Biuletyn WAT
Vol. LVII, Nr 4, 2008

# Blackmail Warning Verifiably Encrypted Signatures from Bilinear Pairing

JACEK POMYKAŁA, TOMASZ TRABSZYS

University of Warsaw, Faculty of Mathematics, Informatics and Mechanics,
2 Banacha str., Warsaw, Poland
pomykala@mimuw.edu.pl

**Abstract.** We present a new cryptographic primitive: blackmail warning signature scheme. In distinction to the ordinary signature it allows the signer to include in the signature the additional information whether it was voluntary or forced. The protocol based on verifiably encrypted signature in the Gap Diffie-Hellman group is provably secure in the random oracle model. It may be applied for the fair exchange and signing rights designation protocols.
**Keywords:** cryptography, blackmail warning signature, verifiable encrypted signature, Id-based cryptosystem, fair exchange
**AMS Classification:** 94A60 (Cryptography)

## 1. Introduction

The signature schemes assuring the privacy and security of signers in the electronic commerce are of the particular interest. The existed solutions based for example on the group signatures (see e.g. [1]), blind signatures [2] or group blind signatures [3] allow to protect the personal data and userłs anonymity, in the electronic transactions. Another feature of signer's security is related to the „fair" contract signing requirements (see e.g. [4]).

The „fair exchange" protocols are traditionally realized by the participation of the trusted party (Trustee) playing the passive role in the corresponding protocol. It „intervenes" only in case when one of the signers is dishonest. Another example where the Trustee might intervene concerns

e.g. the forced signatures. In that case, he is the party (engaged only in case of blackmail) recognizing the forced signature and participating in the construction of the blackmail proof. More extended protocol (which we shall consider in this paper) regards the Trustee as an active party in the sense that it participates also in legitimating the voluntary signature.

In this connection we shall propose a new cryptographic primitive called the blackmail warning signature. In distinction to the ordinary signature it allows the signer to include in the signature the additional information whether it was voluntary or forced. The BWVES enables the Trustee to recognize the blackmail and prepare the suitable proof for the Judge. On the other hand the Verifier is not able to discover whether the signature was voluntary or forced (which may be crucial from the signer's security point of view).

In the protocol, the Signer selects randomly the blackmail value $\alpha$ and sends the corresponding commitment to the Trustee. The Trustee signs it and sends the corresponding cryptogram back to the Signer. The Signer uses the random $r$ or the blackmail value $\alpha$ to compute the corresponding verifiably encrypted signature. If the signature is forced, the Trustee notifies the suitable security service about the blackmail.

In the article we present two types of the blackmail warning verifiably encrypted signatures (ve-signatures). One applies (the particular case) of the short Boneh-Gentry-Lynn-Shacham signature, the other the ID-based verifiably encrypted signature of Cheng, Liu and Wang. The important ingredient in the construction of both protocols is the existence of suitable bilinear pairing in the corresponding group structure. Their provable security is achieved in the random oracle model.

## 2. Related work

The proposed cryptographic protocols work in the Gap Diffie-Hellman groups (GDH groups). The first construction of the Gap Diffie-Hellman group has been proposed in [6]. In [7] and [8] the first examples of digital signatures working in the GDH group were given. The formal definition of security of the Identity-based signature scheme (together with the corresponding protocol) in GDH groups was given in [10]. The first ID-based verifiable encrypted signature based on bilinear pairing has been proposed in [11]. Our security proof of the new cryptographic primitive applies the ideas of [5] and [11].

## 3. Notations and assumptions

In this paper we shall consider the bilinear map $e : G_1 \times G_1 \rightarrow G_2$ where $G_1 = (G_1, +)$ is additive and $G_2 = (G_2, \cdot)$ multiplicative group of prime order $p$ (respectively). We assume that $e$ satisfies the following conditions:

- **Bilinear:** $e(aR, bQ) = e(R, Q)^{ab}, \quad \forall R, Q \in G_1$ and $\forall a, b \in Z_p^*$
- **Nondegenerate:** $e(P, P) \neq 1$ for some $P$ from $G_1$
- **Computable:** there exists an efficient algorithm to compute $e(\cdot, \cdot)$

### Computational Diffie-Hellman problem (CDH)

Given the triple $(P, Q, R)$ compute the point $S \in G_1$ such that the discrete logarithm of $S$ in the base $R$ coincides with the discrete logarithm of $Q$ in the base $P$.

### Decisional Diffie-Hellman problem (DDH)

Given the quadruple $(P, Q, R, S)$ decides whether the discrete logarithm of $S$ in the base $R$ coincides with the discrete logarithm of $Q$ in the base $P$.

The bilinear map $e$ implies that the corresponding DDH problem is tractable in $G_1$. However, if the corresponding CDH problem still remains intractable, the group $G_1$ is called the gap Diffie-Hellman group. For the explicit definitions of the corresponding bilinear Weil or Tate pairings we refer the reader to [9].

