PBES: A Policy Based Encryption System with Application to Data Sharing in the Power Grid

Rakesh Bobba, Himanshu Khurana, Musab AlTurki and Farhana Ashraf
University of Illinois at Urbana-Champaign
{rbobba, hkhurana, alturki, fashraf2}@illinois.edu

ABSTRACT

In distributed systems users need the ability to share sensitive content with multiple other recipients based on their ability to satisfy arbitrary policies. One such system is electricity grids where fine-grained sensor data sharing holds the potential for increased reliability and efficiency. However, effective data sharing requires technical solutions that support flexible access policies, for example, sharing more data when the grid is unstable. In such systems, both the messages and policies are sensitive and, therefore, they need to kept be secret. Furthermore, to allow for such a system to be secure and usable in the presence of untrusted object stores and relays it must be resilient in the presence of active adversaries and provide efficient key management. While several of these properties have been studied in the past we address a new problem in the area of policy based encryption in that we develop a solution with *all* of these capabilities. We develop a Policy and Key Encapsulation Mechanism − Data Encapsulation Mechanism (PKEM-DEM) encryption scheme that is a generic construction secure against adaptive chosen ciphertext attacks and develop a Policy Based Encryption System (PBES) using this scheme that provides these capabilities. We provide an implementation of PBES and measure its performance.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection; E.3 [**Data**]: Data Encryption

General Terms

Design, Security, Performance

Keywords

Policy Based Encryption, Multi-Party Data Sharing, Power Grid, Phasor Measurement Units (PMUs)

1. INTRODUCTION

In distributed systems users need to share sensitive objects with others based on the recipients' ability to satisfy a policy. In this work we develop a Policy Based Encryption System (PBES) that is driven by a real-world, large-scale, policy-based data sharing problem. The problem we look at 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 [40] 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 Independent Systems Operators (ISOs) 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 ISO 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 encryption 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) [11] systems and policy-based cryptographic schemes proposed in [2, 9] 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

ASIACCS '09 March 10-12, 2009, Sydney, NSW, Australia.

the encryption. Several works in the area of "hidden policies and credentials" [12,23,31] provide message and policy secrecy but focus on two-party interactions. Attribute Based Encryption (ABE) system such as Ciphertext-Policy ABE (CP-ABE) [10,14], policy-based cryptographic scheme [8] and cryptographic file system FS-Guard [37] 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. [8] is also vulnerable to collusion attacks. Recent work by [34] extends CP-ABE to support policy secrecy but significantly limits it's policy flexibility and does not support context-based policies. PEA-POD [30] 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.

Our Contribution In this work 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 and then use the solution to design an effective power grid data sharing application. 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 [15]. 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 policies[1]. 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 discuss how PBES can address the requirements of the power grid data sharing application and study design issues for developing applications in general; e.g., key distribution and placement of trust in KDCs. We also implement a prototype toolkit to demonstrate its feasibility and reasonable performance.

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. We show that with its unique properties PBES can naturally address the data sharing needs of the power grid. To that end, we identify actors, applications and processes for data sharing in the grid. While we focus primarily on 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 (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.

[1]A similar notion is independently defined and used by [34].

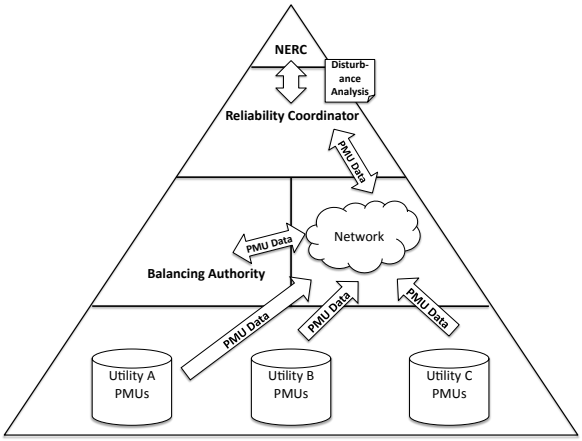

Figure 1: Proposed NASPI PMU Architecture

The rest of this paper is organized as follows. Section 2 provides background on power grids, Section 3 describes the requirements for the data sharing solution, Section 4 presents our approach, Section 5 presents our policy based encryption system, Section 6 describes integration with power grid and Section 7 discusses application design issues, Section 8 describes prototyping efforts along with performance results, Section 9 discusses related work and Section 10 concludes the paper and discusses future directions.

2. BACKGROUND

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 [16, 40]. Furthermore, deregulation of the power industry has moved it away from vertically integrated centralized operations to coordinated decentralized operations. Reliability Coordinators (RCs) such as Independent System Operators (ISOs) or Regional Transmission Operators (RTOs) 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 utilities[2] 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

[2]In 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

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 granularilty 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 increased 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 1 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 [17].

While utilities are currently mandated to share operational data with their local BA and RC (ISO or RTO) 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, brownouts and blackouts [16, 40]. Sharing PMU data will also help in planning, post disturbance/event analysis [16] and for research and development purposes. Currently two pilot deployments each with about 75 PMUs exist in Eastern [19] and Western [13] 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 [17]. 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.

