ON THE SECURITY OF THE VERIFIABLY-ENCRYPTED SIGNATURE SCHEME OF BONEH, GENTRY, LYNN AND SHACHAM

F. HESS

Abstract. We discuss the security of the verifiably-encrypted signature scheme of Boneh, Gentry, Lynn and Shacham. It is quite realistic to allow adversaries access to adjudication oracles for different users but the same adjudicator. This presents an extension of the security model considered by Boneh, Gentry, Lynn and Shacham and we describe an efficient attack on their scheme in that model. We then show how to obtain security in this extended model by applying a small modification to their scheme.

1. Introduction

A verifiably-encrypted signature scheme as introduced by Boneh, Gentry, Lynn and Shacham [1] involves three parties, the user or signer, the verifier and a trusted third party, the adjudicator. The basic idea is that the user creates an encryption of his signature on some message using the public key of the adjudicator. The verifier should be able to verify that this encrypted signature is indeed the encryption of the user's signature on the message, but he should not be able to extract the signature. The adjudicator on the other hand is able to extract the signature.

For a verifiably-encrypted signature scheme to be secure, in addition to the necessary properties for the ordinary signature component of the scheme, two further properties are required. The opacity property or security against extraction is that no one except the adjudicator (or user) can extract an ordinary signature under a given user public key from a verifiably-encrypted signature. The unforgeability property is that no one except the user should be able to create a verifiably-encrypted signature from scratch under the user public key.

Boneh, Gentry, Lynn and Shacham [1] show that their verifiably-encrypted signature scheme satisfies the unforgeability and opacity properties. For both properties they allow adversaries access to verifiably-encrypted signature creation and adjudication oracles which may be queried only on the prescribed target user and adjudicator public keys. While this seems sufficient with respect to unforgeability one could well imagine that an adversary may attempt to trick an adjudicator into adjudicating a verifiably-encrypted signature for a different user public key than the target key. In the security model we would accordingly allow that an adversary queries the adjudication oracle on possibly different user keys but always the same adjudicator key. This extension of the security model enables an effective attack against the scheme of Boneh, Gentry, Lynn and Shacham invalidating the opacity property. A similar problem is mentioned in [1] for the case of aggregate signatures, and a possible countermeasure would be to only employ certified public keys. However, using only a small modification we make their scheme secure in the extended security model avoiding any certificate authority.

2. The modified BGLS-signature scheme

Let G be a (multiplicative) cyclic group of prime order p with generator g. Let $e : G \times G \to V$ be a computable, bilinear and non-degenerate map (the pairing) into the group V. We have $e(x, y) = e(y, x)$ for all $x, y \in G$ because G is cyclic.

Let $\mathbb{M} \subseteq \{0,1\}^*$ be a message space and $H : \mathbb{M} \times G \to G$ a full domain hash function. We consider the following modified versions of the GDH-signature scheme of Boneh, Lynn and Shacham [2] and of the bilinear verifiably-encrypted signature scheme of Boneh, Gentry, Lynn and Shacham [1]. We refer to the latter as BGLS-signature scheme.

MGDH-signature scheme

KeyGen : Pick a random $a \in \mathbb{Z}_p$ and compute $v \leftarrow g^a$. The public key is $v \in G$. The secret key is $a \in \mathbb{Z}_p$.

Sign : Given a message $M \in \mathbb{M}$ and a secret key a, compute $h \leftarrow H(M, v)$ and $\sigma \leftarrow h^a$. The MGDH-signature is $\sigma \in G$.

Verify : Given a message $M \in \mathbb{M}$, a signature $\sigma \in G$ and a public key $v \in G$, compute $h \leftarrow H(M, v)$ and output accept if $e(g, \sigma) = e(v, h)$, reject otherwise.

MBGLS-signature scheme.

KeyGen, Sign, Verify : Same as in the MGDH-signature scheme.

AdjKeyGen : Returns the public key $v' = g^b \in G$ and private key $b \in \mathbb{Z}_p$ of the adjudicator. Same algorithm as **KeyGen**.

VESigCreate : Input is the message $M \in \mathbb{M}$, the user secret key $a \in \mathbb{Z}_p$ and adjudicator public key $v' \in G$. Output is the MBGLS-signature $(\mu, \omega) \in G \times G$ which is computed as follows. Let $h \leftarrow H(M, v)$ and $\sigma \leftarrow h^a$. Select random $s \in \mathbb{Z}_p$ and compute $\mu \leftarrow g^s$ and $\omega \leftarrow \sigma v'^s$. The MBGLS-signature is $(\mu, \omega) \in G \times G$.