## 4. Blackmail Warning Verifiably Encrypted Signature (BWVES) based on BVES scheme

BWVES is founded on a special case of the bilinear verifiably encrypted signature scheme - BVES ([11], section 4.4, with $G_1 := \tilde{G_1} = \tilde{G_2}$ and $G_2 := \tilde{G_T}$, where $\tilde{G_1}, \tilde{G_2}, \tilde{G_T}$ are used in [11]).

There are four parties taking part in the protocol: Signer, Trustee, Judge and Verifier. In the BWVES, the Trustee plays an active role. The encrypted signature may be verified by any user. In order to derive the proper (decrypted) signature, the Verifier must request the Trustee for the justification (decryption). The Trustee decrypts the signature only if the corresponding randomly chosen parameter $r$ does not coincide with the blackmail value $\alpha$. Otherwise he notifies the suitable security service. Below we point out the consecutive steps followed by the protocol:

1. Setup: $(\kappa) \rightarrow (G_1, G_2, e, P, Q, H, h)$
   Having, as an input, the public data and the security parameter $\kappa$, the tuple $(G_1, G_2, e, P, Q, H, h)$ is generated as output, where $P$ and $Q$ are random nonzero elements of the Gap Diffie-Hellman group $G_1$. Here $H$, $h$ are secure hash functions, $H : G_1 \rightarrow G_1$, $h : \{0,1\}^* \rightarrow G_1$.
2. Long-term key generation: $() \rightarrow (u, U)$
   The Signer chooses randomly his private key $u \in Z_p^*$ and computes the corresponding public key $U = uP$. The Certificate Autority certifies the public key. The private-public key pair $(t, T)$ for the Trustee is generated in the same way.
3. Short-term key request: $() \rightarrow (\alpha, Sign))$
   The Signer generates the random $\alpha \in Z_p^*$ and sends to the Trustee the value $\alpha Q$. The Trustee computes the signature $Sign = tH(\alpha Q)$ and sends $Sign$ back to the Signer. The Signer checks the correctness of the signature $tH(\alpha Q)$ i.e. the equality: $e(P, Sign) = e(T, H(\alpha Q))$. If so, the value $\alpha$ can be used by the Signer for the forced ve-signature. Each time the forced ve-signature is generated, the new random $\alpha$ is selected and the updated value $\alpha Q$ is sent to the Trustee. The Trustee signs it and sends $Sign = tH(\alpha Q)$ back to the Signer. The Signer may use this pair as the „proof" of the forced ve-signature.
4. Signing: $(u, m) \rightarrow \omega$
   Given
   the message $m \in \{0,1\}^*$, the Signer generates the random $r \in Z_p^*$ and computes the ve-signature of $m$ : $\omega = [R, uh(m) + rT]$, where $R = rP$.
5. Forced signing: $(u, \alpha, Sign, m) \rightarrow \omega$
   Given the message $m \in \{0,1\}^*$, $\alpha \in Z_p^*$ (together with the $Sign = tH(\alpha Q)$ from the Trustee), the Signer computes the blackmail ve-signature of $m$ : $\omega = [R, W]$, where $W = uh(m) + rT$, $r = \alpha$ and $R = \alpha P$. The ve-signature together with the new, updated value $\alpha Q$ is sent to the Trustee.
6. E-verification: $(U, m, \omega) \rightarrow \{true, false\}$
   To verify the encrypted signature $[R, W = uh(m) + rT]$
   any user checks if $e(P, W) = e(U, h(m))e(R, T)$.
7. Blackmail discovery: $(\alpha Q, \omega) \rightarrow \{true, false\}$
   Given $\omega = [R, W]$, the Trustee checks if $e(R, Q) = e(P, \alpha Q)$ in which case he notifies the corresponding security service.
8. Signature recovery: $(t, \omega) \rightarrow \sigma$
   Given $\omega = [R, W]$, the Trustee computes the proper (decrypted) signature: $\sigma = uh(m) = W - tR$ and sends it to the Verifier (if requested).

9. Verification: $(U, m, \sigma) \rightarrow \{true, false\}$
   The Verifier can check the validity of the decrypted signature. Namely the decrypted signature is accepted if and only if $e(P, \sigma) = e(U, h(m))$.
10. Blackmail proving: $(\omega, \tau) \rightarrow \{true, false\}$
    After generation of the forced ve-signature $\omega = [\alpha P, W]$, the Signer may prove that the ve-signature was forced giving the „proof" $\tau = (\alpha Q, tH(\alpha Q))$ for the Judge. To verify the proof, the Judge checks if $e(P, \alpha Q) = e(\alpha P, Q)$ and whether signature $tH(\alpha Q)$ is correct.

The scheme consists of the corresponding 10 algorithms: BWVES = (Setup, Keygen, KeyRequest, Sign, ForceSign, E-Verify, Blackmail, Recover, Verify, Prove).