3. 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 [16, 40].

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.

By design these requirements are application-independent in nature to allow the development of a widely applicable technical solution for data sharing.

4. APPROACH

4.1 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 [34] 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 ISOs/RTOs 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 for our applications. 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 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 [6] 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 policies 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 [5] expose certain attributes in clear-text for routing purposes, or use encryption-matching functions [38]. Solutions such as [5] carry over limitations of RBAC systems identified above. If we attempt to use a secure pub/sub solution like [38] for our application we again face limitations. First, ensuring policy secrecy for a flexible

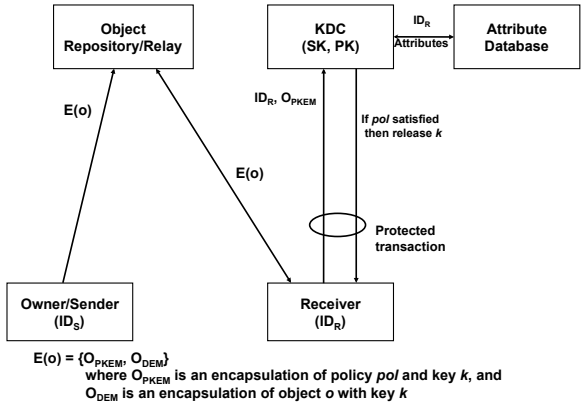

Figure 2: Policy-based Message Encryption and Decryption

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 need ephemeral keys for the various transient events.

4.2 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 2. The approach satisfies the requirements of Section 3 as follows.

The system is illustrated in Figure 2 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 meta-data 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 the receiver side the receiver need not send the encrypted ob-

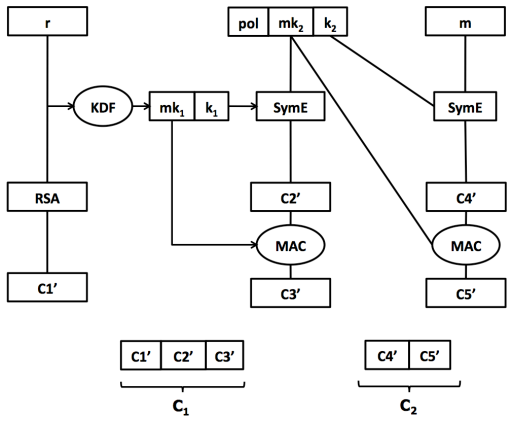

Figure 3: Encryption in PKEM-DEM scheme instantiated using RSA-KEM and DEM1

ject (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.

5. POLICY BASED ENCRYPTION SYSTEM

We first introduce some common notation used throughout the paper. We then define formal notions of security for a policy-based encryption scheme for multiple recipients with policy secrecy followed by our scheme and its security analysis.

5.1 Security Notions

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 $|S|$. $s \xleftarrow{\$} S$ denotes the operation of picking an element s of S uniformly at random. Adversaries are represented by probabilistic polynomial-time (PPT) algorithms $\mathcal{A}$. $v \xleftarrow{\$} \mathcal{A}(\alpha_1, \alpha_2, \ldots \alpha_k)$ denotes the action of running the PPT algorithm $\mathcal{A}$ with input $(\alpha_1, \alpha_2, \ldots \alpha_k)$ and letting v be the output. $\mathcal{A}^{O_1, O_2, \ldots, O_l}(\alpha_1, \alpha_2, \ldots \alpha_k)$ denotes a PPT adversary with input $(\alpha_1, \alpha_2, \ldots \alpha_k)$ and access to oracles $O_1, O_2, \ldots, O_l$. 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 adaptive chosen ciphertext attacks similar to the ones defined in [30].

Definition 5.1. 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.

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

where $\mathbf{G}_{\mathcal{E},\mathcal{A}}^{\mathcal{E}-msg-ind-cca2}(k)$ 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 5.2. 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.

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

where $\mathbf{G}_{\mathcal{E},\mathcal{A}}^{\mathcal{E}-pol-ind-cca2}(k)$ 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 5.3. 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.

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

where $\mathbf{G}_{\mathcal{E},\mathcal{A}}^{\mathcal{E}-pw-ind-cca2}(k)$ 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.

5.2 Encryption Scheme and System

Our encryption scheme is based on KEM-DEM hybrid encryption paradigm [15] 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.

