Information Processing Letters 126 (2017) 7–11

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Issuer-free oblivious transfer with access control revisited

Alfredo Rial

University of Luxembourg, Luxembourg

ARTICLE INFO

Article history:
Received 16 March 2016
Received in revised form 21 May 2017
Accepted 22 May 2017
Available online 26 May 2017
Communicated by L. Viganò

Keywords:
Cryptography
Oblivious transfer
Access control
Ideal functionality

ABSTRACT

Oblivious transfer with access control (OTAC) is an extension of oblivious transfer where each message is associated with an access control policy. A receiver can obtain a message only if her attributes satisfy the access control policy for that message. In most schemes, the receiver's attributes are certified by an issuer. Recently, two Issuer-Free OTAC protocols have been proposed. We show that the security definition for Issuer-Free OTAC fulfilled by those schemes poses a problem. Namely, the sender is not able to attest whether a receiver possesses a claimed attribute. Because of this problem, in both Issuer-Free OTAC protocols, any malicious receiver can obtain any message from the sender, regardless of the access control policy associated with the message. To address this problem, we propose a new security definition for Issuer-Free OTAC. Our definition requires the receiver to prove in zero-knowledge to the sender that her attributes fulfill some predicates. Our definition is suitable for settings with multiple issuers because it allows the design of OTAC protocols where the receiver, when accessing a record, can hide the identity of the issuer that certified her attributes.

1. Introduction

Oblivious transfer (OT) [1] is a two-party protocol between a sender and a receiver. The sender receives as input N messages $(m_1, \dots, m_N)$, while the receiver gets K selection values $(\sigma_1, \dots, \sigma_K)$. As output, the receiver gets the messages $(m_{\sigma_1}, \dots, m_{\sigma_K})$. Sender security requires that the receiver gets no information on the other messages, while receiver privacy requires that the sender does not learn any information on $(\sigma_1, \dots, \sigma_K)$.

Oblivious transfer with access control (OTAC) [2] allows the sender to control access to the messages. The sender receives as input $(m_1, \mathbb{P}_1, \dots, m_N, \mathbb{P}_N)$, where $(\mathbb{P}_1, \dots, \mathbb{P}_N)$ are access control policies for each of the messages. Each receiver possesses a set of attributes $\mathbb{A}$ and is able to obtain the message m_i only if $\mathbb{A}$ satisfies $\mathbb{P}_i$.

OTAC schemes involve three types of parties: the sender, who possesses a database $(m_1, \mathbb{P}_1, \dots, m_N, \mathbb{P}_N)$; the issuer, who certifies the receivers' attributes $\mathbb{A}$ and issues credentials to receivers; the receivers, who first get their attributes certified by the issuer and subsequently employ the issued credentials to access the sender's database.

Receiver privacy requires that the sender does not learn any information on the messages the receiver obtains or on the receiver's attributes. Sender privacy requires that the receiver does not learn any information on messages that were not requested or on messages whose access control policy is not fulfilled by the receiver's attributes. Additionally, in some schemes, the access control policies are hidden from the receivers [3], while in other schemes they are public [2,4]. We describe in detail the security definition for OTAC with public policies in Section 2.

Recently, Guleria and Dutta propose Issuer-Free OTAC with public policies [5,6]. In Issuer-Free OTAC, the role of the issuer is performed by the sender. In this paper, we show that the security definition for issuer-free OTAC in [5,6] poses a problem. In a nutshell, the security definition for OTAC with public policies proposed by Camenisch et al. [2] allows the issuer to learn the receiver's identity and

E-mail address: alfredo.rial@uni.lu.

http://dx.doi.org/10.1016/j.ipl.2017.05.006
0020-0190/© 2017 Elsevier B.V. All rights reserved.

the receiver's attributes in order to attest whether the receiver indeed possesses those attributes. In contrast, in the security definition in [5,6], to protect receiver privacy, the sender learns neither the receiver's identity nor the receiver's attributes, and thus is not able to attest whether the receiver possesses the claimed attributes.