## 5. Security of BWVES

The security of the blackmail warning ve-signature extends the familiar security notion of the verifiably encrypted signature [11] with the additional requirements given below:

**Definition 5.1.** A signature scheme (with a third party Trustee) is blackmail secure if it has the following properties:

- Blackmail validity
  A forced signature can always be proved forced either by the Trustee or the Signer.
- Blackmail indistinguishability
  Forced signature is indistinguishable from an ordinary signature for any party except the Trustee.
- Blackmail unforgeability
  If at least one of the Trustee or the Signer is honest, it is infeasible to prove that the ordinary signature was forced.

The above definition is related to the general signature scheme. However, in view of the announced application from the Introduction we will relate it to the corresponding ve-signatures.

Blackmail validity requires that the forced ve-signature $\omega$ provides the „force proof", which could be verified by the Judge. It means that $Blackmail(\alpha Q, ForceSign(u, \alpha, Sign, M)) = true$ and $Prove(\omega, \tau) = true$ holds for all messages $m$ and for all properly-generated keypairs for the Trustee and signers, where $\tau$ is a properly-generated „force proof" $\tau = (\alpha Q, tH(\alpha Q))$.

Blackmail indistinguishability requires that given a ve-signature it is infeasible to decide if it is forced or voluntary. The advantage of the algorithm $I$ in blackmail distinguishability, given access to the ordinary ve-signature creation oracle $O$, the forced ve-signature creation oracle $F$ along with a hash oracle, is

$$AdvBD_I \stackrel{def}{=} Pr\left[\begin{array}{l} Params \stackrel{R}{\leftarrow} Setup, \\ (PK, SK) \stackrel{R}{\leftarrow} KeyGen, \\ \tilde{b} = b : M \stackrel{R}{\leftarrow} I^{O,F}(Params, PK), \\ b \stackrel{R}{\leftarrow} \{Ordinary, Forced\}, \\ \tilde{b} \stackrel{R}{\leftarrow} I^{O,F}(\omega_b(M)) \end{array}\right] - 1/2.$$

The probability is taken over the coin tosses of the Setup algorithm, the key-generation algorithms, of the oracles, of $b$ and the algorithm $I$. The algorithm is additionally constrained by the fact that $M$ wasn't the subject of a ve-signature query neither for the oracle $O$ nor $F$.

A ve-signature scheme is blackmail indistinguishable if for any polynomial-time algorithm $I, AdvBD_I$ is less a $1/f(\kappa)$ for any polynomial $f$ (i.e. there exists no polynomial-time algorithm $I$ with a significant advantage $AdvBD_I$).

Blackmail unforgeability requires that it is intractable to produce a proof that an ordinary ve-signature is forced. The advantage of the algorithm $A$ in forging a proof that an ordinary ve-signature is forced, given access to the ordinary ve-signature creation oracle $O$, the forced ve-signature creation oracle $F$ along with a hash oracle, is

$$AdvBF_A \stackrel{def}{=}$$

$$Pr\left[\begin{array}{ll} E - Verify(PK, M, \omega) = true & Params \stackrel{R}{\leftarrow} Setup, \\ Prove(\omega, \tau) = true & : (PK, SK) \stackrel{R}{\leftarrow} KeyGen, \\ & (M, \omega, \tau) \stackrel{R}{\leftarrow} A^{O,F}(Params, PK) \end{array}\right].$$

The probability is taken over the coin tosses of the Setup algorithm, the key-generation algorithms, of the oracles and the algorithm $A$. The algorithm is additionally constrained by the fact that the $M$ was the subject of the ordinary ve-signature query, but there was no query for the forced ve-signature of $M$.

A ve-signature scheme is blackmail unforgeable if there exists no polynomial-time algorithm $A$ with a significant advantage $AdvBF_A$.

In this section we refer to the requirements of validity, unforgeability and opacity for the underlying verifiably encrypted signature. We note that in our construction, the ability of creating forced ve-signatures doesn't have any influence for the latter properties. Next, we will show that our scheme is blackmail secure. BWVES is built on the bilinear verifiably encrypted signature scheme ([11], section 4.4, with $G_1 := \tilde{G_1} = \tilde{G_2}$ and $G_2 := \tilde{G_T}$, where $\tilde{G_1}, \tilde{G_2}, \tilde{G_T}$ are used in [11]). Therefore the corresponding proofs follow directly.

### 5.1. Validity, Unforgeability, Opacity

To prove validity it is sufficient to remark that:

$$e(P, W) = e(P, uh(m) + rT) = e(P, uh(m))e(P, rT) = e(U, h(m))e(R, T).$$

Moreover the corresponding Unforgeability and Opacity properties follow by the application of theorems 4.4 and 4.5 in [11], respectively. □

### 5.2. Blackmail security

#### Blackmail validity

We have to show that a forced ve-signature would always be correctly recognized. When a Signer generates a forced ve-signature it's obvious that it would pass the Trustee's verification. Furthermore he could himself supply a proof $(\alpha Q, tH(\alpha Q))$ for his forced ve-signature, because he has received it in the short-term key request. □

#### Blackmail indistinguishability

Since $\alpha$ is chosen randomly, the value $\alpha P$ in the forced ve-signature is indistinguishable from the random element $R$ in an ordinary ve-signature. □

#### Blackmail unforgeability

Consider an adversary who forges a proof that an ordinary ve-signature is forced. If the Trustee is honest the adversary would have to forge the Trustee's signature on the random message $H(rQ)$, given the public key

$T = tP$. Hence, he would have to solve the CDH problem for the tuple $(P, T, H(rQ))$.

