On the Security of Attribute Based Signature Schemes

S Sharmila Deva Selvi, Subhashini Venugopalan, C. Pandu Rangan

Theoretical Computer Science Laboratory
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Abstract

Attribute based signatures allow users possessing a set of credentials to sign documents; although the credentials of the signer can be verified, signers can still continue to retain a reasonable degree of anonymity. In this work we discuss aspects regarding the security of some attribute based signature schemes. In particular, we show multiple breaks in the existing threshold attribute based signature schemes in [9]. We first claim that the scheme is not secure, since it allows, a signer possessing keys for some attributes to perform universal forgery and produce a signature that can satisfy the threshold for a set of attributes she does not possess. We then show a total break in the system, where the attacker can derive a part of the secret key and act as the key generating authority to generate private keys for other users. We also include examples of the attacks to highlight the flaws of this scheme, and other schemes [10, 15, 8, 11] which have the same or a similar key construct.

Keywords

attribute-based, signature, ABS, forgery, security, threshold

1 Introduction

There are several settings where one may need to obtain some permissions or a signature from an authority having certain credentials, but it may be immaterial as to who the specific authorizer is. For example, Alice may need an approval from a major in the army or a captain in the navy but it may not matter as to who exactly gave her the permissions. There are also other situations where one may need to prove that they possess specific credentials but would not want to reveal all their personal details in doing so. Let's take the example of Bob, a captain in the navy, who is going to a secret officers meet. He wouldn't want to reveal more about himself except to prove that he is an officer who is within his rights in attending the meet. These are cases where attribute based signatures can be useful.

Attribute based signatures(ABS) allows the creation of a signature that can satisfy a predicate, using keys given by attribute authorities. Attribute based encryption, which was introduced by Sahai and Waters [14] and developed by several others [5, 1, 16] have been a major inspiration for ABS schemes. Attribute based signatures, by their very nature, provide a reasonable degree of anonymity to the signer since a signer can be identified only by the attributes used in the signature and not by any other specific user identity. This property of providing anonymity to the signer gives ABS a flavor that can be found in signature schemes such as, ring signatures [13], group signatures [3], and mesh signatures [2]. But the primary motivation for ABS schemes is to produce a signature that ensures that the signer can satisfy a predicate of attributes.

Properties. Two important properties of attribute based signatures:

1. They are capable of addressing complex predicates or access control policies.
2. They must satisfy *collusion resistance*. *Collusion resistance* means that, even if multiple users collude, they cannot produce a signature to satisfy a new predicate which none of them were able to satisfy individually.

1.1 Related work

ABS in the form of attribute based group signatures [7, 6] had been introduced by Khader in 2007. However, the first formalization of attribute based signatures was given by [12] Maji *et al.* in 2008. Following this, attribute based ring signatures was introduces by Li and Kim in [10]. In the recent past, there have been several attribute based signature schemes focusing on interesting aspects like threshold [15, 9], and support for multiple attribute authorities [9]. However, in our study of these schemes, we have observed that a number of them are flawed and are susceptible to forgery and total break attacks.

1.2 Overview of our results

Breaks. In this paper we show breaks in the threshold attribute based signature scheme proposed by Li *et al.* in [9]. We claim that the scheme is insecure since,

- a signer who does not possess the necessary number of attributes to satisfy the threshold of the predicate can still produce a valid signature. (threshold forgery)
- a signer with some attributes, completely unrelated to the attributes of the predicate can forge a signature. (universal forgery)
- any attacker can obtain a component of the secret key with which she can pose as the key generating authority. (total break)

Since our attacks are based on the key construct in the scheme, we also show that these breaks hold for a) the ABS scheme with multiple attribute authorities by the authors in the same paper [9]; b) attribute based ring signature scheme in [10]; c) the threshold attribute based signature by Shahandashti and Safavi-Naini [15] which is derived based on [10]; d) ABS for multi-level threshold circuits [8]; and e) Hidden attribute-based signatures without anonymity revocation [11].

Organization of the paper. We begin with the preliminaries in the next section. In Section 3 we give a brief intuition on the attacks, then present the ABS scheme in [9] and give a detailed description of the 3 attacks on the scheme using an example. In the subsequent section we present the attacks on the various related schemes having similar key construction. We conclude with a summary of all the attacks and our observations.

2 Preliminaries

In this section we present some of the preliminaries required for attribute based signatures and threshold ABS.

2.1 Bilinear Pairing

Let $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ be multiplicative groups of prime order p. The elements $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. A bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow G_T$ with the following properties:

1. **Bilinear**: $e({g_1}^a, {g_2}^b) = e(g_1, g_2)^{ab}$ for all $g_1 \in \mathbb{G}_1$,$g_2 \in \mathbb{G}_2$, where $a, b \in \mathbb{Z}_p$.
2. **Non-degenerate**: There exists $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ such that $e(g_1, g_2) \neq 1$; in other words, the map does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_2$ to the identity in $\mathbb{G}_T$.
3. **Computability**: There is an efficient algorithm to compute $e(g_1, g_2)$ for all $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$.

2.2 Lagrange Interpolation

Let $q(x)$ be a $d-1$ degree polynomial with each co-efficients in $\mathbb{Z}_p$. Then, given any set of d points on the polynomial $\{q(i) : i \in S\}$, where S is a set of indices such that $|S| = d$, we can use Lagrange's interpolation to find $q(j)$ for any $j \in \mathbb{Z}_p$ as follows: ($\Delta_{i,S}(j)$ is termed the Lagrange coefficient)

$$q(j) = \sum_{i \in S} q(i)\Delta_{i,S}(j) \text{ , where } \Delta_{i,S}(j) = \prod_{j' \in S, j' \neq i} \frac{j - j'}{i - j'}$$

2.3 Attribute-Based Signature

Attribute based signature schemes consist of four algorithms: setup, key generation, signing and verification algorithms, defined as follows:

Setup is run by a central authority. It takes as input the security parameter (d), and gives as output the set of public parameters denoted by $params$ and a master secret key, msk.

Key-Gen (or **Extract**) is run by the key generating authority. It takes as input, msk and a set of attributes from the signer to produce $\mathcal{D}$, a corresponding set of private keys for the signer.

