이학 박사 학위논문

# Discrete Logarithm with Low Hamming Weight Exponents

## (성긴 지수 이산대수 문제)

2012년 8월

서울대학교 대학원

수리과학부

김성욱

# Discrete Logarithm with Low Hamming Weight Exponents

A dissertation

submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

to the faculty of the Graduate School of

Seoul National University

by

**Sungwook Kim**

Dissertation Director : Professor Jung Hee Cheon

Department of Mathematical Sciences

Seoul National University

August 2012

## Abstract

# Discrete Logarithm with Low Hamming Weight Exponents

Sungwook Kim

Department of Mathematical Sciences

The Graduate School

Seoul National University

The discrete logarithm problem is one of the most important underlying mathematical problems in contemporary public key cryptography. Under the assumption that the problem is infeasible, a great number of cryptosystems have been constructed and researches in this area are still underway actively.

The efficiency of cryptosystems based on the discrete logarithm problem primarily relies on the speed at which exponentiation can be performed. On this line of research to address the issue Hoffstein and Silverman suggested the use of low Hamming weight product exponents to accelerate group exponentiation while maintaining the security level. Taking low Hamming weight product exponents, computation costs on $\mathrm{GF}(2^n)$ or Koblitz elliptic curves can be reduced significantly, where the cost of squaring and elliptic curve doubling is much lower than that of multiplication and elliptic curve addition, respectively.

In the thesis we focus our concern on the security analysis of the discrete logarithm problem of low Hamming weight product exponents. The current estimate on the security of the problem mainly depends on the approaches

for the case of low Hamming weight exponents, which does not fit into the product form well.

We come up with parameterized splitting systems to resolve this problem. We show that it yields an efficient algorithm for the discrete logarithm problem of low Hamming weight exponents with lower complexity than that of any previously known algorithms.

To demonstrate its application, we attack the GPS identification scheme modified by Coron, Lefranc, and Poupard in CHES 2005 and Hoffstein and Silverman's (2,2,11)-exponents. The time complexity of our key recovery attack against the GPS scheme is $2^{61.82}$, which was expected to be $2^{78}$. Hoffstein and Silverman's (2,2,11)-exponent can be recovered with a time complexity of $2^{53.02}$, which is the lowest among the known attacks.

**Key words:** Discrete Logarithm Problem, Low Hamming Weight Discrete Logarithm Problem, Low Hamming Weight Product Discrete Logarithm Problem, Parameterized Splitting Systems

**Student Number:** No. 2005-20314

# Contents

**Abstract** **i**

**1 Introduction** **1**

1.1 Notation . . . . . 5

**2 The Low Hamming Weight Discrete Logarithm Problem** **6**

2.1 The Discrete Logarithm Problem (DLP) . . . . . 7

2.2 The Low Hamming Weight DLP (LHW-DLP) . . . . . 9

2.3 Algorithms for The LHW-DLP . . . . . 10

2.3.1 Heiman-Odlyzko's Algorithm . . . . . 10

2.3.2 Coppersmith's Algorithm . . . . . 12

**3 The Low Hamming Weight Product DLP** **15**

3.1 The Low Hamming Weight Product DLP (LHWP-DLP) . . . 16

3.1.1 The Efficiency of LHWP Exponents . . . . . 16

3.1.2 The Definition of LHWP-DLP . . . . . 18

3.2 Existing Algorithms for LHWP-DLP . . . . . 19

3.2.1 Attack using Coppersmith's Splitting System . . . . . 19

3.2.2 Rotation-Free Elements . . . . . 21

**4 Parameterized Splitting Systems** **24**

4.1 Parameterized Splitting Systems . . . . . 25
4.1.1 The Construction . . . . . 25
4.2 A Randomized Version . . . . . 28

**5 A New Algorithm from Parameterized Splitting Systems 32**
5.1 When The Order of a Group is Known . . . . . 33
5.1.1 Motivations . . . . . 33
5.1.2 Using The Parameterized Splitting Systems . . . . . 34
5.2 When the Order of a Group is Unknown . . . . . 38
5.2.1 The Basic Approach . . . . . 38
5.2.2 Precomputation . . . . . 38
5.2.3 Complexity of the Full Algorithm . . . . . 40

