

ABSTRACT

Title of dissertation: SECURE, POLICY-BASED, MULTI-RECIPIENT DATA SHARING

Rakesh Babu Bobba
Doctor of Philosophy, 2009

Dissertation directed by: Professor Virgil D. Gligor
Department of Electrical and
Computer Engineering

In distributed systems users often need to share sensitive data with other users based on the latter's ability to satisfy various policies. In many cases the data owner may not even know the identities of the data recipients, but deems it crucial that they are legitimate; i.e., satisfy the policy. Enabling such data sharing over the Internet faces the challenge of (1) securely associating access policies with data and enforcing them, and (2) protecting data as it traverses untrusted proxies and intermediate repositories. Furthermore, it is desirable to achieve properties such as: (1) flexibility of access policies; (2) privacy of sensitive access policies; (3) minimal reliance on trusted third parties; and (4) efficiency of access policy enforcement. Often schemes enabling controlled data sharing need to trade one property for another. In this dissertation, we propose two complimentary policy-based data sharing schemes that achieve different subsets of the above desired properties.

In the first part of this dissertation, we focus on CiphertextPolicy Attribute-Based Encryption (CP-ABE) schemes that specify and enforce access policies cryp-

tographically and eliminate trusted mediators. We motivate the need for flexible attribute organization within user keys for efficient support of many practical applications. We then propose Ciphertext-Policy Attribute-Set Based Encryption (CP-ASBE) which is the first CP-ABE scheme to (1) efficiently support naturally occurring compound attributes, (2) support multiple numerical assignments for a given attribute in a single key and (3) provide efficient key management. While the CP-ASBE scheme minimizes reliance on trusted mediators, it can support neither context-based policies nor policy privacy. In the second part of this dissertation, we propose Policy Based Encryption System (PBES), which employs mediated decryption and supports both context-based policies and policy privacy. Finally, we integrate the proposed schemes into practical applications (i.e., CP-ASBE scheme with Attribute-Based Messaging (ABM) and PBES scheme with a conditional data sharing application in the Power Grid) and demonstrate their usefulness in practice.

SECURE, POLICY-BASED, MULTI-RECIPIENT DATA
SHARING

by

Rakesh Babu Bobba

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2009

Advisory Committee:
Professor Virgil D. Gligor, Chair/Advisor
Professor William A. Arbaugh
Dr. Himanshu Khurana
Professor Gang Qu
Professor Charles B. Silio

© Copyright by
Rakesh Babu Bobba
2009

Dedication

To my parents, Hima Bindu and Narasimha Rao, and my wife Lakshmi.

Acknowledgments

I owe my gratitude to several people who have made this dissertation possible. First and foremost I would like to thank my advisor, Professor Virgil D. Gligor, for his guidance, support and encouragement over the years and for letting me work on this dissertation remotely. I owe my deepest gratitude to Dr. Himanshu Khurana who has been a mentor and guide since the early days of my graduate studies. This dissertation would not have been possible without his encouragement and guidance. I would like to thank my committee members Professor Charles B. Silio, Professor Gang Qu and Professor William A. Arbaugh for their valuable time and feedback.

I would like to thank Professor Carl A. Gunter and Professor Manoj M. Prabhakaran for their collaboration, support and advice. I would like to thank my co-authors Fariba Khan, Omid Fatemieh, Musab Alturki, Farhana Ashraf and Arindam Khan for their enthusiastic collaboration. I would like to thank house mates Karthikeyan Chandrasekhar, Manikandan Ramasamy, Akshay Naik, Kalyani Tikekar, Ragunath Sankaranarayanan and Jitamitra Karcherla, and fellow students Radostina Koleva, Laurent Eschenauer and Gelareh Taban for their friendship, help and support. Thanks are also due to fellow student Soo Bum Lee who helped me comply with many university regulations while I was away from the school.

I would also like to thank Dr. Tracy Chung, Maria Hoo and Melanie Prange at the ECE Graduate Studies Office who helped me navigate the many rules and regulations of the department and graduate school.

I am grateful to my parents Hima Bindu and Narasimha Rao, and my wife

Lakshmi for their patience, understanding and support throughout this long endeavor.

Finally I would like to acknowledge the financial support provided by the National Science Foundation under grant numbers CNS 05-24695 and CNS 07-16626, and by the Office of Naval Research under grant numbers N00014-04-1-0562, N00014-06-1-1108 and N00014-07-1-1173.

Table of Contents

List of Tables	vii
List of Figures	viii
List of Abbreviations	ix
1 Introduction	1
1.1 Policy-Based Data Sharing Without a Mediating Server	3
1.2 Data Sharing Based on Private, Context-Sensitive Policies	5
1.3 Applications	6
1.3.1 Attribute-Based Messaging	6
1.3.2 Conditional Sharing of Phasor Measurement Unit (PMU) Data in the Power Grid	6
1.4 Contributions	7
1.5 Dissertation Organization	9
2 Related Work	10
2.1 Policy-Based Data Sharing Without a Mediating Server	10
2.1.1 Secure Two-Party Data Sharing	10
2.1.2 Secure Multi-Party Data Sharing	11
2.2 Data Sharing Based on Private, Context-Sensitive Policies	14
3 Policy Based Data Sharing Without a Mediating Server	16
3.1 Need for Organizing Attributes in CP-ABE User Keys	20
3.1.1 Supporting Compound Attributes Efficiently	20
3.1.2 Supporting Multiple Value Assignments	24
3.2 Preliminaries	25
3.3 Our CP-ASBE Construction	31
3.4 Security	37
3.5 Efficiency Analysis	47
3.6 Implementation	48
4 Data Sharing Based on Private, Context Sensitive Policies	50
4.1 Approach	54
4.1.1 Related Approaches	54
4.1.2 Our Approach - PBES	57
4.2 Notation and Security Notions	59
4.3 Building Blocks	65
4.3.1 Key Encapsulation Mechanism	65
4.3.2 Data Encapsulation Mechanism	67
4.3.3 Policy and Key Encapsulation Mechanism	69
4.4 Policy Based Encryption System	78
4.5 Security Analysis	80
4.6 Application Design Issues	89

5	Application Integration and Evaluation	93
5.1	Attribute-Based Messaging	93
5.1.1	CP-ASBE for ABM	96
5.1.2	ABM Architecture	99
5.1.3	Experimental Evaluation of CP-ASBE in ABM	104
5.2	Context-Sensitive Data Sharing in the Power Grid	108
5.2.1	Requirements	112
5.2.2	PBES for Context-Sensitive Data Sharing in the Power Grid .	114
5.2.3	Prototype Implementation and Performance	119
6	Conclusion	124
	Bibliography	127

List of Tables

3.1	Possible adversary query terms in \mathbb{G}_1	44
5.1	Grammar for ABM Addresses and Rules	99
5.2	Example of Policy Elements	115
5.3	PBES Entities vs. Power Grid Entities	119

List of Figures

4.1	Policy-based Message Encryption and Decryption	57
4.2	Encryption in PKEM-DEM scheme instantiated using RSA-KEM and DEM1	60
5.1	ABM Architecture	100
5.2	Policy Specialization Path	101
5.3	Messaging and Address Resolution Path	102
5.4	Attribute Keying Path	104
5.5	Encryption and Decryption Times	107
5.6	Proposed NASPI PMU Architecture	111

List of Abbreviations

ABE	Attribute-Based Encryption
ABM	Attribute-Based Messaging
AK	Attribute Keying
AR	Address Resolution
BA	Balancing Authority
BAA	Balancing Authority Area
CA	Certificate Authority
CCA2	Adaptive Chosen Ciphertext Attack
CP-ABE	Ciphertext-Policy Attribute-Based Encryption
CP-ASBE	Ciphertext-Policy Attribute-Set-Based Encryption
DEM	Data Encapsulation Mechanism
FERC	Federal Energy Regulation Commission
ISO	Independent System Operator
KDC	Key Distribution Center
KEM	Key Encapsulation Mechanism
KP-ABE	Key-Policy Attribute-Based Encryption
MS	Messaging
MTA	Mail Transport Agent
NASPI	North American SynchroPhasor Initiative
NERC	North American Electric Reliability Council
OTCCA	One-Time Adaptive Chosen Ciphertext Attacks
PBES	Policy Based Encryption System
PDC	Phasor Data Concentrator
PDP	Policy Decision Point
PKEM	Policy and Key Encapsulation Mechanism
PMU	Phasor Measurement Unit
PS	Policy Specialization
PSS	Policy Specialization Server
RC	Reliability Coordinators
SPKEM	Symmetric Key Based Policy and Key Encapsulation Mechanism

Chapter 1

Introduction

In distributed systems users often need to share sensitive data with other users based on the latter's ability to satisfy various policies. In many cases the data owner may not even know the identities of the data recipients, but deems it crucial that they are legitimate; i.e., satisfy the policy. For example, consider a health care setting where an employee of a drug company is trying to target a message regarding the predicted side effects of a new drug to all patients at participating hospitals, who have a certain medical condition and have indicated a willingness to participate in clinical trials at their discretion. The drug company considers the potential side effects of the new drug private data and thus would like to keep the message confidential from anyone who does not satisfy the conditions above. However the drug company and thus the employee are not allowed to know the identity of patients until they actually sign up for the trial. This example also illustrates the expressiveness and flexibility that needs to be provided for such policies.

Enabling such data sharing over the Internet faces the challenge of (1) securely associating access policies with data and enforcing them, and (2) protecting data as it traverses untrusted proxies and intermediate repositories. Furthermore, it is desirable to achieve properties such as: (1) flexibility of access policies; (2) privacy of sensitive access policies; (3) minimal reliance on trusted third parties; and (4)

efficiency of access policy enforcement. Often schemes enabling controlled data sharing need to trade one property for another. In this dissertation, we (1) propose two complimentary policy-based data sharing schemes that achieve different subsets of the above desired properties, (2) formally analyze their security, and (3) and study their application to real systems. The first scheme addresses the problem of policy-based data sharing when there are no trusted mediating servers to enforce policies. In this case, while we minimize reliance on trusted third parties, we do not address policy privacy and limit ourselves to policies that do not use contextual information. The second scheme addresses the problem of policy-based data sharing when context-based policies, that is, policies that use context information, need to be supported and when privacy of policies needs to be protected. However, in this case we leverage a trusted server to enforce policies and incur some trust liability. We integrate the proposed schemes into two practical applications, the first scheme with Attribute-Based Messaging (ABM) and the second scheme with a conditional data sharing application in the Power Grid, and demonstrate their usefulness in practice.

The rest of this chapter is organized as follows. In Section 1.1 we present the problem of enabling policy based data sharing without a mediating server to enforce policies. In Section 4.4 we present the problem of enabling policy based data sharing that supports context-based policies and provides policy privacy. In Section 1.3 we present the applications with which we integrate the schemes proposed in this work. In Section 1.4 we present our contributions and in Section 1.5 we give an outline for the remaining part of this dissertation.

1.1 Policy-Based Data Sharing Without a Mediating Server

In some applications, a data owner may want to share data with other users based on the latter's ability to satisfy various policies but may not have access to or may not be willing to trust a server to enforce the access policies associated with his data. For example, consider users of social networking sites such as Facebook, who do not have access to trusted mechanisms that protect their private data at sufficient level of granularity. Another example is of a data owner who outsources data storage to a third party such as a cloud provider but is not willing to trust the provider with the content or to enforce his policies. In such situations, a typical solution is for the data owner to place encrypted data at a publicly accessible place, for example, the social networking site, and distribute the decryption keys to legitimate users. This requires that the data owner establish a secure channel with every user he wishes to share a given data item with. However, this is not always possible, as the data owner may not even know the identity of users that satisfy the policies associated with his data as illustrated by the health care example discussed above.

Attribute-Based Encryption (ABE) [63, 37, 10, 23, 56, 24, 55, 36, 43, 49] is a new public-key encryption paradigm that provides an appealing alternative in the above scenarios. ABE enables policy-based data sharing by associating and enforcing access policies cryptographically, eliminating the need for trusted mediating servers to enforce the policies. Existing ABE schemes come in two complementary forms, namely, Key-Policy ABE (KP-ABE) or Predicate encryption schemes [63, 37, 23, 56, 43] and Ciphertext-Policy ABE (CP-ABE) schemes [10, 24, 55, 36, 49]. In

KP-ABE or Predicate encryption schemes, as the name indicates, attribute policies (predicates) are associated with keys and data is annotated with attributes. Users are able to decrypt a ciphertext only if the attributes associated with it satisfy the predicate associated with their key. In CP-ABE schemes on the other hand, attribute policies are associated with data and attributes are associated with keys. Users are able to decrypt a ciphertext only if the attributes associated with their keys satisfy the policy associated with the ciphertext. In ABE schemes a trusted entity distributes attribute or predicate keys to users. Data owners encrypt their data using the public parameters provided by the trusted entity and attributes or an attribute policy and can make the encrypted data public. ABE schemes thus provide encrypt-and-publish semantics and minimize reliance on trusted third parties. CP-ABE is more intuitive as it is similar to traditional access control model where data is protected with access policies and users with credentials satisfying the policy are allowed access to the data. While a lot of the research effort in designing CP-ABE schemes has been devoted to: (1) improving expressiveness of policies supported and (2) providing privacy of policies, little attention has been paid to the organization of attributes within user keys. In this work, we motivate the need for flexible attribute organization within user keys of CP-ABE schemes for efficient support of many practical applications and propose Ciphertext-Policy Attribute Set Based Encryption (CP-ASBE) scheme, the first CP-ABE scheme to organizes attributes within user keys.

1.2 Data Sharing Based on Private, Context-Sensitive Policies

In some applications, data sharing is based on policies that include contextual information. For an example, consider the Electric Power Grid where power system operators need to cooperate with each other to operate the grid safely and reliably but they also compete with each other as business entities. A utility company, say A, might be willing to share sensitive sensor data from its electrical network with neighboring utility companies only when there is a frequency or voltage disturbance in the regional grid that adversely affects them but not under normal circumstances. While CP-ABE is a very useful encryption paradigm, existing CP-ABE schemes, including the one proposed in the first part of this dissertation, and ABE schemes in general, cannot efficiently support policies with contextual information, especially, when the context could be ephemeral as in the above example. This is because, for CP-ABE schemes to take contextual information into account, it should be made available in the user keys. But given that (1) contextual information is usually short lived, (2) key generation is very expensive in existing CP-ABE schemes, and (3) existing CP-ABE schemes lack revocation mechanisms, supporting context-based policies in CP-ABE schemes is currently infeasible. In this work, we propose Policy Based Encryption System (PBES), an encryption scheme and system that supports context-sensitive policies and policy privacy by employing mediated decryption while retaining the encrypt-and-publish semantics that CP-ABE schemes provide.

1.3 Applications

1.3.1 Attribute-Based Messaging

Attribute Based Messaging (ABM) enables messages to be addressed using attributes of recipients rather than an explicit list of recipients. Such messaging offers benefits of efficiency, exclusiveness, and intensionality, but faces challenges in access control and confidentiality. In [12] we employed Attribute-Based Access Control (ABAC) to provide a manageable access control mechanism and presented an ABM architecture leveraging existing messaging systems. However providing end-to-end confidentiality remained a challenge given that a message sender may not know who the legitimate recipients of his messages are. In this work we demonstrate use of Ciphertex-Policy Attribute Based Encryption to provide end-to-end confidentiality. We integrate our CP-ASBE scheme with ABM architecture and show that our scheme incurs very little overhead over an existing efficient CP-ABE scheme while providing much more flexibility.

1.3.2 Conditional Sharing of Phasor Measurement Unit (PMU) Data in the Power Grid

Increasing power consumption and major recent events such as the August 2003 blackout [73] means the power system operators are compelled to improve the reliability of the grid through wide area situational awareness, monitoring and control. Phasor Measurement Units (PMUs), envisioned to be deployed across the

grid, have the potential to provide wide area situational awareness when their data is shared among operators. In deregulated grids worldwide and in the North American grid in particular, utilities share sensitive data with their local Reliability Coordinators (RCs) as required by regulatory laws. However as shown by the example in Section 4.4 above, they might not be comfortable disclosing sensitive PMU data with other entities except under certain conditions including transient conditions in the grid at the time of access. In this work, through a prototype implementation and integration, we show that PBES scheme can meet the requirements of most applications that depend on shared PMU data.

1.4 Contributions

In this dissertation we make the following contributions:

1. We motivate the need for flexible organization of attributes within user keys of CP-ABE schemes for efficient support of many practical applications. We propose Ciphertext-Policy Attribute-Set Based Encryption (CP-ASBE), a novel CP-ABE scheme that organizes attributes within user keys. We show that, by organizing attributes in user keys into recursive sets and allowing policies to impose dynamic constraints on how the attributes within a key may be combined to satisfy a policy, we can (1) efficiently support naturally occurring compound attributes, (2) support multiple numerical assignments for a given attribute in a single key, and (3) provide efficient key management. We formally prove that the CP-ASBE scheme is secure against chosen plaintext

attacks (CPA) in the generic group model. We provide a library implementation of the scheme that is easy to integrate with applications.

2. We study the application of CP-ASBE, and CP-ABE in general, to a novel messaging paradigm the we proposed [12], namely, Attribute-Based Messaging (ABM). By integrating CP-ASBE with ABM architecture and evaluating it we show that CP-ASBE incurs very little overhead over an existing efficient CP-ABE scheme while providing much more flexibility.
3. We develop Policy and Key Encapsulation Mechanism Data Encapsulation Mechanism (PKEM-DEM) encryption scheme, which is a generic construction to securely associate and encapsulate policies and data. We present Policy Based Encryption System (PBES) that uses the PKEM-DEM scheme and leverages a trusted server for mediated decryption. We show that PBES, (1) supports flexible and expressive policies, including context-based policies, (2) provides policy privacy, and (3) provides encrypt-and-publish semantics. In addition to the security notions of message indistinguishability and policy indistinguishability, we define a new security notion of pair-wise indistinguishability for PBES where adversaries need to distinguish between pairs of messages and policies. We show that PBES satisfies the above security notions in the chosen ciphertext attack model.
4. We study the application of PBES to a real world problem, namely, conditional Phasor Measurement Unit data sharing in the Power Grid that calls for increased data sharing when the grid is unstable. Through prototype imple-

mentation and evaluation we show that PBES is efficient enough to support most power applications that depend on shared sensor data.

1.5 Dissertation Organization

The remaining part of this dissertation is organized as follows. In Chapter 2 we review the related work and provide a context for the contributions of this dissertation. In Chapter 3 we discuss the problem of enabling policy based data sharing without relying on a mediating server to enforce policies. In Chapter 4 we discuss the problem of enabling data sharing based on context sensitive policies while preserving the privacy of those policies. In Chapter 5 we study the application of the schemes proposed in Chapters 3 and 4 by integrating them with real systems and analyzing their performance. We discuss future directions and conclude in Chapter 6.

Chapter 2

Related Work

In this chapter we review the related work in the areas of policy-based data sharing without mediating servers to enforce the policy and data sharing based on private, context-sensitive policies and present our contributions in the context of the related work.

2.1 Policy-Based Data Sharing Without a Mediating Server

Many cryptography mediated data sharing schemes fall into this area and can be grouped into two classes, namely, data sharing between two parties and data sharing between multiple parties. Here we review relevant works in each class and contrast them with our work.

2.1.1 Secure Two-Party Data Sharing

Traditional hybrid public-key encryption standardized for e-mail in PEM [51], PGP [76], and S/MIME [59], and traditional identity-based encryption [15] enable two-party data sharing. However, they require the sender to know who the recipient is at the time of sending and thus can be considered as supporting data sharing based on a singleton policy which is the identity of the recipient. Policy-based cryptographic schemes proposed in [2, 8], and several works in the area of “hidden

policies and credentials” [18, 33, 48] enable data sharing based on a policy but focus on two-party interactions. In contrast we focus on multi-recipient data sharing based on flexible policies.

2.1.2 Secure Multi-Party Data Sharing

Secure multi-party data sharing is considered in many contexts. Here we focus on those contexts that do not rely on mediating servers to enforce policy including secure group communication, secure mailing lists and policy-based cryptography and attribute-based encryption. In secure group communication, a group of users share a secret-key that is updated whenever the group membership changes. Any member of the group could send a secure message to the group. However, in secure group communication members need to establish a shared secret key before communicating. Key distribution and key agreement schemes for secure group communication include [20, 58, 71, 74, 62, 44, 65, 40, 47, 13]. Furthermore, secure group communication is efficient for long standing data sharing associations and can be considered as supporting data sharing based on group membership attribute, *i.e.*, a singleton attribute policy.

Secure mailing lists [46, 45] provide confidentiality for messages sent on mailing lists using a partially trusted list server, *i.e.*, the messages contents are not revealed to the list server. Similar to secure group communication schemes secure mailing lists are efficient for long standing data sharing associations and can be considered as supporting data sharing based on mailing list membership attribute,

i.e., a singleton attribute policy. In contrast in our work we focus on enabling data sharing based on flexible and expressive policies.

Attribute-Based Encryption is the most closely related work to ours. While the concepts and ideas related to Attribute-Based Encryption have been alluded to in literature as far back as [15, 25] Sahai and Waters [63] proposed what is considered the first ABE scheme. Their scheme supported policies with a single threshold gate. Furthermore, the threshold value k , and size of the gate n used in a policy, are fixed during setup in their *Large Universe* construction. Pirretti *et al.*, [57] showed how to overcome this limitation of fixed k and n and demonstrated the use of threshold access policies for two applications. Traynor *et al.*, [72] further demonstrated its scalability by applying it to massive conditional access systems. Goyal *et al.*, [37] first defined the two complimentary forms of ABE, namely, KP-ABE and CP-ABE, and provided a construction for a KP-ABE¹ scheme. The proposed KP-ABE scheme supported all monotonic boolean encryption policies and was later extended by Ostrovsky *et al.*, [56] to support non-monotonic boolean formulas.

Bethencourt *et al.*, [10] gave the first construction for a CP-ABE scheme. Their construction supported all monotonic boolean encryption policies and the security of their scheme was argued in the generic group model. Cheung and Newport [24] gave the first standard model construction of CP-ABE scheme. While their scheme supported both positive and negative attributes it was limited to policies with single AND gates. Nishide *et al.*, [55] extended the scheme in [24] to support policy secrecy. Goyal *et al.* gave the first standard model construction of CP-ABE scheme that

¹The scheme proposed in [63] can in retrospect be viewed as a KP-ABE scheme.

could support flexible policies [36]. Their scheme can realize all non-monotonic boolean formulas. However, since it is constructed using a KP-ABE scheme of [37], it is inefficient and has bounded ciphertext, *i.e.*, the size of supported policies is fixed at setup. Katz *et al.* proposed a KP-ABE scheme in [43] that can support flexible policies and achieve policy secrecy. This scheme can be used to realize CP-ABE schemes but such schemes have a bounded ciphertext. Most of the past work on CP-ABE schemes and ABE schemes in general, which enable policy-based data sharing without a trusted mediator, is focused on improving the expressibility of encryption policies and providing policy privacy. In contrast ours is the first work to consider the organization of attributes within user keys which we demonstrate is necessary in practical applications. Our CP-ASBE scheme is the first to organize user attributes in keys and allow users to impose dynamic constraints on how attributes can be combined to satisfy policies, allowing our scheme more flexibility and efficiency in practice.

Support for numerical attributes in CP-ABE schemes was first discussed in [10]. While the technique may be applicable to other schemes none of the existing CP-ABE schemes can support multiple value assignments for a given numerical attribute within a single key. Our CP-ASBE scheme is the first scheme to do so allowing it to support applications where such attribute assignments are needed without sacrificing flexibility of range queries (*i.e.*, numerical comparisons) in policies for those attributes.

Policy-based cryptography scheme proposed in [7] enables data sharing based on flexible policies but is vulnerable to collusion attacks. That is, users who do not

individually satisfy the policy associated with a given cipher-text may still be able to decrypt it when they collude.

2.2 Data Sharing Based on Private, Context-Sensitive Policies

Our work in this area touches upon topics in areas of policy/attribute based encryption, hidden credentials and policies, cryptographic file systems and efficient and effective key management. Here we review relevant works in each of these areas and contrast them with our work. Some of these works are reviewed in Section 2.1 above but are repeated here for completeness.

Identity Based Encryption (IBE) [16, 25] schemes and messaging systems employing it [52] allow the association of a flexible policy with objects and support exchange in open distributed systems but do not keep the policy secret and are designed for two-party communication where the sender identifies the recipient in the encryption. Similarly, Policy-based cryptographic schemes proposed in [2, 8] allow the association of a flexible policy with objects and support exchange in open distributed systems but do not keep the policy secret and are designed for two-party communication where the sender identifies the recipient in the encryption. Several works in the area of “hidden policies and credentials” [19, 33, 48] provide message and policy secrecy but focus on two-party interactions.

Attribute Based Encryption (ABE) systems such as Ciphertext-Policy ABE (CP-ABE) [10, 24, 36, 49] including our CP-ASBE scheme and cryptographic file system FSGuard [69] allow the association of flexible policies with objects for multiple

recipients defined by those policies and support exchange in open distributed systems but do not provide policy secrecy and cannot support context-based policies. Recent work by [55] extends CP-ABE to support policy secrecy but significantly limits its policy flexibility and does not support context-based policies. Predicate Encryption scheme proposed in [43] also allows the association of flexible policies with objects for multiple recipients defined by those policies, supports exchange in open distributed systems and provides policy secrecy but does not support context-based policies. PEAPOD [42] focuses on one-to-many messaging with both message and policy secrecy but does not provide efficient key management and is also vulnerable to collusion attacks.

Kapadia *et al.*, [42], leverage the proxy re-encryption solution of [46] and propose an attribute-based publishing scheme that allows users to publish data encrypted under an attribute policy so that only users who satisfy the policy can decrypt it with the additional property that the policy associated with the ciphertext remains private. However, this scheme suffers from the drawback that it is susceptible to collusion.

The work that probably comes closest to ours is the enterprise object encryption architecture proposed by [32] back in 1994. In their architecture a Key Release Agent releases decryption keys to users after authentication in a manner similar to that done by KDC in PBES. However, they do not develop a secure policy based encryption scheme for their architecture.

Chapter 3

Policy Based Data Sharing Without a Mediating Server

In distributed systems users need to share sensitive objects with others users based on the latter's ability to satisfy a policy. In some cases this data sharing needs to be accomplished without relying on trusted mediated servers. For example, users of many social networking sites such as Facebook, do not have access to trusted mechanisms that protect their private data at sufficient level of granularity. Attribute-Based Encryption (ABE) ushers in a new paradigm where such policies are specified and cryptographically enforced in the encryption algorithm itself. Existing ABE schemes come in two complimentary forms, namely, Key-Policy ABE (KP-ABE) schemes and Ciphertext-Policy ABE (CP-ABE) schemes. In KP-ABE schemes [37, 43, 56, 63], as the name indicates, attribute policies are associated with keys and data is annotated with attributes. Only those keys associated with a policy that is satisfied by the attributes annotating the data are able to decrypt the data. In CP-ABE schemes [10, 24, 36, 55], on the other hand, attribute policies are associated with data and attributes are associated with keys. Only those keys whose associated attributes satisfy the policy associated with the data are able to decrypt it.

CP-ABE is more intuitive as it is similar to traditional access control model where data is protected with access policies and users with credentials satisfying

the policy are allowed access to it. Among the various CP-ABE schemes proposed the one proposed by Bethencourt *et al.* [10], which we will hereafter refer to as BSW, is the most practical to date. It supports arbitrary strings as attributes, numerical attributes in keys and integer comparisons in policies and provides a means for periodic key refreshment. Furthermore, the authors have developed a software prototype with a friendly interface for integration in systems. However, BSW and other CP-ABE schemes are still far from being able to support the needs of practical applications, which require considerable flexibility in specifying policies and managing user attributes as well as increased efficiency. This is in part due to the fact that keys in current CP-ABE schemes can only support user attributes that are organized logically as a single set; *i.e.*, users can use all possible combinations of attributes issued in their keys to satisfy policies. This, we observe, imposes some undesirable restrictions which are outlined below.