Sign algorithm takes the signer's private keys $\mathcal{D}$, a predicate Υ, and a message m to produce a signature σ.

Verify algorithm ensures that, a signature σ on a message m is pronounced as valid if it has been signed by a signer having attributes that satisfy the predicate Υ.

2.4 Threshold Attribute-Based Signature

Threshold ABS is an attribute based signature where the predicate Υ is given by a (t, n)-threshold. This says that a signer with atleast t out of the n attributes will be able to generate a valid signature. The formal definition is as follows.

Definition: Threshold attribute-based signature schemes consists of following four algorithms:

Setup is run by a central authority. It takes as input the security parameter (d), and gives as output the set of public parameters denoted by $params$ and a master secret key, msk.

Key-Gen (or **Extract**) is run by the key generating authority. It takes as input msk and a set of signer attributes (U_β) to produce a corresponding set of private keys, $\mathcal{D}$, for the signer.

Sign algorithm takes the signer's private keys $\mathcal{D}$, public parameters $params$, a fixed threshold t (integer), an attribute subset U^*, and a message m to produce a signature σ.

Verify algorithm is run by a verifier. It outputs 1 when a signature σ on a message m is signed by a valid signer who holds atleast t out of the n attributes in U^* i.e. $|U_\beta \cap U^*| \geq t$.

Correctness: A threshold-ABS scheme must satisfy the correctness property, i.e. a signature generated by a signer with attribute set U_β must pass the verification test for any U^* if $|U_\beta \cap U^*| \geq t$.

Unforgeablility: It is required for the above threshold-ABS scheme to be existentially unforgeabile under chosen message and attribute attacks as follows:

Setup phase: The challenger $\mathcal{C}$ runs the Setup algorithm and gives the common public parameters $params$ to adversary $\mathcal{A}$.

Query phase: Adversary can perform polynomially bounded number of queries in an adaptive manner (interactively) as described below.

- KeyGen. $\mathcal{A}$ is allowed to query the keys for any set of attributes U_β.
- Sign. $\mathcal{A}$ requests for the signature of a signer with (any) attribute set U_β on any message m by specifying a threshold t' on an $n-$element attribute set U^*.

Forgery phase: At the end of the game, $\mathcal{A}$ outputs a forgery σ with respect to a set of attributes U^*, threshold t'' and message m'.

$\mathcal{A}$ wins the game if:

1. σ is a valid signature with respect to U^*, t'', m'.
2. for all *KeyGen* queries on attribute set U_β, we have $|U_\beta \cap U^*| < t''$.
3. for all *Sign* queries on message m using signer attributes U_β, $m \neq m'$ or $|U_\beta \cap U^*| < t''$.

If no polynomial adversary has a considerable advantage in the above game, we say that the threshold-ABS scheme is existentially unforgeable against chosen message and attribute set attacks.

Collusion-resistance. Note here that the above game ensures the property *collusion resistance* and guarantees that no colluding group of users can create a signature that could not have been created by any one of the colluders independently. This is because the adversary could have queried for the secret keys of all elements in U^* from *KeyGen* with different attribute sets U_β as input, such that $U^* \subseteq \bigcup_\beta \{U_\beta\}$.

3 Attacks

In this section we will present the breaks on some of the existing attribute-based signature schemes. We will first take a detailed look at the construction of the flexible threshold ABS scheme in [9]. Then, with the help of an example, we will look into detail the algebra involved for partially unblinding values from the private key. This will give the adversary some intermediate values which can be used to create forgeries. We'll finally demonstrate the attacks of threshold forgery, universal forgery and total break on the given scheme. We start with a brief *insight* on the attacks:

- All the attacked schemes use secret sharing to generate the private keys.
- Users of the system have more than sufficient shares -via dummy attributes- to reconstruct the secret, but these are blinded with values purportedly to enforce reconstruction only when the protocol is strictly followed.
- Through algebraic manipulations, we show how the shares can be unblinded by a malicious user and how forgeries can be produced.

3.1 Construction of threshold ABS in [9]

Setup$(d)\rightarrow$(params, msk).
First, define the attributes in universe U as elements in $\mathbb{Z}_p$. A $d-1$ default attribute set from $\mathbb{Z}_p$ is given as $\Omega = \{\Omega_1, \Omega_2, \cdots, \Omega_{d-1}\}$. Select a random generator $g \in \mathbb{G}_1$, a random $x \in \mathbb{Z}_p$, and set $g_1 = g^x$. Next, pick a random element $g_2 \in \mathbb{G}_1$ and compute $\mathcal{Z} = e(g_1, g_2)$. Two hash functions are also chosen such that $H_2, H_4 : \{0,1\}^* \rightarrow \mathbb{G}_1$. The public parameters is *params* and the master secret key is *msk*.

$$params = (g, g_1, g_2, \mathcal{Z}, d, H_2, H_4) \qquad\qquad msk = x.$$

Extract. To generate a private key for an attribute set ω, the following steps are taken:

- First, choose a $d-1$ degree polynomial $q(y)$ randomly such that $q(0) = x$;
- Generate a new attribute set $\hat{\omega} = \omega \cup \Omega$. For each $i \in \hat{\omega}$, choose $r_i \in_R \mathbb{Z}_p$ and compute $d_{i0} = g_2^{q(i)} \cdot H_2(i)^{r_i}$ and $d_{i1} = g^{r_i}$;
- Finally, output $\mathcal{D}_i = (d_{i0}, d_{i1})$ as the private key for each $i \in \hat{\omega}$

Sign. Suppose one has a private key for the attribute set ω. To sign a message m with predicate $\Upsilon_{k,\omega^*}(\cdot)$, namely, to prove owning at least k attributes among an n-element attribute set ω^*, she selects a k-element subset $\omega' \subseteq \omega \cap \omega^*$ and proceeds as follows:

