Pairing-Based Identification Schemes

David Freeman
Information Theory Research
HP Laboratories Palo Alto
HPL-2005-154
August 24, 2005*

public-key cryptography, identification, zero-knowledge, pairings

We present several different identification schemes that make use of bilinear pairings. Each of the schemes is more efficient and/or more secure than any known pairing-based identification scheme.

* Internal Accession Date Only
Approved for External Publication
© Copyright 2005 Hewlett-Packard Development Company, L.P.

PAIRING-BASED IDENTIFICATION SCHEMES

DAVID FREEMAN

Abstract. We present several different identification schemes that make use of bilinear pairings. Each of the schemes is more efficient and/or more secure than any known pairing-based identification scheme.

1. Introduction

An *identification scheme* is a protocol whereby Peggy the Prover proves to Victor the Verifier that she is indeed who she says she is. In practice, Peggy's identity is encoded in a private key a and a public key y. The protocol takes the form of Peggy proving to Victor that she has knowledge of the private key a. For example, the private key might be a and the public key $y = x^a \pmod{p}$, where a and x are integers and p is a prime number, and Peggy proves her identity by demonstrating that she knows the discrete logarithm of y. Now, Peggy could simply tell Victor a, and Victor could verify that a is the correct private key, but then Victor could impersonate Peggy to a third party. A viable identification scheme must prevent this from happening; we require that Victor can't impersonate Peggy even if she proves her identity to him polynomially many times. Because of this property, an identification scheme is also called a *zero-knowledge proof of identity*.

Feige, Fiat, and Shamir [7] introduced the first identification scheme in 1988, based on the difficulty of inverting RSA. Soon thereafter, Guillou and Quisquater [9] and Schnorr [15] introduced their own identification schemes, based on RSA and discrete logarithms respectively. These two schemes are still amongst the most efficient and well-studied identification schemes, though their security has never been reduced to a standard computational problem such as factoring or discrete logarithms.

Identification schemes are closely related to signature schemes. For example, one way for Peggy to prove her identity to Victor is for him to ask her to digitally sign a message of his choice; if the signature is hard to forge, then a valid signature will constitute an acceptable proof of identity. On the other hand, any of the standard identification schemes can be converted to a signature scheme by replacing Victor with a one-way hash function.

Recent years have brought a host of signature schemes that make use of bilinear pairings. The first of these was the short signature scheme of Boneh, Lynn, and Shacham in 2001 [6]. This was quickly followed by a spate of pairing-based schemes designed for various applications that have certain advantages over tradition RSA or discrete-log based signatures: group signatures, ring signatures, aggregate signatures, multisignatures, threshold signatures, and more. Given this plethora of pairing protocols and the close relationship between identification schemes and signatures, it is natural to ask whether there might be a pairing-based identification scheme that has some advantage over the GQ or Schnorr schemes. The first step in

this direction was taken by Kim and Kim in 2002 [11]. Their scheme was later shown to be flawed; others have since proposed pairing-based identification schemes [10], [16], [17], but none has given a convincing proof of security with a tight reduction.

In this paper, we present four new identification schemes based on pairings, and prove their security given certain computational assumptions. We begin in Section 2 by giving a formal definition of security for identification schemes, reviewing some standard computational assumptions, and describing the bilinear pairings useful for cryptography. In Section 3, we describe a basic scheme based on the Boneh-Lynn-Shacham signatures and prove its security in the random oracle model under the Computational Diffie-Hellman assumption. Since the random oracle model is somewhat unsatisfactory for proving security of identification schemes, in Section 4 we modify the scheme so that it does not require the use of hash functions. To prove security of this new scheme we introduce a new assumption, called the "one-more-Computational Diffie-Hellman" assumption, which is related to several existing assumptions in the literature.

In Section 5 we take another tack, adapting a signature scheme that does not make use of random oracles for its proof of security. The proof of security of this scheme relies of the "Strong Diffie-Hellman assumption," an analogue of the "Strong RSA assumption" used to prove security of RSA signatures. Finally, in Section 6 we introduce a scheme whose proof of security relies on the assumption that the pairing used is a one-way function. We show that this assumption is weaker than any other made in this paper, and thus this scheme is the most secure of our new schemes.

Having presented our four new schemes and proved their security, in Section 7 we describe two other pairing-based identification schemes in the literature, and in Section 8 we examine the bandwidth and computational requirements of all six schemes. We conclude that each of our protocols is the preferred identification scheme in some context, for either efficiency or security reasons.

1.1. **Acknowledgments.** Research for this paper was conducted during a summer internship at HP Labs, Palo Alto. I thank Vinay Deolalikar for suggesting this problem and for providing advice and support along the way. I also thank Gadiel Seroussi for bringing me to HP and for supporting my research.

2. Preliminaries

2.1. **Identification schemes.** An *identification scheme* is a protocol whereby Peggy the Prover proves to Victor the Verifier that she is indeed who she says she is, in such a way that after interacting with Peggy, Victor cannot turn around and impersonate Peggy to a third party. The protocol must thus be a *zero-knowledge proof of identity*: Victor must be convinced by Peggy's proof, but he cannot gain any knowledge that allows him to impersonate Peggy.

More formally, an identification scheme consists of a key-generation algorithm $\mathcal{G}$ that creates a valid set of keys a (Peggy's private key) and p_a (Peggy's public key), and an interactive protocol $(\mathcal{P}, \mathcal{V})$ that takes as input the public and private keys, and outputs 1 (accept) or 0 (reject). We require that if both users follow the protocol and use a valid public/private key pair, the protocol always outputs 1 (accepts). We also require that any cheating prover $\mathcal{A}$ that does not know Peggy's private key cannot interact with an honest verifier $\mathcal{V}$ and give output 1. Furthermore, we require that a cheating verifier $\mathcal{B}$ cannot interact with Peggy, pass what he learns

on to the cheating prover $\mathcal{A}$, and have $\mathcal{A}$ interact with an honest verifier $\mathcal{V}$ and output 1. We note that the first security condition is a special case of the second, in which $\mathcal{B}$ outputs nothing. This leads us to the following definition:

Definition 2.1 (cf. [8, Definition 4.7.8])**.** A (t, q, ϵ)-*identification scheme* is a triple $(\mathcal{G}, \mathcal{P}, \mathcal{V})$, where $\mathcal{G}$ is a probabilistic polynomial-time algorithm and $(\mathcal{P}, \mathcal{V})$ is a pair of probabilistic interactive machines running in time at most t, satisfying the following conditions:

- *Viability:* For any $\alpha \in \{0,1\}^n$, let $\mathcal{G}(\alpha) = (a_\alpha, p_\alpha)$. Then
$$\Pr\left[\langle \mathcal{P}(a_\alpha, p_\alpha), \mathcal{V}(p_\alpha)\rangle = 1\right] = 1.$$
- *Security:* For any $\alpha \in \{0,1\}^n$, let $\mathcal{G}(\alpha) = (a_\alpha, p_\alpha)$. For any probabilistic interactive machine $\mathcal{B}$ running in time at most t, let T_α be a random variable describing the output of $\mathcal{B}(p_\alpha)$ after interacting with $\mathcal{P}(a_\alpha, p_\alpha)$ q times. Then for any probabilistic interactive machine $\mathcal{A}$ running in time at most t,
$$\Pr\left[\langle \mathcal{A}(p_\alpha, T_\alpha), \mathcal{V}(p_\alpha)\rangle = 1\right] < \epsilon.$$

Note that the security condition implies that a third party, Malice, cannot impersonate Peggy to Victor, *provided that Malice cannot interact concurrently with Peggy and Victor*. Indeed, if Malice can interact concurrently with both, she may impersonate Peggy by referring Victor's queries to Peggy and relaying the response back to Victor.

2.2. **Computational assumptions.** All of public-key cryptography relies on certain computational assumptions for its security; e.g. that factoring is difficult. The assumptions relevant to our identification schemes are of the *Diffie-Hellman* type, named after the two creators of public-key cryptography. The original Diffie-Hellman problem is known as the Computational Diffie-Hellman (CDH) problem.