If the Signer is honest we can assume without loosing the generality, that the adversary knows the private key $t$ of the Trustee, but does not know the value of $r$ (since the Signer is honest). Therefore to supply the „forged" proof of the force $\tau = (rQ, tH(rQ))$ he would have to know $rQ$, i.e. solve the CDH problem for the tuple $(P, R, Q)$ which we assumed to be untractable [1]. □

## 6. ID-based BWVES scheme

Here we shall describe the analogous protocol which incorporates the ID-based verifiably encrypted signature (see [5]) to our scheme. Below we point out the consecutive steps followed by the protocol:

1. Setup of the system: $(\kappa) \rightarrow (G_1, G_2, e, P, Q, H, h)$
   Having as an input the public data and the security parameter $\kappa$, the tuple $(G_1, G_2, e, P, Q, H, h)$ is generated as output, where $P \in G_1$ and $H, h$ are secure hash functions, $H : G_1 \rightarrow G_1$, $h : \{0,1\}^* \times G_1 \rightarrow Z_p^*$.

   At the end of the setup phase, the Private Key Generator (PKG) generates the master key $s \in Z_p^*$, and publishes the system public key $\Omega = sP$. The Trustee generates his private-public key pair: $(t, T)$, where $T = tP$ for the Trustee. The Certificate Authority certifies public keys of PKG and the Trustee.
2. Long-term key generation: $(s, ID) \rightarrow (Q_{ID}, D_{ID})$
   For the given identity $ID$, PKG generates the private-public key pair: $(D_{ID}, Q_{ID})$, where $Q_{ID} = H(ID)$, $D_{ID} = sQ_{ID}$.
3. Short-term key request: $() \rightarrow (\alpha, Sign))$
   The Signer generates the random $\alpha \in Z_p^*$ and sends to the Trustee the value $\alpha Q$. The Trustee computes the signature $Sign = tH(\alpha Q)$ and sends $Sign$ back to the Signer. The value $\alpha$ is used by the Signer for the forced ve-signature. Each time the forced ve-signature is generated,

[1] The Trustee, given the forced ve-signature $\omega_1 = [\alpha P, W_1]$ of $m_1$ and the ordinary ve-signature $\omega_2 = [R, W_2]$ of $m_2$ is capable to generate the forced ve-signature for $m_2$ with the already used value $\alpha$. Knowing that $W_2 = uh(m) + rT$, he could produce the valid ve-signature $[\alpha P, W_2 - tR + t(\alpha P)]$ and a proof of the force $\tau$ for it, since the Trustee knows $\alpha P$ from the forced ve-signature $\omega_1$ and $\tau$ from the short-term key generation process.
It doesn't contradict the definition of blackmail unforgeability, however it is also easy to overcome this inconvenience by replacing the hash function $h : \{0,1\}^* \rightarrow G_1$ with $\bar{h} : \{0,1\}^* \times G_1 \rightarrow G_1$. Namely, the improved ve-signature $\omega$ would be $[R, u\bar{h}(m, R) + rT]$ and the signature $\sigma$ would be $[R, u\bar{h}(m, R)]$.

the new random $\alpha$ is selected and the updated value $\alpha Q$ is sent to the Trustee. The Trustee signs it and sends $tH(\alpha Q)$ back to the Signer. The Signer may use $Sign$ as the „proof" of the forced ve-signature.