- First, she selects a default attribute subset $\Omega' \subseteq \Omega$ with $|\Omega'| = d-k$ and chooses $n+d-k$ random values $r'_i \in \mathbb{Z}_p$ for $i \in \omega^* \cup \Omega'$.
- She computes,

$$\sigma_0 = [\Pi_{i\in\omega'\cup\Omega'} d_{i0}^{\Delta_{i,S}(0)}][\Pi_{i\in\omega^*\cup\Omega'} H_2(i)^{r'_i}] H_4(m)^s$$

$$\sigma_i = \begin{cases} d_{i1}^{\Delta_{i,S}(0)} \cdot g^{r'_i} & \text{for } i \in \omega' \cup \Omega' \\ g^{r'_i} & \text{for } i \in \omega^* \backslash \omega' \end{cases}$$

$$\sigma'_0 = g^s, \text{ with a randomly chosen value } s \in \mathbb{Z}_p$$

- Signature: $\sigma = (\sigma_0 \ , \ \{\sigma_i\}_{i\in\omega^*\cup\Omega'} \ , \ \sigma'_0)$

Verify. To verify the signature $\sigma = (\sigma_0 \ , \ \{\sigma_i\}_{i\in\omega^*\cup\Omega'} \ , \ \sigma'_0)$ of message m with threshold k for attributes $\omega^* \cup \Omega'$, check if the following equation holds:

$$\frac{e(g, \sigma_0)}{[\Pi_{i\in\omega^*\cup\Omega'} e(H_2(i), \sigma_i)] e(H_4(m), \sigma'_0)} \stackrel{?}{=} \mathcal{Z}$$

3.2 Attack instantiation

In this section we present the attacks on the above scheme. We'll take the following example case into consideration for the attack:

- Let $d = 2$ (for simplicity)
- Accordingly $\Omega = \Omega_1$, ($|\Omega| = d - 1$)
- Users set of attributes $\omega = \{A, B, C, D\}$
- Let $\omega^* = \{A, P\}$ (the signing attributes)
- Let the threshold $k = 2$ i.e the user needs to have both attributes in ω^* to generate a valid signature.
- For simplicity we will denote $H_2(i)$ by H_i.

3.3 Setup and Extract

At the end of Setup(d), we have $params = (g, g_1, g_2, \mathcal{Z}, d, H_2, H_4)$ and $msk = x$.

1. Since $d = 2$ we consider a one-degree polynomial $q(y)$ with $q(0) = x$. Let $q(y) = 7y + x$.
2. $\hat{\omega} = \omega \cup \Omega = \{A, B, C, D, \Omega_1\}$.

So we have the following private key values:

$$D_A = \left\{ d_{A0} = g_2^{q(A)} \cdot H_A^{r_A} \ , \ d_{A1} = g^{r_A} \right\} \qquad D_B = \left\{ d_{B0} = g_2^{q(B)} \cdot H_A^{r_B} \ , \ d_{B1} = g^{r_B} \right\}$$
$$D_C = \left\{ d_{C0} = g_2^{q(C)} \cdot H_A^{r_C} \ , \ d_{C1} = g^{r_C} \right\} \qquad D_D = \left\{ d_{D0} = g_2^{q(D)} \cdot H_A^{r_D} \ , \ d_{D1} = g^{r_D} \right\}$$
$$D_{\Omega_1} = \left\{ d_{\Omega_1 0} = g_2^{q(\Omega_1)} \cdot H_{\Omega_1}^{r_{\Omega_1}} \ , \ d_{\Omega_1 1} = g^{r_{\Omega_1}} \right\}$$

3.4 Preliminaries for the attack

Attacker picks d private keys at a time and does the following computations:

Notations. For a set $S = \{A, B\}$ when we evaluate the Lagrange co-efficient $\Delta_{i,S}(0)$ at $i = A$, we will denote the value as Δ_{AB}. If $i = B$, then we denote it as Δ_{BA}. Please note that Δ_{AB} need not equal Δ_{BA}. Also note that, all the Lagrange coefficients ($\Delta_{i,S}(0)$) are publicly computable.

Computations done by signer. We will evaluate $\Pi_{i \in S} d_{i,0}^{\Delta_{i,S}(0)}$ for pairs of attributes. S takes values from the set $\{\{A, B\}, \{A, C\}, \{A, D\}, \{B, C\}, \{B, D\}, \{C, D\}\}$. This gives us,

$$\left.\begin{aligned} X_1 &= g_2^x H_A^{r_A \Delta_{AB}} H_B^{r_B \Delta_{BA}} & X_2 &= g_2^x H_A^{r_A \Delta_{AC}} H_C^{r_C \Delta_{CA}} \\ X_3 &= g_2^x H_A^{r_A \Delta_{AD}} H_D^{r_D \Delta_{DA}} & X_4 &= g_2^x H_B^{r_B \Delta_{BC}} H_C^{r_C \Delta_{CB}} \\ X_5 &= g_2^x H_B^{r_B \Delta_{BD}} H_D^{r_D \Delta_{DB}} & X_6 &= g_2^x H_C^{r_C \Delta_{CD}} H_D^{r_D \Delta_{DC}} \end{aligned}\right\} \quad \text{(Primary equations)} \quad (1)$$

We will use the following notations for simplicity:

$$\Delta_{A1} = \Delta_{AB} - \Delta_{AC} \qquad \Delta_{B1} = \Delta_{BA} - \Delta_{BC}$$
$$\Delta_{A2} = \Delta_{AB} - \Delta_{AD} \qquad \Delta_{B2} = \Delta_{BA} - \Delta_{BD}$$

Now, the user computes the following:

$$Y_1 = \frac{X_1}{X_2} = H_A^{r_A \Delta_{A1}} H_B^{r_B \Delta_{BA}} H_C^{-r_C \Delta_{CA}}$$
$$Y_2 = \frac{X_1}{X_3} = H_A^{r_A \Delta_{A2}} H_B^{r_B \Delta_{BA}} H_D^{-r_D \Delta_{DA}}$$
$$Y_3 = \frac{X_1}{X_4} = H_A^{r_A \Delta_{AB}} H_B^{r_B \Delta_{B1}} H_C^{-r_C \Delta_{CB}}$$
$$Y_4 = \frac{X_1}{X_5} = H_A^{r_A \Delta_{AB}} H_B^{r_B \Delta_{B2}} H_D^{-r_D \Delta_{DB}}$$