First, this makes it both cumbersome and tedious to capture naturally occurring “compound attributes”, *i.e.*, attributes build intuitively from other (singleton) attributes, and specifying policies using those attributes. For example, attributes that combine a traditional organizational role with short-term responsibilities result in useful compound attributes; *e.g.*, ‘Faculty’ in ‘College of Engineering’ serving as ‘Committee Chair’ of a ‘University Tenure Committee’ in ‘Spring2009’ are all valid attributes in their own right and are likely to be used to describe users. The only way to prevent users from combining such attributes in undesirable ways when using current CP-ABE schemes is by appending the (singleton) attributes as strings; *i.e.*, *faculty_collegeOfEngineering_committeeChair_univTenureCommittee_Spring2009*. But

this approach has an undesirable consequence in that it makes it challenging to support policies that involve other combinations of singleton attributes used to build the compound attribute; *e.g.*, policies targeting “all committee chairs in Spring2009” or “faculty serving on tenure committees”. This is because the underlying crypto in CP-ABE schemes can only check for equality of strings and thus cannot extract the “faculty” or “committeeChair” attributes from a compound attribute such as the one described above.

Second, CP-ABE schemes that support numerical attributes (*i.e.*, allow numerical comparisons in policies) are limited to assigning only one value to any given numerical attribute within a key. But there are many real world systems where multiple numerical value assignments for a given attribute are common; *e.g.*, students enrolled in multiple courses identified by numeric course numbers in a given semester, users with multiple accounts at a particular bank, disease codes for individual diseases and disease classes used widely in health care. Furthermore, the ability to compare across such multiple value assignments adds flexibility to policy specification. For example, consider a college student enrolled in two junior level courses, 357 and 373, and two senior level courses, 411 and 418 respectively. Without support for multiple numerical value assignments for a given attribute specifying policies to target students enrolled in senior level courses, such as “course number greater than or equal to 400 and less than 500” is tedious and cumbersome.

In this chapter, we propose Ciphertext-Policy Attribute-Set Based Encryption (CP-ASBE), a form of CP-ABE, that addresses the above limitations of CP-ABE by introducing a recursive set based structure on attributes associated with user

keys. Specifically CP-ASBE allows, 1) user attributes to be organized into a recursive family of sets and 2) policies that can selectively restrict decrypting users to use attributes from within a single set or allow them to combine attributes from multiple sets. Thus, by grouping user attributes into sets such that those belonging to a single set have no restrictions on how they can be combined, CP-ASBE can support compound attributes without sacrificing the flexibility to easily specify policies involving the underlying singleton attributes. Similarly, multiple numerical assignments for a given attribute can be supported by placing each assignment in a separate set.

While restricting users to use attributes from a single set during decryption can be thought of as a regular CP-ABE scheme, the challenge in constructing a CP-ASBE scheme is in selectively allowing users to combine attributes from multiple sets within a given key while still preventing collusion, *i.e.*, preventing users from combining attributes from multiple keys. We provide a construction for a CP-ASBE scheme that builds on BSW and evaluate its performance through a prototype implementation. We show that our construction is secure against *chosen-plaintext attacks* in the generic group model. However, our construction can be efficiently extended to be secure against *chosen-ciphertext attacks* using a transformation like Fujisaki-Okamoto [34, 75] or the techniques of Canetti, Halevi and Katz [22] just like the BSW scheme [10].

The rest of this chapter is organized as follows. In Section 3.1 we further demonstrate the limitations of existing CP-ABE schemes and motivate the need for CP-ASBE. In Section 3.2 we give some preliminaries. In Section 3.3 we present

our construction of CP-ASBE. In Section 3.4 we formally prove its security. In Section 3.5 we discuss optimizations and discuss the efficiency of the scheme.

3.1 Need for Organizing Attributes in CP-ABE User Keys

The ability to group attributes into sets and to frame policies that can selectively restrict the decrypting key to use attributes belonging to the same set is a powerful feature more than one might realize initially. In this section we illustrate its versatility by solving various problems in different contexts which did not have any reasonably efficient solutions prior to this.

3.1.1 Supporting Compound Attributes Efficiently

While existing CP-ABE schemes offer unprecedented expressive power for addressing users, for several natural scenarios they are inadequate. We illustrate this with the following natural example and show how CP-ASBE provides a simple solution.

Consider attributes for students derived from courses they have taken. Each student has a set of attributes (Course, Year, Grade) for each course she has taken. In the following, consider a simple policy “Students who took a $300 \leq \text{Course} < 400$ in $\text{Year} \geq 2007$ and got $\text{Grade} > 2$.” Using a CP-ABE scheme for this is challenging because, for instance, a student can take multiple courses and obtain different grades in them. The policy circuit will have to ensure that she cannot mix together attributes from different sets to circumvent the policy. We point out

a few possible options of using CP-ABE, but all unrealistic or unsatisfactory. The efficiency parameters considered are the number of designed attributes given to each student, and the size of the designed policy (a circuit, with designed attributes as inputs, for enforcing the policy).

- For each course that the student has taken, let there be a single designed (boolean) attribute that she gets (e.g. `cyg:373_2008_4`). But the designed policy will have to (unrealistically) anticipate all such attributes that will satisfy the policy (e.g., `cyg:300_2007_3` or `cyg:301_2007_3` or ... or `cyg:399_2010_4`).
- Anticipate (again, unrealistically) all possible policies that may occur which the student's attributes will satisfy, and give her compound boolean attributes corresponding to each of these policies (e.g., `cyg:373_2008_4`, `cyg:373_2008`, `cyg:(≥300)_2008`, `cyg:(≥400)_2007-or-cyg:(≥300)_2008-(≥3)`, ...). In this case our designed policy is minimal, with just an input gate (labeled by the attribute `cyg:(≥ 300, < 400)-(≥ 2007)-(> 2)`) and an output gate.
- Fix an upper bound on the number of courses a student could ever take, say 50, and give all attributes indexed by a counter (e.g. `Course#1`, `Year#1`, `Grade#1` etc.); then the policy will have to incorporate several cases (e.g., $(400 < \text{Course\#1} \geq 300 \text{ and } \text{Year\#1} \geq 2007 \text{ and } \text{Grade\#1} > 2)$ or ... or $(400 < \text{Course\#50} \geq 300 \text{ and } \text{Year\#50} \geq 2007 \text{ and } \text{Grade\#50} > 2)$). This increases the policy size by a factor of 50.

If a policy can refer to more than one course, all these approaches will lead to even more inefficiency or restrictions. In particular, in the third (and the most efficient)

approach, if a policy refers to just two courses, the blow up will be by a factor of 2500 instead of 50.

We stress that these are not the only possibilities when using CP-ABE. In general, by giving more attribute keys, the circuit complexity of the policies can be reduced (the first two options above being close to the two extremes). One could achieve slightly smaller policies by adding judiciously chosen auxiliary attributes and adding some structure to values taken by these attributes (for instance, in the third option above, one can let the counter monotonically increase with the course number). However, the resulting schemes are still unrealistically inefficient in terms of policy size and/or number of keys, and *further* makes attribute revocation even less efficient.

A CP-ASBE scheme can be used to overcome these issues by assigning multiple values to the group of attributes but in different sets. In our example, for each course that a student has taken, she gets a separate set of values for the attributes (Course, Grade, Year). Thus the number of designed attributes she receives is comparable to the number of natural attributes she has; further, the designed policy is comparable in size to that of a policy that did not enforce the requirement that attributes from different courses should not be mixed together. In short, using CP-ASBE, we can obtain efficient ciphertext policy encryption schemes for several scenarios where existing CP-ABE scheme are insufficient.

Expressiveness in terms of Attribute-Databases Supported. Some of the flexibility illustrated above can be understood by viewing the association of at-

tributes to a user as an entry in a database table. In such a table — which we will call the *attribute table* — each row stands for a user and each column (other than user identity) for an attribute.¹ The policy associated with a cipher-text could be considered a query into this table, to identify all users whose attributes satisfy a certain predicate.

The expressive power of a CP-ABE scheme is given by the class of queries into this table that the scheme can support. For instance, BSW CP-ABE [10] supports a large class of such queries. One challenge to increase the expressive power would be to broaden this class. However, there is another important dimension in which the expressive power of CP-ABE scheme can be improved, by supporting a more general class of attribute tables. The above description of CP-ABE required that each user ID appears in only one row in the table. (In other words, the user ID must be a “superkey” in the attribute table.) Of course, a table can be forced to have this property, but leading to large blow ups in the number of designed attributes that a user receives or the size of the designed policy. On the other hand, a CP-ASBE scheme can directly support a table with multiple rows per user: attributes in each row is given as a separate set.

¹In the case of a “large universe” of attributes, the number of columns could be very large — say all strings of 256 bits — and the resulting sparse table will never be stored directly as a table. Our examples shall mostly use the small universe scenarios, though they extend to the large universe setting as well.

3.1.2 Supporting Multiple Value Assignments

A major motivation for CP-ASBE is to support multiple value assignments for a given attribute in a single key.² To illustrate this, suppose `score` is a 6-bit integer representing the score a user receives in a game. (The user may possess several other attributes in the system.) The user can play the game several times and receive several values for `score`. This numerical attribute will be represented by 12 boolean attributes: `score_bit0_0`, `score_bit0_1`, . . . , `score_bit6_0` and `score_bit6_1`, corresponding to the values 0 and 1 for the six bits in the binary representation of the value. Now consider a user who has two values of `score`, 33 (binary 100001) and 30 (binary 011110). By obtaining attributes for the bit values of these two numbers, the user gets all 12 boolean attributes, effectively allowing him to pretend to have any score he wants.

CP-ASBE solves this problem elegantly: each value assignment of the numerical attribute is represented in a separate set with six boolean attributes each (one for each bit position). Note that attributes other than `score` need not be repeated.

Application: Efficient revocation. ABE schemes suffer from lack of an effective revocation mechanism for keys that have been issued (just like IBE). To address this in CP-ABE in a limited manner, Bethencourt *et al.* [10] propose adding an `expiration_time` attribute to a user’s key indicating the time (i.e., a numerical value) until which the key is considered to be valid. Then a policy can include a check

²Note that multiple values for an attribute is relevant only when the attribute in question is not a boolean attribute (in a monotonic policy).

on the `expiration_time` attribute as a numerical comparison. However, in practice the validity period of sensitive attributes has to be kept small to reduce the *window of vulnerability* when a key is compromised, *e.g.* a day, a week or a month. At the end of this period the entire key will have to be re-generated and re-distributed with an updated expiration time imposing a heavy burden on the key server and key distribution process.

CP-ASBE solves this problem more efficiently. First, we observe that while key validity is limited because of the window of vulnerability, the actual attribute assignments change far less frequently. Second, we observe that it is possible to add attributes retroactively to a user key, both in BSW CP-ABE and CP-ASBE, if the key server is able to maintain some state information about the user key. Then, by allowing multiple value assignments to the `expiration_time` attribute we can simply add a new expiration value to the existing key. Thus, while we require the key server to maintain some state we avoid the need to generate and distribute new keys on a frequent basis. This reduces the burden on the key server by a factor proportional to the average number of attributes in user keys.

3.2 Preliminaries

Bilinear Maps. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be cyclic (multiplicative) groups of order p , where p is a prime. Let g_1 be a generator of \mathbb{G}_1 , and g_2 be a generator of \mathbb{G}_2 . Then $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a bilinear map if it has the following properties:

1. Bilinearity: for all $u \in \mathbb{G}_1, v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.

2. Non-degeneracy: $e(g, h) \neq 1$.

Usually, $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$. \mathbb{G} is called a bilinear group if the group operation and the bilinear map e are both efficiently computable.

Key Structure. In CP-ABE schemes, an encryptor specifies an access structure for a ciphertext which is referred to as the ciphertext policy. Only users with secret keys whose associated attributes satisfy the access structure can decrypt the ciphertext. In CP-ABE schemes so far, a user's key can logically be thought of as a set of elements each of which corresponds to an associated attribute, such that only elements within a single set may be used to satisfy any given ciphertext policy (*i.e.* collusion resistance). In our scheme however, we use a recursive set based key structure where each element of the set is either a set itself (*i.e.* a key structure) or an element corresponding to an attribute. We define a notion of *depth* for this key structure, which is similar to the notion of depth for a tree, that limits this recursion. That is, for a key structure with depth 2, members of the set at depth 1 can either be attribute elements or sets but members of a set at depth 2 may only be attribute elements. The following is an example of a key structure of depth 2:

$$\left\{ CS\text{-Department}, Grad\text{-Student}, \{ Course101, TA \}, \{ Course525, Grad\text{-Student} \} \right\}$$

The depth of key structures that can be supported by our scheme is a system parameter that should be decided at the time of setup. That is, if the system is setup with a depth parameter of 5, keys of depth 5 or less can be supported. For ease of exposition, we will describe our scheme for key structures of depth 2. But our construction is easily generalized to support keys of any depth d where d is fixed

at setup.

The key structure defines unique labels for sets in the key structure. For key structures of depth 2, just an index (arbitrarily assigned) of the set among sets at depth 2 is sufficient to uniquely identify the sets. Thus if there are m sets at depth 2 then a unique index i where $1 \leq i \leq m$ is (arbitrarily) assigned to each set. The set at depth 1 is referred to as set 0 or simply the outer set. If ψ represents a key structure then let ψ_i represent the i th set in ψ . Individual attributes inherit the label of the set they are contained in and are uniquely defined by the combination of their name and their inherited label. That is, while a given attribute might appear in multiple sets it can appear only once in any set. In the above example, the outer set and $\{Course525, Grad-Student\}$ are assigned labels 0 and 2 respectively, and the two instances of the attribute *Grad-Student* are distinguished by the unique combination of their inherited set label and attribute name, $(0, Grad-Student)$ and $(2, Grad-Student)$, respectively. By default, a user may only use attribute elements within a set to satisfy a given ciphertext policy. That is, a user with the key structure from the above example may combine individual attributes either from the outer set (*i.e.*, $\{CS-Department, Grad-Student\}$) or from the set $\{Course101, TA\}$ or from the set $\{Course525, Grad-Student\}$ to satisfy the policy associated with a given ciphertext but may not combine attributes across the sets. However, an encryptor may choose to allow combining attributes from multiple sets to satisfy the access structure by designating *translating nodes* in the access structure as explained below.

Access Structure. We build on the access structure used in [10] which is a tree whose non-leaf nodes are threshold gates. Each non-leaf node of the tree is defined by its children and a threshold value. Let nc_x denote the number of children and k_x the threshold value of node x , then $0 < k_x \leq nc_x$. When $k_x = 1$, the threshold gate is an OR gate and when $k_x = nc_x$ it is an AND gate. The access tree also defines an ordering on the children of a node, *i.e.*, they are numbered from 1 to nc_x . For node x such a number is denoted by $\mathbf{index}(x)$. Each leaf node y of the tree is associated with an attribute which is denoted by $\mathbf{att}(y)$. Furthermore, the encrypting user may designate some nodes in an access tree as *translating nodes*. Their function will become clear as we discuss below the conditions under which a key structure is said to satisfy an access tree.

Let \mathcal{T} be an access tree whose root node is r . Let \mathcal{T}_x denote a subtree of \mathcal{T} rooted at node x . Thus \mathcal{T}_r is the same as \mathcal{T} . Now we will define the conditions under which a key structure ψ is said to satisfy a given access tree \mathcal{T} assuming there are no designated translating nodes in the access tree. We will then extend the definition to consider the presence of translating nodes. A key structure ψ is said to satisfy the access tree \mathcal{T} if and only if $\mathcal{T}(\psi)$ returns a non-empty set S of labels. We evaluate $\mathcal{T}_x(\psi)$ recursively as follows. If x is a non-leaf node we evaluate $\mathcal{T}_{x'}(\psi)$ for all children x' of x . $\mathcal{T}_x(\psi)$ returns a set S_x containing unique labels such that for every label $lbl \in S_x$ there exists at least one set of $k \geq k_x$ children such that for each child x' of these k children $S_{x'}$ contains the label lbl . If x is a leaf node then the set S_x returned by $\mathcal{T}_x(\psi)$ contains a label lbl if and only if $\mathbf{att}(x) \in \psi_{lbl}$. Thus a key structure is said to satisfy an access tree if it contains at least one set that has

all the attributes needed to satisfy the access tree. Note that attributes belonging to multiple sets in the key structure cannot be combined to satisfy the access tree.

However, if there are designated translating nodes in the access tree, the algorithm $\mathcal{T}(\psi)$ is modified as follows. The algorithm $\mathcal{T}_x(\psi)$ is the same as above when x is a leaf node. When x is a non-leaf node we evaluate $\mathcal{T}_{x'}(\psi)$ for all children x' of x . $\mathcal{T}_x(\psi)$ returns a set S_x containing unique labels such that for every label $lbl \in S_x$ there exists at least one set of $k \geq k_x$ children such that for each child x' of these k children $S_{x'}$ either contains the label lbl or x' is a translation node and $S_{x'} \neq \emptyset$. Thus, if node x is a designated translating node then, even if the attribute elements used to satisfy the predicate represented by the subtree rooted at x belong to a different set in the key structure than those used to satisfy the predicates represented by the siblings of x the decrypting user is able to combine them to satisfy the predicate represented by the parent node of x .

Syntax of CP-ASBE Scheme. A CP-ASBE scheme consists of four algorithms, **Setup**, **KeyGen**, **Encrypt** and **Decrypt**. The algorithm **Setup** produces a master key and a public key for the scheme. **KeyGen** takes as input the master-key, a user's identity and an attribute set; it produces a secret key for the user. **Encrypt** takes as input the public key of the scheme, a message and an access tree, and outputs a ciphertext. Finally, **Decrypt** takes a ciphertext and a secret-key (produced by **KeyGen**), and if the access-tree used to construct the ciphertext is satisfied by the attribute set for which the secret-key was generated, then it recovers the message from the ciphertext.

Security of CP-ASBE Scheme. Our notion of *message indistinguishability* for CP-ASBE scheme against *chosen-plaintext attacks* is similar to that for CP-ABE schemes [10].

Setup. The challenger runs the Setup algorithm and gives public parameters, PK, to the adversary.

Phase 1. The adversary makes repeated queries for private keys corresponding to attribute sets $\mathbb{A}^1, \dots, \mathbb{A}^{q_1}$.

Challenge. The adversary submits two equal length messages M_0 and M_1 , and a challenge access structure \mathcal{T}^* such that none of the private keys obtained in Phase 1 corresponding to attribute sets $\mathbb{A}^1, \dots, \mathbb{A}^{q_1}$ satisfy the access structure. The challenger flips a random coin b , and encrypts M_b under \mathcal{T}^* . The resulting ciphertext CT is given to the adversary.

Phase 2. Phase 1 is repeated with the restriction that none of the attribute sets $\mathbb{A}^{q_1+1}, \dots, \mathbb{A}^q$ satisfy the access structure corresponding to the challenge.

Guess. The adversary outputs a guess b' of b .

The advantage of an adversary \mathcal{A} in this game is defined as $Pr[b' = b] - \frac{1}{2}$.

This game could easily be extended to include chosen-ciphertext attacks by allowing for decryption queries in Phase 1 and Phase 2.

Definition 3.1. A CP-ASBE scheme is secure against chosen-plaintext attacks if all probabilistic polynomial time adversaries have at most a negligible advantage in the game above.

3.3 Our CP-ASBE Construction

A key challenge in designing CP-ABE schemes is preventing users from pooling together their attributes. BSW CP-ABE achieves this by binding together all the attribute key components for each user with a random number unique to the user. Since in a CP-ASBE scheme one must prevent arbitrary combination of attributes belonging to different sets (even if they belong to the same user), a natural idea would be to similarly use a unique random number for binding together attribute key components for each set, in addition to using a random number for each user. However, a CP-ASBE scheme must also support *specific combinations* of attributes from different sets, as specified in an access-tree. The key idea in our construction is to include judiciously chosen additional values in the ciphertext (and in the key) that will allow a user to combine attributes from multiple sets all belonging to the same user. As it turns out, such a modification could introduce new subtle ways for multiple users to combine their attributes. Our construction shows how to thwart such attacks, using appropriate levels of randomization among different users' keys.

Let \mathbb{G}_0 be a bilinear group of prime order p and let g be a generator of \mathbb{G}_0 . Let $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ denote a bilinear map. Let $H : \{0, 1\}^* \rightarrow \mathbb{G}_0$ be a hash function that maps any arbitrary string to a random group element. We will use this function to map attributes described as arbitrary strings to group elements.

Setup($d = 2$). The setup algorithm chooses random exponents $\alpha, \beta_i \in \mathbb{Z}_p \forall i \in \{1, 2\}$. The algorithm sets the public key and master key as:

$$\begin{aligned} \text{PK} &= (\mathbb{G}, g, h_1 = g^{\beta_1}, f_1 = g^{\frac{1}{\beta_1}}, h_2 = g^{\beta_2}, f_2 = g^{\frac{1}{\beta_2}}, e(g, g)^\alpha) \\ \text{MK} &= (\beta_1, \beta_2, g^\alpha) \end{aligned}$$

Note that to support key structures of depth d , i will range from 1 to d .

KeyGen(MK, \mathbb{A} , u). Here u is the identity of a user and $\mathbb{A} = \{A_0, A_1, \dots, A_m\}$ is a key structure. A_0 is the set of individual attributes in the outer set (*i.e.* set 0) and A_1 to A_m are sets of attributes at depth 2 that the user has. Let $A_i = \{a_{i,1}, \dots, a_{i,n_i}\}$. That is, $a_{i,j}$ denotes the j -th attribute appearing in set A_i , and n_i denotes the number of attributes in the set A_i . (Note that for different values of (i, j) , $a_{i,j}$ can be the same attribute.) The key generation algorithm chooses a unique random number, $r^{\{u\}} \in \mathbb{Z}_p$, for user u . It then chooses a set of m unique random numbers, $r_i^{\{u\}} \in \mathbb{Z}_p$, one for each set $A_i \in \mathbb{A}, 1 \leq i \leq m$. For set A_0 , $r_0^{\{u\}}$ is set to be the same as $r^{\{u\}}$. It also chooses a set of unique random numbers, $r_{i,j}^{\{u\}} \in \mathbb{Z}_p$, one for each $(i, j), 0 \leq i \leq m, 1 \leq j \leq n_i$. The issued key is:

$$\begin{aligned} \text{SK}_u &= \left(\mathbb{A}, D = g^{\frac{(\alpha + r^{\{u\}})}{\beta_1}}, \right. \\ &\quad \left. D_{i,j} = g^{r_i^{\{u\}}} \cdot H(a_{i,j})^{r_{i,j}^{\{u\}}}, D'_{i,j} = g^{r_{i,j}^{\{u\}}} \quad \text{for } 0 \leq i \leq m, 1 \leq j \leq n_i, \right. \\ &\quad \left. E_i = g^{\frac{(r^{\{u\}} + r_i^{\{u\}})}{\beta_2}} \quad \text{for } 1 \leq i \leq m \right) \end{aligned}$$

Note that the operations on the exponents in the above equations are modulo the order of the group, which is prime. Hence division in the exponent is well-defined.

We omit the *mod* for convenience. Elements E_i enable translation from $r_i^{\{u\}}$ (*i.e.*, set A_i at depth 2) to $r^{\{u\}}$ (*i.e.*, the outer or parent set A_0 at depth 1) at the translating nodes. Elements E_i and $E_{i'}$ can be combined as $E_i/E_{i'}$ to enable translation from $r_{i'}^{\{u\}}$ (*i.e.*, set $A_{i'}$) to $r_i^{\{u\}}$ (*i.e.*, the set A_i) at the translating nodes. Similarly, for a key structure of depth d , there will be elements that enable translation from a set at depth d to its parent set at depth $d - 1$ and they will use β_d and random numbers corresponding to the appropriate sets.

Encrypt(PK, M, \mathcal{T}). M is the message, \mathcal{T} is an access tree. The algorithm associates a polynomial q_τ with each node τ (including the leaves) in the tree \mathcal{T} . These polynomials are chosen in the following way in a top-down manner, starting from the root node R . For each internal node τ in the tree, the degree d_τ of the polynomial q_τ is set to be one less than the threshold value k_τ of that node, that is, $d_\tau = k_\tau - 1$. For leaf nodes the degree is set to be 0. For the root node R the algorithm picks a random $s \in Z_p$ and sets $q_R(0) = s$. Then, it chooses d_R other points randomly to define the polynomial q_R completely. For any other node τ , it sets $q_\tau(0) = q_{\text{parent}(\tau)}(\text{index}(\tau))$ and chooses d_τ other points randomly to completely define q_τ . Here **parent**(τ) denotes the parent node of τ . Let \mathbb{Y} denote the set of leaf nodes in \mathcal{T} . Let \mathbb{X} denote the set of *translating nodes* in the access tree \mathcal{T} . Then the ciphertext CT returned is as follows:

$$\text{CT} = (\mathcal{T}, \tilde{C} = M \cdot e(g, g)^{\alpha \cdot s}, C = h_1^s, \bar{C} = h_2^s, \forall y \in \mathbb{Y} : C_y = g^{q_y(0)}, \\ C'_x = H(\text{att}(y))^{q_y(0)}, \forall x \in \mathbb{X} : \hat{C}_x = h_2^{q_x(0)})$$

Translating values \hat{C}'_x s together with E'_i s in user keys allow translation between sets at a translating node x as will be described in the Decrypt function. Note that the element \bar{C} is the same as \hat{C}_r where r denotes the root node. A variant of the scheme would be where \bar{C} is not included in the ciphertext but is only released at the discretion of the encrypting user as \hat{C}_r . This would restrict decrypting users to only use individual attributes in the outer set except when explicitly allowed by the encrypting user by designating translating nodes.

Decrypt(CT, SK $_u$). Here we describe the most straightforward decryption algorithm without regard to efficiency. The decryption algorithm is a recursive algorithm similar to the tree satisfaction algorithm described in Section 3.2. The decryption algorithm first runs the tree satisfaction algorithm on the access tree with the key structure *i.e.*, $\mathcal{T}(\mathbb{A})$, and stores the results of each of the recursive calls in the access tree \mathcal{T} . That is, each node t in the tree is associated with a set S_t of labels that was returned by $\mathcal{T}_t(\mathbb{A})$. If \mathbb{A} does not satisfy the tree \mathcal{T} then the decryption algorithm returns \perp . Otherwise the decryption algorithm picks one of the labels, i , from the set returned by $\mathcal{T}(\mathbb{A})$ and calls a recursive function **DecryptNode**(CT, SK $_u$, t , i) on the root node of the tree. Here CT is the ciphertext CT = $(\mathcal{T}, \tilde{C}, C, \forall y \in Y : C_y, C'_y, \forall x \in X : \hat{C}_x)$, SK $_u$ is a private key, which is associated with a key structure denoted by \mathbb{A} , t is a node from \mathcal{T} , and i is a label denoting a set of \mathbb{A} . Note that the ciphertext CT now contains tree information that is augmented by the results from $\mathcal{T}(\mathbb{A})$. **DecryptNode**(CT, SK $_u$, t , i) is defined as follows.

