IEICE TRANS. FUNDAMENTALS, VOL.E86–A, NO.1 JANUARY 2003

PAPER *Special Section on Cryptography and Information Security*

Lenient/Strict Batch Verification in Several Groups

Fumitaka HOSHINO[†a)], ***Nonmember***, **Masayuki ABE**[†], ***and*** **Tetsutaro KOBAYASHI**[†], ***Regular Members***

SUMMARY Batch verification is a useful tool in verifying a large number of cryptographic items all at one time. It is especially effective in verifying predicates based on modular exponentiation. In some cases, however the items can be incorrect although they pass batch verification together. Such leniency can be eliminated by checking the domain of each item in advance. With this in mind, we introduce the strict batch verification and investigate if the strict batch verification can remain more effective than separate verification. In this paper, we estimate the efficiency of such strict batch verification in several types of groups, a prime subgroup of Z_p with special/random prime p and prime subgroups defined on elliptic curves over F_p, F_{2^m} and F_{p^m}, with are often used in DL-based cryptographic primitives. Our analysis concludes that the efficiency differs greatly depending on the choice of the group and parameters determined by the verifying predicate. Furthermore, we even show that there are some cases where batch verification, regardless of strictness, loses its computational advantage.

key words: *batch verification, elliptic curve cryptosystem*

1. Introduction

In cryptographic protocols verification of items such as signatures, zero-knowledge proofs, and ciphertexts plays an important role to maintain robustness. In large-scale applications, such verification could dominate efficiency since these items are typically verified by a central server while generated by each player. A typical example would be a bank serving in an off-line e-cash system, which verifies all electronic coins, i.e., the bank's signatures, spent each day [6]. Another good example would be electronic voting schemes where millions of voters cast encrypted ballots and a group of centralized servers shuffles and opens them in a verifiable way [1]–[3], [17]. When real-time response is not required, batch verifying items would reduce the cost. In [4], [14], [17], [19], it was shown that a type of verification predicates based on modular exponentiation can be efficiently batch verified.

It was shown in [5], however, that a set of items that is surely rejected in individual verification can be accepted in the batch mode with high probability depending on the group that defines the items. Intuitively, it happens when some elements of the items, which are supposed to be in a group, are in fact out of the group in such a way that the deviating parts eliminate each other in combination so that they are accepted in batch mode. As observed in [5], such erroneous items might not be a critical problem when the items are signatures since it is only the legitimate signer who can generate the erroneous signatures. However, when the verifying items are zero-knowledge proofs, it could be a serious threat since the mistakenly accepted items may be used in further processing.

In this paper, we consider *strict* batch verification that provides virtually the same confidence level as that provided by separately verifying each item. We estimate the efficiency of such strict batch verification in several types of groups such as a prime subgroup of Z_p with special/random prime p and prime subgroups defined on elliptic curves over F_p, F_{2^m} and F_{p^m}, which are often used in DL-based cryptography. We show that the efficiency level differs greatly depending on the choice of the group and the parameter determined by the verifying predicate. Furthermore, we even show that there are some cases where batch verification, regardless of the strictness, loses its computational advantage. This result would be an independent interest as batch verification is supposed to be faster than separate verification.

The rest of the paper is organized as follows. In Sect. 2, we define the items to check, and review some verification methods including separate verification and lenient/strict batch verification. In Sect. 3, our results are summarized. Section 4 introduces a criterion for the advantage of batch verification Based on this criterion we estimate the cost of batch verification in Sect. 5.

2. Preliminaries

2.1 Separate Verification of DL-Based Items

Let g be a generator of a cyclic group of order q. In this section, we use multiplicative notation for the group operation and all arithmetic operations are done in the group unless otherwise noted. Let $\mathcal{I}_n^k$ be a collection of n items for verification. Let I_i^k be the i-th item in $\mathcal{I}_n^k$ defined as

Manuscript received March 23, 2002.
Manuscript revised July 11, 2002.
Final manuscript received September 20, 2002.

[†]The authors are with NTT Information Sharing Platform Laboratories, NTT Corporation, Yokosuka-shi, 239-0847 Japan.
a) E-mail: fhoshino@isl.ntt.co.jp

$$I_i^k = (e_{i1}, \ldots, e_{ik}, h_{i0}, h_{i1}, \ldots, h_{ik}) \in Z_q^k \times \langle g \rangle^{k+1}$$

for some $k \geqq 1$. Separate verification, $V_{\rm sep}(\mathcal{I}_n^k)$, consists of two phases

Domain test:

$$h_{ij} \stackrel{?}{\in} \langle g \rangle \quad \text{for } 1 \leqq j \leqq k,\ 1 \leqq i \leqq n, \text{ and} \qquad (1)$$

Separate relation test:

$$h_{i0} \stackrel{?}{=} \prod_{j=1}^{k} h_{ij}^{e_{ij}} \quad \text{for } 1 \leqq i \leqq n. \qquad (2)$$

We denote $V_{\rm sep}(\mathcal{I}_n^k) = \mathsf{true}$ if $\mathcal{I}_n^k$ passes the above tests. The items in $\mathcal{I}_n^k$ are defined to be correct if $V_{\rm sep}(\mathcal{I}_n^k) = \mathsf{true}$. It is stressed that the domain test does not apply to h_{i0} because if an item satisfies the predicate h_{i0} must also pass the domain test.