**6 Cryptanalysis 43**
6.1 GPS identification Scheme . . . . . 44
6.1.1 The Scheme . . . . . 44
6.1.2 The LHWP Private Keys . . . . . 47
6.1.3 Cryptanalysis . . . . . 47
6.2 Hoffstein and Silverman's Exponents . . . . . 50
6.2.1 Silverman and Hoffstein's exponents . . . . . 50
6.2.2 Cryptanalysis . . . . . 50
6.3 Implementation . . . . . 52

**7 Conclusion and Open Problems 54**

**Abstract (in Korean) 61**

# Chapter 1

# Introduction

Let $g$ be a generator of a finite cyclic group $G$ of order $m$. Given $g$ and $h = g^x \in G$, the *discrete logarithm problem* (DLP) is to compute $x \in [0, m-1]$, which is denoted by $\log_g h$. The DLP is one of the most important underlying mathematical problems in cryptographic applications. The security of many of the current cryptosystems and cryptographic protocols is based on the hardness of the DLP.

The efficiency of DL-based cryptosystems primarily relies on the speed at which exponentiation can be performed. One approach to achieve fast exponentiation is to use integers of low Hamming weight (LHW) as secret exponents [AMOV91], because the number of multiplications required for an exponentiation depends on the weight of the exponent. However, more efficient attacks on the DLP with LHW exponents have been proposed by Heiman-Odlyzko [Hei92] and then by Coppersmith [CS84, MvOV97] and so the advantage of LHW exponents becomes insignificant. In fact, the time complexity, which means the number of required group operations, of Coppersmith's algorithm is about the square root of the size of the key space. It can be regarded as almost optimal in the sense that the complexity of the

DLP on a group is lower bounded by the square root of the group order in the generic group model [Sho97].

To resist previous attacks and achieve a greater speed-up, Hoffstein and Silverman suggested the use of low Hamming weight product (LHWP) exponents [HS03]. This was then applied to the GPS identification scheme, recommended by the NESSIE project [GPS02], in which a secret key is taken as a product of two integers having low Hamming weights [GL04, CLP05]. In a general manner, this type of the DLP is a form of $h = g^{xy}$, where $x$ is an integer of length $n$ and Hamming weight $t$, and $y$ is an element of a set $Y$. The essential part of the attack for this exponent is to split $x$ into the sum of $u$ and $v$ and then apply the meet-in-the-middle technique for $h^{y^{-1}}g^{-u} = g^v$ so that the number of group operations required to compute the left-hand side of the above equation is almost equal to that of right-hand side. However, the splitting of $x$ in Heiman-Odlyzko's or Coppersmith's algorithm has a fixed length $n$ or a fixed weight $t/2$, respectively, and thus does not fit into this situation.

### Our Results

In the thesis, we propose a more flexible splitting system, called a *parameterized splitting system*. It can be regarded as a generalization and refinement of Coppersmith's splitting system: given a bit string of length $n$ and weight $t$ and any positive integer $t_1 < t$, there exists a part of the string of length $n_1$ and weight $t_1$ where $\frac{n_1}{t_1} \approx \frac{n}{t}$. By exploiting this property, given an $n$-bit integer $x$ one can find an $n_1$-bit integer $u$ and an $(n-n_1)$-bit integer $v$ of weight $t_1$ and $t - t_1$, respectively, such that $u + v2^{n_1} \equiv x2^k \pmod{2^n - 1}$ for some integer $k$. Furthermore, it has an additional property while Coppersmith's splitting system does not: when we split $x$ into $u$ and $v$, we can take an *odd*

$u$ while maintaining other properties, which reduces the attack complexity further.

