Private Mobile Pay-TV From Priced Oblivious Transfer

Wouter Biesmans, Josep Balasch, Alfredo Rial,
Bart Preneel, *Member, IEEE*, Ingrid Verbauwhede, *Member, IEEE*

Abstract—In pay-TV, a service provider offers TV programs and channels to users. To ensure that only authorized users gain access, conditional access systems (CAS) have been proposed. In existing CAS, users disclose to the service provider the TV programs and channels they purchase. We propose a pay-per-view and a pay-per-channel CAS that protect users' privacy. Our pay-per-view CAS employs priced oblivious transfer (POT) to allow a user to purchase TV programs without disclosing which programs were bought to the service provider. In our pay-per-channel CAS, POT is employed together with broadcast attribute-based encryption (BABE) to achieve low storage overhead, collusion resistance, efficient revocation and broadcast efficiency. We propose a new POT scheme and show its feasibility by implementing and testing our CAS on a representative mobile platform.

Index Terms—ADP-PPRO, ADP-APPS, APC-MMED

1 Introduction

In pay-TV, a service provider offers TV programs and channels to users. To ensure that only authorized users gain access, conditional access systems (CAS) have been proposed [1], [2]. CAS can implement pay-per-channel, where users subscribe to channels or groups of channels for a period of time, or pay-per-view, where users pay for individual TV programs. In flexible pay-per-channel [3], users can subscribe to any combination of channels and change their subscription during the subscription period.

A CAS should provide some functionalities and security properties [4]. *Fine-grained access control* is required to ensure that only authorized users can get access. For example, it should be possible to apply access control policies that deny access to underage users. As in any commercial transaction, *non-repudiation* is necessary to solve disputes. *Backward secrecy* ensures that revoked users cannot get access. Additionally, a CAS should be efficient, which involves efficient transmission, low storage overhead, and efficient key redistribution when users are revoked or change their subscription.

Traditional CAS can roughly be divided into two classes: group key based symmetric schemes and public key based schemes. In group key based schemes [1], [3], [5], [6], [7], users in the same group share a group key. To transmit a video to users in different groups, the service provider determines which group keys should be used and encrypts the video employing those keys. Group key based schemes offer efficient broadcast transmission. However, they suffer collusion attacks, lack of non-repudiation, and inefficient key distribution [4]. Public key based schemes [8], [9] offer non-repudiation, lighter storage overhead and efficient key redistribution. However, transmission is inefficient and fine-grained access control is difficult to implement.

Recently, two CAS which enjoy both efficient broadcast transmission and the security guarantees of public key based schemes have been proposed for pay-TV [10] and mobile pay-TV [4]. They are based on attribute-based encryption (ABE) [11], [12]. The scheme in [4] is compatible with the digital video broadcast standard DVB-H[1] and DVB-SH[2]. However, as explained in Section 2, these CAS have shortcomings.

To the best of our knowledge, user privacy has not been considered in existing CAS. Currently, in pay-per-channel, users disclose to the service provider the channels to which they subscribe, and in pay-per-view, they disclose the TV programs they purchase. This may reveal users' sensitive information, such as political views, religious beliefs and sexual orientation.

Our Contribution. We propose a pay-per-view and a pay-per-channel CAS that protect users' privacy. Both CAS employ priced oblivious transfer (POT) [13] in order to allow the user to purchase channels and programs without disclosing which channels and programs were bought to the service provider. POT employs a prepayment mechanism, where the user makes an initial deposit and, at each purchase, subtracts the price paid from the deposit without the service provider learning the price paid or the new value of the deposit.

In our pay-per-channel CAS, POT is employed together with broadcast attribute-based encryption (BABE) [14], which allows our CAS to enjoy low storage overhead, collusion resistance, efficient revocation and broadcast efficiency. With respect to the CAS in [4], our CAS employs an efficient direct revocation method in which non-revoked users do

Josep Balasch, Bart Preneel and Ingrid Verbauwhede are with the imec-COSIC group of Departement Elektrotechniek (ESAT), KU Leuven, Belgium e-mail: firstname.lastname@esat.kuleuven.be.

Alfredo Rial is with the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg, e-mail: alfredo.rial@uni.lu.

Part of this work was carried out while Wouter Biesmans was a master student at the COSIC research group.

[1] http://www.dvb.org/standards/dvb-h

[2] http://www.dvb.org/standards/dvb-sh

not perform any operation. With respect to the CAS in [10], our CAS avoids the use of an ad-hoc revocation method based on an expiration time attribute, which, as explained in Section 2, requires a large ciphertext size.

To show the feasibility of our approach, we propose a new POT scheme. Our POT scheme is similar to the scheme in [15], but replaces the signature scheme employed there by a structure-preserving signature scheme [16], which allows to shorten the ciphertext size. We implement both our pay-per-view and a pay-per-channel CAS and show performance measurements in an ARM Cortex-A8 processor core. Our results demonstrate that our approach can be deployed in practical settings even when using platforms from early-generation mobile devices.

Outline of the paper. In Section 2, we describe and analyze related work. In Section 3, we describe our model for CAS and its properties. In Section 4, we describe the concepts of priced oblivious transfer and broadcast attribute-based encryption. We describe our pay-per-channel and our pay-per-view CAS in Section 5 and Section 6 respectively. We discuss their properties in Section 7. In Section 8, we propose a new priced oblivious transfer scheme. In Section 9, we analyze the efficiency of our CAS. We describe an implementation and performance measurements of both BABE and POT schemes. We conclude in Section 10.

2 Related Work

The group key based CAS described in the DVB standard[3] consists of four layers: registration, rights management, key stream and traffic encryption. In the registration layer, the user's device is given several decryption keys for authentication and broadcast decryption purposes. In the rights management layer, the service provider determines the inferred encryption key (IEK) that is used to encrypt the service encryption key (SEK). The choice of IEK depends on which devices should receive the SEK, i.e., on which users have subscribed for a particular channel. Zero message broadcast encryption [17] is employed to encrypt the right object (RO), which contains the SEK, on input the IEK. In the key stream layer, the SEK is used to encrypt the traffic encryption key TEK. In the traffic encryption layer, the TEK is used to encrypt the video content. The TEK is updated frequently, e.g., every second.

The DVB CAS, as well as other group key based CAS [1], [3], [5], [6], [7], suffers from lack of non-repudiation and inefficient key distribution. Furthermore, for security reasons, the size of the groups should be small, thus reducing the advantage of broadcast encryption efficiency. Public key based CAS [8], [9] employ inefficient one to one transmission and hinder fine-grained access control.

The CAS proposed in [4] is based on the key-policy attribute-based encryption scheme (KP-ABE) in [12]. The key stream and traffic encryption layers remain unchanged from the DVB standard. During registration, each user obtains a KP-ABE key that binds her to a subscribers category and to other attributes, such as her age. The rights management layer works as follows. To encrypt the RO, the service provider employs KP-ABE. Each KP-ABE ciphertext is associated to a category and to an age rating, such that only users with the right category and age can decrypt it and get the RO. In order to revoke a user, a mechanism in which all the users but the revoked user update their KP-ABE keys is proposed, which is inefficient.

The CAS proposed in [10] is based on ciphertext policy attribute set based encryption (ASBE). The key stream and traffic encryption layers remain unchanged from the DVB standard. During registration, each user obtains an ASBE key for attributes of the form *(channel, start, end)*, where *channel* identifies a channel, and *start* and *end* denote the start and end dates of the subscription. In ASBE, the attributes have a hierarchy, so that the attributes *start* and *end* of a channel cannot be used for other channels. In the rights management layer, to encrypt the RO, the user employs a policy *(channel, $\leq$ starttime, $\geq$ endtime)*, i.e., the *start* and *end* attributes of the user should be lower and greater than the *starttime* and *endtime* attributes in the policy. This mechanism is inefficient and also needs key updates.

The problems of the CAS in [4] and [10] can be solved by using attribute-based encryption with revocation. An ABE with revocation scheme was first proposed in [18]. As described in [19], there are two main revocation methods: direct and indirect. Indirect revocation does not require the senders to know the revocation list, but it requires the keys of non-revoked users to be updated. Direct revocation requires the sender to know the revocation list but no key updates are needed. Therefore, direct revocation is more suitable for a CAS, where the service provider is the only sender and already knows the revocation list. We employ the ABE with direct revocation proposed in [14] in our CAS.

We summarize now recent progress in ABE applicable to the construction of CAS. In [20], a more efficient ABE with direct revocation scheme is proposed. The ABE with direct revocation schemes in [14] and [20] are selectively secure but, in [21], a fully-secure one is presented. In [22], a forward-secure ABE scheme is shown. This scheme gives protection against key exposure and is adequate for settings where the provider does not know the revocation list. Otherwise, ABE with direct revocation is better suited because it avoids key updates. In [23] (resp. [24]), an ABE scheme with white-box (resp. black-box) traceability is presented, which allows one to identify users that leak their secret keys (resp. their decryption box).

Other recent works on CAS include an authentication and subscription protocol between the provider and users [25] and a CAS for portable devices based on selective encryption that provides interoperability between providers [26].

To the best of our knowledge, there is no CAS that protects user privacy. POT was proposed in [13]. Existing POT schemes offer different security levels: half-simulation security [13], full-simulation security [27], and universal composability [15]. We propose a new universally composable POT scheme that shortens the ciphertext size of the scheme in [15].

Oblivious transfer with access control (OTAC) [28], [29] allows the service provider to enforce access control policies in OT protocols. Some OTAC schemes are based on combining OT with ABE [30], [31]. Although POT can be seen as a form of oblivious transfer with access control, OTAC

[3] http://www.dvb.org/standards/dvb-h

schemes do not allow for an efficient implementation of POT. The reason is that, in OTAC schemes, a user possesses some attributes that are certified only once by a credential issuer, while in POT, the user account is an attribute that changes each time the user makes a purchase in a way that depends on the item purchased. Therefore, existing OTAC schemes cannot be applied without modification to construct a privacy-preserving CAS. We note that some OTAC schemes also hide the access control policy from the user [32], but this is not necessary in our case since the price of a channel or program should be public.