4. Signing: $(D_{ID}, m) \rightarrow \omega$
   Given the message $m \in \{0,1\}^*$ the Signer with the identity ID generates the random values $r, v \in Z_p^*$ and computes the ve-signature of $m : [R, V, W]$, where $W = r\Omega + h(m, R)D_{ID} + vT$, $R = rP$ and $V = vP$.
5. Forced signing: $(D_{ID}, \alpha, Sign, m) \rightarrow \omega$
   Given the message $m \in \{0,1\}^*$ and the corresponding „blackmail" random parameter $\alpha$, the Signer generates the random value $r \in Z_p^*$, computes the blackmail ve-signature of $m : [R, V, W]$, where $W = r\Omega + h(m, R)D_{ID} + \alpha T$ and $V = \alpha P$.
6. E-verification: $(ID, m, \omega) \rightarrow \{true, false\}$
   To verify the encrypted signature $[R, V, W]$,
   where $W = r\Omega + h(m, R)D_{ID} + vT$,
   any user can check if $e(P, W) = e(\Omega, R + h(m, R)Q_{ID})e(V, T)$.
7. Blackmail discovery: $(\alpha Q, \omega) \rightarrow \{true, false\}$
   Given $\omega = [R, V, W]$, the Trustee checks if $e(V, Q) = e(P, \alpha Q)$ in which case he notifies the corresponding security service about blackmail.
8. Signature recovery: $(t, \omega) \rightarrow \sigma$
   Given ve-signature $\omega = [R, V, W]$ the Trustee computes $W' = W - tV$ and sends $\sigma = [R, W']$ to the Verifier (if requested).
9. Verification: $(ID, m, \sigma) \rightarrow \{true, false\}$
   The Verifier can check the validity of the decrypted signature. Namely the decrypted signature $\sigma = [R, W']$ is accepted if and only if $e(P, W') = e(\Omega, R + h(m, R)Q_{ID})$.
10. Blackmail proving: $(\omega, \tau) \rightarrow \{true, false\}$
    After generation of the forced ve-signature $\omega = [R, \alpha P, W]$, the Signer may prove that the ve-signature was forced giving the „proof" $\tau = (\alpha Q, tH(\alpha Q))$ for the Judge. To verify the proof, the Judge checks if $e(P, \alpha Q) = e(\alpha P, Q)$ and whether the signature $tH(\alpha Q)$ is correct.

The scheme consists of the corresponding 10 algorithms: ID-BWVES = (Setup, Keygen, KeyRequest, Sign, ForceSign, E-verify, Blackmail, Recover, Verify, Prove).

## 7. Security of ID-Based BWVES

In this section we prove validity, unforgeability and opacity for the underlying ID-based verifiably encrypted signature. We note, that our construction of the blackmail feature doesn't have any influence on the latter

properties. Unforgeability and validity have been proven in [5], proof of opacity will be built from the scratch[2], using the idea presented in the proof for a certificate based signature scheme in [11]. Theorem 6 (in [5]) claiming the property of opacity is not correct[3].

Blackmail security (7.2) in the ID-based enviroment is derived directly from the proof of blackmail security (5.2) of the BWVES scheme. Formal security definitions for the blackmail validity, indistinguishability and unforgeability in the ID-based model are also presented in section (7.2).

### 7.1. Verifiably Encrypted Signature's Security

Validity

$$e(P,W) = e(P, r\Omega + hD_{ID} + vT) = e(P, r\Omega + hD_{ID})e(P, vT) =$$
$$= e(P, s(rP + hQ_{ID}))e(R,T) = e(\Omega, R + hQ_{ID})e(V,T)$$

where $h = H(m,R)$. □

Unforgeability

According to Thorem 4 (in [5]) section 5. our scheme is unforgeable. □

Opacity

For the proof of opacity we refer to Appendix A.

### 7.2. Blackmail security

We prove that ID-BWVES is blackmail secure exactly the same way as in the BWVES scheme (5.2). In the following sections, we provide the formal blackmail warning signature security statements in the ID-based model. □

[2] This proof was presented simultaneously in Encrypted verifiable ID-based signatures in the gap Diffie-Hellman group, master thesis of T. Trabszys under the supervision of J. Pomykała, Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, 2008.

[3] In the paper [5], proof of theorem 6., the authors claim that if one could extract signature from verifiable encrypted signature it would gain a signature which is unforgeable. That's true. However, what we have to show is that the encryption doesn't supply the adversary with any information which could make that task easy. In the other words, in the same way as in [5] we could prove that encryption like $S||vX$ instead of $S + vX$ has also the property of opacity. That is obviously not true, anyone can cut the word $S||vX$ to obtain the signature $S$.

### Blackmail security in the ID-based model

To define the blackmail security for ID-based signature scheme (with a third party Trustee) we introduce slightly modified definitions of the algorithms taking advantage of the following properties: blackmail validity, blackmail indistinguishability and blackmail unforgeability. The adversary's algorithm additionaly can query the identity corruption oracle $C$ for the private-public key pair of the selected identity.

The above definition is related to the general ID-based signature scheme. For better clarity, similarly as in the previous protocol (5.2), we will relate the blackmail security to the corresponding ID-based ve-signature.

#### Blackmail validity in the ID-based model

Blackmail validity requires that the forced ve-signature $\omega$ provides the „force proof", which could be verified by the Judge. It means that $Blackmail(\alpha Q, ForceSign(D_{ID}, \alpha, Sign, m)) = true$ and $Prove(\omega, \tau) = = true$ holds for all messages $m$ and for all properly-generated key-pairs for the Private Key Generator, Trustee and signers, where $\tau$ is the properly-generated „force proof" $\tau = (\alpha Q, tH(\alpha Q))$.