Definition 2.2. Let $\mathbb{G}$ be a cyclic group of order n, let $g \in \mathbb{G}$, and let $a, b \in \mathbb{Z}_n$. The *Computational Diffie-Hellman problem* in $\mathbb{G}$ is as follows: Given $\{g, g^a, g^b\}$, compute g^{ab}.

The (t, ϵ)-*Computational Diffie-Hellman assumption* holds in $\mathbb{G}$ if there is no algorithm $\mathcal{A} : \mathbb{G}^3 \to \mathbb{G}$ running in time at most t such that
$$\Pr\left[\mathcal{A}(g, g^a, g^b) = g^{ab}\right] \geq \epsilon,$$
where the probability is taken over all possible choices of (g, a, b).

The CDH problem arises in the context of Diffie-Hellman key exchange, in which an eavesdropper sees the triple $\{g, g^a, g^b\}$ and wants to compute the secret key g^{ab}. In many cases, even partial information about the secret key g^{ab} may compromise the system. If we want to assume that no such partial information can be gained, we must make an even stronger assumption, known as the Decision Diffie-Hellman (DDH) assumption. The DDH problem asks whether a given quadruple of elements of $\mathbb{G}$ is a solution to the CDH problem; the DDH assumption is that there is no polynomial-time algorithm that correctly solves the DDH problem with non-negligible probability.

Definition 2.3. Let $\mathbb{G}$ be a cyclic group of order n, let $g \in \mathbb{G}$, and let $a, b, c \in \mathbb{Z}_n$. The *Decision Diffie-Hellman problem* in $\mathbb{G}$ is as follows: Given $\{g, g^a, g^b, g^c\}$, determine whether $g^{ab} = g^c$.

The (t, ϵ)-*Decision Diffie-Hellman assumption* holds in $\mathbb{G}$ if there is no algorithm $\mathcal{A} : \mathbb{G}^4 \to \{0, 1\}$ running in time at most t such that

$$\left|\Pr\left[\mathcal{A}(g, g^a, g^b, g^{ab}) = 1\right] - \Pr\left[\mathcal{A}(g, g^a, g^b, g^c) = 1\right]\right| \geq \epsilon,$$

where the probabilities are taken over all possible choices of (g, a, b, c).

Remark 2.4. The *discrete logarithm problem* in $\mathbb{G}$ is to compute a from $\{g, g^a\}$. We note that if the discrete logarithm problem is easy, then the CDH and DDH problems are both easy, for we can compute a from $\{g, g^a\}$ and then compute $g^{ab} = (g^b)^a$. Furthermore, a solution to the CDH problem gives a solution to the DDH problem. It is not known whether solutions to the CDH or DDH problems can be used to solve the discrete logarithm problem.

2.3. **Bilinear maps and pairings.** Joux and Nguyen [12] showed that an efficiently computable bilinear map on $\mathbb{G}$ gives an algorithm for solving the Decision Diffie-Hellman problem on $\mathbb{G}$. Boneh, Lynn, and Shacham [6] make use of this property in their signature algorithm by using the pairing to verify that the signature creates a valid Diffie-Hellman tuple. Our identification schemes will use pairings in their verification procedures in a similar manner.

The following definition gives the conditions necessary for a bilinear map to be useful for cryptographic purposes. To simplify our exposition, we will consider only the case where both arguments of the pairing are in the same group; for the more general case, see [6].

Definition 2.5. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be cyclic groups of prime order p. A map $e \colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is a *cryptographic pairing* if the following conditions hold:

- *Bilinearity:* for all $x, y \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}$, $e(x^a, y^b) = e(x, y)^{ab}$.
- *Non-degeneracy:* if g is a generator of $\mathbb{G}_1$, then $e(g, g)$ is a generator of $\mathbb{G}_2$.

Remark 2.6. A cryptographic pairing e can be used to solve the DDH problem on $\mathbb{G}_1$ as follows: given $\{g, g^a, g^b, g^c\}$, where g is a generator of $\mathbb{G}_1$ and a, b, c are integers, compute $h_1 = e(g, g^c)$ and $h_2 = e(g^a, g^b)$. Then $h_1 = h_2$ in $\mathbb{G}_2$ if and only if $c = ab \pmod{p}$. If the CDH problem in $\mathbb{G}_1$ is hard and the DDH problem is easy (e.g. if there is a cryptographic pairing on $\mathbb{G}_1$), $\mathbb{G}_1$ is known as a *Gap Diffie-Hellman group*. The *Gap Diffie-Hellman problem* is to solve the CDH problem given an oracle for the DDH problem.

The only known examples of cryptographic pairings are derived from the Weil and Tate pairings on elliptic curves over finite fields. The study of these groups is deep and beautiful and is of great interest to current researchers. However, in describing our protocols we will not take into account the structure of the groups involved in the pairing; rather, we will make certain computational assumptions about the group and use the pairing as a "black box." For further information on elliptic curves, see [3] or [4].

3. Identification scheme based on BLS signatures

A particularly simple method of building identification schemes is to use a digital signature algorithm. Victor the Verifier sends a random message to Peggy the Prover, Peggy signs the message with her secret key, and Victor verifies that the signature is correct. If the signature scheme is secure against forgery, the cheating prover has a negligible chance of creating a valid signature on a random message

given him by an honest verifier, no matter how many signatures he has obtained from the honest prover.

3.1. **The BLS signature scheme.** Boneh, Lynn, and Shacham [6] were the first to devise a digital signature scheme based on pairings. The algorithm provides for signatures of half the length of a DSS signature with an equivalent level of security, and as such it makes for a particularly efficient identification scheme in terms of bandwidth. We now describe the BLS signature scheme and the corresponding identification scheme. We describe the scheme in terms of a pairing, but the scheme is in fact valid in any group in which the Decision Diffie-Hellman problem is easy and the Computational Diffie-Hellman problem is hard; such a group is called a *Gap Diffie-Hellman group.*

Protocol 3.1 ([6])**.** Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing. Let g be a generator of $\mathbb{G}_1$. Let $H : \{0,1\}^* \to \mathbb{G}_1$ be a full-domain hash function.

Key generation: Pick random $x \leftarrow \mathbb{Z}_p$, and compute $v \leftarrow g^x$. The public key is v, and the secret key is x.

Signing: Given a secret key $x \in \mathbb{Z}_p$ and a message $M \in \{0,1\}^*$, compute $h \leftarrow H(M)$ and $\sigma \leftarrow h^x$. The signature is $\sigma \in \mathbb{G}$.

Verification: Given a public key $v \in \mathbb{G}$, a message $M \in \{0,1\}^*$, and a signature $\sigma \in \mathbb{G}$, compute $e(g,\sigma)$ and $e(v,h)$. If the two are equal, output `valid`; if not, output `invalid`.

Boneh, Lynn, and Shacham prove the security of their scheme using the following game between a challenger and an adversary $\mathcal{A}$.

Setup: The challenger runs algorithm $KeyGen$ to optain a public key PK and a private key SK. The adversary $\mathcal{A}$ is given PK.

Queries: Proceeding adaptively, $\mathcal{A}$ requests signatures with PK on at most q_S messages of his choice, $M_1, \ldots, M_{q_s} \in \{0,1\}^*$. The challenger responds to each query with a signature $\sigma_i = Sign(SK, M_i)$.

Output: Eventually, $\mathcal{A}$ outputs a pair (M, σ) and wins the game if (1) M is not any of $M_1, \ldots, M_{q_S}$, and (2) $Verify(PK, M, \sigma) = \texttt{valid}$.

The advantage of $\mathcal{A}$, denoted $\mathrm{Adv}(\mathcal{A})$, is the probability that $\mathcal{A}$ wins the above game, taken over the coin tosses of $KeyGen$ and of $\mathcal{A}$ itself. We are now ready to define the security of a signature scheme.

Definition 3.2 ([6, Definition 3.1])**.** A forger $\mathcal{A}$ (t, q_S, q_H, ϵ)*-breaks* a signature scheme if $\mathcal{A}$ runs in time at most t, makes at most q_S signature queries and at most q_H queries to a hash function, and $\mathrm{Adv}(\mathcal{A}) > \epsilon$. A signature scheme is (t, q_S, q_H, ϵ)*-existentially unforgeable under adaptive chosen-message attack* if no forger (t, q_S, q_H, ϵ)-breaks it.