2.2 Lenient Batch Verification

In many applications, some of $\{h_{i1}, \ldots, h_{ik}\}$ may common to all i. Without loss of generality, we assume that $h_{ik_x}, \ldots, h_{ik}$ are common to all i and the rest of the bases are different in each item. To simplify the notation, we omit suffix i if it's unnecessary, thus $h_{ik_x}, \ldots, h_{ik}$ are denoted as $h_{k_x}, \ldots, h_k$. Lenient batch verification $V_{\rm bat}(\mathcal{I}_n^k)$ consists of

Lenient domain test: (the same as in (1))

Batch relation test:

$$1 \stackrel{?}{=} \Big(\prod_{i=1}^{n} \prod_{j=0}^{k_x-1} h_{ij}^{r_i e_{ij}} \Big) \Big(\prod_{j=k_x}^{k} h_j^{\hat{e}_j} \Big), \qquad (3)$$

where $e_{i0} = -1$ and $\hat{e}_j = \sum_{i=1}^{n} r_i e_{ij} \in Z_q$ for $r_i \in_U R \subseteqq Z_q$. We denote $V_{\rm bat}(\mathcal{I}_n^k) = \mathsf{true}$ if $\mathcal{I}_n^k$ passes the above tests. From the results described in [5], it is shown that if all items are taken from the correct domain and $V_{\rm bat}(\mathcal{I}_n^k) = \mathsf{true}$, then $V_{\rm sep}(\mathcal{I}_n^k) = \mathsf{true}$ with probability $1 - 1/|R|$.

2.3 Overlap Factor

In this section, we introduce the overlap factor, which is an essential parameter in the evaluation of verification cost in later sections.

The advantage of batch verification is based on the fact that

- computing $(n \times k)$-base exponentiation is faster than computing k-base exponentiation n times, and
- exponents for overlapped bases can be wrapped up.

Accordingly, batch relation Predicate (3) can be computed faster than repeatedly computing separate Predicate (2) n times. We refer to Algorithm 14.88 in [13] for an exponentiation algorithm with multiple bases. When many of the bases are common to all the items, the advantage clearly grows. We define *overlap factor* k_o as $k - k_x + 1$ such that it represents the number of bases commonly included in all the items.

$k_o = 0$ No base is common among the items. This occurs, for instance, in the predicate of zero-knowledge proofs that proves equality of the discrete logarithms, $\log_{g_1} y_1 = \log_{g_2} y_2, \ldots$

$k_o = 1$ Only one base is common among the items. Such a case would be observed in applications where all items are generated based on a single generator, say g.

$k_o \geqq 2$ More than two bases are common. Such a case would be observed when the items are based on the representation problem such as the Okamoto identification scheme [16] or Brand's electronic cash scheme [6].

The advantage of batch verification seems to be maximized when $k_o = k$ where all bases but h_{i0} are common and minimized when $k_o = 0$ where all bases are different, but later in this paper, we can see that the advantage of batch verification is not always maximized when $k_o = k$.

2.4 Previous Works

In this section, we give an overview of previous works on DL-based batch verification. Batch verification for DSA signatures was introduced by Naccache, David M'Raïhi, Vaudenay and Raphaeli in [14]. Their scheme is designed to verify several DSA signatures at once, and they discussed a technique which called the small exponents test nowadays. Another pioneering work on DL-based batch verification was given by Yen and Laih [19]. They also proposed the small exponents test for Schnorr scheme, and give a rough performance analysis for their small exponents test.

Bellare, Garay and Rabin give a generalized model for batch verification of exponentiation and described several verification algorithms [4]. They prove that $V_{\rm bat}(\mathcal{I}_n^1)$ is a good batch verifier with error bounded by $1/|R|$ as long as the order of $\langle g \rangle$ is prime. They also estimate the cost of $V_{\rm sep}(\mathcal{I}_n^k)$ and $V_{\rm bat}(\mathcal{I}_n^k)$ where $k = k_o = 1$ in our notation, but their estimation of $V_{\rm sep}(\mathcal{I}_n^k)$ is not so clear to evaluate. Therefore it is not confirmed that $V_{\rm bat}(\mathcal{I}_n^k)$ is faster than $V_{\rm sep}(\mathcal{I}_n^k)$.

In [5], Boyd and Pavlovski show that a set of items that is surely rejected in individual verification can be accepted in the batch mode with high probability depending on the group that defines the items. They say such erroneous items might not be a critical problem when the items are signatures since it is only the legitimate signer who can generate the erroneous signatures.

In our previouse paper [8], we show that when the verifying items are zero-knowledge proofs, it could be a

Table 1 Effectiveness of batch verification.

<table>
<tr><th>Section</th><th>Group</th><th>$k_o = 0$</th><th>$k_o = 1$</th><th>$k_o = 2$</th><th>$k_o \geqq 3$</th></tr>
<tr><td>5.1</td><td>SG of Z_p^* (random prime)</td><td colspan="2">d</td><td>q</td><td>k</td></tr>
<tr><td>5.2</td><td>SG of Z_p^* (special prime)</td><td colspan="4" rowspan="4">a</td></tr>
<tr><td rowspan="2">5.3</td><td>Prime Field</td></tr>
<tr><td>OEF (random curve)</td></tr>
<tr><td>5.4</td><td>BF (random curve)</td></tr>
<tr><td rowspan="2">5.5</td><td>OEF (special curve)</td><td>E</td><td>e</td><td colspan="2" rowspan="2"></td></tr>
<tr><td>BF (special curve)</td><td colspan="2">D</td></tr>
</table>

Parameter k_o is the number of common bases among all verification items.
SG means Subgroup and BF means Binary Field.
SBV and LBV means Strict and Lenient Batch Verification respectively.

- **a** : SBV is unconditionally **a**dvantageous.
- **d** : SBV is **d**isadvantageous if no effective test is found.
- **q** : SBV is advantageous if the q-th power is fast (and $k \neq k_o$),
- **k** : SBV is advantageous if $k \neq k_o$,
- **e** : SBV is advantageous if the **e**xtension degree is small.
- **E** : LBV is advantageous, if the **E**xtension degree is small.
- **D** : LBV is unconditionally **D**isadvantageous.

serious threat since the mistakenly accepted items may be used in further processing, and proposed the concept of the strict batch verification. But it didn't care about $k = k_o$.