#### Blackmail indistinguishability in the ID-based model

Blackmail indistinguishability requires that given a ve-signature it is infeasible to decide if it is forced or ordinary. The advantage of the algorithm $I$ in blackmail distinguishability, given access to the ordinary ve-signature creation oracle $O$, the forced ve-signature creation oracle $F$, the identity corruption oracle $C$ and along with a hash oracle, is:

$$AdvBD_I \stackrel{def}{=} Pr\left[\begin{array}{c} Params \stackrel{R}{\leftarrow} Setup, \\ \tilde{b} = b : (M, ID) \stackrel{R}{\leftarrow} I^{O,F,C}(Params, PK), \\ b \stackrel{R}{\leftarrow} \{Ordinary, Forced\}, \\ \tilde{b} \stackrel{R}{\leftarrow} I^{O,F,C}(\omega_b(M, ID)) \end{array}\right] - 1/2.$$

The probability is taken over the coin tosses of the value $b$, the corresponding oracles and the Setup algorithm, and of the algorithm $I$. The algorithm is additionally constrained by the fact that the challenge pair message-identity $(M, ID)$ wasn't the subject of a ve-signature query neither for the oracle $O$ nor $F$. Furthermore, $ID$ wasn't the subject of a corruption query for the oracle $C$.

A ve-signature scheme is blackmail indistinguishable if there exists no polynomial-time algorithm $I$ with a significant advantage $AdvBD_I$.

### Blackmail unforgeability in the ID-based model

Blackmail unforgeability requires that it is intractable to produce a proof that an ordinary ve-signature is forced. The advantage of the algorithm $A$ in forging a proof that an ordinary ve-signature is forced, given access to the ordinary ve-signature creation oracle $O$, the forced ve-signature creation oracle $F$, the identity corruption oracle $C$ along with a hash oracle, is

$$AdvBF_A \stackrel{def}{=}$$

$$Pr\left[\begin{array}{ll} E-Verify(ID,m,\omega)=true : Params \stackrel{R}{\leftarrow} Setup, & \\ Prove(\omega,\tau)=true & (M,ID,\omega,\tau) \stackrel{R}{\leftarrow} A^{O,F,C}(Params,PK) \end{array}\right].$$

The probability is taken over the coin tosses of the Setup algorithm, of the oracles and the algorithm $A$. The algorithm is additionally constrained by the fact that the pair $(M, ID)$ `was` the subject of the ordinary ve-signature query, but there was no query for the forced ve-signature of $(M, ID)$. Furthermore, $ID$ wasn't the subject of a corruption query for the oracle $C$.

A ve-signature scheme is blackmail unforgeable if there exists no polynomial-time algorithm $A$ with a significant advantage $AdvBF_A$.

## Appendix A – Proof of opacity

Let us first define the extraction problem in groups where the computional Diffie-Hellman Problem is considered hard. It is a special case (for $k = 2$) of the subaggregate extraction defined in [11].

**Definition 7.1.** Let $G_1$ be the Diffie-Hellman group of prime order $p$ and let $e$ be the corresponding bilinear pairing (see section 3). We consider the tuple $\varsigma$ of elements from $G_1$, $\varsigma = (P, S_1, S_2, T_1, T_2, Z)$, such that exists $S$ and $T$, that:

- $Z = S + T$,
- $(P, S_1, S_2, S)$ is a DH tuple,
- $(P, T_1, T_2, T)$ is a DH tuple.

Given $\varsigma$ defined above the extraction problem is to compute $(S, T)$.

If the extraction problem is easy, then ID-BWVES doesn't have the property of opacity, since:

$$\omega = (R, V, W) = (rP, vP, S + vT) = (rP, vP, r\Omega + hD + vT) =$$
$$= (rP, vP, rsP + shQ + tvP) = (R, V, s(R + hQ) + tV)$$

let $\theta = R + hQ$, so:

$$\omega = (R, V, s\theta + tV)$$

and now if the extraction would be easy for the tuple $(P, \Omega, \theta, TV, W)$ then we could get $S = W - tV$), where $S$ is the underlying decrypted signature in the ID-BWVES scheme. We note that $(P, \Omega, \theta, T, V, W)$ has the properties listed in the definition (7.1) of the extraction problem.

**Definition 7.2.** We say that the $\varsigma$-generator $\mathcal{G}_\varsigma$ generates randomly $\varsigma = (P, S_1, S_2, T_1, T_2, Z)$ defined above (7.1), when the probability distrubution is identical to the probability distribution of the $\varsigma$-generator $\mathcal{H}_\varsigma$ defined as follows:
$\mathcal{H}_\varsigma$ chooses randomly $s_1, s_2, t_1, t_2$ from $Z_p^*$
and returns the output $(P, s_1P, s_2P, t_1P, t_2P, (s_1s_2)P + (t_1t_2)P)$.

Assume that there exists the polynomial-time algorithm $\mathcal{A}$ breaking the opacity property of the ID-BWVES scheme with the nonnegligible probability $\delta = \delta(\kappa)$. We would construct the polynomial-time algorithm $\mathcal{B}$, which would simulate for $\mathcal{A}$ ID-BWVES enviroment and using it's output would solve the extraction problem given at the begining. Let us see, that generating forced and ordinary ve-signature is independent from the Opacity property – let us therefore restrict ourselves to the voluntary ve-signatures. We could describe $\mathcal{B}$ as a list of responses to queries from $\mathcal{A}$ and procedures *Setup*, *Output*. Assume, that $\mathcal{B}$ records every query from $\mathcal{A}$ with it's in-the-middle computations. Requested memory is linear according to the number of queries.