We extend our notations to include:

$$\Delta_{A3} = \Delta_{A1} - \Delta_{A2} = \Delta_{AD} - \Delta_{AC} \qquad \Delta_{B3} = \Delta_{BA}\Delta_{CB} - \Delta_{B1}\Delta_{CA}$$
$$\Delta_{A4} = \Delta_{A1}\Delta_{CB} - \Delta_{AB}\Delta_{CA} \qquad \Delta_{B4} = \Delta_{BA}\Delta_{DB} - \Delta_{B2}\Delta_{DA}$$
$$\Delta_{A5} = \Delta_{A2}\Delta_{DB} - \Delta_{AB}\Delta_{DA} \qquad \Delta_{B5} = \Delta_{B3}\Delta_{A5} - \Delta_{B4}\Delta_{A4}$$

Then, the user does the following computations:

$$Z_1 = \frac{Y_1}{Y_2} = H_A^{r_A \Delta_{A3}} H_D^{r_D \Delta_{DA}} H_C^{-r_C \Delta_{CA}}$$

$$Z_2 = \frac{Y_1^{\Delta_{CB}}}{Y_3^{\Delta_{CA}}} = H_A^{r_A \Delta_{A4}} H_B^{r_B \Delta_{B3}}$$

$$Z_3 = \frac{Y_2^{\Delta_{DB}}}{Y_4^{\Delta_{DA}}} = H_A^{r_A \Delta_{A5}} H_B^{r_B \Delta_{B4}} \tag{2}$$

$$Z_4 = \frac{Z_2^{\Delta_{A5}}}{Z_3^{\Delta_{A4}}} = H_B^{r_B \Delta_{B5}} \tag{3}$$

$$Z_5 = Z_3^{\frac{1}{\Delta_{B4}}} = H_A^{\Delta_{A5}/\Delta_{B4}} H_B^{r_B} = H_A^{r_A \hat{\Delta}} \cdot H_B^{r_B} \tag{4}$$

With these computations[1] the signer is now ready to forge.

3.5 Attack 1: Forgery without satisfying threshold

The signer (who possesses attributes in ω) is now, going to sign a document claiming she has attribute *2-out-of-*$\{A,P\}$. Note, here that the signer does not have the attribute P and has not received any private key pertaining to the attribute P.

- $\omega = \{A, B, C, D\}$
- $\Omega = \{\Omega_1\}$
- $d = 2$ So, $(d-1) = 1$
- $\omega^* = \{A, P\}$

3.5.1 Sign (Forge)

The attacker computes the following

$$\sigma_{0I} = [\{g_2^{q(A)} H_A^{r_A}\}^{\Delta_{AB}} \cdot \{g_2^{q(B)} H_B^{r_B}\}^{\Delta_{BA}}][H_A^{r'_A} H_P^{r'_P}] H_4(m)^s \tag{5}$$

Note that, r'_A, r'_P and s are values that the signer picks during signing. Now, the signer raises (4) to the exponent $(-\Delta_{BA})$ and multiplies with σ_{0I}.

$$\sigma_0 = Z_5^{(-\Delta_{BA})} \times \sigma_{0I} = g_2^x H_A^{r_A C} [H_A^{r'_A} H_P^{r'_P}] H_4(m)^s \tag{6}$$

Here C is a constant, which can be computed by the signer. Now for the σ_i values:

$$\sigma_A = (g^{r_A})^C . g^{r'_A} \ , \ \sigma_P = g^{r'_P}$$

Finally, $\sigma'_0 = g^s$. The signature is $\sigma = \{\sigma_0 \ , \ \{\sigma_i\}_{i \in \{A,P\}} \ , \ \sigma'_0\}$

3.5.2 Verification correctness

$$\begin{aligned}
\frac{e(g,\sigma_0)}{[\Pi_{i \in \omega^* \cup \Omega'} e(H_2(i), \sigma_i)] e(H_4(m), \sigma'_0)} &= \frac{e(g, g_2^x) e(g, H_A^{r_A C}) e(g, H_A^{r'_A}) e(g, H_P^{r'_P}) e(g, H_4(m)^s)}{[e(H_A, \sigma_A) e(H_P, \sigma_P)] e(H_4(m), g^s)} \\
&= \frac{e(g, g_2^x) e\left(g, H_A^{r_A C}\right) e\left(g, H_A^{r'_A}\right) e\left(g, H_P^{r'_P}\right)}{[e\left(H_A, (g^{r_A})^C g^{r'_A}\right) e\left(H_P, g^{r'_P}\right)]} \\
&= e(g, g_2^x) \\
&= \mathcal{Z}
\end{aligned}$$

3.6 Attack 2: Universal forgery

We now show universal forgery of the proposed signature scheme by extracting the secret component hidden in the signer's keys.

[1] In this example, on careful computation, one can observe that Δ_{A4} is zero and Equation 3 can be obtained directly from Z_2 since Δ_{B3} is a non-zero value computable by the signer. Note that the remaining computation is unaffected by this and is shown as is for simplicity.

Computations for launching the attack. We will continue with the preliminary computations that we have already done. We also know that the Lagrange coefficients for any set of attributes can be found. Thus, the Δ_{ij} values can be computed and therefore we can proceed to find:

$$\delta_B = Z_4^{\Delta_{B5}{}^{-1}} = H_B^{r_B} \tag{7}$$

$$\delta_A = \left(\frac{Z_3}{(\delta_B)^{\Delta_{B4}}}\right)^{\Delta_{A5}{}^{-1}} = H_A^{r_A} \tag{8}$$

Using the components δ_A and δ_B, we can now do the following:

$$\rho = \frac{X_1}{\delta_A{}^{\Delta_{AB}}\delta_B{}^{\Delta_{BA}}} = g_2^x \tag{9}$$

Now, with the value of g_2^x the signer will be in a position to forge for any threshold of attributes and provide the appropriate σ_i components to pass the verification tests.

3.6.1 Sign (Universal forgery)

The attacker (who possesses attributes in ω) is now, capable of signing a document claiming she has attribute *2-out-of-*$\{L,M\}$. Note here that the signer does not have either of the 2 attributes L or M and has not received any private key pertaining to these attributes.