This has serious implications on the security of the protocols proposed in [5,6]. In those protocols, the sender *always* proceeds as if the receiver did possess those attributes without performing any check. This allows any malicious receiver to be issued any attribute, which allows this receiver to obtain any message from the sender, regardless of the access control policy associated with the message. Consequently, the protocols in [5,6] do not enforce any form of access control.

We propose a new security definition for Issuer-Free OTAC. Our definition allows the receiver to prove in zero-knowledge to the sender that her attributes fulfill some predicates. The concrete predicates will depend on the information the sender needs in order to attest the receiver's attributes. In the typical setting where attributes need to be certified by an issuer, we show that our new functionality is useful to handle multiple issuers.

2. Oblivious transfer with access control

Camenisch et al. [2] propose an ideal functionality $\mathcal{F}_{\text{OTAC}}$ for oblivious transfer with access control (OTAC). In this section, we recall that ideal functionality. The interaction between $\mathcal{F}_{\text{OTAC}}$, the sender $\mathcal{E}$, the issuer $\mathcal{I}$, and the receivers $\mathcal{R}_1, \ldots, \mathcal{R}_M$ takes place through the interfaces initdb, issue and transfer. The sender $\mathcal{E}$ possesses a list of messages $(m_1, \ldots, m_N)$. These messages are associated with the access control policies $(\mathbb{P}_1, \ldots, \mathbb{P}_N)$. An access control policy describes the attributes that a receiver must possess in order to be allowed to obtain a message. The attributes that a receiver possesses are certified by $\mathcal{I}$. $\mathcal{F}_{\text{OTAC}}$ maintains an initially empty set $\mathbb{A}_m (m \in [1, M])$ for each of the receivers $\mathcal{R}_m$.

Functionality $\mathcal{F}_{\text{OTAC}}$

1. On input $(\text{initdb}, m_1, \mathbb{P}_1, \ldots, m_N, \mathbb{P}_N)$ from $\mathcal{E}$, $\mathcal{F}_{\text{OTAC}}$ stores $(m_1, \mathbb{P}_1, \ldots, m_N, \mathbb{P}_N)$.
2. On input (issue, a) from $\mathcal{R}_m$, $\mathcal{F}_{\text{OTAC}}$ sends $(\text{issue}, \mathcal{R}_m, a)$ to $\mathcal{I}$. $\mathcal{I}$ sends back a bit b. If $b = 1$, $\mathcal{F}_{\text{OTAC}}$ adds the attribute a to $\mathbb{A}_m$ and sends b to $\mathcal{R}_m$, else $\mathcal{F}_{\text{OTAC}}$ simply sends b to $\mathcal{R}_m$.
3. On input $(\text{transfer}, \sigma)$ from $\mathcal{R}_m$, $\mathcal{F}_{\text{OTAC}}$ proceeds as follows. If $(m_1, \mathbb{P}_1, \ldots, m_N, \mathbb{P}_N)$ is stored, $\mathcal{F}_{\text{OTAC}}$ sends transfer to $\mathcal{E}$. $\mathcal{E}$ sends back a bit b. If $b = 1$ and the attribute set $\mathbb{A}_m$ fulfills the policy $\mathbb{P}_\sigma$, $\mathcal{F}_{\text{OTAC}}$ sends the message m_σ to $\mathcal{R}_m$. If $b = 0$ or if $(m_1, \mathbb{P}_1, \ldots, m_N, \mathbb{P}_N)$ is not stored, $\mathcal{F}_{\text{OTAC}}$ sends $\perp$ to $\mathcal{R}_m$.

As described in [2], $\mathcal{F}_{\text{OTAC}}$ guarantees the following security properties:

Receiver Privacy. When a receiver $\mathcal{R}_m$ obtains a message m_σ, the sender $\mathcal{E}$ learns neither $\mathcal{R}_m$ nor σ, i.e., in the transfer phase, the receiver remains anonymous and the sender does not learn the message that the receiver obtains. The sender only learns that an unknown receiver gets a message whose access control policy is fulfilled by the receiver's attributes.

