A New Baby-Step Giant-Step Algorithm and Some Applications to Cryptanalysis

Jean Sébastien Coron[1], David Lefranc[2] and Guillaume Poupard[3]

[1] Université du Luxembourg
Luxembourg
coron@clipper.ens.fr
[2] France Télécom
42 rue des Coutures
F-14066 Caen, France
david.lefranc@francetelecom.com
[3] DCSSI Crypto Lab
51 Boulevard de Latour-Maubourg
75700 Paris 07 SP, France
Guillaume.Poupard@m4x.org

Abstract. We describe a new variant of the well known Baby-Step Giant-Step algorithm in the case of some discrete logarithms with a special structure. More precisely, we focus on discrete logarithms equal to products in groups of unknown order. As an example of application, we show that this new algorithm enables to cryptanalyse a variant of the GPS scheme proposed by Girault and Lefranc at CHES 2004 conference in which the private key is equal to the product of two sub-private keys of low Hamming weight. We also describe a second attack based on a known variant of the Baby-Step Giant-Step algorithm using the low Hamming weight of the sub-private keys.

Key words: Baby-Step Giant-Step algorithm, discrete logarithm, GPS scheme, binary trees, low Hamming weight.

1 Introduction

In 1976, public key cryptography was introduced by Diffie and Hellman [2]. In their seminal paper, the authors originally explained how to use a mathematical assumption, namely *the discrete logarithm problem*, to obtain a key establishment protocol.

Since this first result, many identification schemes using the discrete logarithm problem have been proposed [4, 8, 15] and the security of this problem has been extensively studied (see [9] for a survey). Two major results are the *Baby-Step Giant-Step algorithm* due to Shanks [1] and the *rho method* due to Pollard [12]. These algorithms, used to recover discrete logarithms, are now the references to provide lower security bounds for the size of discrete logarithms since they apply as a generic method in any mathematical structure.

One of the main advantages of discrete-logarithm-based identification or signature schemes is that, when used with precomputations, they generally require only few computations for the prover or the signer so that such schemes are well designed for an integration in low cost chips. For example, in the well known Schnorr identification scheme [15], a prover using precomputations can be authenticated at the cost of one modular multiplication and one modular addition. On the opposite, in a RSA-based [14] identification schemes [3, 6], the prover usually has to compute at least one modular exponentiation.

Another attractive discrete-logarithm-based identification scheme is the GPS scheme introduced by Girault [4] and proved secure by Poupard and Stern [13]. Like the Schnorr scheme, if used with precomputations, the GPS scheme is very efficient since, during the execution of the protocol, the prover has only to compute one addition and one multiplication without any modular reduction. With the appearance of new technologies, like RFID tags or more generally very low cost chips, in which even a multiplication may be too difficult to compute, embedding cryptographic protocols in such devices is now becoming a new challenge. One solution may be the use of discrete-logarithm-based schemes since they already provide efficient solutions for low cost chips. However, to better the integration of such schemes in very low cost devices, improvements are generally required and two different approaches can be distinguished.

The first one consists in designing new schemes with new computation requirements. For instance, Okamoto, Tada and Miyaji [11] proposed a new identification scheme based on the discrete logarithm problem in which a prover using precomputations has only to compute one modular reduction and one addition. However, Stern and Stern [17] proved this scheme to be insecure with the suggested parameter sizes. In addition to this cryptanalysis, they also proposed a variant of the GPS identification scheme, based on a new operation called *dove-tailing*, which is more efficient than the classical GPS scheme. Finally, Okamoto, Katsuno and Okamoto [10] suggested another variant of the GPS identification scheme in which the original multiplication can be replaced by additions of several private keys.

The second approach consists in using existing schemes but with specific parameters. A classical example may be the use of a low Hamming weight (number of non zero bits in the binary representation) discrete logarithm to decrease the computation cost of the associated exponentiation. However, Stinson [18] proposed some Baby-Step Giant-Step variants for such discrete logarithms. At CHES 2004 conference, Girault and Lefranc [5] proposed some variants of the GPS identification scheme, well designed for an integration in RFID tags. One of these variants is based on the use of specific discrete logarithms equal to products of low Hamming weight numbers. For lack of security guarantees on such private keys, the authors recalled the state-of-the-art on the different Baby-Step Giant-Step algorithms and then checked that it was not efficiently applicable to their new type of private keys. In this paper, we present a new variant of Baby-Step Giant-Step algorithm to attack such private keys. We believe that this variant is also of independent interest.