Theorem 3.3 ([6, Theorem 3.2])**.** *Suppose the (t', ϵ')-Computational Diffie-Hellman assumption holds in $\mathbb{G}_1$. Then the signature scheme defined in Protocol 3.1 is (t, q_S, q_H, ϵ)-secure against existential forgery under an adaptive chosen-message attack (in the random oracle model) for all t and ϵ satisfying*

$$\epsilon \geq e(q_S + 1) \cdot \epsilon' \quad \textit{and} \quad t \leq t' - c(q_H + 2q_S),$$

where c is a constant that depends on $\mathbb{G}_1$, and e is the base of the natural logarithm.

3.2. **The BLS Identification scheme.** We now show how the BLS signature scheme can be adapted nearly verbatim to serve an an identification scheme. We describe the scheme as an interactive protocol between Peggy the prover and Victor the verifier.

Protocol 3.4. Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing. Let g be a generator of $\mathbb{G}_1$. Let $H : \{0,1\}^* \to \mathbb{G}_1$ be a full-domain hash function.

Key generation: Pick random $x \leftarrow \mathbb{Z}_p$, and compute $v \leftarrow g^x$. The public key is v, and Peggy's secret key is x. Let n be a positive integer.

Interactive protocol:

(1) Victor sends Peggy a random $M \in \{0,1\}^n$.
(2) Peggy computes $h = H(M)$ and sends Victor $\sigma = h^x$.
(3) Victor computes $e(g,\sigma)$ and $e(v,h)$. If the two are equal he outputs 1 (accept); else he outputs 0 (reject).

Since our signature makes use of a hash function and the proof of security is in the random oracle model, we must add another parameter to our description of security of identification schemes. We say that a scheme using a hash function is a (t,q,r,ϵ)-identification scheme if the conditions of Definition 2.1 hold, with the additional requirement that $(\mathcal{A},\mathcal{B})$ make no more than r queries to the hash function.

Theorem 3.5. *If the BLS signature scheme of Protocol 3.1 is (t',q,r,ϵ')-secure against existential forgery under an adaptive chosen-message attack, then the scheme of Protocol 3.4 is a (t,q,r,ϵ) identification scheme, provided that*

$$\epsilon \geq \left(\frac{2^n}{2^n - q}\right) \cdot \epsilon' \quad \textit{and} \quad t \leq t' - c$$

for some constant c depending on the groups and pairing used.

Proof. If Peggy and Victor follow the protocol, then Protocol 3.4 satisfies the viability condition of Definition 2.1, since

$$e(g,\sigma) = e(g,h^x) = e(g,h)^x = e(g^x,h) = e(v,h)$$

by bilinearity of e.

Now suppose $(\mathcal{A},\mathcal{B})$ is a pair of algorithms that (t,q,r,ϵ)-breaks the scheme (in the sense of Definition 2.1) for a given public/private-key pair. Define an attacker $\mathcal{C}$ on the BLS scheme with the same public and private keys, as follows:

(1) For each M_i that the cheating verifier $\mathcal{B}$ sends to the honest prover $\mathcal{P}$, have $\mathcal{C}$ request a signature on M_i. Run $\mathcal{B}$ on the output.
(2) Simulate the honest verifier $\mathcal{V}$ by choosing a random M and sending M as input to the cheating prover $\mathcal{A}$.
(3) Output the pair (M,τ), where $\tau \in \mathbb{G}_1$ is the element that the cheating prover $\mathcal{A}$ sends to $\mathcal{V}$.

If $(\mathcal{A},\mathcal{V})$ outputs 1, then the output of algorithm $\mathcal{C}$ is a valid BLS message-signature pair. Thus if M is distinct from all of the queries M_i, then (M,τ) is a valid forgery. Since the probability of $(\mathcal{A},\mathcal{B})$ simulating the prover $\mathcal{P}$ is at least ϵ and the probability that M is equal to one of the M_i is $q/2^n$, the probability of forging a signature is at least $(1 - q/2^n)\cdot\epsilon$. We thus have broken the BLS scheme with an

attacker that runs in time $t+c$ for some constant c. The attacker makes q signature queries and h hash queries. □

Corollary 3.6. *Suppose the (t', ϵ') Computational Diffie-Hellman assumption holds in $\mathbb{G}_1$. Then Protocol 3.4 defines a (t, q_S, q_H, ϵ)-identification scheme for all t and ϵ satisfying*

$$\epsilon \geq \frac{2^n e(q_S+1)}{2^n - q} \cdot \epsilon' \quad \textit{and} \quad t \leq t' - c(q_H + 2q_S),$$

where c is a constant that depends on $\mathbb{G}_1$, and e is the base of the natural logarithm.

4. Identification schemes based on the one-more-CDH assumption

Protocol 3.4, an identification scheme derived directly from the BLS signature scheme, is unsatisfactory in several ways. While the communication overhead is minimal (one element of $\mathbb{G}_1$ and one random string which needs only to be large enough to avoid hash collisions), the prover and verifier must both compute the hash of the parameter M, which adds computational time. In addition, the proof of security is in the random oracle model, which requires us to introduce another security parameter and to assume that the hash function H acts as a random function. Recent attacks on SHA-1 and other hash functions have called into question the credibility of such an assumption, so we would ideally like our identification schemes to be hash-free.

Our first attempt at constructing a pairing-based identification scheme that does not use hash functions is to simply recreate the scheme based on BLS signatures, but do away with the hash function.

Protocol 4.1. Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing. Let g be a generator of $\mathbb{G}_1$.

Key generation: Pick random $x \leftarrow \mathbb{Z}_p$, and compute $v \leftarrow g^x$. The public key is v, and Peggy's secret key is x.

Interactive protocol:

(1) Victor sends Peggy a random challenge $h \in \mathbb{G}_1$.
(2) Peggy computes sends Victor $\sigma = h^x$.
(3) Victor computes $e(g, \sigma)$ and $e(v, h)$. If the two are equal he outputs 1 (accept); else he outputs 0 (reject).

We can think of Protocol 4.1 as Protocol 3.4 where instead of sending a random message M in step (1), Victor sends the hash h of the message M; if the hash is random, then h is just a random element of $\mathbb{G}_1$. With this modification, the reduction of the scheme to the Computational Diffie-Hellman assumption in $\mathbb{G}_1$ breaks down, as that reduction requires that Peggy can't compute M from h. The security of this scheme thus requires a different assumption.

To determine what kind of assumption we need to make, we examine the behavior of an attacker. The cheating verifier $\mathcal{A}$ interacts with the honest prover $\mathcal{P}$ by sending q queries of her choice $h_1, \ldots, h_q$ and receiving the 'signature' of each message, $h_1^x, \ldots, h_q^x$. The cheating prover $\mathcal{B}$ must then take a random query h and return h^x. (Note that by the bilinearity of the pairing e, h^x is the only element that $\mathcal{B}$ can send in step (2) that will cause an honest verifier to accept.) If $q = 0$, then this is the Computational Diffie-Hellman problem: compute h^x from $\{g, g^x, h\}$. If $q > 0$, we are asking for the solution to a CDH problem given the solution to q related CDH problems. We formalize this notion in the following definition.

Definition 4.2. Let $\mathbb{G}$ be a finite cyclic group. Let A be a randomized algorithm that takes input $g, g^a \in \mathbb{G}$ and has access to two oracles. The first is a CDH oracle $CDH_{g,g^a}(\cdot)$, which on input $h \in \mathbb{G}$ returns $h^a \in \mathbb{G}$. The second is a challenge oracle $C()$ that, when invoked, returns a random challenge point $r \in \mathbb{G}$. Furthermore, we require that $\mathcal{A}$ cannot invoke its CDH oracle after it has invoked the challenge oracle. We say that algorithm $\mathcal{A}$ has advantage ϵ in solving the *one-more-CDH problem* in $\mathbb{G}$ if

$$\Pr\left[\mathcal{A}(g, g^a, r \leftarrow C()) = r^a\right] \geq \epsilon,$$

where the probability is taken over the choices g and g^a input to $\mathcal{A}$ and the r output from $C()$.