Sender Security. A corrupt receiver cannot obtain a message whose access control policy is not fulfilled by the receiver's attributes. Colluding receivers are not able to share their attributes, i.e., a group of colluding receivers is not able to get access to a message whose access control policy is not fulfilled by the attributes of a particular receiver in the group. If a corrupt receiver colludes with the issuer, then the receiver can obtain one record at each transfer phase.

3. Issuer-free oblivious transfer with access control in [5, 6]

We recall the ideal functionality $\mathcal{F}_{\text{IOTAC}}$ for issuer-free OTAC proposed by Guleria and Dutta [5,6]. The difference between $\mathcal{F}_{\text{IOTAC}}$ and the functionality $\mathcal{F}_{\text{OTAC}}$ described in Section 2 is in the issuing phase. Therefore, we only recall the issue interface of $\mathcal{F}_{\text{IOTAC}}$.

Functionality $\mathcal{F}_{\text{IOTAC}}$: interface issue

2. On input (issue, a) from $\mathcal{R}_m$, $\mathcal{F}_{\text{IOTAC}}$ sends issue to $\mathcal{E}$. $\mathcal{E}$ sends back a bit b in response to issue. If $b = 1$, $\mathcal{F}_{\text{IOTAC}}$ adds the attribute a to $\mathbb{A}_m$ and sends b to $\mathcal{R}_m$, else $\mathcal{F}_{\text{IOTAC}}$ does nothing.

As can be seen, in $\mathcal{F}_{\text{IOTAC}}$, in contrast to $\mathcal{F}_{\text{OTAC}}$, the issuer is not present and the issuing phase is executed by the sender $\mathcal{E}$ and by the receiver $\mathcal{R}_m$. Additionally, while in $\mathcal{F}_{\text{OTAC}}$ the issuer receives the identity of the receiver $\mathcal{R}_m$ and the attribute a, in $\mathcal{F}_{\text{IOTAC}}$ the sender receives neither $\mathcal{R}_m$ nor a.

The latter difference creates a problem. In a real protocol that realizes $\mathcal{F}_{\text{OTAC}}$, the issuer can receive the identity of the receiver $\mathcal{R}_m$ and the attribute a. Based on that information, the issuer is able to attest whether $\mathcal{R}_m$ possesses the attribute a, and, in that case, the issuer issues a credential on that attribute to $\mathcal{R}_m$. However, in any real protocol that realizes $\mathcal{F}_{\text{IOTAC}}$, the sender cannot receive any information on $\mathcal{R}_m$ or a whatsoever. (The reason is that, in the ideal protocol, the sender does not receive that information.) In that case, how is the sender supposed to decide whether the receiver possesses that attribute? This has serious implications on the security of the real world protocols that realize $\mathcal{F}_{\text{IOTAC}}$ proposed in [5,6]. In those protocols, the sender does not receive any information on the attributes or on the identity of the receiver in the issuing phase, and in fact the sender *always* proceeds as if the receiver did possess those attributes without performing any check. This allows any malicious receiver to be issued any attribute, which allows this receiver to obtain any message from the sender. Consequently, the real world protocols in [5,6] do not enforce any form of access control.

4. New ideal functionality for issuer-free OTAC

A trivial way to solve the problem in the functionality $\mathcal{F}_{\text{IOTAC}}$ for Issuer-Free OTAC proposed by Guleria and Dutta [5,6] would be to modify the issuing phase so that $\mathcal{F}_{\text{IOTAC}}$ sends to the sender the identity $\mathcal{R}_m$ of the receiver and the attribute a. However, this would weaken the privacy properties of $\mathcal{F}_{\text{IOTAC}}$ in comparison to $\mathcal{F}_{\text{OTAC}}$, since in $\mathcal{F}_{\text{OTAC}}$ the sender does not learn those values. The issuer does learn those values, but it is generally considered that receivers are willing to put more trust in a credential issuer than in a sender.