Therefore we will reconsider the definition of the batch verification and complete the missing analysis in this paper.

2.5 Strict Batch Verification

In the batch relation Predicate (3), note that h_{i0} is still outside the domain testing. We define *strict batch verification* $V_{\mathrm{str}}(\mathcal{I}_n^k)$ as a combination of

Strict domain test:
$$h_{ij} \stackrel{?}{\in} \langle g \rangle \quad \text{for } 0 \leqq j \leqq k,\ 1 \leqq i \leqq n, \text{ and} \qquad (4)$$
Batch relation test: (the same as in (3))

Note that h_{i0} is now included in the domain test. We denote $V_{\mathrm{str}}(\mathcal{I}_n^k) = \mathsf{true}$ if $\mathcal{I}_n^k$ passes these tests. When $V_{\mathrm{str}}(\mathcal{I}_n^k) = \mathsf{true}$, this implies that $V_{\mathrm{sep}}(\mathcal{I}_n^k) = \mathsf{true}$ with probability $1 - 1/|R|$.

To see the implications of the strict domain test, we review the results of [5], which state that batch verification without a strict domain test does not provide the same security as does separate verification. The example abuses the implicit assumption that h_{i0} is taken from the correct domain. Here, we assume that $\langle g \rangle$ is a multiplicative subgroup in Z_p where $ord(g) = q$, $p = 2q + 1$, and q is prime. Let $\mathcal{I}_2^k = \{(e_{i1}, \ldots, e_{ik}, h_{i0}, h_{i1} \ldots, h_{ik})\}_{i=1,2}$ be correct items with regard to the separate verification predicate. Note here that -1 is not in $\langle g \rangle$. Then, let $\tilde{\mathcal{I}}_2^k = \{(e_{i1}, \ldots, e_{ik}, (-1)h_{i0}, h_{i1}, \ldots, h_{ik})\}_{i=1,2}$. $\tilde{\mathcal{I}}_2^k$ fails in the separate verification, but it passes the batch verification with probability 1/2. This happens when r_1 and r_2 are both even or odd.

3. Summary of Our Contribution

We introduced a new class of the batch verification and give the concept of the strict batch verification which guarantees the verification results are the same as the logical product of single verifications.

And we analyze the advantage of batch verification by comparing it to the separate verification. The advantage is estimated with respect to the computational cost. We say that it has advantage if it saves more than one exponentiation per item. Our results are summarized in Table 1. See Sect. 5 for the classification of groups and specific schemes of strict batch verification. The symbols **a**,**d** and **D** in Table 1 respectively denote

- **a**: Strict batch verification is unconditionally advantageous,
- **d**: Strict batch verification is disadvantageous, and
- **D**: Lenient batch verification is unconditionally disadvantageous.

The lower case symbols (**a**,**d**,**q**,**k**, and **e**) also have an additional meaning,

> "Lenient batch verification is unconditionally advantageous,"

and the upper case symbols (**E** and **D**) represent

> "Strict batch verification is unconditionally disadvantageous."

The symbols **q**, **k**, **e** and **E** mean Lenient or Strict batch verification is advantageous under some special conditions (see the Table 1 and Sect. 5 for details). What is interesting in Table 1 is

- I-1 Almost all areas are occupied by symbol **a**.
- I-2 Symbols **d** and **D** have completely different reasons why the strict batch verification is disadvantageous.

Item I-1 means that strict batch verification can be

performed efficiently on almost all practical groups on which lenient batch verification can be performed efficiently. Item I-2 supposes the following.

- Symbol **d** implies that no effective domain test is found. If an effective test is found, strict batch verification becomes advantageous.
- Symbol **D** implies that batch verification cannot be faster than separate verification although an effective domain test is available. This means that the separate verification is sufficiently fast by itself. Symbols **E** and **e** are special cases of **D**.

Note that **D** never means that the binary field is unsuited to large-scale applications.

4. Basic Concepts for Analysis

4.1 Gain and Loss

In this section, we introduce the concept of gain and loss, which provides a simple criterion for the advantage of batch verification.

In the rest of this paper, we estimate computation costs by the number of group operations in $\langle g \rangle$ and assume that all other arithmetic operations, equality tests, and random number generations are done for free.

The gain, $c_{\rm gain}$, is defined as the difference in the cost per verification item between separate relation test (2) and batch relation test (3). That is, let $C_{\rm sep}$ and $C_{\rm bat}$ be the computational cost of Predicate (2) and Predicate (3) respectively, then

$$c_{\rm gain} = (C_{\rm sep} - C_{\rm bat})/n \tag{5}$$

and the loss, $c_{\rm loss}$, is defined as the cost to ensure $h_0 \in \langle g \rangle$ for strict batch verification. That is the difference in the cost per verification item between lenient domain test (1) and strict domain test (4).

The concept of gain and loss provides a simple criterion for the advantage of batch verification: If the loss is less than the gain, then the batch verification is meaningful, otherwise it does not make any sense. The advantage of the batch verification is given by

$$c_{\rm loss} < c_{\rm gain} \tag{6}$$

For discussion on lenient batch verification, set $c_{\rm loss} = 0$ since the domain test is the same as that in separate verification, while for strict batch verification, $c_{\rm loss}$ must be estimated.

4.2 Generic Evaluation of the Gain ($k_o \neq k$)

This section estimates $c_{\rm gain}$ more precisely without specifying an underlying group. For this purpose, we focus on an abstract exponentiation scheme called the *'optimal'* exponentiation scheme, and we give an estimation of $c_{\rm gain}$ for such optimal exponentiation scheme. The concrete scheme may be the binary method, the signed binary method, or something window method [13]: however, this is not specified because it varies with the choice of the underlying group. We will discuss such variations in later sections in this paper.