3 Definition of Our Conditional Access System

3.1 Pay-Per-Channel CAS

Our pay-per-channel conditional access system (PPC-CAS) is run by a provider $\mathcal{P}$ and users $\mathcal{U}$ and consists of a registration layer, a rights management layer, a key stream layer and a traffic encryption layer, which we describe in Figure 1. The last two layers work as in the DVB standard, which we summarize in Section 2.

In the registration layer, at setup, the provider creates public parameters and a master secret key. After that, a user and the provider run a protocol by means of which the user obtains right objects that contain parts of a secret key associated with the user's identity and attributes. The user attributes are TV channels for which the user subscribes. The user obtains her secret key without revealing to the provider the TV channels she subscribes for, while still paying the right price for the subscription. Each right object is associated with an access control policy that describes a rating (i.e., an age restriction) for a TV channel, so that a user cannot subscribe for a channel whose rating is not adequate for the user's age.

In the rights management layer, the provider receives as input a SEK, an attribute associated with a TV channel and a revocation list. The provider encrypts the SEK under the identities of non-revoked subscribers and under the attribute.

A PPC-CAS should fulfill the following security properties.

Privacy. No coalition of parties, which can include the provider, must be able to learn the TV channels for which a user subscribes.

Revocation. $\mathcal{P}$ must have the ability to end the subscription of a user when it expires or when it must be terminated because of misbehavior. A revoked user must not be able to access channels for which her subscription is revoked.

Collusion Resistance. A group of colluding users (which could include revoked users) must not be able to access channels that none of them is able to access individually.

Non repudiation. Non repudiation is an important property when a dispute between $\mathcal{P}$ and $\mathcal{U}$ occurs. It requires that $\mathcal{P}$ should be able to prove that $\mathcal{U}$ made a subscription if $\mathcal{U}$ did make the subscription, while $\mathcal{U}$ should be able to prove that she has made the corresponding payment if $\mathcal{U}$ did make the payment. When there is a dispute between $\mathcal{P}$ and $\mathcal{U}$, they can ask a trusted adjudicator to resolve it.

These security properties must hold against an active adversary. The adversary controls the network and is able to eavesdrop, delay and modify messages exchanged between parties. In addition, the adversary is allowed to corrupt parties and control their behavior.

Throughout this paper, we use several abbreviations. They are summarized in Table 1.

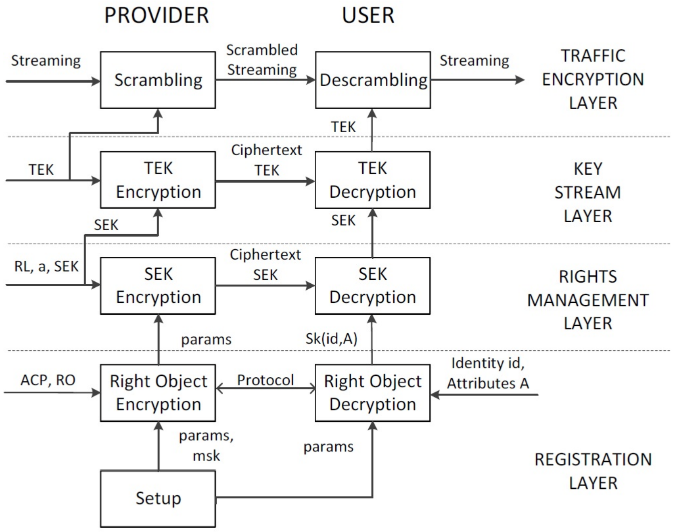

Fig. 1. Pay-Per-Channel Conditional Access System

TABLE 1
Table of Abbreviations

ABE	Attribute-Based Encryption
ASBE	Attribute Set Based Encryption
BABE	Broadcast Attribute-Based Encryption
CAS	Conditional Access System
CNF	Conjuntive Normal Form
DBV-H	Digital Video Broadcasting - Handheld
DBV-SH	DVB - Satellite services to Handhelds
DNF	Disjunctive Normal Form
IEK	Inferred Encryption Key
KP-ABE	Key-Policy Attribute-Based Encryption
OT	Oblivious Transfer
OTAC	Oblivious Transfer with Access Control
$\mathcal{P}$	Provider
POT	Priced Oblivious Transfer
PPC-CAS	Pay-Per-Channel CAS
PPV-CAS	Pay-Per-View CAS
RO	Right Object
SEK	Service Encryption Key
TEK	Traffic Encryption Key
$\mathcal{U}$	User

3.2 Pay-Per-View CAS

Our pay-per-view conditional access system (PPV-CAS) is run by a provider $\mathcal{P}$ and users $\mathcal{U}$ and consists of a rights management layer, a key stream layer and a traffic encryption layer, which we describe in Figure 2. The last two layers work as in the DVB standard, which we summarize in Section 2.

In the rights management layer, the provider gets as input, for each TV program, a right object RO and an access control policy that describes the age rating. The user inputs as attributes its age and a TV program of her choice. The provider and the user run a protocol by means of which

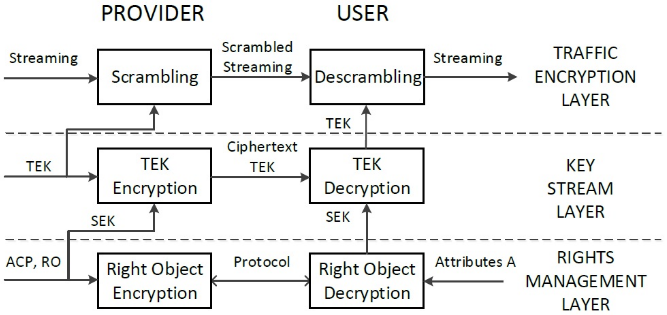

Fig. 2. Pay-Per-View Conditional Access System

the user decrypts the right object associated with the TV program of her choice without revealing the TV program to the provider. A right object contains the SEK needed to decrypt a TV program.

A PPV-CAS should fulfill the properties of privacy, collusion-resistance, and non-repudiation defined in Section 3.1. In our PPV-CAS, the SEK is only valid for the duration of a TV program and there are no long term keys, so revocation is less necessary.

4 Building Blocks

4.1 Broadcast Attribute-Based Encryption

Let $\mathbb{U}$ be the set of all user indexes and $\mathbb{S}$ be the set of all attributes. Let $\mathbb{Q}$ be the set of all policies that are allowed over $\mathbb{S}$. We denote by $\mathbb{A} \vDash \mathbb{P}$ the fact that a set of attributes $\mathbb{A}$ satisfies the policy $\mathbb{P}$.

We use broadcast ciphertext policy attribute based encryption (BABE) [14], where the secret key of a user is associated with a user index id and to a set of attributes $\mathbb{A}$, and a ciphertext is associated to a set of user indexes $\mathbb{I}$ and to a policy $\mathbb{P}$. A BABE scheme consists of the following algorithms:

- $\mathsf{Setup}(1^k)$. On input the security parameter 1^k, output the public parameters par and a master secret key msk.
- $\mathsf{KeyGen}(par, msk, id, \mathbb{A})$. On input the parameters par, the master secret key msk, the user index $id \in \mathbb{U}$ and the set of attributes $\mathbb{A} \in \mathbb{S}$, output a secret key $sk(id, \mathbb{A})$ for the user index id and the attributes $\mathbb{A}$.
- $\mathsf{Enc}(par, \mathbb{I}, \mathbb{P}, m)$. On input the parameters par, the user index set $\mathbb{I} \in \mathbb{U}$, the policy $\mathbb{P} \in \mathbb{Q}$, and the message m, output a ciphertext $ct(\mathbb{I}, \mathbb{P})$.
- $\mathsf{Dec}(par, sk(id, \mathbb{A}), ct(\mathbb{I}, \mathbb{P}))$. On input the parameters par, a secret key $sk(id, \mathbb{P})$ and a ciphertext $ct(\mathbb{I}, \mathbb{A})$, output a message m if $id \in \mathbb{I}$ and if $\mathbb{A} \vDash \mathbb{P}$.

In the rights management layer of a our pay-per-channel CAS, users are assigned a user identifier id, and the set of attributes of a user will be the set of TV channels to which the user subscribes. When a user subscribes to a new TV channel, it must be possible to add the associated TV channel attribute to an existing secret key. For this purpose, we split up the algorithm KeyGen into two algorithms KeyGen_0 and KeyGen_1.

- $\mathsf{KeyGen}_0(par, msk, id)$. On input the parameters par, the master secret key msk, and the user index $id \in \mathbb{U}$, output a secret key part $sk(id)$ for the user index id and auxiliary information aux.
- $\mathsf{KeyGen}_1(aux, s)$. On input the auxiliary information aux and the attribute s, output a secret key part $sk(s)$.

Therefore, a secret key for the user index $id \in \mathbb{U}$ and the set of attributes $\mathbb{A} \in \mathbb{S}$ is computed by first running $(sk(id), aux) \leftarrow \mathsf{KeyGen}_0(par, msk, id)$ and, for each $s \in \mathbb{A}$, $sk(s) \leftarrow \mathsf{KeyGen}_1(aux, s)$, and then setting $sk(id, \mathbb{A}) \leftarrow (sk(id), \langle sk(s) \rangle_{\forall s \in \mathbb{A}})$. The algorithm KeyGen of the broadcast ciphertext policy attribute based encryption in [14] can be split up into KeyGen_0 and KeyGen_1. We summarize the symbols used for BABE in Table 2 (left).

