature scheme) security notions to FS or derivatives. However, Fuchsbauer *et al.* [32] show that simulation-secure PBS schemes can be used to construct likewise secure ABS schemes.

# 3. Conventional Cryptographic Access Control

## 3.1 General

CAC was first introduced as one possible solution to the MLS in the 1980's [6], during the time of a proliferation of MLS implementations[31]. Already in the work of Akl and Taylor in 1983 [6], the authors had identified their solution as an access control method, and realized the benefits of CAC, namely independency of the storage media and of the trustworthiness of the people managing it (assuming the keys are not available to them); additionally also the independency of data-in-rest vs. data-in transit was noted (indicating application-level encryption). As the notion of public-key cryptography had been recently invented in the open cryptology community (RSA was introduced in 1978 [178]), the first CAC's already embraced public-key encryption in trying to solve the CAC problem for MLS. It was also realized early on that cryptography alone cannot solve all of the MLS requirement areas and that current solutions were not perfect either.

The work by Akl and Taylor [6] represents an example of so-called *hierarchical encryption* with public key schemes. The concept refers to a hierarchical key-management practice, where users having their unique keys are related to each other in some hierarchical (computationally non-reversible) way. Hierarchical encryption (formalized currently under the concept of hierarchical key assignment schemes, HKAS) was for a long time the mainstream for theoretical CAC schemes, and examples abound ([6], [7], [59], [103], [138], [188], [218], [175], [125]). Most schemes assumed a strictly hierarchical structure for the access policy (for each two subjects, one is always "above" the other), but this has since been overcome. All of these schemes are also meant for confidentiality policies only.

Since hierarchical encryption, the advent of more flexible public-key cryptography, notably different forms of FE, have transformed the field. Research has remained in the MLS area (or the corresponding theoretical

[31] Thus, for MLS, cryptography was not seen as a "last resort" but a viable alternative among other possibilities right from the beginning

IFC policy models) as well, but also more general, practical and modern approaches have been proposed. We will look at some of the other modern schemes and solutions related to our problem as well as the more general schemes.

Of independent interest are the solutions in the world of digital rights management, as its publishing model closely mimics that of early MLS, and can be, in some circumstances, used also for MLS.

A fairly recent innovation called blockchains ([101], [191], [151]) are mentioned further in this work. Blockchains are, in essence, a distributed, cryptographically verified chain of events. Blockchains are not a general solution to access control directly, but they are able to solve some important problems within CAC, and thus discussed separately at

## 3.2 Schemes for Digital Rights Management

The DRM concept refers to access control technologies intended to protect intellectual property. DRM is mostly concerned about copy protection, which is easily translated to a confidentiality-policy (for example, Advanced Access Content System, AACS [3]). In some cases copy protection is enforced through employing integrity policies, such as digital watermarking [70] and authenticated computing platform startup mechanisms [98].

Many of the schemes built for confidentiality policies use a publishing scheme similar to MLS environments, which makes them useful for some special cases of CAC: the usual scenario for DRM is to compile rarely modified versions of content (e.g., games, music albums) and then publish that to multiple subscribers for a (long) period of time. The DRM security policy is basically aimed at preventing anyone but authorized distributors to act as a publisher and to revoke certain subscribers from the distribution list. This scenario is very reminiscent of how official, classified documents work: after a certain preparation time, the documents become official, are assigned a classification and stored in databases in well-defined environments. Copying (at least more highly classified) documents outside the official delivery system is in general forbidden, and users may also switch jobs, thus losing their clearances and requiring revocation. In a general case, modification to a published, official

document creates merely a new version of the document, thus only prompting the publication of a new version of the document.

Due to the need of bandwidth and computational efficiency in consumer market, the DRM schemes used for publishing content based on a confidentiality policy are typically based on symmetric key mechanisms. Indeed, there are multiple encryption and key-management schemes based on symmetric-key broadcast encryption. These include, for instance the matrix-based key-management in Content Protection for Recordable Media (CPRM [162]) and Content Protection for Pre-recorded Media (CPPM [199], [200][32]) and different hierarchical settings, such as logical key-hierarchy schemes [203], [207] and NNL-trees [152]. The secret key broadcast encryption mechanisms offer an advantage over conventional PKI because of greater receiver anonymity and independence of the key-channel is achieved.

A typical example of a DRM-scheme is the NNL-tree by Naor, Naor and Lotspiech [152]. The scheme groups potential users into a tree-hierarchy, and gives each user approximately $\frac{1}{2}*t^2$ separate keys, where $t$ is the depth of the tree. With these keys, a user can calculate the keys for certain subsets of legitimate users, and the document encryption key is enciphered with only those keys that are owned by groups of legitimate users. Thus the key management, although optimal in this particular category of schemes, is still somewhat burdensome.

DRM symmetric key mechanisms have their conventional application areas, but moving from write-once-read-many paradigm to a more general access control, and especially moving from under single administration to distributed control[33] makes the use of most DRM schemes in modern, pervasive CAC too cumbersome.

Quite many DRM implementations are forced to hide the key in some hidden, but unencrypted format together with the content. This does qual-

[32] Subsequently broken in Borghoff *et al.* [49]

[33] For example, for the NNL trees the root is equal to a key server. To be able to use secret key mechanisms across different administrative domains, each key server would have to ask encryption services from other domains' servers. This is because a user cannot encrypt the actual content keys himself (he doesn't know all the possible subset keys), so he has to send them for the key server. If the subset is from a different domain, the server would have to forward this to another server.

ify them to be CAC schemes, but not very good ones, as the key embedding may lead to total compromise of the schemes [30], [134], [1].

In a more general setting, the broadcast encryption task can be seen as a special case of CAC, which implies that "mere" broadcast encryption is not always sufficient for the more general CAC. This can be seen readily in asymmetric broadcast encryption schemes (which are designed also specifically for that purpose [45], [71], [69], [87], [184] or primarily in order to have efficiency gains in the number of supported users [116]), and more clearly in some general FE schemes designed specifically for CAC [23], [219], which imply broadcast encryption schemes as well.

## 3.3 Schemes for Information Flow Control

The CAC-schemes have historically been primarily trying to solve the "MLS-problem", and thus there are many examples of schemes for IFC policies. We will cover here three of the latest concepts that most closely concern the approach in this work: modern HKAS, CBIS and Object-Level Protection (OLP).

### 3.3.1 Hierarchical Key Assignment Schemes

Hierarchical key assignment is a method to assign an encryption key and some private information to a hierarchy class [20]. The hierarchy classes stem from different user clearances, which can be used to organize users into hierarchical categories according to their access rights. Access is controlled by encrypting the content and distributing the encryption keys in a controlled manner. The private information annexed to users at a certain level of the hierarchy is used by users on that level to compute actual encryption keys for users beneath them.

Despite the long history behind HKAS, they were first formalized and categorized (into five different types) only in 2006 in Crampton's work [64]. Formal security for HKAS appeared at around the same time [19]. As noted by Crampton [64], typical hierarchical key-assignment schemes were devised for a particular security policy type (deep, but narrow hierarchy or vice versa), resulting in efficiency compromises in different types of key material needs.

Modern HKAS can be made quite expressive, as proven by Atallah *et al.* [19], who address role hierarchies and Ateniese *et al.* [20], who employ time-based constraints in the key assignment (note, though, that the time reference is based on a trusted-third-party binding the keys with a time stamp). However, as HKAS are only concerned with content itself, they cannot be considered pervasive.

HKAS have also been mapped to the RBAC-concept a few times. Examples include the "role key hierarchy" (RKH) model by Zhu *et al.* [220] and a mapping of HKAS to RBAC policies by Crampton *et al.* [65]. RKH is a version of HKAS intended for the Core RBAC with hierarchy. In RKH the encryption keys are associated directly with different RBAC elements, mainly groups and users. The paper in [220] presents three IBE instantiations for role-based encryption assuming keys managed in RKH: encryption, signing and authentication, with a type of revocation solution. The encryption instantiation directly translates in CAC-style enforcement of the **`read`** permission[34], but the situation for the **`write`**-permission is markedly more complex than just signing documents based on a role-key[35].

Crampton *et al.* describe their mapping [65] via showing, how a core RBAC policy can be transformed into an IFC policy. The authors show that those RBAC confidentiality policies that can be fulfilled with (the basic version of) CP-ABE [35], can also be realized with some (symmetric-key) HKAS[36].

Using HKAS for RBAC in confidentiality-only policies enjoys the benefits of having a wealth of existing direct implementations and practices backing them up. This includes HKAS revocation, which has basically been optimized as far as it can be. On the other hand, HKAS still suffers from the burden of relatively large set of keying material and cumbersome key-update procedures [219] and seems to be close to its expressive limits with extended RBAC features and pervasive CAC.

[34] This is indeed also demonstrated with an encrypting file-system

[35] More specifically: the signature scheme is assumed to have group-signature type properties, such as the Trace-function to uncover actual users behind the role.

[36] The converse is unfortunately not addressed, so it remains an open question, whether HKAS would actually be more expressive in some sense than CP-ABE

### 3.3.2 Content-Based Information Security

The concept of CBIS was coined in 2000 by a US Department of Defence advanced concept and technology demonstrator (ACTD) aimed to solve cross-domain information security issues. The ACTD lasted up till 2005 ([147], [190]). The demonstrator was the first "official" CBIS system in the sense that it defined the concept of operations, proof-of-concept and user requirements, including system description, user roles, system management responsibilities and strategy for deployment. However, the ACTD was not built from scratch either, but was based on other similar developments such as constructive key management [2] and MITRE Hexagon [146][37].

The defining idea in CBIS was to protect individual data elements with cryptography, according to the CAC principles, but with a more fine-grained control than what typically was manifested in other contemporary CAC implementations. The ACTD expressed need to exert fine-grained control over content and encode access control policy elements into key material. No pervasive elements, such as structured documents were present, though. The publishing model was MLS-based, meaning that typical sanitization processes were required for the content and security labels were semi-automatically created [146].

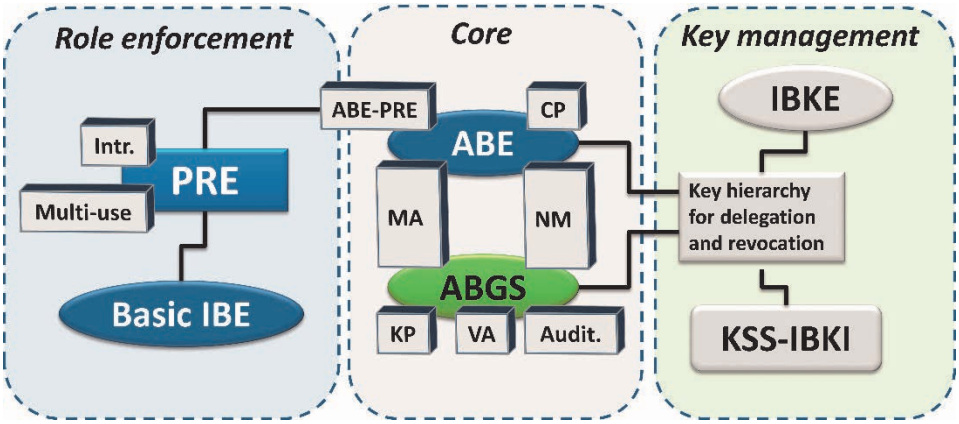



**Figure 15.** CBIS cryptographic architecture according to Kiviharju [122]

[37] MITREs CMHP (Coalition MLS Hexagon Prototype) apparently served as the direct basis for CBIS ACTD PoC (http://forum.prisonplanet.com/index.php?topic=89041.0)

Although the CBIS ACTD finished in 2005, the idea was continued in other countries (Canada [190], Finland [122]) and corporations, such as IBM [67] under different names, such as “data-centric security” (IBM’s approach), “object-based security” (a Swedish concept) and “Object-Level Protection” (NATO, see below). The various approaches are covered more closely in the Finnish CBIS initiative [122].

The main context for this work comes from the Finnish CBIS initiative [122], which gives definitions, requirements and a preliminary architecture for CBIS built on structured documents and enforced with IBE-schemes and their derivatives for environment with disruption-prone data connectivity. The study found “identity aggregation mechanisms”, including, e.g., ABE-schemes, particularly promising for CBIS. However, it was also found that at the time of writing (2008) many ABE-schemes had just recently appeared and were not mature enough for many of the desired functionalities. The study also touched some of the access control model principles, and found that there were no sufficiently flexible cryptographic mechanisms for many of the access control functions (e.g., role binding).

Some of the elements of the cryptographic architecture for CBIS, depicted in Figure 15, are listed below:

- Selection of ABE as the first choice in implementation. This choice has turned out to be more because ABE enjoys many of the benefits of IBE rather than because of expressive constraints of other types of choices. This is discussed more in [123].
- Selection of the basic CP-ABE [35] for the core scheme, for the reason that the encryptor has the main protection responsibility and thus the need to choose the decryption policy and that more advanced schemes with more complete security models were yet to come.
- ABE constructions were required to be both multi-authority (MA) and non-monotonic (NM). The multi-authority requirement states that attributes from multiple issuing authorities should be able to be used together; and the non-monotonic requirement concerns policies containing negative clauses. The multi-authority has remained an essential requirement, although at the time there were

no schemes for completely independent authorities. Non-monotonicity existed then only for key-policy schemes, but later on it became clear, that the particular feature was not that necessary in general (negative clauses in high-level policies are problematic in any case[38]) nor as a specific scheme[39].

- ABGS [115] were proposed, with similar extensions needed for non-monotonicity, multi-authority and other desirable properties for group signatures. However, after the advent of "true" ABS, some of these requirements have become obsolete. ABGS were required to be key-policy (KP), with verifiable attributes (VA) and auditable.
- Proxy re-encryption (PRE [55]). It was considered at the time that in order to enforce the user assignment operation in RBAC cryptographically, it would be necessary to transform the ciphertext such that it could be encrypted with the role credentials but opened solely with user credentials. This has turned out to be questionable during the course of this work [119], but the idea itself can be recommended for other reasons (in order to have PDP independency in the publishing process[40]).
- For key management, such schemes as identity-based key insulated encryption (IBKE, [102]) and identity-based key issuing (KSS-IBKI, by Kumar, Shalaija and Saxena [127]) were proposed.

The Finnish CBIS study [122] was accompanied by a proof-of-concept implementation using Commercial-Off-The-Shelf technologies. Based on

[38] The problems are, for example, mapping the effect of local negative statements throughout the rest of the policy (may cause unwanted side-effects and even conflicts) and difficulty of policy maintenance.

[39] Negative clauses can be broken down to a monotonic combination of positive and negative variables, using DeMorgan's rule. Then it is possible to use separate scheme-level attributes for positive and negative variables. This has an effect on the size of the accepted access sets, number of attributes and in general the ciphertext size. Space-efficiency is technique-dependent, though, and can be optimized fairly well. Non-monotonicity also causes problems with hierarchical delegation schemes, which causes issues with RBAC revocation.

[40] In the world of reference monitors, policy changes usually reflect changes in the PDP only, not in the objects themselves. However, in CAC the PDP function needs to be distributed into the objects, prompting the question, whether objects need to be re-encrypted when policies change. This is not necessarily true, but outside the scope of this work (discussed further in the conclusions).

the results in that PoC a roadmap was presented, which starts from traditional (reference-monitor-based) access control and moves through basic (conventional) PKI-implementation towards ABE-implementations, the final stage being fully RBAC-compliant CAC with ABE at its core. An XML-schema [120] resulted from a need to define a stable document structure supporting both PKI and ABE in the transition phase from PKI to ABE.

### 3.3.3 Object-Level Protection

The OLP concept was developed in the NATO CI Agency [17], [164] and [163]. According to Oudkerk and Wrona [164], the goal of OLP was specifically to enable the use of information classified on multiple levels, and across multiple domains. The main ideas of OLP are [164]:

- Protection is applied to individual data objects instead of their collections (this is the CAC main premise, although OLP does not make a distinction between enforcement methods at this stage [164]).
- Metadata is bound to data objects and is used by protection mechanisms to deduce the actual enforcement requirements for that object. In this sense, the requirements are very similar to our concurrent work [120], [121].

OLP is a system-wide standard approach to data protection [164]. It includes an information-architecture, a model [163] and a type of roadmap via different scenarios, called *evolution stages* [164]. Each evolution stage is further divided into *integration levels*, implying the main characteristics of progressively more advanced implementations of a certain evolution stage.

OLP evolution stages are characterized by their different MLS-related capabilities, called *dimensions*. Each dimension describes certain similar high-level concepts, which form progressive “steps” for an evolution stage to employ in order to better realize the OLP concept as a whole. The dimensions, depicted in Figure 16, include:

1) Level of Object Protection: starting from the usual information domain separation and ending in CAC. An intermediate level is

called deny-or-grant-access (DOGA), which is basically reference-monitor-based access control of individual objects.

2) Granularity of access control: starting from MLS-security levels (without compartments) and ending in ABAC

3) Detail of content description: starting from lack of any description and going through security labels to content properties (as a metadata).

In the OLP dimensions the most advanced form of OLP is described by the tuple <CAC, ABAC, metadata>, which forms a subset of what we are independently pursuing with the pervasive CAC concept in this work.

One of the evolution stages in OLP is called *Content-based Protection and Release*, or CPR. CPR in this framework is defined by <*, ABAC, metadata>, where * = DOGA or CAC. In the OLP concept [164] CBIS-like requirements have been given to the use of CAC, and the authors mention having 18 different variants for CAC in OLP. Of these variants, the use of ABE was considered the most advanced.

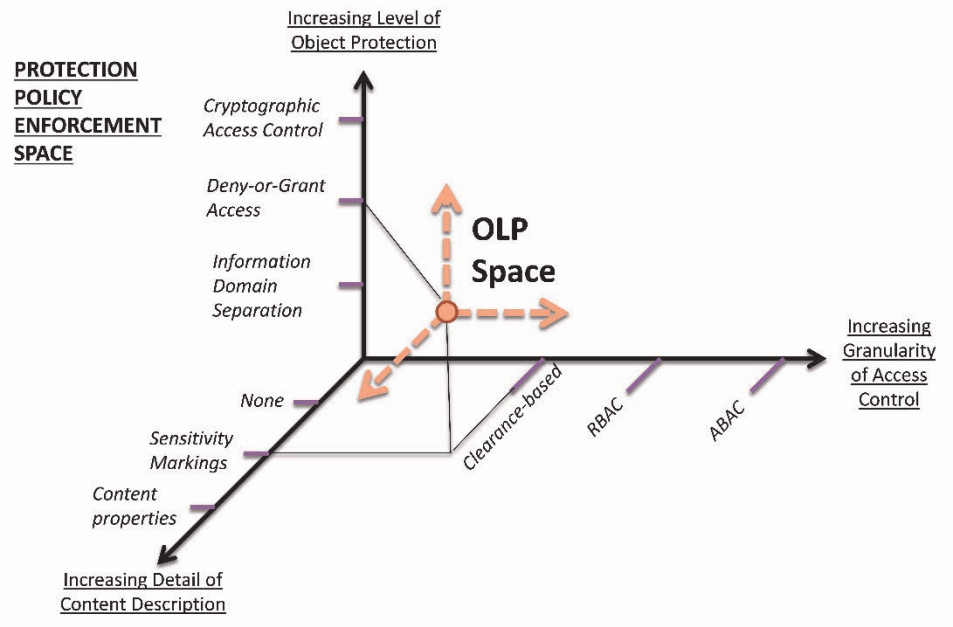



**Figure 16.** OLP dimensions, according to Oudkerk and Wrona [164]

The OLP concept integration levels contain the actual functionalities, called component classes, which include, e.g., key management, policy enforcement mechanism, labelling service and metadata binding service. The CPR model is elaborated more in the works of Armando *et al.* [18] and another paper by Oudkerk and Wrona [163], an example scenario shown in Figure 17. According to the CPR model [163] (corresponding to a currently implementable integration level), a service called CPR Enforcement Separation Service, or CPRESS, acts as the mediator between these component classes. CPRESS is basically an implementation model for ABAC, making access control decisions based on user, computing platform and content attributes matched to (release) policies. In the XACML terms, the CPRESS acts as the PEP. If the CPRESS is implemented as a CAC service [163], it should function much like CP-ABE does: encrypting different content elements (assuming a structured document) with different policies and publishing the whole set of resources, from which recipient with sufficient credentials (or private key material) can then decrypt those portions they are allowed to access.

In MLS, in order to view a classified document, both a cleared user and secure location are required. For this reason, the release policy in CPR needs to combine these two policies. This is taken into account in the CPR-CAC [163], where CP-ABE is considered for the CAC-level implementation of CPR. Although it seems this is a problem of multiple authorities in ABE, it is in fact the opposite: the combination becomes a problem of "pooling" attributes. The solution presented in CPR-CAC [163] is a precursor to the work in [117]. The solution is elaborated more in the chapter about CRBAC confidentiality enforcement and its performance estimated in the implementation considerations.

In contrast to the MLS line of thought, present even with the CBIS concept, CPR does not consider sanitization of "released" objects (objects for which access has been granted), but instead carries everything in the same object, leaving those parts encrypted, which cannot be decrypted by the recipient. The reasons cited for this type of functionality in CPR-CAC [163], include information management easiness (especially versioning), publish-subscribe model and operations other than decryption performed for the data.

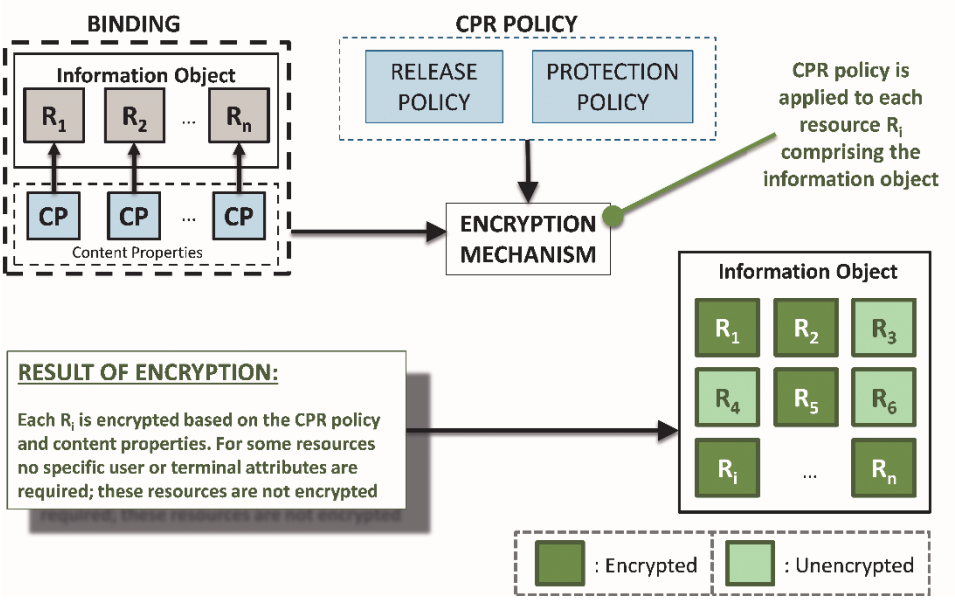



**Figure 17.** CPR-CAC, according to Oudkerk and Wrona [163]

CPR and OLP are geared towards confidentiality policies (although the implicit assumption seems to be that other types of access may also be possible to enforce with encryption only). On the other hand, a very similar line of thought to this work appears in the OLP component class for binding metadata, where it is stated that a cryptographic binding using ABS is the long-term solution to the binding problem.

## 3.4 Generalized CRBAC Schemes

The acronym CRBAC in conjunction with cryptographically enforced RBAC was introduced only in 2010 by Crampton [65]. The actual definition of CRBAC depends on the usage, however:

- There are multiple different RBAC "models" ranging from the very basic idea of Sandhu *et al.* [185] to the $RBAC_3$-standard, which – with all its extensions – starts to border on ABAC.
- Even given the underlying access control model, there are variations in what exactly is enforced cryptographically, and whether the enforcement follows the CAC principles (for example, the work by Crampton and Lim [66] appears at first glance to enforce

integrity policies with CRBAC, but in reality it concentrates on verifying general authorization requests with hierarchical identity-based signatures, which are bound to a role structure).

There is a difference between actual CRBAC and the individual cryptographic building blocks used to construct such a scheme. For example, even though we make use of XML Encryption and Signatures [120], those actual standards are merely methods to encrypt / sign objects, and thus not considered to be CRBAC. In general the cryptographic parts of Web Services Security [156] are rather agnostic to the actual access control model (although the whole Web Services Security was designed in the ABAC paradigm).

The actual CAC schemes that can be stated to implement CRBAC start from the HKAS in the work by Crampton [65] and the paper by Zhu *et al.* [220] (discussed in the HKAS chapter). Crampton defined CRBAC as a key-assignment scheme [65] capable of restricting the **`read`**-access according to the RBAC model defined by Sandhu *et al.* [185]. Zhu *et al.* [220], on the other hand, add role hierarchies to this and are not restricted to **`read`**-permissions; however, some of their use for the hierarchical keying (authentication tokens) is clearly outside the scope of CAC as intended here.

The work of Ferrara *et al.* [77] takes a similar approach to CRBAC to the one presented in this work (independently to [119]). According to the authors, a CRBAC system "is defined by the algorithms executed by parties, when engaging in the different actions stipulated in the RBAC model". Thus CRBAC is defined by a *mapping of cryptographic schemes / algorithms to the different RBAC functionalities.* The mapping presented by Ferrara *et al.* [77] is based on a specific predicate encryption scheme (which can naturally be implemented with ABE schemes as well) using threshold predicates with the threshold size being $\geq 0$, or equivalently that there is at least one common attribute in the attribute set used in ciphertext and corresponding decryption key creation. In contrast to the approach in this work [119] the method by Ferrara *et al.*, called PE for non-disjoint sets or PE-NDS:

- The expressive power of the schemes in PE-NDS involves only disjunctions of attributes

- The model assumes direct protocol between all the players and does not assume intermediate stores. Thus, for example, permission management always includes re-encryption (both in granting and revoking), whereas in [119], encryption and decryption are deferred up till the RBAC `CheckAccess()`-function.
- The concept of a role is an attribute set much like what we have envisioned, but the meaning of the attributes is different: in PE-NDS, the attributes encode also the RBAC UA-relation, whereas in our model they describe the role's capabilities and functions within the system.
- PE-NDS is based on a complete view on the CRBAC **`read`**-access security-wise, giving concrete provable security for the whole CRBAC system.

Another scheme that is built directly to form a CRBAC system, is the "Role-Based Encryption" by Zhou, Varadharajan and Hitchens [219], referred to as ZVH-RBE here. ZVH-RBE uses identity-based broadcast encryption (IBBE)[41] as the underlying enforcement scheme. ZVH-RBE is based on a simplification of the core RBAC, where the user assignment (to a role) is a sufficient credential to gain access to those objects, for which the role has permissions. Hierarchies are considered, but user assignment to multiple roles will cause revocation complications.

ZVH-RBE considers only access enforcement for user assignment, i.e. for the cases of adding, having and deleting a user from a role. This enforcement, however, satisfies certain pervasive CAC principles:

- For encryption, the only key material needed is that of the role.
- For decryption, only per-user key material is required.
- Adding and removing a user changes existing key material
- Removing a user does not affect other users of the role (no extra key material for non-revoked users needs to be sent) or predecessor roles in the role hierarchy

[41] This variant of IBBE is not, despite the name, actually identity-based. The defining characteristic of IBE is that encryptor needs only know system-wide parameters and the naming scheme to encrypt to arbitrary user. In the ZVH-RBE, encryptor needs additional role-specific public-key material to encrypt.

- New users have access to previously sent material (could be a security issue in some cases, though)

The ZVH-RBE scheme basic idea is to rewrite the role public key material each time a user assignment is changed. This, however, leads to the need to re-fetch role public-key material every time something is encrypted. Additionally after revocation, the system global parameters would need to be refreshed.

As a CRBAC scheme, ZVH-RBE is somewhat simplistic. Its merits lie more in the pervasive CAC principles of separating role and user keying, accomplishing yet another type of cryptographic enforcement of the UA function in RBAC.

# 4. CAC and Public Key Authentication Architectures

One of the basic benefits of public key cryptography in general has been that since one part of the key can be public, messages could be sent merely by looking up receiver identities and their public keys much like the conventional phonebooks. However, in the cyberspace it is much more convenient to merely do automatic lookups or ask for public keys remotely (from more varied sources) than to view human-readable directories.

Whereas conventional phonebooks offered a reasonable amount of authenticity (based on the format and delivery method), the public key lookup methods in computer networks are more vulnerable to authenticity attacks.

The actual weak link in the public key authenticity is the association between the public key and actual user identity. To make this link stronger, different cryptographic mechanisms have been suggested, with three main categories:

- Trusted external third party signatures on the combined record of identity and public key, called explicit certificates (conventional PKI)
- Pure identity-based cryptography (IBC), where the identity and public key (components) are equated
- Implicit certification schemes (ICS), where the authenticity is attested for by other means than explicit certificates, but which does not equate the identity to the public key directly.

We call these categories public-key authentication architectures[42]. Nearly all of the existing FE (and FS) schemes are actually IBE- (or IBS-) derivatives and share many of the benefits and drawbacks of IBC, such as having a single-point-of-failure, called the master key, and making revo-

[42] We only consider such cryptographic public key authentication mechanisms, which use cryptographic evidence included in the key material. Thus we consider many schemes to be outside the scope of this categorization, such as, for example, protocol-based solutions that use trusted public-key repositories, like DNS-servers in DomainKeys Identified Mail (DKIM, specified in IETF RFCs 5585 and 6376), or XML Key Management (XKMS, [78]).

cation challenging (as public key revocation implies revocation of the identity as well).

ICS schemes are more varied in the basic techniques compared to what is used in IBC, in which the schemes are very often constructed with bilinear groups. The different ICS variants include examples such as certificate-based encryption (Gentry [85]), self-generated certificates (Liu, Au and Susilo [135]), actual implicit certificates (Pintsov and Vanstone [173], also Brown, Gallant and Vanstone [52]), certificateless schemes (Al-Riyami and Paterson [10]), and self-certified keys (Girault [88]). Also implicit signatures are possible (Lee and Kim [131]). Of these, the certificateless schemes have proven to be most useful, due to their embracing of IBE techniques. Due to revocable key elements present in different ICS schemes, the IBE schemes' escrow and revocability problems are alleviated to some degree. Nevertheless, the certification entity architecture is far simpler than in conventional PKI.

When public-key systems are used for CAC, experience with early HKAS and CBIS has shown that conventional PKC, such as the different PKI architectures can only go so far. The usual drawbacks in existing systems come from laborious key management, weak scalability w.r.t access control policy expressiveness [66] and poor support for all of the versatile requirements in modern access control. In practice, to incorporate multiple different access control functionalities to be implemented with cryptography, the construction will have to integrate multiple rather incompatible schemes. An example from an advanced system along this line of thinking can be found in the work of Popa *et al.* [174].

The PKAA used has indeed turned out to have major implications for the possible functionalities of CAC schemes in general. Especially publish-subscribe-environments require different capabilities from the PKAA: Oudkerk and Wrona use a categorization of different requirements for access control enforced by encrypting objects [164] into three: the purpose of encrypting objects in storage (CAC-S), objects in transit (CAC-T) and objects either in storage or in transit (CAC-ST). For CAC-ST, it is required that such a system may not depend on knowing the set of recipients of the encryption in advance. This implies, according to Oudkerk and Wrona [164], either secure key-distribution channels or schemes that sup-

port *encryption to unknown entities*. This type of property is unique to the IBE PKAA only.

In a more recent and formal study [141] Maurer proves that for the constructive cryptography framework (which models cloud storage in general) there does not exist a conventional public-key scheme that could fulfill the security model there. Maurer's CC-model combines both high functionality and stringent security, but functional encryption (from the IBE PKAA) can still be shown to be able to fulfill the new functionality and security notions.

We investigated the **`write`**-permission enforcement for CRBAC with ABS schemes [118]. ABS schemes fall into the IBE PKAAs, and it was noted that compared to conventional signatures, there are differences in the integrity concepts (also referred to in the research questions 3b and 4a).

- In the conventional PKI signatures, the signature verification answers the questions "*Did the claimed entity produce this signature for this particular message?*" This is origin authenticity combined with data integrity. If additional integrity or authenticity notions, such as content validity or signer authorization, are required, they need to be associated with the origin authenticity.
- In ABS, the authorization concept can be separated from the data integrity by making deductions based on the available attributes and policies. This is a clear benefit for access control functionality, as more integrity questions can be answered.

For the ICS PKAA, we have not been able to pinpoint any attribute-based signature schemes, making the ICS a poor candidate for CAC integrity enforcement.

Multiple capabilities of the IBE PKAA were investigated to be beneficial for the environment in the Finnish CBIS study [122]. Although it is possible to transform many schemes from the IBE PKAA to equal schemes in the ICS PKAA[43], for some of the capabilities more conclusive evidence was deemed necessary. To investigate the capability for attribute-based

[43] Dodis and Katz showed [72] that a certain class of IBE-schemes can be mapped to certificate-based encryption schemes.

schemes within different PKAA, the work described in [123] was launched.

The importance of having cryptographic schemes support attributes is quite evident for CAC, especially for CRBAC and ABAC-versions of CAC, since they map real life policy elements more directly and "pervasively" to scheme elements. Within the IBE PKAA, different ABE and ABS schemes already attest to this capability, but the work on the ICS PKAA seemed to focus on the actual concept more than advanced functionalities.

Since the certificateless schemes (called certificateless public-key encryption, or CL-PKE) initiated by Al-Riyami and Paterson [10] were so close to IBE, the simplest test would be to try to translate the "fuzzy" IBE construction of Sahai and Waters (FIBE) in [183] to CL-PKE. Although the fuzzy ideology was initially intended to be used for biometric applications, it also works as KP-ABE for threshold policies (where $n$ attributes out of $m > n$ need to be satisfied)[44], as shown in [123].

In 2007, we presented a scheme for *fuzzy certificateless encryption* (FC-PKC, [123]). An example of its standalone usage was suggested to be biometrics, as it was for its cousin scheme, fuzzy IBE. The reason "fuzziness" (more accurately, threshold gates) is suitable for biometrics is that biometric readings are never 100% repeatable, but still "close" to some reference material (obtained at the biometrics enrollment phase). In contrast, cryptographic keys should be exactly the same down to the last bit of its representation. While the integration of inherently public (biometric) information to other cryptographic systems in general remains an open question, the application for fuzzy schemes is clear: to "correct" sufficiently accurate measurements to their reference value for different applications. Additional applications arise as well, if the scheme is considered from the KP-ABE perspective.

As FC-PKC is demonstrating the feasibility of combining FIBE with CL-PKE, we first give a brief review of both of them. Sahai and Waters' FIBE-scheme actually consists of two versions, a preliminary and the one presented in EUROCRYPT 2005 [183]. Main differences lie in the adoption of a more general set-intersection metric instead of Hamming dis-

[44] Indeed, Sahai and Waters subsequently refer to fuzzy IBE as "the first ABE"

tance of the identities, and addition of a large-universe construction. FC-PKC is based on the peer-reviewed version.

**Table 3.** FIBE algorithms in [183]

<table>
<tr><td colspan="2">Setup</td></tr>
<tr><td>• Input: security parameter $k$ and element universe $\mathcal{U}$ associated with $\mathbb{Z}_p, p = |\mathbb{G}_1|$)<br>• Select a generator $G \in \mathbb{G}_1$<br>• Select a threshold $d$ signifying, how many common elements two different identities need to have</td><td>• Output a master key:<br>o $\bar{t} = t_1, \dots, t_{|\mathcal{U}|}, y \in_U \mathbb{Z}_p/\{0\}$<br>• Output the (system) public key<br>o $\langle \bar{T}, Y \rangle = \langle \{G^{t_i}\}_{i \in \mathcal{U}}, e(G,G)^y \rangle$</td></tr>
<tr><td colspan="2">KeyGen</td></tr>
<tr><td colspan="2">• Input: an identity $\omega \subset \mathcal{U}$<br>• Select a per-user interpolation polynomial (uniformly randomly):<br>o $q(x)$, where $x \in \mathbb{Z}_p$, deg$(q) = d - 1$, and $q(0) = y$<br>• Set the private key as: $\left(\bar{D}_\omega \in \mathbb{G}_1^{|\omega|}\right) = \{D_i\}_{i \in \omega} = \left\{G^{q(i)t_i^{-1}}\right\}_{i \in \omega}$</td></tr>
<tr><td>Encrypt</td><td>Decrypt</td></tr>
<tr><td>• Input: an identity $\omega$', and a message $M \in \mathbb{G}_2$,<br>• Select a per-message nonce (uniformly randomly) $s \in \mathbb{Z}_p/\{0\}$<br>• Output encrypted $M$ as:<br>o $C = \langle \omega', E' = MY^s, \{E_i = T_i^s\}_{i \in \omega'} \rangle$</td><td>• Input: $C$, $\omega$, $\bar{D}_\omega$, $d$, and $\omega$'.<br>• Check, if $|\omega \cap \omega'| \geq d$. Then select some size $d$ subset $K \subset \omega \cap \omega'$ and compute the plaintext message as:<br>o $M = E' \left( \prod_{i \in K} \left( e(D_i, E_i) \right)^{\Delta_{i,K}(0)} \right)^{-1}$</td></tr>
</table>

*Definition 4.1* (Fuzzy identity-based encryption, FIBE): Given a security parameter $k$, a fuzzy identity-based encryption scheme FIBE is a four-tuple of algorithms ⟨**Setup**, **KeyGen**, **Encrypt**, **Decrypt**⟩ defined in Table 3, where:

- $e$: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is an admissible bilinear map for sufficiently large (w.r.t. $k$) bilinear (multiplicative) groups
- $\Delta_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$, (where $\in \mathbb{Z}_p, S \subset \mathbb{Z}_p$ ) is the Lagrange coefficient used for the interpolation polynomial

The *Definition 4.1* refers to the "small universe" FIBE [183]. This corresponds to the case, where $\mathcal{U}$ is sufficiently small to allow enumeration of all its elements. The security model was defined to be chosen-plaintext-attack (CPA) security under selective identity.