We say the (t, q, ϵ)-*one-more-CDH assumption* holds in $\mathbb{G}$ if there is no algorithm $\mathcal{A}$ that runs in time at most t, makes at most q queries to its CDH oracle, and has advantage at least ϵ in solving the one-more-CDH problem in $\mathbb{G}$.

Definition 4.2, while it has not appeared previously in the literature, is closely related to the "one-more-RSA-inversion" and "one-more-discrete-logarithm" problems defined by Bellare, et al. [1]. Bellare and Palacio [2] use these assumptions to prove the security of the well-known Guillou-Quisquater and Schnorr identification schemes, so it seems eminently reasonable that we should have to use a similar assumption in proving the security of our scheme.

We now prove the security of Protocol 4.1 based on the one-more-CDH assumption.

Theorem 4.3. *Suppose the (t, q, ϵ)-one-more-CDH assumption holds in $\mathbb{G}$. Then Protocol 4.1 is a $(t - O(1), q, \epsilon)$-identification scheme.*

Proof. Let (g, g^x) be the public parameters for Protocol 4.1. Suppose $(\mathcal{A}, \mathcal{B})$ is an attack that (t, q, ϵ)-breaks Protocol 4.1 in the sense of Definition 2.1. Define an algorithm $\mathcal{C}$ that attempts to solve the one-more-CDH problem in $\mathbb{G}_1$, as follows:

(1) For each challenge h_i that the cheating verifier $\mathcal{B}$ sends to the honest prover $\mathcal{P}$ in step (1) of the protocol, query the CDH oracle with h_i. Run $\mathcal{B}$ on the set of outputs $\{h_i^x\}$.
(2) Simulate the honest verifier $\mathcal{V}$ by querying the challenge oracle $C()$. Send the output r as input to the cheating prover $\mathcal{A}$.
(3) Output t, the element of $\mathbb{G}_1$ sent by the cheating prover $\mathcal{A}$ in step (2) of the protocol.

If $(\mathcal{A}, \mathcal{B})$ successfully breaks the identification scheme, then the element t satisfies $e(g, t) = e(g^a, r)$, and thus by the bilinearity of the pairing, $t = r^a$. The probability of success of $\mathcal{C}$ is thus at least ϵ. Furthermore, $\mathcal{C}$ makes at most q queries to the CDH oracle and runs in time $t + O(1)$. □

5. Identification scheme based on the Strong Diffie-Hellman assumption

Protocol 4.1 is very efficient, requiring an exchange of two elements of $\mathbb{G}_1$, one exponentiation for the prover, and two pairing computations for the verifier. The one-more-CDH required to prove the scheme's security seems reasonable, especially given that similar assumptions are used in the proof security of two well-known identification schemes [2]. However, the fact that the one-more-CDH assumption has not previously appeared in the literature may give one pause, as it is generally

not advisable to introduce new assumptions about computational difficulty. Thus we would like to find an identification scheme as efficient as Protocol 4.1 that requires a weaker security assumption, or at least one that is more widely believed.

The difficulty in adapting the BLS signature scheme into an identification scheme resulted from the random oracle nature of the security proof. Thus we may have more success if we try to adapt a signature scheme that does not require random oracles for its security. Boneh and Boyen [5] have devised such a scheme. The security rests on an assumption known as the *Strong Diffie-Hellman assumption.*

Definition 5.1 ([5, §3.2])**.** Let $\mathbb{G}$ be a cyclic group of prime order p, and let g be a generator. The *q-Strong Diffie-Hellman problem* in $\mathbb{G}$ is defined as follows: given a $(q+1)$-tuple $(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)})$ as input, output a pair $(c, g^{1/(x+c)})$, where $c \in \mathbb{Z}_p$. An algorithm $\mathcal{A}$ has advantage ϵ in solving the q-SDH problem in $\mathbb{G}$ if

$$\Pr\left[\mathcal{A}(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)}) = (c, g^{\frac{1}{x+c}})\right] \geq \epsilon,$$

where the probability is over the choice of $g \in \mathbb{G}$ and $x \in \mathbb{Z}_p^*$.

We say that the *(t, q, ϵ)-Strong Diffie-Hellman assumption* holds in $\mathbb{G}$ if there is no algorithm $\mathcal{A}$ that runs in time t and has advantage ϵ in solving the q-SDH problem in $\mathbb{G}$.

The Boneh-Boyen signature scheme based on the q-SDH problem is as follows.

Protocol 5.2 ([5])**.** Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing. Let g be a generator of $\mathbb{G}_1$.

Key generation: Pick random $x, y \leftarrow \mathbb{Z}_p^*$, and compute $u \leftarrow g^x$, $v \leftarrow g^y$, and $z \leftarrow e(g, g)$. The public key is (u, v, z), and the secret key is (x, y).

Signing: Given a secret key $(x, y) \in (\mathbb{Z}_p^*)^2$, and a message $m \in \mathbb{Z}_p^*$, pick a random $r \in \mathbb{Z}_p^*$ and compute $\sigma \leftarrow g^{1/(x+m+yr)} \in \mathbb{G}_1$, where $1/(x+m+yr)$ is computed modulo p. In the (unlikely) event that $x+m+yr = 0 \pmod{p}$, try again with a different random r. The signature is (σ, r) .

Verification: Given a public key $(u, v, z) \in \mathbb{G}_1^2 \times \mathbb{G}_2$, a message $m \in \mathbb{Z}_p^*$, and a signature $(\sigma, r) \in \mathbb{G}_1 \times \mathbb{Z}_p^*$, compute $e(\sigma, u \cdot g^m \cdot v^r)$. If the result is equal to z output `valid`; if not, output `invalid`.

Theorem 5.3 ([5, Theorem 3.1])**.** *Suppose the (q, t', ϵ')-SDH assumption holds in $\mathbb{G}_1$. Then the signature scheme defined by Protocol 5.2 is (t, q_s, ϵ)-secure against existential forgery under adaptive chosen message attack, provided that*

$$q_s \leq q, \quad \epsilon \leq 2\,(\epsilon' + q_S/p) \approx 2\epsilon' \quad \textit{and} \quad t \leq t' - \Theta(q^2 T),$$

where T is the maximum time for an exponentiation in $\mathbb{G}_1$.

In our protocol based on the Boneh-Boyen scheme, Victor the Verifier sends a random challenge message to Peggy the Prover, which Peggy then signs with her private key.

Protocol 5.4. Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing. Let g be a generator of $\mathbb{G}_1$.

Key generation: Pick random $x, y \leftarrow \mathbb{Z}_p^*$, and compute $u \leftarrow g^x$, $v \leftarrow g^y$, and $z \leftarrow e(g, g)$. The public key is (u, v, z), and Peggy's secret key is (x, y).

Interactive protocol:

(1) Victor sends Peggy a random $m \in \mathbb{Z}_p^*$.

(2) Peggy chooses a random $r \in \mathbb{Z}_p^*$, computes $\sigma = g^{1/(x+m+yr)}$, and sends Victor (σ, r).
(3) Victor computes $e(\sigma, u \cdot g^m \cdot v^r)$. If the result is equal to z he outputs 1 (accept); else he outputs 0 (reject).

Theorem 5.5. *Suppose the (q', t', ϵ')-SDH assumption holds in $\mathbb{G}_1$. Then Protocol 5.2 defines a (t, q, ϵ)-identification scheme, provided that*

$$q \leq q', \quad \epsilon \geq 2\epsilon' \cdot \left(\frac{p}{p-q}\right) + \frac{2q}{p-q} \approx 2\epsilon' \quad \textit{and} \quad t \leq t' - \Theta(q'^2 T),$$

where T is the maximum time for an exponentiation in $\mathbb{G}_1$.