- Setup:
  $\mathcal{B}$ given the randomly generated (see 7.2) extraction problem $\xi = (P, \Omega, \Psi, T, \Xi, Z)$, groups $G_1$ and $G_2$ with the bilinear pairing $e$ (the same as in the ID-BWVES scheme). He sends to $\mathcal{A}$ the group descriptions and the map $e$. He also sets $\Omega$ as a PKG master public key in the ID-BWVES scheme and $T$ as a Trustee's public key, needed for encrypting signatures.
- H(ID) queries:
  $\mathcal{B}$ checks, if there is no a recorded value for the argument $ID$ (the query was asked before). If not he chooses the random $q_{ID} \in Z_p^*$, computes $Q_{ID} = q_{ID}P$. He records also $D_{ID} = q\Omega$. Note that $D_{ID} = sQ_{ID}$ is a correct private key for the identity $ID$. $\mathcal{B}$ response is $Q_{ID}$.
- H(m, R) queries:
  $\mathcal{B}$ checks, if there is no a recorded value for the argument $(m, R)$ (the query was asked before or the value has been set in the process

of generating ID-BWVES signature). If not, he chooses the random $h_i \in Z_p^*$ and sets $h_i$ as the response.

- Compromise $ID$ queries:
  If $\mathcal{A}$ hasn't asked for a hash value on $ID$, $\mathcal{B}$ processes this query now. He puts $D_{ID}$ as a response, which is computed while responsing to the $H(ID)$ query.
- ID-BWVES ve-signature $(m, ID)$ queries:
  — it's k-th query for a ID-BWVES ve-signature
  If the value $H(m, \Psi)$ is already set, $\mathcal{B}$ finishes with a failure. If $\mathcal{A}$ hasn't asked for a hash value on $ID$, $\mathcal{B}$ processes this query now. While responsing to the latter query, $\mathcal{B}$ will record the $q_{ID}$ value. Now, he chooses the random $t \in Z_p^*$ and puts $H(m, \Psi) = h = t1/q_{ID} \in Z_p^*$. Next he chooses random $y \in Z_p^*$, computes $Y = yP$, puts $V = \Xi + Y$, $W = t\Omega + Z + yT$ and sets $\omega = (\Psi, V, W)$ as the response.
  — it's not the k-th query:
  If $\mathcal{A}$ hasn't asked for a hash value on $ID$, $\mathcal{B}$ processes this query now. Using the user secret key $D_{ID}$ obtained in the processing $H(ID)$ query, $\mathcal{B}$ can produce correct ID-BWVES ve-signature on $m$ according to the scheme. So, he chooses the random values $r, v \in Z_p^*$, computes $R = rP$ and $V = vP$, asks for $h = H(m, R)$ using hash query defined above. Computes $S = r\Omega + hD_{ID}$, $W = S + vT$ and sets $\omega = (R, V, W)$ as the response.
- Decrypt $(m, ID, \omega)$ queries:
  $\mathcal{B}$ checks if $\omega = (R, V, W)$ is the correct ve-signature. Having in mind that ID-BWVES is unforgeable, $\mathcal{A}$ must have asked a ve-signature query and have acquired $\omega$. If the latter was the chosen, $k$-th signing query, $\mathcal{B}$ finishes with a failure. If not, $\mathcal{B}$ must have recorded $S$ in the process of responding to this query. Hence his response is the decrypted signature $\sigma = (R, S)$.
- Output:
  If $\mathcal{A}$ turns out to decrypt the chosen ve-signature (response to the $k$-th ve-signature query), then we will be able to compute a solution to the extraction problem defined at the begining. $\mathcal{A}$ would output $\sigma = (\Psi, S)$ which passes the verification, so it is equal to $(\Psi, \psi\Omega + H(m, \Psi)D_{ID})$, where $\Psi = \psi P$. $H(m, \Psi)$ has been set to $h = t/q_{ID}$, $D_{ID}$ is also known for $\mathcal{B}$. Therefore he can compute $S - hD_{ID} = \psi\Omega$, which would be the solution to the extraction problem $\xi = (\Omega, \Psi, T, \Xi, Z)$.

We note that, since the given extraction problem is randomly generated, responses to the queries are randomly generated – hash values have uniform distribution and ID-BWVES ve-signatures are generated randomly.

It's obvious in all responses, but the $k$-th ve-signature query. $h_k$ is generated using randomly generated value $t$, therefore is random. The $k$-th ve-signature uses elements $\Psi$ and $\Xi + Y$ as random ones. Since the extraction problem is random so is $\Psi$. $\Xi + Y$ has a uniformed distribution since $Y$ was generated randomly[4].