This paper is organized as follows. After recalling some useful variants of the Baby-Step Giant-Step algorithm in Section 2, we describe in Section 3 our new algorithm for discrete logarithms equal to products of sub-private keys in groups of unknown order. In Section 4, we briefly recall the GPS scheme and the Girault-Lefranc private keys. Then we present two attacks on such private keys: the first one is an application of our new algorithm and the second one makes use of a known variant of the Baby-Step Giant-Step algorithm.

2 Baby-Step Giant-Step Algorithms

In this section, we recall Shanks' *Baby-Step Giant-Step algorithm* [1] and some useful variants. We first specify the notations we use.

Notations Let $G = \langle g \rangle$ be a finite cyclic abelian group generated by the element g and written multiplicatively. Let n be the order of G. As a consequence, we have $G = \left\{g^i;\ i \in [\![0, n-1]\!]\right\}$. For any value v in G, *the discrete logarithm* of v in base g, denoted $\log_g v$, is the unique non-negative integer x less than n such that $v = g^x$.

The *discrete logarithm problem* is to compute $\log_g v$ given g and v.

Let ℓ denote the value $\lceil \log_2 n \rceil$. Then, the binary representation of $x = \log_g v$ requires at most ℓ bits such that we can write

$$x = \sum_{i=0}^{\ell-1} x_i 2^i,$$

where $x_i \in \{0, 1\}$ for $0 \leq i \leq \ell - 1$. The *Hamming weight* of an integer x, denoted $wt(x)$, is equal to the number of 1's in its binary representation.

Let $t < \ell$ be a positive integer. The *Hamming weight t discrete logarithm problem* is to compute $\log_g v$ given g and v with the extra information $wt(\log_g v) = t$.

2.1 The Classical Baby-Step Giant-Step Algorithm

One of the most famous and generic algorithms dealing with the discrete logarithm problem is the so-called Baby-Step Giant-Step algorithm. Introduced by Shanks [1], it is a time-memory trade-off with time complexity $\mathcal{O}\left(\sqrt{n}\right)$ group multiplications.

The algorithm works as follows. Let $m = \lceil n^{1/2} \rceil$. For any given value $v \in \langle g \rangle$, $x = \log_g v$ is less than n so it can be written as $a + b \times m$ with a and b strictly less than m. From the equality $v = g^{\log_g v} = g^{a+b\times m}$, we obtain that $v \times g^{-bm} = g^a$ for some values a and b less than m. Thus, in the two following lists

$$\left(1, g, g^2, \ldots, g^{m-2}, g^{m-1}\right)$$

and

$$\big(v, vg^{-m}, vg^{-2m}, \ldots, vg^{-(m-2)m}, vg^{-(m-1)m}\big),$$

there exists at most two collisions, i.e two couples (g^{a_0},$vg^{-b_0 m}$) and (g^{a_1},$vg^{-b_1 m}$) such that $g^{a_0} = vg^{-b_0 m}$ and $g^{a_1} = vg^{-b_1 m}$. The value x is obtained using the couple with the smallest b_i and $x = a_i + b_i m$. The time complexity of this algorithm mainly relies on the computation of the lists both of which contains m elements. Thus, the time complexity of this algorithm is $\mathcal{O}\big(\lceil n^{1/2} \rceil\big)$ group multiplications. However, in this generic algorithm, the space requirement is also $\mathcal{O}\big(\lceil n^{1/2} \rceil\big)$.

In order to decrease such a large space requirement, Pollard [12] proposed two randomized variants of this algorithm, known as *rho* and *lambda* methods. The generic idea is two find a linear equation over $\log_g v$. The space requirement is then very small and the expected running time of these variants is still $\mathcal{O}\big(\lceil n^{1/2} \rceil\big)$ group multiplications.

2.2 Low Hamming Weight Discrete Logarithms

In 1999, Stinson described some variants [18] of the Baby-Step Giant-Step algorithm in the case of the Hamming weight t discrete logarithm problem, i.e the computation of discrete logarithms for which the Hamming weight is known to be t.

Without loss of generality, let us assume that $\ell = \lceil \log_2 n \rceil$ is even (otherwise we consider $\ell + 1$). This algorithm relies on the concept of *splitting system*.

Definition 1 (Splitting system). *Let t and ℓ be such that $0 < t < \ell$. A (ℓ,t)-splitting system is a pair (X,$\mathcal{B}$) that satisfies:*

- *$|X| = \ell$, and $\mathcal{B}$ is a set of subsets of X, each subset having $\ell/2$ elements.*
- *$\forall\ Y \subset X$ such that $|Y| = t$, $\exists\ B \in \mathcal{B}$ such that $|B \cap Y| = t/2$.*

For example, let t and ℓ be two even integers such that $0 < t < \ell$. Let $X = [\![0, \ell - 1]\!]$ and let $\mathcal{B} = \{B_i\ ;\ 0 \leq i \leq \ell/2 - 1\}$ where for all $0 \leq i \leq \ell/2 - 1$, $B_i = \{i + j \bmod \ell\ ;\ 0 \leq j \leq \ell/2 - 1\}$. The pair ($X$,$\mathcal{B}$) is a ($\ell$,$t$)-splitting system.