- $\omega = \{A, B, C, D\}$
- $\Omega = \{\Omega_1\}$
- $d = 2$ So, $(d-1) = 1$
- $\omega^* = \{L, M\}$

The forger generates the following signature and gives $\sigma = \{\sigma_0\ ,\ \{\sigma_i\}_{i\in\{L,M\}}\ ,\ \sigma'_0\}$.

$$\begin{aligned}\sigma_0 &= g_2^x[H_L^{r'_L}H_M^{r'_M}]H_4(m)^s\\ \sigma_i\ &:\ \sigma_L = g^{r'_L}, \quad \sigma_M = g^{r'_M}\\ \sigma'_0 &= g^s\end{aligned}$$

3.6.2 Verify (Correctness)

$$\begin{aligned}\frac{e(g,\sigma_0)}{[\Pi_{i\in\omega^*\cup\Omega'}e(H_2(i),\sigma_i)]\,e(H_4(m),\sigma'_0)} &= \frac{e(g,g_2^x)e\left(g,H_L^{r'_L}\right)e\left(g,H_M^{r'_M}\right)e\left(g,H_4(m)^s\right)}{[e\left(H_L,\sigma_L\right)e\left(H_M,\sigma_M\right)]\,e\left(H_4(m),g^s\right)}\\ &= \frac{e(g,g_2^x)e\left(g,H_L^{r'_L}\right)e\left(g,H_M^{r'_M}\right)}{[e\left(H_L,g^{r'_L}\right)e\left(H_M,g^{r'_M}\right)]}\\ &= e(g,g_2^x)\\ &= \mathcal{Z}\end{aligned}$$

3.7 Attack 3: Total break - impersonating key issuing authority

The attacker now holds g_2^x. With this, we show how she can generate the private keys for anyone.

To generate a private key for an attribute set ω, the attacker does the following:

- First, choose a $d-1$ degree polynomial $q(y)$. Say $q(y) = 10y + x$
- Generate a new attribute set $\hat{\omega} = \omega \cup \Omega$. For each $i \in \hat{\omega}$, choose $r_i \in_R \mathbb{Z}_p$ and compute d_{i0} and d_{i1}.

$$d_{i0} = g_2^{q(i)} \cdot H_2(i)^{r_i} = g_2^{10i} \cdot g_2^x \cdot H_2(i)^{r_i} \text{ and } d_{i1} = g^{r_i}$$

- Finally, output $\mathcal{D}_i = (d_{i0}, d_{i1})$ as the private key for each $i \in \hat{\omega}$

This shows a total break on the system.

4 Attacks on schemes with similar key construct

In this section, we show that these attacks also hold in the case of other schemes which have the same or similay key construction as in [9]. We'll first look into the attribute based ring signature scheme of Kim and Li in [10]. We will then show the attack on the (k, n) threshold scheme by Shahandashti and Safavi-Naini in [15], then on the ABS with multiple attribute authorities [9]. We then proceed to show that the attacks are also applicable on the ABS scheme for bounded multi-level threshold circuits [8] and also on the scheme for hidden ABS without anonymity revocation presented in [11].

4.1 Total break on attribute based ring signature scheme

Here we would like to remark that [9] has been derived from [10] and that both their key extract phases are exactly the same. We first present the key extraction phase of [10].

4.1.1 Key Extract

To generate a private key for an attribute set ω, the following steps are taken:

- $\rightarrow$ First, choose a $d-1$ degree polynomial $q(y)$ randomly such that $q(0) = x$;
- $\rightarrow$ Generate a new attribute set $\hat{\omega} = \omega \cup \Omega$. For each $i \in \hat{\omega}$, choose $r_i \in_R \mathbb{Z}_p$.
- $\rightarrow$ Compute, $d_{i0} = g_2^{q(i)} \cdot (H_2(i))^{r_i}$ and $d_{i1} = g^{r_i}$;
- $\rightarrow$ Finally, output $\mathcal{D}_i = (d_{i0}, d_{i1})$ as the private key for each $i \in \hat{\omega}$

4.1.2 Total break

It's now easy to see that the d_{i0} components in [10] are exactly the same as before. This allows us derive the set of equations (1) which we called as the Primary Equations. From there we can proceed in the exact same manner to get equations (2), then (3), (7), (8) and finally (9). This will allow the attacker to get g_2^x. Now, the attacker can use the strategy in Section 3.7 and impersonate the key generating authority.

4.2 Break on threshold attribute based signature scheme

The threshold attribute based signature scheme by Shahandashti and Safavi-Naini in [15] has been extended from the previous ring signature scheme [10] to support (k, n) threshold. We will take a closer look at their setup and key extraction phases before we identify the similarities and describe how the attack is applicable.

4.2.1 Construction

Setup(1^k): Pick y randomly from $\mathbb{Z}_p$ and set $g_1 = g^y$. Pick random elements $g_2, h, t_1, t_2, \cdots, t_{n+1}$ from $\mathbb{G}_1$. Define and output the following: $T(x) \triangleq g_2^{x^n} \prod_{i=1}^{n+1} t_i^{\Delta_{i,N}(x)}$ and $msk = y$ and $mpk = (g, g_1, g_2, \{t_i\}, h)$.

KeyGen(msk, A): To generate keys for the user attribute set A. Choose a random $d-1$ degree polynomial $q(x)$ such that $q(0) = y$, choose random elements $r_i \in \mathbb{Z}_p$ for $i \in A$, and output:

$$ssk = \left\{ g_2^{q(i)} T(i)^{r_i} \ , \ g^{r_i} \right\}_{i \in A}$$

4.2.2 Attack

We will first establish the congruence of the key generation in this scheme to that of [9]. Here, we can note that $T(i)$ is a publicly computable function, the definition of which has been established in the setup. Thus, we can consider it to be equivalent to the hash function $H_2(i)$ as used in schemes [9, 10] for the purposes of this attack. If we now use the notation T_i to indicate the value of $T(i)$ where i is the attribute under consideration and then evaluate $\Pi_{i \in S} ssk_{i0}^{\Delta_{i,S}(0)}$ for pairs of attributes, $S \in \{\{A, B\}, \{A, C\}, \{A, D\}, \{B, C\}, \{B, D\}, \{C, D\}\}$ to get:

$$X_1 = g_2^y T_A^{r_A \Delta_{AB}} T_B^{r_B \Delta_{BA}} \qquad X_2 = g_2^y T_A^{r_A \Delta_{AC}} T_C^{r_C \Delta_{CA}}$$
$$X_3 = g_2^y T_A^{r_A \Delta_{AD}} T_D^{r_D \Delta_{DA}} \qquad X_4 = g_2^y T_B^{r_B \Delta_{BC}} T_C^{r_C \Delta_{CB}}$$
$$X_5 = g_2^y T_B^{r_B \Delta_{BD}} T_D^{r_D \Delta_{DB}} \qquad X_6 = g_2^y T_C^{r_C \Delta_{CD}} T_D^{r_D \Delta_{DC}}$$

If we now use the notation H_i to indicate the value of $T(i)$ and also replace the msk which is y here, with x. Then, we can get the set primary equations as in (1). From here on, the rest of the steps in extracting g_2^y is exactly the same as before. Thus, with the help of g_2^y, the attacker is capable of producing keys for any set of attributes.