If in the process of generating $k$-th ve-signature the query for $H(m, \Psi)$ has been asked before by $\mathcal{A}$ then $\mathcal{B}$ ends up with a failure. $\mathcal{A}$ is not familiar with a random element $\Psi$, so the probability $\epsilon$ that this situation occurs is negligible, it's easy to see that $\epsilon \leq H_M/p$, where $H_M$ is the (maximum) number of hash queries on pairs $(m_i, R_i)$ made by $\mathcal{A}$ and $p$ is the order of the group $G_1$.

The other situation when $\mathcal{B}$ can end up with a failure is when $\mathcal{A}$ asks for decryption of the ve-signature from the $k$-th ve-signature query. Combining those 2 remarks we can estimate that $\mathcal{B}$ is successful with the probability at least $\frac{1-\epsilon}{H_S}\delta$, where $H_S$ is the number of ve-signature queries made by $\mathcal{A}$ and $\delta$ is the probability that $\mathcal{A}$ is successful. Since the algorithm $\mathcal{B}$ works in a polynomial-time and this concludes the argument.

*Received 22.09.2008; Revised 14.10.2008.*

REFERENCES

[1] M. Bellare, H. Shi, Ch. Zhang, *Foundations of Group Signatures: The Case of Dynamic Groups*, Topics in Cryptology – CT-RSA'05, LNCS, vol. 3376, Springer-Verlag, 2005, pp. 136–153.

[2] D. Chaum, *Blind signatures for untraceable payments*, Advances in Cryptology, CRYPTO'82, 1982, pp. 199–203.

[3] A. Lysyanskaya, Z. Ramzan, *Group blind digital signatures: A scalable solution to electronic cash*, Financial Cryptography (FC'98), LNCS vol. 1465, Springer-Verlag, 1998, pp. 184–197.

[4] N. Asokan, V. Shoup, M. Waidner, *Optimistic fair exchange of digital signatures*, LNCS 1403, 1998, pp. 591–606.

[5] X. Cheng, J. Liu, and X. Wang, *Identity-based aggregate and verifiably encrypted signatures from bilinear pairing*, LNCS, vol. 3483, pp. 1046–1054.

[6] A. Joux, *A one-round protocol for tripartite Diffie-Hellman*, Journal of Cryptology, vol. 17, no. 4, 2004, pp. 263–276.

[7] D. Boneh, C. Gentry, B. Lynn, H. Shacham, *Short signatures from the Weil pairing*, Journal of Cryptology, vol. 17, no. 4, 2004, pp. 297-319.

[8] A. Lysyanskaya, *Unique signatures and verifiable random functions from the DH-DDH separation*, Proceedings of the 22nd Annual International Cryptology Conference on advances in Cryptology, 2002, pp. 597–612.

[4] If we have replaced the response to the $k$-th signature query to the simpler form $(R, V, W) = (\Psi, \Xi, t\Omega + Z)$ it would be easy to distinguish for $\mathcal{A}$ the $k$-th response from the others, since the equation $e(R, V)e(\Omega, T)e(h(m, R)Q_{ID}, \Omega) = e(P, W)$ would always hold. Hence it wouldn't be randomly generated, so the algorithm $\mathcal{A}$ wouldn't have to work successfuly.

[9] D. Boneh, M. Franklin, *Identity Based Encryption from the Weil Pairing*, Advance in cryptology – Crypto'01, LNCS, vol. 2139, Springer-Verlag, Berlin Heidelberg New York, 2001, pp. 213–229.

[10] J.C. Cha, J.H. Cheon, *An Identity-Based Signature from Gap Diffie-Hellman Groups*, Public Key Cryptography – PKC 2003, LNCS, vol. 2567, Springer-Verlag, 2002, pp. 18–30.

[11] D. Boneh, C. Gentry, B. Lynn, H. Shacham, *Aggregate and Verifiably Encrypted Signatures from Bilinear Maps*, EUROCRYPT 2003, LNCS 2656, Springer-Verlag, 2003, pp. 416–432.

JACEK POMYKAŁA, TOMASZ TRABSZYS

## Weryfikowalnie zaszyfrowane podpisy z ostrzeżeniem o wymuszeniu

**Streszczenie.** Prezentujemy nowe pojęcie podpisu z ostrzeżeniem o wymuszeniu. W odróżnieniu od zwykłego podpisu pozwala podpisującemu na przekazanie dodatkowej informacji czy podpis został złożony dobrowolnie. Pokażemy dwa protokoły implementujące powyższą funkcjonalność, oba oparte na weryfikowalnie zaszyfrowanym podpisie w grupie Diffiego-Hellmana. Mogą one znaleźć zastosowanie np. przy sprawiedliwej wymianie. Ścisły dowód bezpieczeństwa przedstawiony jest w modelu z losową wyrocznią (random oracle model).

**Słowa kluczowe:** podpis cyfrowy, kryptosystem oparty na tożsamości, weryfikowalnie szyfrowany podpis cyfrowy, protokół sprawiedliwej wymiany, grupa Diffie-Hellmana z luką obliczeniową

**Symbole UKD:** 94A60