PKEM-DEM.KeyGen(1^k) :
$(sk, pk) \xleftarrow{\$} PKEM.KeyGen(1^k)$
Return (sk, pk)

PKEM-DEM.Encrypt(m, pol, pk) :
$(K_2, C_1) \xleftarrow{\$} PKEM.Encrypt(pol, pk)$
$C_2 \leftarrow DEM.Encrypt(m, K_2)$
$C \leftarrow C_1 \| C_2$
Return C

PKEM-DEM.Decrypt-I(sk, f, C_1, u) :
$m' \leftarrow PKEM.Decrypt(sk, C_1)$
if $m' = \perp$ Return $\perp$
else parse m' as (pol, K_2)
if $f(u, pol) = 1$ Return K_2
else Return $\perp$

PKEM-DEM.Decrypt-II(K_2, C_2) :
if $K_2 = \perp$ Return $\perp$
$m \leftarrow DEM.Decrypt(K_2, C_2)$
Return m

Where PKEM is constructed as follows:

PKEM.KeyGen(1^k) :
$(sk, pk) \xleftarrow{\$} KEM.KeyGen(1^k)$
Return (sk, pk)

PKEM.Encrypt(pol, pk) :
$(K_1, C_1) \xleftarrow{\$} KEM.Encrypt(1^k, pk)$
$(K_2, C_2) \xleftarrow{\$} SPKEM.Encrypt(pol, K_1)$
$C \leftarrow C_1 \| C_2$
Return (K_2, C)

PKEM.Decrypt(sk, C) :
parse C as $C_1 \| C_2$
$K_1 \leftarrow KEM.Decrypt(sk, C_1)$
if $K_1 \neq \perp$
$m' \leftarrow SPKEM.Decrypt(K_1, C_2)$
if $m' = \perp$ return $\perp$
else Return $m' = (pol, K_2)$

And where SPKEM is constructed as follows:

SPKEM.Encrypt(pol, K) :
$K' \xleftarrow{\$} DEM.KeyGen(1^k)$
$m' \leftarrow pol \| K'$
$C \leftarrow DEM.Encrypt(m', K)$
Return (K', C)

SPKEM.Decrypt(K, C) :
$m' \leftarrow DEM.Decrypt(K, C)$
if $m' = \perp$ or parsing m'
as $pol \| K'$ fails return $\perp$
else Return (pol, K')

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 independent[3]. Figure 3 shows encryption in PKEM-DEM scheme instantiated using RSA-KEM and DEM1 defined in [15]

We use our PKEM-DEM encryption scheme to develop the PBES policy based encryption system whose architecture is illustrated in Figure 2 and described in Section 4.2. 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

[3]KEM-DEM schemes built using secure KEM and secure DEM that are related may not be secure as shown in [26]

(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 enviromental attributes (*i.e.*, context). The KDC uses these attributes of the user and the environment and 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.

5.3 Security Analysis

Since pairwise indistinguishability (in Def. 5.3) implies message indistinguishability (in Def. 5.1) and policy indistinguishability (in Def. 5.2) with restriction 2b, we prove that PKEM-DEM is pairwise indistinguishable in Theorem 5.1 and that it is policy indistinguishable with restriction 2a in Theorem 5.2 to show that PKEM-DEM system is secure against adaptive chosen ciphertext attacks.

Theorem 5.1. *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 5.3.*
In particular we have

$$\mathbf{Adv}^{pkem-dem-pw-ind-cca2}_{PKEM\text{-}DEM}(k) \leq 5 \cdot \mathbf{Adv}^{kem-ind-cca2}_{KEM}(k) + 3 \cdot \mathbf{Adv}^{dem-ind-otcca}_{DEM}(k) \quad (1)$$

Theorem 5.2. *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 5.2 with restriction 2a.*
In particular we have

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

Due to space constraints, definition of PKEM, associated security notions along with their proofs sketches are given in Appendix A. Proof for Theorem 5.1 and proof sketch for Theorem 5.2 are given in Appendix B. Full proofs will be given in an extended version of the paper.

6. PBES APPLICATION INTEGRATION

Table 1: 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

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 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 1.

As an example, consider a Utility A under the jurisdiction of an ISO 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 ISO 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 ISO 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) ISO B, *i.e.*, ISO 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 ISO for decryption. ISO upon verifying that the associated policy is satisfied returns the data decryption key. Note that ISO 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 polices 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 ISOs 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 ISO with all the utilities under its jurisdiction are leveraged to enable data sharing between utilities without the need to establish pairwise trust. Currently ISOs 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 ISO 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 ISO along with the federated identity and attribute databases that the ISO has access to constitute the Attribute Database shown in Figure 2. 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 ISOs suffices to reach all users registered in those ISO 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 ISO is able to enforce the access policy it is unlikely to have the resources to manage the data itself. This is because ISOs may oversee many BAAs, *e.g.*, Midwest ISO (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 [33]. 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 2 shows which power grid entities play the roles of the components in the PBES architecture presented in Figure 2.

Table 2: 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	ISO
Receiver	Utilities, BAs
Attribute Database	Environmental Attributes based on Power Grid State observed at the ISO along with Federated Identity/Attribute Databases at utilities

7. 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. We then present the design choices we made for the power grid data sharing application.

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 [36]. 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. In the power grid data sharing application we adopt a simple approach that each ISO/RTO will manage a KDC and provide mediated policy enforcement for users within the ISO oversight.

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 [29, 36] 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 [4], will improve the security. In the power grid data sharing application we recommend that NERC make the public keys of the ISOs available in a trusted manner; e.g., using regular trusted communication channels.

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 systems[4]. 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 [21]. For the power grid data sharing application PBES uses XACML based tools.