The main idea in FIBE is to use polynomial interpolation (in the exponent) to perform "error-correction" of the identity (or attribute) set.

The CL-PKE was initially an architectural modification of the basic IBE of Boneh and Franklin [44], which merely adds a per-user re-randomization factor to the user's private key. The main goal of CL-PKE is to distribute trust (for key element generation) more evenly between the key-generating authority (KGC, or Key Generation Center in IBE and CL-PKE terminology) and user.

CL-PKE models two types of adversaries: the "usual" IBE-adversary, who is able to corrupt identities and receive their private keys, and additionally replace existing public keys. The second type of adversary models a semi-honest or honest-but-curious KGC. The main technical contribution in CL-PKE is to define a reasonable security model for such adversaries.

CL-PKE is one of the first schemes to try to modify IBE as little as possible, but still achieve both revocable and user-defined elements in the scheme. The scheme accomplishes this by introducing extra key elements in a more distributed architecture than in IBE. We cover this architecture in FC-PKC. Because of this similarity, we chose CL-PKE to be the "platform" to embed a fuzzy scheme to.

A direct application for FC-PKC and FIBE is a biometric authentication scheme (although the implications are directly applicable in ABE as well). In biometric schemes rekeying is inherently challenging due to the static nature of the public key (equated to the actual biometric). With FC-PKC keying architecture, it is possible to separate the actual biometric reading from a revocable element such that the latter could be carried, e.g., inside a smart card or other token. This revocable part would nevertheless be tightly coupled with the actual biometric reading.

Unlike CL-PKE, FC-PKC does not attempt to solve the trust distribution problem, but rather the revocation problem. Thus, in the scheme architecture shown in Figure 18, the **Set-Private-Key** algorithm is shown as one, but it can be split into KGC- and user-portions as well, depending how rekeying is to be solved.

The technical contribution in FC-PKC [123] is a simple re-randomization of FIBE key elements, much the same way the CL-PKE modifies IBE

keys. FC-PKC is focused more on the keying architecture and possible implementation aspects.

The FC-PKC is defined according to the *Definition 4.2* below. The architecture depicting the main elements and algorithms in biometrics scenario, equivalent to that of Sahai and Waters' scenario [183] is presented in Figure 18.

The *Definition 4.2* is slightly different from what is described in [123]:

- We changed the notation from the original FC-PKC to this work to a more standardized one by replacing *G* for *P* as the group generator.
- The user's random value $\bar{r}$ in **Set-User-Random-Value** as well as user's public key in **Set-Public-Key** have been corrected in their length from $|\mathcal{U}|$ to $|\omega|$, as the user has no immediate need for elements outside her identity elements at **Decrypt**. The actual need for them is in the **Encrypt**-algorithm, which in this envisioned application is measuring biometric vectors and encrypting them accordingly. It can, however, be assumed that the application is connected to the already enrolled biometric public keys and can thus select the subset $\omega \cap \omega'$. The alternative used in [123] is to use the whole universe, in which case there is no need for the actual listing of the elements inside $\omega$, but $\overline{PK}$ becomes unduly large.
- We omitted a particular $\overline{ce}$-element present in the first version. Liu, Au and Susilo presented a public-key replacement attack against CL-PKE [135] for the main purpose of incurring denial-of-service[45]. This attack was called denial-of-decryption. Our scheme also tried to prevent this attack, but the problem then arises how to define the identity to be guarded. Our approach used the approach of signing every identity element (or attribute), which is rather inefficient. It does not contribute otherwise to the scheme itself

[45] The CL-PKE security model addresses the main concern in public-key exchange, namely that a man-in-the-middle could replace the recipient's public key with a key for which the attacker knows the corresponding private key. This is a confidentiality concern. However, in the CL-PKE encryption phase one cannot verify the actual identity: the main assurance against the public key replacement is that the attacker cannot break the message confidentiality. On the other hand, were the encryption scheme to be used for integrity purposes or in deferred-access environments, this assurance may only be part of what is desired.

either, so omitting that element should not affect the rest of the scheme.

*Definition 4.2* (Fuzzy Certificateless Public-Key Cryptography, FC-PKC): Given a security parameter *k*, an FC-PKC scheme is a six-tuple of algorithms **Setup, Set-User-Random-Value, Set-Private-Key, Set-Public-Key, Encrypt** and **Decrypt** defined in Table 4, where:

- $e$: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is an admissible bilinear map for sufficiently large (w.r.t. *k*) bilinear groups, of which $\mathbb{G}_1$ is additive and $\mathbb{G}_2$ multiplicative.
- $\Delta_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$, (where $i \in \mathbb{Z}_p, S \subset \mathbb{Z}_p$ ) is the Lagrange coefficient used for the FIBE interpolation polynomial

Since the encryption of CL-PKE is so similar to IBE, we chose to keep the FC-PKC encryption closer to FIBE than CL-PKE. There are some differences between CL-PKE and FC-PKC in the use of hash functions:

- FC-PKC maps the message to the bilinear group element rather than vice versa (and uses group operation as the encryption transformation, rather than XOR)
- FC-PKC assumes an implicit map of the attributes to $\mathbb{Z}_p$ instead of an explicit hash function.

The FC-PKC architecture, depicted in Figure 18 enables independent handling of the attributes (or identity elements) and per-user randomization by tying the identity to the exact set of attributes chosen and keeping the randomization in $\bar{r}$. Note also that $q(x)$ serves to protect against the user collusion rather than revocation, and during a re-key operation, $\bar{r}$ is changed, not $q(x)$.

Al-Riyami and Paterson show [10], how to transform several types of identity-based cryptography schemes to the CL-PKE-realm, including a signature-scheme, authenticated key agreement protocol and a hierarchical IBE scheme. According to Al-RIyami and Paterson [10] these schemes inherit the security properties from CL-PKE.

**Table 4.** FC-PKC algorithms, [123]

| **Setup** | |
|---|---|
| • Input: security parameter $k$<br>• Select a generator $G \in \mathbb{G}_1$<br>• Select element universe $\mathcal{U}$ (associated with a suitable $\mathbb{Z}_p$)<br>• Select a threshold $d$ signifying, how many common elements two different identities need to have | • Select a master key (uniformly randomly):<br>o $\bar{t} = t_1, \ldots, t_{\lvert\mathcal{U}\rvert}, y \in \mathbb{Z}_p/\{0\}$<br>• Compute the (system) public key $\langle \bar{T}, Y \rangle \in \mathbb{G}_1^{\lvert\mathcal{U}\rvert} \times \mathbb{G}_2$ as:<br>o $\langle \bar{T}, Y \rangle = \langle \{t_i G\}_{i \in \mathcal{U}}, e(G,G)^y \rangle$ |
| **Set-User-Random-Value** | **Set-Public-Key** |
| • Select user random value (uniformly randomly):<br>o $\bar{r} = r_1, \ldots, r_{\lvert\omega\rvert} \in \mathbb{Z}_p/\{0\}$ | • Compute the (user) public key<br>o $\overline{PK} = \{PK_i\}_{i \in \omega} = \{r_i t_i G\}_{i \in \omega}$ |
| **Set-Private-Key** (KGC portion) | **Set-Private-Key** (User portion) |
| • Select a per-user interpolation polynomial (uniformly randomly):<br>o $q(x)$, where $x \in \mathbb{Z}_p$, $\mathbf{deg}(q) = d-1$, and $q(0) = y$<br>• Compute partial private key for identity $\omega$:<br>o $\overline{sk}_{KGC,\omega} = \{sk_{KGC,i}\}_{i \in \omega} = \{q(i) t_i^{-1} G\}_{i \in \omega}$ | • Input: partial private key<br>• Compute the full private key for identity $\omega$ (via user randomization):<br>o $\overline{sk}_\omega = \{r_i^{-1} sk_{KGC,i}\}_{i \in \omega}$ |
| **Encrypt** | **Decrypt** |
| • Given a user with a public key $\overline{PK}$ and an identity $\omega$', select the parts of $\overline{PK}$ that correspond to $\omega$', referred to as $\overline{PK}_{\omega'}$.<br>• Select a per-message nonce (uniformly randomly) $s \in_U \mathbb{Z}_p/\{0\}$<br>• Given a message $M \in \mathbb{G}_2$, encrypt it as $C = \langle U, V, W \rangle \in 2^{\mathcal{U}} \times \mathbb{G}_2 \times \mathbb{G}_1^{\lvert\omega'\rvert}$, where:<br>o $U = \omega'; V = MY^s; W = \{sPK_i\}_{i \in \omega'}$ | • Given $C$, $\omega$, $\overline{sk}_\omega$ and $d$, set $\omega$'=$U$ and check if $\lvert\omega \cap \omega'\rvert \geq d$. Then select some size $d$ subset $K \subseteq \omega \cap \omega'$ and compute the plaintext message as:<br>o $M = V \left( \prod_{i \in K} \left( e(sk_i, W_i) \right)^{\Delta_{i,K}(0)} \right)^{-1}$ |

The work with FC-PKC also considered the implementation possibilities as follows:

- Elliptic curve groups and suitable pairing implementations were considered as the main tool for bilinear maps.

- One of the reasons FIBE was selected is that it gives a detailed description of the secret sharing scheme used, in contrast to more abstract schemes.
- The (even today a non-trivial) problem of the encoding of real-world attributes into elements in $\mathbb{Z}_p$ was recognized.

As the FC-PKC **Encrypt**-algorithm protects the actual message exactly like CL-PKE par the hash-functions and the actual identity is shared according to the interpolation polynomial used in FIBE, we conjecture the CL-PKE security properties to be inherited to FC-PKC as well. We did not present a security proof for this conjecture, as this was outside the scope of the study.

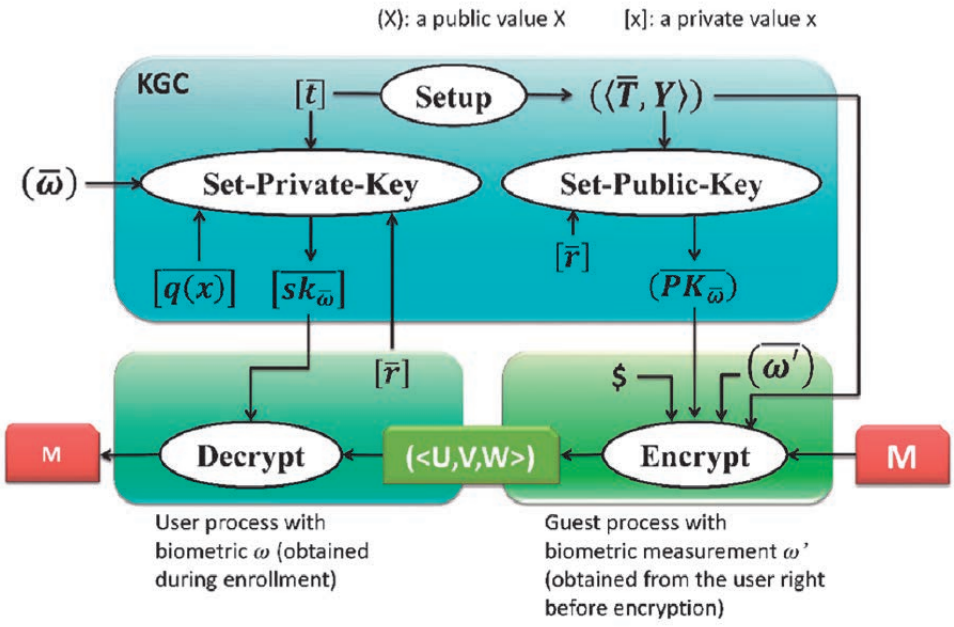



**Figure 18.** FC-PKC architecture and most important elements [123].

To the best of our knowledge, the next time CL-PKE and attributes have been attempted to integrate to each other was in 2013 by Zhang [217]. Compared to FC-PKC, there are also earlier examples of attribute-like behavior for ICS schemes. In particular, the work of Al-Riyami and Waters under "cryptographic workflows" [9], essentially combines monotone secret sharing schemes with CL-PKE. This has similar functionality for

honest users as does the later CP-ABE. However, ABE includes in its security model *collusion resistance for users* (e.g., the CP-ABE by Bethencourt, Sahai and Waters [35]), which means resistance against the case, where two malicious users try to cheat themselves more decryption power by combining their respective attribute keys. The cryptographic workflows - scheme [9] does not consider this[46]. Thus, although visionary in a sense, we do not consider cryptographic workflows to be "pure" attribute-based.

In conclusion, it was found that attributes in general are not particular to the IBE PKAA. This also answers to the research question 2a. Even though the inherent revocation capabilities of the ICS PKAA would be useful for CAC, the proliferation of various ABE- and later FE-schemes together with the property mentioned by Oudkerk and Wrona [164] (encryption to unknown entities), eventually favored the IBE PKAA in order to continue research on that area.

[46] Cryptographic workflows are protected against colluding authorization authorities (AA, attribute authorities in multi-authority ABE) though. To see that the scheme does not protect against colluding *users*, note that those parts of user's decryption key material, which are formed outside the user process (at the AAs), are dependent on the AA's private key (used for all requests) and attribute description only. Thus two users can freely generate compatible private keys. It is possible to alter the naming scheme in such a way that the attribute is padded with the user identification at some trusted third party, but this is no longer a property of the cryptographic scheme.

# 5. CAC and Publish-Subscribe Environments

## 5.1 Background

The work in the Finnish CBIS concept [122] set a roadmap for developing or adopting different cryptographic elements supporting cryptographic access control. The roadmap is given in Figure 19. In around 2010-2011 the work was progressing in the roadmap step three: “MLS-Enhanced PKI”, which is the last stage with conventional CAC. After step three, the RBAC model should be incorporated more deeply to be enforced with PKI and the change in PKAAs from conventional PKC to the IBE-realm will become prevalent.

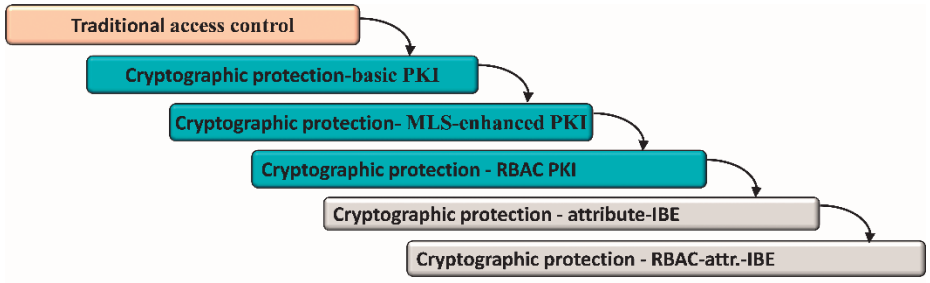



**Figure 19.** Roadmap for CAC from conventional PKC towards ABE schemes according to the Finnish CBIS study [122].

One of the actions taken in the ongoing work then was to model the envisioned MLS-documents and their distribution environment according to (pervasive) CAC principles and limitations in such a way that at least the document format solution would be equally well usable over the PKAA transition. This is due to the fact that moving documents from conventional system to distributed, cloud-based systems is likely to be performed in any case, irrespective of the PKAA used.

The handling rules for classified content combined with conventional official document handling routines nearly always imply publish-subscribe type distribution solutions. This is our basic premise as well, although we relax these rather static requirements to a degree in order to accommodate more dynamic ICT-workflows. We use the word “environment” or “architecture” in the publish-subscribe context as well as in

the access control implementation. These should be understood to represent different views of the same problem: one being access-control- and the other publishing process-centric view. These views are separate but not independent of each other.

Official documents are very typically endowed with different types of metadata, which is naturally described in a structured format, such as XML. Due to this practice and the possibility to describe policies and cryptographic key material as metadata, the document format for CBIS should also be following the XML-paradigm and format.

In this chapter we present the publish-subscribe environment assumptions and limitations presented by CAC for such an environment, how different FE functionalities (or choices in them) generally map to the environment and CAC principles and finally present a PKAA-independent scheme to store MLS-documents.

## 5.2 Environment Assumptions

The publish-subscribe concept is a high-level pattern for information distribution, where information consumers *subscribe* to certain pieces of information, which content producers *publish*. It is considered to be a very flexible model in distributed computing systems due to its many-to-many nature and decoupling of message senders and receivers [216]. The publish-subscribe pattern has its origins in software engineering as a messaging pattern (as in the work of Birman and Joseph [36]).

Current publish-subscribe systems can be typed into two, depending on how subscribers specify their “orders”: topic- and content-based subscription systems (Hiray and Shedge, [108]). In topic-based systems, the publishers assign a single metadata to the content, called “topic”, and the subscriptions are based on that metadata. Content-Based Publish-Subscribe systems (CBPS) are intended to fetch their criteria directly from the content, performed by subscriber-side *filters* [216]. Many schemes aimed at protecting confidentiality aspects of publish-subscribe systems assume CBPS.

The different CBPS roles can be listed as follows (according to e.g Mu, Yuen and Susilo, [216]):

- *Publishers* produce and notify the system about the availability of content. They should not need to know, who will eventually access the content.
- *Subscribers* notify their interest of receiving information to the system. Only information that matches their interest is provided to them. In MLS, the system may enforce some mandatory "interests", such as clearance.
- *Brokers* (in our model also called the Channel or the Cloud) match the publishers' notifications to subscribers' interests and route the information within the brokering infrastructure. *Intermediate brokers* only route packets, and *border brokers* (in our model also called the Filters) perform additional tasks at the edge of the brokering infrastructure, either towards the publishers (called *publisher hosting brokers*, *PHB*) or subscribers (called *subscriber hosting brokers*, *SHB*). The brokers are implicitly assumed also to store the published information – we slightly abuse this assumption by requiring the brokers to keep published documents until further notice.

The main problems in CBPS security revolve around the aim to protect the confidentiality of the content, its metadata and filtering policies from different actors, but nevertheless to allow correct matching of subscriber interests and publisher notifications and also to keep the decoupling between subscribers and publishers.

Several works consider the security of CBPS explicitly:

- Ion, Russello and Crispo [110] consider the confidentiality of the content, metadata and filtering policies in the publisher and subscriber side separately, using KP-ABE and CP-ABE as a product cipher and encrypting the attributes separately with a multi-user searchable data encryption scheme.
- Yuen, Susilo and Mu [216] give a comprehensive security model of CBPS, including, e.g., publisher authenticity and service unforgeability. The CBPS system is modelled as a cryptographic scheme, which is instantiated with CP-ABE and IBS to achieve the various confidentiality and integrity goals.

- Bobba *et al.* trial an implementation of an attribute-based messaging scheme [40], which is based on the publish-subscribe model, using CP-ABE as their main scheme. Attribute-based messaging tackles many practical problems related to the correctness of different types of policies.
- Bertino *et al.* consider additional roles to the publisher in [34], and present some integrity-related concerns of the content in such cases, together with a scheme to enforce integrity for redacted documents based on Merkle hash trees.

The schemes mentioned above can be considered as (integrated) applications of CAC to CBPS, using many of the same methods and schemes presented in this work. However, although publish-subscribe environments are our main focus, we would like the results to *imply* different CBPS security features rather than *depend* on them.

In the work of Bertino *et al.* [34], an infrastructure and technologies for a similar trust model to ours ([120])are presented. Basically, the model by Bertino *et al.* assumes several distinct roles: Owner, Publisher and Subject (producer, storage and reference monitor, and consumer of documents, respectively). The Publisher enforces access control of XML-documents with respect to the Subjects according to security policies provided by the Owner. The subject is able to verify that the documents are complete and unforged. The construction uses Merkle hash trees to compute per-element- and per-attribute hashes attached to the "security-enhanced" document. We used this model as our basis, but introduced "smart edge" acting between the user and the cloud, partly answering the research questions 1a, 1b and 2d. The environment is presented in Figure 20.

The elements in the architecture are as follows:

- **Data Owner** is responsible for the data and decides the access control policy and approves the policy on how the AC policy is changed. Each document has a unique owner, who controls all the sub-elements of the document.
- **User**s are the "consumers" of the data blob. A User has **`read`**- and/or **`write`**-permissions to a set of element. If the user has **`read`**-permissions, she is able to decrypt the content; if she has

**`write`**-permissions, her edits can be considered valid via her digital signature. Some users can act on the behalf of the Owner, and have **`admin`**-permissions (permissions to order changes to the permissions from the Filter). Generally, **`write`**-users are considered to be the publishers and **`read`**-users the subscribers of a typical publish-subscribe-model.

- **Storage** is one element in the cloud where the documents physically may reside. Storage servers are not trusted to view or modify (including redaction and other reference monitor duties) content, but they are trusted to handle versioning and storage functions. Storage does not perform high-assurance authentication of document requests, so it is assumed to be easy to bypass the border brokers. Storage can be viewed as the brokers of conventional pub-sub models, or a Channel in related messaging architectures (see e.g. [119]). We write the element name capitalized, when we refer to the architectural element, to distinguish it from other uses of the word.
- **Border brokers** (CBIS-term: Filter) form the "smart edge" of the cloud, and should not be confused with the CBPS filtering function. They relay the functions between Users, Owners and Storage. Border brokers in this model are semi-trusted in that they are allowed to perform administrative functions to metadata inside a document[47], but they are not trusted to see or alter the actual content or the security policy. In CBIS and in the architecture in [120] especially, the SHB's primary objective was to perform redaction, which requires changing the document and thus its possible signatures as well. Since the redaction procedure could be seen as filtering based on user clearance, SHB was called a "Filter" in CBIS. Due to the use of the word in subscriber filtering in publish-subscribe model, we use the term SHB instead for the CBIS-filter as well. Users must be able to verify the authenticity of the whole document, which means the SHB must provide users with sufficient verification information.

[47] Current document handling principles in guidelines for different authorities also dictate both the existence and automatic updating of various, both security-related and security-unrelated, metadata. An example from Finland is JUHTA [114].

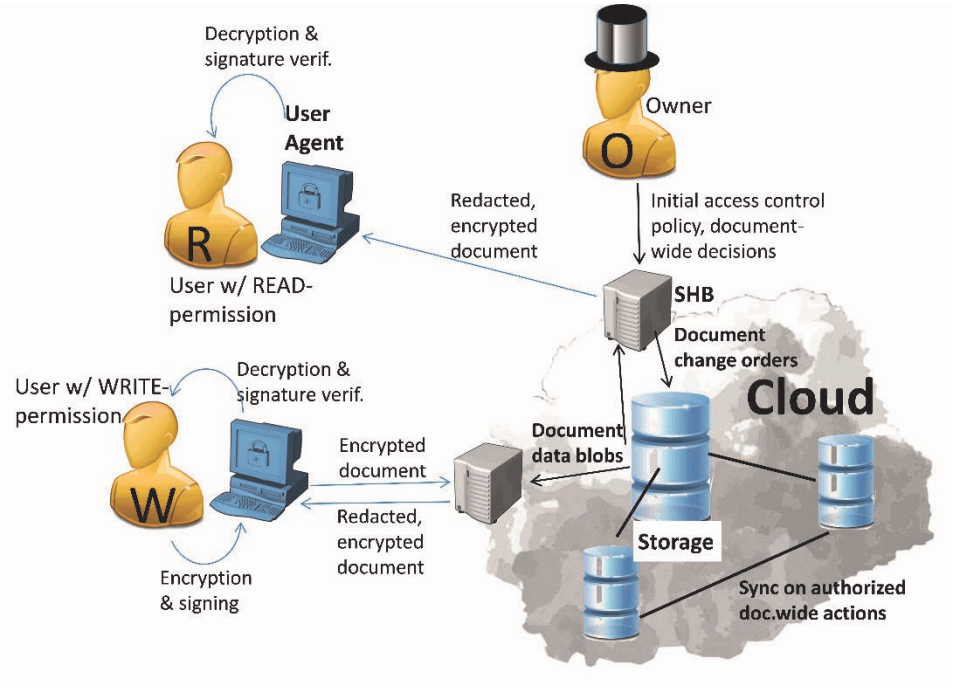



**Figure 20.** The publish-subscribe model used [120]

The original purpose of the border broker's function is valid only if the CAC policy mandates *redaction of encrypted content* (a requirement in the original CBIS-concept). However, as noted in the CPR-concept [163], the redaction of encrypted content may have more drawbacks than advantages, as the encryption itself places high-assurance protection to the content[48]. However, CAC needs additional services, which are best performed at the edge of the cloud. Thus, although the need for redaction service itself might be application-dependent, we still keep the component.

When using CAC in publish-subscribe environment, several implicitly defined factors affect the actual operations there. These factors stem from the passivity of CAC enforcement, when viewed in contrast to (active) reference monitors, answering the architectural research questions (1a and 1b). Cryptography can be used to scramble the information incomprehensible (in effect disabling the **`read`**-permission), but it cannot prevent actual viewing of the scrambled data. Cryptography can also provide infor-

[48] Indeed, the redaction of encrypted content gives a basic passive eavesdropper more adversarial power than a user that might be cleared to at least some MLS level.

mation about tampering (in effect giving proof of the **`write`**-permission), but it cannot prevent the actual modification. All these are a tool's functionalities rather than a principal's. There must still be a principal actually *using* the tool.

Of the conventional information security triad of confidentiality, integrity and availability, cryptography can enforce availability only in special cases. Thus we leave availability for the concern of the cloud. In particular, we bestow the following assumptions to the cloud ([121]):

- A broker or server (in the cloud) acts as storage or an execution platform only, focusing on availability and speed. It may have the capability to remove (all its copies of the) files on an authorized request (if so, the server is able to indicate this, and is called *well-behaving*). The server does not have the capability to perform key-management or cryptographic duties related to the stored content.
- A storage-server is almost always assumed to be able to provide at least one "clean" copy of the requested data blob, although it may not have the ability to identify one. This means that even if unauthorized modifications have been made to the data, or parts of it have been deleted, the server availability functions are able to provide the requestor with at least one copy somewhere with no unauthorized modifications or deletions. This property of the cloud /storage is exactly what blockchains produce: data integrity and availability production by massive distribution and cryptographic verification.
- There are no unpassable reference monitors "close" to the data. For authorized users, some RM-functionality may be expected, but it is also possible to read and write the data blobs by bypassing these RMs. It is *not* assumed to be possible to delete all instances of the stored data blobs via unauthorized channels.
- Border brokers include semi-trusted, automated administrative metadata-handling functions. They should *not* be able to breach the actual content confidentiality or integrity.
- The actions (removals or modifications) of a user before his revocation are considered authorized, thus the storage server is not assumed to be able to contain clean copies of data compromised by

an insider. Actions performed before revocation information reaches the storage are thus a matter of version control.

Many content types have a lifecycle, which is both more detailed and extensive than what we consider here (e.g., content creation, distribution and modification). Other important functions include versioning and archiving, which are both non-trivial tasks in MLS alone, and more so in CAC. They are, however, left outside the scope of this work.

Other implicit consequences of the passivity of CAC include (see also the research question 1b):

- A responsibility shift from the network or brokering infrastructure closer to the users, which manifests itself as the PEP-location: enforcement is practically performed wherever cryptographic transformations are applied. This was studied in more detail in in our later work ([119], [118]).
- Environment attributes used in ABAC lack a trusted reference: for example, time- or location-based attributes can be given in key-material, but without a trusted reference it cannot be checked if they actually correspond to current time or location at all. We postulated tamperproof hardware modules for this purpose ([119], [118]).[49]

All of the existing FE schemes we are aware of have a fixed way of encoding the policy into either the ciphertext, key material or both. In a typical CBPS scenario, however, the publisher and subscriber have different and decoupled policies, which the other party is not aware of. Indeed, in XACML and ABAC as well, assigning attributes is a separate matter from that of assigning, deciding and enforcing a policy. The practice of centralized policy management is yet another example of the implicit assumptions made, when access control is enforced with reference monitors (an important discovery, affecting e.g. the research question 3d). The possibilities for CAC include at least distributing the different PEPs and decoupling attribute assignment from policy encoding *both in key material and ciphertext* in the actual schemes. Technically, this decoupling is pos-

[49] In retrospect, the advent of blockchains appears to actually achieve a type of cryptographically enforced environmental attribute: the massively distributed cryptographic proofs of event sequences, if joined with timestamps, offer high assurance of the actual time-reference.

sible, given the possible dual nature of ABE [21], but this requires a proxy transforming content encrypted with KP-ABE to include a policy and encrypted with CP-ABE. As we are not aware of such schemes, we have chosen the former approach (used, for example in [119] and [118]).

## 5.3 CRBAC in XML documents

To formulate a PKAA-independent method of storing CBIS data, we turned to defining a standardized format for the document [120], in particular, we chose XML for this standard and formulated the corresponding format as an XML schema.

The XML-framework refers here to set of standards and best practices of handling structured content based on the World Wide Web Consortium (W3C) standards around XML [208]. XML itself is a markup language using user-defined tags representing rules to encode documents [208]. XML schemas [209] represent a sort of grammar for certain types of documents, and can be used to check, whether a certain document conforms to a pre-specified rule-set (in this context: check if the document contains sufficient information to enforce and transmit parts of a security policy). XML encryption [109], XML signatures [29] and XML key management [78] are W3C standards to embed encrypted data blobs and digital signatures into an XML document, with the associated key management.

The design principles to achieve PKAA-independency were as follows [120], outlining the more exact requirements for the research questions 2b and 2c:

- Content is enciphered with a symmetric block cipher and the block cipher key (block key) itself is encrypted. The schema should not make more reservations than this to the key management.
- For integrity-checking purposes, content *can* be represented by a hash produced by applying a secure hash-function to the content (this approach is necessary, if content is redacted, as noted by Bertino *et al.* [34])
- The content integrity is enforced by signatures, but their exact type is not specified

- If the permission type is **`write`**, the space occupied by the block key is used to host the public key needed for verification. Note that the User Agent may or may not use this key – this depends on the exact trust model tied to the PKAA.
- If the PKAA mandates the use of certificates, these are included in the signature-element. Certification information required for delegation is an exception for this rule (discussed below).
- The schema *may* make provisions to embrace extensions of a certain PKAA type, provided they do not exclude other PKAAs from the same function. In practice, these include:
    - The block key may be encrypted several times by different public keys, and these listed independently. The encrypted data blob should not give preference to any of these.
    - The role information may be a single identifying string, a list of roles, or a (propositional) logic expression involving attributes.
- The need to separate non-repudiation and basic integrity signatures is PKAA-dependent, so the number of signatures is left open here, and the types of signatures are listed as widely as needed.

Documents can be written by different roles in Figure 20: users with **`write`**-permissions, Owners and border brokers performing redaction. In the absence of schemes which can encode policies in the signatures themselves, we suggest using simple access control lists of which users are allowed to sign this type of content. These ACLs are considered to be part of the document metadata, and signed separately. In practice, for the conventional PKC PKAA, a certificate chain-of-trust is formed, ending in the Owner.

If redaction of encrypted MLS documents is dictated by the security policy, we formulated detailed instructions for border brokers to handle the redaction [120] using the Merkle-tree approach by Bertino *et al.* [34]. Additionally, for a border broker to perform other administrative tasks to a structured document without having access to the actual content, this means that (note also the research question 1b):

- Content and metadata need to have separate signatures

- Layered encryption cannot be used, unless hierarchical documents are required (hiding a whole structure inside one document by encrypting all of the metadata into one blob).

The schema is presented in Figure 21 and Figure 22. The figures reflect the actual XML schema definition file with required enumerations. It can be seen that the document will become hierarchical in nature, consisting of elements and its security-related metadata (SRM). Elements can again have sub-elements with their respective SRM. The schema uses three additional namespaces: `xmldsig` for the XML-signatures [29], `xmlenc` for XML-encryption [109] and web service markup language (WSML) `wsml-syntax` to allow Web Service Modelling Ontology WSML encoding [75] of role information with propositional logic.

The `cbis:AccessSet`-tag of the schema, which is security attribute-specific under the SRM data, contains the actual publisher-intended access control policy. This can be best described using an example, how a subscriber receiving an encrypted CBIS-document would decrypt the data:

- Fetch `xenc:encryptedData`
- Fetch user's key material
- Fetch a `cbis:AccessSet`-tag having type **`read`** in the `cbis:permission`-tag and role information in `cbis:roleExpression`-tag matching the user/role's identity
- Fetch `xenc:encryptedKey` from the same `cbis:AccessSet`-tag
- Decrypt the symmetric key from `xenc:encryptedKey` with user's key material
- Decipher the content in `xenc:encryptedData` with the symmetric key

Note that depending on the PKAA used, the information in the `cbis:roleExpression` may be of vital importance to a reference monitor (if conventional PKC is used) or for informational purpose only (to inform an ABE decryptor, which attributes or policy to use).

We also described a process for updating documents in the cloud. This involves a rather complex process of updating different document signature parts, as well as historical information.

The schema in [120] considered also content and metadata versioning in order to facilitate logging of security events, such as modifying security-related metadata. However, using versioning for this purpose independently of other versioning mechanisms is likely going to cause conflicts, and we leave it out-of-scope in this work.

Enforcing MLS labelling is supported by the schema. Labels are bound to the content via signatures (with appropriate certificate chains, if required by the PKAA). However, expressing user clearance is left for the application unless it can be expressed in the user key material, since binding user attributes to the publish-subscribe-chain cryptographically otherwise is difficult. This again suggests the superiority of the IBE PKAA compared to at least conventional PKC.

The redaction process even with correct labelling will present a performance issue, unless a separate element is defined that specifies the highest and lowest classification of the subtree. In this case, a recursive search of the document element tree will have a substantial number of the subtrees pruned.

Our work described in [120] was aimed at solving the research questions 2b, 2c and 2d, of which question 2b was answered in the affirmative (migration from PKI to ABE is efficient with a PKAA-agnostic XML schema).

During the XML-schema work several issues with respect to CAC, ABE and ABS-schemes surfaced, which seemed to require a wider understanding of CAC together with modern role-based access control paradigm: with cryptography, the **`read`**- and **`write`**-permission are seemingly the *only* permissions enforceable with CAC; using existing CAC-schemes in real environments seemed very often to require reference monitors in any case; the actual use cases seemed to concentrate to very specific functionalities, and if something outside the intended functionality was desired (e.g., integrity of redacted documents), it required non-cryptographic solutions. These problems resulted in the series of work described in [121], [119] and [118].