Thus, let $v \in \langle g \rangle$ such that $wt\big(\log_g v\big) = t$ (assumed to be even). We now use the above splitting system in the algorithm. Any element of $\langle g \rangle$ is now identified to the set of the positions of the non zero bits involved in the binary representation of its discrete logarithm. Thus, v is identified to a subset $Y \subset X$ of t elements. The goal of the algorithm is to find a decomposition of Y into two subsets of $t/2$ elements using the splitting system.

For all B_i in $\mathcal{B}$
- For all $Y_i^j \subset B_i$ of $t/2$ elements, identify the corresponding value A_i^j in G. Let $\mathcal{L}_1$ be the list of pairs (Y_i^j,A_i^j).
- Consider the set $W_i = [\![0, \ell - 1]\!] \setminus B_i$.

– For all $W_i^k \subset W_i$ of $t/2$ elements, identify the corresponding value B_i^k in G. Let $\mathcal{L}_2$ be the list of pairs $(W_i^k, v \times (B_i^k)^{-1})$.
– If two values $A_i^{j_0}$ in $\mathcal{L}_1$ and $v \times (B_i^{k_0})^{-1}$ in $\mathcal{L}_2$ meet for one given set $Y_i^{j_0}$ and one given set $W_i^{k_0}$, then output the element of $\langle g \rangle$ identified to $Y_i^{j_0} \cup W_i^{k_0}$; otherwise go on the loop over B_i.

The time complexity of Stinson's algorithm is $\mathcal{O}(\ell\binom{\ell/2}{t/2})$ group exponentiations[1] and the space requirement is $\mathcal{O}\big(\binom{\ell/2}{t/2}\big)$.

A detailed analysis of the number of group multiplications required to find the result in the worst case is $\ell/2 \times (t-1)\binom{\ell/2}{t/2}$ group multiplications.

2.3 Discrete Logarithms as Products in Groups of Known Order

This second variant of the Baby-Step Giant-Step algorithm focuses on discrete logarithms equal to products of integers. More precisely, it addresses the problem of computing the value $x = \log_g v$ for a given $v \in \langle g \rangle$, whenever $x = x_1 \times x_2 \bmod n$ with $x_1 \in X_1$, $x_2 \in X_2$. We denote by $|X_1|$ and $|X_2|$ the respective cardinalities of X_1 and X_2.

This variant is described in a technical report of Hoffstein and Silverman [7] and works as follows. From the equality $v = g^x = g^{x_1 \times x_2 \bmod n}$, we immediately obtain that

$$v^{x_2^{-1} \bmod n} = g^{x_1}.$$

As a consequence, in the two following sets $\big\{v^{j^{-1} \bmod n}; j \in X_1\big\}$ and $\big\{g^i; i \in X_2\big\}$, there exists at least one collision, i.e a same value obtained for one $v^{j_0^{-1} \bmod n}$ and one g^{i_0} such that x is equal to $j_0 \times i_0 \bmod n$.

The time complexity is $\mathcal{O}\big(|X_1| + |X_2|\big)$ group exponentiations since it mainly relies on the construction of two sets containing respectively $|X_1|$ and $|X_2|$ elements. In terms of number of group multiplications, without any assumption on the structure of the sets X_1 and X_2, the time complexity is $\mathcal{O}\big((|X_1| + |X_2|)\log_2 n\big)$ but this bound can be decreased for some specific choices of those sets. Finally, the smallest set must be stored so that the space complexity is $\mathcal{O}\big(\min(|X_1|, |X_2|)\big)$.

3 Discrete Logarithms as Products in Groups of Unknown Order

We now present a new variant of the Baby-Step Giant-Step algorithm that can be used to compute discrete logarithms equal to a product in a group of unknown order. As an application of this method, we propose a cryptanalysis in the next section.

[1] This algorithm can be turned into a randomized Las Vegas variant the time complexity of which is $\mathcal{O}\big(\sqrt{t}\binom{\ell/2}{t/2}\big)$.

3.1 Preliminaries

In this section, we consider a finite cyclic abelian group $G = \langle g \rangle$, written multiplicatively, of unknown order n. Let X_1 and X_2 be two sets of integers, the problem we address is to compute the discrete logarithm $x = \log_g v$ for a given value $v \in G$, whenever $x = x_1 \times x_2 \bmod n$ with $x_1 \in X_1$, $x_2 \in X_2$. We denote by $|X_1|$ and $|X_2|$ the respective cardinalities of the two sets X_1 and X_2.