Therefore, we propose a new functionality that balances receiver privacy and sender security. Our functionality allows the receiver to prove in zero-knowledge some statements about the receiver's attributes. To this end, we parameterized the functionality $\mathcal{F}_{\text{IOTAC}}$ with a relation R. In the issuing phase, the receiver sends to the functionality $\mathcal{F}^R_{\text{IOTAC}}$ an attribute a, a witness wit and an instance ins. $\mathcal{F}^R_{\text{IOTAC}}$ verifies that $(\langle a, wit\rangle, ins) \in R$. (The attribute a is included in the witness for the relation R.) If the verification succeeds, $\mathcal{F}^R_{\text{IOTAC}}$ sends the instance ins to the sender. We formally describe the issuing phase of $\mathcal{F}^R_{\text{IOTAC}}$ below.

Functionality $\mathcal{F}^R_{\text{IOTAC}}$: interface issue

2. On input $(\mathsf{issue}, a, wit, ins)$ from $\mathcal{R}_m$, $\mathcal{F}^R_{\text{IOTAC}}$ aborts if $(\langle a, wit\rangle, ins) \notin R$. Else, $\mathcal{F}^R_{\text{IOTAC}}$ sends (issue, ins) to $\mathcal{E}$. $\mathcal{E}$ sends back a bit b. If $b = 1$, $\mathcal{F}^R_{\text{IOTAC}}$ adds the attribute a to $\mathbb{A}_m$ and sends b to $\mathcal{R}_m$, else $\mathcal{F}^R_{\text{IOTAC}}$ simply sends b to $\mathcal{R}_m$.

In any real world protocol that realizes $\mathcal{F}^R_{\text{IOTAC}}$, in the issuing phase, the sender can learn the fact that the receiver's attributes fulfill the relation R with respect to the instance ins. This allows the design of real world protocols where the receiver can prove to the sender statements in zero-knowledge about her attributes, so that the sender can certify that the receiver possesses those attributes based on the proven statements. The concrete statements that the receiver must prove depend on the information the sender needs in order to attest the receiver's attributes. In Section 4.1, we describe the use of $\mathcal{F}^R_{\text{IOTAC}}$ in the typical case where attributes need to be certified by an issuer.

4.1. Applications of the new ideal functionality $\mathcal{F}^R_{\text{IOTAC}}$

A protocol that realizes $\mathcal{F}_{\text{OTAC}}$ involves the sender $\mathcal{E}$, the issuer $\mathcal{I}$, and the receivers $\mathcal{R}$ and consists of three phases: initdb, issue, and transfer. In the initdb phase, $\mathcal{E}$ encrypts the messages and publishes an encrypted database where each message is associated with an access control policy. In the issue phase, $\mathcal{R}$ sends some attributes to $\mathcal{I}$, $\mathcal{I}$ certifies that $\mathcal{R}$ indeed possesses those attributes, signs $\mathcal{R}$'s attributes by using $\mathcal{I}$'s secret key and sends the signature to $\mathcal{R}$. In the transfer phase, $\mathcal{R}$ chooses the message that she wants to obtain and proves in zero-knowledge to $\mathcal{E}$ that she possesses a signature from $\mathcal{I}$ on attributes that satisfy the access control policy associated with that message. After $\mathcal{E}$ verifies this zero-knowledge proof, $\mathcal{E}$ helps $\mathcal{R}$ to decrypt the message (without learning the message that $\mathcal{R}$ decrypts).

The reason why all the existing OTAC schemes (except the issuer-free ones by Guleria and Dutta) include an issuer is the following. Receivers need to prove that their attributes fulfill the access control policy associated with a message. This requires that some party certifies $\mathcal{R}$'s attributes. In practical applications, that party needs to learn $\mathcal{R}$'s attribute values in order to certify them. This step is unavoidable because, without learning the attribute values (e.g., age, nationality, address, ...) claimed by $\mathcal{R}$, verifying their truthfulness is not possible. For privacy reasons, receivers may be unwilling to reveal their attributes to $\mathcal{E}$. To solve this problem, OTAC schemes include a third party, the issuer, that is only in charge of certifying receiver's attributes and to whom receiver's accept to disclose their attributes. In practice, $\mathcal{I}$ would be an authority such as a municipality or the police.