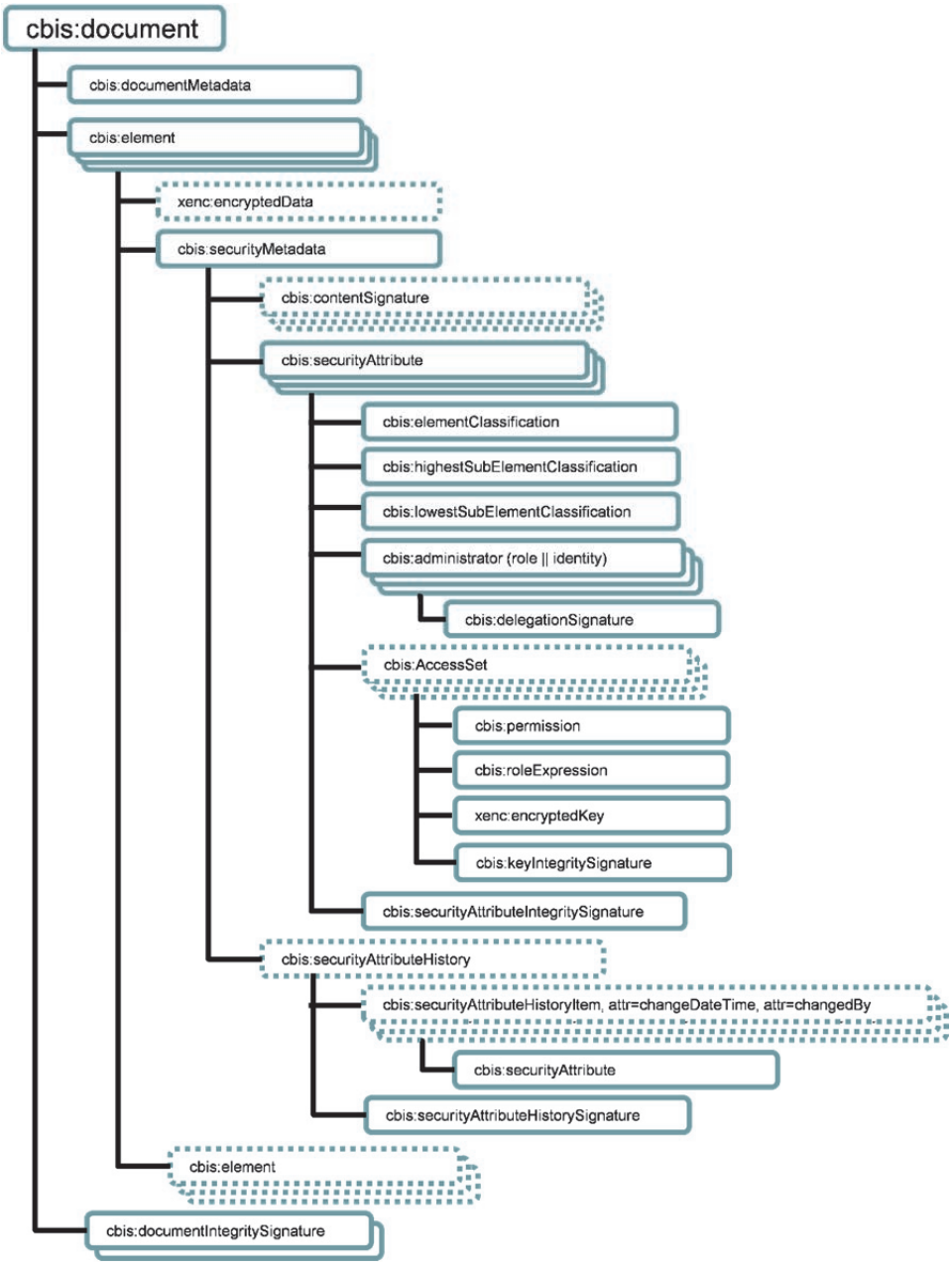



**Figure 21.** The XML-schema structure for CBIS-documents [120].

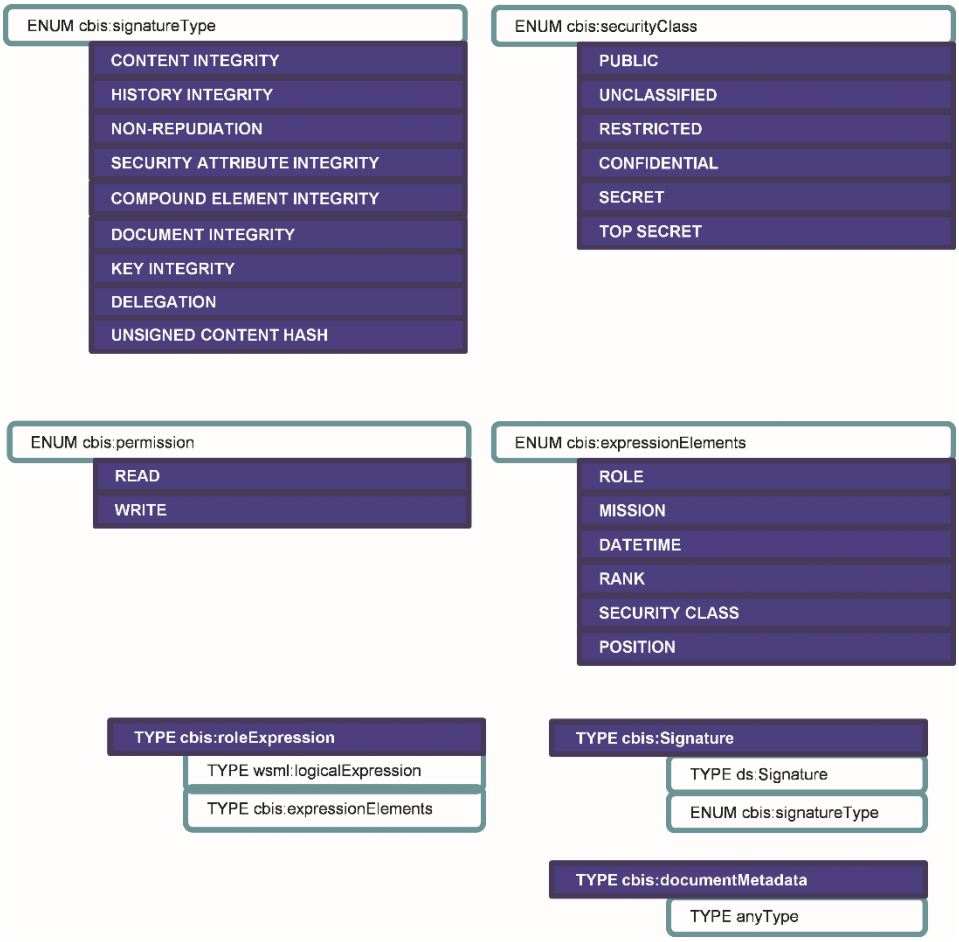



**Figure 22.** The XML-schema enumerations for CBIS-document [120].

# 6. Pervasive CAC Framework

The conventional access control models using reference monitors treat different permission types as an abstract concept. Indeed, the RBAC standard [15] defines permissions as rights to operations, or as rights to run executable computer code. However, in practical database administration and conventional operating systems theory, actual permissions are sometimes rather vague, and either not readily represented as computer programs or not well-defined in the first place[50]. Additionally there seems to be a great variety of permission types in addition to the conventional Hewlett-Packard Unix **`read`**-**`write`**-**`execute`**.

Conventional CAC schemes often start from the presumption that encrypted data or computer code is non-actionable. While this is true in some end-user applications, current automated, distributed and modular systems require multiple types of permissions, which may not be able to be guarded by simply encrypting them. As CAC can directly only support two types of permissions, this seems to severely limit the actual adoption of a more pervasive CAC.

We enumerated 36 different permission types from access control general work ("common knowledge" types, such as **`read`** and **`execute`**), the Windows 7 operating system, Microsoft SQL Server database management system and the Bell-laPadula model [121]. Our main contribution was the realization that most of these types are actually abstract names for **`read`**- or **`write`**-permissions, or sets of them, on different types of metadata (which is basically the intent of the research question 3a). The proposition to extend cryptographic transformations further from the content to other access control functions and metadata was presented in [121], and this was coined under a CAC model property called "pervasiveness". In order to show that it is even possible to extend CAC beyond its current limits, we need to be able to express the different permission types in terms of **`read`** and **`write`** only. This is performed via permission decomposition and by mapping conventional permissions to combinations of **`read`** and **`write`** on both data and metadata.

[50] It is common, for example, to define permissions on top of each other, e.g., permission to change permissions

## 6.1 Permission Decomposition

Our model performs permission type decomposition according to three parameters, shown as axes: read/write (r/w), level of metadata and data-type axis. The axes work as follows:

- The r/w-axis shows, which CAC-enforceable type (encryption for **`read`** or signing for **`write`**) should be used.
- The data-type axis specifies on which data the permission applies to
- The metalevel-axis shows, whether the type of the data type is data, metadata on data or even metadata of metadata.

The data-type axis has several types, but for CAC the relevant information is whether the protected data is content ("payload") or access-control related. Other types are optional, and shown only to show the connection to Table 5. The dimensions are shown in Figure 23.

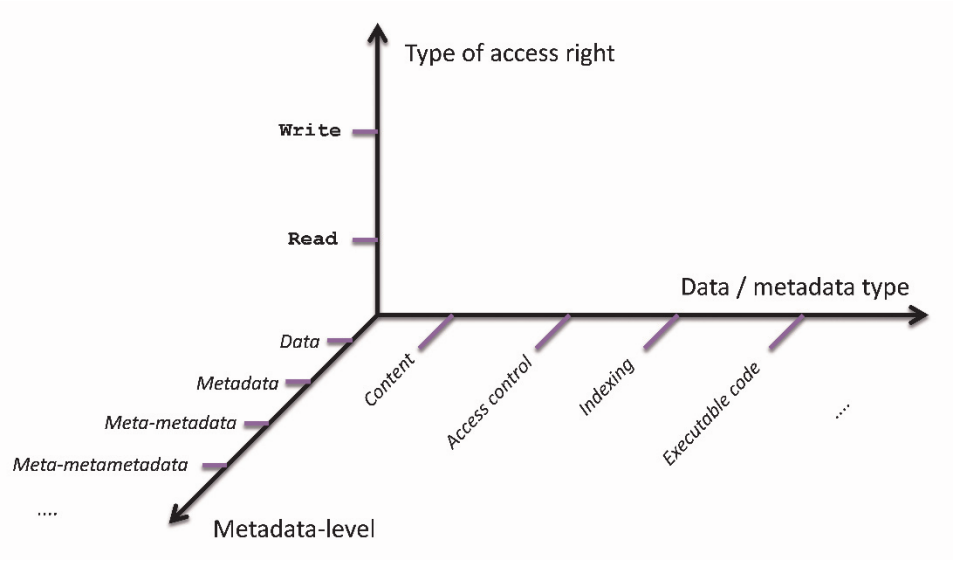



**Figure 23.** The permission decomposition framework [121]

The benefits of the decomposition are straightforward: **`read`**- and **`write`**-permissions can readily be enforced cryptographically (with encryption and signatures, respectively). Additionally, dividing the targets into metadata-levels conceptually places more abstract functions into the data-plane and enables their representation with known methods, such as

structured data (e.g., XML-documents). This in turn enables different classes of data administrators to perform their duties independent of their permissions to access the actual (data-level) content.

The underlying idea in using hierarchy is to represent most of the access rights as existing structured data ([120]), with each type of metadata placed parallel to the actual content node. Access control metadata would typically include encrypted symmetric keys (along with their metadata) and signatures.

Each conventional permission is assumed to be able to be represented by a "small" number of points in the decomposition space. Enumerating and canonizing the permissions this way avoids the translation issues between permission types expressed in natural language between different systems, and clearly states, what is expected of the cryptographic scheme proposed to protect that particular permission type.

We do not expect the decomposition to be universal, merely sufficient enough, so that most frequently occurring permissions could be mapped to the framework. Even then, it does not follow that if permissions can be decomposed according to the framework, they would be practical to enforce with CAC:

- Some of the permissions may simply be more *efficient* to enforce with a reference monitor than with cryptography
- If the (accessed) object is not persistent data, it may not always be reasonable to apply cryptographic transformations to it.

## 6.2 Permission Mapping

The model needs to show that different types of permissions can be enforced cryptographically. This goal is twofold:

1) Showing that a permission type can be reduced to a combination of **`read`**- and **`write`**-permissions in general, possibly using different metadata
2) Showing that enforcing each decomposition of a permission type at the subscriber's security domain is sufficient (i.e. there is no need for actions at the Storage or Channel)

Not all permission types are practical to turn into their **r/w**-decomposition. We selected a set of permissions we deemed somehow general and relevant to the consumer's computing environment. In order to tie the model closer to real systems, in addition to general types, we included permission types from MS SQL Server 2000 database management, Windows XP / Windows 7 - called "special" permissions [150], marked "Win7" in Table 5 - and Bell-laPadula model [31], marked BLP in Table 5. We mapped a total of 36 permission types, shown in Table 5 and 0.

Each permission type is itself metadata of the target it addresses. Thus cryptographic protection of a permission type is always one ladder higher on the metadata axis than its target. For example, **delegate** (**grant** in BLP) is a permission on a permission, or a meta-permission. The **delegate**-permission here means that a subject has a permission to grant selected permissions further, subject to a set of additional restrictions.

**Table 5.** Permission types mapping, part I [121].

| Permission type | Src | r/w | Target of r/w | Permission type | Src | r/w | Target of r/w |
|---|---|---|---|---|---|---|---|
| **read** | gen. | r | <general> | **search** | gen. | r | indexing metadata OR |
| **write** | gen. | w | <general> | | | r | data (in a meaningful way) |
| **create** | gen. | w | data storage metadata | **update** | gen. | w | data |
| **delete** | gen. | w | data storage metadata | **revoke** | BLP | w | access control table metadata |
| **log actions** | gen. | r | system actions metadata | **grant** | BLP | w | access control table metadata |
| | | w | system log data | **control** | BLP | w | access control table metadata |
| **audit logs** | gen. | r | system log data | **own** | BLP | r | data and all related metadata AND |
| **delegate** | gen. | w | access control table metadata (with subject-based restrictions) | | | w | data and all related metadata |
| **append** | gen. | w | selective portions of data (may not overwrite or remove) | **Traverse Folder** | Win7 | r | data storage metadata |
| **execute** | gen. | r | executable code data AND | **ListFolder** | Win7 | r | data storage metadata |
| | | w | program execution data & metadata (in a meaningful way) | **Read Attributes** | Win7 | r | file metadata |

The **search**-permission can be interpreted for CAC differently, depending on the actual implementation of the search: if an indexing structure is built during the search, that structure can be encrypted, with encryption keys granted only to those principals with read permissions. If there is no indexing structure, one may use searchable encrypted databases, using searchable encryption techniques or homomorphic encryption.

Many Windows 7 – specific file and folder permissions (unchanged since Windows XP) are actually just syntactic sugar on direct **read**- and **write**-permissions over different types of data and metadata. Exceptions to this include **Traverse-Folder**, **Take-Ownership** and **Synchronize**:

- **Traverse-Folder** allows (or denies) a user to access folders/directories beyond a certain node to which he does not have permissions. This permission is a compound **read** permission with integral restrictions on the hierarchical relations between two or more objects. **Traverse-Folder** is an example of a permission requiring capability of the access control model to deal with restrictions (as access control decision may depend on other decisions).
- **Synchronize** means that threads are allowed to read file and folder handles and create mutexes based on them to synchronize their operation with other threads. This is a compound **read**- and **write**-permission to metadata in program execution.
- **Take-Ownership**-permission allows or denies someone to "own" an object, that is, allows (or denies) someone to write the access control matrix cell with all the rights in the system. If realized with one attribute, it addresses a whole collection of metadata, and is thus a **write**-permission to a metadatum of metadata.

The BLP-model addresses important permissions with respect to the CAC-model: those that affect the access control matrix itself. We call these access control metadata. It follows then that access control metadata needs to be addressed differently than other metadata. For metadata other than access control, there is no need to elevate the metadata-level for more than one, and CAC adds just another layer on top of traditional structured documents' protection.

Permissions such as **`grant`** and **`revoke`** address the questions of who is allowed to change the access control enforcement function, i.e. the security administrator role.

**Table 6.** Permission types mapping, part II [121].

| Permission type | Src | r/w | Target of r/w | Permission type | Src | r/w | Target of r/w |
|---|---|---|---|---|---|---|---|
| **`Read Ext. Attr.`** | Win7 | r | file metadata | **`BACKUP LOG`** (2) | SQL | w | statement code data |
| **`Write Attr.`** | Win7 | w | file metadata | **`CREATE DB`** (2) | SQL | w | statement code data |
| **`Write Ext. Attr.`** | Win7 | w | file metadata | **`CREATE DEFAULT`** (2) | SQL | w | statement code data |
| **`Read Perms`** | Win7 | r | access control table metadata | **`CREATE FUNCTION`** (2) | SQL | w | statement code data |
| **`Change Perms`** | Win7 | w | access control table metadata | **`CREATE PROCEDURE`** (2) | SQL | w | statement code data |
| **`Take Ownership`** | Win7 | w | access control table meta-metadata | **`CREATE RULE`** (2) | SQL | w | statement code data |
| **`Synchronize`** | Win7 | r | program execution metadata (to read a mutex handle) AND | **`CREATE TABLE`** (2) | SQL | w | statement code data |
| | | w | program execution metadata (to reserve a mutex handle) | **`CREATE VIEW`** (2) | SQL | w | statement code data |
| **`SELECT`** (1) | SQL | r | data and internal db structures to search the object | (1) exercised per database object<br>(2) exercised per database statement | | | |
| **`DRI`** (1) | SQL | w | statement code data | | | | |

Permissions in the database realm, when focused on objects alone, do not differ much from general types of access rights. The database management- specific permissions are:

- **`SELECT`** (on an object), which is a compound permission to first **`search`** the database for matching rows, and then to **`read`** that particular row;
- **`DRI`** (on an object), which means the permission to **`write`** certain database-internal integrity-related indexing conditions to applicable statements;
- In general, databases allow placing statements (~ executable code) in the place of access control matrix objects. Many SQL-

permissions detail these rights, and can be categorized as **`write`**-permissions on statement code (data). These access rights directly affect the storage function, so although they can be enforced cryptographically, it is unlikely they will – at least by another principal than the database itself.

The permissions in Table 5 and Table 6 are generally enforceable in the User's domain. However, we recall that the Storage was entrusted with protection of the availability-attribute. Thus permissions which can be misused in the Channel are an exception to this rule:

- **`delete`**
- **`update`** (in such a way that it erases original data)
- extensive **`append`** (to create a DoS attack)
- extensive **`create`** (to create a DoS attack)
- **`control`**
- **`revoke`**

The **`execute`**-permission refers to computation and is applied to data that is interpreted as program code, which is then run on some platform. Traditionally, **`execute`** only means the ability of a platform to actually operate on the instructions given in the code. Today, "outsourced" computation however, places additional requirements for computation: even if the platform is allowed to execute the code, it may not be allowed to understand all of the levels of execution and the User domain may want to verify the results of the computation. Thus we further divide the execute-permission into three:

- **`execute-A`**: "Availability" of the permission in general: whether the execution platform is *able* to operate on any part of the code
- **`execute-C`**: "Confidentiality" of the execution: what and how much of the code the platform is allowed to understand
- **`execute`**-I: "Integrity" of the execution: can the User-domain verify the result of the computation

For **`execute-A`**, the processor/platform has to have the general permission to access the code altogether (**`read`** permissions) and then the owner of the code needs to have an access to the execution system on a specific

processor (**write** permissions). This is depicted in Figure 24. **Read** can be accomplished by encrypting the executable file, but **write** permissions to the processor are more complex to accomplish cryptographically. It is not meaningful to sign the code, since the platform owner is not necessarily responsible for verifying signatures.

It is vital to be careful not to mix the different execute permission flavors: For example, in some cases, the **execute**-C permission has been attempted to be enforced via different code confidentiality methods (extrapolation of methods designed for **execute-A**), most notably the instruction set randomization (ISR, [28]). However, effective encryption cannot be based on encrypting single machine-code instructions since they typically have very short bit-lengths, and are easily recognized from standard control flow patterns (implemented attacks [197] include recognizing some specific instructions with easily identifiable behavior and using incremental key guessing). On the other hand, fully homomorphic encryption (FHE) addresses **execute-C** but not **execute-A**.

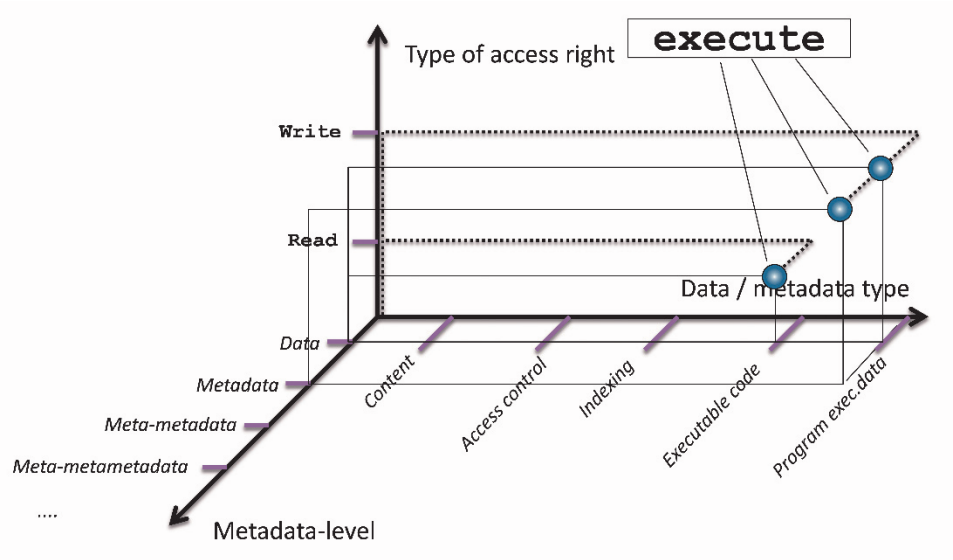



**Figure 24.** Example of a permission decomposition in the framework (**execute-A**)

It is currently even possible (theoretically) to enforce **execute-I**, using a concept called *verifiable computation* [84]. Verifiable computation allows a client (the User domain) to have some measurable confidence on the correct evaluation of a function by a worker (the Storage / Channel

domain), which requires less work than the actual evaluation of the function. An analog concept would be a signature of computation [171]. Some of the concepts within verifiable computation can be instantiated from ABE [172], making it of interest in this work. Some schemes claiming to be general-purpose and efficient exist as well [171].

An interesting example of cryptographic enforcement of a permission, where metadata integrity is enforced on the lower level (actual content) is the use of blockchains [151] to enforce the **append**-permission. In the model suggested here [121], **append** is thought to be enforced like a primitive blockchain: each consecutive addition (e.g., to a security event log file) would be protected by a digital signature, covering also the previous "block" (or event). This approach will safeguard the order of events and integrity of event-chains[51], which are itself metadata. The approach is naturally vulnerable to corrupted event managers, since (depending on the activity profile) possibly even a small set of corrupt event managers could reorder or even remove portions of a log file. The massive distribution property of most blockchain designs was aimed to prevent any corrupted minority to circumvent the append-only restriction.

[51] Indeed, reordering blocks would produce different hashes for subsequent blocks, invalidating their signatures

# 7. CRBAC Confidentiality Enforcement

Based on the work in [121] we postulated that if we can enforce only the **`read`** and **`write`** -permissions in the RBAC model with CAC, other permission types follow. We then considered the model for the **`read`**-permission [119].

In [119], we presented an implementation model based on XACML and evaluated, how the current ABE schemes can realize the different RBAC standard model components. We showed that it is feasible to implement at least the Core RBAC with standard XACML architecture and ABE schemes, and that the expressiveness of the ABE-schemes can reach nearly all the way through hierarchical and constrained RBAC, partly even including Dynamic Separation of Duty. These results answer the first parts of the research questions 3b and 3c.

We showed the feasibility to implement CRBAC with ABE schemes by investigating two points:

- How a publish-subscribe architecture that uses ABE-schemes can implement the different XACML architecture elements and functions
- How the RBAC-standard Core-, Hierarchical and Constrained RBAC commands and functions can be implemented using ABE-schemes. This is parallel to what was accomplished with predicate encryption [77].

## 7.1 Implementation Model

The implementation model assumes publish-subscribe-type functionality, and it is embedded into the XACML architecture. This means that:

- The proposed model needs to elaborate the roles of the typical enforcement components, such as PEP, PDP, etc. CAC is by its nature a distributed paradigm, and some assumptions of a centralized PEP, or having necessary elements temporally or logically close to PEP (or PDP), may not hold.

- The model has to support the usual cryptographic scenario of encryptor and decryptor, with the responsibility of enforcing the **`read`**-permission distributed between the encryptor and key management. The model should thus embrace different policy-embedding methods in CAC (for example, key- and ciphertext policies)

Since the model also follows the RBAC-standard, even the XACML-embedding needs to consider revocation (or removing permissions from a user). With CAC and the **`read`** permission, this translates into revoking the decryption ability from the user. There are basically two approaches (note also the research questions 1b and 2d):

1) Re-encrypting that actual enciphered content, which the revoked principal has had the possibility to access, with a new (statistically independent) symmetric key. This solution is time-consuming and gives full trust to the Storage (the architectural element in Figure 20), but does not give any loopholes to the revoked principal to access.

2) When hybrid encryption (e.g., key-encapsulation mechanisms) is used, there is the possibility to re-encrypt just the revoked user's issue of the symmetric key. This can even be done in such a fashion that the Storage need not decrypt the original key [181]. This is called *ciphertext revocation.* This type of approach presumes that the principal is not able to recover the encapsulated symmetric key to external storage in plaintext while she is still a valid user, and thus implicitly assumes a tamperproof hardware containing parts of the key set as well as the actual symmetric decryption routines. We will assume this model, and thus also presume tamperproof modules.

The XACML-embedding to answer research questions 1a and 1b (XACML-compatible architectural perspective) uses one security domain only, for simplicity. We further restrict the assumptions for the environment such that the processes for document (collaborative) preparation, publishing, re-editing and versioning are considered separate. Thus, although the publish-subscribe-model in [120] would allow editing documents, we consider such editions here to form separate documents.

The model depicts three subdomains, user (USR), object (OBJ) and Channel (the Storage). The encryption is performed in OBJ (associated with the publisher), whereas USR processes (associated with the subscriber) decrypt it.

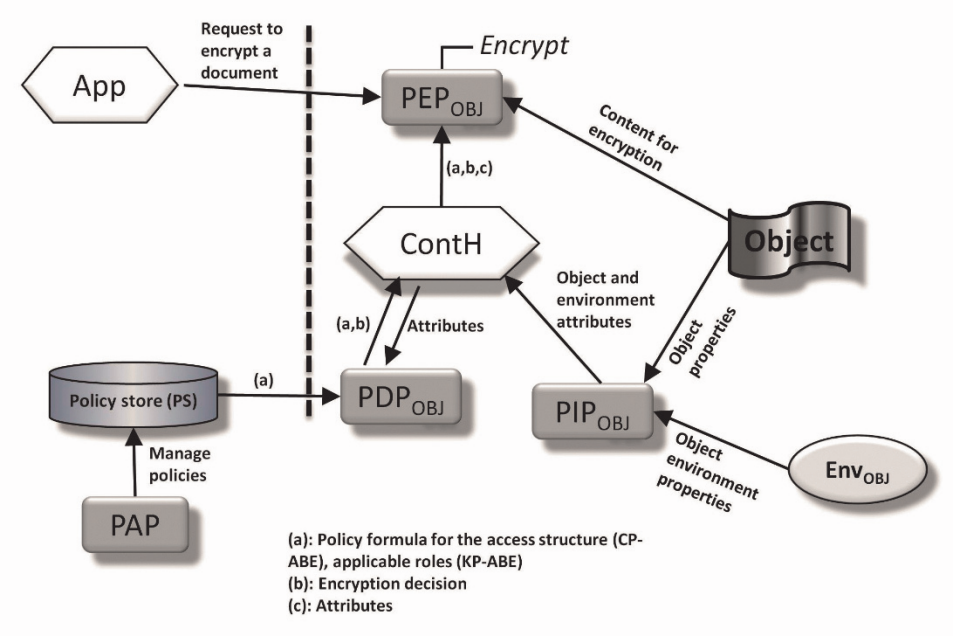



**Figure 25.** The CRBAC-XACML-embedding: OBJ-subdomain [119]

All of the three subdomains share an (authenticated and integrity-protected) policy store, from where the different functions can fetch information. Thus the PAP component need not be distributed for the CAC functionality. All the other functions are distributed, though: PEP and PDP require presence (but different functions) in all of the subdomains; while PIP and the Context Handler (ContH) are found within USR and OBJ only.

The two most important subdomains are shown in Figure 25 and Figure 26. (Figure 25 uses concepts and architectural elements from XACML, see e.g. Figure 5)

As can be seen, the policy processing structure can be copied as such to both of these subdomains. The operation of different components is basically the same as in the RM-case. More specifically, for the OBJ subdomain:

- PIP performs its usual operation as with reference monitors. In the case of ABE, suitable descriptions of attributes translate to a globally agreed hash function from the domain of property description to suitable algebraic structure elements.
- PDP requires the information collected by the PIP, as well as the access policy (as a logical formula in the case of CP-ABE or as a list of applicable roles in the KP-ABE case). The PDP decision together with the policy is forwarded to ContH.
- ContH formats the attributes and the formula to a suitable type required by the actual encryption algorithm, and forwards them (together with PDP decision) to PEP. The translation of the policy into the particular (ABE-) scheme technicalities is a non-trivial task.
- PEP checks whether the requested object should be released to the channel (and thus encrypted) at all, and under which formula / attributes in the positive case. After encryption PEP publishes or sends the encrypted document to the Channel.

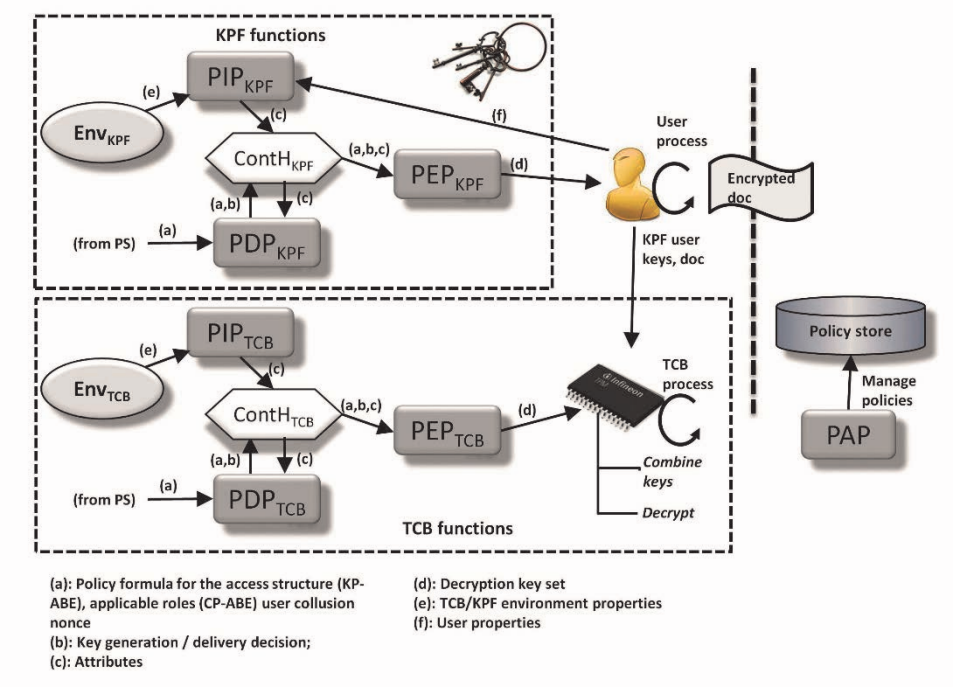



**Figure 26.** The CRBAC-XACML-embedding: USR-subdomain [119]

The USR subdomain is more complex than OBJ, because of the revocation challenges in CAC, mentioned above, joined together with hybrid encryption. We thus propose to use an instance of Trusted Computing Base (TCB), such as TPM, to act as a guard to the hybrid encryption symmetric key. However, as the TCB element is a system-specific element, instead of user-specific, it cannot contain user-related attributes. Indeed, to provide mobility to the user, she should be able to receive her functional decryption keys from a distributed key management system, and join them together to decryption keys from the TCB (which should be fairly constant and independent from other user attributes), in order to actually decrypt.

The USR-subdomain assumes a functionality, which can combine attribute-keys in a trusted computing base element's process. However, as key combination is directly against the ABE security model (collusion prevention), no known scheme can implement this. This is a direct answer to the research question 3d. We addressed the key combination problem in [117] with a "key-pooling" scheme.

## 7.2 CRBAC with ABE

In order to evaluate the feasibility of implementing a standard RBAC-policy in the proposed model, we list here the elements of the full RBAC and the commands from Core RBAC, and compare them to corresponding elements and functionalities in CAC (implemented specifically with ABE), according to the research question 3d. These comparisons are shown in Table 7 and Table 8.

Both ABE policy encoding methods are studied, since the **`read`**-permission enforcement needs to be distributed between the key management and encryption. When the different ABE-schemes are used to enforce RBAC, they perform the permission assignment function on-the-fly during encryption. The role assignment is performed, when a system user has her private key sets generated (not yet delivered, though), while a session is established by receiving the decryption keys.

**Table 7.** $RBAC_3$ elements mapping to ABE [119]

| RBAC element | Applicable model element with KP-ABE schemes | Applicable model element with CP-ABE schemes |
|---|---|---|
| **Default scheme** | Sahai, Waters, Goyal, Pandey, 2006 [35], [97] | Sahai, Waters, Bethencourt, 2007 [18] |
| **Object** | Document / Message | Document / Message |
| **Operation** | `read` | `read` |
| **Permission** | Possibility to encrypt with a given attribute set | Possibility to encrypt with a given attribute set and a formula |
| **User** | User | User |
| **Role** | Attribute set | Attribute set |
| **(Orphan) session** | User's ability to unlock her TPM | User's ability to unlock her TPM |
| **(Active) session** | User's possession of her private key set (with a given attribute set and formula) | User's possession of her private key set (with a given attribute set) |
| **PA** | Included the possibility to encrypt with a given attribute set into the policy store | Included the possibility to encrypt with a given attribute set and formula into the policy store |
| **UA** | Private keys existence for a user (with a given attribute set and formula) | Private keys existence for a user (with a given attribute set) |
| **Role Hierarchy** | Static hierarchy: ciphertext delegation [181] | Static hierarchy: ciphertext delegation [181] |
| **Admin role** | Role on metadata | Role on metadata |
| **SSD** | NM-KP-ABE formulas [96], [162] | NM-CP-ABE formulas [43] + PIP-PDP logic |
| **DSD** | N/A | NM-CP-ABE formulas [43] |

The differences between KP- and CP-ABE can be seen in the constrained RBAC functionality: as key management is assumed to be infrequent, and bound to attributes / roles, KP-ABE is able to perform static separation-of-duty only. In CP-ABE, the restrictions are decided on a document-by-document - basis, making the model ultimately dependent on the session, thus implementing DSD. As the PIP is aware of the object environment, it can constantly give the same conditions and attributes to the CP-ABE elements *on the same UA* making it possible to simulate SSD in the CP-

ABE realm as well. The user assignment with ABE according to Zhou *et al.* [219] is proposed to be performed similarly to our method, namely by *creating* the private keys. The separation-of-duty-functionality requires the use of negative clauses, and thus the most basic ABE constructions do not suffice – instead it is necessary to use so-called non-monotonic versions for this purpose [96], [162].

**Table 8.** $RBAC_3$ commands mapping to ABE from [119]

| RBAC command | Applicable function(s) |
|---|---|
| **AddRole** | Role mgmt and PAP function |
| **GrantPermission** | Add encryption possibility to the policy store |
| **AddUser** | User mgmt function |
| **AssignUser** | Generate user KPF+TCB private key set |
| **CreateSession** | User mgmt function (authentication) |
| **AddActiveRole** | Send user her (updated) private key KPF- and TCB-sets |
| **CheckAccess** | OBJ-subdomain: encrypt; USR-subdomain: Try fetch an encrypted document from the channel and decrypt it |
| **DropActiveRole** | Force refreshing of KPF-keys and disable user's decryption capability for that role in the TCB for the duration of the key-renewal |
| **DeleteSession** | = DropActiveRole (for all the roles used from that session) |
| **DeassignUser (with loss of auth)** | For the discarded role: Discard user KPF-private key sets; use ciphertext revocation for role-accessible documents in the Channel |
| **DeleteUser** | User mgmt function + DeassignUser (for all its roles) |
| **RevokePermission** | Delete encryption possibility from the policy store; ciphertext revocation per role-attributes for documents in the Channel; re-fetching documents in the USR-subdomain caches |
| **DeleteRole** | Role mgmt and PAP function + RevokePermission for all the permissions of the role |

Role hierarchy cannot currently be natively supported by ABE. In a static "snapshot" of roles it is possible to use ABE ciphertext delegation [181] to encrypt a text with attributes from the hierarchy tree from the highest to the lowest level of delegation as higher levels can decrypt anything the lower levels can. However, as this is actually delegation from the scheme perspective, new levels cannot be added on top without re-encrypting the documents or reissuing the keys. In our publish-subscribe-model this would indicate that after adding another senior role with inheritance, there

would be floating documents in the Storage such that this new senior role has no access to them (until re-encryption, that is). In general, any model that makes the access control decision before the creation of additional roles is not able to support role hierarchy dynamically.

In contrast to other similar schemes, e.g., the PE-NDS [77], we proposed encryption only at the `CheckAccess()`-function (in PE-NDS this was performed at the `GrantPermission()`-function). This is because PE-NDS considers RBAC-functions to cover the whole system, whereas in our model, the functions themselves are distributed to several subdomains. Thus `GrantPermission()` in the OBJ-subdomain is a different function (and with different parameters) from that in the USR-subdomain.