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 distribu-

[4] http://www.oasis-open.org/committees/download.php/27298/xacmlRefs-V1-84-1.htm

tion tasks. For example, in IBE [11] 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. However, in the power grid data sharing application the ISOs are already entrusted with access to most of this data for regulatory purposes. (ISOs keep this data only for a short time to ensure reliability and safety but do not typically store the data for long-term use.) 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 the power grid and many distributed applications being developed and deployed today have a largely online nature in that users usually access objects over the network. For example, utilities regularly deliver data to ISOs over wide-area networks. 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 8 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 [30] 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 [7, 24, 28].

8. 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 implementation[5]) 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.

Performance We instantiate our PKEM-DEM scheme using an RSA-based CCA secure KEM, RSA-KEM [35], and an OTCCA secure DEM, DEM1 [15,35] (essentially symmetric encryption with a message authentication code) as shown in Figure 3. 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

[5] `http://sunxacml.sourceforge.net/`

Table 3: Characteristics of schemes providing policy based encryption capabilities

	PKI	IBE [11]	CP-ABE [10, 34]	FSGuard [37]	PEAPOD [30]	IB-mRSA [18]	PBES
Policy Type And Flexibility	Recipient Identity	Flexible context based policy tied to an identity	Flexible attribute-based policy [10] or AND gate with multi-valued attributes [34]	Flexible attribute-based policy. However, limited to policies that can be satisfied by the encrypting entity.	Flexible attribute-based policy	Recipient Identity	Flexible identity, role, attribute and context based policies
Policy Secrecy	Not supported	Not supported	Not Supported in [10]. Supported in [34].	Not supported	Policy secret from eavesdroppers, recipients and CA	Not supported	Policy secret from eavesdroppers and recipients
Collusion	Not applicable	Not applicable	Prevented via per-attribute, per-user private keys	Prevented via per-user decryption key distribution	Vulnerable	Not applicable because of mediated decryption	Not applicable because of mediated decryption
End-to-end Encryption	Achieved	Key escrow with PKG	Key escrow with PKG	Key escrow with PKG	Achieved	Key escrow with CA	Key escrow with KDC
Encryption Key Distribution	Need to distribute recipient public keys to senders	Need to distribute PKG parameters to senders	Need to distribute PKG parameters to senders	Need to distribute per policy encryption key to senders	Need to distribute attribute public keys to senders	Need to distribute CA parameters to senders	Need to distribute KDC public key to senders
Decryption Key Distribution	Offline; Recipient generates the private key	Need to distribute per-policy private keys to recipients	Need to distribute per-attribute, per-user private keys to recipients	Need to distribute per-attribute, per-user private keys (on secure channel) and per-policy decryption keys (on authenticated channels) to recipients	Need to distribute per-attribute, per-user private keys to recipients	Offline; CA generates the private key	Mediated decryption via KDC
Revocation	OCSP/CRLs.	Supported via per-policy short-lived keys	Supported via per-attribute short-lived keys	Supported via per-attribute short-lived keys	Immediate revocation via CA	Immediate revocation via Security Mediator (SEM)	Immediate revocation via KDC.
Server availability	CA can be largely offline	PKG needs to be online to generate private keys	PKG needs to be online to generate private keys	Key/Group Server needs to be online to generate private keys	CA needs to be online for ciphertext transformation	SEM needs to be online for mediated decryption	KDC needs to be online for mediated decryption

of users the attribute private keys for individual users will expire with a distribution that pretty much requires the PKD 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 ISOs. These KDCs will serve hundreds of utilities across the grid with each ISO 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 in Section 6 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.

9. RELATED WORK

Our work touches upon topics in areas of policy/attribute based encryption, hidden credentials and policies, cryptographic file systems and efficient and effective key management. In Table 3 we compare the characteristics of several schemes in these areas. These characteristics are related to the properties discussed in Sections 6 and 7. The schemes that we compare are 1) Certificate Based Public Key Infrastructure (PKI), 2) IBE [11], 3) CP-ABE [10, 34], 4) FSGaurd [37], 5) PEAPOD [30], 6) Identity Based mediated RSA (IB-mRSA) [18] and 7) PBES. In this comparison PKI serves as a control scheme that essentially provides message secrecy for two-party message exchange with no policy secrecy. All other 6 schemes provide new functionality over PKI.

From the table we see that CP-ABE, FSGuard, PEAPOD and PBES provide policy based encryption for multiple recipients, however, only PBES provides support for context based policies (e.g., time of day, location). In terms of policy secrecy only CP-ABE [34], PEAPOD and PBES provide this property where CP-ABE [34] and PEAPOD additionally provide policy secrecy against all servers. In terms of vulnerability to user collusion, which essentially renders a a scheme insecure, the only vulnerable scheme is PEAPOD. PBES provides collusion resistance via the use of mediated decryption that eliminates any partial results at the users. In terms of end-to-end encryption all schemes achieve that goal to some extent, however, IBE, CP-ABE, FSGuard, IB-mRSA and KDC all have implicit key escrow. In terms of encryption key distribution, IBE, CP-ABE, IB-mRSA and PBES are all efficient in that they require the senders to only fetch one public key/parameters per "domain" while for decryption key distribution most schemes need servers to distribute the keys or provide mediated decryption. In terms of revocation PEAPOD, IB-mRSA and PBES provide immediate revocation capability. Furthermore, IBE, IB-mRSA, CP-ABE [14] and PBES are secure against adaptive chosen ciphertext attacks but only PBES has a generic construction. PBES provides a generic construction because it builds on a KEM-DEM scheme [15].