VESigVerify : Input is the message M, data $(\mu, \omega) \in G \times G$, the user public key $v \in G$ and the adjudicator public key $v' \in G$. Output is accept if (μ, ω) is a valid MBGLS-signature on M under v and v', that is if $e(g, \omega) = e(v, h) \cdot e(v', \mu)$ with $h = H(M, v)$. Otherwise output is reject.

Adjudicate : Input is the message $M \in \mathbb{M}$, data $(\mu, \omega) \in G \times G$, the user public key $v \in G$, and the adjudicator public key $v' \in G$ and private key $b \in \mathbb{Z}_p$. Output is reject, if **VESigVerify** rejects $M, (\mu, \omega), v, v'$. Otherwise output is $\sigma \leftarrow \omega / \mu^b$ which is the MGDH-signature on M under v.

We remark that a MBGLS-signature is just a plain ElGamal-encryption (see for example [4]) of the MGDH-signature. Furthermore, the original GDH-signature and BGLS-signature schemes are obtained by replacing H with a function $H' : \mathbb{M} \times G \to G$ which is a full domain hash function in its first argument and constant in its second argument.

These observations together with the homomorphic property of plain ElGamal-encryption and the GDH-signature scheme lead to the following attack. On input of the public keys $v, v' \in G$ of the user and the adjudicator the adversary $\mathcal{E}$ computes a message $M \in \mathbb{M}$ and a valid GDH-signature $\sigma \in G$ on the message M under the public key v as follows. He chooses an arbitrary message $M \in \mathbb{M}$ and uses the signature oracle to obtain a BGLS-signature (μ, ω) on M for the user public key v and the adjudicator public key v'. He then chooses a random number $r \in \mathbb{Z}_p$ coprime to p and computes (μ^r, ω^r). With $h = H'(M, v) = H'(M, v^r)$ we see that $e(g, \omega^r) = e(g, \omega)^r = e(v, h)^r \cdot e(v', \mu)^r = e(v^r, h) \cdot e(v', \mu^r)$, hence (μ^r, ω^r) is a valid

BGLS-signature on the message M for a (different) user with public key v^r and the same adjudicator. The adversary then uses the adjudication oracle to obtain the GDH-signature $\sigma_r \in G$ on input of (μ^r, ω^r), the user public key v^r and the adjudicator public key v. If $v = g^a$ then a is the secret key corresponding to v and ra is the secret key corresponding to v^r. Clearly $\sigma_r = h^{ra}$ because it is a valid GDH-signature on M under v^r. The adversary outputs $\sigma \leftarrow \sigma_r^{1/r}$, a valid GDH-signature on M under v since $\sigma = h^a$. Altogether the adversary $\mathcal{E}$ needs to perform a few calculations in G and issues one query to the signature and adjudication oracle respectively. Clearly, its success probability is 1.

3. Security

An extracting adversary $\mathcal{E}$ is an algorithm which on input of a user public key v and an adjudicator public key v' outputs a message M and a MGDH-signature σ on M under v. We allow $\mathcal{E}$ access to a MBGLS-signature creation oracle S and adjudication oracle A subject to the condition that $\mathcal{E}$ does not query S or A on an adjudicator public key different from v' and that $\mathcal{E}$ does not return M, σ which have been computed by A on input of the user key v.

Theorem 1. *If there is an extracting adversary $\mathcal{E}$ against the MBGLS-signature scheme which runs in time at most $t_{\mathcal{E}}$, makes q_H, q_S and q_A queries to the hash, MBGLS-signature creation and adjudication oracles respectively and succeeds with probability at least $\epsilon_{\mathcal{E}}$ in the random oracle model then there is an algorithm B which computes g^{an} and g^{bm} from $g, g^a, g^b, g^n, g^m, g^{an+bm} \in G$ in time at most $t_B \leq t_{\mathcal{E}} + O(\log(p)(q_H + q_S + q_A))$ with probability at least $\epsilon_B \geq 1/(2eq_A)\epsilon_{\mathcal{E}}$.*

Proof. The general proof idea is fairly similar to that in [1], using a technique from [3]. In the following let $g, g^a, g^b, g^n, g^m, g^{an+bm} \in G$ be public values for randomly chosen g^a, g^b, g^n, g^m. These values will be given as input to Algorithm B which outputs the values g^{an}, g^{bm}. Computing these values is the (hard) underlying problem which makes the scheme secure against extraction.

Fix some $\zeta \in [0, 1]$. The hash oracle H, the MBGLS-signature creation oracle S and adjudicator oracle A are simulated in the random oracle model as follows.

Algorithm c.

Input : $M \in \mathbb{M}$, $v \in G$.
Output : The value $c(M, v) \in \{0, 1\}$.