Generating Keys. To generate keys for the user attribute set A. The attacker chooses a random $d-1$ degree polynomial $q(x)$, such that $q(0) = y$, say $q(x) = 10x + y$. And chooses random elements $r_i \in \mathbb{Z}_p$ for $i \in A$, and outputs:

$$ssk = \left\{ g_2^{10i} \cdot g_2^y \cdot T(i)^{r_i} \ , \ g^{r_i} \right\}_{i \in A}$$

4.3 Attack on ABS with multiple attribute authorities

ABS scheme with multiple attribute authorities is the second ABS scheme proposed in [9]. Here, we assume there are k attribute authorities in addition to one central authority. This scheme has some variations in its construction since it needs to be able to address multiple authorities. We will review the phases of the construction of their scheme before we show how an attacker can break the system.

4.3.1 Construction

Setup. Assume that there are k distributed attribute authorities. Each of them is in charge of the issue of attribute set A_i for $1 \leq i \leq k$. Define a default attribute set Ω_i of d elements for each attribute authority. The central authority chooses $s_1, \cdots, s_k$ for all attribute authorities and hash functions $H_2, H_4, \cdots, H_k, H : \{0,1\}^* \to \mathbb{G}_1$. In addition, the central authority chooses a random generator $g \in \mathbb{G}_1$, a random $x \in \mathbb{Z}_p^*$, and sets $g_1 = g^x$. Next, she picks a random element $g_2 \in \mathbb{G}_1$ and computes $\mathcal{Z} = e(g_1, g_2)$. For the attribute authority i, the secret key is s_i , which is assigned by the central authority.

Extract. First, a user with identity u gets a secret key from the central authority as $d_{ca} = g_2^{x - \Sigma_{i=1}^k f_{s_i}(u)}$. Then, she can request attribute private key from the i-th attribute authority as follows: Assume the user u is eligible to get an attribute set $A_{i,u}$ from the attribute authority i. The attribute authority i chooses a random $d-1$ degree polynomial $q_i(\cdot)$ such that $q_i(0) = f_{s_i}(u)$ and computes the private key $q(j)$ for user u as $\left\{ d_{ij0} = g_2^{q_i(j)} H_i(j)^{r_{ij}} \ , \ d_{ij1} = g^{r_{ij}} \right\}_{j \in A_{i,u} \cup \Omega_i}$.

Sign. Suppose one has a private key for attribute set $A_{i,u}$ for $1 \leq i \leq k$. To sign a message with predicate Υ that for each i, at least k_i out of n_i attributes ω_i^* are issued from the attribute authority i (Note k_i could be equal to 0). The user selects a k_i-value attribute subset $\omega_i' \subseteq A_{i,u} \cap \omega_i^*$. The following steps are taken:

- First, the user chooses $r_{i1}, r_{i2}, \cdots, r_{i,n_i+d-k_i} \in \mathbb{Z}_p^*$ and selects a $d - k_i$ default attribute subset $\Omega_i' \subseteq \Omega_i$. Define $S_i = \omega_i' \cup \Omega_i'$;
- She also computes $(\sigma_0, \{\sigma_{ij}\}_{j \in \omega_i^* \cup \Omega_i'}, \sigma_0')$, where:

$$\sigma_0 = \prod_{1 \leq i \leq k} \left(\prod_{j \in S_i} d_{ij0}^{\Delta_{j,S_i}(0)} \prod_{j \in \omega_i^* \setminus \omega_i'} H_i(j)^{r_{ij}'} \right) (H(m))^s$$

$$\left\{ \sigma_{ij} = d_{ij1}^{\Delta_{i,S}(0)} g^{r_{ij}'} \right\}_{j \in S_i} \ , \ \left\{ \sigma_{ij} = g^{r_{ij}'} \right\}_{j \in \omega_i^* \setminus \omega_i'} \ , \ \sigma_0' = g^s$$

Verify. On input the signature $(\sigma_0, \{\sigma_{ij}\}_{j \in \omega_i^* \cup \Omega_i'}, \sigma_0')$ with predicate Υ, compute

$$e(g, \sigma_0) \stackrel{?}{=} \mathcal{Z} \cdot \prod_{1 \leq i \leq k} \prod_{j \in \omega_i^* \cup \Omega_i'} e(H_i(j), \sigma_{ij}) e(H(m), \sigma_0')$$

If the equation holds, the signature is valid and the algorithm outputs accept. Otherwise, the algorithm outputs reject.

4.3.2 Attack

In order to attack this system, a forger has to try and get part of the secret component of every authority. The attacker can accomplish this as follows:

- Using the set of attributes $\{A_{i,u} \cup \Omega_i\}$ the user has from each authority i, the attacker forms equations as in (1); the only difference being, g_2^x will now be $g_2^{f_{s_i}(u)}$.
- Following along the same way as before, the user can now extract $g_2^{f_{s_i}(u)}$.
- Once the attacker extracts $g_2^{f_{s_i}(u)}$ for all $1 \leq i \leq k$. She computes their product:

$$\prod_{i=1}^{k} g_2^{f_{s_i}(u)} = g_2^{\sum_{i=1}^{k} f_{s_i}(u)}$$

- Multiplying it with the secret component given by the central authority :

$$d_{ca} \times g_2^{\sum_{i=1}^{k} f_{s_i}(u)} = g_2^{x - \Sigma_{i=1}^{k} f_{s_i}(u)} \times g_2^{\sum_{i=1}^{k} f_{s_i}(u)} = g_2^x$$