Let G be the group we focus on where c_{m} is the computational cost of a multiplication on G and c_{s} is the cost of a square on G. Let N_{m} be the average number of multiplications used in the optimal exponentiation scheme and N_{s} be the average number of squares required by the scheme. The average cost of the exponentiation scheme on G is $N_{\mathsf{m}}c_{\mathsf{m}} + N_{\mathsf{s}}c_{\mathsf{s}}$. By extending the scheme [4], [13], k-base exponentiation can be computed with costs $kN_{\mathsf{m}}c_{\mathsf{m}} + N_{\mathsf{s}}c_{\mathsf{s}}$. Therefore,

$$C_{\rm sep} = n \times (kN_{\mathsf{m}}c_{\mathsf{m}} + N_{\mathsf{s}}c_{\mathsf{s}})$$

The dominant task of batch relation test (3) is the product of $nk_x + k_o$ exponentiations which costs

$$C_{\rm bat} = (nk_x + k_o)N_{\mathsf{m}}c_{\mathsf{m}} + N_{\mathsf{s}}c_{\mathsf{s}}$$

In general, $V_{\rm sep}(\mathcal{I}_n^k)$ and $V_{\rm bat}(\mathcal{I}_n^k)$ can have different values of $N_{\mathsf{m}}, N_{\mathsf{s}}, c_{\mathsf{m}}$ and c_{s}. We ignore this effect, since N_{s} has only a slight effect on $V_{\rm bat}(\mathcal{I}_n^k)$, and no significant difference is expected with respect to N_{m}. When G is a subgroup of Z_p^*, c_{m} and c_{s} can be fixed. Even for elliptic curves, there is no significant difference between $V_{\rm sep}(\mathcal{I}_n^k)$ and $V_{\rm bat}(\mathcal{I}_n^k)$ [7]. Then from Definition (5), we have

$$c_{\rm gain} = \left((k_o - 1) - \frac{k_o}{n}\right) N_{\mathsf{m}}c_{\mathsf{m}} + \left(1 - \frac{1}{n}\right) N_{\mathsf{s}}c_{\mathsf{s}}$$

Since we are only concerned with a large number of items, we assume that $k_o \ll n$ (and $1 \ll n$). Then we have

$$c_{\rm gain} \approx (k_o - 1)N_{\mathsf{m}}c_{\mathsf{m}} + N_{\mathsf{s}}c_{\mathsf{s}} \tag{7}$$

Through the rest of this paper, we estimate the efficiency averaged among the verification items, since the above approximation is independent of n. According to Approximation (7), the criterion $c_{\rm loss} < c_{\rm gain}$ is written by

$$c_{\rm loss} < (k_o - 1)N_{\mathsf{m}}c_{\mathsf{m}} + N_{\mathsf{s}}c_{\mathsf{s}} \tag{8}$$

We call this inequality *'Criterion'* (8). Let q be the order of G: for almost all practical exponentiation schemes, we observe

$$N_{\mathsf{m}} = \frac{1}{\omega}\lfloor \log_2 q \rfloor \ , \quad N_{\mathsf{s}} = \lfloor \log_2 q \rfloor \tag{9}$$

where $2 \leqq \omega$. We call factor ω the *'abstract window size,'* which depends on the exponentiation scheme (and q). For example, $\omega = 2$ for the binary method, $\omega = 8/3$ for the signed binary method, and will be greater for a more sophisticated method [13]. In general, $\omega \ll \log_2 q$, and if $q \sim 2^{160}$ then we observe $\omega \lesssim 5$. Suppose that $c_{\mathsf{m}} \approx c_{\mathsf{s}}$, then from Inequality (8) we have

$$c_{\rm loss}/c_{\mathsf{m}} < \left(\frac{1}{\omega}(k_o-1)+1\right)\lfloor \log_2 q \rfloor \tag{10}$$

We also call this inequality Criterion (10). This criterion is the reason why lenient batch verification is more efficient than separate verification. That is if $c_{\rm loss}=0$, then Criterion (10) is always satisfied since $0 \leqq k_o$ and $2 \leqq \omega$. This means that lenient batch verification is always faster than separate verification. However, if either Approximation (7) or Observation (9) does not hold, we may have different results. We will deal with such a special case later in this paper.

Note that when $c_{\rm loss} \approx c_{\rm gain}$ batch verification is not recommended even if $c_{\rm loss} < c_{\rm gain}$, because the cost of random number generation, exponent number generation, etc. are ignored. Therefore, if $c_{\rm gain} - c_{\rm loss}$ is comparable to the decrease of one exponentiation base, that is

$$N_{\mathsf{m}} c_{\mathsf{m}} \lesssim c_{\rm gain} - c_{\rm loss}$$

then we ignore this matter, otherwise it is given special attention.

4.3 Special Case ($k_o = k$)

In this section, we show that the advantage of batch verification is not always maximized when $k_o = k$. If $k_o = k$, we must reconsider the cost evaluation of $C_{\rm sep}$, because the computation of $V_{\rm sep}(\mathcal{I}_n^k)$ can be recognized as a set of *k-fixed* base exponentiation. This can be evaluated as

$$C_{\rm sep} = n \times (kN_{\mathsf{m}} c_{\mathsf{m}})$$

which causes that

$$c_{\rm gain} \approx (k-1) N_{\mathsf{m}} c_{\mathsf{m}}.$$

Therefore $V_{\rm bat}(\mathcal{I}_n^k)$ cannot be advantageous when $k_o = k = 1$ (and $R = Z_q$). When $k \geqq 2$, the situation is the same as Sect. 4.2 as long as the fast domain test is available. If the fast domain test is not available, we must use

$$\frac{c_{\rm loss}}{c_{\mathsf{m}}} < \frac{1}{\omega}(k-1)\lfloor \log_2 q \rfloor \tag{11}$$

instead of Criterion (10).