When looking at all these characteristics, PBES is the only scheme that provides *all* of the following: flexible policy based encryption including context-based policies for multiple intended recipients, message and policy secrecy, support for untrusted repositories and relays, efficient key management and immediate revocation, and

security against adaptive chosen ciphertext attacks with a generic construction. Policy secrecy is not provided against the KDC in PBES but the only schemes that provide such secrecy are PEA-POD which has collusion problems and CP-ABE [34] which does not support flexible or context-based policies. PBES also shares the properties of key escrow and an online nature with some of these other schemes that may limit its applicability for certain applications.

Outside of these schemes the work that probably comes closest to ours is the enterprise object encryption architecture proposed by [22] 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. Additional schemes have been proposed that also share some of the characteristics discussed in Table 3. [12, 23, 31] provide message and policy secrecy but they focus on two-party interactions. [2,8,9] only provide message secrecy and focus on two-party interactions. Key-Policy ABE (KP-ABE) [25] is a dual of CP-ABE that associates attributes with data and policies with users. Attribute certificates have been proposed for use in conjunction with PKI identity certificates [20], however, their use is intended for attribute based access control rather than encryption. [32] expounds of problems with PKI deployment such as costs of managing end-entity certificates and as an alternative suggests the use of Trusted Third Parties for encryption. [42] promote the idea of a single public key per domain (in their case the public key is for a SPKI/SDSI server) by leveraging symmetric-key based systems such as Kerberos.

Recent interest in securing critical infrastructures such as power grid has led to research activity on various aspects of protecting such infrastructures. Secure and redundant cyber architectures for critical infrastructure are considered in [41]. Automatic security assesment of critical cyber infrastructure is considered in [3]. YASIR [39] protects legacy SCADA communication. The work closely related to ours, GridStat [27], is a pub/sub system that provides a low-latency QoS managed communication infrastructure to enable data sharing for real-time applications. GridStat recognises the need for communication security in such a system and allows for publishing encrypted and integrity protected data but it does not deal with key distribution or management. PBES can potentially be used to distribute cryptographic keys used in GridStat.

10. CONCLUSION

In this work we develop a Policy Based Encryption System (PBES) that proposes a solution to a new problem in the area of policy based encryption and apply it to the data sharing problem in electricity grids. PBES supports policy based encryption for multiple recipients, policy secrecy and efficient key management all with an encryption scheme that is secure against adaptive chosen ciphertext attacks. We develop PBES with the PKEM-DEM encryption secure against CCA2 adversaries and then develop the system with untrusted repositories/relays and trusted KDCs. We study how PBES can address the needs of power grids. We prototype the system and demonstrate its performance to be reasonable. We will release a usable library for PBES in the near future to allow easy integration with applications.

We envision several directions of future work. First, the efficiency of PBES can be improved by using other encryption schemes such as the Tag-KEM/DEM framework [1]. 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. Second, new policy based encryption schemes can be developed that support flexible policies and provide policy secrecy against all servers (PBES reveals policies to KDC). Furthermore, providing schemes that allow routing of messages based on the secret policies is an additional challenge. Third, PBES shares the key escrow property with several other schemes that may limit its applicability for certain applications. New schemes that minimize or eliminate this limitation while still providing desirable properties would provide significant advancement.

11. ACKNOWLEDGMENTS

This work was supported by the by National Science Foundation under Grant Nos. CNS 05-24695 and CNS 07-16626, and by the Office of Naval Research under Grant Nos. N00014-06-1-1108 and N00014-07-1-1173. We would like to thank Manoj Prabhakaran for valuable discussions and suggestions.

12. REFERENCES

[1] M. Abe, R. Gennaro, and K. Kurosawa. Tag-KEM/DEM: A new framework for hybrid encryption. *J. Cryptol.*, 21(1):97–130, 2008.

[2] S. S. Al-Riyami, J. Malone-Lee, and N. P. Smart. Escrow-free encryption supporting cryptographic workflow. *Int. J. Inf. Sec.*, 5(4):217–229, 2006.

[3] Z. Anwar, R. Shankesi, and R. H. Campbell. Automatic Security Assessment of Critical Cyber-Infrastructures. In *Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. Springer, July 2008.

[4] R. Arends, R. Austein, M. Larson, and D. Massey. Resource Records for the DNS Security Extensions. Technical report, RFC 4034, March 2005.

[5] J. Bacon, D. M. Eyers, K. Moody, and L. I. W. Pesonen. Securing Publish/Subscribe for Multi-domain Systems. In G. Alonso, editor, *Middleware*, volume 3790 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2005.

[6] J. Bacon, K. Moody, and W. 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.