As part of the work for [119] we made a literature survey of the current state of ABE and FE, including other sub-branches of FE, such as PE and public-key encryption with keyword search. The main benefit of FE is that it is possible to extract different functions of the same encrypted content, if different key material (associated to permissions) is present. However, we have not yet seen such instantiations that would be substantially different from encrypting structured document elements with separate ABE (or PE) key material: basically most of the FE schemes claiming such functionality merely encrypt the same content *n* times for *n* supported function types and append them to the ciphertext.

Other benefits of using FE or PE instead of ABE include more general circuits for the AC policy formulae, and more complete security models. These features have, on the other hand, negative effect on the different efficiency parameters (part of the research questions, 4b), such as key- and ciphertext bandwidth or computational cost of encryption or key generation. In particular:

- In our model, the ABE model has sufficient expressive power, as it can encode policy formulae from the complexity class $\mathbf{NC^1}$ without trouble [83]. The cases where fully general circuits covering the whole of **NC** would be required for CAC are quite rare[52].

[52] This has been studied in an independent study [214]. Indeed the most complex decision encountered there was whether current location resides on some geographic area. Even is such cases the actual computation can be outsourced (environmental references

- The PE additional feature of hiding the policy itself is somewhat redundant in our case, as the policy can reside in the metadata, which can be separately encrypted, if desired. It is also more flexible to be able to handle the policies separately.

Yuen, Susilo and Mu [216] consolidated different publish-subscribe systems' confidentiality-related goals with those by Carzaniga *et al.* [57]. The architecture presented in [119] answers to these goals as follows:

- *Publisher confidentiality* (Secret information in the notification, including the content and some of the topic metadata are viewable only by those subscribers, whose filters match the notification): This is a property of the underlying FE-scheme in the OBJ-subdomain. If ABE is used, the policy is freely delivered. On the other hand, if the CBIS schema approach is used, the keying information lies in the metadata, which can be separately encrypted.
- *Subscriber confidentiality* (Secret information in the subscriber filters remains hidden from the brokers, publishers, other subscribers and outsiders): User's subscription *abilities* are encoded in her keys, which should never be transmitted in the publish-subscribe-system. However, user's preferences (=filters, which necessarily form a subset of her abilities) need to be communicated to the border brokers. This is a property of the underlying scheme, and ABE is not sufficient for this. Instead, other FE schemes, such as PKE with keyword search [216] can be used.
- *Publisher anonymity* (Publisher identity in the publish-notifications is hidden from other principals except those subscribers, whose filters produce a match in the published document, and those border brokers that route the notification): If the CBIS-schema is used, content is signed, but this signature is part of the metadata that could be separately encrypted, if so wished. The traditional approach is to use document originator hierarchies such that individual authors are not revealed.
- *Subscriber anonymity* (Subscriber identity is hidden in the filter information): It is outside the scope of this work to define, how the

---

are currently not cryptographically enforceable in any case), or simple algorithms used (e.g., dividing the geographic area into suitable rectangles up to the desired accuracy)

subscriber will post her requests to the Storage. Anonymity-preserving schemes can thus easily be used.

As the comparison against general publish-subscribe security goals shows [216], the security goals in MLS are somewhat different from commercial goals: inside organizational hierarchy, policy confidentiality is not much of an issue, and anonymity is actually discouraged. On the other hand, our approach is to use CAC independently of the application, and thus publish-subscribe security goals can be seen as application level security goals, which can also be attained with independent components (here using the CBIS schema structure and methods).

## 7.3 Key Pooling

In the work for confidentiality enforcement architecture [119] we noticed a similar lack of functionality in existing schemes to the one in OLP CPR-CAC [163]. Normal ABE schemes and in general all of the FE schemes (that we know of) have a strict position against user collusion. This is in general a desirable security goal, but in our case there is actually a legitimate case for two "users" to combine their key material in order to have more decryption ability, and distribute the threat to exposed key material.

We wanted to relax the collusion prevention requirement to a degree, basically allowing the "collusion", or key pooling as we named it, in controlled situations, but without lessening the expressive power or efficiency of the existing schemes. Intuitively, we would like to be able to define a policy, which states who is allowed to pool keys with whom. In our limited scenario the policy is such that users *inside* a group are not allowed to pool their keys with each other, but are allowed to pool their keys *outside* (to a specified external group, corresponding here with the terminals). Thus, we need both an architecture that is able to group users and give credentials to them, and a scheme which allows pooling but prevents collusion. These were laid out in [117], in the form of scheme called key-pooling decentralized ABE, or KPD-ABE.

This pooling limitation of current FE was addressed by Wrona and Oudkerk [163] for the NATO CPR-concept. The CPR-solution was to use CP-ABE twice to encrypt first with the terminal policy and then again with the user policy. This has, however, two main drawbacks: the result-

ing ciphertext may grow approximately squarely in the complexity of the policy and the allowed policies are not as general as they could be. To see this, consider a combined policy of the form:

„(UserClearance = SECRET(Crypto) AND TerminalArea = 51) OR (UserClearance = SECRET(Nuclear) AND TerminalArea = 42)“

This is a plausible policy, but it cannot be separated into two conjunctive policies including attributes from one user group only[53]. Using secret sharing schemes directly with CP-ABE could be used the same way (and suffer the same drawbacks).

Other constructions seem to be able to solve this challenge as well, such as:

- Existing CRBAC work (PE-NDS [77], ZVH-RBE [219]). However, they do not consider sets of users with different capabilities. Furthermore, PE-NDS is based on collusion-resistant PE, and ZVH-RBE is not attribute-based at all[54].
- In the conventional ABE schemes, such as KP-ABE ([97]) the KGC could in principal store the personalization values used for collusion prevention per agent, and deal them out identically for such agent sets for which key pooling is desired. However, there are side effects to this, mainly making the pooling ability all-or-nothing nature: either the users with the same personalization value (e.g., interpolation polynomial) can collude at all times with everyone within the pooling group, or cannot collude at all, which is against our goals. This follows from the fact that the collusion prevention property is encoded in the agent's private key.
- Set-based CP-ABE [41] is meant for sets of attributes, but not for sets of users, thus it can also be considered collusion resistant (as the intended usage only considers one user, and seemingly leaves no possibility for collusion).

To enable key pooling, we chose an existing ABE-scheme from different ciphertext-policy schemes that would have sufficiently efficient private

[53] Other measures are possible, such as encrypting the symmetric key with two independent disjunctive policies.
[54] Thus making the whole collusion-concept meaningless.

key structure. This is called Lewko-Waters decentralized ABE, or LW-ABE for short [133]. The scheme is defined in *Definition 7.1*.

The LW-ABE main technique is to use globally unique identifiers (GID) as the common linchpin, with which the different (and independent) attribute authorities (AAs) can automatically coordinate their attribute domains. This is based on sharing *two* nonces according to the access control policy: one to blind the actual message and one (equaling to zero) to enforce collusion. If only one GID is used, a zero element is reconstructed in the exponent and the collusion prevention factor cancels out.

The LW-ABE uses an adaptive ABE-security game, where the adversary does not need to declare the access matrix with selected attributes until the challenge-phase. However, the corrupted authorities need to be declared beforehand as in MA-ABE [56], making the model static w.r.t. corruption. The main security game in LW-ABE is IND-CPA.

The element in the LW-ABE user private key, which is responsible for the collusion prevention, is the hash of the GID. Although this is derived via a secure hash function (making it difficult to masquerade as another user with a carefully selected colliding GID), the decryption procedure actually does not check, whether the hash is taken from a valid GID or even from some random GID with only valid format. As long as the first argument for the bilinear map is from the hash domain $\mathbb{G}$, and they are able to cancel each other in the vector spanning operation, decryption will succeed.

If the hashes from two different GIDs could be combined so that the decryptor could reproduce that from public information (e.g., by the group operation in G) and in such a way that no AA secret information would be revealed at the same time, this would enable key pooling. We use this to our advantage in defining a key-pooling scheme.

*Definition 7.1* (Lewko-Waters decentralized ABE, LW-ABE): Given a security parameter $\lambda$, LW-ABE is a five-tuple of algorithms <**Global Setup, Attribute Authority Setup, KeyGen, Encrypt, Decrypt**> defined in Table 9.

**Table 9.** LW-ABE according to [133]

### Global Setup

- Input: security parameter $\lambda$,
- Select randomly three large (w.r.t. $\lambda$) primes: $p_1$, $p_2$ and $p_3$
- Select randomly a bilinear group $\mathbb{G}$ of order $N = p_1p_2p_3$. Call the respective subgroups $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$ and the bilinear maps domain $\mathbb{G}_T$
- Select a generator $g_1 \in \mathbb{G}_{p_1}$
- Select a (globally unique) naming scheme for users of the system
- Select a hash-function $H: \{0,1\}^* \rightarrow \mathbb{G}$
- Publish the global parameters as $GP = \langle \mathbb{G}, N, H, g_1 \rangle$ and employ the naming scheme

### Attribute Authority (AA) Setup

- Input: attributes $i \in \mathbb{Z}_+$ such that they are not used by any other AA
- For each *i*, randomly choose $\alpha_i, y_i \in \mathbb{Z}_N$
- $\forall i$ belonging to AA, set AA private key as $\{\alpha_i, y_i\}$. Call any pair $(a_i, y_i)$ an attribute private key (*apk*)
- $\forall i$ belonging to AA, publish $PK = \{e(g_1, g_1)^{\alpha_i}, g_1^{y_i}\}$

### KeyGen

- Input: user *U*'s global identity $GID \in \{0,1\}^*$, request for attribute *i*
- Check user *U*'s authorization for *i* and if validated, compute and output *U*'s private key for *i*:

$$K_{i,GID} = g_1^{\alpha_i} H(GID)^{y_i}$$

### Encrypt

- Input: message $M \in \mathbb{G}_T$, an *n* x *l* access matrix *A* with a mapping $\rho(.)$ from its rows to attributes, *GP*, and *PK*s for the AAs, whose attributes are used.
- Select $s \in_U \mathbb{Z}_N/\{0\}$, $\bar{v}, \bar{w} \in_U \mathbb{Z}_N^l/\{\bar{0}\}$, $v[0] = s$, $w[0] = 0$
- Let $\lambda_x = A_x \cdot \bar{v}$ and $\omega_x = A_x \cdot \bar{w}$, where $A_x$represents the row *x* of *A*.
- $\forall (A_x \in A)$, choose $r_x \in_U \mathbb{Z}_n$
- $\forall x$ such that $(A_x \in A)$, encrypt *M* as $C = \langle C_0, \{C_{1,x}, C_{2,x}, C_{3,x}\} \rangle$, where:
  - $C_0 = Me(g_1, g_1)^s$,
  - $C_{1,x} = e(g_1, g_1)^{\lambda_x} e(g_1, g_1)^{\alpha_{\rho(x)} r_x}$
  - $C_{2,x} = g_1^{r_x}$
  - $C_{3,x} = g_1^{y_{\rho(x)} r_x} g_1^{\omega_x}$

**Decrypt**

- Input: $C = \langle C_0, \{C_{1,x}, C_{2,x}, C_{3,x}\}\rangle$, *U*'s secret key set $\{K_{i,GID}\}$ corresponding to her attributes and identity GID. Assume the secret key set is sufficient to satisfy *A*.
- By the assumption, the vector (1,0,…,0) can be spanned by those rows $A_x$, for which *U* has access. Then for such *x*, user *U* will compute:
  - $\frac{C_{1,x} \cdot e(H(GID), C_{3,x})}{e(K_{\rho(x),GID}, C_{2,x})} = e(g_1, g_1)^{\lambda_x} e(H(GID), g_1)^{\omega_x}$
- *U* then chooses constants $c_x \in \mathbb{Z}_N$ such that $\sum_x c_x A_x = (1,0,\ldots,0)$ and computes:
  - $\prod_x \left(e(g_1, g_1)^{\lambda_x} e(H(GID), g_1)^{\omega_x}\right)^{c_x} = e(g_1, g_1)^s$
  - $M = \frac{C_0}{e(g_1, g_1)^s}$

The KPD-ABE assumes the same architectural components as the ones in LW-ABE, and two extra functionalities: key pooling itself and key pooling material management. The different elements and their roles are described in Table 10. As our focus is on the actual key pooling and its security model, we have have omitted the detailed description of „cryptographic infrastructure", such as standard PKI components and the actual pooling policy management (i.e. who is allowed to combine keys with whom), which relies on *tickets* or CA-certified and personalized permissions.

**Table 10.** KPD-ABE architectural elements [117]

| Arch. elem. | Role | Tasks |
|---|---|---|
| User (U) | Natural user of a document management system. Users are grouped into disjoint groups. | Fetch encrypted documents from a repository; fetch sufficient cryptographic credentials; submit document and credentials to a terminal for decryption |

| | | |
|---|---|---|
| Terminal (T) | Computing platforms, e.g., a computer or a smart card. Terminals are grouped into disjoint groups. Terminals include trusted computing base modules, TPMs. | Combine different user and terminal *apk*s and credentials to pooled *apk*s; decrypt a document |
| Terminal Group (TG) | Sets of terminals. Pooling across terminal groups is not allowed. Pooling across terminal and user groups is allowed, if sufficient credentials are given | Provide sufficient identification for terminals and strong binding between terminal and group IDs |
| Attribute Authority (AA) | Attribute management for its designated domain | Manage attributes in their domain; generate and distribute pooling material |
| *Certificate Authority (CA)* | *Provider of system-wide parameters and algorithms, central repository of public keys* | *Sign PA and TG public keys; provide all elements directory services on system-wide parameters and public keys* |
| *Pooling Authority (PA)* | *Policy Decision Point (PDP) for pooling* | *Check user pooling requests against a pooling policy; grant users certified tickets for fetching pooling material* |

The architectural elements and their operation are presented in Figure 27. The KPD-ABE scheme will model the Terminal type of agents which are assumed to contain hardware-security-based controls (e.g., tamperproof modules, TPM-chips, etc.) as semi-trusted. We note that in order to have a secure pooling location for *apk*s (LW-ABE term for attribute private keys), we need a type of agents that represent "honest users", i.e., agents, which do not try to misuse exposed *apk*s. A set of terminals is called a *semi-trusted* group (or terminals' group): $TG$. $TG$ is disjoint from $\mathcal{U}$, the set of users. For simplicity, we consider only one such group. $TG$ is assumed to have the following two properties:

- For any $T_i \in TG$, the private key components stored in $T_i$ are unextractable without destroying $T_i$ (and supposedly triggering some alarm).
- For any $T_i \in TG$, the node will reliably destroy those keys it is not planning to operate on (thus it will not gather a list of used keys)

Our scheme is focused on computing new private keys for a Terminal $T$ to use in decrypting a message encrypted with LW-ABE, based on key pooling elements from AAs and the private keys of terminal $T$ and user $U$.

The scheme assumes that actual encryption, decryption and the access control formation are performed independently of this scheme, by the methods provided by LW-ABE (independently except for an identity change in the LW-ABE **Decrypt**-funtion). The scheme here consists of four algorithms, all of them are assumed to be performed after LW-ABE **Global Setup** and **Attribute Authority Setup**. KPD-ABE is defined in *Definition 7.2*.

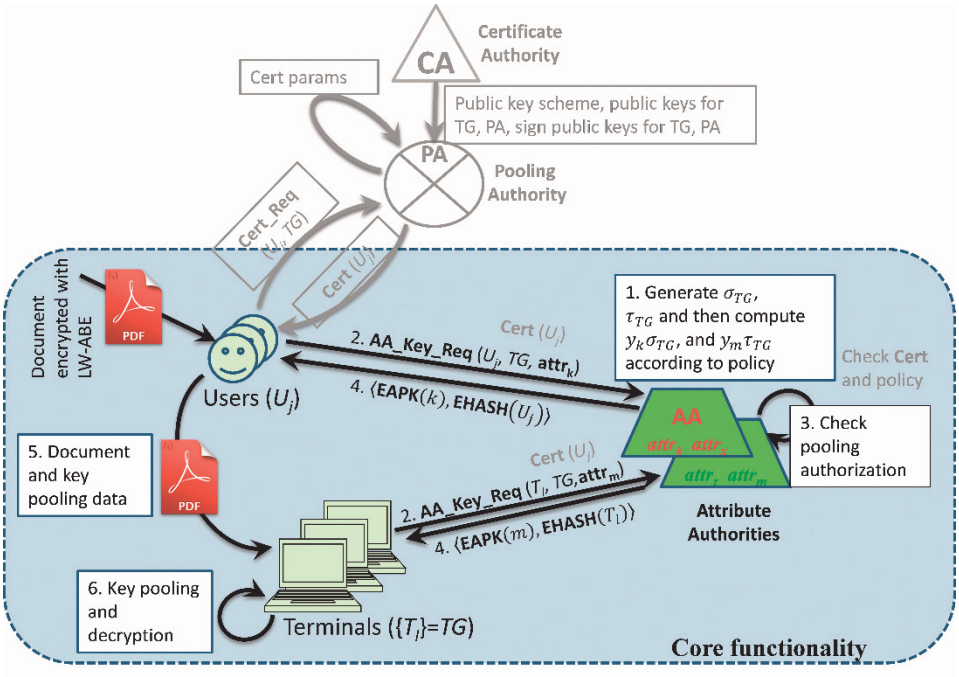



**Figure 27.** The KPD-ABE scheme architectural elements and intended operation [117]

In order to separate users from terminals and KPD-ABE users from LW-ABE users, we talk about *agents*, when we mean LW-ABE users and do not differentiate between (KPD-ABE) terminals and users. Thus users and terminals are both called also agents. The KPD-ABE-users are assumed to form a universe $\mathcal{U}$, and the terminals to form a group TG.

The notation in Figure 27 and Table 11 differs slightly from that in [117], as we updated an old abbreviation in some remaining instances of SG (Semi-trusted Group) to *TG* (Terminals' Group).

KPD-ABE is intended to be used together with LW-ABE: encryptors would use LW-ABE **Encrypt** algorithm for encryption, and decryptors (or

the CRBAC XACML embedding USR-subdomain TCB) would decrypt the text by:

- Running KPD-ABE **PoolKeys** for those attributes used in the policy
- Executing the LW-ABE **Decrypt**-algorithm with modified pseudo-identities $H(T_l)H(U_j)$ as:
  - $\frac{C_{1,x}\cdot e(H(U_j)H(T_l),C_{3,x})}{e(K_{\rho(x),UjvTl},C_{2,x})} = e(g_1,g_1)^{\lambda_x} e(H(U_j)H(T_l),g_1)^{\omega_x}$ for all rows $A_x$ such that the vector (1,0,...,0) can be spanned by the rows.
  - Although pooled keys are created in a pairwise manner, the pairs are by no means predetermined, or "used up" during **PoolKeys**. Thus even with only one key from the other agent type (say, terminal), it is possible to generate common pooled keys with *n* keys from the other agent type (say, user). Indeed, this is mandatory for all attributes used in the encryption policy formula, even if there are only two attributes from different agent types.

*Definition 7.2* (Key-pooling decentralized ABE, KPD-ABE): Given a security parameter $\lambda$, KPD-ABE is a four-tuple of algorithms **<Pooling system setup, TG-Setup, AA-KeyRetrieval, PoolKeys>** defined in Table 11.

**Table 11.** KPD-ABE [117]

| **Pooling system setup** | |
|---|---|
| • Input: security parameter $\lambda$<br>• Assume the existence of an instantiated LW-ABE scheme<br>• Set up pooling authorization infrastructure (or use existing): CA and PA.<br>• Set up a probabilisted public key scheme: PKC={$\mathcal{PK}$, $\mathcal{SK}$, $\mathbb{G}_{LW}$, ***CT***, **Enc**(), **Dec**()}, where:<br>o $\mathcal{PK}$ is a public-key universe for the selected PKC | • $\mathcal{SK}$ is a private-key universe for the selected PKC<br>• $\mathbb{G}_{LW}$ is the bilinear group $\mathbb{G}$ selected for LW-ABE in LW-ABE **Global setup**.<br>• ***CT*** is the ciphertext universe for the selected PKC<br>• **Enc**:$\mathcal{PK}\times \mathbf{G}_{LW} \rightarrow \boldsymbol{CT}$ is the encryption function for PKC<br>• **Dec**:$\mathcal{SK}\times \boldsymbol{CT} \rightarrow \mathbf{G}_{LW}$ is the decryption function for PKC |

**AA-KeyRetrieval** (see steps 2-4 in Figure 27)

- Input: semi-trusted group id *TG*, attribute authority id AA, agent id $Ag = U_j \in \mathcal{U} \vee T_l \in TG$ and attribute $\mathbf{attr}_k$ associated with the LW-ABE AA attribute $k \in \mathbb{Z}_+$
- Input also includes a certified ticket from the CA that the *Ag* is authorized for *k*, but this is omitted for simplicity from the rest of the description.
- If $Ag = U_j \in \mathcal{U}$, compute and output

  $\langle \boldsymbol{EAPK}(k), \boldsymbol{EHASH}(U_j)\rangle = \mathbf{Enc}(TPK_{TG}, y_k\sigma_{TG}), \mathbf{Enc}\left(TPK_{TG}, H(U_j)^{\tau_{TG}^{-1}}\right)$
- If $Ag = T_l \in TG$, compute and output

  $\langle \boldsymbol{EAPK}(k), \boldsymbol{EHASH}(T_l)\rangle = \mathbf{Enc}(TPK_{TG}, y_k\tau_{TG}), \mathbf{Enc}\left(TPK_{TG}, H(T_l)^{\sigma_{TG}^{-1}}\right)$

**TG-Setup** (see step 1 in Figure 27)

- Input: A semi-trusted group identification TG
- Select the TG blinding values $\tau_{TG}, \sigma_{TG} \in_U \mathbb{Z}_N/\{0\}$ (executed at the AAs)
- Generate an asymmetric key pair $\langle TPK_{TG}, TSK_{TG}\rangle \in \mathcal{PK} \times \mathcal{SK}$ for the PKC (executed at the TG). Sign $TPK_{TG}$ at the CA.
- Compute $y_k\sigma_{TG}, y_k\tau_{TG}$ for all $\mathbf{attr}_k$ belonging to the AA (Step 1), and keep them as their pooling parameters sharable with the *TG*. Executed at the AAs.

**PoolKeys** (see step 5 and first part of step 6 in Figure 27)

- Input: user $U_j \in \mathcal{U}$ with associated attribute $\mathbf{attr}_k$, terminal $T_l \in TG$ with associated attribute $\mathbf{attr}_m$ ***EAPK(k), EAPK(m), EHASH(Uj), EHASH(Tl),*** $K_{k,U_j}$, $K_{m,T_l}$, where $K_{i,GID}$ is the LW-ABE secret key for identity GID.
- Executed at $T_l$
- Decrypt the ***EAPK***- and ***EHASH***-values:

  $H(U_j)^{\tau_{TG}^{-1}} = \mathbf{Dec}\left(TSK_{TG}, \boldsymbol{EHASH}(U_j)\right)$

  $H(T_l)^{\sigma_{TG}^{-1}} = \mathbf{Dec}\left(TSK_{TG}, \boldsymbol{EHASH}(T_l)\right)$

  $y_k\sigma_{TG} = \mathbf{Dec}\left(TSK_{TG}, \boldsymbol{EAPK}(k)\right)$

  $y_m\tau_{TG} = \mathbf{Dec}\left(TSK_{TG}, \boldsymbol{EAPK}(m)\right)$
- Compute the common keys for the (hashed) pseudo-identity $H(U_j)H(T_l)$ :

  $DK(U_j, T_l, k) = K_{k,U_j} \cdot \left(H(T_l)^{\tau_{TG}^{-1}}\right)^{y_k\tau_{TG}} =$

  $g_1^{\alpha_k} H(U_j)^{y_k} H(T_l)^{y_k} = g_1^{\alpha_k}\left(H(U_j)H(T_l)\right)^{y_k}$

  $DK(U_j, T_l, m) = K_{m,T_l} \cdot \left(H(U_j)^{\sigma_{TG}^{-1}}\right)^{y_m\sigma_{TG}} =$

  $g_1^{\alpha_m} H(T_l)^{y_m} H(U_j)^{y_m} = g_1^{\alpha_m}\left(H(T_l)H(U_j)\right)^{y_m}$
- Output attribute keys $DK(U_j, T_l, k)$ and $DK(U_j, T_l, m)$

Attributes from different AAs cannot be pooled, unless different AAs can agree on the values $\tau_{TG}, \sigma_{TG}$, over all AAs (per *TG*). This does not give

total independency to the AAs from ech other (requiring separate key management for long-term keys), but this was considered a tolerated drawback in our scenario at this stage, for the following reasons:

- The $TG$ setup (as a whole group of terminals) is envisioned to be occurring very infrequently, at least compared to attribute management time constants.
- The asymmetric key-pair $\langle TPK_{TG}, TSK_{TG} \rangle$ is independently manageable from $\tau_{TG}, \sigma_{TG}$. Thus also the LW-ABE operation is kept independent from the PKC operation.

The intuitive security goals for KPD-ABE are as depicted in Table 12, each of them addressed by different KPD-ABE scheme elements.

**Table 12.** KPD-ABE security goals and corresponding scheme elements

| Security goal | KPD-ABE scheme elements |
|---|---|
| 1: PA security independence from AA | PA does not share any private information from AA |
| 2: TG security independence from AA | A member of TG does not contain, nor is able to compute, AA private keys or the TG blinding value by themselves, during normal operation. A member of TG is assumed to securely delete previous unused values of AA-private key – blinding value combinations. |
| 3: Security of TGs against colluding users | Personalization of pooling information and LW-ABE private keys (hash of the identity included) |
| 4: Secure possession of pooling material with users | Probabilistic encryption of pooling information with TG public keys. |
| 5: Inability for users to fake/produce unauthorized pooling material | AA Key Request verification with a PA certififate, probabilistic encryption of pooling information with TG public keys, LW-ABE personalization of pooling information |
| 6: Inability for users to fake/exchange pooling permission tickets | Probabilistic encryption of pooling information with TG public keys, LW-ABE personalization of pooling information |

The security of KPD-ABE was defined similarly to that of LW-ABE, augmented with KPD-ABE-specific oracles. It is captured in the *Definition 7.3* and *Theorem 7.1*.

*Definition 7.3* (Key-pooling security): A key pooling scheme for decentralized ABE is said to have key-pooling security, if no probabilistic polynomial-time adversarial algorithm $\mathcal{A}$ has a non-negligible advantage in the following game played against the Challenger:

1) The Challenger takes a security parameter $\lambda$, and runs the three setup algorithms for the pooling system and the TG. $\mathcal{A}$ is given descriptions of LW-ABE, PKC, access control policies between agents and TG, and the PKC public keys for TG. $\mathcal{A}$ needs to declare a set of corrupted AAs beforehand.
2) $\mathcal{A}$ is given access to the following oracles:
   - **i.** Attr-Key-Oracle: with the input of an agent identity and attribute for a non-corrupted authority, it will output the attribute private key under LW-ABE
   - **ii.** EAPK-Tuple-Oracle: with the input of agent *Ag* and *TG* identities and attribute index *i*, it will output a valid EAPK-Tuple $\langle EAPK(i), EHASH(A)\rangle$ for that agent identity and attribute index.
   - **iii.** TG-Key-Oracle**:** with the input of an *TG* identity, it will output the *TG* private key $TSK_{\mathrm{TG}}$.
3) After making oracle queries a polynomial amount of times, $\mathcal{A}$ will declare, by its own choosing, a challenge pooling set, consisting of the following:
   - **i.** a semi-trusted terminals‘ group $TG$ , such that no TG-Key-Oracle –query has been performed with that particular *TG*.
   - **ii.** two identities $T_l \in TG$ and $U_j \in \mathcal{U}$
   - **iii.** a set of attributes $\{\mathbf{attr_k}\}$, and an access matrix **M** (corresponding to a policy ***P***) such that
     - **a.** the subset of rows of **M** labeled for attributes from corrupt authorities together with a subset of rows for which $\mathcal{A}$ has called Attr-Key-Oracle will not span a subspace such that the subspace would include the vector (1,0,...,0). (A trivial way to reconstruct the shared secret via oracle queries already specified in LW-ABE)

**b.** neither $T_l$ nor $U_j$ have the sufficient attributes to decrypt anything alone under ***P***, but are able to decrypt jointly

4) $\mathcal{A}$ also selects two messages $M_0$ and $M_1$, and gives the public keys of the corrupt authorities, whose attributes are used in ***P***. The Challenger flips a fair coin $\beta \in \{0,1\}$, and sends $\mathcal{A}$ an encryption of $M_\beta$, encrypted under the challenge set attributes and access matrix.
5) $\mathcal{A}$ may repeat step 2, with the restrictions in step 3 still in place (except that the check on oracle queries against the challenge set needs to be done *before* the query is performed).
6) $\mathcal{A}$ must submit a guess $\beta'$ for $\beta$.
7) The advantage of $\mathcal{A}$ is given as: $\mathbf{Pr}[\beta' = \beta] - 1/2$.

The security of the scheme is captured by Theorem 7.1, which is proven in [117] and in Chapter 7.3.1 for completeness.

**Theorem 7.1 (Security of the KPD-ABE scheme).** *Suppose there is an adversarial algorithm $\mathcal{A}$ against KPD-ABE, as described in Definition 7.1, with advantage $\varepsilon$; then there is also an algorithm $\mathcal{B}$, which either has an advantage $\varepsilon$ in the IND-CPA security game against LW-ABE using at most a constant number of oracle $\mathcal{A}$ calls, or an advantage $\varepsilon$ against the security game of (the probabilistic, IND-CPA-secure, public key encryption scheme) PKC using at most polynomial number of oracle $\mathcal{A}$ calls.*

### 7.3.1 Proof of Key Pooling Security

In this chapter we will give the proof of **Theorem 7.1** (establishing IND-CPA-level security assuming IND-CPA-level security for both LW-ABE and PKC).

The Theorem needs two separate lemmas to be valid. Due to space constraints, the proofs of the lemmas were omitted in [117]. For completeness, we present the whole proof here as well.

**Proof of Theorem 7.1**. We will use $\mathcal{B}$ as follows:

- $\mathcal{B}$ will initially try to use $\mathcal{A}$ as a distinguisher against PKC. If $\mathcal{A}$ uses the EAPK-Tuple-Oracle before the Challenge-phase, $\mathcal{B}$ will gather evidence for a PKC-distinguisher. If, however, $\mathcal{A}$ reaches the Challenge-phase *without* any EAPK-Tuple-Oracle- queries, $\mathcal{B}$ will abort

the PKC-distinguisher strategy, and move to a LW-ABE distinguisher strategy. In case $\mathcal{A}$ will query EAPK-Tuple-Oracle, $\mathcal{B}$ will act out multiple games with $\mathcal{A}$, called *runs*. The following describe individual runs:

- Given instances of LW-ABE and PKC, $\mathcal{B}$ will modify their parameters suitably and forward them to $\mathcal{A}$. $\mathcal{B}$ will also initiate an instance of the PKC-game.
- When $\mathcal{A}$ makes oracle queries as described in *Definition 7.3*, $\mathcal{B}$ will both save them in a database for later use and either forward the query to LW-ABE or play the Challenger in *Definition 7.3*. $\mathcal{B}$ will sometimes modify the queries and their responses. If $\mathcal{A}$ makes queries to the EAPK-Tuple-Oracle, $\mathcal{B}$ will feed $\mathcal{A}$ messages suitable for PKC Query-phase by answering EAPK-Tuple-Oracle-queries either with real identities or randomly generated group elements, whichever is best decided by $\mathcal{B}$'s estimation strategy.
- When $\mathcal{A}$ reaches the Challenge-phase, $\mathcal{B}$ will check saved oracle queries for illegitimate queries. $\mathcal{B}$ will also check its records on queries for the EAPK-Tuple-Oracle. If there are no oracle queries for challenge identities, $\mathcal{B}$ will exit to LW-ABE simulation Challenge phase. Otherwise $\mathcal{B}$ will play the Challenger for $\mathcal{A}$ and initialize PKC-guessing record data structures.
- In the LW-ABE game Challenge phase, $\mathcal{B}$ will forward (and modify) the parameters to LW-ABE.
- When $\mathcal{A}$ reaches the Guess-phase, and has not yet exited to the LW-ABE simulation and $\mathcal{B}$ deems there are not enough samples to make an estimate of $\mathcal{A}$'s performance as a PKC distinguisher, $\mathcal{B}$ will save the result, end the current run, and initiate the next one. In other cases, $\mathcal{B}$ will continue to the LW-ABE Guess-phase or PKC Challenge-phase.
- In the LW-ABE game Guess-phase, when $\mathcal{A}$ outputs its guess for $\beta$, $\mathcal{B}$ will construct a guess for the instance of LW-ABE, based on $\mathcal{A}$'s version of $\beta$.
- In the PKC-game Challenge-phase $\mathcal{B}$ will estimate the number of samples needed to produce sufficient confidence in the distribution

hypothesis. $\mathcal{B}$ will then construct a Challenge message for PKC such that it corresponds to the most queried Challenge identity in EAPK-Tuple-Oracle. After receiving an encryption from PKC for this challenge, $\mathcal{B}$ will feed this to $\mathcal{A}$ sufficiently many times and tabulate, how many times $\mathcal{A}$ was able to distinguish between messages intended for the identity offered. If the number of times exceeds a certain threshold, $\mathcal{B}$ will deduce the identity was real and if not, the identity (encrypted by PKC **Enc()**) was bogus and relay this information to PKC.

**Instance modification**: given instances LW-ABE^ and PKC^, $\mathcal{B}$ will setup a database with tables `AttrKeys`, `EAPKTuples`, `TGKeys` and `Guesses` and determine its running mode: it first runs $\mathcal{A}$ a sufficient number of times in **Real** mode and then a sufficient number of times in **Fake** mode. $\mathcal{B}$ will also form setup parameters for every run of $\mathcal{A}$ identically as follows:

- From **Pooling system setup** $\mathcal{A}$ will receive
  i. LW-ABE^ global public parameters and attribute authorities´ public keys
  ii. The corrupted authorities set to LW-ABE^ **Global Setup**-algorithm
  iii. Description of PKC^ encryption and decryption algorithms **Enc()** and **Dec()**
  iv. Randomly (initially, not between runs) selected PA public keys
  v. Randomly (initially, not between runs) selected value ***EHASH***(*rand*) from the **Enc()** output domain.
- From **Pooling system setup** a description of only one PA and TG, and randomly selected PA certification public parameters.
- From **TG-Setup** $\mathcal{A}$ receives a mapping binding all the users to the TG used, and a public key $TPK_{TG}$ belonging to the scheme PKC as follows:
  i. $\mathcal{B}$ will first check, if it has already saved any public parameters from PKC^. If not, it will initiate the Query-phase with

PKC^, saving $TPK_{TG}$ as the PKC^ Challenge-phase public key.

ii. $\mathcal{B}$ will relay the fetched public parameters of PKC^ to $\mathcal{A}$ as the public parameters of TG.

**Oracle queries**: $\mathcal{B}$ will handle and respond to $\mathcal{A}$'s oracle queries as follows:

- Attr-Key-Oracle: The queries and responses are saved into a database table `AttrKeys` owned by $\mathcal{B}$. $\mathcal{B}$ will first check, if the attribute and agent ID ($\langle \mathbf{attr}_i, w_n \rangle$) are already in the database, and if found, return the matching private key $g_1^{\alpha_i} H(w_n)^{y_i}$. Otherwise, this query and its response are directly forwarded to and from the LW-ABE security game **Key Query Phase 1** (which runs the LW-ABE **KeyGen**-algorithm).
- EAPK-Tuple-Oracle: The queries and responses are saved into a database table `EAPKTuples` owned by $\mathcal{B}$. $\mathcal{B}$ will first check, if the tuple $\langle TG, \text{PA}, \text{AA}, U_j, \mathbf{attr}_k \rangle$ is already in the database, and if found, return the matching EAPK-Tuple $\langle \mathbf{Cert}(U_j), \boldsymbol{EAPK}(k), \boldsymbol{EHASH}(U_j) \rangle$. Otherwise,
  i. $\mathcal{B}$ will execute first **PA_CertRetrieval** (a PA-internal algorithm) to compute $\mathbf{Cert}(U_j)$.
  ii. With $\mathbf{Cert}(U_j)$ $\mathcal{B}$ then executes **AA-KeyRetrieval** to compute $\boldsymbol{EAPK}(k)$ and $\boldsymbol{EHASH}(U_j)$. Note that $\boldsymbol{EHASH}(U_j)$ now contains $H(u_j)^{\tau_{TG}^{-1}}$ encrypted with the PKC^ Challenge public key.
  iii. If the running mode of $\mathcal{B}$ is **`Real`** or **`LWABE`**, $\mathcal{B}$ will return $\langle \mathbf{Cert}(U_j), \boldsymbol{EAPK}(k), \boldsymbol{EHASH}(U_j) \rangle$ to $\mathcal{A}$. Otherwise (running mode being **`Fake`**) $\mathcal{B}$ will return $\langle \mathbf{Cert}(U_j), \boldsymbol{EAPK}(k), \boldsymbol{EHASH}(rand) \rangle$.
- TG-Key-Oracle: The queries and responses are saved into a database table `TGKeys` owned by $\mathcal{B}$. $\mathcal{B}$ will first check, if TG is already in the database, and if found, return the matching private key $TSK_{TG}$. Otherwise, $\mathcal{B}$ will execute **TG-Setup** to obtain $\langle TPK_{TG}, TSK_{TG} \rangle$. $\mathcal{B}$ will then return $TSK_{TG}$.