If $t \in \mathbb{Y}$, *i.e.*, node t is a leaf node, then $\mathbf{DecryptNode}(\text{CT}, \text{SK}_u, t, i)$ is defined as follows. If $\text{att}(t) \notin A_i$ where $A_i \in \mathbb{A}$ then $\text{DecryptNode}(\text{CT}, \text{SK}_u, t, i) = \perp$. If $\text{att}(t) = a_{i,j} \in A_i$ where $A_i \in \mathbb{A}$ then:

$$\begin{aligned} \text{DecryptNode}(\text{CT}, \text{SK}_u, t, i) &= \frac{e(D_{i,j}, C_t)}{e(D'_{i,j}, C'_t)} \\ &= \frac{e(g^{r_i^{\{u\}}} \cdot H(a_{i,j})^{r_{i,j}^{\{u\}}}, g^{q_t(0)})}{e(g^{r_{i,j}^{\{u\}}}, H(a_{i,j})^{q_t(0)})} = e(g, g)^{r_i^{\{u\}} \cdot q_t(0)} \end{aligned}$$

Note that set from which the satisfying attribute $a_{i,j}$ was picked is implicit in the result $e(g, g)^{r_i^{\{u\}} \cdot q_t(0)}$ (*i.e.*, indicated by $r_i^{\{u\}}$). When $t \notin \mathbb{Y}$, *i.e.*, node t is a non-leaf node, then $\text{DecryptNode}(\text{CT}, \text{SK}_u, t, i)$ proceeds as follows:

1. Compute \mathbb{B}_t which is an arbitrary k_t sized set of child nodes z such that $z \in B_t$ only if either (1) label $i \in S_z$ or (2) label $i' \in S_z$ for some $i' \neq i$ and z is a translating node. If no such set exists then return \perp .
2. For each node $z \in B_t$ such that label $i \in S_z$ call $\text{DecryptNode}(\text{CT}, \text{SK}_u, t, i)$ and store output in F_z .
3. For each node $z \in B_t$ such that $i' \in S_z$ and $i' \neq i$ call $\text{DecryptNode}(\text{CT}, \text{SK}_u, t, i')$ store output in F'_z . If $i \neq 0$ then translate F'_z to F_z as follows:

$$\begin{aligned} F_z &= e(\hat{C}_z, E_i/E_{i'}) \cdot F'_z \\ &= e(g^{\beta_2 \cdot q_z(0)}, g^{\frac{r_i^{\{u\}} - r_{i'}^{\{u\}}}{\beta_2}}) \cdot e(g, g)^{r_{i'}^{\{u\}} \cdot q_z(0)} = e(g, g)^{r_i^{\{u\}} \cdot q_z(0)} \end{aligned}$$

Otherwise, translate F'_z to F_z as follows:

$$F_z = \frac{e(\hat{C}_z, E_{i'})}{F'_z} = \frac{e(g^{\beta_2 \cdot q_z(0)}, g^{\frac{r_i^{\{u\}} + r_{i'}^{\{u\}}}{\beta_2}})}{e(g, g)^{r_{i'}^{\{u\}} \cdot q_z(0)}} = e(g, g)^{r_i^{\{u\}} \cdot q_z(0)}$$

4. Compute F_t using polynomial interpolation in the exponent as follows:

$$F_t = \prod_{z \in B_t} F_z^{\Delta_{k, B'_z}(0)}, \quad \text{where } k = \text{index}(z), B'_z = \{\text{index}(z) : z \in B_t\}$$

$$\text{and Lagrange coefficient } \Delta_{i, S}(x) = \prod_{j \in S, j \neq i} \frac{x - j}{i - j}$$

$$= \begin{cases} e(g, g)^{r_i^{\{u\}} \cdot q_t(0)} & \text{when } i \neq 0 \\ e(g, g)^{r^{\{u\}} \cdot q_t(0)} & \text{when } i = 0 \end{cases}$$

The output of $\text{DecryptNode}(\text{CT}, \text{SK}_u, r, i)$ function on the root node r is stored in F_r . If $i = 0$ we have $F_r = e(g, g)^{r^{\{u\}} \cdot q_r(0)} = e(g, g)^{r^{\{u\}} \cdot s}$ otherwise we have $F_r = e(g, g)^{r_i^{\{u\}} \cdot s}$. If $i \neq 0$ then we compute F as follows:

$$F = \frac{e(\hat{C}_r, E_i)}{F_r} = \frac{e(g^{\beta_2 \cdot q_r(0)}, g^{\frac{r^{\{u\}} + r_i^{\{u\}}}{\beta_2}})}{e(g, g)^{r_i^{\{u\}} \cdot q_r(0)}} = e(g, g)^{r^{\{u\}} \cdot q_r(0)} = e(g, g)^{r^{\{u\}} \cdot s}$$

Otherwise $F = F_r$. The decryption algorithm then computes following:

$$\frac{\tilde{C} \cdot F}{e(C, D)} = \frac{M \cdot e(g, g)^{\alpha \cdot s} \cdot e(g, g)^{r^{\{u\}} \cdot s}}{e(g^{s \cdot \beta_1}, g^{\frac{(r^{\{u\}} + \alpha)}{\beta_1}})} = M$$

Note how two elements E_i and $E_{i'}$ together with a translating value \hat{C}_t at a node t were used to translate between sets i and i' at node t in step 3. Similarly, note how a single element E_i together with a translating value was used to translate between set i and the outer set. We note that if $\beta_1 = \beta_2$ then the scheme would become insecure as colluding users could transitively translate from inner set i to outer set and then from one key to the other by using the D elements from their keys. Thus we need a unique β for every level that we need to support. When using key structures of depth d , translating values, \hat{C} s, that help translate between sets

at depth d or between a set at depth d and its parent at depth $d - 1$ will use β_d . And to allow translations across multiple levels at a given node, multiple translating values using different β s will need to be released at that node.

Usage Example. We now demonstrate the usage of CP-ASBE with the example policy from Section 3.1.1. When using two level key structures, the policy can be written as follows using threshold gates:

$$4 \text{ OF } 4 \left((Course > 300), (Course < 400), (Grade > 2), (Year > 2007) \right)$$

Here, predicates such as $Course > 300$ will further be expanded and written using their constituent boolean attributes. Recall that numerical attributes in CP-ASBE are represented using a bag of bits representation, with a boolean attribute used to represent each bit of the numerical value, as described in Section 3.1.2. Users can be given keys with two levels. For example, for a user who has taken two courses the structure of the issued key is as follows:

$$\left\{ \{ Course = 304, Grade = 2, Year = 2007 \}, \{ Course = 425, Grade = 3, Year = 2008 \} \right\}$$

While the user's key will contain translation elements E_i 's, as long as there is no designated translation node in the policy (*i.e.*, ciphertext) the user will not be able to combine his $Grade$ and $Year$ attributes for $Course$ 425 with that of $Course$ 304 to satisfy the above policy.

3.4 Security

The security proof for our scheme closely follows that of BSW CP-ABE [10] and uses generic group [17, 66] and random oracle models [9]. Such a proof implies

that the advantage of any adversary in the CP-ASBE security game is negligible, as long as it uses the underlying groups and hash functions in a generic manner, and makes only polynomially many accesses to them.

Generic Bilinear Group [17]. A generic group \mathbb{G}_0 with a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ can be modeled by an oracle which uses random strings as handles for the elements in the two groups \mathbb{G}_0 and \mathbb{G}_1 .³ More precisely, we consider an oracle \mathcal{O} , which picks two random encodings of the additive group \mathbb{F}_p into sufficiently long strings, *i.e.*, injective maps $\psi_0, \psi_1 : \mathbb{F}_p \rightarrow \{0, 1\}^m$, where $m > 3 \log(p)$. We write $\mathbb{G}_0 = \{\psi_0(x) | x \in \mathbb{F}_p\}$ and $\mathbb{G}_1 = \{\psi_1(x) | x \in \mathbb{F}_p\}$. The oracle provides access to the group operations (which we shall refer to as multiplication) in either group: for example, queries of the form $(\text{multiply}_0, h, h')$ and $(\text{inverse}_0, h)$, will be answered respectively by $\psi_0(\psi_0^{-1}(h) + \psi_0^{-1}(h'))$, $\psi_0(-\psi_0^{-1}(h))$. If h or h' is not in the range of ψ_0 , then the oracle returns \perp . The oracle also provides access to the identity elements $(\psi_0(0), \psi_1(0))$, and canonical generators $(\psi_0(1), \psi_1(1))$ in the two groups, as well as the ability to sample random elements in the groups. In addition, given a query (pair, h, h') , where $h = \psi_0(\alpha)$ and $h' = \psi_0(\beta)$, \mathcal{O} returns $h'' = \psi_1(\alpha\beta)$. To relate to the notation of bilinear groups used in our construction, we will denote $\psi_0(1)$ by g and $\psi_0(x)$ by g^x . Similarly we will let $e(g, g)^y$ denote $\psi_1(y)$. Then the above pairing query to the oracle will be written as $e(g^\alpha, g^\beta)$ and the response as

³We remark that it is not important to model the handles as *random* strings, but only as distinct handles that can be named by the adversary. But we stick to the convention from [17], that was used in [10], whose proof ours most closely resemble.

$$e(g, g)^{\alpha\beta}.$$

Finally, the oracle \mathcal{O} also includes a random function $H : \{0, 1\}^* \rightarrow \mathbb{G}_0$. It takes queries of the form (hash, a) for arbitrarily long strings a and returns $H(a)$.

Theorem 3.1. *Let \mathcal{O} , \mathbb{G}_0 , \mathbb{G}_1 , and H be as defined above. For any adversary \mathcal{A} with access to \mathcal{O} in the security game for the CP-ASBE scheme in Section 3.3 (using \mathbb{G}_0 , \mathbb{G}_1 , and H), suppose q is an upper-bound on the total number of group elements it receives from queries to \mathcal{O} and interaction with the CP-ASBE security game. Then the advantage of \mathcal{A} in the CP-ASBE security game is $O(q^2/p)$.*

Proof Intuition. Let us say that s is the random secret split according to the access structure \mathcal{T} as described in the **Encrypt** function of Section 3.3. Let \mathcal{T}' be an access structure derived from \mathcal{T} by removing the sub-trees under all translating nodes, *i.e.*, translating nodes become leaf nodes. For simplicity, let us assume for now that all the leaves of \mathcal{T}' are translating nodes in the original access structure \mathcal{T} . Let $q_t(0)$ represent the secret share associated with a translating node t . A user has to obtain $e(g, g)^{\alpha s}$ to recover the message encrypted using the access structure \mathcal{T} . He could pair $C = g^{\beta_1 s}$ given in the ciphertext with $D = g^{(\alpha + r^{\{u\}})/\beta_1}$ in his key to obtain $e(g, g)^{\alpha s + r^{\{u\}} s}$, *i.e.*, $e(g, g)^{\alpha s}$ blinded by $e(g, g)^{r^{\{u\}} s}$. A user can cancel out $e(g, g)^{r^{\{u\}} s}$ only if he satisfies the tree, *i.e.*, by obtaining a set of $e(g, g)^{r^{\{u\}} q_t(0)}$ that can reconstruct $e(g, g)^{r^{\{u\}} s}$. One can think of the key components given for each set of attributes in the key structure as a unique key under the BSW scheme. That is, if $r^{\{u\}}$ is the unique random number used in our CP-ASBE key then the set of key components (including the translation element) corresponding to each set A_i can be

thought of as a BSW key issued using a master secret key $(\beta_2, g^{r^{\{u\}}})$. Furthermore, each of the sub-trees rooted at a translating node can be thought of an access structure under the BSW scheme. Thus a given sub-tree can only be satisfied using attributes from a single set, *i.e.* a single BSW key, as BSW is collusion resistant. The proof below shows that the additional group elements that are available to an adversary in our scheme do not adversely affect this collusion resistance. Thus a user who has a key with a set that can satisfy the sub-tree under a translating node t can obtain $e(g, g)^{r^{\{u\}}q_t(0)}$. And since $r^{\{u\}}$ is unique to a CP-ASBE key, only attributes from sets within a single CP-ASBE key can be used to satisfy \mathcal{T}' and thus the original access structure.

Proof. Recall that in the CP-ASBE security game the adversary has to distinguish between challenge ciphertexts $M_0 \cdot e(g, g)^{\alpha \cdot s}$ and $M_1 \cdot e(g, g)^{\alpha \cdot s}$. By a standard hybrid argument one can consider a modified security game in which the adversary has to distinguish between challenge ciphertexts $e(g, g)^{\alpha \cdot s}$ and $e(g, g)^\theta$, where θ is selected uniformly at random from \mathbb{F}_p . It is easy to show that any adversary that has advantage ϵ in the original CP-ASBE game can be transformed into an adversary that has an advantage of at least $\epsilon/2$ in the modified game. We will now bound the adversary's advantage in the modified game to prove the theorem.

For this we describe a simulation of the modified game, such that the adversary's view in the simulation is distributed identical to that in the modified security game with challenge ciphertext $e(g, g)^\theta$. Further, conditioned on an event of probability $1 - O(q^2/p)$ in the simulation, we will show that this view is identical to

what it would have been in the modified security game, with challenge ciphertext $e(g, g)^{\alpha \cdot s}$. Thus we will conclude that the advantage of the adversary is at most $O(q^2/p)$.

At the setup time the simulation chooses α, β_1, β_2 at random from \mathbb{F}_p . If $\beta_1 = 0$ or $\beta_2 = 0$ then setup is aborted just as it would be in the actual scheme. The public parameters $h_1 = g^{\beta_1}, h_2 = g^{\beta_2}, f_1 = g^{\frac{1}{\beta_1}}, f_2 = g^{\frac{1}{\beta_2}}$, and $e(g, g)^\alpha$ are sent to the adversary.

When the adversary or the simulation call for evaluation of H on any new attribute string a , a new random value t_a is chosen from \mathbb{F}_p , and the simulation provides g^{t_a} as the response to $H(a)$ and stores it to respond to future queries on a . When the adversary makes its k 'th key generation query for attribute set \mathbb{A}^k , the simulation picks, (1) a new random value $r^{\{k\}}$ from \mathbb{F}_p , (2) a new random value $r_i^{\{k\}}$ for every subset $A_i^k \in \mathbb{A}^k$ and (3) a new random value $r_{i,j}^{\{k\}}$ for every $a_{i,j} \in A_i^k$, for every $A_i^k \in \mathbb{A}^k$. It then computes, (1) $D = g^{(\alpha + r^{\{k\}})/\beta_1}$, (2) $D_{i,j} = g^{r_i^{\{k\}} + t_j \cdot r_{i,j}^{\{k\}}}$ and $D'_{i,j} = g^{r_{i,j}^{\{k\}}}$ for each $j \in A_i^k$ and each $A_i^k \in \mathbb{A}^k$ and (3) $E_i = g^{(r^{\{k\}} + r_i^{\{k\}})/\beta_2}$ for each $A_i^k \in \mathbb{A}^k$. These values are passed on to the adversary.

In the challenge phase the adversary outputs a challenge access structure \mathcal{T}^* along with two equal length messages $M_0, M_1 \in \mathbb{G}_0$. Let \mathbb{X}^* denote the set of *converging nodes* in \mathcal{T}^* . Let \mathbb{Y}^* denote the set of leaf nodes in the access tree \mathcal{T}^* . The simulator chooses a random $s \in \mathbb{F}_p$ and uses the linear secret sharing scheme associated with \mathcal{T}^* as described in **Encrypt** function of Section 3.3 to construct shares $q_y(0)$ of f for $\forall y \in \mathbb{Y}^*$. Let δ_x represent $q_x(0)$ for all $x \in \mathbb{X}^*$, where q_x is polynomial associated with node x as described in **Encrypt** function of Section 3.3.

Note that every leaf node is a descendent of a unique *converging node*. Let $\lambda_{x,j}$ represent $q_y(0)$ where $y \in \mathbb{Y}^*$ is a descendent of $x \in \mathbb{X}^*$ and represents attribute j^4 . Furthermore, note that the choice of δ_x can be perfectly simulated by choosing v random values μ_1, \dots, μ_v uniformly and independently from \mathbb{F}_p , for some value of v , and then letting δ_x be fixed public linear combination of μ_v 's and s . Similarly, we can think of $\lambda_{x,j}$'s as a fixed linear combination of some constants μ'_1, \dots, μ'_u and δ_x . We will think of the δ_x 's and $\lambda_{x,j}$'s as such linear combinations later.

Finally the simulation chooses a random $\theta \in \mathbb{F}_p$, and constructs the encryption as follows: $\tilde{C} = e(g, g)^\theta$, $C = h_1^s$, $\forall y \in \mathbb{Y}^*, C_y = g^{\lambda_{x,j}}$ and $C'_y = g^{t_j \cdot \lambda_{x,j}}$, and $\forall x \in \mathbb{X}^* \hat{C}_x = h_2^{\delta_x}$. These values are sent to the adversary.

When the adversary makes a query to the group oracles, if the adversary provides as input a handle h that it did not receive from the oracle, then with probability $1 - O(1/p^2)$ such a handle is not in the range of ψ_0 or ψ_1 . So, by conditioning on an event of probability $1 - O(q/p^2)$ (q being an upperbound on the number of oracle queries made during the entire simulation), we can assume that the oracle provides answers to only queries which use handles already given out by the oracle. As such we may keep track of the algebraic expressions being called for from the oracles as long as no “accidental collisions” occur. Specifically, we think of an oracle query being a rational function $\nu = \eta/\xi$ in the variables

⁴Note we are assuming here that a given attribute is not represented by multiple leaf nodes descending from the same *converging node*. We can accommodate such policies by adding one more variable subscript to λ that identifies its position among descendents of a given *converging node*. We omit it here for clarity.

$\theta, \alpha, \beta_1, \beta_2, t_j' s, r_i^{k'} s, r_{i,j}^{k'} s, s,$ and $\mu_v' s$. An accidental collision is said to occur if two distinct formal queries $\eta/\xi \neq \eta'/\xi'$ have the same value due to random choices of the variables. We now condition that no such “accidental collisions” in either group \mathbb{G}_0 or \mathbb{G}_1 . For any pair of distinct formal queries η/ξ and η'/ξ' within a group, an accidental collision occurs only if the non-zero polynomial $\eta\xi' - \xi\eta'$ evaluates to zero. The total degree of $\eta\xi' - \xi\eta'$ in our case is a constant (at most 5). Then by the Schwartz-Zippel lemma [64, 77] the probability of an accidental collision between any pair of formal queries is $O(1/p)$. By a union bound, the probability that any such collision happens in our simulation is at most $O(q^2/p)$. Thus we can condition on no “accidental collisions” occurring while retaining $1 - O(q^2/p)$ of the probability mass.

Now we show that, subject to the condition that no “accidental collisions” occur, the view of the adversary is identically distributed when we set $\theta = \alpha \cdot s$ in the simulation. Since we are in the generic group model where each group element is uniformly and independently chosen the only way that the adversary’s view can differ in the case $\theta = \alpha \cdot s$ is if there are two queries ν and ν' into \mathbb{G}_1 such that $\nu \neq \nu'$ but $\nu|_{\theta=\alpha \cdot s} = \nu'|_{\theta=\alpha \cdot s}$. Since θ only occurs as $e(g, g)^\theta$ in \mathbb{G}_1 , the only dependence ν or ν' can have on θ is by having some additive terms of the form $\gamma \cdot \theta$ where γ is a constant. Therefore we must have that $\nu - \nu' = \gamma \cdot \alpha \cdot s - \gamma \cdot \theta$ for some constant $\gamma \neq 0$ for the adversary’s view to be different in the two simulations. We can then artificially add the query $\nu - \nu' + \gamma \cdot \theta = \gamma \cdot \alpha \cdot s$ to the adversary’s queries. We will now do a case analysis based on the information given to the adversary by the simulation to show that an adversary can never construct a query for $e(g, g)^{\gamma \cdot \alpha \cdot s}$

Table 3.1: Possible adversary query terms in \mathbb{G}_1

$r_{i,j}^{\{k\}}$	t_j	$r_i^{\{k\}} + t_j r_{i,j}^{\{k\}}$	$t_j \lambda_{x,j}$
$r_{i,j}^{\{k\}} r_{i',j'}^{\{k'\}}$	$t_j t_{j'}$	$\lambda_{x,j} \lambda_{x',j'}$	$t_j t_{j'} \lambda_{x,j} \lambda_{x',j'}$
$r_{i,j}^{\{k\}} t_{j'}$	$t_j \lambda_{x,j'}$	$\lambda_{x,j} \lambda_{x',j'} t_{j'}$	$\lambda_{x,j} t_j r_i^{\{k\}} + \lambda_{x,j} t_j t_{j'} r_{i,j'}^{\{k\}}$
$r_{i,j}^{\{k\}} \lambda_{x,j'}$	$t_j t_{j'} \lambda_{x,j'}$	$r_i^{\{k\}} \lambda_{x,j'} + t_j r_{i,j}^{\{k\}} \lambda_{x,j'}$	$r_{i,j}^{\{k\}} r_{i'}^{\{k'\}} + r_{i,j}^{\{k\}} t_{j'} r_{i',j'}^{\{k'\}}$
$\alpha + r^k$	δ_x	$r_i^{\{k\}} t_{j'} + t_j t_{j'} r_{i,j}^{\{k\}}$	$r_{i,j}^{\{k\}} t_{j'} \lambda_{x,j'}$
s	$r^k + r_i^{\{k\}}$	$\alpha s + r^k s$	$r^k \delta_x + r_i^{\{k\}} \delta_x$
$\lambda_{x,j}$	$(r_i^{\{k\}} + t_j r_{i,j}^{\{k\}})(r_{i'}^{\{k'\}} + t_{j'} r_{i',j'}^{\{k'\}})$		

which will establish the theorem.

Table 3.1 enumerates all the possible queries into \mathcal{G}_1 by means of the bilinear map and group elements given to the adversary by the simulation except for those that involve β_1 or β_2 in every monomial as they will not be relevant for constructing a query involving $\alpha \cdot s$. Here the variables j and j' are possible attribute strings, variables k and k' are indices of secret key queries made by the adversary, and variables i and i' are indices of attribute subsets in a given secret key. The queries are given in terms of $\lambda_{x,j}'s$ and $\delta_x's$ and not in terms of $\mu_v's$ and $\mu_u's$. The adversary has access to 1 and α in group \mathbb{G}_1 in addition to the queries shown in Table 3.1. The adversary can query for an arbitrary linear combination of these and we will show that no such linear combination can produce a polynomial of the form $\gamma \cdot \alpha \cdot s$ for some constant $\gamma \neq 0$.

From Table 3.1 we see that the only way for an adversary to create a term containing αs is by pairing $s\beta_1$ with $(\alpha + r^{\{k\}})/\beta_1$ to get $\alpha s + r^k s$. The adversary could create a query polynomial of the form $\gamma\alpha s + \sum_{k \in T} \gamma_k r^k s$, for some set T and constants $\gamma, \gamma_k \neq 0$. In order for the adversary to get a query polynomial of the form $\gamma\alpha s$ the adversary must add other terms in order to cancel the terms of $\sum_{k \in T} \gamma_k r^k s$. Observe that the only terms of the form $r^k s$ that the adversary has access to are obtained by pairing $\beta_2 \delta_x$ terms with $(r^k + r_i^{\{k\}})/\beta_2$ terms, since δ_x 's are linear combinations of μ_v 's and s . The adversary could create a query polynomial of the form for sets T'_k and constants $\gamma_{(i,k,x)} \neq 0$:

$$\gamma\alpha s + \sum_{k \in T} \left(\gamma_k r^{\{k\}} s + \sum_{(i,x) \in T'_k} \gamma_{(i,k,x)} (r^{\{k\}} \delta_x + r_i^{\{k\}} \delta_x) \right)$$

By design there exists at least one non empty set of δ_x 's that can reconstruct s . Without loss of generality we will assume that $\forall k \in T$ the adversary picks a set T'_k such that the set $\{\delta_x | (i,x) \in T'_k\}$ can reconstruct s . (For otherwise the adversary's polynomial cannot be of the form $\gamma\alpha s$ thus proving the theorem.) The adversary still needs to add other terms to cancel terms of the form $\sum_{(i,x) \in T'_k} \gamma_{(i,k,x)} r_i^{\{k\}} \delta_x, \forall k \in T$ in order to obtain a query polynomial of the form $\gamma\alpha s$. Note that the only other terms of the form $r_i^{\{k\}} \delta_x$ that the adversary has access to are obtained by pairing $r_i^{\{k\}} + t_j r_{i,j}^{\{k\}}$ with some $\lambda_{x,j'}$ as $\lambda_{x,j'}$ is a linear combinations of μ_u 's and δ_x . Thus the adversary could create a query polynomial of the following form:

$$\gamma\alpha s + \sum_{k \in T} \left(\gamma_k r^{\{k\}} s + \sum_{(i,x) \in T'_k} \gamma_{(k,i,x)} (r^{\{k\}} \delta_x) + \sum_{(i,x) \in T'_k} \gamma_{(k,i,x)} \left(r_i^{\{k\}} \delta_x + \sum_{(j,j') \in T''_{k,i,x}} \gamma_{(k,i,x,j,j')} (r_i^{\{k\}} \lambda_{x,j'} + t_j r_{i,j}^{\{k\}} \lambda_{x,j'}) \right) \right)$$

The following case analysis concludes the proof:

Case 1: There exists some $k \in T : \exists(i, x) \in T'_k$ such that the set of shares $L_{k,i,x} = \{\lambda_{x,j'} : \exists j : (j, j') \in T''_{k,i,x}\}$ do not allow for the reconstruction of δ_x . In this case, the term $r_i^{\{k\}}\delta_x$ will not be canceled and the adversary's query polynomial cannot be of the form $\gamma\alpha s$.

Case 2: For all $k \in T$ and $\forall(i, x) \in T'_k$ the set of shares $L_{k,i,x} = \{\lambda_{x,j'} : \exists j : (j, j') \in T''_{k,i,x}\}$ allow for the reconstruction of δ_x . Then the only terms left in the adversary's query polynomial other than $\gamma\alpha s$ are of the form $t_j r_{i,j}^{\{k\}} \lambda_{x,j'}$ and the adversary needs to add other terms to cancel them from the query. As seen in Table 3.1, the only term the adversary has access to that he can use to cancel terms of the form $t_j r_{i,j}^{\{k\}} \lambda_{x,j'}$ is the term $r_{i,j}^{\{k\}} t_{j'} \lambda_{x,j'}$ but only when $j = j'$. We will now show that there is at least one term of the form $t_j r_{i,j}^{\{k\}} \lambda_{x,j'}$ in the adversary's query polynomial such that $j \neq j'$ to complete the proof.

Fix any $k \in T$. Consider \mathbb{A}^k the set of attributes corresponding to k 'th adversary key. By the assumption that no key should pass the challenge access structure, and the properties of the secret sharing scheme we know that there exists a $\delta_x : \exists i : (i, x) \in T'_k$ such that the set of shares $L'_{k,i,x} = \{\lambda_{x,j} : j \in A_i^k : A_i^k \in \mathbb{A}^k\}$ cannot reconstruct δ_x for any $i : A_i^k \in \mathbb{A}^k$. Thus there must exist at least one $\lambda_{x,j'} \in L_{k,i,x}$ that is linearly independent of $L'_{k,i,x}$ when written in terms of δ_x and μ_v 's. Thus for at least for one $\lambda_{x,j'} \in L_{k,i,x}$ there will be a term of the form $t_j r_{i,j}^{\{k\}} \lambda_{x,j'} : j \neq j'$ left behind in the query for the adversary does not have access to a term that can cancel it as evident from Table 3.1.

Therefore no adversary query polynomial can be of the form $\gamma\alpha s$.

□

3.5 Efficiency Analysis

In this section we discuss the efficiency of CP-ASBE scheme instantiated with two-levels. It is straightforward to estimate the efficiency of our key generation and encryption algorithms. In terms of computation, our key generation algorithm requires two exponentiations for every attribute in the key issued to the user and two exponentiations for every set (including recursive sets for a scheme with levels > 2) in the key. In terms of key size, the private key contains two group elements per attribute and one group element per attribute set. Compared to BSW the additional key generation cost is two exponentiations for every attribute set in terms of computation and one group element per attribute set in terms of size. Encryption involves two exponentiations per leaf node in the tree and one exponentiation per translating node in the tree. The ciphertext contains two group elements per leaf node and one group element per translating node. Compared to BSW the additional cost is one exponentiation per translating node in terms of computation and one group element per translating node in terms of size. The cost of decrypting a given ciphertext however varies depending on the key used for decryption. Even for a given key there might be multiple ways to satisfy the associated access tree. The decrypt algorithm needs, 1) two pairings for every leaf node used to satisfy the tree, 2) one pairing for every translating node on the path from the leaf node used to the

root and 3) one exponentiation for every node on the path from the leaf node to the root. However, by employing the optimization technique of flattening the recursive calls to *DecryptNode*, as described in BSW [10] albeit modified to accommodate translating nodes, we can reduce the cost to 1) two pairings and one exponentiation per leaf node used and 2) one pairing and one exponentiation per translating node on the path from a used leaf node to the root. Compared to BSW the additional cost is one pairing and one exponentiation per translating node on the path from a used leaf node to the root. In a multi-level (level > 2) instantiation the overhead will be per translation rather than per translating node as multiple translations may be needed at a given translating node for such instantiations.