The variant of the Baby-Step Giant-Step suggested by Hoffstein and Silverman and recalled in Section 2.3 cannot be applied, since it requires modular inversion modulo the unknown order n of the group. Our new variant enables us to overcome this problem.

3.2 Overview of the Full Algorithm

We first present the general method used in the algorithm. As for the second variant of the baby-Step Giant-Step algorithm presented in Section 2.3, we first consider the general equation in G

$$v^{x_1^{-1} \bmod n} = g^{x_2}. \tag{1}$$

As explained above, considering directly this equation is no longer interesting since computing inverses modulo the unknown order is not feasible.

However, we can use a trick that has for example already been used by Shoup [16]. We denote

$$\pi_1 = \prod_{i \in X_1} i,$$

and we raise Equation 1 to the power π_1, so that we obtain $v^{\pi_1 x_1^{-1} \bmod n} = g^{\pi_1 x_2}$ which can be rewritten as:

$$v^{\prod_{i \in X_1 \setminus \{x_1\}} i} = g^{\pi_1 \times x_2} \tag{2}$$

With this new equation, the knowledge of the order of g is no longer necessary. We can now use the classical method like in the other Baby-Step Giant-Step algorithms. More precisely, in a first time we can compute the two following sets

$$\mathcal{S}_1 = \left\{ v^{\prod_{k \in X_1 \setminus \{i\}} k}; \forall i \in X_1 \right\} \text{ and } \mathcal{S}_2 = \{ g^{\pi_1 \times j}; \forall j \in X_2 \}.$$

In the two sets, two values meet for one value $v^{\prod_{k \in X_1 \setminus \{i_0\}} k}$ and one value $g^{\pi_1 j_0}$ such that x is equal to $i_0 \times j_0$.

However, we must be careful with the actual time complexity of this algorithm. More precisely, let us evaluate the time complexity of the construction of the set $\mathcal{S}_1$; in the different Baby-Step Giant-Step algorithms recalled in Section 2, the computation of each element of the relative sets requires (at most) a modular inversion and a group exponentiation. In our new algorithm, the value π_1 cannot be reduced modulo the unknown order n so that, computing each element in a classical way is equivalent to computing $(|X_1| - 1)$ group exponentiations with $\log_2 n$ bits exponents. Thus the naïve computation of the full set

$\mathcal{S}_1$ has a time complexity $\mathcal{O}\big(|X_1|^2\big)$ group exponentiations. Taking into account the time complexity required for the computation of $\mathcal{S}_2$, we finally obtain a time complexity $\mathcal{O}\big(|X_1|^2 + |X_2|\big)$ group exponentiations, i.e $\mathcal{O}\big((|X_1|^2 + |X_2|)\log_2 n\big)$ group multiplications.

Since the complexity of the exhaustive search over all the possible $x_1 \in X_1$ and $x_2 \in X_2$ is obviously in $\mathcal{O}\big(|X_1| \times |X_2|\big)$ group exponentiations, the practical gain of our algorithm may not be significant (for instance if $|X_1| \approxeq |X_2|$), with a naïve computation of $\mathcal{S}_1$.

Note that there is no problem with the computation of the set $\mathcal{S}_2$ since after computing one time g^{π_1}, each element of $\mathcal{S}_2$ can be obtained using a single group exponentiation.

Let us now present how to construct efficiently the set $\mathcal{S}_1$.

3.3 Efficient Construction of $\mathcal{S}_1$

The algorithm. For a better overview of the construction, we consider in the following a set X of 2^q elements denoted by x_i, $i \in [\![1, 2^q]\!]$ for which we want to obtain the set of values

$$\mathcal{S} = \big\{ v^{\prod_{x_i \in X \setminus \{x_j\}} x_i};\ \forall j \in [\![1, 2^q]\!] \big\}.$$

The method we present relies on an implicit binary tree structure. The algorithm starts from the root equal to v and it ends with 2^q leafs equal to the elements of $\mathcal{S}$. All the nodes can be computed using the following algorithm:

$node_{(0,0)} := v$
For $i \in [\![0, q-1]\!]$
  For $j \in [\![0, 2^i - 1]\!]$

`(Left Son)`

$$exp_L := \prod_{k=1+(2j+1)(2^{q-i-1})}^{(2j+2)(2^{q-i-1})} x_k$$

$$node_{(i+1,2j)} = (node_{(i,j)})^{exp_L}$$

`(Right Son)`

$$exp_R := \prod_{k=1+2j(2^{q-i-1})}^{(2j+1)(2^{q-i-1})} x_k$$

$$node_{(i+1,2j+1)} = (node_{(i,j)})^{exp_R}$$

This algorithm iteratively computes

$$node(i,j) = v^{\left(\prod_{k=1}^{j \times 2^{q-i}} x_k\right) \times \left(\prod_{\ell=1+(j+1)2^{q-i}}^{2^q} x_\ell\right)}$$

for $i \in [\![0, q]\!]$ and $j \in [\![0, 2^i - 1]\!]$, i.e v to the power the product of all the $x_k \in X$, but a "gap" of 2^{q-i} consecutive elements. This property is easily proved using a recursive argument. As a consequence, for all $i \in [\![0, 2^q - 1]\!]$ the leaf $node_{(q,i)}$, is equal to the value $v^{\prod_{x_j \in X \setminus \{x_{i+1}\}} x_j}$.

Example. Let $q = 2$. Using the algorithm for the first generation, the left son of the root, denoted by $node_{(1,0)}$ is equal to $v^{x_3 x_4}$ and the right son, denoted $node_{(1,1)}$, is equal $v^{x_1 x_2}$. The second generation is described in Fig. 1.

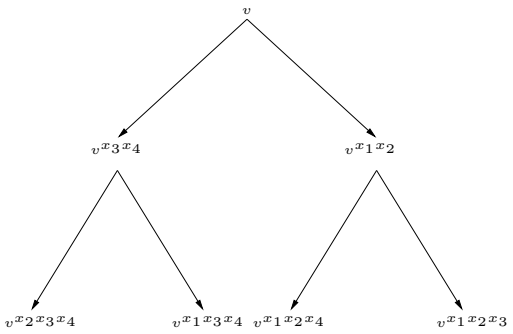

Fig. 1. Iteration of the algorithm for $q = 2$

Complexity of the construction. The advantage of our algorithm is twofold:

1. Once the two sons $node_{(i+1,2j)}$ and $node_{(i+1,2j+1)}$ of $node_{(i,j)}$ are computed, the $node_{(i,j)}$ is no longer required, so that it can be erased. Thus, as mentioned previously, the algorithm uses a binary tree structure but does not require the storage of the entire tree. As a consequence, the space requirement during the algorithm execution is optimal, i.e equal to the space required for the storage of the set $\mathcal{S}$.
2. The different exponents x_k are used only once during each loop over i. Thus, the time complexity (in terms of group exponentiations) is equal to $\mathcal{O}\big(|X| \log_2 |X|\big)$ as there are exactly q loops involving $|X|$ group exponentiations. This complexity can also be obtained considering the overall number of group exponentiations given by:

$$\sum_{i=0}^{q-1} \sum_{j=0}^{2^i-1} (2 \times 2^{q-i-1}) = 2 \sum_{i=0}^{q-1} (2^{q-1}) = q \times 2^q$$

In terms of group multiplications complexity, without additional assumption on the structure of X, all the exponents we use have $\log_2 n$ bits so the complexity is $\mathcal{O}\big(|X| \log_2 |X| \log_2 n\big)$ group multiplications.

3.4 Complexity of the Full Algorithm

Our new variant of the Baby-Step Giant-Step algorithm described in Section 3.2, is based on the computation of two sets $\mathcal{S}_1$ and $\mathcal{S}_2$. Using the method of Section 3.3, the computation of the set

$$\mathcal{S}_1 = \left\{v^{\prod_{k \in X_1 \setminus \{i\}} k}; \forall i \in X_1\right\}$$

has time complexity $\mathcal{O}\Big(|X_1| \log_2 |X_1| \log_2 n\Big)$ group multiplications and the computation of

$$\mathcal{S}_2 = \{g^{\pi_1 \times j}; \forall j \in X_2\}$$

has time complexity $\mathcal{O}\Big(|X_2| \log_2 n\Big)$ group multiplications. Thus, the overall time complexity, for computing the two sets, is $\mathcal{O}\Big(\big(|X_2| + |X_1| \log_2 |X_1|\big) \log_2 n\Big)$.

Finally, finding two identical values in the two sets can be done efficiently (for example if one set is sorted), so that the time complexity of the algorithm mainly relies on the construction of the sets. Thus, the time complexity of the full algorithm is also $\mathcal{O}\Big(\big(|X_2| + |X_1| \log_2 |X_1|\big) \log_2 n\Big)$ group multiplications.

4 Attacks on GPS with Private Keys From CHES'04

In this section, we briefly recall the basic GPS scheme [4, 13]. Next, we recall the private keys suggested by Girault and Lefranc at CHES 2004 conference and we finally present two attacks on such private keys. The first one relies on our new algorithm from Section 3 taking advantage of a first weakness of the private key and the second attack relies on a second weakness of the private keys.