Proof. We first check the viability condition. If Peggy and Victor both follow the protocol, then Victor will always accept, since

$$e(\sigma, u \cdot g^m \cdot v^r) = e(g^{1/(x+m+yr)}, g^x \cdot g^m \cdot g^{yr}) = e(g, g) = z$$

by bilinearity of e. To check the soundness condition, given an attacker $(\mathcal{A}, \mathcal{B})$ that (t, q, ϵ)-breaks the scheme (in the sense of Definition 2.1), we can define an attacker $\mathcal{C}$ that $(t + O(1), q, \epsilon')$-breaks the Boneh-Boyen signature scheme, where $\epsilon' = \epsilon(1 - q/p)$. The reduction is identical to that in the proof of Theorem 3.5, and we choose not to repeat the details. □

6. Identification scheme based on pairing as a one-way function

The identification scheme of Protocol 5.4 is less efficient than that of Protocol 4.1, requiring both more bandwidth and more computation. However, the assumption required to prove security is weaker for the former, implying a tradeoff between efficiency and security. One may ask how far we can carry this tradeoff: what is the weakest possible assumption necessary for a secure identification scheme? We now propose a scheme whose proof of security rests solely on the assumption that the pairing $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is a one-way function when one argument is fixed. This assumption is weaker than both Computational Diffie-Hellman in $\mathbb{G}_1$ and Decision Diffie-Hellman in $\mathbb{G}_2$, both of which are standard assumptions that have been used to prove the security of a wide variety of cryptosystems.

When we say than a pairing is a one-way function, we mean that given $g \in \mathbb{G}_1$ and $y \in \mathbb{G}_2$, it is hard to invert the pairing; that is, to find an element $h \in \mathbb{G}_1$ such that $e(g, h) = y$.

Definition 6.1. Let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing. We say that e is a (t, ϵ)-*one-way pairing* if for any algorithm $\mathcal{A}$ that takes as input $g \in \mathbb{G}_1$ and $x \in \mathbb{G}_2$, produces as output an element of $\mathbb{G}_1$, and runs in time at most t,

$$\Pr\left[e(g, \mathcal{A}(g, x)) = x\right] < \epsilon,$$

where the probability is taken over the possible values of g and x. Given any such $\mathcal{A}$, we say that $\mathcal{A}$ *inverts the pairing* with probability at most ϵ.

As evidence that one-wayness of pairings is a weak assumption, we show that inverting a pairing is no easier than solving either the Computational Diffie-Hellman problem in $\mathbb{G}_1$ or the Decision Diffie-Hellman problem in $\mathbb{G}_2$.

Proposition 6.2. *Let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing between groups of order p. Suppose the (t, ϵ) Computational Diffie-Hellman assumption holds in $\mathbb{G}_1$. Then e is a $(t - O(1), \epsilon)$-one-way pairing.*

Proof. Let $\mathcal{A}(g, x)$ be an algorithm that runs in time t and inverts the pairing with probability at least ϵ. Given a triple (h, h^a, h^b) of elements in $\mathbb{G}_1$, let $y = e(h^a, h^b)$, and run $\mathcal{A}(h, y)$. Then $\mathcal{A}$ outputs h^{ab} with probability at least ϵ. □

Proposition 6.3. *Let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing between groups of order p. Suppose the (t, ϵ)-Decision Diffie-Hellman assumption holds in $\mathbb{G}_2$. Then e is a $(t/\epsilon - O(1), \sqrt[4]{\epsilon})$-one-way pairing.*

Proof. Let $\mathcal{A}(g, x)$ be an algorithm that runs in time t and inverts the pairing with probability at least ϵ. We are given a quadruple $\{y, y^a, y^b, y^c\}$ of elements of $\mathbb{G}_2$ and asked to determine if $c = ab \pmod p$. Define algorithm $\mathcal{B}$ as follows.

(1) Choose a random $g \in \mathbb{G}_1$, and compute

$$h_1 = \mathcal{A}(g, y), \quad h_2 = \mathcal{A}(g, y^a),$$
$$h_3 = \mathcal{A}(g, y^b), \quad h_4 = \mathcal{A}(g, y^c).$$

(2) Compute $e(h_1, h_4)$ and $e(h_2, h_3)$. If the two are equal output 1; else output 0.

Suppose all four outputs of algorithm $\mathcal{A}$ are correct. Then $h_2 = h_1^a$, $h_3 = h_1^b$, and $h_4 = h_1^c$. We therefore have $e(h_1, h_4) = e(h_1, h_1)^c$ and $e(h_2, h_3) = e(h_1, h_1)^{ab}$. The two are equal if and only if $c = ab \pmod p$. Thus if all four outputs are correct $\mathcal{B}$ gives a correct output to the Decision Diffie-Hellman problem. The probability that all four outputs are correct is at least ϵ^4, which gives the stated security bound. Furthermore, $\mathcal{B}$ runs in time $4t + O(1)$. □

Remark 6.4. We can increase the probability of success of $\mathcal{B}$ by iterating the algorithm. Performing each computation of h_i ϵ^{-4} times increases the probability of success to a constant; fewer repetitions lead to different time/success ratios.

Given these two propositions as evidence, we are confident that inverting a pairing is a sufficiently hard problem. We thus forge ahead and define an identification scheme based on the difficulty of inverting a pairing.

Protocol 6.5. Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing.

Key generation: Pick random $P, Q \leftarrow \mathbb{G}_1$, random $y \leftarrow \mathbb{G}_1$, and random $s \leftarrow \mathbb{Z}_p^*$. Compute $v \leftarrow e(P, Q)^{-1} \cdot y^{-s} \in \mathbb{G}_2$. The public key is (P, y, v), and Peggy's secret key is (Q, s).

Interactive protocol:

(1) Peggy chooses random $R \leftarrow \mathbb{G}_1$ and $r \leftarrow \mathbb{Z}_p$, and sends Victor $x = e(P, R) \cdot y^r \in \mathbb{G}_2$.
(2) Victor sends Peggy a random $m \in \mathbb{Z}_p^*$.
(3) Peggy computes $T = R \cdot Q^m \in \mathbb{G}_1$ and $a = r + ms \in \mathbb{Z}_p$, and sends Victor (T, a).
(4) Victor computes $e(P, T) \cdot y^a \cdot v^m \in \mathbb{G}_2$. If the result is equal to x he outputs 1 (accept); else he outputs 0 (reject).

Remark 6.6. It is easy to see that this protocol is viable: if Peggy and Victor both follow the protocol, Victor will always output 1, since

$$\begin{aligned} e(P,T)\cdot y^a\cdot v^e &= e(P,R\cdot Q^m)\cdot y^{r+ms}\cdot (e(P,Q)^{-1}\cdot y^{-s})^m \\ &= e(P,R)\cdot e(P,Q)^m\cdot y^{r+ms}\cdot e(P,Q)^{-m}\cdot y^{-ms} \\ &= e(P,R)\cdot y^r \\ &= x. \end{aligned}$$

Showing security is a trickier matter. Our proof uses the "heavy row" technique introduced by Feige, Fiat, and Shamir [7] in their seminal paper on proofs of identity. The proof closely follows those of Okamoto's schemes [14] based on the discrete logarithm and RSA inversion.

The idea of the proof is as follows. We suppose there is an algorithm $(\mathcal{A},\mathcal{B})$ that breaks Protocol 6.5, and construct an algorithm that tries to invert the pairing. Given $P\in\mathbb{G}_1$ and $y\in\mathbb{G}_2$, we simulate Protocol 6.5 using (P,y) as the public key and our own randomly chosen private key. Successful execution of the algorithm $(\mathcal{A},\mathcal{B})$ on this instance of the protocol gives a valid interaction between the cheating prover $\mathcal{A}$ and the honest verifier $\mathcal{V}$. If we run the algorithm again and have the cheating prover $\mathcal{A}$ use the same random coins, the "heavy row" lemma tells us that we will, with high probability, find a second valid interaction between $\mathcal{A}$ and $\mathcal{V}$. From the transcripts of these two interactions we can compute $X\in\mathbb{G}_1$ such that $e(P,X)=y$, and we have inverted the pairing.

We begin the detailed proof by defining a "heavy row" and proving some useful lemmas.