3.6 Implementation

We have implemented a two-level CP-ASBE scheme with an optimized decryption function. Our implementation leverages the *cpabe* toolkit (<http://acsc.cs1.sri.com/cpabe/>) developed for BSW which uses the Pairing-Based Cryptography library (<http://crypto.stanford.edu/abc/>). The interface for the *cpasbe* toolkit is similar to that of *cpabe* toolkit and is as follows:

cpasbe-setup Generates a public key and a master key.

cpasbe-keygen Given a master key, generates a private key for a given set of attributes; compiles numerical attributes into 'bag of bits' representation and treats the resulting attributes as a 'set'.

cpasbe-enc Given a public key, encrypts a file under a given access policy; numerical comparisons in the policy are represented by access sub-trees comprising 'bag of bits' representation of the numerical attribute with the root node of the sub-tree treated as a translating node.

cpasbe-dec Decrypts a file, given a private key.

The *cpasbe* toolkit is similar to *cpabe* toolkit in that it supports numerical attributes and range queries (*i.e.*, numerical comparisons) in access policies. However, unlike in *cpabe* toolkit, numerical attributes in *cpasbe* are treated as sets and thus *cpasbe* toolkit supports multiple numerical value assignments to a given attribute in a single private key. Thus a user with a private key generated using the following command cannot claim any score other than 33 and 30.

```
$ cpasbe-keygen -o tom-priv-key pub-key master-key 'score=33' 'score=30' tom
```

An initial performance evaluation of two-level CP-ASBE using this implementation is discussed in Section 5.1.3.

Chapter 4

Data Sharing Based on Private, Context Sensitive Policies

In many scenarios, data sharing is based on policies that include contextual information. The scenario we consider is data sharing in the electric power grid where power system operators need to cooperate with each other to operate the grid safely and reliably but they also compete with each other as business entities. Increasing power consumption and major recent events such as the August 2003 blackout [73] means the system operators are compelled to share sensitive data to improve the reliability of the grid through wide area measurement, monitoring and control. In deregulated grids worldwide and in the North American grid in particular, utilities share sensitive data with their local Reliability Coordinators (RCs) as required by regulatory laws. However, they might not be comfortable disclosing sensitive data to other entities except under certain conditions including transient conditions in the grid at the time of access. For example, Utility A might be willing to share certain data, 1) with some utilities right away while with others only after four hours have elapsed since the data is generated or 2) with any Utility X under the jurisdiction of RC B during a frequency or voltage disturbance. In many cases it is the context-based policy that drives the data sharing while the number or recipients or their identities may not be known in advance. Interestingly, it is not just the data that is sensitive but also the policies for sharing the data.

For example, if the second policy rule in the example above involving the context of a major transmission disturbance were to be in clear-text then anyone observing significant network traffic with that policy might be able to conclude that a major event has occurred. This could result in negative publicity, loss of market revenue or an increase in attacks for Utility A. In general, policies may be sensitive because they directly contain sensitive information, reveal information about underlying data protected by the policy or reveal information about the data owner or the data recipients.

An effective approach for addressing requirements for the power grid data sharing problem requires techniques that go beyond the capabilities of today's solutions in the area. Specifically, there is a need for policy-based data sharing techniques that support 1) multiple recipients, 2) data and policy secrecy and 3) context-based policy enforcement. Furthermore, in order to be practical, techniques with these properties must be efficient (in terms of key management), support flexible policy specifications, be secure in the presence of active adversaries, and be compatible with existing distributed networking and systems technologies. Past work in this area has addressed only a subset of these problems. Identity Based Encryption (IBE) [16] systems and policy-based cryptographic schemes proposed in [2, 8] allow the association of a flexible policy with objects and support exchange in open distributed systems but do not keep the policy secret and are designed for two-party communication where the sender identifies the recipient in the encryption. Several works in the area of "hidden policies and credentials" [18, 33, 48] provide message and policy secrecy but focus on two-party interactions. Attribute Based Encryption

(ABE) systems such as Ciphertext-Policy ABE (CP-ABE) [10, 24, 36, 49] including our CP-ASBE scheme and cryptographic file system FSGuard [69] allow the association of flexible policies with objects for multiple recipients defined by those policies and support exchange in open distributed systems but do not provide policy secrecy and cannot support context-based policies. Recent work by [55] extends CP-ABE to support policy secrecy but significantly limits its policy flexibility and does not support context-based policies. Predicate Encryption scheme proposed in [43] also allows the association of flexible policies with objects for multiple recipients defined by those policies, supports exchange in open distributed systems and provides policy secrecy but does not support context-based policies. PEAPOD [42] focuses on one-to-many messaging with both message and policy secrecy but does not provide efficient key management and is also vulnerable to collusion attacks. Policy-based cryptography scheme in [7] is also vulnerable to collusion attacks.

In this chapter we develop an application-independent Policy Based Encryption System (PBES) that proposes a solution to this new problem of providing all of the above-mentioned properties. We first build a new encryption scheme PKEM-DEM (Policy and Key Encapsulation Mechanism - Data Encapsulation Mechanism) for encrypting objects and policies and show that it is secure against adaptive chosen ciphertext attacks in the random oracle model. The encryption scheme builds on recent work in KEM-DEM hybrid encryption schemes [27]. In addition to the notions of message indistinguishability and policy indistinguishability we define and prove a new notion of pairwise indistinguishability where adversaries need to distinguish

between pairs of messages and policies¹. We then use this scheme to construct the PBES system that provides the three properties mentioned above. For decryption PBES utilizes trusted Key Distribution Centers (KDC)s that mediate decryption of objects for recipients and enforce the policies associated with the objects. We leverage the KDCs for policy enforcement and provide very efficient key management as well as immediate revocation. We then discuss design issues for developing applications using PBES; e.g., key distribution and placement of trust in KDCs.

PBES employs trusted key servers and from a systems perspective this approach is reasonable for regulated environments such as the power grid; in fact, the grid regularly uses trusted servers for ensuring reliability and security. In terms of encryption techniques this design approach first made it seem like the solution might be easy, however, it turned out that was not the case. We looked at leading Public Key Infrastructure (PKI), Role-Based Access Control (RBAC) and secure publish/subscribe systems that typically employ trusted servers for mediated access control but were unable to satisfy the requirements. Specifically, the requirements for policy secrecy and context-based policy enforcement could not be satisfied. PBES satisfies these requirements and also provides efficiency, security and flexibility. While the scheme is motivated by the data sharing problem in the power grid, PBES is suitable for many large-scale systems that share features with the power grid. Regulated environments such as medical and financial information systems often employ trusted mediators that share environmental features like the power grid; examples of trusted entities include Centers for Disease Control and Prevention

¹A similar notion is independently defined and used by [55].

(CDC) in the public health domain and the Securities and Exchange Commission in the financial domain. Even outside regulated domains suitable application domains include those where domains have partial trust or provide auditing capabilities of the services provided by the trusted servers.

The rest of this chapter is organized as follows. In Section 4.1 we presents our approach. In Section 4.2 we present the notation used and present security notions. In Section 4.3 we present the building blocks used in our system. In Section 4.4 we present our policy based encryption system and analyze its security in Section 4.5. In Section 4.6 we discuss application design issues when using PBES.

4.1 Approach

4.1.1 Related Approaches

An ideal solution for the data sharing problem in the power grid would be one that does not require trusted servers to enforce the policy. However existing techniques that enforce the policy cryptographically and provide policy secrecy like CP-ABE [55, 43] are not adequate as they cannot support flexible context based policies. Furthermore, the power grid data sharing application and its properties discussed above indicate that the presence of a Trusted Third Party (TTP) that enforces access control is acceptable and perhaps even needed. The RCs and Independent System Operators (ISOs) regularly mediate power flow and markets to keep the system stable and provide a means for establishing TTPs for access control. With a TTP the problem of developing an appropriate policy-based data sharing

solution appears within reach at first in that it can leverage many existing tools and technologies already developed in the area. However, it turns out that none of these leading technologies can satisfy the requirements above. In particular, they are unable to efficiently and securely provide policy secrecy and flexible context-based policy enforcement. To show this we evaluate the suitability of Public Key Infrastructure tools, Role-Based Access Control systems, and secure publish-subscribe event dissemination systems and discuss their shortcomings.

PKI, RBAC and context-based policy enforcement . Public Key Infrastructure (PKI) tools with identity and attribute certificates provide data sharing between parties with the help of trusted certificate authorities. One can design policy-based data sharing solutions where a combination of attributes in attribute certificates are used to specify the policy. Unfortunately, such solutions would be vulnerable to collusion and would also fail to provide policy secrecy. RBAC systems take PKI one step forward by providing a level of indirection between users and permissions. They achieve this by assigning users to roles via role membership certificates and roles to permissions for access control. This indirection has been utilized by several RBAC solutions such as OASIS [5] to provide context-based policy enforcement whereby users can “activate” their roles and execute operations based on the assumed role permissions only if certain context/environment policies (as verified by trusted access control servers) are satisfied. If we attempt to extend such solutions to address the requirements specified above we would face two limitations. First, in order to ensure policy secrecy, data generators would have to specify poli-

cies at every access control server over secure channels for every data distribution action. Second, specifying multi-domain contexts for policy enforcement may impose impractical constraints on role activation because users may need special roles dedicated to this multi-domain data sharing application.

Secure Publish Subscribe Systems . Pub/sub systems are related to policy-based data sharing systems discussed in this work in that publishers and subscribers relate to data generators and consumers, and brokers in the pub/sub infrastructure relate to servers enforcing access control policies. Research in secure pub/sub systems, in general, and those that provide content encryption, in particular, offers potential solutions to the problem at hand. In essence techniques for encrypted content distribution via pub/sub systems use symmetric keys to encrypt events with selective attributes and then employ fully or partially trusted key servers to distribute those keys to subscribers based on their subscriptions. To allow routing for encrypted content these schemes may share keys with routers [4] expose certain attributes in clear-text for routing purposes, or use encryption-matching functions [70]. Solutions such as [4] carry over limitations of RBAC systems identified above. If we attempt to use a secure pub/sub solution like [70] for our application we again face limitations. First, ensuring policy secrecy for a flexible policy language requires publishers and subscribers to maintain a large number of keys and requires the system to maintain a significant amount of auxiliary data that allows mapping of policies with keys. Second, the solution uses time epochs for coarse-grained revocation and the system would have to be significantly enhanced to support context-based policies that may

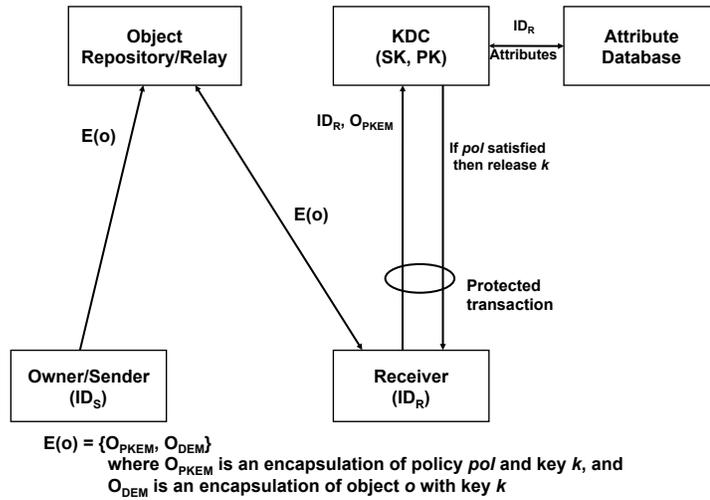


Figure 4.1: Policy-based Message Encryption and Decryption

need ephemeral keys for the various transient events.

4.1.2 Our Approach - PBES

The above analysis is not intended to conclude that these existing technologies cannot be adapted for the problem at hand. Instead, we argue with this analysis that even with TTPs solutions to this problem are not obvious. To address this we have developed the PBES system with a high-level architecture described in Figure 4.1. The approach satisfies the requirements of Section 3 as follows.

The system is illustrated in Figure 4.1 and contains five main components: the data owner/sender, the object repository/relay, the Key Distribution Center (KDC), the attribute database and the data receiver. A data owner in our system specifies a policy pol and generates a data object o (e.g. file) that is intended for one or more recipients satisfying the policy. The sender uses an encryption scheme

to encrypt the object and the policy. The object repository/relay represents any content distribution network, for example, a file server, an email relay or a publish-subscribe system. We assume that the encrypted object contains sufficient metadata to allow for routing/searching of the data for intended/interested recipients but that does not reveal the policy; e.g., keywords. Since the object is encrypted the repository/relay need not be trusted to protect the object or enforce access control on it. Recipients obtain the encrypted object from this repository/relay via available pull/push mechanisms. Once a recipient gets the encrypted object it contacts the KDC to obtain the object decryption key. The KDC may contact an Attribute Database that manages user attributes and privileges and keeps track of environmental attributes. The Attribute Database abstracted here is a logical entity and in practice may be composed of multiple databases/services.

There are key design choices here that affect the efficiency, security, flexibility and compatibility. We require that the object and the policy be encrypted and stored together but that they be separable for decryption purposes. This improves efficiency because on the sender side the sender need not specify the policy at multiple servers (KDCs) that may be trusted with policy enforcement and on the receiver side the receiver need not send the encrypted object (which could be large) to the KDC for policy enforcement and decryption. We *associate* the object and policy with a key rather than generate the key from the policy. This allows for considerable flexibility and compatibility as any policy language may be used; e.g., one that is already used by the application for other purposes. While there are a range of potential languages and tools we believe that tools based on XACML are

a good candidate for PBES. The approach for associating data and policies with keys, however, imposes the need for an encryption scheme that is secure against active adversaries. In the absence of this adversaries may be able to manipulate the encrypted objects and policies stored at the repository in unauthorized ways; e.g., associate a new object with an existing policy or vice-versa. To that end we develop a PKEM-DEM hybrid encryption that provides adequate security for PBES.

4.2 Notation and Security Notions

We first introduce some common notation used throughout this chapter. We then define formal notions of security for a policy-based encryption scheme for multiple recipients with policy secrecy.

Notation Bit strings are denoted using small case letters, x , and the length of such strings is denoted by $|x|$. Sets are denoted using capital case letters, S , and the size of the such sets is denoted by $|S|$. $s \stackrel{\$}{\leftarrow} S$ denotes the operation of picking an element s of S uniformly at random. Adversaries are represented by probabilistic polynomial-time (PPT) algorithms \mathcal{A} . $v \stackrel{\$}{\leftarrow} \mathcal{A}(\alpha_1, \alpha_2, \dots, \alpha_k)$ denotes the action of running the PPT algorithm \mathcal{A} with input $(\alpha_1, \alpha_2, \dots, \alpha_k)$ and letting v be the output. $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \dots, \mathcal{O}_l}(\alpha_1, \alpha_2, \dots, \alpha_k)$ denotes a PPT adversary with input $(\alpha_1, \alpha_2, \dots, \alpha_k)$ and access to oracles $\mathcal{O}_1, \mathcal{O}_2, \dots, \mathcal{O}_l$. Let \mathcal{E} denote a policy-based encryption scheme for multiple recipients with policy secrecy.

Given that we want to protect both message and policy secrecy we define the notions of *message indistinguishability* and *policy indistinguishability* against

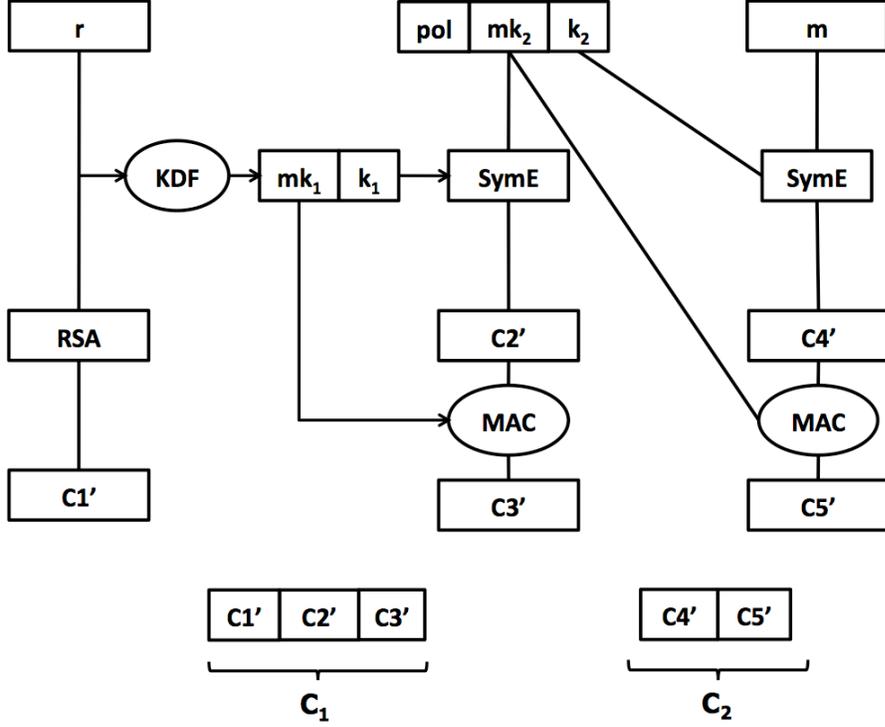


Figure 4.2: Encryption in PKEM-DEM scheme instantiated using RSA-KEM and DEM1

adaptive chosen ciphertext attacks similar to the ones defined in [42].

Definition 4.1. Message Indistinguishability

\mathcal{E} has message indistinguishability against an adaptive chosen ciphertext attack if the guessing advantage, of any PPT adversary, $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, as defined below is negligible.

$$\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-msg-ind-cca2}}(k) = \left| \Pr \left[\mathbf{G}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-msg-ind-cca2}}(k) = b \right] - 1/2 \right|$$

where $\mathbf{G}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-msg-ind-cca2}}(k)$ is the game described below:

Setup The environment generates a key-pair (sk, pk) and gives pk to \mathcal{A} .

Phase 1 \mathcal{A}_1 is provided with a decryption oracle for \mathcal{E} with above generated key-pair. It is also allowed to arbitrarily and adaptively add/corrupt users. That is it can get access to arbitrary sets of attributes represented by corrupted users u_i .

Challenge \mathcal{A}_1 outputs two messages, m_0, m_1 of equal length, a policy p of his choice and some state information St with the following restriction:

Restriction 1: None of the corrupted users satisfy the policy p throughout the game.

The environment then picks a random bit, $b \xleftarrow{\$} \{0, 1\}$, and encrypts message m_b under policy p and returns the challenge ciphertext C^* along with St to \mathcal{A}_2 .

Phase 2 \mathcal{A}_2 is provided with a decryption oracle for \mathcal{E} with above generated key-pair and is allowed to do everything \mathcal{A}_1 is allowed in Phase 1 with the constraint that Restriction 1 must be satisfied and that it cannot query the decryption oracle on C^* .

Output \mathcal{A}_2 outputs his guess $b' \in \{0, 1\}$. \mathcal{A} wins if $b' = b$.

That is, an adversary cannot distinguish between encryptions of two messages under a given policy. Restriction 1 is needed because otherwise the adversary can trivially win the game by decrypting the challenge ciphertext as he has access to keying material of a user who satisfies the policy.

Definition 4.2. Policy Indistinguishability

\mathcal{E} has policy indistinguishability against an adaptive chosen ciphertext attack if the guessing advantage, of any PPT adversary, $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, as defined below is negligible.

$$\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-pol-ind-cca2}}(k) = \left| \Pr \left[\mathbf{G}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-pol-ind-cca2}}(k) = b \right] - 1/2 \right|$$

where $\mathbf{G}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-pol-ind-cca2}}(k)$ is the game described below:

Setup The environment generates a key-pair (sk, pk) and gives pk to \mathcal{A} .

Phase 1 \mathcal{A}_1 is provided with a decryption oracle for \mathcal{E} with above generated key-pair. It is also allowed to arbitrarily and adaptively add/corrupt users. That is it can get access to arbitrary sets of attributes represented by corrupted users u_i .

Challenge \mathcal{A}_1 outputs state information St , a message, m , and two policies p_0 , p_1 of equal length satisfying one of the following restrictions:

Restriction 2a: All of the corrupted users satisfy both policies p_0 and p_1 throughout the game. OR

Restriction 2b: None of the corrupted users satisfy either policy p_0 or policy p_1 throughout the game.

The environment then picks a random bit, $b \xleftarrow{\$} \{0, 1\}$, and encrypts message m under policy p_b and returns the challenge ciphertext (C^*) along with St to \mathcal{A}_2 .

Phase 2 \mathcal{A}_2 is provided with a decryption oracle access for \mathcal{E} and is allowed to do everything \mathcal{A}_1 is allowed in Phase 1 with the constraint that either Restriction 2a or 2b must be satisfied and that it cannot query the decryption oracle on C^* .

Output \mathcal{A}_2 outputs his guess $b' \in \{0, 1\}$. \mathcal{A} wins if $b' = b$.

That is, an adversary cannot distinguish between encryptions of a given message under two policies. Restriction 2a or 2b is needed because otherwise the adversary can trivially win the game by picking two policies such that he (*i.e.*, one of the corrupted users) satisfies one of them and not the other.

We now define a security notion called *pairwise indistinguishability* for pairs $(m_0, pol_0), (m_1, pol_1)$ which is motivated by the following scenario. Let us say an adversary knows that either message “Buy” is encrypted under policy “Aggressive” or message “Sell” is encrypted under policy “Moderate” (where “Aggressive” and “Moderate” are known investor profiles) but doesn’t know which action is being recommended by a paid investment service.

Definition 4.3. Pairwise Indistinguishability

\mathcal{E} has *pairwise indistinguishability* against an adaptive chosen ciphertext attack if the guessing advantage, of any PPT adversary, $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, as defined below is negligible.

$$\text{Adv}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-pw-ind-cca2}}(k) = \left| \Pr \left[\mathbf{G}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-pw-ind-cca2}}(k) = b \right] - 1/2 \right|$$

where $\mathbf{G}_{\mathcal{E}, \mathcal{A}}^{\mathcal{E}\text{-pw-ind-cca2}}(k)$ is the game described below:

Setup The environment generates a key-pair (sk, pk) and gives pk to \mathcal{A} .

Phase 1 \mathcal{A}_1 is provided with a decryption oracle for \mathcal{E} with above generated key-pair. It is also allowed to arbitrarily and adaptively add/corrupt users. That is it can get access to arbitrary sets of attributes represented by corrupted users u_i .

Challenge \mathcal{A} outputs two messages, m_0, m_1 , of equal length and two policies p_0, p_1 , of equal length along with state information St under the following restriction:

Restriction 3: None of the corrupted users satisfy either policy p_0 or p_1 throughout the game.

The environment then picks a random bit, $b \xleftarrow{\$} \{0, 1\}$, and encrypts message m_b under policy p_b and returns the challenge ciphertext (C^*) along with state information St to \mathcal{A}_2 .

Phase 2 \mathcal{A}_2 is provided with a decryption oracle for \mathcal{E} and is allowed to do everything \mathcal{A}_1 is allowed in Phase 1 with the constraints that Restriction 3 must be satisfied and that it cannot query the decryption oracle on C^* .

Output \mathcal{A}_2 outputs his guess $b' \in \{0, 1\}$. \mathcal{A} wins if $b' = b$.

That is, an adversary cannot distinguish between encryptions of two message and policy pairs. Restriction 3 is needed because otherwise the adversary can trivially win the game by decrypting the challenge ciphertext as he has access to keying

material of a user who satisfies the policy. By definition, pairwise indistinguishability implies message indistinguishability (when both policies are the same) and policy indistinguishability with restriction 2b (when both messages are the same) and hence is a stronger notion of security. Furthermore, we note that using standard hybrid argument one can show that message indistinguishability together with policy indistinguishability (with restriction 2b) imply pairwise indistinguishability. In all the above definitions the adversary is allowed to corrupt multiple users and obtain their keying material thus user-collusion attacks are taken into account.

4.3 Building Blocks

Our encryption scheme is based on the KEM-DEM hybrid encryption paradigm. We now introduce some crypto primitives that will be used to build our scheme and define associated security notions.

4.3.1 Key Encapsulation Mechanism

A public-key based *key encapsulation mechanism* (KEM), consists of the following three algorithms: KEM.KeyGen , KEM.Encrypt and KEM.Decrypt . KEM.KeyGen is a PPT algorithm that takes as input a security parameter, $k \in \mathbb{N}$, and outputs a key pair (sk, pk) , *i.e.*, $(sk, pk) \xleftarrow{\$} \text{KEM.KeyGen}(1^k)$. KEM.Encrypt is a PPT algorithm that takes as input a security parameter, $k \in \mathbb{N}$, and a public-key, pk , generated by KEM.KeyGen and outputs a pair (K, C) , where $K \in \{0, 1\}^{\text{KEM.KeyLen}(k)}$ is a key, $\text{KEM.KeyLen}(k)$ is key-length and C is a ciphertext,

i.e., $(K, C) \stackrel{\$}{\leftarrow} KEM.Encrypt(1^k, pk)$. $KEM.Decrypt$ is a deterministic polynomial-time algorithm that takes as input a secret-key, sk , and ciphertext, C , and returns either a key K or a rejection symbol \perp , *i.e.*, $\{K, \perp\} \leftarrow KEM.Decrypt(sk, C)$. For correctness, we require that $\forall k \in \mathbb{N}$, and $\forall (sk, pk) \stackrel{\$}{\leftarrow} KEM.KeyGen(1^k)$ we have $KEM.Decrypt(sk, C) = K$ for any $(K, C) \stackrel{\$}{\leftarrow} KEM.Encrypt(1^k, pk)$.

We define the notion of indistinguishability for KEMs against an adaptive chosen ciphertext attack (CCA2) as established in [27].

Definition 4.4. *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a PPT CCA2 adversary. We define the guessing advantage of \mathcal{A} as follows:*

$$\mathbf{Adv}_{KEM, \mathcal{A}}^{kem-ind-cca2}(k) = |\Pr [\mathbf{G}_{KEM, \mathcal{A}}^{kem-ind-cca2}(k) = b] - 1/2|$$

where

Game $\mathbf{G}_{KEM, \mathcal{A}}^{kem-ind-cca2}(k)$

$$(sk, pk) \stackrel{\$}{\leftarrow} KEM.KeyGen(1^k); St \stackrel{\$}{\leftarrow} \mathcal{A}_1^{DEC(\cdot)}(pk)$$

$$b \stackrel{\$}{\leftarrow} \{0, 1\}; K_0^* \stackrel{\$}{\leftarrow} \{0, 1\}^{KEM.KeyLen(k)}$$

$$(K_1^*, C^*) \stackrel{\$}{\leftarrow} KEM.Encrypt(1^k, pk); K^* \stackrel{\$}{\leftarrow} K_b^*$$

$$b' \stackrel{\$}{\leftarrow} \mathcal{A}_2^{DEC(\cdot)}(pk, C^*, K^*, St); \text{Return } b'$$

and oracle $DEC(\cdot)$ is defined as $KEM.Decrypt(sk, \cdot)$ with the condition that the oracle rejects queries on C^* after the target ciphertext is given to the adversary.

That is, given a ciphertext and key pair an adversary cannot tell whether the given key is the one encapsulated by the ciphertext or a random one.