The issuer-free OTAC schemes by Guleria and Dutta do not work because attribute certification does not take place and thus receivers can claim any attribute they wish and obtain any message. Therefore, since in typical settings issuers are necessary, the reason why we propose a new functionality $\mathcal{F}^R_{\text{IOTAC}}$ is not to eliminate the need of an issuer. Instead, the new functionality allows the creation of OTAC protocols for multiple issuers.

Consider a setting where $\mathcal{E}$ only trusts one issuer to certify $\mathcal{R}$'s attributes. In this case, using the new functionality $\mathcal{F}^R_{\text{IOTAC}}$ is cumbersome. The reason is that the relation R that parameterizes the functionality would require that $\mathcal{R}$ proves in zero-knowledge possession of a credential from the issuer trusted by $\mathcal{E}$. Therefore, in a protocol that realizes $\mathcal{F}^R_{\text{IOTAC}}$, $\mathcal{R}$ would first get his attributes certified by $\mathcal{I}$ to get a credential from $\mathcal{I}$, then $\mathcal{R}$ would prove in zero-knowledge possession of the credential to $\mathcal{E}$, and finally $\mathcal{E}$ would give yet a second credential that signs the same attributes to $\mathcal{R}$. Consequently, in settings where $\mathcal{E}$ trusts only one issuer, the new functionality $\mathcal{F}^R_{\text{IOTAC}}$ should not be used. Instead, the existing functionality $\mathcal{F}_{\text{OTAC}}$ should be used.

However, the new functionality $\mathcal{F}^R_{\text{IOTAC}}$ is appealing when $\mathcal{E}$ trusts more than one issuer. This setting cannot be handled by the existing functionality $\mathcal{F}_{\text{OTAC}}$. Additionally, modifying the functionality and the corresponding protocols to include more than one issuer is not straightforward. The problem is the following. Consider a setting with two issuers $\mathcal{I}_1$ and $\mathcal{I}_2$, where some access control policies require that attributes be certified by $\mathcal{I}_1$ and others by $\mathcal{I}_2$. In that case, when $\mathcal{R}$ proves in zero-knowledge possession of her credential, despite the fact that the attribute values are not revealed to $\mathcal{E}$, $\mathcal{E}$ learns the identity of the issuer of

$\underline{\mathcal{R}(c, A_c, s_c, r_c, st_U, pk_I, pk_E)}$		$\underline{\mathcal{E}(sk_E, pk_E, pk_I, P')}$
Parse $pk_I = (g_1, h_0, h_1, h_2, u, v, w, g_t, h_t, y_I)$		
Parse $pk_E = (\hat{g}_1, \hat{h}_0, \hat{h}_1, \hat{h}_2, \hat{h}_3, \hat{u}, \hat{v}, \hat{w}, \hat{g}_t, \hat{h}_t, y_E)$		
Parse $st_U = (P', z_U)$ and pick random $r_1, t, t' \leftarrow \mathbb{Z}_p$		
Compute $P_1 \leftarrow \hat{h}_1^c \hat{h}_3^{r_1}$, $\bar{A} \leftarrow A_c u^t$ and $B \leftarrow v^t u^{t'}$		
Compute $\mathrm{PK}_1 = \mathrm{PK}\{(z_u, c, r_1, I, s_c, r_c, t, t', \alpha, \beta)$:		
$P' = \hat{h}_0^{z_U} \ \wedge \ P_1 = \hat{h}_1^c \hat{h}_3^{r_1} \ \wedge \ B = v^t u^{t'} \ \wedge$		
$1 = B^{-s_c} v^{\alpha} u^{\beta} \ \wedge \ \frac{e(\bar{A}, y_I)}{e(g_1, g_1)} = e(\bar{A}, g_1)^{-s_c} e(u, y_I)^t \cdot$		
$e(u, g_1)^{\alpha} e(h_2, g_1)^{r_c} e(h_0, g_1)^{z_U} e(h_1, g_1)^c\}$		
Send $(P', P_1, \mathrm{PK}_1, \bar{A}, B)$	→	Verify PK_1 and parse $sk_E = x_E$
		Parse $pk_E = (\hat{g}_1, \hat{h}_0, \hat{h}_1, \hat{h}_2, \hat{h}_3, \hat{u}, \hat{v}, \hat{w}, \hat{g}_t, \hat{h}_t, y_E)$
		Pick random $r_2, s'_c \leftarrow \mathbb{Z}_p$
		Compute $A'_c \leftarrow (\hat{g}_1 P' P_1 \hat{h}_2^I \hat{h}_3^{r_2})^{1/(x_E + s'_c)}$
Set $r'_c \leftarrow r_1 + r_2$	←	Send (A'_c, s'_c, r_2)
Verify $e(A'_c, \hat{g}_1^{s'_c} y_E) = e(\hat{g}_1 P' \hat{h}_1^c \hat{h}_2^I \hat{h}_3^{r'_c}, \hat{g}_1)$		