[7] J. Baek and Y. Zheng. Identity-Based Threshold Decryption. *Proc. of PKC*, 4:262–276, 2004.

[8] W. Bagga and R. Molva. Policy-Based Cryptography and Applications. In A. S. Patrick and M. Yung, editors, *Financial Cryptography*, volume 3570 of *Lecture Notes in Computer Science*, pages 72–87. Springer, 2005.

[9] W. Bagga and R. Molva. Collusion-Free Policy-Based Encryption. In S. K. Katsikas, J. Lopez, M. Backes, S. Gritzalis, and B. Preneel, editors, *ISC*, volume 4176 of *Lecture Notes in Computer Science*, pages 233–245. Springer, 2006.

[10] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-Policy Attribute-Based Encryption. In *IEEE Symposium on Security and Privacy*, 2007.

[11] 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.

[12] R. W. Bradshaw, J. E. Holt, and K. 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.

[13] J. Cai, Z. 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.

[14] L. Cheung and C. 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.

[15] R. Cramer and V. Shoup. Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack. *SIAM Journal on Computing*, 33(1):167–226, Feb. 2004.

[16] J. Dagle. Postmortem analysis of power grid blackouts - The role of measurement systems. *Power and Energy Magazine, IEEE*, 4(5):30–35, Sept.-Oct. 2006.

[17] J. E. Dagle. North American SynchroPhasor Initiative. In *Hawaii International Conference on System Sciences*, 2008.

[18] X. Ding and G. Tsudik. Simple Identity-Based Cryptography with Mediated RSA. *Topics in Cryptology, CT-RSA 2003: The Cryptographers' Track at the Rsa Conference 2003, San Francisco, CA, USA April 13-17, 2003, Proceedings*, 2003.

[19] M. Donnelly, M. Ingram, and J. R. Carroll. Eastern Interconnection Phasor Project. In *Hawaii International International Conference on Systems Science (HICSS-39 2006)*, January 2006.

[20] S. Farrell and R. Housley. An Internet Attribute Certificate Profile for Authorization (RFC 3281). *Internet Engineering Task Force, Network Working Group, April*, 2002.

[21] K. Fisler, S. Krishnamurthi, L. Meyerovich, and M. Tschantz. Verification and change-impact analysis of access-control policies. *Proceedings of the 27th international conference on Software engineering*, pages 196–205, 2005.

[22] W. Ford and M. 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.

[23] K. B. Frikken, M. J. Atallah, and J. Li. Attribute-Based Access Control with Hidden Policies and Hidden Credentials. *IEEE Trans. Computers*, 55(10):1259–1270, 2006.

[24] R. Gennaro. Robust and Efficient Sharing of RSA Functions. *Journal of Cryptology*, 13(2):273–300, 2000.

[25] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98, 2006.

[26] L. Granboulan. RSA hybrid encryption schemes. Technical report, Dec. 2001.

[27] C. H. Hauser, D. E. Bakken, I. Dionysiou, K. H. Gjermundrød, V. S. Irava, J. Helkey, and A. Bose. Security, Trust, and QoS in Next-Generation Control and Communication for Large Power Systems. *International Journal of System of Critical Infrastructures*, 4(1/2), 2008.

[28] 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.

[29] J. P. Jones, D. F. Berger, and C. V. Ravishankar. Layering Public Key Distribution Over Secure DNS using Authenticated Delegation. In *ACSAC*, pages 409–418. IEEE Computer Society, 2005.

[30] A. Kapadia, P. P. Tsang, and S. 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.

[31] J. Li and N. 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.

[32] J. Linn and M. Branchaud. An examination of asserted PKI issues and proposed alternatives. *Proceedings of the 3rd Annual PKI R & D WorkshopGaithers-burg: NIST*, 2004.

[33] P. Myrda, E. Gunther, M. Gehrs, and J. Melcher. EIPP Data Management Task Team Architecture. In *Hawaii International International Conference on Systems Science (HICSS-40 2007)*, page 118, January 2007.

[34] T. Nishide, K. Yoneyama, and K. Ohta. Attribute-Based Encryption with Partially Hidden Encryptor-Specified Access Structures. In S. M. Bellovin, R. Gennaro, A. D. Keromytis, and M. Yung, editors, *ACNS*, volume 5037 of *Lecture Notes in Computer Science*, pages 111–129, June 2008.

[35] V. Shoup. A Proposal for an ISO Standard for Public Key Encryption. Cryptology ePrint Archive, Report 2001/112, 2001. `http://eprint.iacr.org/`.

[36] D. K. Smetters and G. Durfee. Domain-Based Authentication of Identity-Based Cryptosystems for Secure Email and IPsec. In *12th Usenix Security Symposium*, Washington, D.C., August 2003.

[37] M. Srivatsa and L. 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.

[38] M. Srivatsa and L. 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.

[39] P. P. Tsang and S. W. Smith. YASIR: A Low-Latency, High-Integrity Security Retrofit for Legacy SCADA Systems. In S. Jajodia, P. Samarati, and S. Cimato, editors, *SEC*, volume 278 of *IFIP*, pages 445–459. Springer, 2008.