Universal Forgery. Now, using the values g_2^x and $\left\{g_2^{f_{s_i}(u)}\right\}_{i=1,\cdots,k}$, the attacker can forge on any message for any threshold of attributes. The signature is computed as $(\sigma_0, \{\sigma_{ij}\}_{j \in \omega_i^* \cup \Omega_i'}, \sigma_0')$ where:

$$\sigma_0 = d_{ca} \cdot \prod_{1 \leq i \leq k} \left(g_2^{f_{s_i}(u)} \prod_{j \in \omega_i^*} H_i(j)^{r'_{ij}} \right) (H(m))^s$$

$$\left\{\sigma_{ij} = g^{r'_{ij}}\right\}_{j \in \omega_i^*} \ , \ \sigma_0' = g^s$$

Verification.

$$e(g, \sigma_0) = \mathcal{Z} \cdot \left(\prod_{1 \leq i \leq k} \prod_{j \in \omega_i^* \cup \Omega_i'} e(H_i(j), \sigma_{ij}) \right) e(H(m), \sigma_0')$$

Correctness:

$$\begin{aligned}
e(g \ , \ \sigma_0) &= e\left(g \ , \ d_{ca} \cdot \prod_{1 \leq i \leq k} \left(g_2^{f_{s_i}(u)} \prod_{j \in \omega_i^*} H_i(j)^{r'_{ij}} \right) (H(m))^s \right) \\
&= e\left(g \ , \ g_2^{x - \Sigma_{i=1}^{k} f_{s_i}(u)} \cdot g_2^{\sum_{i=1}^{k} f_{s_i}(u)} \prod_{1 \leq i \leq k} \prod_{j \in \omega_i^*} H_i(j)^{r'_{ij}} \right) e(g, H(m))^s \\
&= e(g, g_2^x) \cdot \prod_{1 \leq i \leq k} \prod_{j \in \omega_i^*} e(g \ , \ H_i(j))^{r'_{ij}}) \ e(g, H(m))^s \\
&= \mathcal{Z} \cdot \prod_{1 \leq i \leq k} \prod_{j \in \omega_i^*} e(g^{r'_{ij}} \ , \ H_i(j))) \ e(H(m), \sigma_0') \\
&= \mathcal{Z} \cdot \left(\prod_{1 \leq i \leq k} \prod_{j \in \omega_i^* \cup \Omega_i'} e(H_i(j), \sigma_{ij}) \right) e(H(m), \sigma_0')
\end{aligned}$$

Hence, the verification passes. This shows that universal forgery can be performed on this scheme by any attacker.

4.4 Attack on multi-level threshold ABS scheme

In this subsection we will focus on how the attack is possible on the multi-level threshold ABS scheme proposed by Swarun et al in [8]. Here, inorder to achieve multi-level threshold properties, the authors use the access structure that was first described in Goyal et al's paper on bounded ciphertext policy ABE [4]. They denote the access tree structure by $\mathcal{T}$. Now, let us look at how the key is generated for a signer possessing a set of attributes θ such that only those users satisfying the access tree can produce a signature:

4.4.1 Key Generation

The key generation algorithm outputs a private key that enables the signer to sign on any message m under a bounded threshold circuit $\mathcal{T}$, as long as $\mathcal{T}(\theta) = 1$. Choose a random polynomial $q(x)$ for each non-leaf node x in the universal bounded threshold circuit $\mathcal{T}_u$. These polynomials are chosen in the following way in a top-down manner, starting from the root node r. For each x, set the degree c_x of the polynomial q_x to be one less than the threshold value, i.e., $c_x = num - 1$. Now, for the root node r, set $q_r(0) = \chi$ and choose c_r other points of the polynomial q_r randomly to define it completely. For any other non-leaf node x, assign $q_x(0) = q_{par(x)}(index(x))$ and choose c_x other points randomly to completely define q_x.

The secret values are given to the user by generating a new attribute set $\theta^* = \theta \cup \delta$. For all $i \in \theta^*$:

1. If $i \in \theta$, for each $x \in \Psi_{\mathcal{T}_u}$
   (a) Choose $r_{x,i} \in \mathbb{Z}_p^*$
   (b) Compute $d_{x,i0} = \left\{ g_2^{q_x(i)} H_2(x||i)^{r_{x,i}} \right\}$, $d_{x,i1} = g^{r_{x.i}}$
2. If $j \in \delta$, for each $y \in \Phi_{\mathcal{T}_u}$
   (a) Choose $r_{y,j} \in \mathbb{Z}_p^*$
   (b) Compute $d_{y,j0} = \left\{ g_2^{q_y(j)} H_2(y||j)^{r_{y,j}} \right\}, d_{y,j1} = g^{r_{y,j}}$

Thus the private key is $\{d_{x,i0}, d_{x,i1} | x \in \Psi_{\mathcal{T}_u}, i \in \theta\} \cup \{d_{y,i0}, d_{y,i1} | y \in \Phi_{\mathcal{T}_u}, i \in \delta\}$.

4.4.2 Attack

In order to model this attack, we will first consider all the non-leaf nodes x at depth $d-1$ i.e all $x \in \Psi_{\mathcal{T}_u}$. By using our method of attack, we will try to extract $g_2^{q_x(0)}$, for each of these nodes. Once, we get these values, we can use these to satisfy the threshold of the parent node and get its share of the secret and so on until we reach the root. To do this we will first establish the congruence of the key generation in this scheme to that of [9]. Here, we can note that $H_2(x||i)$ is a publicly computable function, the definition of which has been established in the setup. Thus, we can consider it to be equivalent to the hash function $H_2(i)$ as used in schemes [9, 10]. If we now use the notation H_{xi} to indicate the value of $H_2(x||i)$ where x is the node under consideration and i is the attribute under consideration and then evaluate $\Pi_{i \in S} d_{x,i0}^{\Delta_{i,S}(0)}$ for pairs of attributes, $S \in \{\{A,B\},\{A,C\},\{A,D\},\{B,C\},\{B,D\},\{C,D\}\}$ to get the following equations. (It is important here to note that this is possible because, a private key component is given for all of the users attributes for each and every non-leaf node in $\Psi_{\mathcal{T}_u}$).