5. Detailed Analysis

5.1 Subgroup of Z_p^* for Random Prime p

In this section, we examine the order-q subgroup of Z_p^* without any special conditions of r such that $p = 2qr + 1$. We call such a prime, p, '*random prime.*' We show that lenient batch verification is always faster than separate verification for this group, and investigate the conditions under which strict batch verification has an advantage over separate verification.

Let p, q be prime such that $q|(p-1)$, and let G be the order-q subgroup of Z_p^*. We can use Criteria (8) and (10) in this case, which means that lenient batch verification is always faster than separate verification for this group. Therefore, we only discuss strict batch verification in this section. The ideal input of h_0 is an element of G. However, an element of G is practically represented as an element of Z_p. There is an additional cost to ensure $h_0 \in G$. If p is a random prime such that $p-1$ has many small factors, there is no means except $h_0^q = 1$. Suppose that h_0^q is calculated by the window method or its derivation, then we have an approximation of $c_{\rm loss}$ as

$$c_{\rm loss} \approx N_{\mathsf{m}} c_{\mathsf{m}} + N_{\mathsf{s}} c_{\mathsf{s}}$$

Then Criterion (8) is equivalent to $2 < k_o \neq k$. In this case, strict batch verification is useless unless it has three or more common exponentiation bases among all verification items. However, the gain $c_{\rm gain}$ is so large even if $k_o = 0$ that strict batch verification can be constructed if we find an efficient domain test for $h_0 \in G$. For example, if we use the binary method to calculate h_0^q, the loss can be estimated precisely as

$$c_{\rm loss} = (w_{\rm H}(q) - 1)\, c_{\mathsf{m}} + \lfloor \log_2 q \rfloor c_{\mathsf{s}}$$

where $w_{\rm H}(q)$ is the Hamming weight of q. In this case Criterion (10) is equivalent to

$$w_{\rm H}(q) - 1 < \frac{1}{\omega}(k_o - 1)\lfloor \log_2 q \rfloor$$

If we choose q as $w_{\rm H}(q) \ll \lfloor \log_2 q \rfloor/\omega$, then we can relax the necessary condition for strict batch verification[†], that is $1 < k_o$.

Because no fast domain test is available in this case, we must consider the special condition $k = k_o$. We can derive a miserable result, $4 < \omega + 2 < k$ from Criterion (11) for general q.

5.2 Subgroup of Z_p^* for Special Prime p

In this section, we examine the order-q subgroup of Z_p^* on condition that all prime factors of r are greater than q. We call such a prime, p, '*special prime.*' According to the results in the previous section, lenient batch verification is always faster than separate verification in this case. Therefore, we discuss only strict batch verification. We show that strict batch verification is always faster than separate verification in this case.

Under this special condition, we can substitute $h_0^{qr} = 1$ for $h_0^q = 1$, and clearly the Legendre symbol (h_0/p) is available for this purpose, in which the computational complexity is $O((\log_2 p)^2)$. We assume

[†]Clearly, it is sufficient that the (quasi)optimal q-th power costs much less than $c_{\rm gain}$, the necessary condition is independent of $w_{\rm H}(q)$.

that the Legendre symbol costs $\mathcal{L} \times c_{\mathsf{m}}$. Thus,

$$c_{\text{loss}} = \mathcal{L} \times c_{\mathsf{m}}$$

Then Criterion (10) is equivalent to

$$\mathcal{L} < \left(\frac{1}{\omega}(k_o - 1) + 1\right) \lfloor \log_2 q \rfloor$$

If $p \sim 2^{1024}$, then we experimentally observe $\mathcal{L} \sim 10$ (according to GP/PARI CALCULATOR). We can satisfy this inequality for practical p, q (e.g., $p \sim 2^{1024},\ q \sim 2^{160}$) even if $k_o = 0$ because $2 \leqq \omega$. In this case, we conclude that the strict batch verification is always faster than separate verification for practical p, q.

5.3 F_p-Rational Points on E/F_p ($3 <$ char F_p)

Let F_p be a finite field, we define an elliptic curve over F_p as

$$E/F_p : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \quad (a_i \in F_p) \qquad (12)$$

Let F_{p^m} be the extension of F_p of degree m. The group of F_{p^m}-rational points on E/F_p which we denote $E(F_{p^m})$, is defined as a set of solutions $(x, y) \in F_{p^m} \times F_{p^m}$ on E/F_p together with a point at infinity denoted by $\mathcal{O}$.

Groups of elliptic curves are classified by its definition field. In this section we assume $m = 1$. If p is a large prime, we call it '*prime field*.' If p is a prime power such that $3 <$ char F_p, we call it '*OEF*.' We discuss both of them here. Criterion (10) is available in this case. Therefore, we can recapitulate the results of Sect. 4.2 for lenient batch verification. Thus, we discuss only strict batch verification and show that it is always faster than separate verification.

Although h_0 must be an element of $E(F_p)$, an element of $E(F_p)$ is practically represented as $F_p \times F_p$. In this case, we can choose curves with a prime order, and all recommended curves over a prime field in [15] have a prime order. Here h_0 is given as $h_0 = (x, y) \in F_p^2$ and clearly all that is necessary is to check if the given point satisfies Curve equation (12). It costs less than only one elliptic addition on $E(F_p)$. It is sufficient for Criterion (10) if

$$1 < \left(\frac{1}{\omega}(k_o - 1) + 1\right) \lfloor \log_2 q \rfloor$$

When $q > 4$, this will always be satisfied because we hold $0 \leqq k_o$ and $2 \leqq \omega$. We conclude that strict batch verification is practically always faster than separate verification in this case. Note that if this check is omitted, attackers could easily cheat the system into performing operations beyond the group, and we should acknowledge that lenient batch verification is useless for any practical cryptographic applications in this case. If we choose compressed representation (See Appendix C), there is less of an advantage for the attacker than in the case of redundant representation, but they can choose x such that we cannot solve y on F_p. Some algorithms to calculate y may not detect this attack and give a false solution, in such a case we must check Curve equation (12).