[40] 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.

[41] P. Veríssimo, N. F. Neves, and M. Correia. The CRUTIAL reference critical information infrastructure architecture: a blueprint. *International Journal of System of Systems Engineering*, 1(1/2), 2008.

[42] H. Wang, S. Jha, T. W. Reps, S. Schwoon, and S. G. Stubblebine. Reducing the Dependence of SPKI/SDSI on PKI. In *European Symposium on Research in Computer Security, Hamburg, Germany*, pages 156–173, September 2006.

APPENDIX

A. POLICY AND KEY ENCAPSULATION MECHANISM (PKEM)

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 except that unlike KEM, PKEM.Encrypt also accepts an additional bit string from the message space (interpreted as policy) as input and PKEM.Decrypt outputs both the key and policy as seen in Section 5.2.

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 A.1. 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:*

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

where

$$\begin{aligned}
&\textbf{Game } \mathbf{G}^{pkem-key-ind-cca2}_{PKEM,\mathcal{A}}(k): \\
&(sk, pk) \xleftarrow{\$} PKEM.KeyGen(1^k);\ (St, pol) \xleftarrow{\$} \mathcal{A}_1^{DEC(.)}(pk) \\
&b \xleftarrow{\$} \{0,1\};\ K_0^* \xleftarrow{\$} \{0,1\}^{PKEM.KeyLen(k)} \\
&(K_1^*, C^*) \xleftarrow{\$} PKEM.Encrypt(pk, pol);\ K^* \xleftarrow{\$} K_b^* \\
&b' \xleftarrow{\$} \mathcal{A}_2^{DEC(.)}(pk, C^*, K^*, St);\ \textit{Return } b'
\end{aligned}$$

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

Definition A.2. 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:*

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

where

$$\begin{aligned}
&\textbf{Game } \mathbf{G}^{pkem-pol-ind-cca2}_{PKEM,\mathcal{A}}(k): \\
&(sk, pk) \xleftarrow{\$} PKEM.KeyGen(1^k);\ (St, pol_0, pol_1) \xleftarrow{\$} \mathcal{A}_1^{DEC(.)}(pk) \\
&b \xleftarrow{\$} \{0,1\};\ (K^*, C^*) \xleftarrow{\$} PKEM.Encrypt(pk, pol_b) \\
&b' \xleftarrow{\$} \mathcal{A}_2^{DEC(.)}(pk, (K^*, C^*), St);\ \textit{Return } b'
\end{aligned}$$

and oracle $DEC(.)$ is defined as $PKEM.Decrypt(sk,.)$ 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 can construct a SPKEM using a DEM and then build a PKEM using SPKEM and KEM as shown in Section 5.2. The SPKEM scheme of Section 5.2 is secure against OTCCA attacks on key and policy indistinguishability as stated by the following theorems.

Theorem A.1. *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

$$\mathbf{Adv}^{spkem-key-ind-otcca}_{SPKEM,\ \mathcal{A}}(k) = \mathbf{Adv}^{dem-ind-otcca}_{DEM,\mathcal{B}}(k) \tag{3}$$

Theorem A.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 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

$$\mathbf{Adv}^{spkem-pol-ind-otcca}_{SPKEM,\ \mathcal{A}}(k) = \mathbf{Adv}^{dem-ind-otcca}_{DEM,\mathcal{B}}(k) \tag{4}$$

PROOF SKETCHES OF THEOREMS A.1 AND A.2. Since policy and key in the SPKEM scheme of Section 5.2 are encapsulated using a DEM, *i.e.*, policy concatenated with a random key constitute the message for DEM, it is straight forward to show that attacks on key or policy indistinguishability of SPKEM reduce to attacks on indistinguishability of the underlying DEM.

The PKEM scheme of Section 5.2 is secure against adaptive chosen ciphertext attacks on both key and policy indistinguishability. In particular, the following theorems hold.

Theorem A.3. *If KEM and SPKEM schemes are secure against adaptive chosen ciphertext attack and one-time adaptive chosen ciphertext attack on key indistinguishability respectively 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,

$$\begin{aligned}
&\mathbf{Adv}^{pkem-key-ind-cca2}_{PKEM,\ \mathcal{A}}(k) \leq \\
&\quad 2 \cdot \mathbf{Adv}^{kem-ind-cca2}_{KEM,\mathcal{B}_1}(k) + \mathbf{Adv}^{SPKEM-key-ind-otcca}_{SPKEM,\mathcal{B}_2}(k)
\end{aligned} \tag{5}$$

PROOF SKETCH OF THEOREM A.3. Let $\mathbf{G}_0$ be the original attack game defined by Definition A.1. 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

$$\mathbf{Adv}^{pkem-key-ind-cca2}_{\mathcal{PKEM},\mathcal{A}}(k) = |Pr[\mathcal{E}_0] - 1/2| \tag{6}$$

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

$$Pr[\mathcal{E}_0] = Pr[\mathcal{E}_1] \tag{7}$$

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, one can show the following:

Lemma A.3. *There exists a probabilistic algorithm $\mathcal{B}_1$ whose running time is essentially the same as that of $\mathcal{A}$, such that*

$$|Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]| = 2 \cdot \mathbf{Adv}^{kem-ind-cca2}_{KEM,\mathcal{B}_1}(k) \tag{8}$$

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

$$|Pr[\mathcal{E}_2] - 1/2| = \mathbf{Adv}^{spkem-key-ind-cca2}_{SPKEM,\ \mathcal{B}_2}(k) \tag{9}$$

The theorem now follows from equations 6, 7, 8 and 9. □

Theorem A.4. *If the underlying KEM and SPKEM schemes are secure against adaptive chosen ciphertext attack and one-time adaptive chosen ciphertext attack 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,

$$\begin{aligned}
&\mathbf{Adv}^{pkem-pol-ind-cca2}_{PKEM,\ \mathcal{A}}(k) = \\
&\quad \mathbf{Adv}^{kem-ind-cca2}_{KEM,\mathcal{B}}(k) + \mathbf{Adv}^{spkem-pol-ind-otcca}_{SPKEM,\mathcal{B}}(k)
\end{aligned} \tag{10}$$

PROOF SKETCH OF THEOREM A.4. This proof is very similar to that of Theorem A.3 above except that in Game $\mathbf{G}_2$ the adversary is launching an OTCCA attack against policy indistinguishability of SPKEM instead of key indistingusiahbility. □

B. PKEM-DEM SECURITY

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.

PROOF OF THEOREM 5.1. Let $\mathbf{G}_0$ be the original attack game, *i.e.*, $\mathbf{G}^{pkem-dem-ind-cca2}_{PKEM\text{-}DEM,\mathcal{A}}(k)$, described in Definition 5.3. 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

$$\mathbf{Adv}^{pkem-dem-ind-cca2-cu}_{PKEM\text{-}DEM,\mathcal{A}}(k) = |Pr[\mathcal{E}_0] - 1/2| \tag{11}$$

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 $1 \le i \le 2$.

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

$$Pr[\mathcal{E}_0] = Pr[\mathcal{E}_1] \tag{12}$$

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 B.1. *There exists a probabilistic algorithm $\mathcal{B}_1$ whose running time is essentially the same as that of $\mathcal{A}$, such that*

$$|Pr[\mathcal{E}_1] - Pr[\mathcal{E}_2]| = 2 \cdot \mathbf{Adv}^{pkem-key-ind-cca2}_{PKEM,\mathcal{B}_1}(k) \tag{13}$$

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 B.2. *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*

$$|Pr[\mathcal{E}_2] - 1/2| \le \mathbf{Adv}^{dem-ind-cca2}_{DEM,\mathcal{B}_2}(k) + \mathbf{Adv}^{pkem-pol-ind-cca2}_{PKEM,\mathcal{B}_3}(k) \tag{14}$$

The theorem now follows from equations 3, 4, 5, 10, 11, 12, 13 and 14. □

PROOF OF LEMMA B.1. $\mathcal{B}_1$ 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 $\mathbf{Adv}^{pkem-key-ind-cca2}_{PKEM,\mathcal{B}_1}(k)$ is,

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

□

PROOF OF LEMMA B.2. Let probability of success of $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in game $\mathbf{G}_2$ be $1/2 + \varepsilon$. Then, $|Pr[\mathcal{E}_2] - 1/2| = \varepsilon$. 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 B.1 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, $\mathbf{Adv}^{dem-ind-cca2}_{DEM,\mathcal{B}_2}(k) = |Pr[\delta' = \delta] - 1/2|$ is

$$\begin{aligned} &= \frac{1}{2} \cdot \left(Pr[\delta' = 0 | \delta = 0] + Pr[\delta' = 1 | \delta = 1] \right) - \frac{1}{2} \\ &= \frac{1}{2} \cdot \Big(\big(Pr[b_0 = 0 \wedge b_1 = 0 | \delta = 0] + Pr[\theta = 0 \wedge b_0 = 0 \wedge b_1 = 1 | \delta = 0] \\ &\quad + Pr[\theta = 1 \wedge b_0 = 1 \wedge b_1 = 0 | \delta = 0] \big) + \big(Pr[b_0 = 1 \wedge b_1 = 1 | \delta = 1] \\ &\quad + Pr[\theta = 0 \wedge b_0 = 1 \wedge b_1 = 0 | \delta = 1] + Pr[\theta = \wedge b_0 = 0 \wedge b_1 = 1 | \delta = 1] \big) \Big) - \frac{1}{2} \\ &= \frac{\varepsilon}{2} + \frac{(\beta - \alpha)}{4} \end{aligned} \tag{15}$$

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. We then have

$$\mathbf{Adv}^{pkem-pol-ind-cca2}_{PKEM,\mathcal{B}_3}(k) = \frac{\varepsilon}{2} - \frac{(\beta - \alpha)}{4} \tag{16}$$

The lemma follows from equations 15 and 16. □

PROOF SKETCH OF THEOREM 5.2. 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.

□