Step 1 : If $c(M, v)$ is defined, return it.
Step 2 : If $v \neq g^a$ define $c(M, v) \leftarrow 0$ and output 0.
Step 3 : Otherwise, choose $c \in \{0, 1\}$ such that $\Pr(c = 1) = \zeta$. Define $c(M, v) \leftarrow c$ and output c.

Simulator for H.

Input : $M \in \mathbb{M}$, $v \in G$.
Output : The hash value $H(M, v) \in G$.

Step 1 : If $H(M, v)$ is defined, return it.
Step 2 : Choose random $r \in \mathbb{Z}_p$.
Step 3 : If $c(M, v) = 1$ compute $h \leftarrow g^n g^r$. If $c(M, v) = 0$ compute $h \leftarrow g^r$.
Step 4 : Define $H(M, v) \leftarrow h$ and $R(M, v) \leftarrow r$. Output h.

The hash values of H are uniformly and independently distributed in G. Hence the simulator properly simulates the random oracle H.

Simulator for S.

Input : Public keys $v, v' \in G$ of the user and the adjudicator. A message $M \in \mathbb{M}$.

Output : A MBGLS-signature (μ, ω) on M under the user key v and adjudicator key v'.

Step 1 : If $v' \neq g^b$ then the simulation fails.
Step 2 : Choose a random $s \in \mathbb{Z}_p$ and compute $r \leftarrow R(M, v)$ by a call to H.
Step 3 : If $c(M, v) = 0$ then compute $\sigma \leftarrow v^r$, $\mu \leftarrow g^s$ and $\omega \leftarrow \sigma {v'}^s$.
Step 4 : If $c(M, v) = 1$ then compute $\mu \leftarrow g^m g^s$ and $\omega \leftarrow g^{an+bm} v^r {v'}^s$.
Step 5 : Output (μ, ω).

Observing that $c(M, v) = 1$ implies $v = g^a$ it is straightforward to check that the simulator for S creates valid MBGLS-signatures on M under v and v'. Moreover, since μ in step 3 or step 4 is in effect uniformly and randomly chosen from G, the simulator of S and the oracle S itself are indistinguishable.

Simulator for A.

Input : Public keys $v, v' \in G$ of the user and the adjudicator. A message $M \in \mathbb{M}$ and a purportedly MBGLS-signature (μ, ω) on M under v and v'.
Output : The MGDH-signature σ, if (μ, ω) is indeed valid, reject otherwise. The simulation may also fail.

Step 1 : Compute $h \leftarrow H(M, v)$. If $e(g, \omega) \neq e(v, h)e(v', \mu)$ then output reject.
Step 2 : If $c(M, v) = 1$ then the simulation fails.
Step 3 : Let $r \leftarrow R(M, v)$ and compute $\sigma \leftarrow v^r$. Output σ.

Again it is easy to check that the simulator of A returns valid MGDH-signatures on M under v. Its output coincides with the output of the oracle A if $c(M, v) = 0$.

With these simulators we can construct an algorithm B using the extracting adversary $\mathcal{E}$ to solve the above computational problem. We may assume that $\mathcal{E}$ makes a query to H prior to its termination to ensure that the returned signature is valid, or to indicate failure otherwise.

Algorithm B.

Input : Elements $g, g^a, g^b, g^n, g^m, g^{an+bm} \in G$.
Output : Elements $g^{an}, g^{bm} \in G$ or failure.

Step 1 : Run the adversary $\mathcal{E}$ on input of g, $v = g^a$, $v' = g^b$.
Step 2 : The adversary may call the oracle simulators adaptively with possibly different user keys v but always the same adjudicator key $v' = g^b$. If A or $\mathcal{E}$ fail at some stage, algorithm B stops and outputs failure.
Step 3 : The adversary $\mathcal{E}$ returns M, σ.
Step 4 : If $c(M, v) = 0$ then output failure.
Step 5 : Let $r \leftarrow R(M, v)$ and compute $z \leftarrow \sigma / v^r$. Output z and g^{an+bm}/z.

We prove that Algorithm B is sound. The value of $R(M, v)$ in step 5 is defined because of the assumption about $\mathcal{E}$. Assume z is returned from step 5. Then $v = g^a$ because $c(M, v) = 1$, and $e(g, \sigma) = e(v, h)$ with $h = H(M, v)$ because of the definition of $\mathcal{E}$. Moreover, $h = g^n g^r$ with $r = R(M, v)$ because $c(M, v) = 1$. Then $z = \sigma / v^r = h^a / v^r = g^{an}$, so z returned from step 5 is correct.