Definition 6.7. Let $(\mathcal{A},\mathcal{B})$ be an algorithm attacking Protocol 6.5. Let $R_{\mathcal{AB}}$ denote the random coins consumed by $(\mathcal{A},\mathcal{B})$. Let M be a matrix summarizing all of the possible outcomes of the cheating prover $\mathcal{A}$ interacting with an honest verifier $\mathcal{V}$, as follows: the rows of M are indexed by the possible choices of $R_{\mathcal{AB}}$, the columns of M are indexed by all the possible choices e of the verifier $\mathcal{V}$ in step (2), and the entries are 1 if $\mathcal{V}$ accepts $\mathcal{A}$'s proof, and 0 otherwise.

Suppose the probability of success of $(\mathcal{A},\mathcal{B})$ (i.e. the fraction of 1's in M) is ϵ. A row of M is a *heavy row* if its fraction of 1's is at least $\epsilon/2$.

Lemma 6.8. *Suppose the success probability of $(\mathcal{A},\mathcal{B})$ in attacking Protocol 6.5 is at least $2/p$. Then at least half of the 1's in M are located in heavy rows.*

Proof. Assume the contrary, i.e. at least half the 1's in M are located in non-heavy rows. Then the fraction of 1's in all of the non-heavy rows combined is at least $1/p$. On the other hand, in each non-heavy row the fraction of 1's is by definition less than $1/p$, a contradiction. □

Lemma 6.9. *Let $(\mathcal{A},\mathcal{B})$ be an algorithm attacking Protocol 6.5 that runs in time t and has success probability $\epsilon > 2/p$. Then there is a algorithm that runs in expected time $O(t/\epsilon)$ and, with probability at least $\frac{1}{2}(1-\frac{1}{e})^2$ outputs the history of two accepted interactions (x,m,T,a) and (x,m',T',a') of the cheating prover $\mathcal{A}$ with an honest verifier $\mathcal{V}$, where $m\neq m'$.*

Proof. We adopt the following two-step "probing strategy" (cf. [13], [14]) to find two 1's in the same row of M.

Step 1: Probe random entries in M to find an entry a_0 that is a 1. Denote the row in which a_0 is located by M_0.

Step 2: Probe random entries along M_0 to find another entry a_1 with 1.

Let p_1 be the success probability of Step 1 after probing $1/\epsilon$ random entries of M. Since the fraction of 1's in M is ϵ, we have

$$p_1 \geq 1 - (1-\epsilon)^{1/\epsilon} > 1 - \frac{1}{e}.$$

Let p_2 be the success probability of Step 2 after probing $2/\epsilon$ random entries of M_0. If M_0 is a heavy row, then the fraction of 1's in M_0 is at least $\epsilon/2$, and thus the probability of success is at least

$$1 - \left(1 - \frac{\epsilon}{2}\right)^{2/\epsilon} > 1 - \frac{1}{e}.$$

By Lemma 6.8, the probability that M_0 is a heavy row is at least $1/2$, and thus $p_2 > \frac{1}{2}(1 - \frac{1}{e})$. Therefore the overall success probability of our strategy is at least $\frac{1}{2}(1 - \frac{1}{e})^2$, and the total running time is approximately $3t/\epsilon$.

If the strategy finds two entries a_0, a_1 in the same row of M, we output the transcripts (x, e, T, a) and (x, e', T', a') of the interaction between $\mathcal{A}$ and $\mathcal{V}$ when given the random coins corresponding to a_0 and a_1 respectively. Since the entries are in the same row, the random coins of $(\mathcal{A}, \mathcal{B})$ are the same for the two interactions, and thus the first output x is the same for the two interactions. Since the entries are in different columns, the random coins of $\mathcal{V}$ are different for the two interactions, and thus $m \neq m'$. □

With this setup, we may now prove the security of our identification scheme.

Theorem 6.10. *Suppose $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is a (t', ϵ')-one-way pairing, where $\epsilon' > 3/16$ and $p = |\mathbb{G}_1| = |\mathbb{G}_2| \geq 17$. Then Protocol 6.5 is a (t, q, ϵ)-identification scheme, provided that either*

$$\epsilon \leq \frac{2}{p} \quad \text{or} \quad c_0 + \frac{3(t + c_s q)}{\epsilon} \leq t'$$

for some constants c_0, c_s depending on $\mathbb{G}_1$, $\mathbb{G}_2$, and the pairing e.

Proof. In Remark 6.6 we demonstrated the viability condition of Definition 2.1, so we need only show the security condition. Suppose $(\mathcal{A}, \mathcal{B})$ is an algorithm that runs in time t and attacks Protocol 6.5 with success probability $\epsilon > 2/p$. Define an algorithm $\mathcal{C}$ that attempts to invert the pairing, as follows:

(1) Given input $P \in \mathbb{G}_1$ and $y \in \mathbb{G}_2$, choose random $Q^* \in \mathbb{G}_1$ and $s^* \in \mathbb{Z}_p$, and compute $v = e(P, Q^*)^{-1} y^{-s}$.
(2) Simulate Protocol 6.5 with (P, y, v) as the public key and (Q^*, s^*) as the private key.
(3) Run $(\mathcal{A}, \mathcal{B})$ on the simulated protocol $1/\epsilon$ times. If the attack succeeds, record $R_{\mathcal{AB}}$ (the random coins of $(\mathcal{A}, \mathcal{B})$) and the transcript (x, m, T, a).
(4) Run $(\mathcal{A}, \mathcal{B})$ on the simulated protocol $2/\epsilon$ times, using $R_{\mathcal{AB}}$ as the random coins. If the attack succeeds, record the transcript (x, m', T', a').
(5) Let $Q = (T/T')^{1/(m-m')} \in \mathbb{G}_1$ and $s = (a - a')/(m - m') \in \mathbb{Z}_p$. Output

$$Z = (Q/Q^*)^{1/(s^* - s)}.$$

We now analyze the algorithm $\mathcal{C}$. By Lemma 6.9, the probability that steps (3) and (4) both succeed and output valid transcripts with $m \neq m'$ is at least $\frac{1}{2}(1 - \frac{1}{e})^2$. We now claim that if steps (3) and (4) both succeed, then $(Q, s) \neq (Q^*, s^*)$ with

probability almost 1. To prove this, we show that if (Q, s) and (Q^*, s^*) are both valid private keys for the public key (P, y, v), then even an infinitely powerful cheater $\mathcal{B}$ cannot distinguish the two solely from his interaction with an honest prover $\mathcal{P}$. The condition (Q, s) and (Q^*, s^*) both being valid private keys for the public key (P, y, v) implies that

$$e(P, Q) \cdot y^s = e(P, Q^*) \cdot y^{s^*}. \tag{6.1}$$

Let $R^* = R + (Q - Q^*)^m \in \mathbb{G}_1$ and $r^* = r + m(s - s^*) \in \mathbb{Z}_p$. Then the following relations hold:

$$e(P, R) \cdot y^r = x = e(P, R^*) \cdot y^{r^*}$$
$$R + Q^m = T = R^* + Q^{*m}$$
$$r + ms = a = r^* + ms^*$$

Furthermore, for given (Q, Q^*, s, s^*, m), the distribution of (R, r) is identical to that of (R^*, r^*). Since the cheating verifier $\mathcal{B}$ receives only (x, T, a) from the honest prover $\mathcal{P}$, we see that there is no way for $\mathcal{B}$ to determine which private key was used. Since there are p possible pairs (Q, s) satisfying $e(P, Q)^{-1} y^{-s} = v$, the probability that $(Q, s) \neq (Q^*, s^*)$ is $(p - 1)/p$, or nearly 1.

We now show that if steps (3) and (4) succeed and $(Q, s) \neq (Q^*, s^*)$, then step (5) outputs a Z such that $e(P, Z) = y$. We first note that if $(Q, s) \neq (Q^*, s^*)$, then equation (6.1) implies that $Q \neq Q^*$ and $s \neq s^*$, so Z is well-defined. Since x is the same in both transcripts, we have

$$e(P, T) \cdot y^a \cdot v^m = e(P, T') \cdot y^{a'} \cdot v^{m'}.$$

By the bilinearity of the pairing, this implies that

$$e(P, T/T') \cdot y^{a - a'} = v^{m' - m},$$

so by definition of Q and s we have

$$e(P, Q^{m - m'}) \cdot y^{s(m - m')} = v^{m' - m}$$

Raising the whole equation to the power $1/(m - m')$ and applying the definition $v = e(P, Q^*)^{-1} \cdot y^{-s}$ gives

$$e(P, Q) \cdot y^s = e(P, Q^*) y^{s^*}.$$

Again using the bilinearity of the pairing, this gives us

$$e(P, Q/Q^*) = y^{s^* - s},$$

and raising both sides to the power $1/(s^* - s)$ gives

$$e(P, Z) = y,$$

as desired.