4.3.2 Data Encapsulation Mechanism

A *data encapsulation mechanism* (DEM) is a symmetric key encryption scheme and consists of the following three algorithms: DEM.KeyGen, DEM.Encrypt and DEM.Decrypt. DEM.KeyGen is a PPT algorithm that takes as input a security parameter, $k \in \mathbb{N}$, and outputs a key K , *i.e.*, $K \stackrel{\$}{\leftarrow} \text{DEM.KeyGen}(1^k)$. DEM.Encrypt is a polynomial-time algorithm that takes as input a message m , and a key, K , generated by DEM.KeyGen and outputs a ciphertext C , *i.e.*, $C \stackrel{\$}{\leftarrow} \text{DEM.Encrypt}(m, K)$. DEM.Decrypt is a deterministic polynomial-time algorithm that takes as input a key, K , and ciphertext, C , and returns either the message m or a rejection symbol \perp , *i.e.*, $\{m, \perp\} \leftarrow \text{DEM.Decrypt}(K, C)$. For correctness, we require that $\forall k \in \mathbb{N}, \forall K \stackrel{\$}{\leftarrow} \text{DEM.KeyGen}(1^k)$ and $\forall m$ we have

$$\text{DEM.Decrypt}(K, \text{DEM.Encrypt}(m, K)) = m$$

We define the notion of indistinguishability for a DEM against a one-time attack (OT) and a one-time adaptive chosen ciphertext attack (OTCCA) as established in [27].

Definition 4.5. *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a PPT adversary. We define the guessing advantage of \mathcal{A} as follows:*

$$\mathbf{Adv}_{\text{DEM}, \mathcal{A}}^{\text{dem-ind-atk}}(k) = \left| \Pr [\mathbf{G}_{\text{DEM}, \mathcal{A}}^{\text{dem-ind-atk}}(k) = b] - 1/2 \right|$$

where,

Game $\mathbf{G}_{DEM, \mathcal{A}}^{dem-ind-atk}(k)$

$$K \xleftarrow{\$} DEM.KeyGen(1^k); (St, m_0, m_1) \xleftarrow{\$} \mathcal{A}_1(1^k)$$

$$b \xleftarrow{\$} \{0, 1\}; C^* \xleftarrow{\$} DEM.Encrypt(m_b, K)$$

$$b' \xleftarrow{\$} \mathcal{A}_2^{DEC(\cdot)}(C^*, St); \text{Return } b'$$

and oracle $DEC(\cdot)$ is defined as ε in the OT attack case and as $DEM.Decrypt(K, \cdot)$ in the OTCCA case with the condition that the oracle rejects queries on C^* after the target ciphertext is give to the adversary.

That is, an adversary cannot distinguish between the encryption of two messages. Note that \mathcal{A}_1 does not have access to an encryption oracle as this is a one-time scheme, *i.e.*, the key is used for only one encryption.

KEM-DEM schemes are hybrid encryption schemes where the key generated by the KEM scheme is used by the DEM for data encapsulation. *KEM-DEM* schemes were shown to be secure [67, 38, 27]. The result is stated in the following theorem.

Theorem 4.1. *If KEM is secure against an adaptive chosen ciphertext attacks and DEM is secure against one-time adaptive chosen ciphertext attacks then the hybrid encryption scheme KEM-DEM is secure against adaptive chosen ciphertext attacks. Specifically, if \mathcal{A} is a PPT adversary, then there exist PPT adversaries \mathcal{B}_1 and \mathcal{B}_2 , whose running times are essentially the same as that of \mathcal{A} , such that for all $k \in \mathbb{N}$,*

we have

$$\begin{aligned} \mathbf{Adv}_{KEM-DEM, \mathcal{A}}^{kem-dem-ind-cca2}(k) = \\ \mathbf{Adv}_{KEM, \mathcal{B}_1}^{kem-ind-cca2}(k) + \mathbf{Adv}_{DEM, \mathcal{B}_2}^{dem-ind-otcca}(k) \end{aligned}$$

4.3.3 Policy and Key Encapsulation Mechanism

A *policy and key encapsulation mechanism* (PKEM) is an encapsulation mechanism, which we define to encapsulate both a key and a policy. Similar to a KEM a PKEM consists of three algorithms, namely, PKEM.KeyGen, PKEM.Encrypt and PKEM.Decrypt and it provides the following interface. PKEM.KeyGen is a PPT algorithm that takes as input a security parameter, $k \in \mathbb{N}$, and outputs a key pair (sk, pk) , *i.e.*, $(sk, pk) \stackrel{\$}{\leftarrow} PKEM.KeyGen(1^k)$. PKEM.Encrypt is a PPT algorithm that takes as input a bit string from the message space (interpreted as a policy), pol , and a public-key, pk , generated by PKEM.KeyGen and outputs a pair (K, C) , where $K \in \{0, 1\}^{PKEM.KeyLen(k)}$ is a key ($KEM.KeyLen(k)$ is key-length) and C is a ciphertext, *i.e.*, $(K, C) \stackrel{\$}{\leftarrow} KEM.Encrypt(pk, pol)$. PKEM.Decrypt is a deterministic polynomial-time algorithm that takes as input a secret-key, sk , and ciphertext, C , and returns either key and policy pair (K, pol) or a rejection symbol \perp , *i.e.*, $\{(K, pol), \perp\} \leftarrow PKEM.Decrypt(sk, C)$. For correctness, we require that $\forall k \in \mathbb{N}$, and $\forall (sk, pk) \stackrel{\$}{\leftarrow} KEM.KeyGen(1^k)$ we have $PKEM.Decrypt(sk, C) = (K, pol)$ for any $(K, C) \stackrel{\$}{\leftarrow} PKEM.Encrypt(pk, pol)$.

Given that a PKEM encapsulates both a key and policy we define two notions of indistinguishability for a PKEM against an adaptive chosen ciphertext attack, *viz*, *key indistinguishability* and *policy indistinguishability*. We define each of them

as follows.

Definition 4.6. Key Indistinguishability. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a PPT CCA2 adversary. We define the guessing advantage of \mathcal{A} as follows:

$$\text{Adv}_{PKEM, \mathcal{A}}^{pkem-key-ind-cca2}(k) = \left| \Pr \left[\mathbf{G}_{PKEM, \mathcal{A}}^{pkem-key-ind-cca2}(k) = b \right] - 1/2 \right|$$

where

Game $\mathbf{G}_{PKEM, \mathcal{A}}^{pkem-key-ind-cca2}(k)$:

$$(sk, pk) \stackrel{\$}{\leftarrow} PKEM.KeyGen(1^k); (St, pol) \stackrel{\$}{\leftarrow} \mathcal{A}_1^{DEC(\cdot)}(pk)$$

$$b \stackrel{\$}{\leftarrow} \{0, 1\}; K_0^* \stackrel{\$}{\leftarrow} \{0, 1\}^{KEM.KeyLen(k)}$$

$$(K_1^*, C^*) \stackrel{\$}{\leftarrow} PKEM.Encrypt(pk, pol); K^* \stackrel{\$}{\leftarrow} K_b^*$$

$$b' \stackrel{\$}{\leftarrow} \mathcal{A}_2^{DEC(\cdot)}(pk, C^*, K^*, St); \text{Return } b'$$

and oracle $DEC(\cdot)$ is defined as $PKEM.Decrypt(sk, \cdot)$ with the condition that the oracle rejects queries on C^* after the target ciphertext is given to the adversary.

Definition 4.7. Policy Indistinguishability. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a PPT CCA2 adversary. We define the guessing advantage of \mathcal{A} as follows:

$$\text{Adv}_{PKEM, \mathcal{A}}^{pkem-pol-ind-cca2}(k) = \left| \Pr \left[\mathbf{G}_{PKEM, \mathcal{A}}^{pkem-pol-ind-cca2}(k) = b \right] - 1/2 \right|$$

where

Game $\mathbf{G}_{PKEM, \mathcal{A}}^{pkem-pol-ind-cca2}(k)$:

$$(sk, pk) \stackrel{\$}{\leftarrow} PKEM.KeyGen(1^k); (St, pol_0, pol_1) \stackrel{\$}{\leftarrow} \mathcal{A}_1^{DEC(\cdot)}(pk)$$

$$b \stackrel{\$}{\leftarrow} \{0, 1\}; (K^*, C^*) \stackrel{\$}{\leftarrow} PKEM.Encrypt(pk, pol_b)$$

$$b' \stackrel{\$}{\leftarrow} \mathcal{A}_2^{DEC(\cdot)}(pk, (K^*, C^*), St); \text{Return } b'$$

and oracle $DEC(\cdot)$ is defined as $PKEM.Decrypt(sk, \cdot)$ with the condition that the oracle rejects queries on C^* after the target ciphertext is given to the adversary.

A *symmetric-key based PKEM* (SPKEM) is similar to the public-key based PKEM described above except that a symmetric key is used instead of the asymmetric key-pair. Notions of key and policy indistinguishability for SPKEM are defined similarly to that of PKEM except that they are defined for an OTCCA adversary, *i.e.*, the adversary doesn't get access to encryption oracle in the first phase. We construct a SPKEM using a DEM as shown below and then build a PKEM using SPKEM and KEM.

<p>SPKEM.Encrypt(pol, K) :</p> <p>$K' \xleftarrow{\\$} DEM.KeyGen(1^k)$</p> <p>$m' \leftarrow pol \ K'$</p> <p>$C \leftarrow DEM.Encrypt(m', K)$</p> <p>Return (K', C)</p>	<p>SPKEM.Decrypt(K, C) :</p> <p>$m' \leftarrow DEM.Decrypt(K, C)$</p> <p>if $m' = \perp$ or parsing m'</p> <p>as $pol \ K'$ fails return \perp</p> <p>else Return (pol, K')</p>
---	--

where K is generated by $DEM.KeyGen(1^K)$.

This scheme is secure against OTCCA attacks on key and policy indistinguishability as stated by the following theorems.

Theorem 4.2. *If DEM is secure against one-time adaptive chosen ciphertext attacks (OTCCA) on (message) indistinguishability then SPKEM is secure against one-time adaptive chosen ciphertext attacks (OTCCA) on key indistinguishability.*

In particular, for every PPT adversary \mathcal{A} , there exists a PPT adversary \mathcal{B} whose running time is essentially the same as that of \mathcal{A} such that for all $k \in \mathbb{N}$, we have

$$\mathbf{Adv}_{SPKEM, \mathcal{A}}^{spkem-key-ind-cca2}(k) = \mathbf{Adv}_{DEM, \mathcal{B}}^{dem-ind-cca2}(k) \quad (4.1)$$

Proof of Theorem 4.2. Adversary \mathcal{B} runs adversary \mathcal{A} and accurately simulates a SPKEM game environment as follows. When \mathcal{A} outputs a policy pol and requests challenge key and ciphertext pair, (K^*, C^*) , \mathcal{B} does the following. It generates two random keys K_0 and K_1 for the underlying DEM scheme and creates two messages $m_0 = pol \| K_0$ and $m_1 = pol \| K_1$ and outputs them to its DEM game environment and gets the challenge ciphertext $C^* = DEM.Encrypt(m_b, K)$ back. \mathcal{B} gives \mathcal{A} the following challenge key and ciphertext pair (K_1, C^*) . \mathcal{B} forwards any decryption queries \mathcal{A} has to the decryption oracle it has access to and parses the output as $pol \| K$ before returning it \mathcal{A} . \mathcal{B} outputs the guess bit that \mathcal{A} outputs. Note that when $b = 1$, \mathcal{A} gets the real key and when $b = 0$, \mathcal{A} gets the random key. So if \mathcal{A} outputs 1 when $b = 1$ or if \mathcal{A} outputs 0 when $b = 0$ then \mathcal{A} is successful by design. Success probability of adversary \mathcal{B} , $Pr[\mathbf{Adv}_{DEM, \mathcal{B}}^{dem-ind-otcca}(k) = b]$ is,

$$\begin{aligned} &= 1/2 \cdot Pr[\mathbf{Adv}_{DEM, \mathcal{B}}^{dem-ind-otcca}(k) = 1 | b = 1] + \\ &\quad 1/2 \cdot Pr[\mathbf{Adv}_{DEM, \mathcal{B}}^{dem-ind-otcca}(k) = 0 | b = 0] \\ &= 1/2 \cdot Pr[\mathbf{Adv}_{SPKEM, \mathcal{A}}^{spkem-key-ind-otcca}(k) = 1 | b = 1] + \\ &\quad 1/2 \cdot Pr[\mathbf{Adv}_{SPKEM, \mathcal{A}}^{spkem-key-ind-otcca}(k) = 0 | b = 0] \\ &= Pr[\mathbf{Adv}_{SPKEM, \mathcal{A}}^{spkem-key-ind-otcca}(k) = b] \end{aligned}$$

□

Theorem 4.3. *If DEM is secure against one-time adaptive chosen ciphertext attacks (OTCCA) on (message) indistinguishability then SPKEM is secure against one-time adaptive chosen ciphertext attacks (OTCCA) on policy indistinguishability.*

In particular, for every PPT adversary \mathcal{A} , there exists a PPT adversary \mathcal{B} whose running time is essentially the same as that of \mathcal{A} such that for all $k \in \mathbb{N}$, we have

$$\mathbf{Adv}_{SPKEM, \mathcal{A}}^{spkem-pol-ind-cca2}(k) = \mathbf{Adv}_{DEM, \mathcal{B}}^{dem-ind-cca2}(k) \quad (4.2)$$

Proof of Theorem 4.3. Adversary \mathcal{B} simply runs \mathcal{A} . On receiving pol_1 and pol_2 from \mathcal{A} , \mathcal{B} generates a random key, K^* , for the underlying DEM scheme, sends $pol_1 \| K^*$ and $pol_2 \| K^*$ to its DEM game environment and returns the target ciphertext it receives to \mathcal{A} . For decryption queries from \mathcal{A} , \mathcal{B} queries its own decryption oracle, parses the reply (if not \perp) as pol and K and returns it to \mathcal{A} . When \mathcal{A} outputs a guess \mathcal{B} outputs the same value. Clearly, \mathcal{B} accurately simulates the SPKEM game environment for \mathcal{A} . Thus any advantage \mathcal{A} has in breaking policy indistinguishability of SPKEM is translated into advantage in breaking (message) indistinguishability of DEM. □

We now construct a PKEM scheme using a SPKEM and a KEM as follows:

PKEM.KeyGen (1^k) :	PKEM.Decrypt (sk, C) :
$(sk, pk) \xleftarrow{\$} KEM.KeyGen(1^k)$	parse C as $C_1 \ C_2$
Return (sk, pk)	$K_1 \leftarrow KEM.Decrypt(sk, C_1)$
PKEM.Encrypt (pol, pk) :	if $K_1 \neq \perp$
$(K_1, C_1) \xleftarrow{\$} KEM.Encrypt(1^k, pk)$	$m' \leftarrow SPKEM.Decrypt(K_1, C_2)$
$(K_2, C_2) \xleftarrow{\$} SPKEM.Encrypt(pol, K_1)$	if $m' = \perp$ return \perp
$C \leftarrow C_1 \ C_2$	else Return $m' = (pol, K_2)$
Return (K_2, C)	

The above PKEM scheme is secure against adaptive chosen ciphertext attacks on both key and policy indistinguishability. In particular, the following theorems hold.

Theorem 4.4. *If KEM and SPKEM schemes are secure against adaptive chosen ciphertext attacks on key indistinguishability then PKEM is secure against adaptive chosen ciphertext attacks on key indistinguishability.*

In particular, for every PPT adversary \mathcal{A} , there exist PPT adversaries \mathcal{B}_1 and \mathcal{B}_2 , whose running times are essentially the same as that of \mathcal{A} , such that for all $k \in \mathbb{N}$, we have,

$$\begin{aligned} \mathbf{Adv}_{PKEM, \mathcal{A}}^{pkem-key-ind-cca2}(k) &\leq \\ &2 \cdot \mathbf{Adv}_{KEM, \mathcal{B}_1}^{kem-ind-cca2}(k) + \mathbf{Adv}_{SPKEM, \mathcal{B}_2}^{SPKEM-key-ind-otcca}(k) \end{aligned} \tag{4.3}$$

Proof of Theorem 4.4. Let \mathbf{G}_0 be the original attack game defined by Definition 4.6. Fix \mathcal{A} and k and let $C^* = (C_1^*, C_2^*)$ denote the target ciphertext. Let \mathcal{E}_0 denote the event that $b' = b$ in \mathbf{G}_0 so that

$$\text{Adv}_{\mathcal{PKEM}, \mathcal{A}}^{\text{pkem-key-ind-cca2}}(k) = |\text{Pr}[\mathcal{E}_0] - 1/2| \quad (4.4)$$

We shall define two modified attack games \mathbf{G}_1 and \mathbf{G}_2 . Each of the games $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2$ operates on the same underlying probability space. That is, the cryptographic keys, coin tosses of \mathcal{A} and hidden bit b take identical values across all games. However the games differ in how the environment responds to oracle queries. Let \mathcal{E}_i be the event that $b' = b$ in game \mathbf{G}_i for $1 \leq i \leq 2$.

Game \mathbf{G}_1 In this game whenever a ciphertext (C_1, C_2) is submitted to the decryption oracle after the invocation of the encryption oracle, if $C_1 = C_1^*$ but $C_2 \neq C_2^*$, then the decryption oracle does not apply *KEM.Decrypt* to obtain the symmetric key but uses K_1^* produced by the encryption oracle instead. This is just a conceptual change and

$$\text{Pr}[\mathcal{E}_0] = \text{Pr}[\mathcal{E}_1] \quad (4.5)$$

Game \mathbf{G}_2 This game is similar to the game \mathbf{G}_1 except that a completely random key, K_1^\dagger , is used in place of K_1^* in both encryption and decryption oracles. Any difference in the success probability of \mathcal{A} against games \mathbf{G}_1 and \mathbf{G}_2 can be leveraged to construct an adversary algorithm that can break CCA security of KEM. More precisely we have:

Lemma 4.8. *There exists a probabilistic algorithm \mathcal{B}_1 whose running time is essen-*

tially the same as that of \mathcal{A} , such that

$$|Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]| = 2 \cdot \mathbf{Adv}_{KEM, \mathcal{B}_1}^{kem-ind-cca2}(k) \quad (4.6)$$

Furthermore, in game \mathbf{G}_2 , since a random key, K_1^\dagger , independent of the one encapsulated by C_1^* , is used to produce the target ciphertext C_2^* and by the decryption oracle, \mathcal{A} is essentially carrying out a one-time adaptive chosen ciphertext attack against the SPKEM scheme described above. Thus we have

$$|Pr[\mathcal{E}_2] - 1/2| = \mathbf{Adv}_{SPKEM, \mathcal{A}}^{spkem-key-ind-cca2}(k) \quad (4.7)$$

The theorem now follows from equations 4.4, 4.5, 4.6 and 4.7.

□

Proof of Lemma 4.8. In the game against KEM, \mathcal{B} is given public-key, pk , and access to a decryption oracle for KEM. \mathcal{B} runs \mathcal{A} with the public-key pk . Decryption queries from \mathcal{A} are answered by \mathcal{B} using the decryption oracle. When \mathcal{A} outputs a policy pol and asks for the challenge key and ciphertext pair, \mathcal{B} does the following: 1) it gets a challenge key and ciphertext pair, (K^*, C_1^*) , from the KEM game environment, 2) it generates two random keys, K_0 and K_1 for the underlying DEM, 3) picks a bit $b \xleftarrow{\$} \{0, 1\}$ and 4) computes $C_2^* = DEM.Encrypt((pol \| K_1), K^*)$ and gives the challenge pair $(K_b, (C_1^*, C_2^*))$ to \mathcal{A} . Here K^* is the key encapsulated by C_1^* if $\delta = 1$ or a random key if $\delta = 0$ where $\delta \xleftarrow{\$} \{0, 1\}$ is chosen by KEM game environment. For decryption queries from \mathcal{A} in the second phase, if $C_1 = C_1^*$, \mathcal{B} uses K^* to decrypt C_2 otherwise it uses the decryption oracle for KEM. If \mathcal{A} outputs a guess bit $b' = b$ then \mathcal{B} outputs $\delta' = 1$ else it outputs $\delta' = 0$. Note that when $\delta = 1$ \mathcal{A} is in game

\mathbf{G}_1 and when $\delta = 0$ \mathcal{A} is in game \mathbf{G}_2 . Therefore $Pr[b' = b|\delta = 1] = Pr[\mathcal{E}_1]$ and $Pr[b' = b|\delta = 0] = Pr[\mathcal{E}_2]$.

$$\begin{aligned} Pr[\delta' = \delta] - 1/2 &= 1/2 \cdot |Pr[\delta' = 1|\delta = 1] - Pr[\delta' = 1|\delta = 0]| \\ &= 1/2 \cdot |Pr[b' = b|\delta = 1] - Pr[b' = b|\delta = 0]| \\ &= 1/2 \cdot |Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]| \end{aligned}$$

But $Pr[\delta' = \delta] - 1/2 = \mathbf{Adv}_{KEM, \mathcal{B}_1}^{kem-ind-cca2}(k)$ therefore

$$|Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]| = 2 \cdot \mathbf{Adv}_{KEM, \mathcal{B}_1}^{kem-ind-cca2}(k)$$

□

Theorem 4.5. *If the underlying KEM and SPKEM schemes are secure against adaptive chosen ciphertext attacks on key and policy indistinguishability, respectively, then PKEM is secure against adaptive chosen ciphertext attacks on policy indistinguishability.*

In particular, for every PPT adversary \mathcal{A} , there exists a PPT adversary \mathcal{B}_1 and \mathcal{B}_2 , whose running time is essentially the same as that of \mathcal{A} , such that for all $k \in \mathbb{N}$, we have,

$$\begin{aligned} \mathbf{Adv}_{PKEM, \mathcal{A}}^{pkem-pol-ind-cca2}(k) &= \\ &= \mathbf{Adv}_{KEM, \mathcal{B}}^{kem-ind-cca2}(k) + \mathbf{Adv}_{SPKEM, \mathcal{B}}^{spkem-pol-ind-otcca}(k) \end{aligned} \tag{4.8}$$

Proof Sketch of Theorem 4.5. This proof very similar to that of Theorem 4.4 above except that in Game \mathbf{G}_2 the adversary is launching an OTCCA attack against policy indistinguishability of SPKEM instead of key indistinguishability. □

4.4 Policy Based Encryption System

Our encryption scheme is based on KEM-DEM hybrid encryption paradigm [27] and uses Key Encapsulation Mechanism (KEM) and Data Encapsulation Mechanism (DEM) as building blocks. For ease of exposition we define and use a construction *Policy and Key Encapsulation Mechanism* (PKEM) to build our scheme dubbed PKEM-DEM. In our PKEM-DEM encryption scheme a file/message, m , is encapsulated using a DEM where the key used by the DEM and the policy associated with the message, pol , are encapsulated using PKEM as defined below.

<p>PKEM-DEM.KeyGen(1^k) :</p> <p>$(sk, pk) \xleftarrow{\\$} PKEM.KeyGen(1^k)$</p> <p>Return (sk, pk)</p>	<p>PKEM-DEM.Decrypt-I(sk, f, C_1, u) :</p> <p>$m' \leftarrow PKEM.Decrypt(sk, C_1)$</p> <p>if $m' = \perp$ Return \perp</p> <p>else parse m' as (pol, K_2)</p>
<p>PKEM-DEM.Encrypt(m, pol, pk) :</p> <p>$(K_2, C_1) \xleftarrow{\\$} PKEM.Encrypt(pol, pk)$</p> <p>$C_2 \leftarrow DEM.Encrypt(m, K_2)$</p> <p>$C \leftarrow C_1 C_2$</p> <p>Return C</p>	<p>if $f(u, pol) = 1$ Return K_2</p> <p>else Return \perp</p> <p>PKEM-DEM.Decrypt-II(K_2, C_2) :</p> <p>if $K_2 = \perp$ Return \perp</p> <p>$m \leftarrow DEM.Decrypt(K_2, C_2)$</p> <p>Return m</p>

Here, u represents a user and his associated attributes along with contextual attributes and f represents the policy evaluation function and is a deterministic

polynomial-time function that takes as input u , and a policy, pol , and returns a 1 if the user along with context satisfies the policy or a 0 otherwise. A PKEM-DEM scheme can be constructed using any KEM and DEM where the two schemes are independent². Figure 4.2 shows encryption in PKEM-DEM scheme instantiated using RSA-KEM and DEM1 defined in [27]

We use our PKEM-DEM encryption scheme to develop the PBES policy based encryption system whose architecture is illustrated in Figure 4.1 and described in Section 4.1.2. The data owner in our system specifies a policy pol and uses the PKEM-DEM scheme to securely associate the policy with the data m and generate an encrypted object $E(o)$ that hides both the policy and the data. In order to do so it chooses a KDC that it trusts to enforce the policy and release the DEM object encryption key to recipient(s) that satisfy the policy. It then obtains the public key of the KDC, PK , via a trusted source (e.g., a Certificate Authority – CA) and encrypts the object using the PKEM-DEM scheme.

Once a recipient obtains the encrypted object it must contact the KDC represented by the public key PK in the encrypted object in order to obtain the DEM object decryption key. To do so it initiates a protected transaction (e.g., over TLS) with the KDC and submits the PKEM part of the encrypted object (i.e., it excludes the encrypted object in the DEM part). The KDC then contacts the Attribute Database that manages user attributes and privileges and environmental attributes (*i.e.*, context). The KDC uses these attributes of the user and the environment and

²KEM-DEM schemes built using secure KEM and secure DEM that are related may not be secure as shown in [38]

the PKEM part of the object as inputs to PKEM-DEM.Decrypt-I to obtain the DEM keys. The KDC releases the object decryption key, K , to the recipient and the recipient uses this key to decrypt the object using PKEM-DEM.Decrypt-II.

4.5 Security Analysis

Since pairwise indistinguishability (in Def. 4.3) implies message indistinguishability (in Def. 4.1) and policy indistinguishability (in Def. 4.2) with restriction 2b, we prove that PKEM-DEM is pairwise indistinguishable in Theorem 4.6 and that it is policy indistinguishable with restriction 2a in Theorem 4.7 to show that PKEM-DEM system is secure against adaptive chosen ciphertext attacks.

In the proofs for the following Theorems, decryption oracle for PKEM-DEM executes PKEM-DEM.Decrypt-I and PKEM-DEM.Decrypt-II on the decryption query and returns the output of both the algorithms to the adversary.

Theorem 4.6. *If DEM is secure against one-time chosen ciphertext attacks and PKEM is secure against chosen ciphertext attacks against both key and policy indistinguishability then PKEM-DEM is secure against chosen ciphertext attacks on pairwise indistinguishability as given in Definition 4.3.*

In particular we have

$$\begin{aligned} \mathbf{Adv}_{PKEM-DEM}^{pkem-dem-pw-ind-cca2}(k) \leq \\ 5 \cdot \mathbf{Adv}_{KEM}^{kem-ind-cca2}(k) + 3 \cdot \mathbf{Adv}_{DEM}^{dem-ind-otcca}(k) \end{aligned} \tag{4.9}$$

Proof of Theorem 4.6. Let \mathbf{G}_0 be the original attack game, *i.e.*, $\mathbf{G}_{PKEM-DEM, \mathcal{A}}^{pkem-dem-ind-cca2}(k)$, described in Definition 4.3. Fix \mathcal{A} and k and let $C^* = (C_1^*, C_2^*)$ denote the target

ciphertext. Let \mathcal{E}_0 denote the event that $b' = b$ in \mathbf{G}_0 so that

$$\mathbf{Adv}_{PKEM-DEM, \mathcal{A}}^{pkem-dem-ind-cca2-cu}(k) = |Pr[\mathcal{E}_0] - 1/2| \quad (4.10)$$

We shall define two modified attack games \mathbf{G}_1 and \mathbf{G}_2 . Each of the games $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2$ operates on the same underlying probability space. That is, the cryptographic keys, coin tosses of \mathcal{A} and hidden bit b take identical values across all games. However, the games differ in how the environment responds to oracle queries. Let \mathcal{E}_i be the event that $b' = b$ in game \mathbf{G}_i for $1 \leq i \leq 2$.