4.2 Priced Oblivious Transfer

Priced oblivious transfer (POT) is a two-party protocol that provides privacy in e-commerce of digital goods by hiding from the vendor which items are bought. More formally, a provider $\mathcal{P}$ sells a set of messages $(m_1, \ldots, m_N)$ with prices $(p_1, \ldots, p_N)$ to a user $\mathcal{U}$. At each purchase, $\mathcal{U}$ chooses $i \in \{1, \ldots, N\}$, gets m_i and pays p_i. The following security properties must hold.

Provider security. For any message m_i bought by $\mathcal{U}$, $\mathcal{U}$ must pay a price p_i. $\mathcal{U}$ must not learn any information about messages that $\mathcal{U}$ has not bought.

User privacy. For any message m_i, when $\mathcal{U}$ buys m_i, $\mathcal{P}$ learns nothing about m_i or p_i.

POT schemes usually employ a prepaid mechanism. A user pays an initial deposit to the provider. When a user buys a message, the price of the message is subtracted from the deposit. The provider learns neither the price of the message bought nor the new value of the user's account, but the provider has the guarantee that the new account value is correct and that a user cannot buy a message if the account value is lower than the message price.

A POT scheme consist of an initialization phase, a payment phase, a request phase and a sell phase, and it is parameterized by a message space $\mathcal{M}$, a number of messages N, a universe of prices U_{pr} and a universe of accounts U_{ac}. The initialization phase consists of the algorithms InitP and InitU.

- $\mathsf{InitP}(1^k, m_1, p_1, \ldots, m_N, p_N)$. On input the security parameter 1^k, the messages $(m_1, \ldots, m_N)$ and the prices $(p_1, \ldots, p_N)$, output state information st_p for $\mathcal{P}$ and a message M_{init} to be sent to $\mathcal{U}$.
- $\mathsf{InitU}(1^k, M_{init})$. On input the security parameter 1^k and the message M_{init}, output state information st_u for $\mathcal{U}$ and prices $(p_1, \ldots, p_N)$.

The payment phase consists of the algorithms PayP and PayU.

- $\mathsf{PayU}(st_u, p)$. On input state information st_u and payment p, output a payment message M_{pay} and updated state information st_u.
- $\mathsf{PayP}(st_p, M_{pay})$. On input state information st_p and message M_{pay}, output updated state information st_p and payment p.

The request phase consists of the algorithms ReqP and ReqU.

- $\mathsf{ReqU}(st_u, i)$. On input state information st_u and index i, output state information st_u and message M_{req} to be sent to $\mathcal{P}$.
- $\mathsf{ReqP}(st_p, M_{req})$. On input state information st_p and message M_{req}, output updated state information st_p.

The request phase consists of the algorithms SellP and SellU.

- $\mathsf{SellP}(st_p)$. On input state information st_p, output updated state information st_p and message M_{sell} to be sent to $\mathcal{U}$.
- $\mathsf{SellU}(st_u, M_{sell})$. On input state information st_u, output updated state information st_u and message m_i.

We summarize the symbols used for POT in Table 2 (right).

5 Pay-Per-Channel CAS

We describe a mobile pay-per-channel conditional access system (PPC-CAS). The key stream and traffic encryption layers remain as in the DVB standard. We thus describe a protocol for the registration and rights management layer of our PPC-CAS.

The service encryption key (SEK) of a TV channel is updated periodically, e.g., every week. The user $\mathcal{U}$ purchases a subscription to a TV channel by running a priced oblivious transfer protocol (POT) with the provider $\mathcal{P}$. The right object RO for a TV channel, which the user $\mathcal{U}$ receives as output of the POT protocol, contains a key that allows the user to obtain the SEK each time the provider $\mathcal{P}$ updates the SEK.

The provider $\mathcal{P}$ employs broadcast ciphertext-policy attribute based encryption (BABE) in order to release updates of the SEK. The reason why $\mathcal{P}$ employs BABE is threefold. First, it is more efficient than purchasing the SEK each time it is updated by employing the POT protocol. Second, BABE allows for efficient revocation. When a user is revoked, the provider simply removes the user identifier id from the set $\mathbb{I}$ used to compute the ciphertext that encrypts the updated SEK. Third, the user secret key of a BABE scheme can be stored in a tamper-resistant element of the mobile device in order to prevent the user from sharing her keys. This is possible thanks to the fact that the key $sk(id, \mathbb{A})$ can be split up into $(sk(id), \langle sk(s)\rangle_{\forall s\in\mathbb{A}})$. The key part $sk(id)$, which depends only on the user identifier id, can be stored permanently in a tamper-resistance element. The key parts $\langle sk(s)\rangle_{\forall s\in\mathbb{A}}$, which are purchased through the POT protocol, are useless without $sk(id)$. We note that the collusion resistance property of the BABE scheme ensures that users cannot combine their keys to produce a key for more attributes.

The protocol for the registration layer of our PPC-CAS consists of a setup phase, an initialization phase, a payment phase, and a purchase phase. The rights management layer consists of a SEK update phase and a revocation phase. We depict them in Figure 3.

Setup Phase. $\mathcal{P}$ runs $(par, msk) \leftarrow \mathsf{Setup}(1^k)$. $\mathcal{P}$ keeps msk and publishes par.

Initialization Phase. The user $\mathcal{U}$ receives as input a certified age age. The provider $\mathcal{P}$ receives as input a list of right objects RO. Each right object RO_i contains a channel attribute s_i, a price p_i and a rating rt_i. The rating indicates the minimum age age a user must have in order to be allowed to subscribe to a TV channel. The interaction between $\mathcal{U}$ and $\mathcal{P}$ is as follows:

- $\mathcal{P}$ assigns an identifier id to the user $\mathcal{U}$ and runs $(sk(id), aux) \leftarrow \mathsf{KeyGen}_0(par, msk, id)$. $\mathcal{P}$ stores the tuple $(\mathcal{U}, id, aux)$ and includes id in a set $\mathbb{I}$ of user identifiers. $\mathcal{P}$ stores the key part $sk(id)$ in a tamper-resistant element of the user mobile device.
- $\mathcal{U}$ sends age to $\mathcal{P}$ through an authenticated channel.
- $\mathcal{P}$ verifies the certified age age. We leave open the way the attribute age is verified.
- $\mathcal{P}$ creates a list $(RO_1, \ldots, RO_N)$ of right objects such that the attribute age of $\mathcal{U}$ fulfills the rating rt of the right object. $\mathcal{P}$ retrieves aux from the tuple $(\mathcal{U}, id, aux)$. For each RO_i, $\mathcal{P}$ runs $sk(s_i) \leftarrow \mathsf{KeyGen}_1(aux, s_i)$ and appends s_i to RO_i.
- $\mathcal{P}$ runs $(st_p, M_{init}) \leftarrow \mathsf{InitP}(1^k, RO_1, p_1, \ldots, RO_N, p_N)$, stores st_p and sends M_{init} to $\mathcal{U}$ through an authenticated channel.
- $\mathcal{U}$ receives M_{init}, runs $(st_u, p_1, \ldots, p_N) \leftarrow \mathsf{InitU}(1^k, M_{init})$ and stores st_u.

Payment Phase. The user $\mathcal{U}$ receives as input a payment p. The provider $\mathcal{P}$ receives no input. The interaction between $\mathcal{U}$ and $\mathcal{P}$ is as follows:

- $\mathcal{U}$ runs $(st_u, M_{pay}) \leftarrow \mathsf{PayU}(st_u, p)$, updates the stored st_u and sends M_{pay} to $\mathcal{P}$ through an authenticated channel.
- $\mathcal{P}$ receives M_{pay} and runs $(st_p, p) \leftarrow \mathsf{PayP}(st_p, M_{pay})$. $\mathcal{P}$ updates the stored st_p and verifies that a payment of value p has been received through any existing payment protocol.

Purchase Phase. The user $\mathcal{U}$ receives as input an index $i \in [1, N]$. The provider $\mathcal{P}$ receives no input. The interaction between $\mathcal{U}$ and $\mathcal{P}$ is as follows:

- $\mathcal{U}$ runs $(st_u, M_{req}) \leftarrow \mathsf{ReqU}(st_u, i)$, stores st_u and sends M_{req} to $\mathcal{P}$ through an authenticated channel.
- $\mathcal{P}$ receives M_{req}, runs $st_p \leftarrow \mathsf{ReqP}(st_p, M_{req})$ and updates the stored st_p. $\mathcal{P}$ runs $(st_p, M_{sell}) \leftarrow \mathsf{SellP}(st_p)$, updates the stored st_p and sends M_{sell} to $\mathcal{U}$ through an authenticated channel.
- $\mathcal{U}$ receives M_{sell}, runs $(st_u, RO_i) \leftarrow \mathsf{SellU}(st_u, M_{sell})$ and updates the stored st_u. $\mathcal{U}$ retrieves $sk(s_i)$ from RO_i, sets $\mathbb{A} \leftarrow \mathbb{A} \cup s_i$ and adds $sk(s_i)$ to her secret key $sk(id, \mathbb{A})$.

SEK Update Phase. The provider $\mathcal{P}$ receives as input a channel policy $\mathbb{P}$ and a service encryption key SEK. The channel policy allows any user to obtain the SEK if the user possesses a secret key for the channel attribute. $\mathcal{P}$ and $\mathcal{U}$ do the following:

- $\mathcal{P}$ retrieves the stored user identifier set $\mathbb{I}$ and runs $ct(\mathbb{I}, \mathbb{P}) \leftarrow \mathsf{Enc}(par, \mathbb{I}, \mathbb{P}, \mathrm{SEK})$. $\mathcal{P}$ broadcasts the ciphertext $ct(\mathbb{I}, \mathbb{P})$ to the users.
- A user $\mathcal{U}$ receives $ct(\mathbb{I}, \mathbb{P})$ and checks whether, for her secret key $sk(id, \mathbb{A})$, $id \in \mathbb{I}$ and $\mathbb{A} \vDash \mathbb{P}$. In that case, $\mathcal{U}$ runs $\mathrm{SEK} \leftarrow \mathsf{Dec}(par, sk(id, \mathbb{A}), ct(\mathbb{I}, \mathbb{P}))$ and outputs SEK.