Fig. 1. Construction for functionality $\mathcal{F}_{\mathrm{IOTAC}}^R$: interface issue.

the credential. This harms receiver privacy because it reveals to $\mathcal{E}$ whether the message that $\mathcal{R}$ wishes to obtain is associated with an access control policy that requires attributes certified by that issuer.

This problem can be solved by using a protocol that realizes the new functionality $\mathcal{F}_{\mathrm{IOTAC}}^R$. In our example, the relation R requires $\mathcal{R}$ to prove in zero-knowledge possession of credentials from $\mathcal{I}_1$ or $\mathcal{I}_2$. In the issuing phase of a protocol that realizes $\mathcal{F}_{\mathrm{IOTAC}}^R$, $\mathcal{R}$ proves in zero-knowledge possession of a credential from $\mathcal{I}_1$ or $\mathcal{I}_2$ to $\mathcal{E}$, and then $\mathcal{E}$ sends a new credential to $\mathcal{R}$ that signs the same attribute values plus the identity of $\mathcal{I}_1$ or $\mathcal{I}_2$. The access control policies only accept credentials signed by $\mathcal{E}$ and include $\mathcal{I}$'s identity as an attribute. Therefore, because $\mathcal{I}$'s identity is now an attribute, $\mathcal{I}$'s identity is not revealed to $\mathcal{E}$ when $\mathcal{R}$ proves possession of a credential in order to obtain a message.

As a concrete example, we show how to modify the OTAC protocol by Camenisch et al [2] so that it can handle multiple issuers. At setup, each of the issuers $\mathcal{I}$ run the protocol ISetup described in Figure 2 in [2] in order to compute a private key $sk_I = x_I$ and a public key $pk_I = (g_1, h_0, h_1, h_2, u, v, w, g_t, h_t, y_I)$ to compute and verify signatures. The sender $\mathcal{E}$ also runs the protocol in Figure 2 in [2] to compute sk_E and pk_E, but pk_E consists of an additional element $\hat{h}_3$. The database setup algorithm DBSetup in Figure 3 in [2] remains unchanged, except that, for each record of data R_i, the attribute c_{i1} in the access control list $\mathrm{ACL}_i = (c_{i1}, \ldots, c_{il})$ is the identity of an issuer. A receiver must get her attributes certified by this issuer in order to satisfy ACL_i.