**Challenge phase ($\mathcal{A}$)**: When $\mathcal{A}$ is ready to give its challenge pooling set and messages, $\mathcal{B}$ will first check that the table `TGKeys` does not contain private keys for the TG in the challenge pooling set or for any attribute in the corrupted AA set, or $\mathcal{B}$ will end this run of $\mathcal{A}$ without any markings to `Guesses`-table. $\mathcal{B}$ will then check the `EAPKTuples`-table. If the table does not contain either $\boldsymbol{EHASH}(U_j)$ or $\boldsymbol{EHASH}(V_l)$, $\mathcal{B}$ will initiate Challenge-phase with LW-ABE^:

- The running mode is changed to **`LWABE`**
- $\mathcal{B}$ checks the access matrix **M** provided by $\mathcal{A}$ against the restrictions given in *Definition 7.3*. If any of the restrictions are violated, $\mathcal{B}$ ends the current run of $\mathcal{A}$ without any markings to `Guesses`-table.
- $\mathcal{B}$ will then forward **M**, the public keys of the corrupt AAs, and the challenge messages to LW-ABE^.
- When $\mathcal{B}$ receives the encrypted message from LW-ABE^, it is forwarded to $\mathcal{A}$.

If the `EAPKTuples`-table contains either of the challenge identities, $\mathcal{B}$ will flip a fair coin to determine $\beta$ and use LW-ABE^ to encrypt $M_\beta$ under **M**. $\mathcal{B}$ will relay the encryption back to $\mathcal{A}$, and save the value of $\beta$ under the `Guesses`-table identified by the run mode, number and the agent identities used in $\boldsymbol{EHASH}$ (one entry for each challenge identity; if the running mode is **`Fake`**, $\mathcal{B}$ places the requested agent identity in the place of the random noise sent to $\mathcal{A}$).

**Guess phase and result interpretation ($\mathcal{A}$)**: If the running mode is currently **`LWABE`**, it means that $\mathcal{A}$ has not queried the EAPK-Tuples at all, and $\mathcal{B}$ has mostly been relaying parameters between $\mathcal{A}$ and LW-ABE^. In this case, $\mathcal{A}$ has not been able to use any other oracles than those given to LW-ABE in general. We can then conclude that $\mathcal{A}$ must have deduced some vulnerability within LW-ABE^, and can forward the guess to the LW-ABE^, terminate both the runs for $\mathcal{A}$ and $\mathcal{B}$ as well. The forwarding is performed as follows:

- When $\mathcal{A}$ returns $\beta$, $\mathcal{B}$ will state its guess to LW-ABE^ as $\beta$
- Since the LW-ABE Challenge-phase is entered only once during the LW-ABE^ game, this yields an identical advantage for $\mathcal{A}$ in breaking both KPD-ABE and LW-ABE.

If the running mode is **Real** or **Fake** (i.e. $\mathcal{B}$ is in the middle of PKC e-stimation), $\mathcal{B}$ will first record, in the proper row in Guesses-table, the value of $\beta$, and a boolean value signifying the correctness of the guess of $\mathcal{A}$. $\mathcal{B}$ will then tabulate

- the number of correct and incorrect guesses per identity in **Real** mode
- the number of correct and incorrect guesses per identity in **Fake** mode

If there are not „sufficient" number of guesses available, $\mathcal{B}$ will flip the running modes (between **Real** and **Fake**), end the current run of $\mathcal{A}$ and initiate another. If, on the other hand, $\mathcal{B}$ finds enough samples of rows belonging to one particular identity in **Real** mode to be able to reliably identify the distributions of $\mathcal{A}$'s guesses in **Real** and **Fake** modes (with confidence of at least $1\text{-}2^{-n}$), $\mathcal{B}$ will enter into the Challenge-phase with PKC^:

- We use the following notation:
  i. $U_j$: the identity found in the EAPKTuples-table samples computation
  ii. $\mathbf{attr}_k$, the attribute found in the EAPKTuples-table samples computation
  iii. $N_R=N_r+N_f$: the total number of runs of $\mathcal{A}$ up until this stage, the sum of runs in **Real** and **Fake** modes.
  iv. $n_j=n_T+n_F$: the number of runs of $\mathcal{A}$ in **Real** mode up until this stage, where the identity $U_j$ was queried and was used as one of the Challenge identities; the sum of the number of runs where the guess was True and when the guess was False.
- $\mathcal{B}$ will clear Guesses-table
- $\mathcal{B}$ will construct two challenge messages: $m_0 = H(U_j)^{\tau_{TG}^{-1}}$ and $m_1 = \mathbf{rand}()$, and select the challenge public key as $TPK_{TG}$, and send them to PKC^.
- When PKC^ sends back the encrypted challenge $\mathbf{Enc}(m_\beta)$, $\mathcal{B}$ will start a series of runs with $\mathcal{A}$ in **Real** mode.

- $\mathcal{B}$ will act as it did in the phases until now, with two exceptions:
  i. If $\mathcal{A}$ asks $U_j$ in the EAPK-Tuple-Oracle, $\mathcal{B}$ will give **Enc**($m_\beta$) as the value of ***EHASH***($U_j$).
  ii. Running modes between **`Real`** and **`Fake`** are not flipped any more
- After $N_r$ runs of $\mathcal{A}$, $\mathcal{B}$ will again tabulate $n_T$ and $n_F$. If $n_T/n_j > ½ + \varepsilon$, $\mathcal{B}$ will guess $\beta$=0 to PKC^ and otherwise $\beta$=1.

During the estimation phase $N_R$, $N_r$, $N_f$ ,$n_j$, $n_T$ and $n_F$ are selected such that $\mathcal{B}$ has overwhelming confidence of the „identity“ of the distribution, and thus the advantage of $\mathcal{A}$ in distinguishing the challenge messages is inherited by $\mathcal{B}$ directly, making $\mathcal{B}$'s advantage ε. This can be achieved in a polynomial number of runs of $\mathcal{A}$, as stated in Lemma 2.

**Run cleanup**. At the end of each run of $\mathcal{A}$, $\mathcal{B}$ will clear the oracle tables `AttrKeys`, `EAPKTuples` and `TGKeys`

Using the TG-Key-Oracle together with EAPK-Tuple-Oracle would seem to undermine the LW-ABE security model, as extracting TG private keys may appear to expose sets of non-corrupted AA‘s private keys. However, these private keys are blinded. If the AA private keys and the per-TG blinding factors are chosen uniformly randomly and extracting discrete logarithm is infeasible in the underlying algebraic group, the „bare“ private keys as well as the blinding factors remain secure. This is formalized in Lemma 1.

**Lemma 1.** *Given any (finite) set of elements $y_i\sigma_k \in \mathbb{Z}_n$ for any $i, k \in \mathbb{N}$, it is information-theoretically impossible to output $y_i$ or $\sigma_k$ with a probability better than a random guess.*

**Proof of Lemma 1**: The setting includes a set of equations $y_i\sigma_k = a_{i,k}$, where $y_i$ and $\sigma_k$represent the unknown variables. We denote this equation by its indices ($i$,$k$). For a single equation, this clearly has one unknown variable more than there are equations. Then, for any (finite) set **EQ** of equations in the stated form that has at least one variable more than there are equations, we can add one additional equation. Then there are four cases:

a) ($i$,$k$) such that $(i, k) \in$ **EQ** . This is a duplication of an existing equation and does not provide new information.

b) (*i*,*k*) such that $(i,k) \notin \mathbf{EQ}$ but $\exists(k' \neq k): (i,k') \in \mathbf{EQ}$. This adds one new variable and one new equation, thus not lessening the total number of variables w.r.t the equations

c) (*i*,*k*) such that $(i,k) \notin \mathbf{EQ}$ but $\exists(i' \neq i): (i',k) \in \mathbf{EQ}$. This adds one new variable and one new equation, thus not lessening the total number of variables w.r.t the equations

d) (*i*,*k*) such that $(i,k) \notin \mathbf{EQ}$ and $(\nexists(k' \neq k): (i,k') \in \mathbf{EQ}) \wedge (\nexists(i' \neq i): (i',k) \in \mathbf{EQ})$ (both *i* and *k* have not appeared before in **EQ**). This adds two new variables and one new equation, thus increasing the total number of variables w.r.t the equations.

As adding equations never decreases the number of variables w.r.t number of equations, the proof follows by induction.

**Lemma 2.** *Given two binary random variables $X_n$ and $Y_n$, with distributions defined in Definition 7.4, the number of samples needed to be able identify the distribution with confidence of at least 1-$2^{-n}$ is at most polynomial in n.*

*Definition 7.4* (Binary random variables with a bias): Random variables $X_n$ and $Y_n$, with parameter *n* are defined with distributions

$$P\{X_n = 0\} = \frac{1+\varepsilon(n)}{2}, P\{X_n = 1\} = \frac{1-\varepsilon(n)}{2}$$

$$P\{Y_n = 0\} = P\{Y_n = 1\} = \frac{1}{2}$$

where $\varepsilon(n)$ is a non-negligible function of *n*.

**Proof of Lemma 2**: This follows e.g. from Chernoff information and Sanov's Theorem [189] applied to biased coins, giving as the number of samples (based on the results of Baignères and Vaudenay [26]):

$$N = \left(-log\left(1 - \frac{\varepsilon^2(n)}{8}\right)\right)^{-1}$$

which for small ε becomes $\approx \frac{8ln(2)}{\varepsilon^2(n)}$, which is at most polynomial in *n* if ε(*n*) is at least inversely polynomial (non-negligible) in *n*.

# 8. CRBAC Integrity Enforcement

## 8.1 Integrity Policies and CAC

Integrity in cryptography is, for historical reasons, much less studied than confidentiality. However, in reality many security needs stress integrity over confidentiality. For example, a recent trend in cyberattacks, called *data sabotage* (subtle alteration of data to achieve primary attacker goals[55] evident, for example, in IoT environments [42]) is targeting primarily the integrity of data-plane objects. While confidentiality is a rather straightforward concept, even the definition of integrity in general usage is very fluctuating

As cryptography in general can enforce only data-level concepts, CAC is also restricted to guard only those concepts of integrity, which can be expressed on the data plane (as opposed to the knowledge pyramid “higher” planes). However, it is customary to make assumptions on higher level integrity goals, such as content validity, based on different data-plane factors. These factors include, for example, data origin authenticity, trusted-third-party-verified properties of the data originator, integrity of sequence of events[56] and absence on unauthorized modifications in the data.

In using conventional PKI signatures, all of the implications to integrity are based on one signing identity. It is possible to attest to different properties with, e.g., attribute certificates [76], but these have limited expressive power. ABS and FS, on the other hand, combine the expressive power and techniques of ABE (and FE, respectively) to be used in integrity policies. We investigated the capability of different ABS and FS schemes in [118], with the aim to enforce integrity-policies in CRBAC, mainly addressing the research questions 3b and 3c.

[55] There is no official definition of ”data sabotage”, as of end of 2016. However, the main difference seems to be to gain immediate real-world advantages directly as the result of e.g. decision making system data alteration. This is in contrast to e.g. cryptanalytic attacks against protocols, although data sabotage may well incorporate cryptanalysis as well.

[56] As manifested in blockchains

In [118] we defined the scope of the **`write`**-permission discussion to exclude content validity (when viewed as an integrity goal) as such, since the decision is made via a more complex process than merely checking signature validity.

In making integrity-related access decisions in systems using attributes, it is important to have visibility and influence on (some parts of) the policies used. To this aim, we surveyed the main ABS and FS schemes available during the work [118]. The results of this survey are displayed in Table 13. Ideally, signatures using policies should be able to:

- Encode the access control policy either to the private key or the signature
- Select the policy both at the signer and (at least partly) by the verifier
- Choose, whether the policy remains public or if it is kept private.
- Be able to express as complex policies as needed.

The second point of the signature schemes is not immediately obvious: in a typical ABS setting (e.g., the ABS by Maji, Prabhakaran and Rosulek [139]) it is assumed that the signing policy will reveal sensitive information about the organization responsible for signing. Thus the signing policies are sometimes hidden completely, but this has a variety of drawbacks in the CRBAC setting:

- If access control decisions are based on the signing attributes, it may not be possible or at least it is inefficient to come to the actual decision.
- It may be difficult to come up with a satisfying verification policy, if this has not been agreed upon beforehand
- The verifier has the main protection responsibility, and thus also should have some freedom in selecting under which policy verifying occurs.

The suitability for ABS for enforcing RBAC is explored in Table 13, with properties relevant to RBAC enforcement given. The columns 3-5 include policy encoding type, policy selection place, and process privacy, respectively.

**Table 13.** Policy encoding and processing properties of the main ABS and FS schemes [118]

| *Scheme* | *Novelties* | *P.encoding* | *P.Selection* | *Process privacy* (1) | *Policy expressiveness* | *Main technique* |
|---|---|---|---|---|---|---|
| MPR-ABS (a) | First ABS | σ | S | $I_S$, $I_A$ | Monotonic Bool. formulas over attributes | NIWI (h) |
| DMA-ABS (b) | No signature trustee | σ | S | $I_S$ | Non-monotonic Boolean formulas over attributes | DMA-FE (b) |
| NM-ABS (c) | Non-monotonicity, small signature size | σ | S | $I_S$ | Non-monotonic Boolean formulas over attributes | CP-FE (i) |
| R-ABS (d) | "Revocability" (of anonymity of individual signer) | σ | S | $I_S$(6) | Monotonic Boolean formulas over attributes | NIWI (h) |
| PBS (e) | All policy languages in P | K, σ | $T_S$ | p, $I_S$, ($I_A$) | P-language over messages | Groth-Sahai Proofs (h) |
| FS (f) | Signature size independency of policy size | K, σ | $T_S$ | p, $I_S$, ($I_A$) | All policies expr. with a poly-size circuit | NIZKAoK (j) |
| DFS (g) | Delegation, limited malleability | K(3), σ | S, $T_S$(2) | p(5), $I_S$(4), ($I_A$) | Efficiently computable functions | NIZK required (k) |

S = individual signer, V = verifier, K = private key of S, σ = signature, p = policy, Ts = signature trustee, Is = identity of S, IA = identity of attributes.

(1): The elements hidden from the verifier (in parenthesis, if not applicable)
(2): Signature trustee is able to assign a family of functions to the signer to delegate further
(3): The deleg. key with restr. on the functionalities allowed to be delegated
(4): Including delegated signers
(5): Policy is public for intermediate signers
(6): Unless revoked

a: (Maji, 2011 [139])
b: (Okamoto, 2013 [158])
c: (Okamoto, 2011 [159])
d: (Escala, 2011 [74])
e: (Bellare, 2014 [32])
f: (Boyle, 2013 [51])

g: (Backes, 2013 [24])
h: (Groth, 2008 [100])
i: (Okamoto, 2010 [160])
j: (Bitansky, 2013 [39])
k: (Groth, 2006 [99])

As the Table 13 shows, it is highly typical for ABS schemes to encode the policy in the signature only. Also, the policy is selected solely by the signing principal. Policy privacy is present in the FS schemes mainly, while ABS schemes still allow it to be visible. Main ABS-schemes are surprisingly expressive in their policies: non-monotonic structures are supported as well [159], [158]. This is more due to the underlying FE schemes than actual signature-only constructions. In contrast to ABS schemes, FS-schemes have the signature trustee generate the policy and embed that into the user key. From the CRBAC point of view, the decoupling of user assignment and role activation cannot be easily realized with FS, as the signature trustee will need to generate new private keys per each new set of active roles.

The policy visibility to verifier is interesting by itself (see research question 3d) and it is naturally beneficial to the verifier decision making process, but it plays a more technical role as well: if the border brokers perform redaction or other metadata-level modification to the documents, the signatures may need to be added or modified based on their policies – which then need to be visible. If the signer wishes to have two portions on the policy: one visible to the verifier and one private (organizational structure, for example), DFS is a viable candidate for this, due to its property of delegation and malleability.

Since the ABS expressive power for defining typical CRBAC policies is (by the argument presented for ABE vs. more general FE) also sufficient, we chose ABS schemes over the more general FS schemes as our tool to model CRBAC integrity policies implementation.

## 8.2 Implementation Model

Similarly to our approach in [119], where we investigated the implementation model for CRBAC confidentiality enforcement via XACML, we embedded different XACML functionality in [118] into three different subdomains: **SIG**, **Channel** and **VRF**, for the signer, storage medium and verifier functions, respectively. We also employ the publish-subscribe architecture depicted earlier in this work, in the manner that the publishers reside in the **SIG**-subdomain and subscribers in the **VRF**-subdomain.

The domain-level architecture together with the **Channel**-subdomain is very similar to those in the confidentiality policy architecture.

Using signatures is essentially a two-party protocol, involving both the signer and the verifier. As such, it is not sufficient to have correct private keys in order to publish authorized material – the verifier action is also needed (in case the signer uses outdated or revoked keys, for example). Thus the enforcement function is necessarily *decoupled* for **`write`** in CAC, resulting in an architectural change searched for in research question 1a.

The most frequently used definitions for the XACML Core specification [177] and data flow architectural elements[57] do not place restrictions on the policy handling point locations as such. Thus we conclude that the decoupling of PEP is not against any previous models, only outside them. The **VRF**-subdomain is depicted in Figure 28. Its different functions are detailed in [118].

In contrast to confidentiality enforcement, the cryptographic tasks are not performed by the PEP, but instead by PDP. This is required, since the verifier is assumed to have some influence in the acceptable policy selection. Thus, although the PDP could send desirable policies to the PEP to be verified, the decision can actually be made only after the cryptographic task. Then, instead of requesting alternative policies and transferring trivially interpretable verification results back and forth between PEP and PDP, it is simpler to perform the whole verification in PDP.

In integrity enforcement, the cryptographic private key usage roles are reversed compared to confidentiality enforcement. Thus also the architectural models are reversed: the subscriber domain in the integrity enforcement (**VRF**) resembles more the publisher subdomain (**OBJ**) and vice versa.

[57] XACML itself, different RFCs (RFC 3198, RFC2904, RFC2753) and an ISO-standard 10181-3.

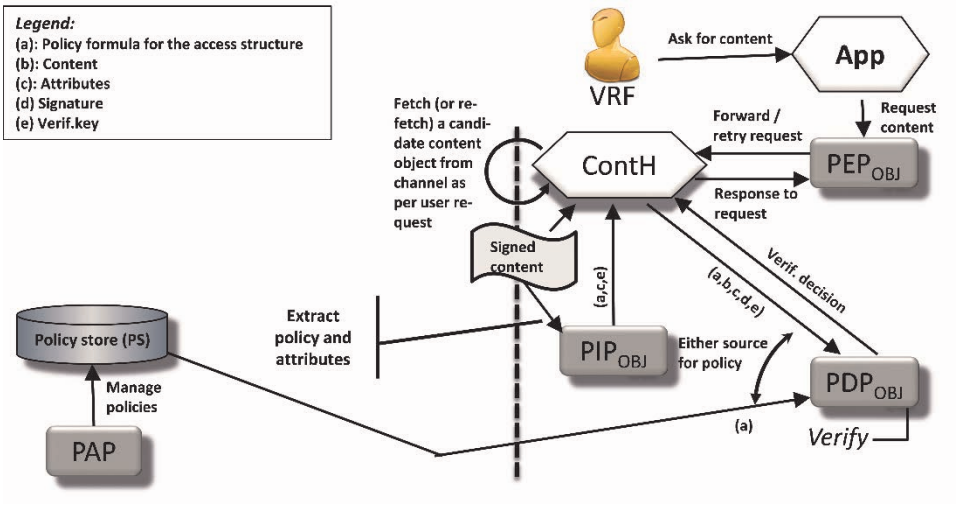



**Figure 28.** CRBAC integrity enforcement architecture, **VRF**-subdomain [118]

We constructed the **SIG**-subdomain with the same principles as the **USR**-subdomain in the confidentiality mapping. The **SIG**-subdomain is presented in Figure 29.

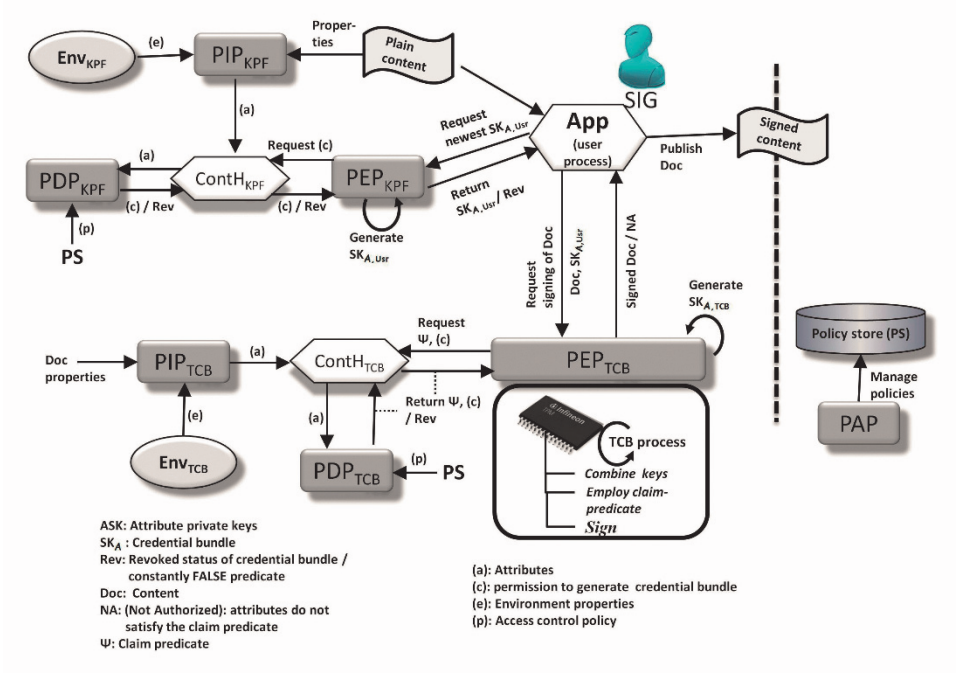



**Figure 29.** CRBAC integrity enforcement architecture, **SIG**-subdomain [118]

In order to be able to separate user assignment from role activation (as described in the model mapping) in, for example, the MPR-ABS scheme functionality, the claim predicate needs to be separately controlled. In the MPR-ABS, the claim predicate can effectively be freely selected by the signer (his attributes allowing), which is more coarse-grained control than the role activation functionality in RBAC requires. Thus in our model we employ trusted computing base (TCB), which controls the claim predicate[58], and also performs similar key-combination as in the confidentiality model. Unlike the case with KPD-ABE, there does not seem to be a need for a separate scheme in this case.

In the **SIG**-subdomain, both the user and the TCB-element need to employ PEPs. PDPs in both portions are responsible for translating the current policy and attributes into a decision, whether individual currently valid signing keys (credential bundles) can be released or not. They also communicate attribute private keys to the PEP to use in the actual creation of the bundles. $PDP_{TCB}$ communicates the active roles information in the form of claim predicate, to $PEP_{TCB}$.

## 8.3 CRBAC with ABS

The RBAC model is, as such, functionally more fine-grained than merely using encryption schemes and signatures without distributing keying material to more than one entity type. For example, the RBAC model **Supporting System Functions** require that active roles can be changed within a session, if e.g., environmental conditions change.

The different RBAC elements and functions are mapped to ABS elements in Table 14 and 0. Some of the most crucial concepts were elaborated separately in [118], according to the goals set in research questions 3b and 3c. These include:

[58] In contrast to earlier statements in the end of Chapter 5.2, blockchains are *not* a general solution for removing TCB in CAC. The *policy store* is a Channel component, which benefits from blockchain technologies, but the reason we need TCB in the PEP has nothing to do with integrity of order of events. A efficient way to enforce the integrity of the association between a user and her claim predicate, would be to use SNARKs (or functional signatures) of the claim predicate within the document metadata.

- *Role*: whether a role is represented by a single attribute or a set of attributes; a single attribute was chosen
- *Session management*: How different sessions need to be reflected in the ABS user claim-predicate
- *Session separation*: In contrast to reference monitors, CAC cannot separate different sessions, unless the scheme supports collusion prevention, thus mandating the use of at least ABS or FS in the **`write`**-permission enforcement.
- *User assignment:* how the UA is performed without actually activating the role yet
- *Revocation:* User revocation is of different nature, when digital signatures are used: firstly, as the subscriber (or Storage) has the verification responsibility, damage control depends on the actions of the non-corrupted principals rather than corrupted ones; secondly, compromise detection does not place limits to the number of suspect corrupt documents. Revocation will require resigning security-related metadata (more specifically, separately authenticated key material), if the **`read`**-enforcement with ciphertext delegation is at use at the same time.
- *Administrative roles*: can be supported via the CBIS-schema metadata

**Table 14.** RBAC elements mapping to ABS scheme elements [118]

| RBAC element | Applicable model element w/ ABS |
|---|---|
| **Default scheme** | Maji, Prabhakaran and Rosulek, 2011 [139] |
| **Object** | Document / Message / Content |
| **Operation** | **`write`** |
| **Permission** | Private key existence for an attribute |
| **User** | User |
| **Role** | Attribute |
| **Session (differentiator)** | User credential bundle personalization |
| **Session (active roles)** | User's possession of her personalized credentials (with a given attribute set) and a defined claim-predicate |

| | |
|---|---|
| **PA** | Attribute secret key creation |
| **UA** | User's possession of her personalized credentials (with a given attribute) |
| **Role Hierarchy** | Static hierarchy: Attribute delegation |
| **Admin role** | Role on metadata |
| **Static SoD** | Non-monotonic claim-predicates [159] |
| **Dynamic SoD** | Non-monotonic claim-predicates [159] |

The conclusion (in [118]) is that current ABS-schemes can already support the Core RBAC, in a distributed implementation model and considering the **`write`**-permission. There are problems still, especially with dynamic hierarchies and providing support to both role activation separation from user assignment and strict control of role activation at the same time.

The ABS schemes are a sufficient and necessary class of signature schemes for implementing the most common access control needs and policies. The reasons for going beyond ABS to FS would include most importantly:

- Complex policies requiring evaluation of arguments beyond $\mathbf{NC^1}$
- Moving the claim-predicate enforcement from trusted hardware to key management (and accepting a more frequent or hierarchical key updates)

However, these do not appear to be sufficient reasons to move to FS, not at least for typical MLS document handling environments.

**Table 15.** RBAC commands mapping to ABS scheme elements [118]

| RBAC command | Applicable function(s) |
|---|---|
| **AddRole** | Role mgmt and PAP function |
| **GrantPermission** | Create private key for an attribute |
| **AddUser** | User mgmt function |
| **AssignUser** | Generate user's (new) credential bundle |
| **CreateSession** | Create user's claim predicate |

| | |
|---|---|
| **AddActiveRole** | Change user's current claim predicate |
| **CheckAccess** | **SIG**-subdomain: sign; **VRF**-subdomain: Try fetching an instance of the signed content from the **Channel** and verify it. |
| **DropActiveRole** | Change user's claim predicate |
| **DeleteSession** | Invalidate user's claim predicate |
| **DeassignUser (with loss of auth)** | For the deassigned role: Exclude user from next credential bundle update (time-stamped attribute names) and/or Revocation list distribution |
| **DeleteUser** | User mgmt function + DeassignUser (for all its roles) |
| **RevokePermission** | Exclude attribute from next attribute private key update (time-stamped attribute-names) and/or Revocation list distribution |
| **DeleteRole** | Role mgmt and PAP function + RevokePermission for all the permissions of the role |

# 9. Implementation Considerations

The main goal, under which also this research was conducted, was to find (or develop) reasonably efficient and secure cryptographic schemes to use for enforcing CRBAC in MLS scenarios. This is a very ambitious goal, given the abundance of previous work, attempts towards working systems and a lack of comprehensive solutions. Our first steps in this vein were to establish a general view of what is feasible and how far it is possible to go using state-of-the-art functional cryptography. This is not to say that we had not looked into the various minute details and caveats possibly residing in individual schemes or the general assumptions permeating most of the research in the area. We will take a brief look at some of these implementation-related detailed issues, as they play a relevant role also in the larger feasibility picture. This chapter aims at answering some of the research questions from the efficiency and security perspective (see research questions 4a and 4b).

In cryptography in general and especially in functional cryptography the schemes are built in a layered model in the sense that a scheme with desired practical functionality builds upon simpler schemes and primitives. Then, although it might not be obvious from the top-level scheme description, there are multiple implicit assumptions of the availability and practicality of the actual building blocks. Thus, for example, the mapping of real-world policies into actual FE structures is a non-trivial and often overlooked process.

Policies for access control are typically specified with different policy definition languages, such as Security Policy Language [176], XACML [177], or Authorization Specification Language in Flexible Authorization Framework [111] or proprietary ones (Windows Active Directory Markup Language [149], SELinux policy language [136], [202], and NATO CPR Language [16]). Rule-based definitions such as those in SELinux are very common in practice.

These languages do not, however, translate directly into FE scheme structures. In fact, the translation is a rather complex process, and requires at least the following steps, assuming that the actual scheme has already been chosen:

1) from a (rule-based) policy definition language representation to a logical formula,
2) from a logical formula into a propositional logic formula, including transformation of the formula into a predicate containing only "standard" logical connectives (**`AND,OR,NOT`**), optimizations of the length of the predicate's expression, optimizations of the formula corresponding to the access structure model used, such as propagating the usage instances of the **`NOT`**-operation into the literals,
3) from a propositional logic formula into a general access structure,
4) from a general access structure to a scheme-specific access structure, including operations such as re-naming duplicate-use attributes to unique attributes
5) scheme-specific optimizations of the access structure.

In particular, in step 3 it should be considered that the ABE-term "attribute" does not translate one-to-one to the non-cryptographic use of the word. This in turn will easily lead to misconceptions about the capabilities of a particular scheme. More specifically, the scheme internal attributes are usually merely binary statements, which can be translated into descriptive language, such as an (internal) attribute $A_1$="`classification=RESTRICTED`". The internal attributes, however, are not typed variables, thus the expression "`classification=SECRET`" is a completely independent (internal) attribute from $A_1$, instead of being the same attribute-variable with a different value.

This difference between internal attributes and policy-level attributes is not so obvious with those policy expressions that can be expressed with relatively static categorical identifiers. However, those predicates containing quantified constants or variables that are evaluated very often (e.g., hour of the day), are a different matter altogether. Thus, if we want to encode, for example, a policy with the constraint (`100 < Badge_ID < 200`) to CP-ABE attributes, we may want to optimize the predicate size by encoding binary statements of the values of individual bits of the policy-attribute `Badge_ID`, and use these as the actual scheme level attributes (a method suggested in the first CP-ABE [35]).

A general view of internal attributes in FE schemes also sometimes neglects the need to associate multiple policy-level attributes to entities. Thus, for example, in inner-product-based schemes (such as Okamoto and Takashima's FE [160]) the access structure is defined such that all authorized sets are singletons. Then, although a scheme using inner-product access structure allows multiple attributes and their complex interrelations, it does *not* allow inherent association of multiple attributes to a single user. This makes such schemes impractical and the implicit assumption in research question 4b valid.

Some schemes (e.g., the ABE by Lewko and Waters ABE [132]) that provide adaptive and non-selective security (according to *Definition 2.19*) assume the existence of composite-order groups with efficient bilinear maps. These are usually implemented with the help of elliptic curve group (ECG) pairing functions, such as Weil or Tate pairing ([113]). However, it is far from trivial to select a suitable ECG and corresponding pairing function: first of all the construction of a composite-order ECG is not straightforward (see, for example Freeman's argumentation [79]), and furthermore the selection of the pairing function and ECG parameters may have a large impact on the performance of the scheme (for example, the ciphertext length for a fixed security parameter using Weil pairing for ECGs in supersingular curves depends on the embedding degree of the pairing function). In particular, since the subgroup elements used in the actual encryption (in schemes using composite-order groups) are of the same length as the whole composite-order group elements, the mapping of the security parameter to sufficient ECG group size is not performed based on the subgroup size but rather on the whole composite-order group size. For example, for 80-bit security, composite-order group sizes of 1024 bits are required, making the bandwidth efficiency comparable to that of RSA.

Many of the current functional cryptography schemes have been constructed functionality first, or with the goal of having provable security along one element of security only. However, having schemes which are both sufficiently functional as well as secure along all of the possible security axes seems difficult.

One aspect is the need for CCA-security (for schemes, where game-based security is sufficient): FE schemes are customarily proven secure in the

CPA-setting only. The CCA-setting is usually also attainable, but it is regularly only outlined with some generic CPA-CCA-transformation (such as Fujisaki-Okamoto transformation [81] or the method used by Boneh *et al.* for IBE [43], which was suggested in the first CP-ABE [35]). These generic transformations are usually exponential in the security parameter, which reflects very negatively on the scheme performance. Our setting does not call for protocols, but a simple high-level transaction involving a user process. One recommendation in this case could be, in the absence of relevant and efficient CCA-secure schemes, to augment possible automated processes with additional non-cryptographic measures, such as session or query counters.

All of the IBE-related schemes have to incorporate the selection of the queried and challenged identities (or attributes) in the security model. If the model is game-based, it then becomes a question when to actually introduce the different types of identities (challenge, corrupted or other) to different parties. Early ABE-schemes took a shortcut in this sense, called selective security. However, this type of security intuitively means that if a scheme uses some attribute set, security guarantees can generally be given exactly for that set (so, for example, using expressions that contain only a subset of the full attribute set or adding attributes are outside the security proof). Thus using selectively secure schemes in feasibility studies is quite another thing than actually implementing them.

The efficiency of ABE schemes in terms of bandwidth and computational complexity was briefly touched in [119] and [118]. With basic ABE schemes and straightforward implementations, the document overhead for 128-bit security and access control policy size of about 30 variables was estimated to be around 1-2 kB per document, using space-efficient pairings. In ABS the overhead depends on multiple factors, and for the similar security parameter and policy size of about 10 clauses can vary between 1 to 24 kB. Also, according to Wang *et al.* [204] there is a drastic increase of processing time when moving from 80-bit security to 128-bit security level.

Efficiency can luckily be optimized in various ways. If the policy is encoded in the key, the ciphertext overhead is naturally smaller. Good example is in the NM-ABE by Yamada *et al.* [215], where the ciphertext overhead is constant: two group elements only, giving 0,5 – 1kB overhead

for 128-bit security, which is already well comparable to RSA. Even at the other end of the spectrum, as security guarantees and expressive power are increased, Agrawal *et al.* note in [4] that, for example, 1-NA-SIM-SM-secure[59] general FE with a bounded number of colluders is necessarily at least (only) linear in the number of colluders.

## 9.1 OLP Implementation possibilities

One of the most immediate matches between the selected schemes and concepts is OLP with its default suggested implementation scheme and with KPD-ABE. We investigated possible implementations with both of the intended schemes and their use within OLP in [124]. We will refer to the OLP default implementation as *Layered ABE*[60] and the key-pooling approach presented in chapter 7.3 as *KPD-ABE*.

The need for layered encryption in CPR-CAC follows from the fact that documents should be unencrypted only, if both user clearance and environment conditions are suitable. This translates to combining user and terminal policies. However, combining attributes from two different users in ABE is viewed as a violation of the security goals (called collusion). A simple way of producing a conjunction of the two policies is to use super-encryption, or to first encrypt with the terminal policy and afterwards with user policy[61]. The procedure is depicted in Figure 30, in a simplified manner (focusing on the encryption procedure only).

The Layered ABE instantiation proposes to use the CP-ABE realization by Waters [206]. The Waters' paper on ABE gives actually two schemes, but only the latter one could be considered usable, since it does not restrict attributes as one-use only ("unrestricted" version).

The symbols in Figure 30 are as follows: ***M***: message / content to be encrypted, stored and published (CPR-CAC assumes a publish-subscribe environment and processes); $K_{Pr}$: symmetric encryption algorithm key, $P_u$: user policy, $P_t$, terminal policy, *Pr*: content properties (attributes),

[59] Simulation secure in the standard model, using one challenge ciphertext only and non-adaptive access to key derivation oracle
[60] Due to the use of double encryption of the protected key
[61] It is necessary to perform these in this order, lest the user try to carry the document with him to unsecure locations *after* the terminal phase decryption

**ABE**(): encryption with CP-ABE, using a given policy, **Enc**(): Encryption with a symmetric encryption algorithm, using a given key; ABEA: ABE key management authority. The process in Figure 30 omits a number of policy validity checks and signature generation steps, as well as strips the model of the policy management elements, as they are not relevant to the discussion in here. We adopt the presentation style of Figure 30 for the *application* of schemes for OLP, and use different style to describe schemes *independently* of OLP.