Revocation Phase. The provider $\mathcal{P}$ receives as input the identity $\mathcal{U}$ of a user. To revoke this user, $\mathcal{P}$ retrieves id from

TABLE 2
Table of Symbols for BABE (left) and POT (right)

$\vDash$	Satisfy	$\mathbb{P}$	Policy
1^k	Security parameter	par	Public parameters
$\mathbb{A}$	Attribute set	$\mathbb{Q}$	Universe of policies
aux	Auxiliary information	s	Attribute
$ct(\mathbb{I}, \mathbb{P})$	Ciphertext	$\mathbb{S}$	Universe of attributes
$\mathbb{I}$	User indices set	$sk(id)$	Secret key - identity part
id	User index	$sk(s)$	Secret key - attribute part
m	Message	$sk(id, \mathbb{A})$	Secret key
msk	Master secret key	$\mathbb{U}$	Universe of user indices

1^k	Security parameter	p	Value of payment
i	Message index	$\mathcal{P}$	Provider
m	Message	p	Price
$\mathcal{M}$	Message space	st_p	State information for $\mathcal{P}$
M_{init}	Initialization message	st_u	State information for $\mathcal{U}$
M_{pay}	Payment message	$\mathcal{U}$	User
M_{req}	Request message	U_{ac}	Universe of accounts
M_{sell}	Sell message	U_{pr}	Universe of prices
N	Number of messages		

the stored tuple $(\mathcal{U}, id, aux)$ and sets $\mathbb{I} \leftarrow \mathbb{I} \setminus id$.

The provider $\mathcal{P}$ establishes subscription periods of, e.g., one year. At the beginning of a new period, $\mathcal{P}$ changes the identifiers of the channel attributes s_i and the corresponding policies $\mathbb{P}$ used to encrypt the SEK. Therefore, at the beginning of a new period, users must renew their subscriptions by running the POT protocol to obtain key parts $sk(s_i)$ for the new channel attributes s_i in order to be able to obtain the updated SEK's. We note that the key part $sk(id)$ does not need to be updated. We also note that a user can subscribe to a channel at any time. If the subscription is not carried out at the beginning of the subscription period, the prices of the channels are adjusted for the remaining of the period.

6 Pay-Per-View CAS

In our pay-per-view CAS (PPV-CAS), the key stream and traffic encryption layers remain as in the DVB standard, which we briefly describe in Section 2. A user $\mathcal{U}$ needs only one SEK to decrypt the TEK's throughout the broadcast. The SEK is also updated every week, but we consider that TV programs last shorter.

Through the rights management layer, a user obtains right objects, which contain the SEK, for TV programs of her choice. The rights management layer uses priced oblivious transfer to let users purchase right objects without revealing the TV programs bought to the provider.

The protocol consists of an initialization phase, a payment phase and a purchase phase. Basically, it consists on executing a POT protocol where the provider sells right objects that contain the SEK of TV programs.

Initialization Phase. The user $\mathcal{U}$ receives as input a certified age age. The provider $\mathcal{P}$ receives as input a list of right objects RO. Each right object RO_i contains a program attribute s_i, the SEK, a price p_i and a rating rt_i. The rating indicates the minimum age age a user must have in order to be allowed to buy a TV program. The interaction between $\mathcal{U}$ and $\mathcal{P}$ is as follows:

- $\mathcal{U}$ sends age to $\mathcal{P}$ through an authenticated channel.
- $\mathcal{P}$ verifies the certified age age. We leave open the way the attribute age is verified.
- $\mathcal{P}$ creates a list $(RO_1, \ldots, RO_N)$ of right objects such that the attribute age of $\mathcal{U}$ fulfills the rating rt of the right object. $\mathcal{P}$ runs $(st_p, M_{init}) \leftarrow \mathsf{InitP}(1^k, RO_1, p_1, \ldots, RO_N, p_N)$, stores st_p and sends M_{init} to $\mathcal{U}$ through an authenticated channel.
- $\mathcal{U}$ receives M_{init}, runs $(st_u, p_1, \ldots, p_N) \leftarrow \mathsf{InitU}(1^k, M_{init})$ and stores st_u.

Payment Phase. The payment phase works as the payment phase of our PPC-CAS in Section 5.

Purchase Phase. The user $\mathcal{U}$ receives as input an index $i \in [1, N]$. The provider $\mathcal{P}$ receives no input. The interaction between $\mathcal{U}$ and $\mathcal{P}$ is as follows:

- $\mathcal{U}$ runs $(st_u, M_{req}) \leftarrow \mathsf{ReqU}(st_u, i)$, stores st_u and sends M_{req} to $\mathcal{P}$ through an authenticated channel.
- $\mathcal{P}$ receives M_{req}, runs $st_p \leftarrow \mathsf{ReqP}(st_p, M_{req})$ and updates the stored st_p. $\mathcal{P}$ runs $(st_p, M_{sell}) \leftarrow \mathsf{SellP}(st_p)$, updates the stored st_p and sends M_{sell} to $\mathcal{U}$ through an authenticated channel.
- $\mathcal{U}$ receives M_{sell}, runs $(st_u, RO_i) \leftarrow \mathsf{SellU}(st_u, M_{sell})$ and updates the stored st_u. $\mathcal{U}$ retrieves the SEK contained in RO_i.

7 Discussion of Our CAS

We discuss the properties of our pay-per-view and our pay-per-channel CAS.

Privacy. The user privacy property of the priced oblivious transfer scheme guarantees that the service provider does not learn any information on the TV channels for which a user subscribes.

Revocation. The BABE scheme employs a direct revocation method. $\mathcal{P}$ associates each user to a user index id. Each user receives a secret key that allows to decrypt ciphertexts that include id. $\mathcal{P}$, who knows the revocation list, encrypts the SEK of each channel on input the list of non-revoked users. A revoked user will not be able to obtain the next SEK for that channel, which is updated weekly. Therefore, our CAS provides backward secrecy.

Collusion Resistance. The BABE scheme also ensures that users cannot collude to gain access to channels that they cannot access on their own. The user secret keys in a BABE scheme cannot be combined to obtain a secret key that encompasses the attributes of each of the secret keys. Additionally, by storing the key part $sk(id)$ in a tamper-resistant element, a user is prevented from sharing her key.

Non repudiation. For POT schemes, the problem of non repudiation has been addressed in [33], which provides a method to turn any POT scheme into a POT scheme that offers fair exchange. This method can be applied to our POT scheme, so that, at the end of a purchase, $\mathcal{P}$ gets a signature by $\mathcal{U}$ that acknowledges that the purchase has been made, while $\mathcal{U}$ receives a signature by $\mathcal{P}$ that acknowledges that the payment was received. These signatures should be used in case of a dispute between $\mathcal{P}$ and $\mathcal{U}$.

Low Storage Overhead. The storage overhead of our pay-per-channel CAS requires the storage of a BABE secret key

Provider	**Setup**	**User**
$(par, msk) \leftarrow \mathsf{Setup}(1^k)$	$\xrightarrow{par}$	Store par
Provider	**Initialization**	**User**
Input: $(RO_1, \ldots, RO_{N'})$		Input: age
$(sk(id), aux) \leftarrow \mathsf{KeyGen}_0(par, msk, id)$	$\xrightarrow{sk(id)}$	Store $sk(id)$
Verify and store age	$\xleftarrow{age}$	
Pick $(RO_1, \ldots, RO_N)$ satisfied by age		
$\{sk(s_i) \leftarrow \mathsf{KeyGen}_1(aux, s_i)\}_{i=1}^{N}$		
$\{$Append $sk(s_i)$ to $RO_i\}_{i=1}^{N}$		
$(st_p, M_{init}) \leftarrow \mathsf{InitP}(1^k, RO_1, p_1, \ldots, RO_N, p_N)$	$\xrightarrow{M_{init}}$	$(st_u, p_1, \ldots, p_N) \leftarrow \mathsf{InitU}(1^k, M_{init})$
Provider	**Payment**	**User**
		Input: p
$(st_p, p) \leftarrow \mathsf{PayP}(st_p, M_{pay})$	$\xleftarrow{M_{pay}}$	$(st_u, M_{pay}) \leftarrow \mathsf{PayU}(st_u, p)$
Provider	**Purchase**	**User**
		Input: i
$st_p \leftarrow \mathsf{ReqP}(st_p, M_{req})$	$\xleftarrow{M_{req}}$	$(st_u, M_{req}) \leftarrow \mathsf{ReqU}(st_u, i)$
$(st_p, M_{sell}) \leftarrow \mathsf{SellP}(st_p)$	$\xrightarrow{M_{sell}}$	$(st_u, RO_i) \leftarrow \mathsf{SellU}(st_u, M_{sell})$
		$sk(s_i) \leftarrow RO_i$, $\mathbb{A} \leftarrow \mathbb{A} \cup s_i$, $sk(id, \mathbb{A}) \leftarrow sk(id, \mathbb{A}) \cup sk(s_i)$
Provider	**SEK Update**	**User**
Input: SEK, $\mathbb{P}$		
$ct(\mathbb{I}, \mathbb{P}) \leftarrow \mathsf{Enc}(par, \mathbb{I}, \mathbb{P}, \mathrm{SEK})$	$\xrightarrow{ct(\mathbb{I}, \mathbb{P})}$	$\mathrm{SEK} \leftarrow \mathsf{Dec}(par, sk(id, \mathbb{A}), ct(\mathbb{I}, \mathbb{P}))$
Provider	**Revocation**	**User**
Input: id		
$\mathbb{I} \leftarrow \mathbb{I} \setminus id$		

Fig. 3. Our Pay-Per-Channel CAS.

for a user identifier id and for the TV channels to which the user has subscribed, and of one SEK for each TV channel to which the user has subscribed. Our pay-per-view CAS requires to store one SEK for each purchased TV program.