5.4 Subgroup of F_{2^m}-Rational Points on E/F_{2^m}

On Curve equation (12), if p is a power of 2, we call the groups of elliptic curve '*binary field*,' which is the subject of this section. Criterion (10) is valid for this case, then we can apply the results of Sect. 4.2 for lenient batch verification. Therefore, we discuss only strict batch verification and show that it is always faster than separate verification.

In this case, it is not sufficient only to test Curve equation (12) if we want to ensure $h_0 \in G$, because all non-supersingular curves must have an even order. We can choose a curve whose order can be factorized into a 2 fold large prime and these types of curves have an effective large subgroup domain test. That is

$$T(x) \stackrel{?}{=} T(a_2) \qquad (13)$$

where $T(x)$ is the *trace* of x-coordinate and $T(a_2)$ is the trace of the curve parameter a_2. See [18] for details. All recommended random curves over F_{2^m} in [15] have this type of order. Slight calculations are necessary for this (and Curve equation (12)).

The situation is the same as in Sect. 5.3. We conclude that strict batch verification is always faster than separate verification.

5.5 Subgroup of F_{p^m}-Rational Points on E/F_p ($1 < m$)

On '*binary field*' and '*OEF*,' we can choose '*special curves*' called '*Koblitz curve*' [11], [12] to accelerate the calculation. In this section, we discuss such special curves and we show that lenient batch verification is not always faster than separate verification in this case. Afterwards, we propose an efficient domain test and investigate the conditions under which strict batch verification becomes faster than separate verification.

Let p be prime or a prime power. Let $E(F_{p^m})$ be a F_{p^m}-rational point group on E/F_p. Because $E(F_{p^m})$ must have a small order subgroup $E(F_p)$, the order of $E(F_{p^m})$ is essentially a composite. We must choose the curve parameters such that the curve has no small subgroups except $E(F_p)$. Therefore, m must be prime at least. All recommended curves over F_2 in [15] have a composite order which can be factorized into a $\#E(F_p)$ fold large prime. Let q be the large prime. Let G be an order-q subgroup of $E(F_{p^m})$.

The optimal elliptic exponentiation scheme on G

uses the Frobenius map ϕ which is a special constant multiplication that costs almost 0 [11], [12]. And many elliptic doublings are replaced by ϕ. This does not allow us to apply Criteria (10) because Observation (9) is not true in this case. Instead of Observation (9) we should apply

$$N_{\mathsf{m}} = \frac{1}{\omega}\lfloor \log_2 q \rfloor \ , \ \ N_{\mathsf{s}} = \lfloor \log_2 p \rfloor$$

in this case [11], [12]. Assume that $\log_2 q \sim (m-1) \times \log_2 p$, then

$$N_{\mathsf{s}} \approx \left\lfloor \frac{\log_2 q}{m-1} \right\rfloor$$

This means N_{s} becomes very small. If we suppose $c_{\mathsf{m}} \approx c_{\mathsf{s}}$ then $c_{\rm loss} < c_{\rm gain}$ is equivalent to

$$c_{\rm loss}/c_{\mathsf{m}} < \frac{1}{\omega}(k_o - 1)\lfloor \log_2 q \rfloor + \left\lfloor \frac{\log_2 q}{m-1} \right\rfloor \qquad (14)$$

We discuss lenient batch verification with the right side of Inequality (14) later in this section.

On the other hand we need the estimation of the loss for strict batch verification. In this case it is not sufficient to check Curve equation (12) if we want to ensure $h_0 \in G$ (however, it is still necessary). We must find an effective test to ensure that h_0 belongs to the order-q subgroup of $E(F_{q^m})$. We can apply the Frobenius map ϕ for this purpose and we propose an efficient domain test[†]. Assume that $m > 1$ and m is relatively prime to $\#E(F_p)$, then for given point P on the curve, $P \in G$ is equivalent to

$$\sum_{i=0}^{m-1} \phi^i P = \mathcal{O} \qquad (15)$$

(See Appendix A). In general, the computational complexity of a Frobenius map ϕ^i is at most as much as that of a multiplication on F_{p^m}, and can be much smaller if we choose a special representation for F_{p^m}. Therefore, the cost of the domain test is estimated as the number of necessary elliptic additions for (15). We obtain at most

$$c_{\rm loss}/c_{\mathsf{m}} \approx \lfloor \log_2 m \rfloor + w_{\rm H}(m) - 1$$

where $w_{\rm H}(m)$ is the Hamming weight of m (See Appendix B). According to the overlap factor, k_o, the advantage of lenient or strict batch verification becomes apparent.

$k_o = 0$: If $q \sim 2^{160}$ then the abstract window size, ω, is at most 5 or a similar value in this case. Therefore, if $\omega < m - 1$, then the right side of Inequality (14) will become negative. This means if $k_o = 0$, then lenient batch verification is not faster than separate verification on $E(F_{2^m})$ and most of $E(F_{p^m})$. Obviously strict batch verification is not faster than that.

$k_o = 1$: If $m > \log_2 q + 1$, then the right side of Inequality (14) becomes 0, and if m is comparable to $\log_2 q$ then the gain becomes almost 0. This means lenient batch verification is slower then separate verification on $E(F_{2^m})$. Furthermore, when $q \sim 2^{160}$, to satisfy Inequality (14) for strict batch verification, we need $m < 23$. The gain is so small that we do not recommend strict batch verification in this case.

$k_o \geqq 2$: The right side of Inequality (14) is always positive, and lenient and strict batch verifications are always faster than separate verification for a practical case.

[†]The trace test is also available for some special case of $q = 2$, but finally the same results are obtained.