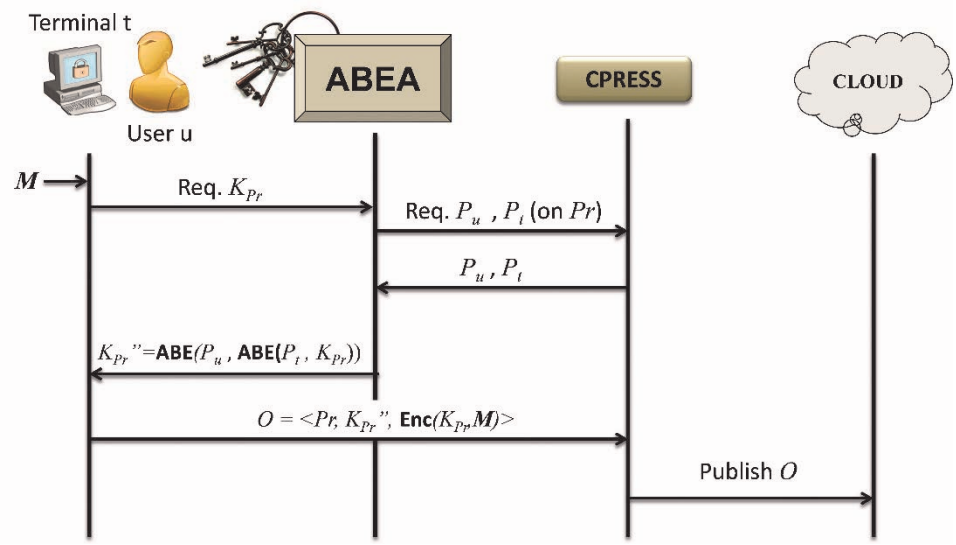



**Figure 30.** Layered ABE publishing process with OLP, according to Oudkerk & Wrona [163]

Using KPD-ABE inside another framework requires describing in more detail the various operational functions and roles of the KPD-ABE architecture. The cryptographic functions were defined in chapter 7.3.

The operational functions of KPD-ABE are described in the series of figures from Figure 31 to Figure 35. Initially, the Pooling Authority needs to retrieve the highest level configuration and key pairs from an authorized entity (here: the CA), as well as the Attribute Authorities to decide on the association between attributes and users/ terminals. This is depicted in Figure 31.

After root-level key material has been established, the users will need to request their pooling policy from the pooling authority (in the format of certificate tickets). Attribute Authorities will create and agree on per attri-

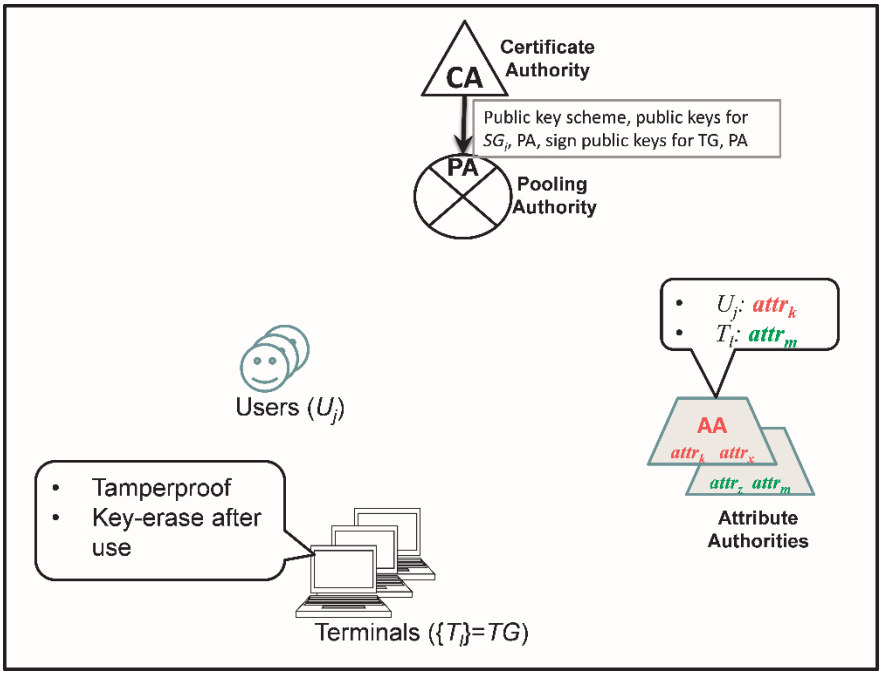



**Figure 31.** KPD-ABE elements at setup [117]

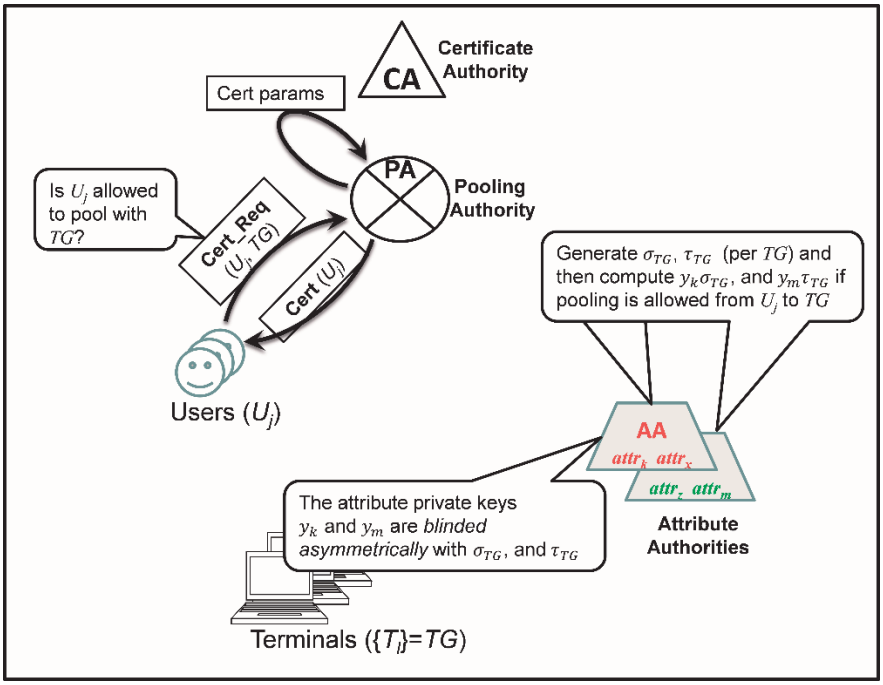



**Figure 32.** KPD-ABE pooling policy and parameters setup [117]

bute private keys for users and terminal groups (note that currently AAs need to have same private keys for the same TG between each other for the pooling to work). The private key material is also blinded separately (see Figure 32).

After the system is set up, the users will exercise their "tickets" to retrieve the pooling material associated with the attributes they are authorized for. Users will need to retrieve the pooling material only once per new attributes after which no further interaction with the AA is needed. AAs will use the tickets (PA-signed certificates, with a trust chain reaching the CA) to verify that users are authorized to pool their particular attributes before sending out the actual material. The procedure is identical for the terminals in TG. This is depicted in Figure 33 and Figure 34.

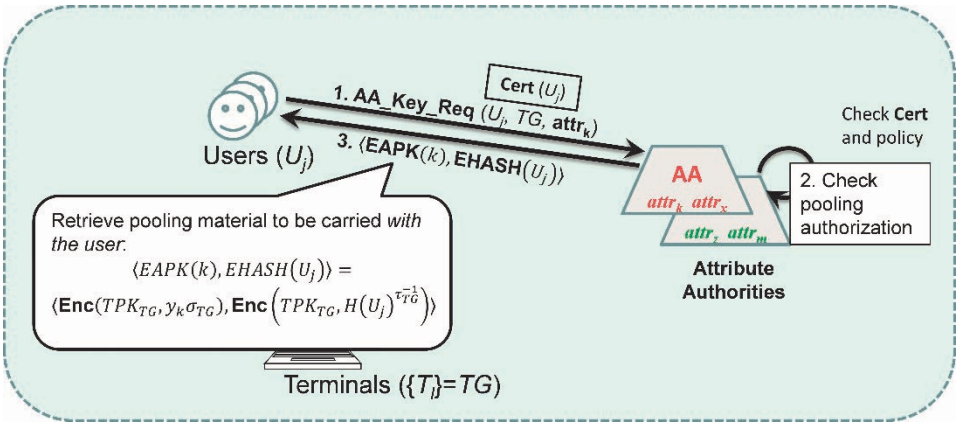



**Figure 33.** KPD-ABE pooling tickets retrieval for users [117]

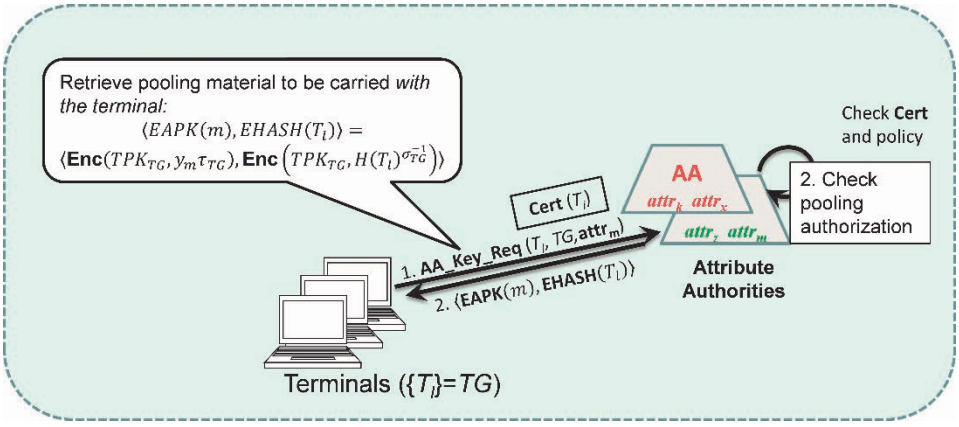



**Figure 34.** KPD-ABE pooling tickets retrieval for terminals [117]

After receiving the pooling material both users and terminals are set to decrypt documents encrypted using user-terminal combined policies. KPD-ABE assumes the documents retrieved from the repository are encrypted with LW-ABE. Users will fetch an LW-ABE-encrypted document from the publish-subscribe system Channel, and forward that together with their pooling material to the terminals. Terminals will then perform pairwise attribute key pooling, before submitting the pooled key material and the document LW-ABE decryption module. This last step is depicted in Figure 35.

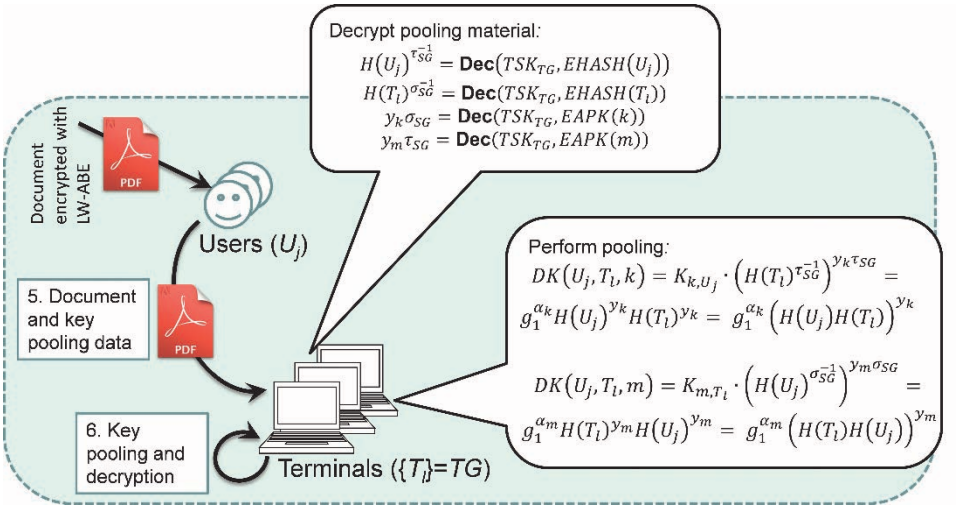



**Figure 35.** KPD-ABE key pooling [117]

Embedding KPD-ABE inside OLP requires using the CPRESS as at least the Pooling Authority (possibly the Attribute Authority functions could as well be integrated inside CPRESS). Additionally, AA will need to outsource the PDP functions (policy decisions) to CPRESS. Pooling material retrieval (corresponding to the figures Figure 33 and Figure 34 above), is depicted in Figure 36 using OLP terms.

When a user needs to publish a document using KPD-ABE, the process is similar to that shown in Figure 30, with some exceptions:

- There is no need to request an encrypted key from ABEA, if the user environment is allowed to create its own key material. However, if only the ABEA is trusted to generate symmetric content keys, this part of the process does not change.

- After CPRESS has output the current (combined user-terminal-)policy, the user (or ABEA, if key-generation is not trusted for the user) will encrypt the content key with LW-ABE and the common policy.

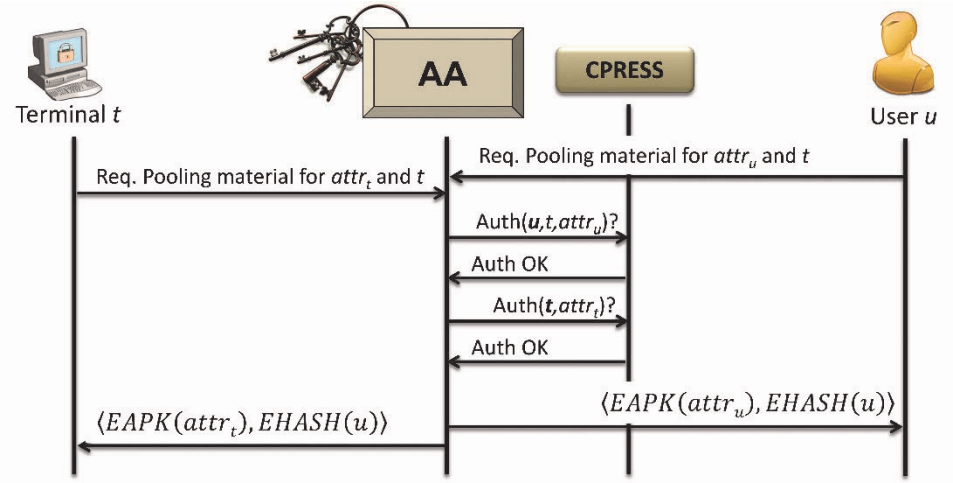



**Figure 36.** KPD-ABE key pooling inside OLP [124]

The subscriber part of the process (document retrieval is shown in Figure 37), without the involvement of ABEA (which can be used as well as an intermediary, if dictated by current decryption policies). As can be seen, KPD-ABE can work also rather independlyent on the OLP architecture, while still maintaining compatibility.

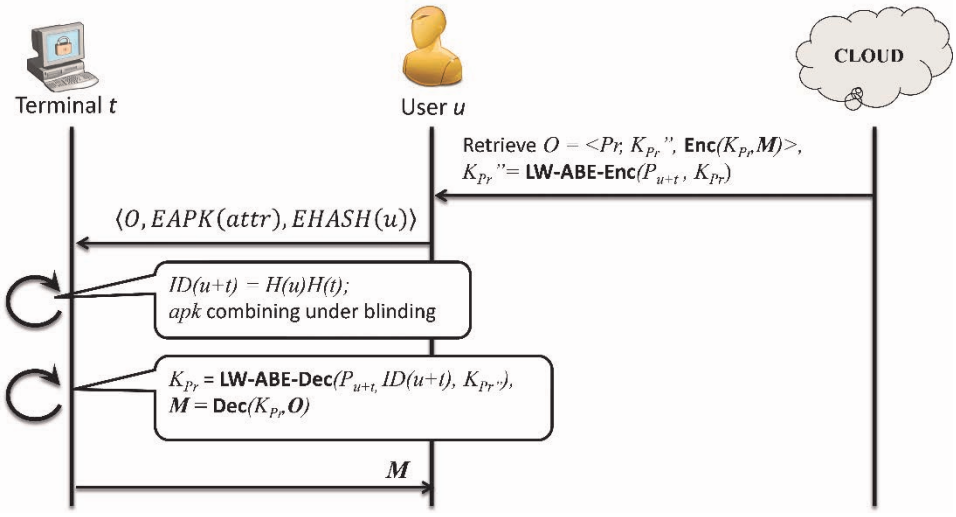



**Figure 37.** OLP subscriber functions using KPD-ABE [124]

## 9.2 OLP Performance Estimations

One of the main contributions in our OLP implementation study [124] was to estimate the bandwidth and computational performance of both Layered and KPD-ABE. We were interested especially in the dependency of the performance on the desired security level as well as on the policies used. We presented the relative performance (in terms of abstract operations and element sizes) as well as absolute performance at a certain benchmark level for both the KPD- and Layered ABE. We focused more on bandwidth performance, as the processing overhead in the intended use case (documents instead of sensors or IoT) is negligible in the assumed processing environment.

Estimating performance, when a desired security level is fixed, can be tricky: relative estimations hide the security level inside the (algebraic) group element sizes used, and the security model actually defines the final relation between the group size and security level. Even the security levels themselves may be incomparable, if the security models are markedly different.

The CP-ABE version by Waters, adopted in the Layered ABE, uses selective security (see *Definition 2.19*). Technically, selective security refers to the security model used for the security proof, where the adversary should commit to attacking against a certain set of attributes even before the actual threat scenario begins. This corresponds to a scenario, where the defender already knows what the adversary is going to do, which rarely happens in reality. The implication of this type of model to the actual system is that unless a system uses a static set (excluding even subsets) of attributes, the security of the construction is not known. Thus changing the attributes (merely by giving different permissions to different documents) would require a system-level reset, *per document*. Thus the Layered ABE, despite of its apparent performance in several security levels, cannot be used in as dynamic environments as KPD-ABE.

It is not clear, how exactly Layered ABE encryption is intended to work. In particular, after encrypting to the terminal policy and using the CP-ABE by Waters, the encryption result $\bar{C}_1$ would be a number of group elements, which are not directly in the domain of the CP-ABE **Encrypt**()-function. The possibilities are then:

- Use symmetric encryption to this set of elements and input the second symmetric key be encrypted with the user policy
- Input all of the group elements of $\bar{C}_1$ to be encrypted with the user policy. This requires an efficiently reversible mapping between the two bilinear groups used, but they are straightforward to construct.
- Input only the message-carrying group element of $\bar{C}_1$ to be encrypted with the user policy. (The other group elements are public shares anyway, and assuming attribute name space separation, can only be opened by terminals).

The first and third choices have equal effect on the ciphertext and key lengths[62], but the second choice will increase processing time and ciphertext sizes squarely (in the number of used attributes in the policies). Third option does not expose extra material to outside adversaries, as the ciphertext elements seemingly "left unencrypted" in the second encryption round are public information in any case; if the terminal and user attribute namespaces are adequately separated, third option does not enable inside users to decrypt material independently of the terminal either (as user private keys should not be credentials to terminal attributes).

Using the third option in Layered ABE will result in the relative performance characteristics given in Table 16.

**Table 16.** Layered ABE relative performance in OLP

| | |
|---|---|
| Ciphertext size | $(2n_c + 4)G + 2\|\rho(\cdot)\|$ |
| Private key size | $(u{+}t{+}4)G$ |
| (Global public) key size | $3G$ |
| Encryption | $(3n_c + 4)E + (n_c + 2)M + n_cH + 2P + (n_u^2 + n_t^2)Z$ |
| Decryption | $n_kE + (n_k + 2)M + (2n_k + 2)P + {}^2/_3\,(n_{ku}^3 + n_{kt}^3)Z$ |

[62] roughly doubling the sizes and encryption time, as symmetric encryption does not increase element sizes, and in each both of the cases the second layer CP-ABE is fed only one element to be encrypted

Table 16 gives the performance in terms of group elements and types of operations; the other parameters used are as follows:

- $n_u$: number of one-use attributes used for encryption (policy in ciphertext $|P_u|$, for publisher user $u$)
- $n_t$: number of one-use attributes used for encryption (policy in ciphertext $|P_t|$, for publisher terminal $t$)
- $n_c = n_u + n_t$
- $n_{ku}$: number of key attributes needed to satisfy the encryption policy, for the user
- $n_{kt}$: number of key attributes needed to satisfy the encryption policy, for the terminal
- $n_k = n_{ku} + n_{kt}$
- $u,t$: number of attribute credentials granted to user (resp. terminal, for subscriber-end)
- $H$: hash function evaluation
- $M$: bilinear group multiplication
- $E$: bilinear group exponentiation
- $P$: bilinear pairing function evaluation
- $Z$: Integer residue group multiplication (addition neglected here, due to the low complexity, compared to multiplication). These operations are needed to create and reconstruct the shares used in the policy encoding into the scheme attributes.
- $G$: Group element size at the given security level
- $\rho(\cdot)$: description of the access matrix and corresponding mapping function, changed whenever per-document policy changes, so need to be included in every ciphertext

The decryption step's share reconstruction includes a step requiring essentially matrix inversion using Gauss elimination resulting in cubic dependency of $Z$.

KPD-ABE uses LW-ABE encryption and decryption procedures, and additionally extra key material and operations to pool user and terminal keys. Table 17 uses the same notation as Table 16. Additional notation includes the use of $N_A$: total number of attributes in use in the system. It can be seen that, at least in the relative inspection, the added flexibility of using an integrated protection and release policy in KPD-ABE comes with a performance penalty partly due to the less efficient LW-ABE scheme and partly due to the benefit of dividing the total policy into two smaller components (more easily handled) in the layered approach. Addi-

tionally, the selected algebraic group has immense implications to the performance.

**Table 17.** KPD-ABE relative performance in OLP

| | |
|---|---|
| Ciphertext size | $(3n_c + 1)G + \lvert\rho(\cdot)\rvert$ |
| Private key size | $(u+t)G$ |
| (Global public) key size | $(2N_A + 1)G$ |
| Encryption | $(5n_c + 1)E + (2n_c + 1)M + (2n_c + 1)P + n_c^2 Z$ |
| Decryption | $n_k E + (3n_k + 1)M + n_k H + (2n_k + 1)P + {}^2/_3\, n_k^3 Z$ |

On the scheme level (which overlooks the actual, rather different, implementation specifics of the schemes) the comparison between t1 and t2 show that:

- Ciphertext size becomes about 50% larger with KPD-ABE than with Layered ABE
- Private key sizes are approximately equal
- The global public key size is constant in the layered approach, whereas the KPD-ABE's underlying LW-ABE attribute public keys are counted into the global keys, increasing their size markedly.
- Encryption time in KPD-ABE suffers heavily from the distributed nature of the LW-ABE scheme and LW-ABEs overuse of the expensive pairing operation, which has been optimized to a constant number in the CP-ABE used in Layered ABE.
- Decryption time becomes nearly equal, due to the closeness of the number of pairing operations and with only slight deviations in other operations.

The absolute bandwidth performance of Layered ABE and KPD-ABE within OLP was estimated by implementing a dual-policy ABE scheme by Attrapadung [22] using a python[63] (Charm [5]) implementation on top of the PBC cryptographic library [137] and changing the underlying ellip-

[63] A C-language implementation was also used, but it turned out to be more difficult to implement DP-ABE this way on top of PBC (as a C-language software project).

tic curves according to scheme and desried security level. The reasons for using DP-ABE are:

- DP-ABE uses exactly the same scheme for the subjective (ciphertext-) policy as Layered ABE, making the results easy to translate.
- It is simpler, simulation-wise, to switch between key- and ciphertext-policy evaluations.
- Should a proxy DP-ABE exist (see the end of this chapter) to enable a more comprehensive and flexible solution, the results give an indication on how such a scheme would perform.

The benchmark measurements to establish a baseline to be used for computing scheme- and security-level-dependent estimates were performed first. In this measurement we used the MNT-224 curve by Page, Smart and Vercauteren [165], corresponding to a security level of about about 100 bits [165]. We note that these benchmark measurements do *not*, by themselves, represent OLP performance. Measurements are depicted in Figure 38. The actual more accurate (average) sizes are, for ciphertext and private keys, respectively:

- $CT = 144|\omega| + 150|\psi|$ bytes (B)
- $K = 186|\omega| + 132|\psi|$ bytes (B)

Here $\omega$ and $\psi$ are the objective and subjective attribute-sets used (duplicate use included), respectively, using DP-ABE notation. If $\omega = \emptyset$ above, the formulas above describe ciphertext- (*CT*) and private key (*K*) sizes for the CP-ABE in Layered ABE. CT needs an additional factor of two to account for the "unrestricted" ABE and both *K* and *CT* a factor of two for the layered use of ABE in OLP.

We tested $10^5$ encryption runs, enumerating different policy formulas, but only the actual number of attribute instances appeared to matter. We extracted a benchmark-policy from an existing CPR use case related to software-defined networking. A typical release policy there was built from 4-5 high-level rules, together consisting of 12 high-level statements. However, some of the statements were expanded up to about 30 different attributes (e.g. the list of NATO nations), making the number of attributes to be in the order of 30-100 (in the release policy). Protection policies

were of the same order, thus our benchmark number of attributes was selected to be 50, to be the cross-section point of measurements.

The actual byte-number is dependent also on the actual presentation and serialization method used (e.g. attribute "`one`" consumes fewer bytes than attribute "`NATO-member-nation`"), so the formulas should not be taken as exact numbers for all cases. Subjective and objective access structures are both accounted for in the test runs, making possible their use with the relative figures mentioned in Tables Table 16 and Table 17.

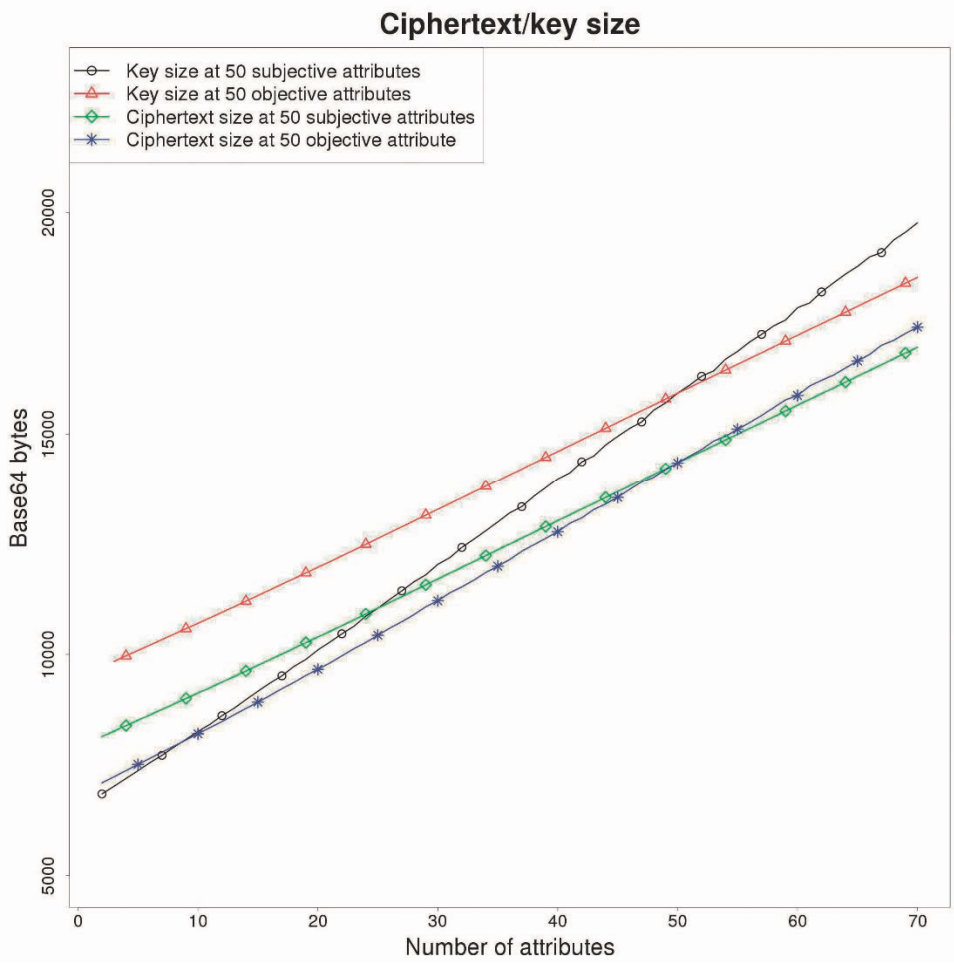



**Figure 38.** DP-ABE bandwidth measurements, at 50-attribute benchmark [124]

Using the 50-attribute benchmark and the extra layering factors it was possible to estimate the Layered ABE performance in OLP for the 100-bit security level. To extend this result to higher security levels, sufficient

size of elliptic curve groups with a given extension field size (here: 6) in general grows comparably to that of RSA (as a function of the security parameter). Luckily, increasing the extension field size proves to be more efficient than increasing elliptic curve group bit-length. Using the curves mentioned by Scott [192] for higher security levels, we can obtain the ciphertext- and key-size estimates for Layered ABE with policy complexity of 50 attributes (including both release- and protection policies) to be as shown in Table 18.

**Table 18.** Layered ABE absolute performance in different security levels at 50-attributes benchmark

| Security level (bits) | 100 | 128 | 192 | 256 |
|---|---|---|---|---|
| Elliptic curve used | MNT-224 | BN-128 | KSS-192 | BLS-256 |
| EC group size (bits) | 224 | 256 | 512 | 640 |
| Ciphertext length | 15 kB | 17 kB | 34 kB | 43 kB |
| Private key length | 7 kB | 8 kB | 15 kB | 19 kB |

KPD-ABE uses composite-order bilinear groups, which means that the equivalent key length for a certain security level is directly that of RSA, as the security depends on the (difficulty of) factorization of the group order. The software library we used, only supports groups up to 1024 bits, and even there only supersingular groups. We only estimate the bandwidth efficiency, so this is sufficient, but we had to extrapolate over 80-bit security (trivial, since the group size affects linearly to the bandwidth efficiency). We show the estimates for key- and ciphertext-sizes for KPD-ABE using 50 attributes in Table 19. It can be seen that the requirement of having composite groups halves the bandwidth performance at low security levels and drops it to nearly 10% of layered ABE at the higher levels.

The absolute computational performance of the schemes is difficult to estimate, even with a standardized computation platform and test cases, as multiple implementation- and scheme-level optimizations may drastically alter the situation.

**Table 19.** KPD-ABE absolute performance in different security levels at 50-attributes benchmark

| Security level | 100 | 128 | 192 | 256 |
|---|---|---|---|---|
| EC group size (bits) | 1344 | 2540 | 6710 | 13540 |
| Ciphertext length | 34 kB | 65 kB | 171 kB | 345 kB |
| Private key length | 8,6 kB | 16 kB | 43 kB | 87 kB |

The most time-consuming operation in ABE implementations is usually the pairing function evaluation (*P*). Thus schemes which require a number of them per scheme-level operation are usually more expensive to execute. However, the performance penalty may often be not as drastic as implied in the relative performance: precomputation together with consecutive applications of the pairing can reduce the number of completely new pairings considerably.

As shown by Scott [192], given certain carefully selected elliptic curves[64] it is possible even to push the cost of *P* to the level of *E* (exponentiation) in the bilinear group domain. Scott has demonstrated [192] that the ABE scheme used in Layered ABE, with small to moderate number of attributes (12 to 20), and even with the highest security level of 256 bits on a 2.4 GHz single-core processor takes less than 0.2s.This implies that a single quad-core COTS-laptop could make tens of release-/protection policy enforcements per second at the highest security level (hundreds, if 128-bit security is sufficient).

The cost of *Z* is mostly neglected in the actual cryptographic scheme descriptions. To ascertain that this assumption is valid even for large policies, *Z* can be compared to *E* and the total time estimated relative to exponentiation operations. The relation between *Z* and *E* can be estimated from the number of point-doubling operations in the EC group [106]: one point doubling (*D*) requires approximately one field element inversion (*I*) and two field multiplications (*Z*). Inversion requires $\log_2(q)$ multiplications ($q$ being the field size in bits) and elliptic curve group exponentiation again $\log_2(q)$ point doublings. Thus $E \approx (\log_2 q)^2 Z$, and extrapolating, we may get a “break-even point” (where the share recovery gets more

[64] called pairing-friendly curves

expensive than other exponentiations in decryption) for policies of little less than 200 attributes, for “usual” 224-bit curves. For larger curves, this break-even is even higher (~ 880 attributes for 1024-bit curves), which makes the concern of the share reconstruction for large policies void, unless some *very* large and detailed policies are used.

The computational cost is, in general, highly dependent on the underlying elliptic-curve and pairing implementations, which in turn are typically not comparable over different size security parameters. The most efficiently implemented curves for one security parameter may have wildly different characteristics to an efficient curve in another security parameter setting. For example in the measurements performed by Scott [192], the computational performance may actually *increase*, when moving from 80-bit security to 128-bit security (in most cases, however, the computational cost increased linearly with respect to the security parameter). Thus especially, when the use profile is more low-bandwidth rather than small-power, we do not consider computational cost to be of large importance.

In OLP and in publishing process in general it is likely to be easier to support dynamic policies (which appear to be a current trend in access control), if both the encryptor (publisher) and the decryptor (subscriber) use only attributes without policies for their functions. The policy should be embedded in the content only very close to the delivery of the content to the subscriber.

These kinds of ideas, however, require again an architecture with a centralized PEP, and also such ABE schemes that are capable of doing ciphertext transformations without actually decrypting, acting as a proxy encryption scheme. However, currently no dual-policy proxy-ABEs are known.

Thus it is also seen that the current distributed XACML-architecture may not be ideal for even more dynamic policy actions than envisioned currently, but this limitation is today also necessary due to the limited functionality of existing ABE-schemes. Future research should tackle proxy ABEs as its first task.

# 10. Conclusions

In this work we researched the feasibility of cryptographically enforced role-based access control using state-of-the-art functional cryptography schemes, in a distributed, multi-level security setting. Cryptographic access control is a somewhat narrowly understood concept from the implementation perspective. Thus we explicitly defined, what is meant by cryptographic access control in general and in particular if it is pervasive, i.e., addressing different access control elements cryptographically more widely than merely the access check. This places certain restrictions on the basic premises of what actually can be enforced and also fundamentally changes which elements are responsible for which actions. In this study we showed that if cryptographic access control is to be taken into a more wide-spread use, it requires a profound change in thinking of how access control objects and operations are perceived, and how they should be handled. On the positive side, if it is possible to change the perception, it provides natural methods of enforcing security in the most challenging environments for traditional perimeter security, such as internet-of-things, cloud services, and multi-level security. We investigated the current state of cryptographic access control and found that the concept of pervasiveness is only beginning: only few schemes were found that consider enforcing even one of the RBAC functionalities cryptographically, and only one scheme considered the whole of (simplified, core) RBAC security in cryptographic terms.

The main concepts of changing perception towards pervasive CAC include the capability of cryptography to support confidentiality and integrity only, the passivity of cryptographic controls (the need for an active process to act on cryptographic information or cryptographically transformed information of the protected object), moving the responsibility of the availability of content completely to cloud services (or similar), and distributing many of the traditionally centralized concepts, for example, XACML reference architecture policy enforcement point. Distribution reaches even as far as the permission types themselves, which we show to be possible to be translated into sets of **`read`**s and **`write`**s, combined with suitable metadata.

The transition of current systems to the new paradigm involves a shift from conventional PKI to functional cryptography. This shift is more profound than it at first appears, as it involves a change in public-key authentication architectures from the conventional PKI PKAA to the IBE PKAA. In this research we achieved two results related to this challenge. First we investigated whether the ability to support attributes, an essential element if roles or user capabilities are to be expressed, is particular only to ABE, and found that it is rather straightforward to support attributes as such in other public-key authentication architectures (PKAA) as well (even though it appears that this line of thinking was uncommon at the time, in view of the lack other similar constructions). On another vein, we also separated the PKAA choice from actual document management by developing principles for handling MLS documents in a web-services context and by defining a suitable PKAA-independent template (an XML schema) for MLS documents using CAC-support.

As for the main body of our work, we showed how the ANSI standardized $RBAC_3$ main functionalities and elements can already be mostly supported by existing functional cryptography schemes (with both ABE and ABS). We divided our research for **`write`**- and **`read`**-permissions (or integrity and confidentiality policies, respectively) into two independent works for simplicity, but it was already evident that enforcing compound integrity- and confidentiality policies is more challenging than merely attaching ABS signatures to ABE-encrypted documents. Both of the mappings also included an embedding of ABE-or ABS-elements into XACML-compliant architecture, meaning that the XACML reference architecture is, in fact, general enough to support also pervasive CAC. In both cases some of the fundamental limitations of CAC were visible: after the cryptographic transformation, making changes to content or metadata becomes difficult, making such concepts as dynamic separation of duty, dynamic role hierarchies or policy changes in the middle of document lifecycle difficult or requiring alternative solutions. However, attribute-based cryptography and, more generally, functional cryptography were shown to be adequate choices for CRBAC schemes.