Key Redistribution. Our scheme enjoys efficient key redistribution when a user is revoked. Thanks to the use of a direct revocation method, other users do not need to perform any operation. Users only need to renew their keys at the beginning of each subscription period, e.g., every year. We note that this update does not involve the key part stored in the tamper-resistant element of the mobile device.

Broadcast Efficiency. We note that, in our pay-per-view CAS, $\mathcal{U}$ and $\mathcal{P}$ need to run a POT protocol, which involves one to one communication. However, other CAS also require one to one communication so that $\mathcal{U}$ communicates to $\mathcal{P}$ the program purchased and gets the SEK. After getting the SEK, both our CAS and other CAS enjoy broadcast transmission efficiency. In our pay-per channel CAS, $\mathcal{U}$ and $\mathcal{P}$ run a POT protocol when $\mathcal{U}$ subscribes to a channel. Other CAS also involve one to one communication in the subscription phase, where $\mathcal{U}$ communicates to $\mathcal{P}$ the TV channels to which she wishes to subscribe. After that, our scheme enjoys broadcast efficiency because the ciphertexts that encrypt the RO are the same for all the users and can thus be broadcast.

Broadcast Anonymity. We note that the privacy provided by our pay-per-view and pay-per-channel CAS is lost if the video content is transmitted on demand rather than being broadcast to users. In that case, users can employ an anonymous communication network [34] to access the video content in order to protect their privacy.

Fine-Grained Access Control. The BABE scheme allows the provider $\mathcal{P}$ to apply fine-grained access control. In our pay-per-channel CAS, the provider enforces a simple policy that requires users to subscribe to a channel. However, the BABE scheme allows to apply any policy described by a CNF or DNF formula, and threshold predicates, which allows the provider to enforce more complex access control policies.

In Table 3, we compare the features of our PPC-CAS with those of the PPC-CAS in [4] and [10]. As can be seen, the main advantages of our CAS in comparison with those in [4] and [10] are that our CAS provides user privacy and efficient key redistribution. The former is due to the use of POT, which allows users to subscribe to channels without revealing the channels they purchase to the provider. The latter is due to the use of BABE, which allows for a revocation method in which non-revoked users do not need to update their keys.

We use the symbol $\perp$ for properties that none of the analyzed PPC-CAS attains, but that can be achieved by extending them with suitable tools. The non-repudiation claim of [4] is not completely sound. They should use a signature scheme or message authentication code to prove

TABLE 3
Feature comparison of our CAS with those in [4] and [10].

	Our CAS	[4]	[10]
Privacy	Yes	No	No
Revocation	Yes	Yes	Yes
Collusion Resistance	Yes	Yes	Yes
Non-repudiation	$\perp$	$\perp$	$\perp$
Efficient Key Redistribution	Yes	No	No
Broadcast Efficiency	Yes	Yes	Yes
Broadcast Anonymity	$\perp$	$\perp$	$\perp$
Fine-grained access control	Yes	Yes	Yes

that the ciphertexts are not tampered. In our CAS, broadcast anonymity can be achieved by using an anonymous communication network. Non-repudiation can be achieved by using digital signature schemes and fair exchange protocols.

8 Priced Oblivious Transfer

8.1 Building Blocks of Our POT Protocol

Bilinear Maps. Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be groups of prime order p. A map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ must satisfy bilinearity, i.e., $e(g^x, \tilde{g}^y) = e(g, \tilde{g})^{xy}$; non-degeneracy, i.e., for all generators $g \in \mathbb{G}_1$ and $\tilde{g} \in \mathbb{G}_2$, $e(g, \tilde{g})$ generates $\mathbb{G}_T$; and efficiency, i.e., there exists an efficient algorithm $\mathsf{BMSetup}(1^k)$ that outputs the pairing group setup $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \tilde{g})$ and an efficient algorithm to compute $e(a, b)$ for any $a \in \mathbb{G}_1$, $b \in \mathbb{G}_2$.

Non-Interactive ZKPK. Let R be a polynomial time computable binary relation. For tuples $(wit, ins) \in R$ we call wit the witness and ins the instance. Let L be the NP-language consisting of the instances ins for which there exist witnesses wit such that $(wit, ins) \in R$. A non-interactive zero-knowledge proof of knowledge (NIPK) system for R consists of the algorithms PKSetup, PKProve and PKVerify. On input a security parameter 1^k, $\mathsf{PKSetup}(1^k)$ outputs the parameters par_{PK}. $\mathsf{PKProve}(par_{PK}, wit, ins)$ checks whether $(wit, ins) \in R$ and in that case outputs a proof π. $\mathsf{PKVerify}(par_{PK}, ins, \pi)$ outputs 1 if π is a valid proof that $ins \in L$ or 0 if that is not the case. A NIPK system must fulfill the zero-knowledge and simulation-sound extractability properties [35].

Signatures. A signature scheme consists of the algorithms KeyGen, Sign, and VfSig. $\mathsf{KeyGen}(1^k)$ outputs a secret key sk and a public key pk, which include a description of the message space $\mathcal{M}$. $\mathsf{Sign}(sk, m)$ outputs a signature s on a message $m \in \mathcal{M}$. $\mathsf{VfSig}(pk, s, m)$ outputs 1 if s is a valid signature on m and 0 otherwise. This definition can be extended to blocks of messages $\bar{m} = (m_1, \ldots, m_n)$. In this case, $\mathsf{KeyGen}(1^k, n)$ receives the maximum number of messages as input. A signature scheme must be existentially unforgeable [36].

Commitments. A commitment scheme consists of the algorithms CSetup, Com and VfCom. $\mathsf{CSetup}(1^k)$ generates the parameters par_c, which include a description of the message space $\mathcal{M}$ and of the randomness space $\mathcal{R}$. $\mathsf{Com}(par_c, x, open)$ outputs a commitment com to $x \in \mathcal{M}$ and random value $open \in \mathcal{R}$. $\mathsf{VfCom}(par_c, com, x, open)$ outputs 1 if com is a commitment to $x \in \mathcal{M}$ and random value $open \in \mathcal{R}$ or 0 otherwise. A commitment scheme must fulfill the hiding and binding properties [37].

8.2 Our Priced Oblivious Transfer Scheme

Our construction POT for priced oblivious transfer involves a provider $\mathcal{P}$ and a user $\mathcal{U}$ and is parameterized by a message space $\mathcal{M} = \mathbb{G}_1$, a number of messages N, a universe of prices $U_{pr} = [0, p_{max}]$ and a universe of accounts $U_{ac} = [0, ac_{max}]$, where $ac_{max} = p_{max} \cdot N$. Construction POT uses a commitment scheme (CSetup, Com, VfCom), a signature scheme (KeyGen_1, Sign_1, VfSig_1) that signs messages in $\mathbb{G}_1$ and $\mathbb{G}_2$, a signature scheme (KeyGen_2, Sign_2, VfSig_2) that signs messages in $\mathbb{Z}_p$, a double trapdoor public-key encryption scheme, and a NIPK scheme (PKSetup, PKProve, PKVerify) for the relations R_1, R_2 and R_3 described in our construction. We employ a NIPK scheme where the setup algorithm PKSetup is common for those relations. Our scheme uses a common references string, which is computed by a trusted party as follows.

- Run $\varPhi = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \tilde{g}) \leftarrow \mathsf{BMSetup}(1^k)$, $par_c \leftarrow \mathsf{CSetup}(1^k)$ and $par_{PK} \leftarrow \mathsf{PKSetup}(1^k)$.
- Pick random $a, b, c \leftarrow \mathbb{Z}_p$ and set $(h_1, h_2, h_3, \tilde{h}_1, \tilde{h}_2, \tilde{h}_3) = (g^a, g^b, g^c, \tilde{g}^a, \tilde{g}^b, \tilde{g}^c)$.
- Set $crs \leftarrow (\varPhi, h_1, h_2, h_3, \tilde{h}_1, \tilde{h}_2, \tilde{h}_3, par_c, par_{PK})$.

The crs is used by $\mathcal{P}$ and $\mathcal{U}$. In the following, we describe the algorithms of our scheme.
$\mathsf{InitP}(1^k, m_1, p_1, \ldots, m_N, p_N)$.

- $\mathcal{P}$ picks random $x_1, x_2 \leftarrow \mathbb{Z}_p$ and sets $(w_1, w_2) = (h_3^{1/x_1}, h_3^{1/x_2})$ and $(\tilde{w}_1, \tilde{w}_2) = (\tilde{h}_3^{1/x_1}, \tilde{h}_3^{1/x_2})$.
- $\mathcal{P}$ runs $\mathsf{KeyGen}_1(1^k, 3)$ to obtain (pk_1, sk_1) and $\mathsf{KeyGen}_2(1^k, 1)$ to obtain (pk_2, sk_2).
- For $i = 1$ to N, $\mathcal{P}$ picks $r_1, r_2 \leftarrow \mathbb{Z}_p$, runs $s_i \leftarrow \mathsf{Sign}_1(sk_1, \langle w_1^{r_1}, w_2^{r_2}, \tilde{g}^{p_i} \rangle)$ and sets $t_i \leftarrow (w_1^{r_1}, w_2^{r_2}, h_1^{r_1}, h_2^{r_2}, m_i \cdot h_3^{r_1+r_2}, s_i, p_i)$. The tuple $(w_1^{r_1}, w_2^{r_2}, h_1^{r_1}, h_2^{r_2}, m_i \cdot h_3^{r_1+r_2})$ is a double trapdoor encryption of m_i.
- For $j = 0$ to ac_{max}, $\mathcal{P}$ runs $s'_j \leftarrow \mathsf{Sign}_2(sk_2, j)$.
- $\mathcal{P}$ sets $com \leftarrow \perp$ and $st_p \leftarrow (x_1, x_2, \tilde{w}_1, \tilde{w}_2, pk_1, pk_2, com)$.
- $\mathcal{P}$ sets $M_{init} \leftarrow (w_1, w_2, \tilde{w}_1, \tilde{w}_2, pk_1, pk_2, \langle s'_j \rangle_{j=0}^{ac_{max}}, t_1, \ldots, t_N \rangle)$.
- $\mathcal{P}$ outputs (st_p, M_{init}).