Game \mathbf{G}_1 In this game whenever a ciphertext (C_1, C_2) is submitted to the decryption oracle after the invocation of the encryption oracle, if $C_1 = C_1^*$ but $C_2 \neq C_2^*$, then the decryption oracle does not apply $PKEM.Decrypt$ to obtain the symmetric key but uses K_2^* produced by the encryption oracle instead. This is just a conceptual change and

$$Pr[\mathcal{E}_0] = Pr[\mathcal{E}_1] \quad (4.11)$$

Game \mathbf{G}_2 This game is similar to the game \mathbf{G}_1 except that a completely random key, K_2^\dagger , is used in place of K_2^* in both encryption and decryption oracles. Any difference in the success probability of \mathcal{A} against games \mathbf{G}_1 and \mathbf{G}_2 can be leveraged to construct an adversary algorithm that can break key indistinguishability of PKEM. More precisely we have:

Lemma 4.9. *There exists a probabilistic algorithm \mathcal{B}_1 whose running time is essentially the same as that of \mathcal{A} , such that*

$$|Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]| = 2 \cdot \mathbf{Adv}_{PKEM, \mathcal{B}_1}^{pkem-key-ind-cca2}(k) \quad (4.12)$$

We observe that in game \mathbf{G}_2 , message m_b is encapsulated with a DEM using a key, K_2^\dagger , that is independent of the one encapsulated by PKEM. Thus, in game \mathbf{G}_2 , adversary \mathcal{A} is essentially carrying out one-time adaptive chosen ciphertext attack against an instance of DEM or an adaptive chosen ciphertext attack on the policy indistinguishability against an instance of PKEM. Specifically, we have:

Lemma 4.10. *There exists probabilistic algorithms \mathcal{B}_2 and \mathcal{B}_3 whose running times (and number of decryption queries) are at most twice that of \mathcal{A} , such that*

$$|Pr[\mathcal{E}_3] - 1/2| \leq \mathbf{Adv}_{DEM, \mathcal{B}_2}^{dem-ind-cca2}(k) + \mathbf{Adv}_{PKEM, \mathcal{B}_3}^{pkem-pol-ind-cca2}(k) \quad (4.13)$$

The theorem now follows from equations 4.10, 4.11, 4.12 and 4.13. \square

We now give proofs of Lemmas 4.9 and 4.10 to complete the proof of Theorem 4.6.

Proof of Lemma 4.9. \mathcal{B}_1 is an adversary against key indistinguishability of PKEM and is given public-key, pk , and access to a decryption oracle for PKEM. \mathcal{B}_1 runs \mathcal{A} with the public-key pk . When adversary \mathcal{A} adds/corrupts a user, u_i , \mathcal{B}_1 stores the user u_i and associated attributes in a list. Decryption queries, $C = (C_1, C_2)$, with privileges of user u_i from \mathcal{A} are answered by \mathcal{B}_1 as follows: 1) \mathcal{B}_1 submits C_1 to its PKEM oracle and gets either a \perp or (pol, K_2) , 2) if \perp , it returns \perp to \mathcal{A} , 3) else, if $f(u_i, pol) = 1$ returns K_2 and $DEM.Decrypt(K_2, C_2)$ otherwise it returns \perp . When \mathcal{A} outputs a message and policy pairs (m_0, m_1) and (pol_1, pol_2) and asks for the challenge ciphertext, \mathcal{B}_1 does the following: 1) verifies that none of the of the corrupted users u_i satisfies either pol_0 or pol_1 , 2) picks a bit $b \xleftarrow{\$} \{0, 1\}$, 3) gives

pol_b to the PKEM game environment and gets a challenge key and ciphertext pair, (K^*, C_1^*) , and 4) computes $C_2^* = DEM.Encrypt(m_b, K^*)$ and gives the challenge pair (C_1^*, C_2^*) to \mathcal{A} . Here K^* is the key encapsulated by C_1^* if $\delta = 1$ or a random key if $\delta = 0$ where $\delta \xleftarrow{\$} \{0, 1\}$ is chosen by PKEM game environment. In the second phase, when \mathcal{A} adds/corrupts a user u_i , \mathcal{B}_1 verifies that u_i does not satisfy either pol_0 or pol_1 . To answer decryption queries, $C = (C_1, C_2)$ from \mathcal{A} in the second phase, \mathcal{B}_1 uses the decryption oracle for PKEM as described above. Note that if \mathcal{A} asks queries where $C_1 = C_1^*$ then \mathcal{B}_1 returns \perp since none of the users compromised by \mathcal{A} satisfy either pol_1 or pol_2 . If \mathcal{A} outputs a guess bit $b' = b$ then \mathcal{B}_1 outputs $\delta' = 1$ else it outputs $\delta' = 0$. Note that when $\delta = 1$ \mathcal{A} is in game \mathbf{G}_1 and when $\delta = 0$ \mathcal{A} is in game \mathbf{G}_2 . Therefore $Pr[b' = b | \delta = 1] = Pr[\mathcal{E}_1]$ and $Pr[b' = b | \delta = 0] = Pr[\mathcal{E}_2]$. Then,

$$\begin{aligned}
Pr[\delta' = \delta] - 1/2 &= 1/2 \cdot |Pr[\delta' = 1 | \delta = 1] - Pr[\delta' = 1 | \delta = 0]| \\
&= 1/2 \cdot |Pr[b' = b | \delta = 1] - Pr[b' = b | \delta = 0]| \\
&= 1/2 \cdot |Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]|
\end{aligned}$$

But $Pr[\delta' = \delta] - 1/2 = \mathbf{Adv}_{PKEM, \mathcal{B}_1}^{pkem-key-ind-cca2}(k)$ therefore

$$|Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]| = 2 \cdot \mathbf{Adv}_{PKEM, \mathcal{B}_1}^{pkem-key-ind-cca2}(k)$$

□

□

Proof of Lemma 4.10. Let probability of success of $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in game \mathbf{G}_2 be $1/2 + \epsilon$. Then, $|Pr[\mathcal{E}_2] - 1/2| = \epsilon$. Furthermore, let $1/2 + \alpha$ be the probability

that \mathcal{A} outputs 1 when the challenge ciphertext it is given encrypts m_0 and pol_1 and $1/2 + \beta$ be the probability that \mathcal{A} outputs 1 when the challenge ciphertext it is given encrypts m_1 and pol_0 .

Part 1. \mathcal{B}_2 is OTCCA adversary against (message) indistinguishability of DEM that runs \mathcal{A} . In particular, \mathcal{B}_2 generates a KEM key pair, (sk, pk) , and runs one instance of \mathcal{A}_1 giving it pk and two instances of \mathcal{A}_2 (*i.e.*, $\mathcal{A}_{2,0}$ and $\mathcal{A}_{2,1}$) with different challenge ciphertexts as follows. Phase 1 queries of \mathcal{A}_1 are answered similar to the way described in proof of Lemma 4.9 above except that \mathcal{B}_2 has access to sk . When \mathcal{A}_1 outputs a message pair (m_0, m_1) and policy pair (pol_0, pol_1) and state information St , \mathcal{B}_2 does the following: 1) verifies that none of the of the corrupted users u_i satisfies either pol_0 or pol_1 , 2) gives the pair (m_0, m_1) to the DEM game environment and obtains the challenge ciphertext $C_2^* = DEM.Encrypt(m_\delta, K_{dem})$, 2) computes the following $C_{1,0}^* = PKEM.Encrypt(pol_0, pk)$, $C_{1,1}^* = PKEM.Encrypt(pol_1, pk)$ and 3) runs $\mathcal{A}_{2,0}$ with $(C_{1,0}^*, C_2^*)$ and $\mathcal{A}_{2,1}$ with $(C_{1,1}^*, C_2^*)$ as the challenge ciphertexts. Phase 2 queries of \mathcal{A}_2 are answered just like phase 1 except, 1) when \mathcal{A}_2 adds/corrupts a user u_i , \mathcal{B}_2 verifies that u_i does not satisfy either pol_0 or pol_1 and 2) when decryption query of $\mathcal{A}_{2,\psi}$ has $C_1 = C_{1,\psi}^*$ in which case \mathcal{B}_2 returns \perp as the adversary does not satisfy either of the policies. Let $\mathcal{A}_{2,0}$'s output be b_0 and $\mathcal{A}_{2,1}$'s output be b_1 . \mathcal{B}_2 outputs $\delta' = b_0$ if $b_0 = b_1$ and outputs $\delta' = b_\theta$ otherwise, where $\theta \xleftarrow{\$} \{0, 1\}$. Thus, the success probability of \mathcal{B}_2 is $Pr[\delta' = \delta]$ is

$$\begin{aligned}
&= \frac{1}{2} \cdot (Pr[\delta' = 0|\delta = 0] + Pr[\delta' = 1|\delta = 1]) \\
&= \frac{1}{2} \cdot \left((Pr[b_0 = 0 \wedge b_1 = 0|\delta = 0] + Pr[\theta = 0 \wedge b_0 = 0 \wedge b_1 = 1|\delta = 0] \right. \\
&\quad \left. + Pr[\theta = 1 \wedge b_0 = 1 \wedge b_1 = 0|\delta = 0]) + (Pr[b_0 = 1 \wedge b_1 = 1|\delta = 1] \right. \\
&\quad \left. + Pr[\theta = 0 \wedge b_0 = 1 \wedge b_1 = 0|\delta = 1] \right. \\
&\quad \left. + Pr[\theta = 1 \wedge b_0 = 0 \wedge b_1 = 1|\delta = 1]) \right) \\
&= \frac{1}{2} \cdot \left(\left(\left(\frac{1}{2} + \epsilon \right) \left(\frac{1}{2} - \alpha \right) + \frac{1}{2} \cdot \left(\frac{1}{2} + \epsilon \right) \left(\frac{1}{2} + \alpha \right) + \frac{1}{2} \cdot \left(\frac{1}{2} - \epsilon \right) \left(\frac{1}{2} - \alpha \right) \right) \right. \\
&\quad \left. + \left(\left(\frac{1}{2} + \beta \right) \left(\frac{1}{2} + \epsilon \right) + \frac{1}{2} \cdot \left(\frac{1}{2} + \beta \right) \left(\frac{1}{2} - \epsilon \right) + \frac{1}{2} \cdot \left(\frac{1}{2} - \beta \right) \left(\frac{1}{2} + \epsilon \right) \right) \right) \\
&= \frac{1}{2} + \frac{\epsilon}{2} + \frac{(\beta - \alpha)}{4}
\end{aligned}$$

But $|Pr[\delta' = \delta] - 1/2| = \mathbf{Adv}_{DEM, \mathcal{B}_2}^{dem-ind-cca2}(k) = \epsilon_{DEM}$ (say) therefore we have

$$\epsilon = 2 \cdot \epsilon_{DEM} + \frac{(\alpha - \beta)}{2} \quad (4.14)$$

Part 2. \mathcal{B}_3 is CCA adversary against policy indistinguishability of PKEM that runs \mathcal{A} . \mathcal{B}_3 is constructed similarly to \mathcal{B}_2 with obvious modifications.

In particular, \mathcal{B}_4 generates a KEM keypair, (sk, pk) , and runs one instance of \mathcal{A}_1 giving it pk and two instances of \mathcal{A}_2 (*i.e.*, $\mathcal{A}_{2,0}$ and $\mathcal{A}_{2,1}$) with different challenge ciphertexts. Decryption queries of \mathcal{A}_1 are answered in the obvious way using sk . When \mathcal{A}_1 outputs a message pair (m_0, m_1) and policy pair (pol_0, pol_1) and state information St , \mathcal{B}_4 does the following: 1) gives the pair (pol_0, pol_1) to the SPKEM game environment and obtains the challenge ciphertext $C_2^* = DEM.Encrypt((pol_0 || K'), K_{SPKEM})$, 2) generates a random DEM keys K and computes the following $C_{3,0}^* = DEM.Encrypt(m_0, K)$,

$C_{3,1}^* = DEM.Encrypt(m_1, K)$ and $(K^*, C_1^*) = KEM.Encrypt(1^k, pk)$ and 3) runs $\mathcal{A}_{2,0}$ with $(C_1^*, C_2^*, C_{3,0}^*)$ and $\mathcal{A}_{2,1}$ with $(C_1^*, C_2^*, C_{3,1}^*)$ as the challenge ciphertexts. To answer decryption queries, $C = (C_1, C_2, C_3)$ from \mathcal{A} in the second phase, if $C_1 = C_1^*$, \mathcal{B} uses decryption oracle provided by SPKEM environment to decrypt C_2 otherwise it uses sk . Let $\mathcal{A}_{2,0}$'s output be b_0 and $\mathcal{A}_{2,1}$'s output be b_1 . \mathcal{B}_4 outputs $\delta' = b_0$ if $b_0 = b_1$ and outputs $\delta' = b_\theta$ otherwise, where $\theta \stackrel{\$}{\leftarrow} \{0, 1\}$. Thus, the success probability of \mathcal{B}_4 is $Pr[\delta' = \delta]$ is

$$\begin{aligned}
&= \frac{1}{2} \cdot (Pr[\delta' = 0|\delta = 0] + Pr[\delta' = 1|\delta = 1]) \\
&= \frac{1}{2} \cdot \left((Pr[b_0 = 0 \wedge b_1 = 0|delta = 0] \right. \\
&\quad + Pr[\theta = 0 \wedge b_0 = 0 \wedge b_1 = 1|\delta = 0] + Pr[\theta = 1 \wedge b_0 = 1 \wedge b_1 = 0|\delta = 0]) \\
&\quad + (Pr[b_0 = 1 \wedge b_1 = 1|delta = 1] + Pr[\theta = 0 \wedge b_0 = 1 \wedge b_1 = 0|\delta = 1] \\
&\quad \left. + Pr[\theta = 1 \wedge b_0 = 0 \wedge b_1 = 1|\delta = 1]) \right) \\
&= \frac{1}{2} \cdot \left(\left(\left(\frac{1}{2} + \epsilon \right) \left(\frac{1}{2} - \beta \right) + \frac{1}{2} \cdot \left(\frac{1}{2} + \epsilon \right) \left(\frac{1}{2} + \beta \right) + \right. \right. \\
&\quad \left. \frac{1}{2} \cdot \left(\frac{1}{2} - \epsilon \right) \left(\frac{1}{2} - \beta \right) \right) + \left(\left(\frac{1}{2} + \alpha \right) \left(\frac{1}{2} + \epsilon \right) + \right. \\
&\quad \left. \frac{1}{2} \cdot \left(\frac{1}{2} + \alpha \right) \left(\frac{1}{2} - \epsilon \right) + \frac{1}{2} \cdot \left(\frac{1}{2} - \alpha \right) \left(\frac{1}{2} + \epsilon \right) \right) \\
&= \frac{1}{2} + \frac{\epsilon}{2} + \frac{(\alpha - \beta)}{4}
\end{aligned}$$

But $|Pr[\delta' = \delta] - 1/2| = \mathbf{Adv}_{PKEM, \mathcal{B}_3}^{pkem-pol-ind-cca2}(k) = \epsilon_{PKEM}$ (say) therefore we

have

$$\epsilon = 2 \cdot \epsilon_{PKEM} + \frac{(\beta - \alpha)}{2} \tag{4.15}$$

From equations 4.14 and 4.15 we have

$$\epsilon_{PKEM} - \epsilon_{DEM} = \frac{(\alpha - \beta)}{2} \text{Therefore, } \epsilon = \epsilon_{PKEM} + \epsilon_{DEM} \quad (4.16)$$

Thus we have

$$|Pr[\mathcal{E}_2] - 1/2| = \mathbf{Adv}_{PKEM, \mathcal{B}_3}^{pkem-pol-ind-cca2}(k) + \mathbf{Adv}_{DEM, \mathcal{B}_2}^{dem-ind-cca2}(k)$$

□

Theorem 4.7. *If PKEM is secure against chosen ciphertext attacks against policy-indistinguishability then PKEM-DEM is secure against chosen ciphertext attacks on policy indistinguishability as given in Definition 4.2 with restriction 2a.*

In particular we have

$$\begin{aligned} \mathbf{Adv}_{PKEM-DEM}^{pkem-dem-pol-ind-2a-cca2}(k) \leq \\ \mathbf{Adv}_{KEM}^{kem-ind-cca2}(k) + \mathbf{Adv}_{DEM}^{dem-ind-otcca}(k) \end{aligned} \quad (4.17)$$

Proof of Theorem 4.7. Intuitively, since the message encrypted under the both the policies is the same any advantage an adversary has in distinguishing between the two policies encapsulated by the PKEM-DEM scheme must be due to an advantage the adversary has in distinguishing between two policies encapsulated by the PKEM scheme. In other words, any advantage an adversary has in breaking policy indistinguishability of PKEM-DEM can be translated into advantage in breaking policy indistinguishability of PKEM.

Specifically, \mathcal{B} is an adversary against policy indistinguishability of PKEM. It runs \mathcal{A} and accurately simulates the game, $\mathbf{G}_{PKEM-DEM, \mathcal{A}}^{pbes-pol-ind-2a-cca2-cu}$, for \mathcal{A} . \mathcal{B} runs \mathcal{A} with the public-key pk . When adversary \mathcal{A} adds/corrupts a user, u_i , in phase 1, \mathcal{B} stores the user u_i and associated attributes in a list. For decryption queries in phase 1 \mathcal{B} does the following: 1) \mathcal{B} submits C_1 to its PKEM oracle and gets either a \perp or (pol, K_2) , 2) if \perp , it returns \perp to \mathcal{A} , 3) else if $f(u, pol) = 1$ returns K_2 and $DEM.Decrypt(K_2, C_2)$ otherwise it returns \perp . On receiving m and pol_1 and pol_2 from \mathcal{A} , \mathcal{B} does the following: 1) verifies that all of the of the corrupted users u_i satisfy both pol_0 and pol_1 , 2) submits pol_1 and pol_2 to its PKEM oracle and gets back (K_2^*, C_1^*) where C_1^* encapsulates pol_b , 3) it computes $C_2^* \leftarrow DEM.Encrypt(m, K_2^*)$ and returns challenge ciphertext (C_1^*, C_2^*) . For user corruption requests in phase 2, \mathcal{B} verifies that the corrupted user satisfies both pol_0 and pol_1 . For decryption queries from \mathcal{A} in phase 2, \mathcal{B} responds similarly to phase 1 except that when $C_1 = C_1^*$, \mathcal{B} returns K_2 and $DEM.Decrypt(K_2, C_2)$ if $f(u, pol) = 1$ and \perp otherwise where u is submitted along with decryption query. When \mathcal{A} outputs a guess bit δ , \mathcal{B} outputs its guess bit $b' = \delta$. Clearly, \mathcal{B} accurately simulates the PKEM-DEM game environment for \mathcal{A} . Therefore we have

$$\begin{aligned}
Pr[\mathbf{G}_{PKEM, \mathcal{B}}^{pkem-pol-ind-cca2}(k) = b] &= Pr[b' = b] = Pr[\delta = b] \\
&= Pr\left[\mathbf{G}_{PKEM-DEM, \mathcal{B}}^{pkem-dem-pol-ind-2a-cca2}(k) = b\right] \\
\Rightarrow \mathbf{Adv}_{PKEM, \mathcal{B}}^{pkem-pol-ind-cca2}(k) &= \mathbf{Adv}_{PKEM-DEM, \mathcal{B}}^{pkem-dem-pol-ind-2a-cca2}(k)
\end{aligned}$$

Thus any advantage \mathcal{A} has in breaking policy indistinguishability of PKEM-DEM

is translated into advantage in breaking policy indistinguishability of PKEM. \square

4.6 Application Design Issues

We now discuss some design challenges that need to be addressed when developing applications with PBES and certain properties of PBES that potentially limit PBES' suitability for certain kinds of applications.

Trust Model for KDCs An important issue in deploying PBES for an application in a distributed setting is identifying a trust model, *i.e.*, identifying KDCs that an object encryptor can trust to distribute the object decryption key to appropriate recipients. A simple trust model is for all users to trust a single KDC to appropriately distribute decryption keys for their objects. However, a more scalable model would be to have multiple KDCs that users can trust for different sets of users and objects. For example, every domain may have its own KDC that is responsible for distributing message decryption keys to users within the domain appropriately as was proposed for IBE [68]. The choice of trust model varies from application to application and we believe that a domain-based approach will be suitable for many applications. This trust model is similar to that of other policy-based encryption schemes that trust key distribution servers in recipient domains to distribute keys to appropriate users.

KDC Public-Key Distribution and Revocation Another challenge is distributing authentic public-keys of KDCs and timely revocation information for those

keys. Recently, schemes to distribute keys via DNS have been proposed [68, 41] and such an approach would be suitable for distribution of KDC public keys. While these schemes do not provide strong security guarantees (e.g., they are vulnerable to DNS cache poisoning attacks), wider deployment of the secure version of DNS, namely, DNSSEC [3], will improve the security.

Policy Specification Language and Enforcement Engine Another issue is the identification, deployment and use of an appropriate policy specification language and enforcement engine. The language should be sufficiently expressive and the engine should be user-friendly, have strong performance and ideally should have formally verified assurances. Furthermore, standardization of tools can significantly aid in achieving software and interface compatibility when exchanging objects across domains. While there are a range of potential languages and tools we believe that tools based on XACML are a good candidate for PBES. These tools have been used to specify flexible policies in various types of access control systems³. In particular, they allow us to specify flexible policies of the types described above including the use of attribute based expressions with string and numerical attributes that may be combined with AND, OR and NOT operands as well as context variables (e.g., time of day). The XACML language has been standardized and there exist several implementations of engines for policy verification among which Sun's implementation is quite popular and Margrave has been formally verified [31].

³<http://www.oasis-open.org/committees/download.php/27298/xacmlRefs-V1-84-1.htm>

Key escrow Given the PKEM part of any encrypted object the KDC can always decrypt it to reveal the DEM decryption keys for the object. Therefore, our system provides key escrow service via the KDC for the symmetric object keys. Note that in regular mode of operation the KDC never sees the encrypted objects, just the encrypted object DEM keys. This kind of key escrow is common to several encryption systems that minimize encryption key distribution tasks. For example, in IBE [16] or CP-ABE [10] the PKG can always generate a private key for any given public key, however, under normal mode of operation the PKG never sees encrypted messages. The difference being that a PKG provides escrow for private keys while we provide escrow for symmetric keys. This key escrow property may limit the applicability of our scheme in certain applications that demand strong end-to-end confidentiality assurances. For example, exchange of sensitive content between two parties that know each other. In general, in large systems where senders wish to send confidential messages to a set of (possibly unknown) recipients that satisfy a given policy such strong assurances may not be needed.

Online nature Since recipients need to contact the KDC for every decryption, the KDC needs to be always online and have adequate throughput to support this mediated decryption. This property of being always online may limit the applicability of our scheme for applications that have an offline nature. For example, exchange of secure messages in a sensor network that have limited connectivity to CAs/KDCs. However, we observe that many distributed applications being developed and deployed today have a largely online nature in that users usually access objects over

the network. We argue that in such an online world many of these applications can accommodate the presence of an online KDC. Furthermore, in applications where auditing and accountability is needed, mediated decryption offers an ideal opportunity for providing such capabilities. In Section 5.2.3 we study the throughput of a prototype implementation of a KDC and demonstrate that adequate throughput can be achieved with today's general purpose compute systems.

Arguments that support the need for online key generation/ distribution servers have also been implicitly made by other policy encryption systems such as IBE and CP-ABE for PKGs to be available to generate and distribute private keys to users on a regular basis as these system employ short-lived keys to support revocation capabilities. Other systems such as PEAPOD [42] require recipients to contact an online CA for *every* object as well. In all these systems a security concern that arises from their online nature is the potential compromise of the KDC/CA/PKG private keys. To minimize this possibility, threshold decryption and key generation functions can be deployed over multiple servers to provide both increased intrusion tolerance and availability [6, 35, 39].

Chapter 5

Application Integration and Evaluation

In this chapter we demonstrate the use of CP-ASBE and PBES schemes proposed in this work by integrating them with practical applications. We also undertake a preliminary performance evaluation of the proposed schemes. Specifically, in Section 5.1 we illustrate the use of CP-ASBE by employing it to provide message confidentiality in a novel messaging system that we proposed, namely, Attribute-Based Messaging (ABM). In Section 5.2 we illustrate the use of PBES by employing it to enable conditional sharing of sensitive sensor data among the operators of the Power Grid.

5.1 Attribute-Based Messaging

Attribute-Based Messaging (ABM) enables messages to be addressed using attributes of recipients rather than an explicit list of recipients or mailing lists with pre-defined members. Such attributes can be derived from any available source, including enterprise databases, and dispatched as Internet electronic mail (email) messages or other types of messaging. For example, a message about a restricted fellowship opportunity could be emailed to all of the female graduate students in engineering who have passed their qualifying exams. Such dynamic lists provide three primary advantages over static mailing lists: efficiency, exclusiveness, and intension-

ality. *Efficiency* means that messages are more likely to reach only the recipients that care about them. For example, if a message for the faculty on sabbatical is sent only to the ones with that attribute rather than the general faculty mailing list, then six sevenths of the faculty will be spared an unwanted message. *Exclusiveness* means that a sensitive message excludes parties that should not receive the message. For example a message from the dean to the untenured faculty in a given department to solicit feedback on the clarity of tenure standards provided by the department's senior faculty might have this feature. *Intensionality* means that an address *describes* the recipients rather than listing them. For example, a message to the attending and primary-care physicians for Sara Smith saves the sender the need to know the names or addresses of the recipients. ABM has applications in enterprises using the enterprise database for internal messages. It provides benefits for Customer Relationship Management (CRM) and similar circumstances where a sender needs targeted messaging to clients, members, and so on. It also has applications to alert messaging, like health alert networks.

However, to achieve its full potential, an ABM design must resolve significant security concerns. Access control and confidentiality are two such concerns. ABM messaging becomes more beneficial as it exploits richer attribute information. However, user attribute information is sensitive and allowing anyone and everyone to target messages based on any or all user attributes could increase spam and violate the privacy of recipients. For example, who, if anyone, should be able to target a message to all of the employees who earn more than \$150,000? It is possible to appoint an ABM super user as the only party that can send ABM messages, but a more

scalable solution would regulate the rights of potential senders based on a general ‘address authorization’ policy. On the other hand, it is not obvious how to do this, since solutions like Access Control Lists (ACL) are likely to be unmanageable. As for confidentiality, current email systems offer the ability to encrypt messages end-to-end using public keys. For sensitive messages this provides valuable protection against compromised email relays or eavesdropping relay administrators. However, ABM cannot directly use this solution since message senders may not have an explicit list of the recipients of a message and, even if the recipients were known, it is probably not scalable to collect all of the necessary certificates to provide encryption for each recipient.

We addressed the first concern using Attribute-Based Access Control (ABAC) in [12] and focus on addressing message confidentiality in this chapter. Specifically, we address the message confidentiality challenge by employing CP-ABE to encrypt messages using attributes. Translating this to ABM system, a sender can encrypt his message using attributes so that only users that satisfy the specified attributes can decrypt the message. This approach has two advantages. First, a message sender can encrypt his message directly (end-to-end) to the recipients without having to trust intermediate servers with the message contents. Second, the attribute expression used to target the message can also be used to specify the users that could decrypt the message. We show how CP-ABE is naturally integrated into an intra-enterprise ABM architecture and perform a preliminary evaluation of CP-ASBE against BSW CP-ABE scheme in this architecture.

5.1.1 CP-ASBE for ABM

ABM assumes a context where there is a set of attributes that can be accessed and used for authorization and messaging. In particular, any enterprise has attribute data about its employees in its databases. We will refer generally to the parties who can send or receive ABM messages based on these attributes as *users*. A user can have zero or more values for any attribute¹. For example, a university might have the following attribute data on a user:

UserID	=	user089
Position	=	Faculty
Designation	=	Professor
Department	=	Computer Science
Department	=	Mathematics
Course Teaching	=	CS219
Course Teaching	=	CS486
Course Teaching	=	MATH523
Date of Join	=	06/24/1988
Annual Salary	=	80,000

In the above example the user is affiliated with two departments and is teaching three courses. So he has multiple values for those attributes. In general, the attribute value pairs used in the system can be classified as, 1) *boolean*: those with a yes or no value, 2) *enumerated*: those with multiple non-numerical values and 3) *numerical*: those with multiple numerical values. This attribute information may not all be available in one centralized database but, instead, might be distributed over multiple databases that are managed by different units of an enterprise. An ABM system makes use of this information, abstracted as user attributes, to dynamically create

¹We restrict users from having multiple numerical values for the same attribute during our performance evaluations as BSW CP-ABE encryption system cannot handle multiple numerical values for a given attribute.

recipient lists. To have this attribute information available to the ABM system ABM envisions the use of a data services layer that presents a view of the attribute data after extracting it from the disparate databases. Some attributes are verified or established by the enterprise, like immigration status, age and salary, whereas others may be maintained by users, like a list of hobbies. In this work we focus on the former attributes.