We investigated many of the security models in FE, ABE, FS and ABS, and found that the leap in expressiveness from complexity class $\mathbf{NC^1}$ to **NC** also often seems to require a leap in the security models from game-based security to simulation based security. As the simulation-based secu-

rity models are yet somewhat debatable and the required constructions less efficient than those schemes using game-based models, we conclude that for the MLS case, game-based security model is sufficient (provided the scheme itself "fits" under game-based security definitions).

We also found that, with respect to ABAC, the intuition of having a straightforward mapping from ABE and ABS to ABAC does not withstand closer scrutiny. This follows from the inability to fully support some dynamic RBAC features, such as DSD, meaning that the same functionalities would be lacking in ABAC as well. In a more generalized point of view, this casts doubt whether it is possible to model constrained RBAC as a multi-party computation scenario, or if additional measures are required.

MLS appeared to be a context, which is not very much considered by the mainstream functional cryptography work: such functionalities as content redaction and combined user-terminal policies are poorly, if at all supported. We presented a solution for the latter functionality in the form of a provably-secure key-management scheme using a particular ABE-scheme as the basis.

We summarize the publications with respect to the perspectives and research questions and some other natural questions arising here in a more detailed format in Table 20 and Table 21 below.

**Table 20.** Research findings grouped according to the research questions group by perspective

| Perspective | Research question | Findings |
|---|---|---|
| **Architectural** | What kind of architecture and architectural elements in XACML and publish-subscribe need to be supported, if access control to MLS-documents is to be enforced with CAC, instead of RM? | The existing publish-subscribe architectures suffice for CAC as well. The roles of the different elements are somewhat shifted, though. As CAC performs many functions at the edge of the channel (or storage) function, reduced architectures without border brokers are insufficient. Likewise, publish-subscribe processes with CAC can be adapted to XACML framework in a straightforward manner. The roles and duties of each element need to be revised. The more detailed architecture (below the XACML level) will require additional elements due to the use of ABE / ABS. These include attribute- and pooling authorities, for instance. |

| | | |
|---|---|---|
| | Are the responsibilities of different architectural elements (in publish-subscribe and XACML) the same for CAC as without CAC? If not, what are the main differences? | The responsibilities are shifted. The two main issues to consider are the responsibility for availability and integrity. These are intertwined in the manner that availability in general is seen solely as the concern of the storage, but the availability of uncorrupted content needs to be performed jointly with user or border broker processes to determine, if the provided instance of content is uncorrupted. Thus the integrity is checked closer to the user than initially. Additionally, key management is assumed to be part of the user domain (either directly at the user process or in the border brokers). A third main difference is the distributed function of policy enforcement: checking the **`write`**-permission requires first the signing and then verification. These are, however, performed at separate domains in the publish-subscribe architecture. |
| **Document management, MLS** | Are ABE and ABS the only possible choices? Are there other mechanisms to support attributes? | Many types of cryptographic schemes are *possible*, but ABE and ABS appear to be the most efficient and versatile for document management purposes. Implicit certification schemes can even support attributes in the same manner as ABE. Attribute certificates and conventional PKI are also viable, but their level of policy encoding enforcement is the lowest of these choices. |
| | Is it possible to support transition from PKI-protected MLS-documents to ABE-/ABS-protected documents with XML? | Yes. An example of a document format supporting MLS and both PKI and ABE/ABS was given in [120], Chapter 5.3 |
| | Which MLS-functionalities can be accounted for? Does there arise any new challenges when using ABE/ABS? | Using XML and ABE, at least document labelling, label bindings, redaction and information flow separation can be performed. User clearance can only be cryptographically enforced, if it can be encoded in the key material. Thus pervasive CAC needs key-policy functionality to support MLS fully. Migrating from PKI or purely symmetric key management to ABE / ABS is not foreseen to present new challenges (in addition to the actual migration work) compared to the conventional functionality provided by PKI-based MLS enforcement systems. |
| | From document management perspective, what are the major differences in using CAC (instead of using RM)? | These differences can be grouped according to different document lifecycle events: during document publishing phase, policies for different roles need to be clarified before encryption and signing; Document modification and publishing processes |

| | | |
|---|---|---|
| | | need to be clearly separated and integrated into version control; permission revocation procedures nearly always currently require re-encryption of key material or content at some level; Backup management is, however, intrinsically woven into the Storage assumptions, and becomes more reliable than with RM. |
| **Modelling** | Can other permission types than just **`read`** and **`write`** be enforced cryptographically and how? Is it equally efficient to support different types? | Many permission types are actually metadata for document management, and if they are partitioned into **`read`**- and **`write`**-permissions of content and different levels of metadata, other types can be translated into a set of **`read`**- and **`write`**-permissions. Not all permission types are efficient (or even known in detail how) to support with CAC even when translated. Such examples include **`execute`**, and (Win7) `Traverse-Folder`. Enforcing permissions requiring access to versioning history will have to be integrated with possible versioning solutions. |
| | Can the **`read`** (resp. **`write`**) permission be enforced in the publish-subscribe environment, where XACML-architecture and RBAC access control model are the defining factors? | Yes, but not fully, at least with current ABE- and ABS schemes. A publish-subscribe architecture adapted to the XACML framework is presented in Chapters 5.2, 7.1 and 8.2. ABE and ABS can enforce sufficiently arbitrary policies for typical access control needs (the only exception to this that we have found, is determining, whether location is in an arbitrary geographical area). There are, however, policies that violate the security models of ABE and ABS (such as attribute combination across users and visibility of signing policy) which cannot be implemented, at least not without additional schemes. |
| | Can the standard RBAC-commands and elements be implemented with existing ABE- (/resp. ABS-) schemes? Which cannot? | Most elements are implementable. Exceptions include, for confidentiality enforcement: hierarchical RBAC dynamics (adding or removing administrative roles); and constrained RBAC dynamic separation of duty, unless the encryption scheme supports non-monotonic ciphertext-policies. For integrity enforcement, hierarchical RBAC can only be enforced with schemes supporting delegation and *any* constrained RBAC element requires non-monotonic claim predicates (but unlike with confidentiality enforcement, dynamic separation of duty can be equally well supported). |
| | Does using CAC (instead of RM) imply any profound access control | Two main changes are: 1) combining attributes is considered an attack in ABE. It can be accomplished by using CP-ABE in a layered manner or |

| | | |
|---|---|---|
| | policy handling changes? | by using a key-management scheme presented in [117]. 2) With RM, policies are managed centrally, while with ABE and ABS, the different sides (encryptor/signer and decryptor/verifier) have their distinct view of acceptable access control policy, which may or may not be shared (with each other or centrally). |
| **Efficiency and sec.** | Does the security model of some particular scheme allow "normal" dynamics of an ICT system, i.e. multiple instantiation, peering, change of different principals and system attributes (or even the *use* of typical system attributes, such as complex policies) efficiently? | Different schemes are optimized for different tasks. Thus there is no single known scheme that can cater to all of the services, but for all of the functionalities investigated here, save the extended version of RBAC UA enforcement (see the end of this chapter), there are several optimized schemes. Security-wise, only such schemes should be considered, which provide so-called "full" (=non-selective) security, as this severely restricts policy-update processes. (The security assumptions underneath may, however, require more exotic or impractically large elliptic curves, as is the case with composite-order ECGs) Furthermore, distributed management of attributes and other administrative functions are typical simplification points for many schemes to achieve provable security. |
| | Given "normal" system operation, what are the relative processing delay and bandwidth overheads for a scheme? In particular, the overhead should be at most in the same order of magnitude as the parameters of the system *without* the scheme. | ABE and ABS implementations rely customarily heavily on pairing functions, which tend to be inefficient. Also the bandwidth overhead is in the order of kilobytes, and depending on the actual curves, their embedding degrees and policy complexity, may even approach megabytes. However, compared to other document management processes, the computational overhead is not usually an issue. In distributed environments the bandwidth overhead becomes predominant, and for a typical modern document size an additional overhead in the range of tens or even hundreds of kilobytes might be acceptable. We have not yet encountered a scheme which would, with moderate-size policies (at most tens of attributes) exceed the megabyte-limit by itself. |

**Table 21.** Summary ABS and ABE support for RBAC by element and command, using "traffic-light" notation

| RBAC-command | read support | write support | RBAC-element | read support | write support |
|---|---|---|---|---|---|
| AddRole | -[65] | - | Object | - | - |
| Grant-Permission | Good | Good | Operation | - | - |
| AddUser | - | - | Permission | Good | Good |
| AssignUser | Good | Good | User | - | - |
| CreateSession | Good | Good | Role | Good | Good |
| AddActiveRole | Good | Good | Orphan session | Good | Good |
| CheckAccess | Good | Good | Active session | Good | Good |
| DropActiveRole | Good | Good | PA | Good | Good |
| DeleteSession | Good | Good | UA | Good | Good |
| DeassignUser (with loss of auth) | Medi-ocre | Med. | Role Hierarchy | **Poor** | **Poor** |
| DeleteUser | Good | Good | Admin role | - | - |
| Revoke-Permission | Medi-ocre | Good | SSD | Med. | Med. |
| DeleteRole | Good | Good | DSD | **Poor** | Med. |

One shortcoming of the architecture presented here is that policy changes for confidentiality policies require re-encryption at the Storage. However, if the ciphertext elements in the Storage are (computationally) independent from those sent to the user processes, this re-encryption need not be done. One solution for this is to encrypt the content using attributes only (as in KP-ABE), and transform the resulting ciphertext according to re-encryption principles into such that it can be decrypted with a key consisting of attributes only (as in CP-ABE). This has other advantages as well, such as keeping the ciphertext overhead smaller in the Storage (since the

[65] We evaluate only CAC-functions, not those which are equivalent whether or not CAC is used.

policy would be encoded into the ciphertext only in the re-encryption phase). This solution seems perfectly possible, given the existence of dual policies in ABE, even using standard FE security proof methodologies, and is one of the most immediate directions of future work. On the architectural side, the different combinations of integrity- and confidentiality policies are far from trivial, and require more work.

In the course of writing this work it has become apparent that the capabilities provided by attribute-based cryptography are likely to represent the *minimum* functionality (rather than a "nice" set of features) to be able to support RBAC in a pervasive manner at all. Evidence to this statement includes the requirement to support user collusion prevention (in order to provide RBAC session separation), the need to support encryption to unknown entities and the impossibility result by Maurer's study [141] showing that as versatile functionality as the one provided by FE cannot securely be implemented with conventional PKI.

When the history of CAC and functional cryptography are considered, it is possible to see a trend that both integrates ever more functional cryptography functionalities to solve practical access control problems, and develops more practical-oriented new functionalities. Parallel to this, CRBAC research seems to be evolving as an independent discipline, finally bringing access control concepts purely to cryptographic security models, which is exactly, what we pursued with the pervasiveness concept in this research. All in all, the current work should also be seen as a next iteration of what would one day become a usable, but secure system to be able to handle also the most sensitive information.

# References


[1] [nickname:] "Atari Vampire": *WinDVD 8 Device Key found!* Doom9 Forum, *http://forum.doom9.org/showthread.php?t=122664,* Accessed: 2016/Jan/25, (2007)

[2] Adamouski, F. J.*: Encryption technology other than PKI*. In: Sanson, L.D. (ed.) 32nd Annual IEEE International Carnahan Conference on Security Technology, Alexandria, VA, USA, October 12-14, 1998. Proceedings , pp. 108-116 (1998)

[3] Advanced Access Content System Licensing Administrator (AACS LA): *AACS Specifications (Final Specifcations). http://www.aacsla.com/specifications/,* Accessed: 2016/Jan/23, (2012)

[4] Agrawal, S., Gorbunov, S., Vaikuntanathan, V., Wee, H.*: Functional encryption: New perspectives and lower bounds.* In: Canetti, R. and Garay, J.A. (eds.) 33rd Annual Cryptology Conference - CRYPTO 2013, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, pp. 500-518. Springer (2013)

[5] Akinyele, J. A., Garman, C., Miers, I., Pagano, M. W., Rushanan, M., Green, M., Rubin, A. D.: Charm: A Framework for Rapidly Prototyping Cryptosystems. *Journal of Cryptographic Engineering,* **3**(2): 111-128, IEEE. (2013)

[6] Akl, S. G., Taylor, P. D.*: Cryptographic Solution to a Multilevel Security Problem*. In: Chaum, D., Rivest, R. and Sherman, A.T. (eds.) Workshop on the Theory and Application of Cryptographic Techniques - CRYPTO 1983, Santa Barbara, CA, USA, August 21-24, 1983. Proceedings, pp. 237-249. Plenum, New York. (1983)

[7] Akl, S. G., Taylor, P. D.: Cryptographic Solution to a Problem of Access Control in a Hierarchy. *ACM Transactions on Computer Systems,* **1**(3): 239-248, ACM. (1983)

[8] Al-Kahtani, M. A., Sandhu, R.*: Rule-based RBAC with negative authorization*. In: Thomsen, D., Schuba, C. and Samarati, P. (eds.) 20th Annual Computer Security Applications Conference - ACSAC 2004, Tucson, AZ, USA, December 6-10, 2004. Proceedings, pp. 405-415 (2004)

[9] Al-Riyami, S. S., Malone-Lee, J., Smart, N. P.: Escrow-Free Encryption Supporting Cryptographic Workflow. *International Journal of Information Security,* **5**(4): 217-229, Springer. (2006)

[10] Al-Riyami, S. S., Paterson, K. G.*: Certificateless Public Key Cryptography*. In: Laih, C. (ed.) 9th International Conference on the Theory and Application of Cryptology and Information Security - ASIACRYPT 2003, Taipei, Taiwan, November 30 – December 4, 2003. Proceedings, pp. 452-473. Springer Berlin Heidelberg, Berlin, Heidelberg. (2003)

[11] Anada, H., Arita, S., Sakurai, K.*: Attribute-based Signatures Without Pairings via the Fiat-shamir Paradigm*. In: Moriai, S., Jaeger, T. and Sakurai, K. (eds.) 2nd ACM Workshop on ASIA Public-Key Cryptography - ASIAPKC 2004, Kyoto, Japan, June 4-6, 2014. Proceedings, pp. 49-58. ACM, New York, NY, USA. (2014)

[12] Ananth, P., Bhaskar, R.*: Non Observability in the Random Oracle Model*. In: Susilo, W. and Reyhanitabar, R. (eds.) 7th International Conference of Provable Security - PROVSEC 2013, Melaka, Malaysia, October 23-25, 2013. Proceedings, vol. 8209, pp. 86-103. Springer Berlin Heidelberg (2013)

[13] Anderson, J.*: Computer security technology planning study*. ESD-TR-73-51, Vol. 2. US Air Force Electronic Systems Division, Massachussetts. (1972)

[14] Anderson, R. J.*: Security engineering: A guide to building dependable distributed systems*. 2nd edn. Wiley Publishing (2008)

[15] ANSI: *American National Standard for Information Technology - Role Based Access Control (INCITS 359-2012).* ANSI Standards, 61 pages, (2012)

[16] Armando, A., Ranise, S., Traverso, R., Wrona, K.*: Compiling NATO authorization policies for enforcement in the cloud and SDNs*. In: Li, M. (ed.) IEEE Conference on Communications and Network Security - CNS 2015, San Francisco, CA, USA, October 21-24, 2014. Proceedings, pp. 741-742 (2015)

[17] Armando, A., Grasso, M., Oudkerk, S., Ranise, S., Wrona, K.*: Content-based Information Protection and Release in NATO Operations*. In: Conti, M., Vaidya, J. and Schaad, A. (eds.) 18th ACM Symposium on Access Control Models and Technologies - SACMAT 2013, Amsterdam, The Netherlands, June 12-14, 2013. Proceedings, pp. 261-264. ACM, New York, NY, USA. (2013)

[18] Armando, A., Oudkerk, S., Ranise, S., Wrona, K.*: Formal Modelling of Content-Based Protection and Release for Access Control in NATO Operations*. In: Danger, L.J., Debbabi, M., Marion, J., et al (eds.) 6th Interna-

tional Symposium on Foundations and Practice of Security - FPS 2013, La Rochelle, France, October 21-22, 2013, Revised Selected Papers, pp. 227-244. Springer International Publishing, Cham. (2014)

[19] Atallah, M. J., Blanton, M., Fazio, N., Frikken, K. B.: Dynamic and Efficient Key Management for Access Hierarchies. *ACM Transactions on Information System Security (TISSEC)*, **12**(3): 1-43, ACM. (2009)

[20] Ateniese, G., Santis, A., Ferrara, A. L., Masucci, B.: Provably-Secure Time-Bound Hierarchical Key Assignment Schemes. *Journal of Cryptology*, **25**(2): 243-270, IACR. (2010)

[21] Attrapadung, N.: Dual-Policy Attribute Based Encryption: Simultaneous Access Control with Ciphertext and Key Policies. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **93**(1): 116-125, The Institute of Electronics, Information and Communication Engineers. (2010)

[22] Attrapadung, N., Imai, H.*: Attribute-based encryption supporting direct/indirect revocation modes.* In: Cryptography and Coding, pp. 278-300. Springer Berlin Heidelberg (2009)

[23] Attrapadung, N., Libert, B.: Functional Encryption for Public-Attribute Inner Products: Achieving Constant-Size Ciphertexts with Adaptive Security Or Support for Negation. *Journal of Mathematical Cryptology*, **5**(2): 115-158, (2012)

[24] Backes, M., Meiser, S., Schröder, D.: *Delegatable Functional Signatures.* Cryptology ePrint Archive, Report 2013/408, (2013)

[25] Bacon, J., Moody, K., Yao, W.: A Model of OASIS Role-Based Access Control and its Support for Active Security. *ACM Transactions on Information and System Security (TISSEC)*, **5**(4): 492-540, ACM. (2002)

[26] Baigneres, T., Vaudenay, S.*: The Complexity of Distinguishing Distributions (Invited Talk)*. In: Safavi-Naini, R. (ed.) 3rd International Conference on Information Theoretic Security - ICITS 2008, Calgary, Canada, August 10-13, 2008. Proceedings, pp. 210-222. Springer Berlin Heidelberg, Berlin, Heidelberg. (2008)

[27] Barbosa, M., Farshim, P.*: On the semantic security of functional encryption schemes*. In: Public-Key Cryptography--PKC 2013, pp. 143-161. Springer Berlin Heidelberg (2013)

[28] Barrantes, E. G., Ackley, D. H., Palmer, T. S., Stefanovic, D., Zovi, D. D.*: Randomized Instruction Set Emulation to Disrupt Binary Code Injection Attacks*. In: Atluri, V. and Jaeger, T. (eds.) 10th ACM Conference on Computer and Communications Security - ACM CCS 2003, Washington DC, USA, October 27-30, 2003. Proceedings, pp. 281-289. ACM, New York, NY, USA. (2003)

[29] Bartel, M., Boyer, J., Fox, B., LaMacchia, B., Simon, E.: *XML-Signature Syntax and Processing.* W3C Recommendations, *http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/Overview.html,* Accessed: 2016/Feb/6, (2002)

[30] BBC News: *Hi-Def DVD Security is Bypassed. http://news.bbc.co.uk/2/hi/technology/6301301.stm,* Accessed: 2016/Jan/25, (2007)

[31] Bell, D., LaPadula, L.*: Secure Computer Systems: Mathematical Foundations,* MITRE Technical Report, MTR-2547, 33p. (1973)

[32] Bellare, M., Fuchsbauer, G.*: Policy-Based Signatures*. In: Krawczyk, H. (ed.) 17th International Conference on Practice and Theory in Public-Key Cryptography - PKC 2014, Buenos Aires, Argentina, March 26-28, 2014. Proceedings, vol. 8383, pp. 520-537. Springer Berlin Heidelberg (2014)

[33] Bellare, M., Rogaway, P.*: Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols*. In: Denning, D.E., Pyle, R., Ganesan, R., et al (eds.) 1st ACM Conference on Computer and Communications Security - CCS 1993, Fairfax, VA, USA, November 3-5, 1993. Proceedings, pp. 62-73. ACM, New York, NY, USA. (1993)

[34] Bertino, E., Carminati, B., Ferrari, E., Thuraisingham, B., Gupta, A.: Selective and Authentic Third-Party Distribution of XML Documents. *Knowledge and Data Engineering, IEEE Transactions on,* **16**(10): 1263-1278, (2004)

[35] Bethencourt, J., Sahai, A., Waters, B.*: Ciphertext-policy attribute-based encryption*. In: Pfitzmann, B. and McDaniel, P. (eds.) 2007 IEEE Symposium on Security and Privacy - IEEE S&P 2007, Berkeley, CA, USA, May 20-23, 2007. Proceedings, pp. 321-334. IEEE (2007)

[36] Birman, K., Joseph, T.*: Exploiting Virtual Synchrony in Distributed Systems*. In: Belady, L. (ed.) 11th ACM Symposium on Operating Systems Principles - SOSP 1987, Austin, TX, USA, November 8-11, 1987. Proceedings, pp. 123-138. ACM, New York, NY, USA. (1987)

[37] Bishop, M., Snyder, L.*: The Transfer of Information and Authority in a Protection System*. In: Schroeder, M.D. and Jones, A.K. (eds.) 7th ACM Symposium on Operating Systems Principles - SOSP 1979, Pacific Grove, CA, USA, December 10-12, 1979. Proceedings, pp. 45-54. ACM, New York, NY, USA. (1979)

[38] Bishop, M. A.*: The art and science of computer security*. Addison-Wesley Longman Publishing Co., Inc, Boston, MA, USA. (2002)

[39] Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.*: Recursive Composition and Bootstrapping for SNARKS and Proof-carrying Data*. In: Boneh, D. and Roughgarden, T. (eds.) 45th Annual ACM Symposium on Theory of Computing - STOC 2013, Palo Alto, CA, USA, June 01 - 04, 2013. Proceedings, pp. 111-120. ACM, New York, NY, USA. (2013)

[40] Bobba, R., Fatemieh, O., Khan, F., Khan, A., Gunter, C. A., Khurana, H., Prabhakaran, M.: Attribute-Based Messaging: Access Control and Confidentiality. *ACM Transactions on Information and System Security*, **13**(4): 31:1-31:35, ACM. (2010)

[41] Bobba, R., Khurana, H., Prabhakaran, M.*: Attribute-Sets: A Practically Motivated Enhancement to Attribute-Based Encryption*. In: Backes, M. and Ning, P. (eds.) 14th European Symposium on Research in Computer Security - ESORICS 2009, Saint-Malo, France, September 21-23, 2009. Proceedings, pp. 587-604. Springer Berlin Heidelberg, Berlin, Heidelberg. (2009)

[42] Bonderud, D.: *Data Sabotage: The Serious Security Risk of Smart Cities. https://securityintelligence.com/data-sabotage-the-serious-security-risk-of-smart-cities/*, Accessed: 2017/Jan/6, (2016)

[43] Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. *SIAM Journal on Computing*, **36**(5): 1301-1328, Society for Industrial and Applied Mathematics. (2006)

[44] Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. *SIAM Journal on Computing*, **32**(3): 586-615, (2003)

[45] Boneh, D., Gentry, C., Waters, B.*: Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys*. In: Shoup, V. (ed.) 25th Annual Cryptology Conference - CRYPTO 2005, Santa Barbara, CA, USA, August 14-18, 2005. Proceedings, pp. 258-275. Springer Berlin Heidelberg, Berlin, Heidelberg. (2005)

[46] Boneh, D., Goh, E., Nissim, K.*: Evaluating 2-DNF Formulas on Ciphertexts*. In: Kilian, J. (ed.) 2nd Annual Theory of Cryptography Conference - TCC 2005, Cambridge, MA, USA, February 10-12, 2005. Proceedings, pp. 325-341. Springer-Verlag, Berlin, Heidelberg. (2005)

[47] Boneh, D., Sahai, A., Waters, B.*: Functional encryption: Definitions and challenges*. In: Yuval, I. (ed.) 8th International Conference on Theory of Cryptography - TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings, vol. 6597, pp. 253-273. Springer Berlin Heidelberg, Berlin. (2011)

[48] Boneh, D., Silverberg, A.: Applications of Multilinear Forms to Cryptography. *Contemporary Mathematics,* **324**: 71-90, American Mathematical Society (AMS). (2003)

[49] Borghoff, J., Knudsen, L., Leander, G., Matusiewicz, K.*: Cryptanalysis of C2*. In: Halevi, S. (ed.) 29th Annual Cryptology Conference - CRYPTO 2009, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings, vol. 5677, pp. 250-266. Springer Berlin Heidelberg (2009)

[50] Boyen, X.*: Mesh Signatures*. In: Naor, M. (ed.) 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques - EUROCRYPT 2007, Barcelona, Spain, May 20 – 24, 2007., vol. 4515, pp. 210-227. Berlin: Springer-Verlag (2007)

[51] Boyle, E., Goldwasser, S., Ivan, I.*: Functional Signatures and Pseudorandom Functions*. In: Krawczyk, H. (ed.) 17th International Conference on Practice and Theory in Public-Key Cryptography - PKC 2014, Buenos Aires, Argentina, March 26-28, 2014. Proceedings, vol. 8383, pp. 501-519. Springer Berlin Heidelberg (2014)

[52] Brown, D. R. L., Gallant, R. P., Vanstone, S. A.*: Provably Secure Implicit Certificate Schemes*. In: Blaze, M. (ed.) 6th International Conference on Financial Cryptography - FC 2002, Southampton, Bermuda, March 11-14, 2002. Revised Papers, pp. 156-165. Springer-Verlag, London, UK. (2002)

[53] Canetti, R.: Security and Composition of Multi-Party Cryptographic Protocols. *Journal of Cryptology,* **13**(1): 143-202, Springer. (2000)

[54] Canetti, R., Fischlin, M.*: Universally Composable Commitments*. In: Kilian, J. (ed.) 21st Annual Cryptology Conference - CRYPTO 2001, Santa Barbara, CA, USA, August 19-23, 2001. Proceedings, pp. 19-40 (2001)

[55] Canetti, R., Hohenberger, S.*: Chosen-ciphertext Secure Proxy Re-encryption*. In: Ning, P., De Capitani di Vimercati, Sabrina and Syverson,

P.F. (eds.) 14th ACM Conference on Computer and Communications Security - CCS 2007, Alexandria, VA, USA, October 28-31, 2007. Proceedings, pp. 185-194. ACM, New York, NY, USA. (2007)

[56] Chase, M.*: Multi-authority Attribute Based Encryption*. In: Vadhan, S.P. (ed.) 4th Theory of Cryptography Conference - TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007. Proceedings, pp. 515-534. Springer-Verlag, Berlin, Heidelberg. (2007)

[57] Chenxi Wang, Carzaniga, A., Evans, D., Wolf, A. L.*: Security issues and requirements for Internet-scale publish-subscribe systems*. In: Sprague, R.H. (ed.) 35th Hawaii International Conference on System Sciences - HICSS 2002, Big Island, HI, USA, January 7-10, 2002. CD-ROM / Abstracts Proceedings, pp. 3940-3947 (2002)

[58] Cheon, J. H., Fouque, P., Lee, C., Minaud, B., Ryu, H.: *Cryptanalysis of the New CLT Multilinear Map Over the Integers.* Cryptology ePrint Archive, Report 2016/135, (2016)

[59] Chick, G. C., Tavares, S. E.*: Flexible Access Control with Master Keys*. In: Brassard, G. (ed.) 9th Annual International Cryptology Conference - CRYPTO 1989 Santa Barbara, California, USA, August 20–24, 1989. Proceedings, vol. 435, pp. 316-322. Springer New York (1990)

[60] Common Criteria Management Board (CCMB): *Common Criteria for for Information Technology Security Evaluation, Part 3: Security Assurance Components (September 2012), Version 3.1, Rev. 4* . Common Criteria Recognition Arrangement documents, 233p., *http://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R4.pdf*, Accessed: 2016/Feb/8, (2012)

[61] Coron, J., Gentry, C., Halevi, S., Lepoint, T., Maji, H. K., Miles, E., Raykova, M. et al.*: Zeroizing Without Low-Level Zeroes: New MMAP Attacks and their Limitations*. In: Gennaro, R. and Robshaw, M. (eds.) 35th Annual Cryptology Conference - CRYPTO 2015, Santa Barbara, CA, USA, August 16-20, 2015. Proceedings, pp. 247-266. Springer Berlin Heidelberg, Berlin, Heidelberg. (2015)

[62] Coron, J., Holenstein, T., Künzler, R., Patarin, J., Seurin, Y., Tessaro, S.: How to Build an Ideal Cipher: The Indifferentiability of the Feistel Construction. *Journal of Cryptology*, **29**(1): 61-114, Springer US. (2016)

[63] Coron, J., Lepoint, T., Tibouchi, M.*: Practical Multilinear Maps over the Integers*. In: Canetti, R. and Garay, J.A. (eds.) 33rd Annual Cryptology Conference - CRYPTO 2013, Santa Barbara, CA, USA, August 18-22,

2013. Proceedings, pp. 476-493. Springer Berlin Heidelberg, Berlin, Heidelberg. (2013)

[64] Crampton, J., Martin, K., Wild, P.*: On key assignment for hierarchical access control*. In: Herzog, J. (ed.) 19th IEEE Computer Security Foundations Workshop - IEEE CSFW 2006, Venice, Italy, July 5-7, 2006. Proceedings, pp. 98-111 (2006)

[65] Crampton, J.*: Cryptographic Enforcement of Role-based Access Control*. In: Barthe, G., Datta, A. and Etalle, S. (eds.) 8th International Workshop on Formal Aspects of Security and Trust - FAST 2011, Leuven, Belgium, September 12-14, 2011. Revised Selected Papers, pp. 191-205. Springer-Verlag, Berlin, Heidelberg. (2011)

[66] Crampton, J., Lim, H. W.*: Role Signatures for Access Control in Open Distributed Systems*. In: Jajodia, S., Samarati, P. and Cimato, S. (eds.) 23rd International Information Security Conference, IFIP 20th World Computer Congress, IFIP SEC 2008, Milano, Italy, September 7-10, 2008. Proceedings, pp. 205-220. Springer US, Boston, MA. (2008)

[67] Daly, C.*: Patterns for Cross-Domain Information Sharing*, IBM White Paper, (2006)

[68] De Caro, A., Iovino, V., Jain, A., O'Neill, A., Paneth, O., Persiano, G.*: On the achievability of simulation-based security for functional encryption*. In: Canetti, R. and Garay, J.A. (eds.) 33rd Annual Cryptology Conference - CRYPTO 2013, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, pp. 519-535. Springer Berlin Heidelberg (2013)

[69] Delerablee, C.*: Identity-based Broadcast Encryption with Constant Size Ciphertexts and Private Keys*. In: Kurosawa, K. (ed.) 13th International Conference on the Theory and Application of Cryptology and Information Security - ASIACRYPT 2007, Kuching, Malaysia, December 2-6, 2007. Proceedings, pp. 200-215. Springer-Verlag, Berlin, Heidelberg. (2007)

[70] Digital Watermarking Alliance: *The Case for Content Identification: Supporting New Business Models and Effectively Managing Key Business Assets*. Digital Watermarking Alliance white papers, 9p., *http://www.digitalwatermarkingalliance.org/docs/papers/DWA_WhitePaper_Case4ContentID.pdf*, Accessed: 2016/Jan/23, (2012)

[71] Dodis, Y., Fazio, N.*: Public Key Broadcast Encryption for Stateless Receivers*. In: Feigenbaum, J. (ed.) 9th ACM CCS-9 Workshop on Digital Rights Management - ACM DRM 2002, Chicago, IL, USA, November 9, 2002. Proceedings, pp. 61-80. Springer, Berlin, Heidelberg. (2003)

[72] Dodis, Y., Katz, J.*: Chosen-Ciphertext Security of Multiple Encryption*. In: Kilian, J. (ed.) 2nd Annual Theory of Cryptography Conference - TCC 2005, Cambridge, MA, USA, February 10-12, 2005. Proceedings, pp. 188-209. Springer, Berlin, Heidelberg. (2005)

[73] Dong, C., Chen, L., Camenisch, J., Russello, G.*: Fair Private Set Intersection with a Semi-trusted Arbiter*. In: Wang, L. and Shafiq, B. (eds.) 27th Annual IFIP WG 11.3 Conference on Data and Applications Security and Privacy, DBSec 2013, Newark, NJ, USA, July 15-17, 2013. Proceedings, vol. 7964, pp. 128-144. Springer Berlin Heidelberg (2013)

[74] Escala, A., Herranz, J., Morillo, P.*: Revocable Attribute-Based Signatures with Adaptive Security in the Standard Model*. In: Nitaj, A. and Pointcheval, D. (eds.) Progress in Cryptology -- AFRICACRYPT 2011: 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings, pp. 224-241. Springer, Berlin, Heidelberg. (2011)

[75] ESSI WSML Working Group: *Web Service Modelling Language (WSML).* WSMO publications, *http://www.wsmo.org/wsml/wsml-syntax#%E2%80%9D,* Accessed: 2016/Feb/6, (2008)

[76] Farrell, S., Housley, R.: *RFC 3281: An Internet Architecture Certificate Profile for Authorization.* IETF Request for Comments, 39p., *https://www.ietf.org/rfc/rfc3281.txt,* Accessed: 2016/Feb/2, (2002)

[77] Ferrara, A. L., Fuchsbauer, G., Warinschi, B.*: Cryptographically Enforced RBAC*. In: Garg, D. (ed.) 26th IEEE Computer Security Foundations Symposium - IEEE CSF 2013, New Orleans, LA, USA, June 26-28, 2013. Proceedings, pp. 115-129. IEEE Computer Society, Washington, DC, USA. (2013)

[78] Ford, W., Hallam-Baker, P., Fox, B., Dillaway, B., LaMacchia, B., Epstein, J., Lapp, J.: *XML Key Management Specification (XKMS).* W3C Recommendations, *http://www.w3.org/TR/xkms/,* Accessed: 2016/Feb/6, (2001)

[79] Freeman, D. M.*: Converting Pairing-based Cryptosystems from Composite-order Groups to Prime-order Groups*. In: Gilbert, H. (ed.) 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques - EUROCRYPT 2010, French Riviera, May 30 – June 3, 2010. Proceedings, pp. 44-61. Springer-Verlag, Berlin, Heidelberg. (2010)

[80] Fu, V., Ferraiolo, D., Kuhn, D. R., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K.*: Guide to attribute based access control (ABAC) definition and considerations, NIST SP 800-162*. National Institute of Standards and Technology (NIST) Special Publication. NIST, http://dx.doi.org/10.6028/NIST.SP.800-162. (2014)

[81] Fujisaki, E., Okamoto, T.*: Secure Integration of Asymmetric and Symmetric Encryption Schemes*. In: Wiener, M. (ed.) 19th Annual International Cryptology Conference - CRYPTO 1999 Santa Barbara, California, USA, August 15–19, 1999. Proceedings, pp. 537-554. Springer-Verlag, London, UK. (1999)

[82] Garg, S., Gentry, C., Halevi, S.*: Candidate Multilinear Maps from Ideal Lattices*. In: Johansson, T. and Nguyen, P.Q. (eds.) Advances in Cryptology - EUROCRYPT 2013: 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings, pp. 1-17. Springer, Berlin, Heidelberg. (2013)

[83] Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.*: Attribute-Based Encryption for Circuits from Multilinear Maps*. In: Canetti, R. and Garay, J.A. (eds.) Advances in Cryptology - CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II, pp. 479-499. Springer, Berlin, Heidelberg. (2013)

[84] Gennaro, R., Gentry, C., Parno, B.*: Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers*. In: Rabin, T. (ed.) Advances in Cryptology - CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings, pp. 465-482 (2010)

[85] Gentry, C.*: Certificate-based Encryption and the Certificate Revocation Problem*. In: Biham, E. (ed.) 22nd International Conference on the Theory and Applications of Cryptographic Techniques - EUROCRYPT 2003, Warsaw, Poland, May 4–8, 2003. Proceedings, pp. 272-293. Springer-Verlag, Berlin, Heidelberg. (2003)

[86] Gentry, C., Gorbunov, S., Halevi, S.*: Graph-Induced Multilinear Maps from Lattices*. In: Dodis, Y. and Nielsen, J.B. (eds.) 12th Theory of Cryptography Conference - TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II, pp. 498-527. Springer, Berlin, Heidelberg. (2015)

[87] Gentry, C., Waters, B.*: Adaptive Security in Broadcast Encryption Systems (with Short Ciphertexts)*. In: Joux, A. (ed.) Advances in Cryptology - EUROCRYPT 2009: 28th Annual International Conference on the Theory

and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings, pp. 171-188. Springer, Berlin, Heidelberg. (2009)