Finally, we analyze the running time and success probability of $\mathcal{C}$. If c_s is the time taken to simulate the protocol with the private key (Q^*, s^*), then each iteration of steps (3) and (4) takes time $t + c_s q$, so those two steps take time $3(t + c_s q)/\epsilon$. Steps (1) and (5) take a constant amount of time, say c_0, so the total running time is $c_0 + 3(t + c_s q)/\epsilon$. By Lemma 6.9 and our computations above, if steps (3) and (4) succeed and $(Q, s) \neq (Q^*, s^*)$, then step (5) outputs a valid Z. The probability of the former is at least $\frac{1}{2}(1 - \frac{1}{e})$, while the probability of the latter is $(p - 1)/p$. If $p \geq 17$ then the simultaneous probability of the two events is at least $3/16$. Thus our reduction gives the stated bounds. □

The assumption $p \geq 17$ is trivial, since in cryptographic applications $p \approx 2^{160}$. However, the assumption that e is a (t', ϵ')-one-way pairing with $\epsilon' > 3/16$ is a bit stronger than we would like. If we remove both of these conditions we get the following reduction:

Corollary 6.11. *Suppose $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is a (t', ϵ')-one-way pairing. Then Protocol 6.5 is a (t, q, ϵ)-identification scheme, provided that*

$$\epsilon \geq 3\sqrt{\epsilon'} \quad \textit{and} \quad t \leq \frac{t'}{2} - c_0 - c_s q,$$

for some constants c_0, c_s depending on $\mathbb{G}_1$, $\mathbb{G}_2$, and the pairing e.

Proof. The reduction is the same as in the proof of Theorem 6.10, except we don't iterate steps (3) and (4) of algorithm $\mathcal{C}$. Then the success probability of step (3) is ϵ. By Lemma 6.8 the entry of the summary matrix M corresponding to the output of step (3) is in a heavy row with probability at least $1/2$, and if this is the case then the success probability of step (4) is at least $\epsilon/2$. The success probability of step (5) is still $(p-1)/p$, which is at least $1/2$ since $p \geq 2$. Thus the total success probability π of the algorithm satisfies

$$\pi \geq \epsilon \cdot \frac{1}{2} \cdot \frac{\epsilon}{2} \cdot \frac{1}{2} > \frac{\epsilon^2}{9}.$$

The algorithm takes time $2(t + c_s q) + 2c_0$, where c_s is the time taken to simulate the protocol and $2c_0$ is the time taken to perform the computations in steps (1) and (5). Thus our reduction gives the stated bounds. □

7. Other identification schemes

While there have been several pairing-based identification schemes proposed in the literature, none of these have been given satisfactory proofs of security. The first such scheme, proposed by Kim and Kim [11] and based on the Gap Diffie-Hellman problem, was shown to be breakable in constant time by any adversary knowing only the public key. Yao, Wang, and Wang [17] proposed a modification of the scheme and proved it to be secure if the Gap Diffie-Hellman problem (cf. Remark 2.6) is hard. However, their reduction requires exponential time, and thus the proof is unsatisfactory. We will therefore not consider these two schemes when comparing the various pairing-based identification schemes.

More recently, two pairing-based identification schemes have been proposed that appear to be more promising. Shao, Cao, and Lu [16] have proposed a scheme very similar to our Protocol 5.4, based on the Boneh-Boyen signature scheme. The authors claim that the scheme's security depends on the intractability of the Strong Diffie-Hellman problem, but they do not give a proof, and we have not been able to come up with a reduction. The scheme is as follows:

Protocol 7.1 ([16])**.** Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing.

Key generation: Pick random $g \leftarrow \mathbb{G}_1$ and $x \leftarrow \mathbb{Z}_p^*$, and compute $v \leftarrow g^x \in \mathbb{G}_1$ and $z \leftarrow e(g, g) \in \mathbb{G}_2$. The public key is (g, v, z), and Peggy's secret key is x.

Interactive protocol:

(1) Peggy chooses a random $w \in \mathbb{Z}_p^*$ and sends Victor $\tau = g^w$.
(2) Victor sends Peggy a random $r \in \mathbb{Z}_p^*$.

(3) Peggy sends Victor $\sigma = g^{1/(xr+w)}$.
(4) Victor computes $e(\sigma, \tau \cdot v^r)$. If the result is equal to z he outputs 1 (accept); else he outputs 0 (reject).

Conjecture 7.2. Suppose there exists an algorithm $(\mathcal{A}, \mathcal{B})$ that (t, q, ϵ)-breaks Protocol 7.1. Then there is an algorithm $\mathcal{C}$ that runs in time polynomial in t and q and succeeds in solving the Strong Diffie-Hellman problem with probability polynomial in ϵ.

The final pairing-based identification scheme we consider was proposed by Hufschmitt, Lefranc, and Sibert [10]. The scheme is similar to our Protocol 6.5. The authors assert that breaking the scheme is equivalent to solving the Gap Diffie-Hellman problem (cf. Remark 2.6), and that the reduction is based on the same ideas as Theorem 6.10. Indeed, the scheme appears to be amenable to the same type of reduction as in our proof, but the details have yet to be worked out. We thus state the security result as a conjecture.

Protocol 7.3 ([10])**.** Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic groups of prime order p, and let $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a cryptographic pairing..

Key generation: Pick random $g \leftarrow \mathbb{G}_1$ and $a, b \leftarrow \mathbb{Z}_p^*$, and compute $h \leftarrow g^a, k \leftarrow g^b, s \leftarrow g^{ab} \in \mathbb{G}_1$ and $z \leftarrow e(g,g), v \leftarrow e(g,g)^{ab} \in \mathbb{G}_2$. The public key is (g, g^a, g^b, s, v, z), and Peggy's secret key is g^{ab}.

Interactive protocol:

(1) Peggy sends Victor a random $r \in \mathbb{Z}_p^*$ and sends Victor $w = z^r = e(g,g)^r$.
(2) Victor sends Peggy a random $c \in \mathbb{Z}_p^*$.
(3) Peggy sends Victor $\sigma = g^r \cdot s^c$.
(4) Victor computes $e(g, \sigma)$ and $w \cdot v^c$ in $\mathbb{G}_2$. If the two are equal he outputs 1 (accept); else he outputs 0 (reject).

Remark 7.4. The public parameters $h = g^a$ and $k = g^b$ are not used anywhere in the protocol; it appears that they are only included to allow us to reduce breaking the protocol to breaking the Computational Diffie-Hellman problem in $\mathbb{G}_1$. If these two parameters are ignored, then the relevant computational problem is not CDH but inverting the pairing. We therefore conjecture that breaking the protocol will reduce to inverting the paring as in our Theorem 6.10.

Conjecture 7.5. Suppose there exists an algorithm $(\mathcal{A}, \mathcal{B})$ that (t, q, ϵ)-breaks Protocol 7.3. Then there is an algorithm $\mathcal{C}$ that runs in time polynomial in t and q and succeeds in inverting the pairing e with probability polynomial in ϵ.

8. Comparison of identification schemes

We now compare the various identification schemes we have presented so far in terms of bandwidth and computation required for one iteration of each protocol. The results are summarized in Table 1.

Currently, the only pairings used in cryptographic applications are derived from the Weil and Tate pairings on elliptic curves over finite fields $\mathbb{F}_q$. These pairings map from the elliptic curve group $E(\mathbb{F}_q)$ to some extension field $\mathbb{F}_{q^k}$; the parameter k is called the *embedding degree* of the curve E. For the pairing to be useful, it is necessary that the discrete logarithm problems in $E(\mathbb{F}_q)$ and $\mathbb{F}_{q^k}$ are both hard.