Next we need to estimate the success probability of Algorithm B. The simulations of H, S and A are indistinguishable from the oracles unless the simulation of A fails when $c(M, v) = 1$. The simulator of S cannot fail because of the requirement that all queries satisfy $v' = g^b$. Since the whole algorithm B is stopped in the case of failure, $\mathcal{E}$ does not know that it is run in a simulation and in particular learns nothing about the values of $c(M, v)$. Thus the probability of no failure during q_A queries to the simulator of A is $(1 - \zeta)^{q_A}$, and $c(M, v) \neq 0$ in step 4 happens with conditional probability ζ because $\mathcal{E}$ has not called the simulator A with M

and $v = g^a$ as input. The overall success probability ϵ_B of B is therefore at least $\zeta(1-\zeta)^{q_A}\epsilon_{\mathcal{E}}$. This is maximized for $\zeta = 1/(q_A + 1)$ and we obtain as in [1] that $\epsilon_B = \epsilon_{\mathcal{E}}$ for $q_A = 1$ and $\epsilon_B \geq 1/(2eq_A)\epsilon_{\mathcal{E}}$ for $q_A > 1$.

The running time t_B of B contains the running time $t_{\mathcal{E}}$ of $\mathcal{E}$ plus the additional time for the simulators which is roughly $O(\log(p)(q_H + q_S + q_A))$ with small constants. This proves Theorem 1. □

A forging adversary $\mathcal{F}$ against the MBGLS-signature scheme is an algorithm which on input of a user public key v and an adjudicator public key v' outputs a message M and a MBGLS-signature (μ, ω) on M under v and v'. We allow $\mathcal{F}$ access to a MBGLS-signature creation oracle S and adjudication oracle A as above subject to the condition that $\mathcal{F}$ does not query S or A on an adjudicator public key different from v' and that $\mathcal{F}$ does not query S on M, v either.

A forging adversary C against the MGDH-signature scheme is an algorithm which on input of a user public key v computes a message M and a MGDH-signature σ on M under v. We allow C access to a MGDH-signature oracle subject to the condition that C does not query the oracle on M, v.

For such forging adversaries we have the following theorem similar to that in [1]. Note that the security model here is again more general since $\mathcal{F}$ is allowed to query S on user public keys different from v. The proof is nevertheless essentially the same as in [1] so we omit it here.

Theorem 2. *If there is a forging adversary $\mathcal{F}$ against the MBGLS-signature scheme which runs in time at most $t_{\mathcal{F}}$, makes q_H, q_S and q_A queries to the hash, MBGLS-signature creation and adjudication oracles respectively and succeeds with probability at least $\epsilon_{\mathcal{F}}$ in the random oracle model then there is a forging adversary C against the MGDH-signature scheme which runs in time at most $t_C \leq t_{\mathcal{F}} + O(\log(p)(q_S + q_A))$, makes at most q_H and q_S queries to the hash and MGDH-signature oracle respectively and succeeds with probability at least $\epsilon_C \geq \epsilon_{\mathcal{F}}$ in the random oracle model.*

We remark that Theorem 1 also implies the security of the MGDH-signature scheme, which we have not discussed yet. The combination of the oracles S and A yields a MGDH-signature oracle for the MGDH-signature scheme, and it is admissible to query such a signature oracle on signatures for other public user keys than the target key, because this was allowed in the case of S and A. As a result any forging adversary against the MGDH-signature scheme is trivially an extracting adversary $\mathcal{E}$ against the MBGLS-signature scheme. The GDH-signature scheme using H' is clearly not secure under such circumstances, so Theorem 1 provides an extended security model and proof for the MGDH-signature scheme using H. A more direct proof would relate the security of the scheme to the computational Diffie-Hellman problem in G, as in [2].

Acknowledgements

The author thanks Nigel Smart and the referees for helpful comments.

References

1. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003*, LNCS 2656, pages 416–432, Warsaw, Poland, 2003. Springer-Verlag, Berlin-Heidelberg-New York.
2. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In C. Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001*, LNCS 2248, pages 514–532, Gold Coast, Australia, 2001. Springer-Verlag, Berlin-Heidelberg-New York.

3. J.-S. Coron. On the exact security of full domain hash. In M. Bellare, editor, *Advances in Cryptology - CRYPTO 2000*, LNCS 1880, pages 229–235, Santa Barbara, 2000. Springer-Verlag, Berlin-Heidelberg-New York.
4. N. P. Smart. *Cryptography, An Introduction*. McGraw-Hill, New York, 2002.

Technical University of Berlin, Faculty II – Institute of Mathematics, Secr. MA8-1, Strasse des 17. Juni 136, 10623 Berlin, Germany

E-mail address: `hess@math.tu-berlin.de`