The CP-ASBE (and BSW CP-ABE) scheme considers attributes simply as labels, *i.e.*, arbitrary strings, rather than as *attribute, value* pairs as described above. Furthermore, the underlying mathematics can only check for equality of strings and hence only equality of strings is supported by default in encryption policies. However, the three types of attribute value pairs used by ABM system are supported as follows. Boolean attributes are represented using just the attribute name since only positive Boolean attributes are ever used as only monotonic policies are supported. Enumerated attributes are converted into multiple unique strings by concatenating the attribute name, a delimiter and one of the attribute values. In order to support numerical attributes and allow numerical comparisons in policies, CP-ASBE (and BSW CP-ABE) uses strings to represent individual bits of the numerical value. That is, numerical values are represented using a bag of strings, one for each bit of the value. For example, a 3-bit numerical attribute *Level* with value 4 is represented using the following strings: *Level:1***, *Level:*0**, and *Level:**0*. Now a policy with a numerical comparison can be represented using equalities on the strings representing bits. For example, the policy $Level \geq 2$ can be translated as *Level:1** OR Level:*1**.

An ABM system has three primary types of policies as described below.

1. The delivery policy is a sender-defined policy that specifies the set of users his message is targeted for. This is the ‘ABM address’ associated with a message. The message is routed only to users who have an attribute-set that satisfies this policy.
2. The address authorization policy controls the ability of a user to target messages using an ABM address. This is a system-wide policy and specifies which users have permission to target messages to a given attribute based on their own attributes. Conceptually, this policy determines the set of users that have permission to send messages to a given ABM address and the set of ABM addresses to whom a given user is allowed to send messages. This policy therefore controls access to the system.
3. The encryption policy is another sender-defined policy that specifies which users will have the ability to decrypt an encrypted message. The encryption policy associated with an encrypted message defines the combination of attributes needed to decrypt the message. This would typically be the attributes held by the recipients as specified by the delivery policy, but there are cases where key management is improved by allowing the delivery policy to be a subset of the encryption policy.

Table 5.1 describes the language used for ABM addresses and address authorization policy.

Table 5.1: Grammar for ABM Addresses and Rules

	a	\in	$Attribute$
	v	\in	$Numerals \mid Strings$
Delivery Policy	R	$::=$	c
Condition	c	$::=$	$l \mid (c \text{ or } c) \mid (c \text{ and } c)$
Literal	l	$::=$	$(a \text{ rel } v) \mid (v \text{ rel } a \text{ rel } v)$
Relation	rel	$::=$	$< \mid > \mid = \mid \leq \mid \geq$
Authorization Rule	S	$::=$	$l \leftarrow c$

In ABM, the encryption policy is effectively same as the ABM address (delivery policy) that routes the message to recipients. However, while the ABM system and thus the delivery policy use *attribute, value* pairs, CP-ASBE (and BSW CP-ABE) scheme considers attributes simply as labels, *i.e.*, arbitrary strings. Thus we implemented a policy translator that converts a given delivery policy into a valid encryption policy by converting all attribute value pairs, except for numerical attributes (numerical attributes are automatically converted by the CP-ASBE and BSW CP-ABE implementations), in the delivery policy into unique attribute strings as described above.

5.1.2 ABM Architecture

Figure 5.1 illustrates the architecture of our ABM system and its associated security system, which strongly influences the overall structure. The ABM system comprises a Policy Specialization Server (PSS) to authenticate and help users compose policy compliant ABM addresses, a Policy Decision Point (PDP) with the address authorization policy, an attribute database, an ABM server associated with an enterprise Mail Transport Agent (MTA) that resolves ABM addresses to recip-

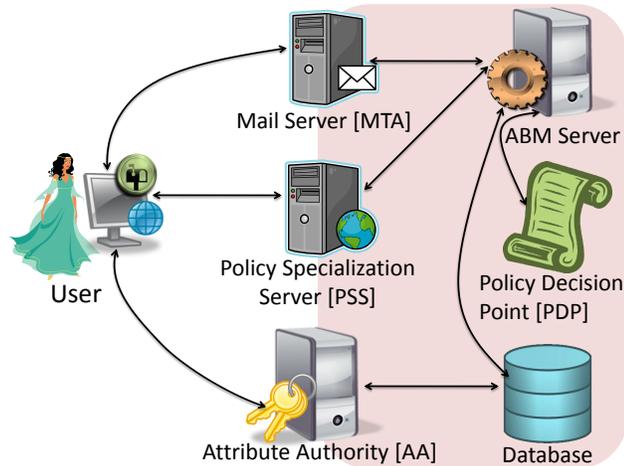


Figure 5.1: ABM Architecture

ient lists and mediates other components, and an Attribute Authority that issues attribute keys to the users. These components provide an infrastructure for three attribute-based policies for messaging, the system-wide address authorization policy, the user-defined delivery policy and the encryption policy.

The *Policy Specialization (PS) Path* authenticates the user, evaluates his attributes from the database with the policy decision point (PDP), and retrieves the address authorization policy specialized to the user. This eight step communication is represented by solid lines in Figure 5.2. In the first step, PS1, the user logs into the PSS. The PSS uses the enterprise authentication infrastructure to authenticate the user. Next at PS2, the PSS sends the user’s information to the ABM server and requests the specialized address authorization policy for the user. In steps PS3 and PS4 the ABM server queries and retrieves the user’s attributes from the attribute database. In step PS5 the ABM server sends the user’s attributes to the PDP and requests for the specialized policy. The PDP then evaluates all the

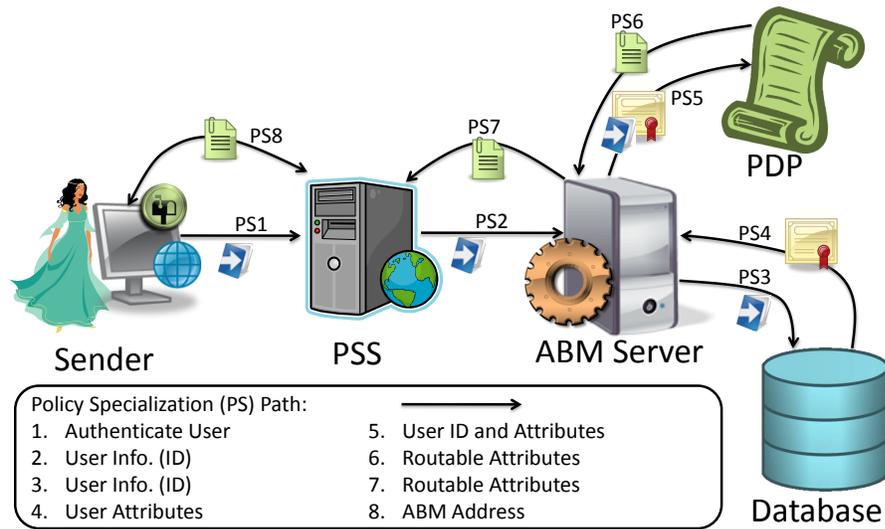


Figure 5.2: Policy Specialization Path

address authorization rules in the policy against the user's attributes to form the specialized address authorization policy. In this case it is a list of attributes that the user is allowed to use in his delivery policies or ABM addresses. In step PS6 it returns the routable attributes (literals) from the specialized authorization policy that the user can route on. The ABM server then returns the specialized authorization policy to the policy specialization server in step PS7. In step PS8, the policy specialization server provides an interface to the user to create a delivery policy by combining the routable address literals in the specialized authorization policy with ranges and boolean connectives as permitted. This is then saved by the user as the ABM address (delivery policy).

In the *Messaging (MS) Path*, users send and receive ABM messages using any standard MUA (dashed lines in Figure 5.3). The ABM address is translated to

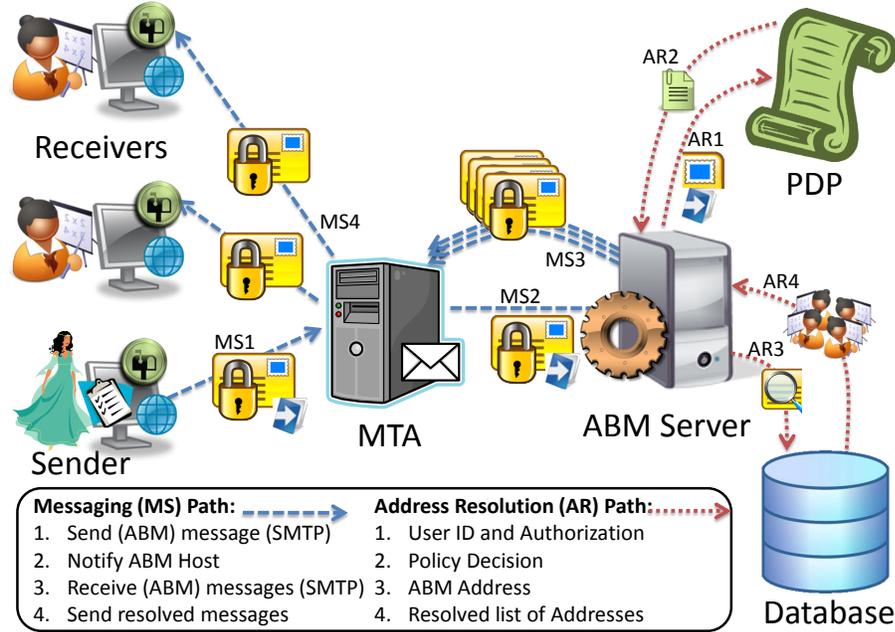


Figure 5.3: Messaging and Address Resolution Path

a valid encryption policy using the policy translator described in Section 5.1.1. A user composes a message and encrypts the body of the message using the encryption policy.

The delivery policy (or ABM address) is included in the message as an attachment. The message with encrypted body and ABM address as the attachment is signed using S/MIME [60, 61] for sender authentication. The user then sends the message to a pre-specified email address such as `abm@localdomain.com` in step MS1. The enterprise MTA is configured to notify the ABM server when it receives a message for the pre-specified address as shown in step MS2. After processing the message, as described in address resolution path below, the ABM server invokes the enterprise MTA in step MS3 to deliver the message to a list of recipients as specified

by the ABM address. Each receiver gets her message in her inbox in step MS4.

In the *Address Resolution (AR) Path*, the ABM server processes messages to authenticate the sender, determine whether the sender is authorized to target the message based on the associated delivery policy, and determine the recipients defined by the delivery policy (dotted lines in Figure 5.3). Upon receiving the message, the ABM server: 1) verifies the S/MIME signature on the message to authenticate the user, and 2) queries the attribute database for the sender's attributes. In step AR1, the ABM server checks with the PDP that the sender is authorized to send the message to the ABM address included in the message. In step AR2, the PDP evaluates the delivery policy for accessing the attributes contained in the ABM address against the sender's attributes and responds in the affirmative only if the user is allowed access to all attribute literals in the ABM address. The ABM server then resolves the ABM address to a list of email addresses by querying the attribute database in steps AR3 and AR4.

The *Attribute Keying (AK) Path* describes steps of the AA, which is similar to a certificate authority and supports keying needs of users such as attributes and S/MIME certificates. After receiving an encrypted message, if the user does not have a current set of keys to decrypt the message, she requests them from the AA (dashed-dotted lines in Figure 5.4). A user authenticates to the AA in step AK1. The AA sends the user information (*e.g.* user id) to the enterprise database in step AK2. The database responds with the most current information about the user's attributes in step AK3. With the attribute set the AA gets from the database, it generates cryptographic attribute key set using CP-ASBE scheme after converting

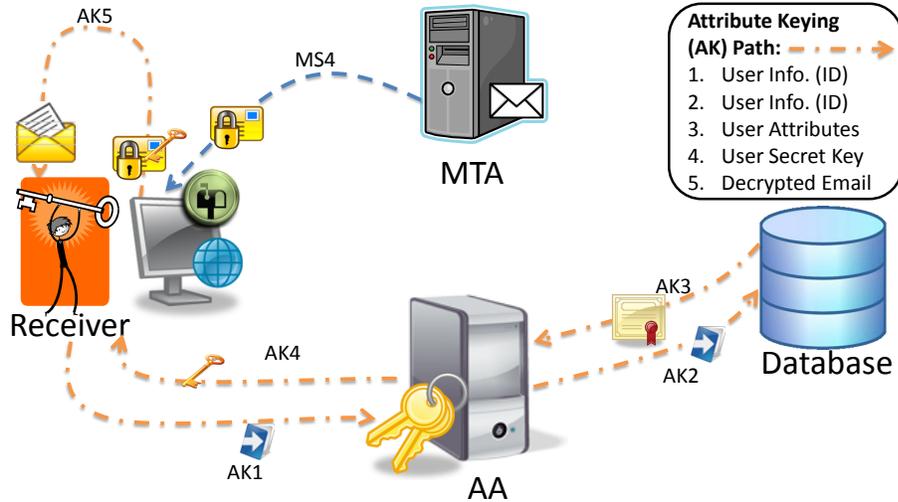


Figure 5.4: Attribute Keying Path

attribute, value pairs into attribute strings as described in Section 5.1.1. AA sends the generated CP-ABE key back to the user over a secure channel in step AK4. The user can now decrypt her message using this key in step AK5.

5.1.3 Experimental Evaluation of CP-ASBE in ABM

To evaluate the architectural framework presented in Section 5.1.2, we implemented a prototype ABM system in [12, 11]. In this section we describe how this prototype is used to evaluate CP-ASBE. We first describe the experimental setup and then present the results from the experimental evaluation of CP-ASBE and compare them with those obtained when using BSW CP-ABE.

A two-level CP-ASBE scheme provides better functionality over CP-ABE schemes in terms of, 1) better supporting compound attributes and 2) supporting multiple numerical value assignments for a given attribute in a single key. In

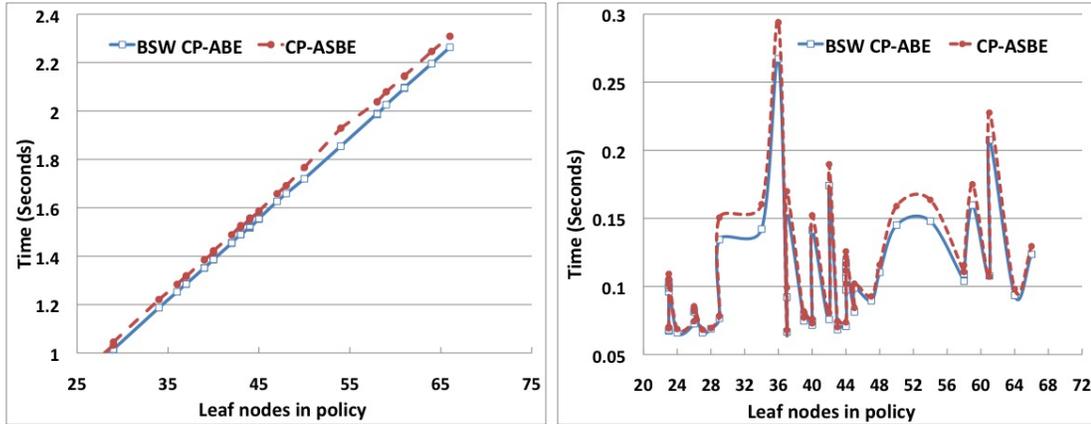
order to gauge the cost of this additional functionality we compared the encryption, decryption and key generation times using ABM addresses that were used to evaluate the ABM prototype after converting them into encryption policies. We now describe how the ABM addresses and user keys were generated.

Attribute Distribution and Database Population We populated a SQL database with 60,000 users and assigned attributes to them in the following manner. The system had a total of 100 attributes and about half of them are numerical attributes. The distribution of attributes in the user population affects the number of recipients a given ABM address resolves to. The number and type of attributes a user has also affects the attribute-key generation time. Users were assigned an attribute based on the incidence probability of that attribute. For example, if an attribute has an incidence probability of 0.1 then 10% of the user population is assigned that attribute. For our test database, most of the attributes (80%), had a probability of incidence that ranged from 0.0001 to 0.01, 10% had a probability of incidence that was between 0.5 and 0.9 and the remaining 10% had the probability close to 1. This distribution allowed a big range in the number of recipients per message, and, intuitively, this distribution also reflects organizations where all the users have some common attributes and rest of the attributes are sparsely distributed in the population.

Encryption Policy Generation ABM addresses served as encryption policies after appropriate translation by our tool. The complexity of an ABM address affects

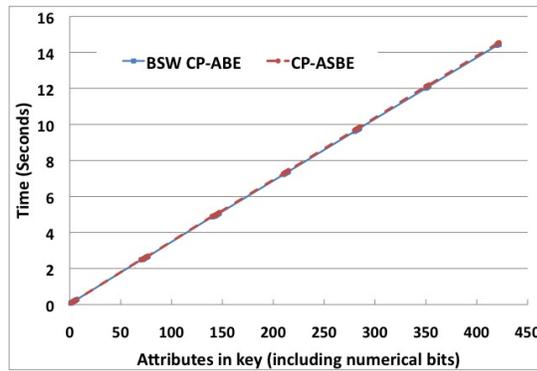
the performance on the address resolution path by affecting both the number of recipients it resolves to and the database query resolution time. It also affects the encryption and decryption latencies as ABM addresses. We wrote a probabilistic ABM address generator using Java, which created uniformly random ABM addresses of varying complexity in a disjunctive normal form. Each ABM address consists of a number of *terms* combined with the *OR* operand. Each term consists of a number of literals (as defined in the grammar of Table 5.1) combined with the *and* operand. Specifically, we varied the number of terms for a given ABM address between one and five (chosen uniformly randomly) and the number of literals in each term between one and three (also chosen uniformly randomly). Each literal was randomly assigned an attribute from the routable list of attributes of the message sender. These addresses were then translated into encryption policies. The resulting policies had the number of leaf nodes ranging from 23 to 66 (including the “bag of bits” representation of numerical attributes).

User Key Generation For each encryption policy, a representative set of keys that satisfy the policy are generated and used for decryption. Specifically, 1) a key is generated for each conjunctive clause in the policy such that it satisfies the clause and 2) a key is generated for each combination of conjunctive clauses in the policy such that the key satisfies all the clauses in the combination. The generated keys had boolean attributes, ranging from 1 to 422, *i.e.*, including the “bag of bits” representation for numbers with 64 bits used to represent each integer.



(a) Encryption Time

(b) Decryption Time



(c) Key Generation Time

Figure 5.5: Encryption and Decryption Times

Results For encryption, decryption and key generation when using BSW CP-ABE we used the CP-ABE toolkit (available at <http://acsc.csl.sri.com/cpabe/>). For encryption, decryption and key generation when using CP-ASBE we used the CP-ASBE toolkit that we developed by extending the CP-ABE toolkit as described in Section 3.6. Both implementations used a 160-bit elliptic curve group constructed on the curve $y^2 = x^3 + x$ over a 512-bit field. Decryption time for a policy is the average of decryption times with all the keys generated for that policy as described above. Experiments were run on a Linux box with quad core 3.0Ghz Intel Xeon and 2GB of RAM.

Key generation, encryption and decryption times are shown in Figure 5.5. As expected, key generation time was found to be linear in the number of attributes in the key, and CP-ASBE imposed very little overhead over BSW CP-ABE. On an average, CP-ASBE imposed $18ms$ overhead per numerical attribute, *i.e.*, per set, in the key and no overhead when there are no numerical attributes. To put this overhead in perspective, generating a key with 2 numerical attributes (and 145 boolean attributes in total) took $5s$ seconds when using BSW CP-ABE scheme and $5.035s$ when using CP-ASBE scheme. Encryption time is also, as expected, linear in the number of leaves in the policy tree, and CP-ASBE imposed very little overhead when compared to BSW CP-ABE. On an average, CP-ASBE imposed $8.3ms$ overhead per translating node in the policy. Since decryption time is dependent on both the structure of the policy tree and the key used for decryption, it varied significantly even for a given policy size. However, in this case too CP-ASBE scheme imposed very little to no overhead over BSW CP-ABE, $6.7ms$ on average. Overhead results are consistent with our efficiency analysis and performance numbers in general are consistent with those reported in [10].

5.2 Context-Sensitive Data Sharing in the Power Grid

The North American electric power grid is a highly interconnected system hailed as one of the greatest engineering feats of the 20th century. However, increasing demand for electricity and an aging infrastructure are putting increasing pressure on the reliability and safety of the grid as witnessed in recent blackouts [73, 29]. Fur-

thermore, deregulation of the power industry has moved it away from vertically integrated centralized operations to coordinated decentralized operations. Reliability Coordinators (RCs) are tasked by Federal Energy Regulation Commission (FERC) and North American Electric Reliability Council (NERC) with overseeing reliable operation of the grid and providing reliability coordination and oversight over a wide area. Balancing Authorities (BAs) are tasked with balancing load, generation and scheduled interchange in real-time in a given Balancing Authority Area (BAA). BAA is a geographic area where a single entity balances generation and loads in real-time to maintain reliable operation. BAA are the primary operational entities that are subject to NERC regulatory standards for reliability. Every generator, transmission facility, and end-use customer is in a BAA.

Currently, sensor readings from substations in utilities² are sent via a communication network to the Supervisory Control And Data Acquisition (SCADA) systems in the local BA that controls the system and to the RC that oversees reliable operation of the system. There are operations taking place at various time granularities to keep the power system stable and reliable. Among the frequent operations protection and control mechanisms at substation operate at the granularity of milliseconds, state estimators and contingency analysis in BAs and RCs operate at the granularity of minutes and hourly and day ahead power markets run by RCs operate at the granularity of hour and day respectively.

In order to improve the reliability of the power grid while meeting the in-

²In this paper the term 'utility' is used to refer to power grid entities in a broad sense including generator owners/operators, transmission owners/operators, distributors and load serving entities

creased power demand, the industry is moving towards wide-area measurement, monitoring and control. The Department of Energy (DOE), NERC and electric utility companies formed the North American SynchroPhasor Initiative (NASPI) (www.naspi.org) with a vision to improve the reliability of the power grid through wide area measurement, monitoring and control. It's mission is to create a robust, widely available and secure synchronized data measurement infrastructure with associated monitoring and analysis tools for better planning and reliable operation of the power grid. NASPI envisions deployment of hundreds of thousands of Phasor Measurement Units (PMUs) across the grid that pump data at 30 samples/second to hundreds of applications in approximately 140 BAAs across the country. PMUs are clock synchronized (through GPS) sensors that can read current and voltage phasors at a substation bus on the transmission power network. Phasor Data Concentrators (PDCs) at substations or control centers time align the data from multiple PMUs before sending them to applications. PMUs give direct access to the state of the grid at any given instant in contrast to having to estimate the state as is done today. Figure 5.6 shows a high-level architecture envisioned for PMUs. Applications envisioned to utilize this data have varying requirements. Open loop control applications like state estimation have critical time alignment requirements while post event analysis applications like disturbance analysis have critical accuracy and message rate requirements. Feedback control applications like transient stability control have critical latency, availability, accuracy, message rate and time alignment requirements [28].

While utilities are currently mandated to share operational data with their

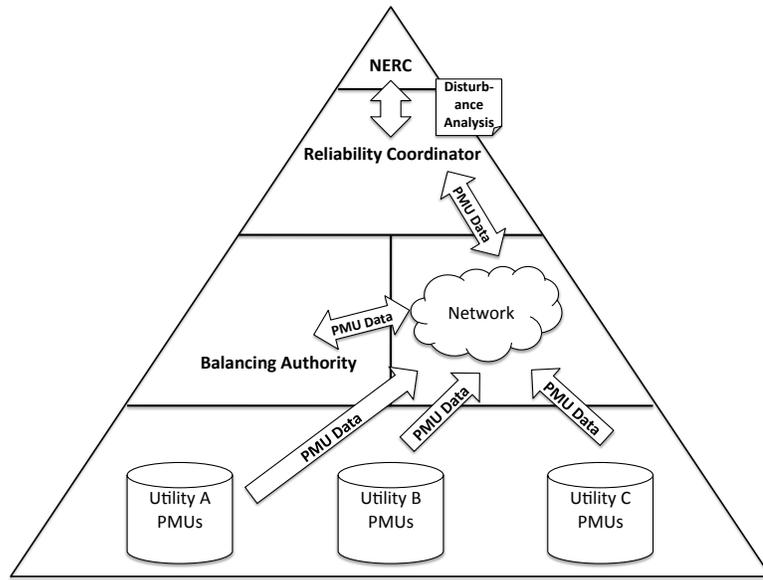


Figure 5.6: Proposed NASPI PMU Architecture

local BA and RC they are not required or expected to share data with other utilities. This is because, while the utilities have to cooperate with each other to operate the grid safely and reliably they are also business competitors. Furthermore, this data can reveal a fine grained view of a utilities network and the current state of that network. In the wrong hands the former can make the utility a target of attacks and the latter can affect the wholesale electricity markets and as a consequence the utility itself adversely. Another consideration hampering data sharing is the concern of utilities that they might open themselves up for continuous compliance monitoring. However there is mutual benefit in sharing PMU data widely as it will help in operating the grid safely and reliably and in avoiding overloading, outages, brown-outs and blackouts [73, 29]. Sharing PMU data will also help in planning, post disturbance/event analysis [29] and for research and development purposes.

Currently two pilot deployments each with about 75 PMUs exist in Eastern [30] and Western [21] Interconnects. There is need for a framework that provides for secure and flexible data sharing before a wide area full scale deployment of PMUs can be realized [28]. While we discussed North American power grid above, similar data sharing problem exists in other power grids such as that of Australia, Europe and Japan that are either in the process of deregulation or are already deregulated. The use of PMUs for wide area monitoring and control is also being considered in those grids.

5.2.1 Requirements

Given the sensitive nature of the data and the reluctance of utilities to share data, realizing wide area data sharing poses many challenges. First, establishing pair-wise trust between all the entities in a wide area is a $O(n^2)$ problem and does not scale. Second, while the system is inherently transitive, *i.e.*, highly interconnected where a local disturbance can have impact over a wide area, trust relationships are not always transitive. Third, data is usually shared on a need to know basis and it is not known in advance who might be needing the data, *e.g.*, for applications like post event analysis.

In studying the data sharing needs in the power grid we argue that a natural approach is to enable conditional access to data whereby utilities make data available to each other based on their ability to satisfy policies. Any solution requires a viable architecture, a data protection mechanism and a flexible policy enforcement

mechanism. Specifically a desirable solution should satisfy the following requirements:

Data sharing with multiple recipients Support data sharing with multiple recipients all of whom may not be known in advance. In the power grid for example, when data is to be shared based on prevailing or past conditions in the grid, *e.g.*, post event analysis applications like disturbance analysis, it is not possible for the data owner to know ahead of time with whom or how many entities the data might need to be shared. For example, consider that the tripping of a line in Ohio caused a disturbance that eventually lead to the August 2003 blackout - the largest in the North American Power Grid's history [73, 29].

Flexible policy specification and enforcement Data owners should be able to specify and associate flexible policies with data in a secure manner such that only entities that satisfy the policies can access the data. These policies may be context-based in that data may only be shared based on the current state of environment. Furthermore, the context-based policies may be such that the data owner may or may not be able to verify the satisfaction of such policies on his own. For example, voltage disturbances in the power grid are only visible in the vicinity of the event, which may be outside the data owner's range of observability, but their effect might propagate over a wide area eventually.

Data exchange on open and untrusted networks Given that the data sharing is needed between many entities dispersed over a wide geographic area requiring a

trusted or even a closed network for data sharing is impractical and very expensive.

Protect data and policy secrecy Given the sensitive nature of the data and the need for sharing over open and untrusted networks data secrecy must be protected. Furthermore, in open and untrusted networks the secrecy of policies associated with the data should also be protected from general public as they might reveal sensitive information about the data and since the data owning organizations would consider their policies themselves to be confidential. In some cases the policies need to be kept secret even from an authorized recipient as the policies might reveal who else might have access to the data thereby revealing business relationships of the data owner which is undesirable.