ID Scheme	Security Assumption	Bandwidth			Computation		
		$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{Z}_p$	$\mathbb{G}_1$ exp.	$\mathbb{G}_2$ exp.	Pairings
3.4	CDH in $\mathbb{G}_1$ (ROM)	1	0	1	1P	0	2V
4.1	one-more-CDH	2	0	0	1P	0	2V
5.4	SDH in $\mathbb{G}_1$	1	0	2	1P, 2V	0	1V
6.5	e is one-way	1	1	2	1P	1P, 2V	1P, 1V
7.1	SDH in $\mathbb{G}_1$(?)	2	0	1	2P, 1V	0	1V
7.3	e is one-way(?)	1	1	1	2P	1P, 1V	1V

TABLE 1. Comparison of proposed identification schemes. The Bandwidth column indicates the number of elements of $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{Z}_p$ exchanged during one instance of the protocol. The Computation column indicates how many exponentiations in $\mathbb{G}_1$, exponentiations in $\mathbb{G}_2$, and pairing computations the Prover and Verifier must execute during one instance of the protocol. We note that the security proof of Protocol 3.4 is in the Random Oracle Model.

Given current discrete logarithm algorithms, $q \sim 2^{160}$ and $k \sim 2^{1024}$ appear to be reasonable choices for the parameters.

We now assume that $\mathbb{G}_1 = E(\mathbb{F}_q)$, $\mathbb{G}_2 = \mathbb{F}_{q^k}$, and $p \approx q$. An element P of $E(\mathbb{F}_q)$ can be represented by an element of $\mathbb{F}_q$ corresponding to the x-coordinate of P, plus one bit for the sign of the y-coordinate. Thus elements of $\mathbb{G}_1$ and $\mathbb{Z}_p$ are of about the same size ($\log_2 p$ bits), while elements of $\mathbb{G}_2$ will be k times as large. Therefore if minimizing bandwidth is a primary concern, one of Protocols 3.4 or 4.1 should be used. Protocols 6.5 and 7.3 require an element of $\mathbb{G}_2$ to be transmitted, so they should be avoided.

If minimizing computational time is a primary concern, we will wish to minimize pairing computation and perform as few exponentiations as possible in the larger group. Thus Protocols 5.4 and 7.1 are ideal for this application. If we only care about minimizing the Prover's computational time, as in a smart card application, then one of Protocols 3.4, 4.1, or 5.4 will be best. However, Protocol 3.4 may be less preferable since the prover and verifier must each compute a hash function in addition to performing the group computations.

Finally, if security is the foremost concern, then we should choose a scheme whose proof requires the weakest security assumption. Table 2 shows the implications between the various computational assumptions used to prove security of our protocols. We see that the weakest assumption is that the pairing is a one-way function. Protocols 6.5 and 7.3 are based on this assumption, so these two schemes are the most secure. Which of the two is preferable will depend on the particular implementation: if pairing computation is faster then exponentiation in $\mathbb{G}_1$ then our Protocol 6.5 is preferable; if the reverse is true then Protocol 7.3 will be faster.

9. Conclusion

We have presented four new identification schemes based on pairings, and proved their security given various computational assumptions. Each of our schemes is at least as efficient and/or secure as any scheme currently in the literature. Our main contribution is Protocol 6.5, a scheme which is secure if the pairing in question

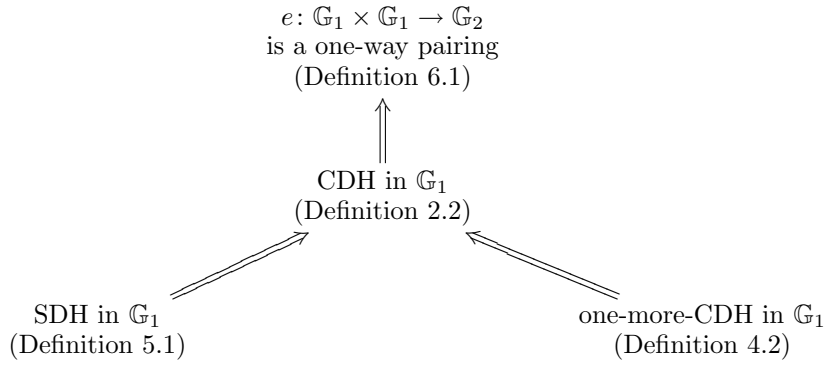

TABLE 2. Implications between various computational assumptions.

is a one-way function; this assumption is weaker than that made for any other pairing-based scheme currently in the literature.

For another of our schemes, Protocol 4.1, we introduced an assumption called the "one-more-CDH" assumption, analogous to the "one-more-discrete-log" and "one-more-RSA-inversion" assumptions, and proved our scheme secure under this assumption. An important open question is what relation this assumption has to other computational assumptions in the literature.

REFERENCES

[1] M. Bellare, C. Namprempre, D. Pointcheval, M. Semanko, "The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme," *Journal of Cryptology* **16**:3 (2003), 185-215.

[2] M. Bellare, A. Palacio, "GQ and Schnorr identification schemes: proofs of security against impersonation under active and concurrent attacks," in *CRYPTO '02*, ed. M. Yung, Springer LNCS **2442** (2002), 162-177.

[3] I. Blake, G. Seroussi, N. Smart, *Elliptic Curves in Cryptography*, LMS Lecture Note Series **265**, Cambridge University Press, 1999.

[4] I. Blake, G. Seroussi, N. Smart, eds., *Advances in Elliptic Curve Cryptography*, LMS Lecture Note Series **317**, Cambridge Unviersity Press, 2005.

[5] D. Boneh, X. Boyen, "Short signatures without random oracles," in *EUROCRYPT '04*, ed. C. Cachin, J. Camenisch, Springer LNCS **3027**, 2004, 56-73.

[6] D. Boneh, B. Lynn, H. Shacham, "Short signatures from the Weil pairing," in *ASIACRYPT '01*, ed. C. Boyd, Springer LNCS **2248** (2001), 514-532.

[7] U. Feige, A. Fiat, A. Shamir, "Zero knowledge proofs of identity," *Journal of Cryptology* **1**:2 (1988), 77-94.

[8] O. Goldreich, *Foundations of Cryptography*, Vol. 1, Cambridge University Press, Cambridge, 2001.

[9] L. S. Guillou, J. J. Quisquater, "A 'paradoxical' identity-based signature scheme resulting from zero-knowledge," in *CRYPTO '88*, ed. S. Goldwasser, Springer LNCS **403** (1990), 216-231.

[10] E. Hufschmitt, D. Lefranc, H. Sibert, "A zero-knowledge identification scheme in Gap Diffie-Hellman groups," in *Western European Workshop on Research in Cryptology, 2005* (conference records available online at `http://www.weworc.org`), 8-12.

[11] M. Kim, K. Kim, "A new identification scheme based on the Bilinear Diffie-Hellman problem," in *ACISP '02*, Springer LNCS **2384** (2002), 362-378.

[12] A. Joux, K. Nguyen, "Separating Decision Diffie-Hellman from Computational Diffie-Hellman in cryptographic groups," *Journal of Cryptology* **16**:4 (2003), 239-247.

[13] K. Ohta, T. Okamoto, "On concrete security treatment of signatures derived from identification," in *CRYPTO '98*, ed. H. Krawczek, Springer LNCS **1462** (1998), 354-370.
[14] T. Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," in *CRYPTO '92*, ed. E. F. Brickell, Springer LNCS **740** (1993), 31-53.
[15] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology* **4**:3 (1991), 161-174.
[16] J. Shao, R. Lu, Z. Cao, "A new efficient identification scheme based on the Strong Diffie-Hellman assumption," in *International Symposium on Future Software Technology*, 2004.
[17] G. Yao, G. Wang, Y. Wang, "An improved identification scheme," in *Coding, Cryptography, and Combinatorics*, Berkhäuser-Verlag Progress in Computer Science and Applied Logic **23** (2004), 397-405.

Hewlett-Packard Laboratories
E-mail address: dfreeman@math.berkeley.edu