4.1 The GPS Scheme

We denote by $\mathbb{Z}_n$ the residue class ring modulo n and $\mathbb{Z}_n^*$ the multiplicative group of invertible elements in $\mathbb{Z}_n$. The GPS identification scheme from [4, 13], is an interactive protocol between a prover and a verifier which contains one or several rounds of three passes. The security is based on the intractability of *the short discrete logarithm problem* defined as follows.

Definition 2. *Let n be a composite integer the factorization of which is unknown. Let g be an element in $\mathbb{Z}_n^*$ of maximal order $\lambda(n)$. Let S be an integer such that $2^S < \lambda(n)$. The short discrete logarithm problem consists in computing the value $s \in [\![0, 2^S[\![$ given $v = g^s \bmod n$.*

Assumption. *The short discrete logarithm problem is polynomially intractable.*

During a round of identification, a prover uses his knowledge of a private value s related to the public value v by the equation $v = g^{-s} \bmod n$. More

precisely, in typical applications, a prover holds a private key s and a public key (n,g,v) such that:

- $n = pq$ is the product of two prime integers such that factoring n is difficult,
- g is an element of $\mathbb{Z}_n^*$ of maximal order $\lambda(n)$,
- $v = g^{-s} \bmod n$.

There are four security parameters S, k, R and m defined as follows:

- S is the binary size of the private key s; $S = 160$ is a typical choice.
- k is the binary size of the challenges sent to the prover and determines the level of security of the scheme.
- R is the binary size of the exponents used in the commitment computation. It typically verifies $R = S + k + 80$.
- m is the number of rounds the scheme is iterated. Theoretically, m is polynomial in the size of the security parameter; but, in practice, m is often chosen equal to 1.

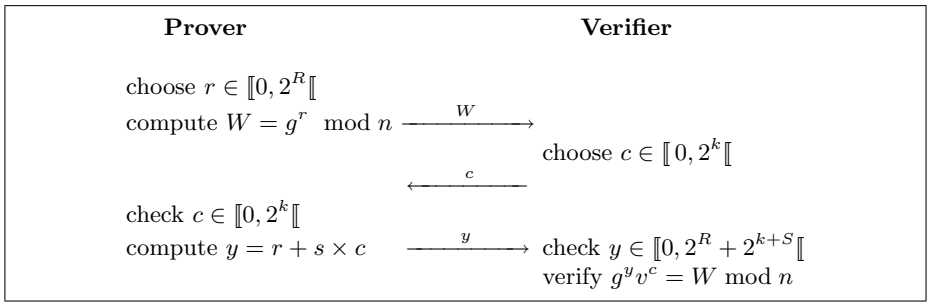

Fig. 2. The basic GPS identification scheme

Security of the scheme. The security of the GPS scheme is recalled in the following Theorem (the proof is given in [13]).

Theorem 1. *The GPS identification scheme is a secure identification scheme under the intractability of the short discrete logarithm problem if m and 2^k are polynomial in $|n|$, $m \times 2^{S+k-R}$ is negligible in $|n|$ and $\log |n| = o(m \times k)$.*

4.2 The Girault-Lefranc Private Keys

As many other discrete-logarithm-based schemes, the GPS identification scheme, used with precomputations of the commitments $W = g^r \bmod n$, is a very efficient scheme for the prover. Thus, during the protocol, the computations of the prover can be reduced to the computation of the value $y = r + sc$ so that it is well

designed for low cost chips. However, in new chips like RFID tags, even computing a multiplication may be too expensive. Thus, at the CHES 2004 conference, Girault and Lefranc [5] proposed three solutions to make easier the integration of the GPS identification scheme in such chips. Their goal was to remove (or at least reduce) the computation requirement for $y = r + sc$.

One of these solutions consists in using specific private keys equal to the product of low Hamming weight sub-private keys. More precisely, they consider a private key s equal to $s_1 \times s_2$ with s_1 in X_1 and s_2 in X_2 such that both s_1 and s_2 have a low Hamming weight.

With such private keys, the computation of $s \times c$ is then replaced successively by the computation of $s_2 \times c$ and $s_1 \times (s_2 c)$. Thus, since the Hamming weight of the sub-private keys s_1 and s_2 is low, the computation of y using the shift-and-add paradigm does not involve too many shifts.

Security of such Private keys. During this analysis, we require a security over the private key of 2^{80} group multiplications. However, the level of security is highly application dependant and some lower levels can be accepted.

The authors give some general security arguments on such private keys. Indeed, they suggest the use of sub-private keys s_1 and s_2 such that:

1. the private key $s = s_1 s_2$ is sufficiently large, i.e around 160-bit, such that the classical Baby-Step Giant-Step has a time complexity of around 2^{80} group multiplications;
2. the average Hamming weight of $s = s_1 s_2$ is around 64 such that Stinson's algorithm requires more than 2^{80} group multiplications.