The issuing phase is divided into two parts. First, a receiver runs multiple times with one or more issuers the issuing protocol Issue described in Figure 4 in [2]. As a result of each execution of the issuing protocol, the receiver obtains a signature $(A_c = (g_1 P h_1^c h_2^{r_c})^{1/(x_I + s_c)}, s_c, r_c)$ that signs her secret key z_U $(P = h_0^{z_u})$ and one of her attributes c. Second, the receiver gets her attributes signed by the sender. This step corresponds to the realization of the issue interface of our functionality $\mathcal{F}_{\mathrm{IOTAC}}^R$. We describe the protocol in Fig. 1. A receiver executes this protocol once for each of the signatures issued by the issuers. In a single execution, the receiver inputs a signature (A_c, s_c, r_c) and the identity I of the issuer, while the sender inputs his signing key pair (sk_E, pk_E). As result, the receiver obtains a signature $(A'_c = (g_1 P' \hat{h}_1^c \hat{h}_2^I \hat{h}_3^{r'_c})^{1/(x_I + s'_c)}, s'_c, r'_c)$ signed by the sender on her secret z_U, on her attribute c, and on the issuer identity I.

Finally, a receiver and the sender execute the transfer algorithm Transfer described in Figure 5 in [2]. In Figure 5 in [2], the receiver proves in zero-knowledge to the sender that she possesses one signature (A_c, s_c, r_c) issued by the issuer for each of the attributes in ACL_i. We require two modifications. First, instead of using signatures (A_c, s_c, r_c) signed by issuers, the receiver uses signatures (A'_c, s'_c, r'_c) signed by the sender. Second, because now ACL_i also contains the identity of the issuer, the receiver, in addition to proving that the attribute c signed in (A'_c, s'_c, r'_c) equals one of the attributes $(c_{i2}, \ldots, c_{il})$ in ACL_i, proves also that the issuer identity signed in (A'_c, s'_c, r'_c) equals c_{i1}.

5. Conclusion

We have shown that the security definition for Issuer-Free OTAC in [5,6] poses a problem. Namely, it does not allow the sender to attest whether a receiver possesses the claimed attributes. In the protocols proposed in [5,6], any malicious receiver can obtain any message from the sender, regardless of the access control policy associated with the message. Consequently, the protocols in [5,6] do not enforce any form of access control. To address this problem, we have proposed a new security definition for Issuer-Free OTAC. Our definition requires the receiver to prove in zero-knowledge to the sender that her attributes fulfill some predicates. Our definition is suitable for settings with multiple issuers because it allows the design of OTAC protocols where the receiver, when accessing a record, can hide the identity of the issuer that certified her attributes. In the table below, we compare the properties of our approach with OTAC in [2] and Issuer-Free OTAC in [5,6].

	Receiver Privacy	Sender Security	Multiple Issuers Support
OTAC [2]	✓	✓	⊥
Issuer-Free OTAC [5,6]	✓	⊥	⊥
Our approach	✓	✓	✓

References

[1] J. Camenisch, G. Neven, A. Shelat, Simulatable adaptive oblivious transfer, in: M. Naor (Ed.), EUROCRYPT, in: Lecture Notes in Computer Science, vol. 4515, Springer, 2007, pp. 573–590.

[2] J. Camenisch, M. Dubovitskaya, G. Neven, Oblivious transfer with access control, in: E. Al-Shaer, S. Jha, A.D. Keromytis (Eds.), ACM Conference on Computer and Communications Security, ACM, 2009, pp. 131–140.

[3] J. Camenisch, M. Dubovitskaya, G. Neven, G.M. Zaverucha, Oblivious transfer with hidden access control policies, in: D. Catalano, N. Fazio, R. Gennaro, A. Nicolosi (Eds.), Public Key Cryptography, in: Lecture Notes in Computer Science, vol. 6571, Springer, 2011, pp. 192–209.

[4] A. Rial, Blind attribute-based encryption and oblivious transfer with fine-grained access control, Des. Codes Cryptogr. 81 (2) (2016) 179–223.

[5] V. Guleria, R. Dutta, Issuer-free adaptive oblivious transfer with access policy, in: Information Security and Cryptology, ICISC 2014, Springer, 2014, pp. 402–418.

[6] V. Guleria, R. Dutta, Universally composable issuer-free adaptive oblivious transfer with access policy, Secur. Commun. Netw. 8 (18) (2015) 3615–3633.