$\mathsf{InitU}(1^k, M_{init})$.

- For $j = 0$ to ac_{max}, $\mathcal{U}$ aborts if $1 \neq \mathsf{VfSig}_2(pk_2, s'_j, j)$.
- For $i = 1$ to N, $\mathcal{U}$ parses t_i as $(c_1, c_2, c_3, c_4, c_5, s_i, p_i)$ and aborts if $p_i \notin U_{pr}$, or if $1 \neq \mathsf{VfSig}_1(pk_1, \langle c_1, c_2, \tilde{g}^{p_i} \rangle, s_i)$, or if $e(c_1, \tilde{h}_1) \neq e(c_3, \tilde{w}_1)$, or if $e(c_2, \tilde{h}_2) \neq e(c_4, \tilde{w}_2)$.
- $\mathcal{U}$ sets $st_u \leftarrow (w_1, w_2, \tilde{w}_1, \tilde{w}_2, pk_1, pk_2, \langle s'_j \rangle_{j=0}^{ac_{max}}, t_1, \ldots, t_N, ac \leftarrow 0, com \leftarrow \perp, open \leftarrow \perp)$.
- $\mathcal{U}$ outputs $(st_u, p_1, \ldots, p_N)$.

$\mathsf{PayU}(st_u, p)$.

- $\mathcal{U}$ aborts if $p + ac \notin [0, ac_{max}]$.
- If $com = \perp$ and $open = \perp$, $\mathcal{U}$ sets $open' \leftarrow 0$, else picks random $open' \leftarrow \mathcal{R}$.
- $\mathcal{U}$ sets $ac' \leftarrow ac + p$, runs $com' \leftarrow \mathsf{Com}(par_c, ac', open')$, picks $l_1, l_2 \leftarrow \mathbb{Z}_p$ and encrypts p as $(c_1, c_2, c_3) \leftarrow (w_1^{l_1}, w_2^{l_2}, p \cdot h_3^{l_1+l_2})$.

- If $com \neq \perp$ and $open \neq \perp$, $\mathcal{U}$ sets $ins \leftarrow (par_c, com, com', p, pk_2)$ and $wit \leftarrow (ac, ac', s'_{ac'}, open, open')$, and runs $\pi \leftarrow \mathsf{PKProve}(par_{PK}, wit, ins)$ for the relation

$$\begin{aligned} R_1 = \{(ins, wit) : 1 = \mathsf{VfCom}(par_c, com, ac, open) \wedge \\ 1 = \mathsf{VfCom}(par_c, com', ac', open') \ \wedge \ ac' = ac + p \wedge \\ 1 = \mathsf{VfSig}_2(pk_2, ac', s'_{ac'})\} \end{aligned}$$

- $\mathcal{U}$ replaces $(ac, com, open)$ in st_u by $(ac', com', open')$.
- If $com = \perp$ and $open = \perp$, $\mathcal{U}$ sets $M_{pay} \leftarrow (c_1, c_2, c_3)$, else sets $M_{pay} \leftarrow (\pi, com', c_1, c_2, c_3\rangle)$.
- $\mathcal{U}$ outputs st_u and M_{pay}.

$\mathsf{PayP}(st_p, M_{pay})$.

- If $com = \perp$, $\mathcal{P}$ parses M_{pay} as (c_1, c_2, c_3), else parses M_{pay} as $(\pi, com', c_1, c_2, c_3)$.
- $\mathcal{P}$ decrypts $p \leftarrow c_3/(c_1^{x_1} c_2^{x_2})$.
- If $com \neq \perp$, $\mathcal{P}$ sets $ins \leftarrow (par_c, com, com', p, pk_2)$ and aborts if $1 \neq \mathsf{PKVerify}(par_{PK}, ins, \pi)$.
- If $com = \perp$, $\mathcal{P}$ sets $open' \leftarrow 0$ and runs $com' \leftarrow \mathsf{Com}(par_c, p, open')$.
- $\mathcal{P}$ replaces com by com' in st_p.
- $\mathcal{P}$ outputs st_p and p.

$\mathsf{ReqU}(st_u, i)$.

- $\mathcal{U}$ aborts if $i \notin [1, N]$ or if st_u contains the values (i', y_1, y_2, d_1, d_2).
- $\mathcal{U}$ parses t_i as $(c_1, c_2, c_3, c_4, c_5, s_i, p_i)$ and aborts if $ac < p_i$.
- $\mathcal{U}$ picks random $y_1, y_2 \leftarrow \mathbb{Z}_p$ and computes $(d_1, d_2) \leftarrow (c_1 \cdot w_1^{y_1}, c_2 \cdot w_2^{y_2})$ and $(u_1, u_2) \leftarrow (h_3^{y_1}, h_3^{y_2})$.
- $\mathcal{U}$ sets $ac' \leftarrow ac - p_i$, picks random $open' \leftarrow \mathcal{R}$ and computes $com' \leftarrow \mathsf{Com}(par_c, ac', open')$.
- $\mathcal{U}$ sets $ins \leftarrow (par_c, com, com', pk_1, pk_2, d_1, d_2, \tilde{h}_3, \tilde{w}_1, \tilde{w}_2)$ and $wit \leftarrow (ac, ac', open, open', p_i, s_i, s'_{ac'}, c_1, c_2, u_1, u_2)$ and runs $\pi \leftarrow \mathsf{PKProve}(par_{PK}, wit, ins)$ for a relation

$$\begin{aligned} R_2 = \{(ins, wit) : 1 = \mathsf{VfCom}(par_c, com, ac, open) \wedge \\ 1 = \mathsf{VfCom}(par_c, com', ac', open') \ \wedge \ ac' = ac - p_i \wedge \\ 1 = \mathsf{VfSig}_1(pk_1, \langle c_1, c_2, \tilde{g}^{p_i} \rangle, s_i) \wedge \\ 1 = \mathsf{VfSig}_2(pk_2, ac', s'_{ac'}) \wedge \\ e(c_1, \tilde{h}_3) e(u_1, \tilde{w}_1) = e(d_1, \tilde{h}_3) \wedge \\ e(c_2, \tilde{h}_3) e(u_2, \tilde{w}_2) = e(d_2, \tilde{h}_3)\} \end{aligned}$$

- $\mathcal{U}$ sets $st_u \leftarrow st_u \cup (i, y_1, y_2, d_1, d_2)$.
- $\mathcal{U}$ sets $M_{req} \leftarrow (d_1, d_2, \pi, com')$.
- $\mathcal{U}$ outputs st_u and M_{req}.

$\mathsf{ReqP}(st_p, M_{req})$.

- $\mathcal{P}$ aborts if st_p contains the values (d'_1, d'_2).
- $\mathcal{P}$ sets $ins \leftarrow (par_c, com, com', pk_1, pk_2, d_1, d_2, \tilde{h}_3, \tilde{w}_1, \tilde{w}_2)$ and aborts if $1 \neq \mathsf{PKVerify}(par_{PK}, ins, \pi)$.
- $\mathcal{P}$ replaces com by com' in st_p and sets $st_p \leftarrow st_p \cup (d_1, d_2)$.
- $\mathcal{P}$ outputs st_p.

$\mathsf{SellP}(st_p)$.

- $\mathcal{P}$ aborts if st_p does not contain the values (d_1, d_2).
- $\mathcal{P}$ calculates $(z_1, z_2) \leftarrow (d_1^{x_1}, d_2^{x_2})$ and $z \leftarrow z_1 \cdot z_2$.
- $\mathcal{P}$ sets $ins \leftarrow (\tilde{w}_1, \tilde{w}_2, d_1, d_2, \tilde{h}_3, z)$ and $wit \leftarrow (z_1, z_2)$ and runs $\pi \leftarrow \mathsf{PKProve}(par_{PK}, wit, ins)$ for a relation

$$\begin{aligned} R_3 = \{(ins, wit) : e(z_1, \tilde{w}_1) = e(d_1, \tilde{h}_3) \wedge \\ e(z_2, \tilde{w}_2) = e(d_2, \tilde{h}_3) \ \wedge \ e(z_1, \tilde{h}_3) e(z_2, \tilde{h}_3) = e(z, \tilde{h}_3)\} \end{aligned}$$

- $\mathcal{P}$ deletes (d_1, d_2) from st_p and sets $M_{sell} \leftarrow (z, \pi)$.
- $\mathcal{P}$ outputs st_p and M_{sell}.

$\mathsf{SellU}(st_u, M_{sell})$.