References

[1] M. Abe, "Universally verifiable mix-net with verification work independent of the number of mix-servers," K. Nyberg, ed., Advances in Cryptology—EUROCRYPT'98, vol.1403 of Lecture Notes in Computer Science, pp.437–447, Springer-Verlag, Berlin; Heidelberg; New York, 1998.

[2] M. Abe, "Mix-networks on permutation networks," K. Lam, E. Okamoto, and C. Xing, eds., Advances in Cryptology—ASIACRYPT'99, vol.1716 of Lecture Notes in Computer Science, pp.258–273, Springer-Verlag, Berlin; Heidelberg; New York, 1999.

[3] M. Abe and F. Hoshino, "Remarks on mix-network based on permutation network," K. Kim, ed., Public Key Cryptography 4th International Conference, ISC 2001, vol.1992 of Lecture Notes in Computer Science, pp.317–324, Springer-Verlag, Berlin; Heidelberg; New York, 2001.

[4] M. Bellare, J.A. Garay, and T. Rabin, "Fast batch verification for modular exponentiation and digital signatures," K. Nyberg, ed., Advances in Cryptology—EUROCRYPT'98, vol.1403 of Lecture Notes in Computer Science, pp.236–250, Springer-Verlag, Berlin; Heidelberg; New York, 1998.

[5] C. Boyd and C. Pavlovski, "Attacking and repairing batch verification schemes," T. Okamoto, ed., Advances in Cryptology—ASIACRYPT2000, vol.1976 of Lecture Notes in Computer Science, pp.58–71, Springer-Verlag, Berlin; Heidelberg; New York, 2000.

[6] S. Brands, "Untraceable off-line cash in wallet with observers," D. Stinson, ed., Advances in Cryptology—CRYPTO'93, vol.773 of Lecture Notes in Computer Science, pp.302–318, Springer-Verlag, Berlin; Heidelberg; New York, 1993.

[7] H. Cohen, A. Miyaji, and T. Ono, "Efficient elliptic curve exponentiation using mixed coordinates," K. Ohta and D. Pei, eds., Advances in Cryptology—ASIACRYPT'98, vol.1514 of Lecture Notes in Computer Science, pp.51–65, Springer-Verlag, Berlin; Heidelberg; New York, 1998.

[8] F. Hoshino, M. Abe, and T. Kobayashi, "Lenient/strict batch verification in several groups," G.I. Davida and Y. Frankel, eds., Information Security 4th 2001, vol.2200 of Lecture Notes in Computer Science, pp.81–94, Springer-Verlag, Berlin; Heidelberg; New York, 2001.

[9] IEEE P1363/D13 (Draft Version 13). Standard Specifications for Public Key Cryptography Annex E (Informative) Formats, 1999. (available at http://grouper.ieee.org/groups/1363/P1363/draft.html).
[10] D.E. Knuth, Seminumerical Algorithms, vol.2 of The Art of Computer Programming, third ed., Addison Wesley, 1997.
[11] T. Kobayashi, H. Morita, K. Kobayashi, and F. Hoshino, "Fast elliptic curve algorithm combining Frobenius map and table reference to adapt to higher characteristic," J. Stern, ed., Advances in Cryptology—EUROCRYPT'99, vol.1592 of Lecture Notes in Computer Science, pp.176–189, Springer-Verlag, Berlin; Heidelberg; New York, 1999. (A preliminary version was written in Japanese and presented at SCIS'99-W4-1.4).
[12] N. Koblitz, "CM-curves with good cryptographic properties," J. Feigenbaum, ed., Advances in Cryptology—CRYPTO'91, vol.576 of Lecture Notes in Computer Science, pp.279–287, Springer-Verlag, Berlin; Heidelberg; New York, 1992.
[13] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, Handbook of applied cryptography, CRC Press, 1997.
[14] D. Naccache, D. M'Raïhi, S. Vaudenay, and D. Raphaeli, "Can D.S.A be improved?—Complexity trade-offs with the digital signature standard," A. De Santis, ed., Advances in Cryptology—EUROCRYPT'94, vol.950 of Lecture Notes in Computer Science, pp.77–85, Springer-Verlag, Berlin; Heidelberg; New York, 1995.
[15] NIST, Recommended Elliptic Curves for Federal Government Use, 1999. (available at http://csrc.nist.gov/csrc/fedstandards.html/).
[16] T. Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," E.F. Brickell, ed., Advances in Cryptology—CRYPTO'92, vol.740 of Lecture Notes in Computer Science, pp.31–53, Springer-Verlag, Berlin; Heidelberg; New York, 1993.
[17] K. Sako and J. Kilian, "Receipt-free mix-type voting scheme—A practical solution to the implementation of a voting booth," L.C. Guillou and J.-J. Quisquater, eds., Advances in Cryptology—EUROCRYPT'95, vol.921 of Lecture Notes in Computer Science, pp.393–403, Springer-Verlag, Berlin; Heidelberg; New York, 1995.
[18] C. Seroussi, Compact Representation of Elliptic Curve Points over F_{2^n}, Research Manuscript, Hewlett-Packard Laboratories, April 1998.
[19] S.-M. Yen and C.-S. Laih, "Improved digital signature suitable for batch verification," IEEE Trans. Comput., vol.44, no.7, pp.957–959, July 1995.

Appendix A

Why $\sum_{i=0}^{m-1} \phi^i P = \mathcal{O}$ is equivalent to $P \in G$?