$$X_1 = g_2^{q_x(0)} H_{xA}^{r_{xA}\Delta_{AB}} H_{xB}^{r_{xB}\Delta_{BA}} \qquad X_2 = g_2^{q_x(0)} H_{xA}^{r_{xA}\Delta_{AC}} H_{xC}^{r_{xC}\Delta_{CA}}$$
$$X_3 = g_2^{q_x(0)} H_{xA}^{r_{xA}\Delta_{AD}} H_{xD}^{r_{xD}\Delta_{DA}} \qquad X_4 = g_2^{q_x(0)} H_{xB}^{r_{xB}\Delta_{BC}} H_{xC}^{r_{xC}\Delta_{CB}}$$
$$X_5 = g_2^{q_x(0)} H_{xB}^{r_{xB}\Delta_{BD}} H_{xD}^{r_{xD}\Delta_{DB}} \qquad X_6 = g_2^{q_x(0)} H_{xC}^{r_{xC}\Delta_{CD}} H_{xD}^{r_{xD}\Delta_{DC}}$$

This leads us to equations that have the exact same form as the primary equations in (1). Thus, we can extract $g_2^{q_x(0)}$ for all $x \in \Psi_{\mathcal{T}_u}$. Now, we have the secret shares of all the children nodes at depth $d-1$. We are in a position to interpolate and get the share of the parent nodes as well. By definition, $q_x(0) = q_{par(x)}(index(x))$, so we can use interpolation and get $q_{par(x)}(0)$. Thus, by doing the procedure of interpolation recursively at each level, we can get the value at the root, $g_2^{q_r(0)} = g^{\chi}$. Now, the attacker is just as capable of generating keys for any set of attributes as the key generating authority.

4.5 Total break on hidden ABS without anonymity revocation

The hidden attribute-based signatures without anonymity revocation was proposed by Li and Kim in [11]. This scheme has been largely derived from the work in [9] and hence it has the exact setup and key-extract phases as in [9] and [10].

Now, any signer who gets a set of keys from the key-generating authority, can use the same procedure as described in Section 3.4 to retrieve the component g_2^x. This will enable the attacker to use the strategy in Section 3.7 and impersonate the key generating authority. Hence, the key extract algorithm in this scheme [11] leads to a total break attack on the system.

ABS Scheme	Type of break	Source
Flexible Threshold	Total Break	[LAS+10][9]
Multiple Authorities	Universal Forgery	[LAS+10][9]
Attribute-Based Ring	Total Break	[LK08][10]
Threshold	Total Break	[SSN09][15]
Multi-level threshold	Total Break	[KABPR10][8]
Hidden ABS	Total Break	[LK10][11]

Table 1: Table summarizing the attacks

5 Conclusion

The following table summarizes the attacks shown in the previous sections.

Our observations have led us to conclude that the attack is possible when the following conditions are satisfied:

1. The universe of attributes U contains a sufficiently large number of elements as compared to the threshold for a signature.

2. The user possesses a lot more than d attributes although (s)he might not have t out of the given U^*.

It is important here to note that the attack was made on the key construction and not on the signature itself. The primary flaw in the key construction is the combination of Waters' signature along with the idea of secret shares to distribute the master secret, and at the same time giving each person lot more shares than what is required for the computation. These additional shares - in the form of dummies and other attributes the signer has which are not a part of ω^*, gave the signer multiple ways to recover a derivative of the master secret key, which in turn lead to the attack. We are yet to see a scheme that makes use of the linear secret sharing scheme combined with Waters' signature, which is resistant to the attacks we have proposed here.

References

[1] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334, 2007.

[2] Xavier Boyen. Mesh signatures. In *Proceedings of the 26th annual international conference on Advances in Cryptology*, EUROCRYPT '07, pages 210–227, Berlin, Heidelberg, 2007. Springer-Verlag.

[3] Jan Camenisch. Efficient and generalized group signatures. In *Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'97, pages 465–479, Berlin, Heidelberg, 1997. Springer-Verlag.

[4] Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. Bounded ciphertext policy attribute based encryption. In *Proceedings of the 35th international colloquium on Automata, Languages and Programming, Part II*, ICALP '08, pages 579–591, Berlin, Heidelberg, 2008. Springer-Verlag.

[5] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, CCS '06, pages 89–98, New York, NY, USA, 2006. ACM.

[6] Dalia Khader. Attribute based group signature with revocation. Cryptology ePrint Archive, Report 2007/241, 2007. `http://eprint.iacr.org/`.

[7] Dalia Khader. Attribute based group signatures. Cryptology ePrint Archive, Report 2007/159, 2007. `http://eprint.iacr.org/`.

[8] Swarun Kumar, Shivank Agrawal, Subha Balaraman, and C Pandu Rangan. Attribute based signatures for bounded multi-level threshold circuits. In *Proceedings of the 7th European Workshop on Public Key Services, Applications and Infrastructures*, EuroPKI '10, 2010 (to appear).

[9] Jin Li, Man Ho Au, Willy Susilo, Dongqing Xie, and Kui Ren. Attribute-based signature and its applications. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '10, pages 60–69, New York, NY, USA, 2010. ACM.

[10] Jin Li and Kwangjo Kim. Attribute-based ring signatures. Cryptology ePrint Archive, Report 2008/394, 2008. http://eprint.iacr.org/.

[11] Jin Li and Kwangjo Kim. Hidden attribute-based signatures without anonymity revocation. *Inf. Sci.*, 180:1681–1689, May 2010.

[12] Hemanta Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. Cryptology ePrint Archive, Report 2008/328, 2008. http://eprint.iacr.org/.

[13] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, pages 554–567. Springer-Verlag, 2001.

[14] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.

[15] Siamak F. Shahandashti and Reihaneh Safavi-Naini. Threshold attribute-based signatures and their application to anonymous credential systems. In *Proceedings of the 2nd International Conference on Cryptology in Africa: Progress in Cryptology*, AFRICACRYPT '09, pages 198–216, Berlin, Heidelberg, 2009. Springer-Verlag.

[16] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. Cryptology ePrint Archive, Report 2008/290, 2008. http://eprint.iacr.org/.