Finally, since the group generator g is of unknown order $\lambda(n)$, the second variant of the Baby-Step Giant-Step algorithm recalled in Section 2.3 cannot be used. Thus, Girault and Lefranc consider that the best attack on such private keys is the exhaustive search over s_1 and s_2 which time complexity is obviously in $\mathcal{O}\big(|X_1| \times |X_2|\big)$ group exponentiations, where $|X_1|$ and $|X_2|$ denote the respective cardinalities of X_1 and X_2. Thus, they suggest to consider two adequate sets X_1 and X_2 such that $|X_1| \times |X_2| \approxeq 2^{80}$.

Numerical Application. In [5], Girault and Lefranc give the following example. To obtain a 160-bit private key s with an Hamming weight equal on average to 64, s_2 should be a 142-bit number with 16 random bits equal to 1 chosen among the 138 least significant ones and s_1 a 19-bit number with 5 random bits equal to 1 chosen among the 16 least significant ones.

With such sub-private keys, the cardinality of X_1 is equal to $\binom{16}{5} \approxeq 2^{12}$ and the cardinality of X_2 is equal to $\binom{138}{16} \approxeq 2^{68}$; so that the exhaustive search is at least as infeasible as the classical Baby-Step Giant-Step algorithm for a 160-bit key.

Then, if we assume that c is a 32-bit number, computing $r + s \times c$ in a classical way involves on average 16 additions of a 160-bit numbers, i.e 2560 bit additions.

Using the structure of the private key equal to a product of low Hamming weight sub-private keys, then computing $s_2 \times c$ requires exactly 5 additions of a 32-bit number, then $s_1 \times (s_2 c)$ requires 16 additions of a 51-bit number and adding $s_1(s_2 c)$ to r requires a final addition of a 192-bit number. Finally, only 1168 bit additions are required.

4.3 Two Attacks

This first attack relies on our new Baby-Step Giant-Step algorithm described in Section 3. Thus, to prove the efficiency of our algorithm, we apply it to the numerical application given in Section 4.2.

In this example, we recall that the two sets X_1 and X_2 are of cardinalities respectively upper bounded by 2^{12} and 2^{68}. We also recall that the time complexity of our new algorithm is $\mathcal{O}\big(|X_2| + |X_1| \log_2 |X_1|\big)$ group exponentiations. Thus, applied to the given example, our algorithm recovers the private key s with about 2^{68} group exponentiations, which is significantly less than the complexity of the exhaustive search equal to 2^{80} group exponentiations.

A detailed enumeration of the number of group multiplication shows that the computation of the set S_2 in the algorithm of section 4.2 has the highest complexity; using precomputation in the usual square and multiply exponentiation algorithm leads to a complexity of 2^{73} group multiplications (the exhaustive search requires around 2^{84} group multiplications).

Whereas our first attack only takes advantage of the structure of product of the private keys, we now describe a second attack which takes advantage of both the product structure and the low Hamming weight of the sub-private keys.

Indeed, from the basic equation $v = g^s = g^{s_1 \times s_2}$, and denoting g^{s_1} by h, we then obtain the new equation

$$v = h^{s_2}.$$

With this change of base, the discrete logarithm of v in base h is a low Hamming weight number so that Stinson's algorithm can now be easily applied. The attack consists in using Stinson's algorithm (see section 2.2) for all possible bases h defined as g^i for any i in X_1. The complexity of this attack is then obviously $\mathcal{O}\big(|X_1| \times \ell \binom{\ell/2}{t/2}\big)$ group exponentiations, ℓ denoting the binary size of s_2 and t its Hamming weight. Note that the space requirement of this attack is the same as the one of Stinson's algorithm, i.e $\mathcal{O}(\binom{\ell/2}{t/2})$.

In the numerical application of section 4.2, we recalled that $|X_1| \approxeq 2^{12}$ and that s_2 is a 138-bit number with a Hamming weight equal to 16. As $\binom{138/2}{16/2} \approxeq 2^{33}$, we recover the private key with about 2^{52} group exponentiations.

A detailed analysis of the exact number of group multiplications required by this attack shows that the private key recovery requires 2^{54} group multiplications in the worst case.

To keep using Girault-Lefranc private keys, we suggest to consider new types of sub-private keys. Indeed, s_1 should be a 30 bit number with 12 non zero bits and s_2 should be a 130-bit number with 26 non zero bits.