Let ϕ be the Frobenius map on $E(F_{p^m})$. $\phi - 1$ is an endomorphism of $E(F_{p^m})$. According to the homomorphism theorem, we can decompose $E(F_{p^m})$ into

$$\mathrm{Ker}(\phi - 1) \oplus \mathrm{Im}(\phi - 1)$$

Note that $\mathrm{Ker}(\phi - 1) = E(F_p)$ and we define G as $\mathrm{Im}(\phi - 1)$, that is, for all $P \in E(F_{p^m})$, there exists $P_{\mathrm{Ker}} \in E(F_p)$ and $P_{\mathrm{Im}} \in G$ such that $P = P_{\mathrm{Ker}} + P_{\mathrm{Im}}$. Thus

$$\sum_{i=0}^{m-1} \phi^i P = mP_{\mathrm{Ker}} + \sum_{i=0}^{m-1} \phi^i P_{\mathrm{Im}} = mP_{\mathrm{Ker}}$$

since

$$\sum_{i=0}^{m-1} \phi^i P_{\mathrm{Im}} \in \mathrm{Im}(\phi^m - 1) = \{\mathcal{O}\}$$

Suppose that m is relatively prime to $\#E(F_p)$, then we obtain

$$P \in G \iff \sum_{i=0}^{m-1} \phi^i P = \mathcal{O}$$

Note that m must be a prime when the curve has no small subgroup except $E(F_p)$. Therefore, the condition is the same as $m \nmid \#E(F_p)$ for our purposes.

Appendix B

Optimal Addition Sequence of $\sum_{i=0}^{m-1} \phi^i P$

Let $a_0, ..., a_n \in N$ be an addition sequence of $m(= a_n)$ such that

$$a_0 = 1, \quad a_i = a_j + a_k, \ (j, k < i)$$

This yields the addition sequence of $\sum_{i=0}^{m-1} \phi^i P$ given by $A_0, ..., A_n \in E(F_{p^m})$ such that

$$A_0 = P, \quad A_i = \phi^{a_k} A_j + A_k, \ (j, k < i)$$

then $A_n = \sum_{i=0}^{m-1} \phi^i P$. This causes the computational complexity of A_n to be $n(m)$ elliptic additions, where $n(m)$ represents the length of the optimal addition chain of m. According to the binary addition chain of m, at most

$$n(m) \leqq \lfloor \log_2 m \rfloor + w_{\mathrm{H}}(m) - 1$$

See [10] for more a precise evaluation of $n(m)$.

Appendix C

Representation of Rational Point

In this appendix, we observe that the gain varies with the representation of a group if we use an elliptic curve.

Let h_0 be an element of $E(F_p)$, which is practically represented as $F_p \times F_p$ or $F_p \times \{0, 1\}$. We call the former *'redundant representation,'* and the latter *'compressed representation'* [9]. If we choose redundant representation, Approximation (7) and Criterion (8) are available for cost evaluation. On the other hand, Approximation (7) is not accurate if we choose the compressed representation. We estimate the gain in the compressed representation and show that the disadvantage is very small.

Since we need y or some replacement before we perform some group operations on h_0 and given h_0 has only x (and 1 bit for y) in compressed representation, we must solve y such that it satisfies Curve equation

(12) on F_p. Therefore, $c_{\rm gain}$ becomes smaller than Approximation (7), and unfortunately Criterion (10) is not available for this case. We must find a replacement for Criterion (10).

The dominant task of solving y is a square root on F_p or a similar calculation, which costs at most as much as one exponentiation on F_p. In general, we may suppose that it works about 10 times faster than the elliptic exponentiation (scalar multiplication) because elliptic curve addition and doubling both require about 10 multiplications on F_p. We assume that an exponentiation on F_p costs about $\epsilon_{\mathsf{m}} N_{\mathsf{m}} c_{\mathsf{m}} + \epsilon_{\mathsf{s}} N_{\mathsf{s}} c_{\mathsf{s}}$ where $\epsilon_{\mathsf{m}}, \epsilon_{\mathsf{s}} \lesssim 1/10$, because $p \sim q$ in this case. In general, the smaller the characteristic of F_p is, the smaller $\epsilon_{\mathsf{m}}, \epsilon_{\mathsf{s}}$ we obtain. Thus, we should substitute

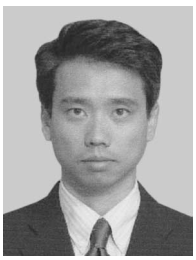
$$c_{\rm gain} \approx (k_c - 1 - \epsilon_{\mathsf{m}}) N_{\mathsf{m}} c_{\mathsf{m}} + (1 - \epsilon_{\mathsf{s}}) N_{\mathsf{s}} c_{\mathsf{s}}$$

for Approximation (7). We obtain the following instead of Criterion (10)

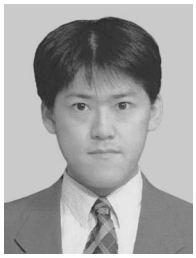
$$c_{\rm loss}/c_{\mathsf{m}} < \left(\frac{1}{w}(k_c - 1 - \epsilon_{\mathsf{m}}) + 1 - \epsilon_{\mathsf{s}} \right) \lfloor \log_2 q \rfloor$$

Thus, the necessary conditions for batch verification are not so different from those of Criterion (10) except for ϵ_{m} and ϵ_{s}. This representation is more suited to communications.

Fumitaka Hoshino received his B.Eng. and M.Eng. degrees from Tokyo University, Tokyo, Japan, in 1996 and 1998, respectively. He is a Researcher in the NTT Information Sharing Platform Laboratories.



Masayuki Abe received the M.E. degree from Science University of Tokyo in 1992. He is a senior research engineer of NTT Information Sharing Platform Laboratories. His research interests are cryptographic protocols and complexity theory. He is a member of IACR.



Tetsutaro Kobayashi received his B.Eng. and M.Eng. degrees from Tokyo Institute of Technology, Tokyo, Japan, in 1993 and 1995, respectively. He is a Researcher in the NTT Information Sharing Platform Laboratories. He is presently engaged in research on Information Security. He is interests lie in elliptic curve cryptosystems. He was awarded the SCIS'00 paper prize.