- $\mathcal{U}$ aborts if st_u does not store a tuple (i, y_1, y_2, d_1, d_2).
- $\mathcal{U}$ sets $ins \leftarrow (\tilde{w}_1, \tilde{w}_2, d_1, d_2, \tilde{h}_3, z)$ and aborts if $1 \neq \mathsf{PKVerify}(par_{PK}, ins, \pi)$.
- $\mathcal{U}$ parses t_i as $(c_1, c_2, c_3, c_4, c_5, s_i, p_i)$ and computes $m_i \leftarrow c_5/(z \cdot h_3^{-y_1} \cdot h_3^{-y_2})$.
- $\mathcal{U}$ deletes (i, y_1, y_2, d_1, d_2) from st_u.
- $\mathcal{U}$ outputs st_u and m_i.

8.3 Security Analysis

In the full version, we prove that our construction POT securely realizes our ideal functionality for priced oblivious transfer $\mathcal{F}_{\mathrm{POT}}$ in the universal composability framework. Our construction is secure in a hybrid model where parties in the real world use ideal functionalities for common reference string $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{CRS.Setup}}$ and for authenticated channels $\mathcal{F}_{\mathrm{AUT}}$.

The user privacy property holds if the commitment scheme (CSetup, Com, VfCom) is hiding and if the scheme (PKSetup, PKProve, PKVerify) is zero-knowledge and sound. We note that soundness is implied by simulation-sound extractability.

The sender security property holds if the scheme (PKSetup, PKProve, PKVerify) is zero-knowledge and simulation-sound extractable, if the signature schemes (KeyGen_1, Sign_1, VfSig_1) and (KeyGen_2, Sign_2, VfSig_2) are existentially unforgeable, if the double trapdoor encryption scheme fulfills the property of indistinguishability under chosen plaintext attack, and if the commitment scheme (CSetup, Com, VfCom) is binding.

9 Efficiency Analysis of Our CAS

In this section we analyze the efficiency of our CAS proposals. Results for PPC-CAS are provided in Section 9.1, along with a comparison to the scheme in [10]. The results for PPV-CAS are summarized in Section 9.2.

Our platform selection for the provider side is an off-the-shelf laptop running Linux on an Intel Core i5-3317U CPU at 1.70 GHz. On the user side, we select a Beaglebone Black equipped with a 32-bit ARM Cortex-A8 processor and clocked at 1 GHz. This platform is representative of mobile devices, as the same processor core can be found in early-generation smartphones such as Apple iPhone 4, Samsung Galaxy S or Google Nexus S. Our implementation is coded in C language to ensure platform independence. We leverage on the Pairing Based Cryptography (PBC) library[4] developed by Ben Lynn to implement bilinear maps. In particular, we employ type A1 curves to instantiate the ABE constructions and type D curves with a base field length of

[4] https://crypto.stanford.edu/pbc/

201 bits and an embedding degree of 6 to instantiate the POT constructions. Note that PBC can be ported and executed in Android platforms, see e.g. [38], and thus we expect the performance of an Android application to be very similar to that of our Beaglebone implementation.

To instantiate the cryptographic building blocks of our POT protocol, we employ the commitment scheme in [37] for (CSetup, Com, VfCom), the signature scheme in [16] for (KeyGen_1, Sign_1, VfSig_1), and the weakly secure signature scheme in [39] for (KeyGen_2, Sign_2, VfSig_2). For the sake of efficiency, to instantiate (PKSetup, PKProve, PKVerify), we employ the Fiat-Shamir transform, which is simulation-sound extractable in the random oracle model [40]. Therefore, this instantiation does not achieve universal composability, which is achieved when using other simulation-sound extractable NIPK scheme or the Groth-Sahai proof system [41] in the manner of [15].

9.1 Efficiency Analysis of PPC-CAS

In Table 4, we summarize the performance results for each phase of the PPC-CAS schemes. Results on the left hand side of the table are given depending on the parameters of the system, namely, the number of users $|U|$, the number of TV channels $|N|$, the number of channels the user is subscribed to $|A|$, and the average number of user subscriptions per channel $|I|$. The results on the right hand side correspond to a particular initialization of the system with $|U| = 100k$ users, $|N| = 100$ TV channels, $|A| = 10$ channels/user (in average), and $|I| = 10k$ users/channel (in average, assuming uniform distribution). Note that in our scheme, the value of $|A|$ corresponds to the number of attributes. This is however not the case in [10], where the number of attributes is fixed to four.

In our scheme, the running time of the setup algorithm is dominated by the number of users $|U|$ which, at the same time, determines the size of the public parameters. The running time of the key generation algorithm and the key size grow with $|A|$, which in this case corresponds to the number of attributes in BABE. More specifically, the execution time of KeyGen_0 is 50 ms and is independent of the number of attributes, whereas the running time of KeyGen_1 demands 0.8 ms for each user attribute. The running time of the POT initialization phase is dominated by the loops iterating over the number of messages N and the universe of accounts U_{ac}, which in our application correspond to the number of channels $|N|$ and users $|U|$, respectively. Timings on the user side are significantly larger due to the use of a less powerful computing platform, but also due to the fact that our building blocks demand the computation of bilinear maps during signature verification but not during signature generation. The running times of the payment and purchase algorithms are constant and independent of any parameter of the scheme. The running time of the encryption algorithm grows with the number of users $|I|$ that have subscribed to that particular channel. Note however that the ciphertext size is constant at 1 040 bytes. Finally for decryption, the running time grows similarly with $|I|$.

The comparison with the results obtained from the scheme in [10] yields some clear differences. First, we observe that both running time and communication in the setup algorithm are independent of any parameter in the PPC-CAS construction and significantly lower than in our scheme. Second, the complexity of the user registration phase (i.e. invoking user key generation) increases with $|A|$. This is the same behaviour as our PPC-CAS, yet in this case the scheme in [10] performs worse. Third, the payment and purchase algorithms are not present in the comparison, since, in contrast to our work, the scheme in [10] does not protect users' privacy and therefore omits any description of the payment mechanisms. Finally we observe that only the encryption algorithm depends on $|I|$, and that both encryption and decryption are more costly than in our scheme. Moreover, the CAS in [10] needs key updates for revocation purposes.

In summary, except for the setup phase, our CAS is more efficient. The reason is that our CAS uses BABE, which provides an efficient direct revocation method. The CAS in [10] needs to include attributes for revocation purposes in the user keys and in the ciphertexts, which affects negatively the efficiency of the user registration and SEK update phases.

Independently of this observation, the results provided in Table 4 prove that our PPC-CAS scheme is practical. The time required for encrypting and decrypting the SEK is perfectly assumable, and the size of the ciphertext that is broadcast is only 1040 bytes. Recall that the SEK is updated weekly. The most time consuming result corresponds to the verification for $\mathcal{U}$, which requires over 2 hours for $|U| = 100k$. While this seems a prohibitive result, we note that, in practice, at initialization the mobile pay-tv users receive the same message from the provider. Therefore, the verification step can be carried out by a trusted authority. The authority can verify the message once, compute its hash and sign the hash. That way, each user only needs to compute the hash of the message and verify the signature computed by the authority.

9.2 Efficiency Analysis of PPV-CAS

The results of our PPV-CAS implementation are summarized in Table 5 (initialization phase) and Table 6 (payment and purchase phases). Recall that our PPV-CAS leverages on POT to achieve privacy but, in contrast to our PPC-CAS scheme, it does not rely on BABE.

Similar to the case of PPC-CAS, the running time of the initialization phase is dominated by the loops iterating over the number of messages N and the universe of accounts U_{ac}. We give in Table 5 and overview of the timings for exemplary values of ac_{max} and N. Independent of the parameter selections, we stress once again that the verification can be outsourced to a third party in a real setting. Results for the post-initialization phases of our PPV-CAS scheme are given in Table 6. The execution times and communication overhead of each phases are perfectly assumable. The payment and the sell phases require in particular less than 1 second to execute, and the request phase is completed within 2.5 seconds. The bottleneck for each phase corresponds to proving the relations R_1, R_2 and R_3, respectively. The length of the messages exchanged between $\mathcal{U}$ and $\mathcal{P}$ during a purchase is below 2 kbytes.

All in all, it is once again clear from the results that deploying our PPC-CAS scheme in mobile TV settings is perfectly feasible.

TABLE 4
Performance analysis of our PPC-CAS and comparison.

	Protocol Phase	Timings (in ms)		Communication (in bytes)	Timings		Communication
		$\mathcal{P}$	$\mathcal{U}$	$\mathcal{P} \leftrightarrow \mathcal{U}$	$\mathcal{P}$	$\mathcal{U}$	$\mathcal{P} \leftrightarrow \mathcal{U}$
Our Scheme	Setup						
	Setup	$50\vert U\vert$	—	$2\,401 + 50\vert U\vert$	5 000 s	—	4 885 Kbytes
	InitP,InitU	$100\vert N\vert + \vert U\vert$	$882\vert N\vert + 78\vert U\vert$	$1040\vert N\vert + 52\vert U\vert$	110 s	7 888 s	5 180 Kbytes
	User Registration						
	KeyGen$_0$,KeyGen$_1$	$50 + 0.8\vert A\vert$	—	$520 + 260 \cdot \vert A\vert$	58 ms	—	3 120 bytes
	Payment						
	PayU,PayP	61	462	288	61 ms	462 ms	288 bytes
	Purchase						
	ReqU,ReqP	174	2 390	1 235	174 ms	2 390 ms	1 235 bytes
	SellP,SellU	71	720	491	71 ms	720 ms	491 bytes
	SEK Update						
	Enc	$106 + 0.016\vert I\vert$	—	1 040	266 ms	—	1 040 bytes
	Dec	—	$1\,580 + 0.16\vert I\vert$	—	—	3 180 ms	—
Scheme [10]	Setup	140	—	1 560	140 ms	—	1 560 bytes
	User Registration	$2\,352 \cdot \vert A\vert$	—	$2\,340 \cdot \vert A\vert$	23.5 s	—	23 400 bytes
	SEK Update						
	Enc	2 380	—	3 120	2 380 ms	—	3 120 bytes
	Dec	—	59 040	—	—	59 s	—

TABLE 5
PPV-CAS: Initialization phase.