Security Any solution should provide adequate security for both the data and associated policies. Specifically it should secure them against active and colluding adversaries.

Efficiency and Compatibility Any solution should be efficient in key management including revocation and should have low communication and computation overheads. Furthermore, the solution should be compatible with other infrastructure components.

5.2.2 PBES for Context-Sensitive Data Sharing in the Power Grid

In this section we illustrate how PBES is used to enable policy based data sharing in the power grid using an example usage scenario. First, we note that

Table 5.2: Example of Policy Elements

Policy Element	Example
Identity	Email address, Distinguished Name
Group or Role	Transmission System Operator, Reliability Engineer
Attribute	Certified Dispatcher
Context	Location of voltage disturbance, Status of a relay, Time of the day

policies in our system are arbitrary strings that can be parsed and enforced by the KDC. Therefore, they are very flexible in nature. Policy elements of interest for object encryption and in particular for data sharing in power grid include: 1) identities where recipients must demonstrate ownership of identifiers, 2) groups or roles where recipients must demonstrate membership to a group or role, 3) attributes where recipients must demonstrate ownership of attributes that satisfy an attribute expression, and 4) context where the KDC must verify environmental properties. Policies may combine any of these elements and some example elements in the power grid are shown in Table 5.2.

As an example, consider a Utility A under the jurisdiction of an RC B. While Utility A is not willing to share its data with all other utilities in the area under normal circumstances, it might find that it is in its interest to share that data with some of them when they are experiencing a combination of events that might potentially lead to a voltage collapse especially if no coordinated mitigation actions are taken. Possible combination of events for voltage collapse are identified by system planning static load flow analysis undertaken by NERC or the RC B. Specifically, the policy of utility A for sharing data with any *Utility X* is as follows:

Grant Access if

*(Reliability Engineer in Utility X) AND (Utility X in RC B) AND
(Overloaded Tie Line between Utility X and Utility A) AND ((Below
Critical Reactive Power Reserves in Utility X) OR (Reactive Limiters active in Utility
X))*

Utility A associates this policy with the data and encrypts it using the PKEM-DEM scheme entrusting access control enforcement to (local) RC B, *i.e.*, RC acts as the KDC. It then posts this data on its public data repository (which may use coarse-grained access control, for example, to limit write operations). If and when the Transmission System Operator in utility C in the neighboring BAA notices an overload on the tie line connecting utility C with A and the Generation System Operator notices low reactive power reserves or reactive limiters turning on they initiate mitigation procedures along with the Reliability Engineer. Reliability Engineer obtains the relevant encrypted data from utility A's repository based on the meta data associated with encrypted objects. Example of useful meta data are the start time and end time of data samples contained in the encrypted object and coarse grained PMU location information. Reliability Engineer then submits the encrypted data key to the RC for decryption. RC upon verifying that the associated policy is satisfied returns the data decryption key. Note that RC having a wider view of the grid than Utility A is able to verify the occurrence of specified conditions in Utility C. Reliability Engineer may repeat this action with all utilities with which their organization shares a tie line that is overloaded. He may or may not be successful in obtaining data based on the current policies of individual utilities.

Reliability Engineer then feeds the data obtained into his contingency planning tool and coordinates the mitigation plan with data sharing utilities based on the results.

Utility A might also have additional time constraints in its policy limiting the data shared to a time window starting 30 minutes before the event (*i.e.*, tie line overload) and ending 30 minutes after the conditions are mitigated. We omitted this detail in the policy example above for brevity. Furthermore, Utility A might be sharing the data from its sensors with different entities under different conditions. So in practice the policy associated with the data will be a complex policy consisting of many sub policies similar to the one in the example above. So it is necessary to preserve policy secrecy from legitimate recipients (apart from general public) to prevent a recipient satisfying one sub policy to obtain the data from knowing other sub policies. While PBES provides policy secrecy from general public and from legitimate recipients it is possible to gain some information about the policy by gaming the system and from side channels such as traffic patterns. Some of this information leakage to outsiders can be mitigated by using secure TLS channels to upload and download data from the data repositories but a full analysis of policy information leakage is out of the scope of this paper.

Choosing RCs to act as KDCs to enforce access control on data owned by utilities under their jurisdiction has the following two advantages. First, the trust relationships of the RC with all the utilities under its jurisdiction are leveraged to enable data sharing between utilities without the need to establish pairwise trust. Currently RCs already administer Certificate Authorities (CAs) that issue certificates to users in the utilities based on the federated user identity databases at the

utilities that it has access to. Second, an RC is ideally suited to enforce certain context based policies that condition upon prevailing conditions in the grid, as in the example above, as it has a much wider view of the grid than any single utility. The environment/context attributes extracted from the current state of the power grid by the RC along with the federated identity and attribute databases that the RC has access to constitute the Attribute Database shown in Figure 4.1. In terms of key management, in our system data owners only need to obtain the public keys of KDCs in order to encrypt objects intended for any recipient that trusts those KDCs. In the power grid knowing the public keys of the dozen or so RCs suffices to reach all users registered in those RC domains. For data recipients we do not add any additional key management burden but we require recipients to contact the KDC for every decryption, which also provides support for immediate revocation because the KDC verifies policies for *every* object it decrypts. In systems where objects can potentially reside in repositories for a long time, immediate revocation provides effective policy compliance at the time of access.

While the RC is able to enforce the access policy it is unlikely to have the resources to manage the data itself. This is because RCs may oversee many BAAs, *e.g.*, Midwest Independent System Operator (MISO) manages 37 BAAs, and they might have to manage large amounts of data (tens of thousands of objects adding up to hundreds of petabytes) and enforce different access policies on data from different control areas and utilities. A more feasible solution is the utilization of data warehousing solutions whereby encrypted data with an associated (encrypted) access policy is posted on a semi-trusted storage facility. The facility may be trusted

to enforce coarse-grained access control such as limiting write operations to trusted utilities and ensure availability but should not be trusted for access to content; otherwise, it will become an attack target for access to all data [54]. So either utilities themselves might host repositories for data they are willing to share or utilize an external data warehousing facility to provide semi-trusted storage. Table 5.3 shows which power grid entities play the roles of the components in the PBES architecture presented in Figure 4.1.

Table 5.3: PBES Entities vs. Power Grid Entities

PBES Entities	Power Grid Entities
Data Owner/Sender	Utilities
Data Repository	Hosted by BAs/Utilities or Data Warehousing Providers
KDC	RC
Receiver	Utilities, BAs
Attribute Database	Environmental Attributes based on Power Grid State observed at the RC along with Federated Identity/Attribute Databases at utilities

5.2.3 Prototype Implementation and Performance

We have implemented the PBES system and the PKEM-DEM construction and measured its performance. The implementation is aimed at releasing an easy-to-use toolkit in the near future that allows for integration in distributed applications. The implementation is built using the Java Bouncycastle Library and its S/MIME and CMS Processors. These libraries and processors were chosen to allow for platform independence, flexible licensing of the toolkit and a simplified process for its standardization. Bouncycastle has an open source license, CMS is a well accepted standard for message encapsulation and S/MIME is a well accepted standard for

public key encryption for multi-part messages (typically used in e-mail systems).

The PBES implementation provides interfaces for the following components: 1) object encryption, 2) policy decryption and verification and 3) object decryption. KDC private/public keys are assumed to be pre-created (e.g., using RSA key generation tools) and installed. Using the provided KDC public key, the object encryption component expects as input two files – one providing the message and one providing the policy. It then encrypts these files using the PKEM-DEM encryption scheme. While the object encryption interface treats both files as arbitrary strings, we use XACML as the policy language in our system. To allow for the encryption and transmission of the XACML policy within the S/MIME processors, we use the *OtherRecipientInfo* type and value fields in S/MIME to specify the policy. The policy decryption and verification interface expects as input an S/MIME encrypted object with the PKEM format, the KDC private key, and an authenticated user identity. For authentication we require users to initiate a TLS channel and provide a username/password, which are checked against a salted password database. This component then contacts the Attribute Database, which in our case is a SQL server, using a SQL query with the authenticated identity. After receiving the attributes it uses the XACML engine (in our case Sun's Java implementation³) to verify the decrypted policy. If the policy is satisfied it releases the DEM keys over the secure channel. Finally, the object decryption component expects as input an encrypted file and a DEM key using which it applies the DEM decryption and outputs a file with the decrypted message.

³<http://sunxacml.sourceforge.net/>

Performance We instantiate our PKEM-DEM scheme using an RSA-based CCA secure KEM, RSA-KEM [67], and an OTCCA secure DEM, DEM1 [67, 27] (essentially symmetric encryption with a message authentication code) as shown in Figure 4.2. We use a sample XACML policy with rules that involve the combination of 10 different attributes each. We use boolean, string and numerical attributes as well as a range of operands including AND, OR and NOT. Note that we do not limit the number of attributes used in the system but just those used in each policy rule for this evaluation. Such policies intuitively match the complexity of policies that users can typically conceive of to protect data. Since PKEM-DEM is essentially a very efficient encryption/decryption scheme the only potential performance bottleneck for an application is the policy decryption and verification component. To evaluate the performance we measure the throughput of this component, which involves the following tasks: perform a RSA and an AES decryption, verify the MAC, setup and message exchange over secure TLS channel, fetch attributes from the Attribute Database and verify the policy. We use a 1024 bit RSA, 128 bit and 256 bit SHA-1, 128 bit AES, a SQL Attribute Database server located in the same subnet over a gigabit link and the Sun XACML engine placed on the same server as the KDC. The KDC server is a workstation with a 32-bit, 2.4 Ghz Pentium 4 processor while the database is a Windows 2003 Server with dual Intel Xeon 3.2GHz processors. Averaged over 10,000 runs the latency for the various tasks is as follows: *20.2ms* for the RSA and AES decryption, negligible for the MAC, *44.7ms* for the TLS channel, *40ms* to fetch attributes and *12.8ms* to verify the policy for a total of *117.7ms*. That is, we can support 510 requests/min.

Performance Comparison PEAPOD requires mediated access similar to PBES and while they do not implement their system, their calculations indicate a similar performance of hundreds of requests per minute for the mediation server. Both PBES and PEAPOD require mediated access while CP-ABE does not, therefore, it is hard to compare the performance of these systems. However, we would like to note that in practice CP-ABE also needs to be online for the simple reason that in any system with a large number of users the attribute private keys for individual users will expire with a distribution that pretty much requires the PKG to be online at all times to generate and distribute new private keys to the users. Furthermore, performance requirements for key generation are not trivial. Using the cp-abe toolkit [10] the average cost for generating 10 attribute private keys is 2.64 seconds where 3 attributes are numerical and 7 are boolean. In a system where a single PKG supports 50,000 users (essentially a medium size organization) with each user having 10 attributes all with a lifetime of one week (note that in the absence of revocation all CP-ABE private keys need to be short lived) it will take a PKG 36 hours to complete one round of key generation.

Application Analysis For the power grid data sharing application we envision one or more KDCs (for fault tolerance and/or load balancing) being maintained at each of the dozen or so RCs. These KDCs will serve hundreds of utilities across the grid with each RC focusing more on the tens of utilities in their jurisdiction. Based on an informal analysis of the data sharing needs in the grid we argue that each KDC being able to support 510 requests/min is sufficient to satisfy the requirements. Also,

the policy examples discussed above match the kind of policy complexity studied in the performance analysis above. However, a formal analysis of data sharing transaction patterns as well as a more comprehensive performance analysis taking into account networking and storage components will be the topic of future work.

Chapter 6

Conclusion

In this dissertation, we addressed the problem of secure, policy-based multi-recipient data sharing and presented two novel policy-based data sharing schemes. The first scheme we presented, namely, Ciphertext-Policy Attribute-Set Based Encryption (CP-ASBE) enables policy-based data sharing with multiple recipients without the need for trusted mediating servers to enforce the policy and thus minimizes trust liability. We showed that CP-ASBE is the first Ciphertext Policy-Attribute-Based Encryption (CP-ABE) scheme to provide the ability to organize attributes in user keys, after demonstrating the need for such ability in CP-ABE schemes in order for them to be practical and efficient. We also showed that its ability to organize attributes in user keys enables CP-ASBE to support, (1) naturally occurring compound attributes, (2) multiple numerical assignments for a given attribute in a single key and (3) efficient key management, all of which are properties needed in practical scenarios but are not provided by existing CP-ABE schemes. We showed that it achieves this versatility with very little overhead through efficiency analysis and performance evaluation of a prototype implementation integrated into a novel application we proposed, called, Attribute-Based Messaging (ABM).

The second scheme we presented, namely, Policy Based Encryption System (PBES), supports context-based policies and provides policy privacy. We showed

that while PBES incurs some trust liability by leveraging a trusted mediator, it achieves good properties of both mediated and unmediated solutions. We showed that PBES is a suitable candidate to enable context-based conditional sharing of sensitive sensor data among the operators of the Power Grid which improves efficiency and reliability of the Power Grid. We prototyped the system and demonstrated its performance to be reasonable.

While our proposed schemes achieve unique set of properties they do not achieve all the desirable properties of a scheme addressing secure policy-based, multi-recipient data sharing. Several open problems remain to be solved in this area. Supporting context-based policies without relying on trusted mediating servers is an important one. A related open problem is the lack of revocation in CP-ABE schemes and ABE schemes in general which is a significant hindrance to their adoption and deployment. While our CP-ASBE scheme and the key update scheme of [14] alleviate this problem to a certain extent more work remains to be done in this area. While we support flexible policies, and flexible attribute keys in our CP-ASBE scheme we do not provide policy privacy nor do we support multiple attribute authorities. The ABE scheme of [63] was extended to multiple authorities in [23, 50]. However, like [63] they can only support a single threshold gate for policies. The CP-ABE scheme of [10] was extended to multiple authorities in [53] but was limited to supporting disjunctive normal forms (DNF). Extending our work or proposing new schemes that can achieve policy privacy and support multiple authorities while retaining policy and attribute flexibility is an interesting open problem. CP-ASBE is shown to be secure in Generic Group Model. Designing schemes secure in stan-

standard model that achieve the flexibility of CP-ASBE is another direction of future work.

We envision two directions of future work related to PBES. First, the efficiency of PBES can be improved by using other encryption schemes such as the Tag-KEM/DEM framework [1]. Second, the practicality of PBES can be further explored by deeper integration with the power grid and by studying other real-world applications such as distributed file sharing.

Bibliography

- [1] Masayuki Abe, Rosario Gennaro, and Kaoru Kurosawa. Tag-KEM/DEM: A New Framework for Hybrid Encryption. *J. Cryptol.*, 21(1):97–130, 2008.
- [2] Sattam S. Al-Riyami, John Malone-Lee, and Nigel P. Smart. Escrow-free encryption supporting cryptographic workflow. *Int. J. Inf. Sec.*, 5(4):217–229, 2006.
- [3] R. Arends, R. Austein, M. Larson, and D. Massey. Resource Records for the DNS Security Extensions. Technical report, RFC 4034, March 2005.
- [4] Jean Bacon, David M. Eyers, Ken Moody, and Lauri I. W. Pesonen. Securing publish/subscribe for multi-domain systems. In Gustavo Alonso, editor, *Middleware*, volume 3790 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2005.
- [5] Jean Bacon, Ken Moody, and Walt Yao. A model of oasis role-based access control and its support for active security. *ACM Trans. Inf. Syst. Secur.*, 5(4):492–540, 2002.
- [6] J. Baek and Y. Zheng. Identity-Based Threshold Decryption. *Proc. of PKC*, 4:262–276, 2004.
- [7] Walid Bagga and Refik Molva. Policy-based cryptography and applications. In Andrew S. Patrick and Moti Yung, editors, *Financial Cryptography*, volume 3570 of *Lecture Notes in Computer Science*, pages 72–87. Springer, 2005.
- [8] Walid Bagga and Refik Molva. Collusion-free policy-based encryption. In Sokratis K. Katsikas, Javier Lopez, Michael Backes, Stefanos Gritzalis, and Bart Preneel, editors, *ISC*, volume 4176 of *Lecture Notes in Computer Science*, pages 233–245. Springer, 2006.
- [9] Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
- [10] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-Policy Attribute-Based Encryption. In *IEEE Symposium on Security and Privacy*, 2007.
- [11] Rakesh Bobba, Omid Fatemieh, Fariba Khan, Arindam Khan, Carl A. Gunter, Himanshu Khurana, and Manoj Prabhakaran. Attribute-Based Messaging: Access Control and Confidentiality. Under Revision for *ACM Transactions on Information and System Security*.
- [12] Rakesh Bobba, Omid Fatemieh, Fariba M. Khan, Carl A. Gunter, and Himanshu Khurana. Using attribute-based access control to enable attribute-based messaging. In *Annual Computer Security Applications Conference*, 2006.

- [13] Rakesh Bobba and Himanshu Khurana. DLPKH: Distributed logical public-key hierarchy. In *ICISS*, Lecture Notes in Computer Science. Springer, 2007.
- [14] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security*, pages 417–426. ACM, 2008.
- [15] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. *Advances in Cryptology-Crypto 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, 2001.
- [16] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. *Advances in Cryptology-Crypto 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, 2001.
- [17] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In Cramer [26], pages 440–456.
- [18] Robert W. Bradshaw, Jason E. Holt, and Kent E. Seamons. Concealing complex policies with hidden credentials. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 146–157, New York, NY, USA, 2004. ACM.
- [19] Robert W. Bradshaw, Jason E. Holt, and Kent E. Seamons. Concealing complex policies with hidden credentials. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 146–157, New York, NY, USA, 2004. ACM.
- [20] Mike Burmester and Yvo Desmedt. A secure and efficient conference key distribution system (extended abstract). In *EUROCRYPT*, pages 275–286, 1994.
- [21] J.Y. Cai, Zhenyu Huang, J. Hauer, and K. Martin. Current status and experience of wams implementation in north america. *Transmission and Distribution Conference and Exhibition: Asia and Pacific, 2005 IEEE/PES*, pages 1–7, 2005.
- [22] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2004.
- [23] Melissa Chase. Multi-authority Attribute Based Encryption. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 515–534. Springer, 2007.

- [24] Ling Cheung and Calvin Newport. Provably secure ciphertext policy ABE. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 456–465, New York, NY, USA, 2007. ACM.
- [25] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *IMA Int. Conf.*, volume 2260 of *LNCS*, pages 360–363. Springer, 2001.
- [26] Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.
- [27] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, February 2004.
- [28] J. E. Dagle. North american synchrophasor initiative. In *Hawaii International Conference on System Sciences*, 2008.
- [29] J.E. Dagle. Postmortem analysis of power grid blackouts - the role of measurement systems. *Power and Energy Magazine, IEEE*, 4(5):30–35, Sept.-Oct. 2006.
- [30] Matt Donnelly, Mike Ingram, and James Ritchie Carroll. Eastern interconnection phasor project. In *Hawaii International International Conference on Systems Science (HICSS-39 2006)*, January 2006.
- [31] K. Fidler, S. Krishnamurthi, L.A. Meyerovich, and M.C. Tschantz. Verification and change-impact analysis of access-control policies. *Proceedings of the 27th international conference on Software engineering*, pages 196–205, 2005.
- [32] Warwick Ford and Michael J. Wiener. A key distribution method for object-based protection. In *CCS '94: Proceedings of the 2nd ACM Conference on Computer and communications security*, pages 193–197, New York, NY, USA, 1994. ACM.
- [33] Keith B. Frikken, Mikhail J. Atallah, and Jiangtao Li. Attribute-based access control with hidden policies and hidden credentials. *IEEE Trans. Computers*, 55(10):1259–1270, 2006.
- [34] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554. Springer, 1999.
- [35] R. Gennaro. Robust and Efficient Sharing of RSA Functions. *Journal of Cryptology*, 13(2):273–300, 2000.

- [36] Vipul Goyal, Abhishek Jain 0002, Omkant Pandey, and Amit Sahai. Bounded ciphertext policy attribute based encryption. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfssdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 579–591. Springer, 2008.
- [37] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.
- [38] Louis Granboulan. RSA hybrid encryption schemes. Technical report, December 2001.
- [39] A. Herzberg, M. Jakobsson, S. Jarecki, H. Krawczyk, and M. Yung. Proactive public key and signature systems. *Proceedings of the 4th ACM conference on Computer and communications security*, pages 100–110, 1997.
- [40] Daisuke Inoue and Masahiro Kuroda. FDLKH: Fully decentralized key management scheme on logical key hierarchy. In *ACNS*, pages 339–354, 2004.
- [41] John P. Jones, Daniel F. Berger, and Chinya V. Ravishankar. Layering public key distribution over secure dns using authenticated delegation. In *ACSAC '05: Proceedings of the 21st Annual Computer Security Applications Conference*, pages 409–418, Washington, DC, USA, 2005. IEEE Computer Society.
- [42] Apu Kapadia, Patrick P. Tsang, and Sean W. Smith. Attribute-based publishing with hidden credentials and hidden policies. In *Proceedings of The 14th Annual Network and Distributed System Security Symposium (NDSS)*, pages 179–192, March 2007.
- [43] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.
- [44] Jonathan Katz and Moti Yung. Scalable protocols for authenticated group key exchange. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 110–125. Springer, 2003.
- [45] Himanshu Khurana, Jin Heo, and Meenal Pant. From proxy encryption primitives to a deployable secure-mailing-list solution. In Peng Ning, Sihon Qing, and Ninghui Li, editors, *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*, volume 4307 of *LNCS*, pages 260–281, Berlin, Heidelberg, 2006. Springer.
- [46] Himanshu Khurana, Adam Slagell, and Rafael Bonilla. Sels: a secure e-mail list service. In *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 306–313, New York, NY, USA, 2005. ACM.

- [47] Yongdae Kim, Adrian Perrig, and Gene Tsudik. Tree-based group key agreement. *ACM Trans. Inf. Syst. Secur.*, 7(1):60–96, 2004.
- [48] Jiangtao Li and Ninghui Li. Policy-hiding access control in open environment. In *PODC '05: Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, pages 29–38, New York, NY, USA, 2005. ACM.
- [49] Xiaohui Liang, Zhenfu Cao, Huang Lin, and Dongsheng Xing. Provably secure and efficient bounded ciphertext policy attribute based encryption. In *ASIACCS '09: Proceedings of the 2009 ACM symposium on Information, computer and communications security*, New York, NY, USA, March 2009. ACM.
- [50] Huang Lin, Zhenfu Cao, Xiaohui Liang, and Jun Shao. Secure threshold multi authority attribute based encryption without a central authority. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *INDOCRYPT*, volume 5365 of *Lecture Notes in Computer Science*, pages 426–436. Springer, 2008.
- [51] J. Linn. RFC 1421: Privacy enhancement for Internet electronic mail: Part I: Message encryption and authentication procedures, February 1993.
- [52] Marco Casassa Mont, Pete Bramhall, and Keith Harrison. A flexible role-based secure messaging service: Exploiting IBE technology for privacy in health care. In *DEXA '03: Proceedings of the 14th International Workshop on Database and Expert Systems Applications*, page 432, Prague, Czech Republic, 2003. IEEE.
- [53] Sascha Müller, Stefan Katzenbeisser, and Claudia Eckert. Distributed attribute-based encryption. In Pil Joong Lee and Jung Hee Cheon, editors, *ICISC*, volume 5461 of *Lecture Notes in Computer Science*, pages 20–36. Springer, 2008.
- [54] Paul Myrda, Erich Gunther, Mel Gehrs, and Jerry Melcher. Eipp data management task team architecture. In *Hawaii International International Conference on Systems Science (HICSS-40 2007)*, page 118, January 2007.
- [55] Takashi Nishide, Kazuki Yoneyama, and Kazuo Ohta. Attribute-based encryption with partially hidden encryptor-specified access structures. In Steven M. Bellovin, Rosario Gennaro, Angelos D. Keromytis, and Moti Yung, editors, *ACNS*, volume 5037 of *Lecture Notes in Computer Science*, pages 111–129, 2008.
- [56] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM Conference on Computer and Communications Security*, pages 195–203. ACM, 2007.

- [57] Matthew Pirretti, Patrick Traynor, Patrick McDaniel, and Brent Waters. Secure attribute-based systems. In *ACM Conference on Computer and Communications Security*, pages 99–112, 2006.
- [58] R. Poovendran and J. S. Baras. An information theoretic analysis of rooted-tree based secure multicast key distribution schemes. In Michael Wiener, editor, *Advances in cryptology — CRYPTO '99: 19th annual international cryptology conference, Santa Barbara, California, USA, August 15–19, 1999 proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 624–638, pub-SV:adr, 1999. Springer-Verlag.
- [59] B. Ramsdell. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification. RFC 3851, July 2004.
- [60] Editor Ramsdell. Secure/Multipurpose internet mail extensions (S/MIME) version 3.1 certificate handling. RFC 3850, Internet Engineering Task Force, July 2004.
- [61] Editor Ramsdell. Secure/Multipurpose internet mail extensions (S/MIME) version 3.1 message specification. RFC 3851, Internet Engineering Task Force, July 2004.
- [62] Ohad Rodeh, Kenneth P. Birman, and Danny Dolev. Using AVL trees for fault-tolerant group key management. *Int. J. Inf. Sec.*, 1(2):84–99, 2002.
- [63] Amit Sahai and Brent Waters. Fuzzy Identity-Based Encryption. In Cramer [26], pages 457–473.
- [64] Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.
- [65] Alan T. Sherman and David A. McGrew. Key establishment in large dynamic groups using one-way function trees. *IEEE Trans. Software Eng.*, 29(5):444–458, 2003.
- [66] Victor Shoup. Lower Bounds for Discrete Logarithms and Related Problems. In *EUROCRYPT*, Lecture Notes in Computer Science, pages 256–266, 1997.
- [67] Victor Shoup. A proposal for an iso standard for public key encryption. Cryptology ePrint Archive, Report 2001/112, 2001. <http://eprint.iacr.org/>.
- [68] D. K. Smetters and Glenn Durfee. Domain-based authentication of identity-based cryptosystems for secure email and IPsec. In *12th Usenix Security Symposium*, Washington, D.C., August 2003.
- [69] Mudhakar Srivatsa and Ling Liu. Key derivation algorithms for monotone access structures in cryptographic file systems. In *European Symposium on Research in Computer Security, Hamburg, Germany*, pages 347–361, September 2006.

- [70] Mudhakar Srivatsa and Ling Liu. Secure event dissemination in publish-subscribe networks. In *ICDCS '07: Proceedings of the 27th International Conference on Distributed Computing Systems*, page 22, Washington, DC, USA, 2007. IEEE Computer Society.
- [71] Michael Steiner, Gene Tsudik, and Michael Waidner. Key agreement in dynamic peer groups. *IEEE Trans. Parallel Distrib. Syst.*, 11(8):769–780, 2000.
- [72] Patrick Traynor, Kevin Butler, William Enck, and Patrick McDaniel. Realizing Massive-Scale Conditional Access Systems Through Attribute-Based Cryptosystems. In *Proceedings of The 15th Annual Network and Distributed System Security Symposium (NDSS)*, February 2008.
- [73] U.S.-Canada Power System Outage Task Force. Final report on the august 14, 2003 blackout in the united states and canada: Causes and recommendations, April 2004.
- [74] Chung Kei Wong, Mohamed G. Gouda, and Simon S. Lam. Secure group communications using key graphs. *IEEE/ACM Trans. Netw.*, 8(1):16–30, 2000.
- [75] Peng Yang, Takashi Kitagawa, Goichiro Hanaoka, Rui Zhang, Kanta Matsuura, and Hideki Imai. Applying Fujisaki-Okamoto to Identity-Based Encryption. In Marc P. C. Fossorier, Hideki Imai, Shu Lin, and Alain Poli, editors, *AAECC*, volume 3857 of *Lecture Notes in Computer Science*, pages 183–192. Springer, 2006.
- [76] Philip R. Zimmermann. *The Official PGP User's Guide*. MIT Press, pub-MIT:adr, 1995.
- [77] Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *EUROSAM*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.