[88] Girault, M.*: Self-certified Public Keys*. In: Davies, D.W. (ed.) Workshop on the Theory and Application of Cryptographic Techniques - EUROCRYPT 1991, Brighton, UK, April 8–11, 1991. Proceedings, pp. 490-497. Springer-Verlag, Berlin, Heidelberg. (1991)

[89] Goguen, J. A., Meseguer, J.*: Security Policies and Security Models*. In: Neumann, P.G. and Morris, R. (eds.) IEEE Symposium on Security and Privacy - IEEE S&P 1982, Berkeley, CA, USA, May 20-23, 1982. Proceedings, pp. 11-20 (1982)

[90] Goldreich, O.*: Foundations of cryptography: Volume 2, basic applications*. Cambridge University Press, New York, NY, USA. (2004)

[91] Goldreich, O., Micali, S., Wigderson, A.*: Proofs that yield nothing but their validity and a methodology of cryptographic protocol design*. In: Hopcroft, J. (ed.) 27th Annual Symposium on Foundations of Computer Science - FOCS 1986, Toronto, Canada, October 27-29 1986. Proceedings, pp. 174-187. IEEE (1986)

[92] Goldwasser, S., Micali, S.: Probabilistic Encryption. *Journal of Computer and System Sciences*, **28**(2): 270-299, Elsevier. (1984)

[93] Goldwasser, S., Micali, S., Rackoff, C.*: The knowledge complexity of interactive proof systems*. In: 26th Annual Symposium on Foundations of Computer Science - FOCS 1985, Portland, Oregon, USA, October 21-23 1985. Proceedings, pp. 291-304. IEEE (1985)

[94] Gorbunov, S., Vaikuntanathan, V., Wee, H.*: Attribute-based Encryption for Circuits*. In: Boneh, D. and Roughgarden, T. (eds.) 45th Annual ACM Symposium on Theory of Computing - STOC 2013, Palo Alto, CA, USA, June 01 - 04, 2013. Proceedings, pp. 545-554. ACM, New York, NY, USA. (2013)

[95] Gorbunov, S., Vaikuntanathan, V., Wee, H.*: Functional Encryption with Bounded Collusions via Multi-party Computation*. In: Safavi-Naini, R. and Canetti, R. (eds.) Advances in Cryptology - CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings, vol. 7417, pp. 162-179. Springer Berlin Heidelberg (2012)

[96] Goyal, V., Jain, A., Pandey, O., Sahai, A.*: Bounded ciphertext policy attribute based encryption*. In: Aceto, L., Damgård, I., Goldberg, L.A., et al (eds.) 35th International Colloquium on Automata, Languages and Programming - ICALP 2008, Reykjavik, Iceland, July 7-11, 2008. Proceedings, pp. 579-591. Springer Berlin Heidelberg (2008)

[97] Goyal, V., Pandey, O., Sahai, A., Waters, B.*: Attribute-based encryption for fine-grained access control of encrypted data*. In: Juels, A., Wright, R.N. and De Capitani di Vimercati, Sabrina (eds.) 13th ACM Conference on Computer and Communications Security - CCS 2006, Alexandria, VA, USA, October 30 - November 3, 2006. Proceedings, pp. 89-98. ACM (2006)

[98] Graham-Rowe, D.: *Sony Sues Over PS3 Encryption Hack.* New Scientist news, *https://www.newscientist.com/article/dn19973-sony-sues-over-ps3-encryption-hack/*, Accessed: 2016/Jan/23, (2011)

[99] Groth, J.*: Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures*. In: Lai, X. and Chen, K. (eds.) Advances in Cryptology - ASIACRYPT 2006: 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, December 3-7, 2006. Proceedings, pp. 444-459. Springer, Berlin, Heidelberg. (2006)

[100] Groth, J., Sahai, A.*: Efficient Non-interactive Proof Systems for Bilinear Groups*. In: Smart, N. (ed.) Advances in Cryptology - EUROCRYPT 2008: 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings, pp. 415-432. Springer, Berlin, Heidelberg. (2008)

[101] Haber, S., Stornetta, W., Scott: How to Time-Stamp a Digital Document. *Journal of Cryptology*, **2**(3): 99-111, Springer US. (1991)

[102] Hanaoka, Y., Hanaoka, G., Shikata, J., Imai, H.*: Identity-Based Hierarchical Strongly Key-insulated Encryption and Its Application*. In: Bimal, R. (ed.) 11th International Conference on the Theory and Application of Cryptology and Information Security - ASIACRYPT 2005, Chennai, India, December 4-8, 2005., pp. 495-514. Springer-Verlag, Berlin, Heidelberg. (2005)

[103] Harn, L., Lin, H.: Refereed Article: A Cryptographic Key Generation Scheme for Multilevel Data Security. *Journal of Computers and Security*, **9**(6): 539-546, Elsevier Advanced Technology Publications. (1990)

[104] Harrington, A., Jensen, C.*: Cryptographic Access Control in a Distributed File System*. In: Ferrari, E. and Ferraiolo, D. (eds.) 8th ACM Symposium on Access Control Models and Technologies - SACMAT 2003, Como, Italy, June 2-3, 2003. Proceedings, pp. 158-165. ACM, New York, NY, USA. (2003)

[105] Harrison, M. A., Ruzzo, W. L., Ullman, J. D.: Protection in Operating Systems. *Communication of the ACM*, **19**(8): 461-471, ACM. (1976)

[106] Hasan, E. A.*: Efficient arithmetic for the implementation of elliptic curve cryptography.* Electronic Thesis and Dissertation Repository153. The University of Western Ontario, London, Ontario, Canada. (2013)

[107] Hiner, J.: *Is Perimeter Security Dead and is Protecting the Data all that Matters?* TechRepublic, Tech Sanity Check, 3p., *http://www.techrepublic.com/blog/tech-sanity-check/is-perimeter-security-dead-and-is-protecting-the-data-all-that-matters/,* Accessed: 2016/Jan/12, (2007)

[108] Hiray, N., Shedge, K.: A Review on Confidentiality and Authentication in Content Based Publish/Subcribe System. *International Journal of Advanced Research in Computer and Communication Engineering*, **4**(12): 179-181, IJARCCE. (2015)

[109] Imamura, T., Dillaway, B., Simon, E.: *XML Encryption Syntax and Processing.* W3C Recommendations, *http://www.w3.org/TR/xmlenc-core/,* Accessed: 2016/Feb/6, (2002)

[110] Ion, M., Russello, G., Crispo, B.*: Supporting Publication and Subscription Confidentiality in Pub/Sub Networks*. In: Jajodia, S. and Zhou, J. (eds.) 6th International ICST Conference on Security and Privacy in Communication Networks - SecureComm 2010, Singapore, September 7-9, 2010. Proceedings, pp. 272-289. Springer, Berlin, Heidelberg. (2010)

[111] Jajodia, S., Samarati, P., Sapino, M. L., Subrahmanian, V. S.: Flexible Support for Multiple Access Control Policies. *ACM Transactions on Database Systems*, **26**(2): 214-260, ACM. (2001)

[112] Jin, X.*: Attribute-based access control models and implementation in cloud infrastructure as a service.* The University of Texas at San Antonio Dissertations160. The University of Texas at San Antonio, Texas, San Antonio. (2014)

[113] Joux, A.*: The Weil and Tate Pairings As Building Blocks for Public Key Cryptosystems*. In: Fieker, K. and Kohel, D.R. (eds.) 5th International Symposium on Algorithmic Number Theory - ANTS 2002, Sydney, Australia, July 7-12, 2002. Proceedings, pp. 20-32. Springer-Verlag, London, UK. (2002)

[114] JUHTA - Julkisen hallinnon neuvottelukunta: *JHS 191 Tiedonohjaussuunnitelman Rakenne (in Finnish) - the Structure of Information Metadata Plan.* JHS Recommendations, 17 pages, (2015)

[115] Khader, D.: *Attribute Based Group Signatures.* Cryptology ePrint Archive, Report 2007/159, (2007)

[116] Kim, J., Susilo, W., Au, M. H., Seberry, J.*: Efficient Semi-static Secure Broadcast Encryption Scheme*. In: Cao, Z. and Zhang, F. (eds.) 6th International Conference on Pairing-Based Cryptography - Pairing 2013, Beijing, China, November 22-24, 2013. Revised Selected Papers, pp. 62-76. Springer International Publishing, Cham. (2014)

[117] Kiviharju, M.*: Attribute Pooling for Cryptographic Access Control: Enabling Cryptographical User-Terminal Policies for MLS-Content*. In: Amanowicz, M. (ed.) International Conference on Military Communications and Information Systems - ICMCIS 2015, Krakow, Poland, May 18-19, 2015. Proceedings, pp. 1-12. IEEE (2015)

[118] Kiviharju, M.*: RBAC with ABS: Implementation Practicalities for RBAC Integrity Policies*. In: Obaidat, M.S., Holzinger, A. and Samarati, P. (eds.) International Conference on Security and Cryptography - SECRYPT 2014, Vienna, Austria, August 28-30, 2014. Proceedings, pp. 500-509. INSTICC (2014)

[119] Kiviharju, M.*: Cryptographic Roles in the Age of Wikileaks: Implementation Models for Cryptographically Enforced RBAC.* In: O'Conner, L. (ed.) Military Communications Conference – MILCOM 2013, San Diego, CA, USA, November 18-20, 2013, Proceedings, pp. 1779-1788. IEEE (2013)

[120] Kiviharju, M.*: On Multi-Level Structured Content: A cryptographic key management-independent XML schema for MLS content*. In: Amanowicz, M. (ed.) Military Communications and Information Systems Conference - MCC 2012, Gdansk, Poland, October 8-9, 2012. Proceedings, pp. 1-8. IEEE (2012)

[121] Kiviharju, M.*: Towards Pervasive Cryptographic Access Control Models*. In: Samarati, P., Lou, W. and Zhou, J. (eds.) International Confer-

ence on Security and Cryptography - SECRYPT 2012, Rome, Italy, July 24-27, 2012, Proceedings, pp. 239-244. INSTICC (2012)

[122] Kiviharju, M.*: Content-based information security (CBIS) : Definitions, requirements and cryptographic architecture.* Puolustusvoimien Teknillinen Tutkimuslaitos - Defence Forces Technical Research Centre, Julkaisusarja - Series, Vol. 21. 1st edn. Puolustusvoimien Teknillinen Tutkimuslaitos - Defence Forces Technical Research Centre, Riihimaki. (2010)

[123] Kiviharju, M.*: Fuzzy Pairings-Based CL-PKC*. In: Chaumine, J. and Rolland, R. (eds.) 1st Symposium on Algebraic Geometry and its Applications - SAGA 2007, Papeete, France. Series on Number Theory and Its Applications: Algebraic Geometry and Its Applications. Proceedings, vol. 5, pp. 168-187. World Scientific Publishing Company (2008)

[124] Kiviharju, M., Kurnikov, A.*: Tactical CAC profile for NATO OLP? Performance estimations for NATO OLP cryptographic evolution stage*. In: Brand, J., Valenti, M., Akinpelu, A., et al (eds.) Military Communications Conference – MILCOM 2016, Baltimore, MD, USA, November 1-3, 2016, Proceedings, pp. 533-538. IEEE (2016)

[125] Kong, Y., Seberry, J., Getta, J. R., Yu, P.*: A Cryptographic Solution for General Access Control*. In: Kong, Y., Seberry, J., Getta, J.R., et al (eds.) 8th International Conference on Information Security - ISC 2005, Singapore, September 20-23, 2005. Proceedings, pp. 461-473. Springer-Verlag, Berlin, Heidelberg. (2005)

[126] Kuhn, D. R.*: Role Based Access Control on MLS Systems Without Kernel Changes*. In: Youman, C. and Jaeger, T. (eds.) 3rd ACM Workshop on Role-based Access Control - RBAC 1998, Fairfax, VA, USA, October 22-23, 1998. Proceedings, pp. 25-32. ACM, New York, NY, USA. (1998)

[127] Kumar, K. P., Shailaja, G., Saxena, A.: *Secure and Efficient Threshold Key Issuing Protocol for ID-Based Cryptosystems.* Cryptology ePrint Archive, Report 2006/245, (2006)

[128] Küsters, R., Datta, A., Mitchell, J., Ramanathan, A.: On the Relationships between Notions of Simulation-Based Security. *J. Cryptol.*, **21**(4): 492-546, Springer-Verlag. (2008)

[129] Lampson, B. W.: Protection. *SIGOPS Operational System Review Letters*, **8**(1): 18-24, ACM. (1974)

[130] Laur, S.*: Cryptographic protocol design*. TKK Dissertations in Information and Computer Science. TKK, Espoo. (2008)

[131] Lee, B., Kim, K.*: Self-certified Signatures*. In: Menezes, A. and Sarkar, P. (eds.) Progress in Cryptology --- INDOCRYPT 2002: 3rd International Conference on Cryptology in India Hyderabad, India, December 16-18, 2002. Proceedings, pp. 199-214. Springer, Berlin, Heidelberg. (2002)

[132] Lewko, A., Waters, B.*: New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques*. In: Safavi-Naini, R. and Canetti, R. (eds.) Advances in Cryptology -- CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings, pp. 180-198. Springer, Berlin, Heidelberg. (2012)

[133] Lewko, A., Waters, B.*: Decentralizing attribute-based encryption.* In: Paterson, K.G. (ed.) 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques - EUROCRYPT 2011, Tallinn, Estonia, May 15–19, 2011. Proceedings, pp. 568-588. Springer Berlin Heidelberg (2011)

[134] Leyden, J.: *Blu-Ray DRM Defeated.* The Register news, *http://www.theregister.co.uk/2007/01/23/blu-ray_drm_cracked/*, Accessed: 2016/Jan/25, (2007)

[135] Liu, J. K., Au, M. H., Susilo, W.*: Self-Generated-Certificate Public Key Cryptography and Certificateless Signature/Encryption Scheme in the Standard Model: Extended Abstract.* In: Bao, F. and Miller, S. (eds.) 2nd ACM Symposium on Information, Computer and Communications Security - ASIACCS 2007, Singapore, March 20-22, 2007. Proceedings, pp. 273-283. ACM, New York, NY, USA. (2007)

[136] Loscocco, P., Smalley, S.*: Integrating Flexible Support for Security Policies into the Linux Operating System*, NSA Technical Report, (2001)

[137] Lynn, B.: *PBC Library: The Pairing-Based Cryptography Library.* Stanford University Applied Cryptography Group, *https://crypto.stanford.edu/pbc/*, Accessed: 2016/April/25, (2013)

[138] MacKinnon, S. J., Taylor, P. D., Meijer, H., Akl, S. G.: An Optimal Algorithm for Assigning Cryptographic Keys to Control Access in a Hierarchy. *IEEE Transactions on Computing.*, **34**(9): 797-802, IEEE Computer Society. (1985)

[139] Maji, H. K., Prabhakaran, M., Rosulek, M.*: Attribute-based Signatures*. In: Kiayias, A. (ed.) The Cryptographers' Track at the RSA Conference - CT-RSA 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings, pp. 376-392. Springer-Verlag, Berlin, Heidelberg. (2011)

[140] Maji, H., Prabhakaran, M., Rosulek, M.: *Attribute-Based Signatures: Achieving Attribute-Privacy and Collusion-Resistance.* Cryptology ePrint Archive, Report 2008/328, (2008)

[141] Matt, C., Maurer, U.*: A Definitional Framework for Functional Encryption*. In: Fournet, C., Hicks, M.W. and Vigano, L. (eds.) 28th IEEE Computer Security Foundations Symposium - IEEE CSF 2015, Verona, Italy, July 13-17, 2015. Proceedings, pp. 217-231 (2015)

[142] Maurer, U.*: Constructive cryptography -- A new paradigm for security definitions and proofs*. In: S. Moedersheim and C. Palamidessi (eds.) Theory of Security and Applications (TOSCA 2011), vol. 6993, pp. 33-56. Springer-Verlag (2011)

[143] Maurer, U., Renner, R.*: Abstract Cryptography*. In: Bernard Chazelle (ed.) The Second Symposium on Innovations in Computer Science, ICS 2011, pp. 1-21. Tsinghua University Press (2011)

[144] Maurer, U., Renner, R., Holenstein, C.*: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology*. In: Moni Naor (ed.) 1st Theory of Cryptography Conference - TCC 2004, Cambridge, MA, USA, February 19-21, 2004. Proceedings, vol. 2951, pp. 21-39. Springer-Verlag (2004)

[145] Maurer, U., Rüedlinger, A., Tackmann, B.*: Confidentiality and Integrity: A Constructive Perspective*. In: Ronald Cramer (ed.) Theory of Cryptography --- TCC 2012, vol. 7194, pp. 209-229. IACR; Springer (2012)

[146] McClure, A.: Hexagon: A US Joint Force Command Solution to Coalition Interoperability. *The EDGE, (July 2001)*, **5**(2)MITRE. (2001)

[147] McGovern, S.*: Information security requirements for a coalition wide area network.* Thesis in Naval Postgraduate School, Monterey, California1-96 (2001)

[148] McKerrow, G.: *Multilevel Security Networks: An Explanation of the Problem.* SANS Information Security Reading Room, 16p., *https://www.sans.org/reading-room/whitepapers/standards/multilevel-security-networks-explanation-problem-546*, Accessed: 2016/Jan/12, (2001)

[149] Microsoft: *How to Create and Manage the Central Store for Group Policy Administrative Templates in Windows.* Microsoft Technet Library, *https://support.microsoft.com/en-us/kb/3087759,* Accessed: 2015/Feb/8, (2015)

[150] Microsoft: *Permission Entry Dialog Box.* Microsoft Technet Library, *https://technet.microsoft.com/fi-fi/library/cc753992.aspx,* Accessed: 2016/Feb/7, (2015)

[151] Nakamoto, S.: *Bitcoin: A Peer-to-Peer Electronic Cash System.* Bitcoin.org Documentation, 9p., *https://bitcoin.org/bitcoin.pdf,* Accessed: 2016/Dec/13, (2008)

[152] Naor, D., Naor, M., Lotspiech, J. B.*: Revocation and Tracing Schemes for Stateless Receivers*. In: Kilian, J. (ed.) 21st Annual Cryptology Conference - CRYPTO 2001, Santa Barbara, CA, USA, August 19-23, 2001. Proceedings, pp. 41-62. Springer-Verlag, London, UK. (2001)

[153] Nielsen, J. B.*: Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case*. In: Young, M. (ed.) 22nd Annual Cryptology Conference - CRYPTO 2002, Santa Barbara, CA, USA, August 18-22, 2002. Proceedings, pp. 111-126. Springer (2002)

[154] OASIS: *SAML 2.0 Profile of XACML, Version 2.0, Committee Draft 1.* OASIS Standards, *https://docs.oasis-open.org/xacml/3.0/xacml-profile-saml2.0-v2-spec-cd-1-en.html,* Accessed: 2016/Jan/18, (2009)

[155] OASIS: *Security Assertion Markup Language (SAML) V2.0 Technical Overview, Committee Draft 02.* 51p., *http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html,* Accessed: 2016/Jan/18, (2008)

[156] OASIS: *OASIS Web Services Security: SOAP Message Security 1.1 (WS-Security 2004).* OASIS Standards, 76p., *https://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf,* Accessed: 2016/Feb/4, (2006)

[157] Oh, S., Park, S.: Task–role-Based Access Control Model. *Journal of Information Systems,* **28**(6): 533-562, Elsevier. (2003)

[158] Okamoto, T., Takashima, K.*: Decentralized Attribute-Based Signatures.* In: Kurosawa, K. and Hanaoka, G. (eds.) Public-Key Cryptography - PKC 2013: 16th International Conference on Practice and Theory in

Public-Key Cryptography, Nara, Japan, February 26 - March 1, 2013. Proceedings, pp. 125-142. Springer (2013)

[159] Okamoto, T., Takashima, K.*: Efficient Attribute-Based Signatures for Non-monotone Predicates in the Standard Model*. In: Catalano, D., Fazio, N., Gennaro, R., et al (eds.) Public Key Cryptography - PKC 2011: 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings, pp. 35-52. Springer (2011)

[160] Okamoto, T., Takashima, K.*: Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption*. In: Rabin, T. (ed.) Advances in Cryptology - CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings, pp. 191-208 (2010)

[161] O'Neill, A.: *Definitional Issues in Functional Encryption.* Cryptology ePrint Archive, Report 2010/556, 11 pages, (2010)

[162] Ostrovsky, R., Sahai, A., Waters, B.*: Attribute-based encryption with non-monotonic access structures*. In: Ning, P., De Capitani di Vimercati, Sabrina and Syverson, P.F. (eds.) 14th ACM Conference on Computer and Communications Security - CCS 2007, Alexandria, VA, USA, October 28-31, 2007. Proceedings, pp. 195-203. ACM (2007)

[163] Oudkerk, S., Wrona, K.*: Cryptographic access control in support of Object Level Protection*. In: Guivarch, P. (ed.) Military Communications and Information Systems Conference - MCC 2013, Saint-Malo, France, October 7-9, 2013. Proceedings, pp. 1-10. IEEE (2013)

[164] Oudkerk, S., Wrona, K.*: A Common Approach to the Integration of Object Level Protection in NATO*. In: McCallam, D. and Luiijf, E. (eds.) NATO STO/IST-122 Symposium on Cyber Security Science and Engineering, Tallinn, Estonia, October 13-14, 2014. Proceedings, vol. 1. NATO STO (2014)

[165] Page, D., Smart, N. P., Vercauteren, F.: A Comparison of MNT Curves and Supersingular Curves. *Applicable Algebra in Engineering, Communication and Computing,* **17**(5): 379-392, Springer. (2006)

[166] Papadimitriou, C.*: Section 15.1 - Parallel algorithms*. In: Papadimitriou, C. (ed.) Computational Complexity, pp. 359-368. Addison-Wesley (1994)

[167] Papadimitriou, C.*: Section 4 - Boolean logic*. In: Papadimitriou, C. (ed.) Computational Complexity, pp. 73-86. Addison-Wesley (1994)

[168] Papadimitriou, C.*: Section 5: First-order logic*. In: Papadimitriou, C. (ed.) Computational Complexity, pp. 87-122. Addison-Wesley, USA. (1994)

[169] Papadimitriou, C.*: Sections 15.3-4: The class NC; RNC algorithms*. In: Papadimitriou, C. (ed.) Computational Complexity, pp. 375-384. Addison-Wesley, USA. (1994)

[170] Park, J., Sandhu, R.: The UCONABC Usage Control Model. *ACM Transactions on Information Systems Security*, **7**(1): 128-174, ACM. (2004)

[171] Parno, B., Howell, J., Gentry, C., Raykova, M.*: Pinocchio: Nearly Practical Verifiable Computation*. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy, pp. 238-252. IEEE Computer Society, Washington, DC, USA. (2013)

[172] Parno, B., Raykova, M., Vaikuntanathan, V.*: How to Delegate and Verify in Public: Verifiable Computation from Attribute-based Encryption*. In: Proceedings of the 9th International Conference on Theory of Cryptography, pp. 422-439. Springer-Verlag, Berlin, Heidelberg. (2012)

[173] Pintsov, L. A., Vanstone, S. A.*: Postal Revenue Collection in the Digital Age*. In: Frankel, Y. (ed.) 4th International Conference on Financial Cryptography - FC 2000, Anguilla, British West Indies, February 20-24, 2000. Proceedings, pp. 105-120. Springer, Berlin, Heidelberg. (2001)

[174] Popa, R. A., Redfield, C. M. S., Zeldovich, N., Balakrishnan, H.*: CryptDB: Protecting Confidentiality with Encrypted Query Processing*. In: Wobber, T. and Druschel, P. (eds.) 23rd ACM Symposium on Operating Systems Principles - SOSP 2011, Cascais, Portugal, October 23-26, 2011. Proceedings, pp. 85-100. ACM, New York, NY, USA. (2011)

[175] Ray, I., Ray, I., Narasimhamurthi, N.*: A Cryptographic Solution to Implement Access Control in a Hierarchy and More*. In: Sandhu, R. and Bertino, E. (eds.) 7th ACM Symposium on Access Control Models and Technologies - SACMAT 2002, Monterey, CA, USA, June 3-4, 2002. Proceedings, pp. 65-73. ACM, New York, NY, USA. (2002)

[176] Ribeiro, C., Zúquete, A., Ferreira, P., Guedes, P.*: SPL: An access control language for security policies with complex constraints*. In: Kent, S. and Tsudik, G. (eds.) 3rd IETF Symposium on Network and Distributed

System Security - NDSS 1999, San Diego, CA, USA, February 3-5, 1999. Proceedings, pp. 89-107 (1999)

[177] Rissanen, E.: *Extensible Access Control Markup Language (XACML) Version 3.0.* OASIS Standards, 154p., *http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html,* Accessed: 1/(2013)

[178] Rivest, R. L., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, **21**(2): 120-126, ACM. (1978)

[179] Rivest, R. L., Shamir, A., Tauman, Y.*: How to leak a secret*. In: Kilian, J. (ed.) 21st Annual Cryptology Conference - CRYPTO 2001, Santa Barbara, CA, USA, August 19-23, 2001. Proceedings, pp. 554-567. Springer-Verlag (2001)

[180] Sahai, A., Waters, B.: *Slides on Functional Encryption.* Powerpoint presentation, 37p., *http://www.cs.utexas.edu/~bwaters/presentations/files/functional.ppt*, Accessed: 2015/04/01, (2008)

[181] Sahai, A., Seyalioglu, H., Waters, B.*: Dynamic Credentials and Ciphertext Delegation for Attribute-Based Encryption*. In: Safavi-Naini, R. and Canetti, R. (eds.) 32nd Annual Cryptology Conference - CRYPTO 2012, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings, vol. 7417, pp. 199-217. Springer Berlin Heidelberg (2012)

[182] Sahai, A., Seyalioglu, H., Waters, B.*: Dynamic credentials and ciphertext delegation for attribute-based encryption*. In: Safavi-Naini, R. and Canetti, R. (eds.) 32nd Annual Cryptology Conference - CRYPTO 2012, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings, pp. 199-217. Springer Berlin Heidelberg (2012)

[183] Sahai, A., Waters, B.*: Fuzzy identity-based encryption*. In: Cramer, R. (ed.) 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques - EUROCRYPT 2005, Aarhus, Denmark, May 22 – 26, 2005., pp. 457-473. Springer Berlin Heidelberg (2005)

[184] Sakai, R.,Furukawa, J.: *Identity-Based Broadcast Encryption.* Cryptology ePrint Archive, Report 2007/217, 14 pages, (2007)

[185] Sandhu, R. S.*: Role Hierarchies and Constraints for Lattice-Based Access Controls*. In: Bertino, E., Kurth, H., Giancarlo, M., et al (eds.) 4th European Symposium on Research in Computer Security - ESORICS

1996, Rome, Italy, September 25-27, 1996. Proceedings, pp. 65-79. Springer-Verlag, London, UK. (1996)

[186] Sandhu, R., Bhamidipati, V., Munawer, Q.: The ARBAC97 Model for Role-Based Administration of Roles. *ACM Transactions on Information Systems Security,* **2**(1): 105-135, ACM. (1999)

[187] Sandhu, R., Ferraiolo, D., Kuhn, R.*: The NIST Model for Role-based Access Control: Towards a Unified Standard*. In: Rebensburg, K., Youman, C. and Atruli, V. (eds.) 5th ACM Workshop on Role-based Access Control - RBAC 2000, Berlin, Germany, July 26-28, 2000. Proceedings, pp. 47-63. ACM, New York, NY, USA. (2000)

[188] Sandhu, R. S.: Cryptographic Implementation of a Tree Hierarchy for Access Control. *Journal of Information Processing Letters,* **27**(2): 95-98, Elsevier North-Holland, Inc. (1988)

[189] Sanov, I. N.: On the Probability of Large Deviations of Random Variables (English Translation). *Selected Translations in Mathematical Statistics and Probability I,* **1**: 213-244, (1961)

[190] Savoie, J.: *A Strong Three-Factor Authentication Device: TrustedDAVE and the New Generic Content-Based Information Security (CBIS) Architecture.* Technical Memorandum TM 2004-198, DRDC Canada, 46p., *http://pubs.drdc.gc.ca/PDFS/unc32/p522843.pdf,* Accessed: 2016/Feb/2, (2004)

[191] Schneier, B., Kelsey, J.*: Cryptographic Support for Secure Logs on Untrusted Machines*. In: Proceedings of the 7th Conference on USENIX Security Symposium - Volume 7, pp. 4-4. USENIX Association, Berkeley, CA, USA. (1998)

[192] Scott, M.*: On the Efficient Implementation of Pairing-Based Protocols.* In: Chen, L. (ed.) 13th IMA International Conference on Cryptography and Coding - IMACC 2011, Oxford, UK, December 12-15, 2011. Proceedings, pp. 296-308. Springer, Berlin, Heidelberg. (2011)

[193] Shahandashti, S. F., Safavi-Naini, R.*: Threshold Attribute-Based Signatures and Their Application to Anonymous Credential Systems*. In: Preneel, B. (ed.) 2nd International Conference on Cryptology in Africa - AFRICACRYPT 2009, Gammarth, Tunisia, June 21-25, 2009. Proceedings, vol. 5580, pp. 198-216. Springer Berlin Heidelberg (2009)

[194] Shannon, C. E.: Communication Theory of Secrecy Systems. *Bell Systems Technical Journal,* **28**(4): 656-715, (1949)

[195] Shao, J., Cao, Z.*: CCA-Secure Proxy Re-encryption without Pairings*. In: Jarecki, S. and Tsudik, G. (eds.) 12th International Conference on Practice and Theory in Public Key Cryptography - PKC 2009, Irvine, CA, USA, March 18-20, 2009. Proceedings, vol. 5443, pp. 357-376. Springer (2009)

[196] Shoup, V.*: Lower Bounds for Discrete Logarithms and Related Problems*. In: Fumy, W. (ed.) 11th International Conference on the Theory and Application of Cryptographic Techniques - EUROCRYPT 1997, Konstanz, Germany, May 11–15, 1997. Proceedings, pp. 256-266. Springer Berlin Heidelberg, Berlin, Heidelberg. (1997)

[197] Sovarel, A. N., Evans, D., Paul, N.*: Where's the FEEB? The Effectiveness of Instruction Set Randomization*. In: McDaniel, P. (ed.) 14th Symposium on USENIX Security - SSYM 2005, Baltimore, MD, USA, July 31-August 5, 2005. Proceedings., vol. 14, pp. 1-10. USENIX Association, Berkeley, CA, USA. (2005)

[198] Szabo, N.: *A Formal Language for Analyzing Contracts. http://szabo.best.vwh.net/contractlanguage.html,* Accessed: 2016/Dec/13, (2002)

[199] The 4C Entity: *CPRM Specification, Introduction and Common Cryptographic Elements, Revision 1.1.* 4C Entity CPRM/CPPM/C2 Specifications, 17p., *http://www.4centity.com/specification.aspx,* Accessed: 2016/Jan/25, (2010)

[200] The 4C Entity: *CPPM Specification, Introduction and Common Cryptographic Elements, Rev. 1.0.* 4C Entity CPRM/CPPM/C2 Specifications, *http://www.4centity.com/specification.aspx,* Accessed: 2016/Jan/25, (2003)

[201] US Department of Defense*: DoD 5200.28-STD: Department of defense standard - department of defense trusted computer system evaluation criteria*. "Rainbow Series" (NCSC TG). National Computer Security Center, Fort Meade. (1985)

[202] US NSA: *Configuring the SELinux Policy: Policy Language and Example Policy Configuration.* SELinux Documentation, *https://www.nsa.gov/research/_files/selinux/papers/policy2/x109.shtml,* Accessed: 2016/Feb/10, (2011)

[203] Wallner, D., Harder, E., Agee, R.: *Key Management for Multicast: Issues and Architectures (RFC 2627).* IETF Request for Comments, 23p., *https://www.ietf.org/rfc/rfc2627.txt,* Accessed: 2016/Jan/13, (1999)

[204] Wang, G., Liu, Q., Wu, J., Guo, M.: Hierarchical Attribute-Based Encryption and Scalable User Revocation for Sharing Data in Cloud Servers. *Journal of Computer Security*, **30**(5): 320-331, Elsevier Advanced Technology Publications. (2011)

[205] Waters, B.*: Functional Encryption for Regular Languages*. In: Safavi-Naini, R. and Canetti, R. (eds.) 32nd Annual Cryptology Conference - CRYPTO 2012, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings, vol. 7417, pp. 218-235. Springer (2012)

[206] Waters, B.*: Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization*. In: Catalano, D., Fazio, N., Gennaro, R., et al (eds.) Public Key Cryptography PKC 2011, vol. 6571, pp. 53-70. Springer Berlin Heidelberg (2011)

[207] Wong, C. K., Gouda, M., Lam, S. S.: Secure Group Communications using Key Graphs. *IEEE/ACM Transactions on Networking*, **8**(1): 16-30, IEEE Press. (2000)

[208] World Wide Web Consortium: *Extensible Markup Language (XML).* W3C Recommendations, *http://www.w3.org/XML/Overview.html*, Accessed: 2016/Feb/6, (2015)

[209] World Wide Web Consortium: *XML Schema*. W3C Recommendations, *http://www.w3.org/XML/Schema*, Accessed: 2016/Feb/6, (2015)

[210] World Wide Web Consortium: *SOAP Version 1.2 Part 0: Primer (Second Edition).* W3C Recommendation, *http://www.w3.org/TR/2007/REC-soap12-part0-20070427/*, Accessed: 2016/Jan/18, (2007)

[211] World Wide Web Consortium: *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition).* W3C Recommendations, *http://www.w3.org/TR/2007/REC-soap12-part1-20070427/*, Accessed: 2016/Jan/18, (2007)

[212] World Wide Web Consortium: *SOAP Version 1.2 Part 2: Adjuncts (Second Edition).* W3C Recommendations, *http://www.w3.org/TR/2007/REC-soap12-part2-20070427/*, Accessed: 2016/Jan/18, (2007)

[213] World Wide Web Consortium: *SOAP Version 1.2 Specification Assertions and Test Collection (Second Edition).* W3C Recommendations, *http://www.w3.org/TR/2007/REC-soap12-testcollection-20070427/*, Accessed: 2016/Jan/18, (2007)

[214] Wrona, K., de Haan, R., de Jonge, A.*: Implementation Aspects of Attribute-Based Encryption and Program Obfuscation*, NCIA Technical Report, 2014/FIN009930/01, 1-106p. (2015)

[215] Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.*: A Framework and Compact Constructions for Non-monotonic Attribute-Based Encryption*. In: Krawczyk, H. (ed.) 17th International Conference on Practice and Theory in Public-Key Cryptography - PKC 2014, Buenos Aires, Argentina, March 26-28, 2014. Proceedings, pp. 275-292. Springer, Berlin, Heidelberg. (2014)

[216] Yuen, T. H., Susilo, W., Mu, Y.: Towards a Cryptographic Treatment of Publish/Subscribe Systems. *Journal of Computer Security*, **22**(1): 33-67, IOS Press. (2014)

[217] Zhang, G.: Attribute-Based Certificateless Cryptographic System. *Journal of Computers*, **9**(1): 72-77, Academy Publisher. (2014)

[218] Zheng, Y., Hardjono, T., Seberry, J.*: New Solutions to the Problem of Access Control in a Hierarchy*, University of Wollongong, Australia, reports, Department of Computer Science, University of Wollongong, Australia(1993)

[219] Zhou, L., Varadharajan, V., Hitchens, M.: Enforcing Role-Based Access Control for Secure Data Storage in the Cloud. *Computer Journal*, **54**(10): 1675-1687, Oxford University Press. (2011)

[220] Zhu, Y., Ahn, G. -., Hu, H., Ma, D., Wang, S.: Role-Based Cryptosystem: A New Cryptographic RBAC System Based on Role-Key Hierarchy. *Information Forensics and Security, IEEE Transactions on*, **8**(12): 2138-2153, (2013)

**Puolustusvoimien tutkimuslaitos**

**Ylöjärven toimipiste**
PL 5, 34111 Lakiala
- Esikunta
- Asetekniikkaosasto
- Räjähde- ja suojelutekniikkaosasto

**Riihimäen toimipiste**
PL 10, 11311 Riihimäki
- Doktriiniosasto
- Informaatiotekniikkaosasto
- Tutkimussuunnitteluyksikkö

**Tuusulan toimipiste**
PL 5, 04401 Järvenpää
- Toimintakykyosasto

Puh. 0299 800

puolustusvoimat.fi > Tietoa meistä > Tutkimuslaitos

ISBN 978-951-25-2883-7 (painettu)
ISBN 978-951-25-2884-4 (verkkojulkaisu)
ISSN 2432-3129 (painettu)
ISSN 2432-3137 (verkkojulkaisu)