With such sub-private keys, our first algorithm requires around 2^{94} group multiplications and the complexity of the second attack is then around 2^{80} group multiplications. However, the computation advantage of the method is obviously decreased. Using the same consideration as in the numerical application of section 4.2, the number of bit additions is then equal to 2100; the practical gain is less significant, specially with small challenges c.

5 Conclusion

We have proposed a new variant of the Baby-Step Giant-Step algorithm for discrete logarithms equal to products in groups of unknown order. More precisely, our algorithm recovers $x = x_1 \times x_2$ with $x_1 \in X_1$, $x_2 \in X_2$ from the given value g^x in time $\mathcal{O}\big((|X_2| + |X_1| \log_2 |X_1|) \log_2 n\big)$ group multiplications.

This new variant finds a direct application with the GPS scheme used with such private keys as described by Girault and Lefranc at CHES 2004 conference. Thus, whereas the time complexity of the best known attack (the exhaustive search) on such private key was 2^{84} group multiplications, using our new algorithm, this complexity falls downto 2^{73} group multiplications. Moreover, using the fact that such private keys require sub-private keys of low Hamming weight, we have constructed a second attack the time complexity of which is 2^{54} group multiplications.

Acknowledgements

The authors wish to thank Aline Gouget and Marc Girault for valuable and helpful discussions and comments.

References

1. H. Cohen. *A Course in Computational Algebraic Number Theory*, volume 138 of *Graduate Texts in Mathematics*. Springer-Verlag, 1993.
2. W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.
3. A. Fiat and A. Shamir. How to Prove Yourself : Practical Solutions to Identification and Signature Problems. In A. M. Odlyzko, editor, *Advances in Cryptology - Crypto '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer Verlag, 1986.
4. M. Girault. Self-Certified Public Keys. In D. W. Davies, editor, *Advances in Cryptology - Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer-Verlag, 1991.
5. M. Girault and D. Lefranc. Public Key Authentication with one Single (on-line) Addition. In M. Joye and J. J. Quisquater, editors, *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 413–427. Springer-Verlag, 2004.

6. L. C. Guillou and J. J. Quisquater. A Practical Zero-knowledge Protocol Fitted to Security Microprocessor Minimizing both Transmission and Memory. In C. G. Günther, editor, *Advances in Cryptology - Eurocrypt '88*, volume 330 of *Lecture Notes in Computer Science*, pages 123–128. Springer-Verlag, 1988.
7. J. Hoffstein and J.H. Silverman. Random Small Hamming Weight Products with Applications to Cryptography. Technical report, NTRU Cryptosystems.
8. National Institute of Standards and Technologies. Digital Signature Standard (DSS). Federal Information Processing Standards, Publication 186, november 1994.
9. A. M. Odlyzko. Discrete Logarithms: The Past and The Future. *Designs, Codes, and Cryptography*, 19(2/3):129–145, 2000.
10. T. Okamoto, H. Katsuno, and E. Okamoto. A Fast Signature Scheme based on new on-line Computation. In C. Boyd and W. Mao, editors, *Information Security Conference '02*, number 2851 in Lecture Notes in Computer Science, pages 111–121. Springer-Verlag, 2003.
11. T. Okamoto, M. Tada, and A. Miyaji. An Improved Fast Signature Scheme without on-line Multiplication. In M. Blaze, editor, *Financial Crypto*, volume 2357 of *Lecture Notes in Computer Science*. Springer-Verlag, 2003.
12. J. M. Pollard. Monte Carlo Methods for Index Computations (mod p). *Mathematics of Computation*, 32(143):918–924, 1978.
13. G. Poupard and J. Stern. Security Analysis of a Practical "on the fly" Authentication and Signature Generation. In K. Nyberg, editor, *Advances in Cryptology - Eurocrypt '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 422–436. Springer-Verlag, 1998.
14. R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communication of the ACM*, 21(2):120–126, 1978.
15. C. P. Schnorr. Efficient Identification and Signatures for Smart Cards. In G. Brassard, editor, *Advances in Cryptology - Crypto '89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer-Verlag, 1990.
16. V. Shoup. Practical Threshold Signatures. In B. Preneel, editor, *Advances in Cryptology - Eurocrypt '00*, volume 1807 of *Lecture Notes in Computer Science*, pages 207–220. Springer-Verlag, 2000.
17. J. Stern and J. P. Stern. Cryptanalysis of the OTM Signature Scheme from FC'02. In R. N. Wright, editor, *Financial Cryptography '03*, volume 2742 of *Lecture Notes in Computer Science*, pages 138–148. Springer-Verlag, 2003.
18. D. R. Stinson. Some Baby-Step Giant-Step Algorithms for the Low Hamming Weight Discrete Logarithm Problem. *Mathematics of Computation*, 71(237):379–391, 2002.