PARAMETERS		TIMINGS		COMMUNICATION
ac_{max}	N	$\mathcal{P}$	$\mathcal{U}$	$\mathcal{P} \leftrightarrow \mathcal{U}$
10 k	50	15 s	824 s	559 kbytes
	100	20 s	868 s	610 kbytes
	200	30 s	956 s	711 kbytes
100 k	50	105 s	7 844 s	5 129 kbytes
	100	110 s	7 888 s	5 180 kbytes
	200	120 s	7 976 s	5 282 kbytes

TABLE 6
PPV-CAS: Payment, request and sell phases.

INTERFACE	TIMINGS		COMMUNICATION
	$\mathcal{P}$	$\mathcal{U}$	$\mathcal{P} \leftrightarrow \mathcal{U}$
Payment Phase	61 ms	462 ms	288 bytes
Request Phase	174 ms	2 390 ms	1 235 bytes
Sell Phase	71 ms	720 ms	491 bytes

10 Conclusion

We proposed a pay-per-view and a pay-per-channel CAS that protect user privacy. Both our CAS employ priced oblivious transfer to allow the user to purchase programs and to subscribe to channels without disclosing sensitive information to the service provider. Our pay-per-channel CAS also employs broadcast attribute-based encryption, which provides broadcast efficiency, simple revocation, low storage overhead and protection against user collusion. Our implementation shows that our CAS is perfectly feasible for mobile devices.

References

[1] S.-Y. Wang and C.-S. Laih, "Efficient key distribution for access control in pay-tv systems," *Multimedia, IEEE Transactions on*, vol. 10, no. 3, pp. 480–492, 2008.

[2] B. Liu, W. Zhang, and T. Jiang, "A scalable key distribution scheme for conditional access system in digital pay-tv system," *Consumer Electronics, IEEE Transactions on*, vol. 50, no. 2, pp. 632–637, 2004.

[3] H.-M. Sun, C.-M. Chen, and C.-Z. Shieh, "Flexible-pay-per-channel: A new model for content access control in pay-tv broadcasting systems," *Multimedia, IEEE Transactions on*, vol. 10, no. 6, pp. 1109–1120, 2008.

[4] L.-Y. Yeh and J.-L. Huang, "A conditional access system with efficient key distribution and revocation for mobile pay-tv systems," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, vol. 9, no. 3, p. 18, 2013.

[5] Y.-L. Huang, S. Shieh, F.-S. Ho, and J.-C. Wang, "Efficient key distribution schemes for secure media delivery in pay-tv systems," *Multimedia, IEEE Transactions on*, vol. 6, no. 5, pp. 760–769, 2004.

[6] W. T. Zhu, "A cost-efficient secure multimedia proxy system," *Multimedia, IEEE Transactions on*, vol. 10, no. 6, pp. 1214–1220, 2008.

[7] J.-Y. Kim and H.-K. Choi, "Improvements on sun's conditional access system in pay-tv broadcasting systems," *Multimedia, IEEE Transactions on*, vol. 12, no. 4, pp. 337–340, 2010.

[8] S. F. Yeung, J. C. Lui, and D. K. Yau, "A multikey secure multimedia proxy using asymmetric reversible parametric sequences: theory, design, and implementation," *Multimedia, IEEE Transactions on*, vol. 7, no. 2, pp. 330–338, 2005.

[9] H.-M. Sun and M.-C. Leu, "An efficient authentication scheme for access control in mobile pay-tv systems," *Multimedia, IEEE Transactions on*, vol. 11, no. 5, pp. 947–959, 2009.

[10] Z. Wan, J. Liu, R. Zhang, R. H. Deng *et al.*, "A collusion-resistant conditional access system for flexible-pay-per-channel pay-tv broadcasting," *IEEE transactions on multimedia*, vol. 15, no. 6, pp. 1353–1364, 2013.

[11] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology–EUROCRYPT 2005*. Springer, 2005, pp. 457–473.

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

[13] B. Aiello, Y. Ishai, and O. Reingold, "Priced oblivious transfer: How to sell digital goods," in *Advances in CryptologyEUROCRYPT 2001*. Springer, 2001, pp. 119–135.

[14] N. Attrapadung and H. Imai, "Conjunctive broadcast and attribute-based encryption," in *Pairing-Based Cryptography–Pairing 2009*. Springer, 2009, pp. 248–265.

[15] A. Rial, M. Kohlweiss, and B. Preneel, "Universally composable adaptive priced oblivious transfer," in *Pairing-Based Cryptography–Pairing 2009*. Springer, 2009, pp. 231–247.

[16] M. Abe, J. Groth, K. Haralambiev, and M. Ohkubo, "Optimal structure-preserving signatures in asymmetric bilinear groups," in *CRYPTO*, ser. Lecture Notes in Computer Science, P. Rogaway, Ed., vol. 6841. Springer, 2011, pp. 649–666.

[17] A. Fiat and M. Naor, "Broadcast encryption," in *Advances in CryptologyCRYPTO93*. Springer, 1994, pp. 480–491.

[18] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proceedings of the 15th ACM conference on Computer and communications security*, 2008, pp. 417–426.

[19] N. Attrapadung and H. Imai, "Attribute-based encryption supporting direct/indirect revocation modes," in *Cryptography and Coding*. Springer, 2009, pp. 278–300.

[20] B. Wesolowski and P. Junod, "Ciphertext-policy attribute-based broadcast encryption with small keys," in *International Conference on Information Security and Cryptology*. Springer, 2015, pp. 53–68.

[21] Q. Li and F. Zhang, "A fully secure attribute based broadcast encryption scheme." *IJ Network Security*, vol. 17, no. 3, pp. 255–263, 2015.

[22] T. Kitagawa, H. Kojima, N. Attrapadung, and H. Imai, *Efficient and Fully Secure Forward Secure Ciphertext-Policy Attribute-Based Encryption*. Cham: Springer International Publishing, 2015, pp. 87–99. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-27659-5_6

[23] J. Ning, Z. Cao, X. Dong, L. Wei, and X. Lin, "Large universe ciphertext-policy attribute-based encryption with white-box traceability," in *European Symposium on Research in Computer Security*. Springer, 2014, pp. 55–72.

[24] Z. Liu and D. S. Wong, "Practical ciphertext-policy attribute-based encryption: Traitor tracing, revocation, and large universe," in *International Conference on Applied Cryptography and Network Security*. Springer, 2015, pp. 127–146.

[25] M. S. Farash and M. A. Attari, "A provably secure and efficient authentication scheme for access control in mobile pay-tv systems," *Multimedia Tools and Applications*, vol. 75, no. 1, pp. 405–424, 2016.

[26] M. N. Asghar, M. Fleury, and S. Makki, "Interoperable conditional access with video selective encryption for portable devices," *Multimedia Tools and Applications*, pp. 1–14, 2016.

[27] J. Camenisch, M. Dubovitskaya, and G. Neven, "Unlinkable priced oblivious transfer with rechargeable wallets," in *Financial Cryptography and Data Security*. Springer, 2010, pp. 66–81.

[28] ——, "Oblivious transfer with access control," in *Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9-13, 2009*, 2009, pp. 131–140. [Online]. Available: http://doi.acm.org/10.1145/1653662.1653679

[29] S. Coull, M. Green, and S. Hohenberger, "Controlling access to an oblivious database using stateful anonymous credentials," in *Public Key Cryptography–PKC 2009*. Springer, 2009, pp. 501–520.

[30] Y. Zhang, M. H. Au, D. S. Wong, Q. Huang, N. Mamoulis, D. W. Cheung, and S.-M. Yiu, "Oblivious transfer with access control: realizing disjunction without duplication," in *Pairing-Based Cryptography-Pairing 2010*. Springer, 2010, pp. 96–115.

[31] A. Rial and B. Preneel, "Blind attribute-based encryption and oblivious transfer with fine-grained access control," in *Proc. 2010th Benelux Workshop Information and System Security (WISSec10)*, 2010, pp. 1–20.

[32] J. Camenisch, M. Dubovitskaya, G. Neven, and G. M. Zaverucha, "Oblivious transfer with hidden access control policies," in *Public Key Cryptography–PKC 2011*. Springer, 2011, pp. 192–209.

[33] A. Rial and B. Preneel, "Optimistic fair priced oblivious transfer," in *Progress in Cryptology–AFRICACRYPT 2010*. Springer, 2010, pp. 131–147.

[34] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," DTIC Document, Tech. Rep., 2004.

[35] J. Groth, "Simulation-sound nizk proofs for a practical language and constant size group signatures," in *Advances in Cryptology–ASIACRYPT 2006*. Springer, 2006, pp. 444–459.

[36] S. Goldwasser, S. Micali, and R. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM J. Comput.*, vol. 17, no. 2, pp. 281–308, 1988.

[37] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in CryptologyCRYPTO91*. Springer, 1992, pp. 129–140.

[38] M. Pirker, D. Slamanig, and J. Winter, "Practical privacy preserving cloud resource-payment for constrained clients," in *Privacy Enhancing Technologies*, 2012, pp. 201–220.

[39] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Advances in Cryptology-EUROCRYPT 2004*. Springer, 2004, pp. 56–73.

[40] S. Faust, M. Kohlweiss, G. A. Marson, and D. Venturi, "On the non-malleability of the fiat-shamir transform," in *Progress in Cryptology-INDOCRYPT 2012*. Springer, 2012, pp. 60–79.

[41] J. Groth and A. Sahai, "Efficient non-interactive proof systems for bilinear groups," in *Advances in Cryptology–EUROCRYPT 2008*. Springer, 2008, pp. 415–432.