We apply a parameterized splitting system to the private key of the GPS identification scheme in [CLP05] and [GL04] and to Hoffstein and Silverman's (2,2,11)-exponent in [HS03], both of which are originally designated for 80-bit security. In CHES 2005, Coron, Lefranc, and Poupard proposed an attack with $2^{52}$ complexity to recover the private key of the modified GPS identification scheme from CHES 2004 and suggested a new private key that they claimed had a security level of $2^{78}$ [CLP05]. But our parameterized splitting system reduces them to $2^{45.57}$ and $2^{64.53}$, respectively, and its randomized version reduces them to $2^{44.57}$ and $2^{61.82}$, respectively. In [CK08], Cheon and Kim introduced the notion of rotation-free elements and proposed an attack with $2^{55.9}$ group exponentiations to Hoffstein and Silverman's (2,2,11)-exponent. We reduce it further to $2^{53.02}$ by combining parameterized splitting systems and the notion of rotation-freeness.

### Outline

We now present the organization of the thesis. The thesis is divided into two parts. The 1st part (Chapters 2 and 3) deals with some backgrounds of topics which are concerns of the thesis.

In Chapter 2, we describe the LHW-DLP and its motivation. Next we briefly give an overview on two algorithms for solving LHW-DLP, Heiman-Odlyzko's and Coppersmith's methods. In particular, Coppersmith's algorithm uses a symmetric splitting system which motivates our method.

Chapter 3 focuses on the LHWP-DLP which is our main topic. Two existing methods for the LHWP-DLP will be presented. The change of equation for checking a solution and the notion of rotation-free elements in Section 3.2

plays an important role in our new algorithm.

The 2nd part (Chapters 4–6) is devoted to the description of our new algorithm for solving the LHWP-DLP. In Chapter 4, we construct parameterize splitting systems, which is a refinement and generalization of Coppersmith's symmetric splitting system. We present both deterministic and randomized versions.

In Chapter 5, we describe how parameterized splitting systems can be used to solve the DLP with LHWP exponents. We describe attacks for both cases when the order of the group is known and unknown.

In Chapter 6, we analyze the security of the GPS identification scheme and Hoffstein and Silverman's (2,2,11)-exponent. And the implementation on toy example is going to be presented. Finally, the conclusion and open problems are given in Chapter 7.

### Previous Publication

The 1st construction of parameterized splitting systems originally appeared in "A Parameterized Splitting System and its Application to the Discrete Logarithm Problem with Low Hamming Weight Product Exponents," joint work with Jung Hee Cheon, which was presented at PKC 2008 [KC08]. The basic idea to solve LHWP-DLP of Section 5 appeared in [KC08].

A refinement of parameterized splitting systems in Section 4.1 and materials in Chapter 6 originally appeared in "Parameterized Splitting Systems for the Discrete Logarithm," joint work with Jung Hee Cheon, which was published in IEEE Transactions on Information Theory in 2010 [KC10].

## 1.1 Notation

Throughout the thesis we use the following notation:

- $\mathbb{Z}_m$: residue classes $\mathbb{Z}/m\mathbb{Z}$. We represent $\mathbb{Z}_m$ as a set $\{0, 1, 2, \ldots, m-1\}$. Then $n = \lceil \log_2 m \rceil$ bits are required to represent an element of $\mathbb{Z}_m$ as a binary string,
- $x \bmod n$: the remainder of an integer $x$ when divided by an integer $n$,
- $wt(x)$: the Hamming weight of an integer $x$, which is defined as the number of nonzero coefficients in its binary representation,
- $\#A$: the cardinality of a finite set $A$,
- $\log(\cdot)$: the logarithm to base 2, *i.e.,* $\log_2(\cdot)$,
- $[a, b)_n$: given integers $a$, $b$, and $n$ with $0 \leq a, b < n$ and $a \neq b$, we define

$$[a, b)_n := \begin{cases} \{a, a+1, \ldots, b-1\}, & \text{if } a < b, \\ [a, n)_n \cup [0, b)_n, & \text{if } b < a. \end{cases}$$

  We call $a$ the starting element of the interval $[a, b)_n$.

# Chapter 2

# The Low Hamming Weight Discrete Logarithm Problem

Given $g$ and $h$ in a cyclic group $G$, the discrete logarithm problem (DLP) is to find the smallest non-negative inter $x$ such that $h = g^x$. In this chapter we introduce the *low Hamming weight* DLP (LHW-DLP) and the existing algorithms for the problem. First, we explain benefits from the use of LHW exponents in DL-based cryptosystems and security issues appearing in these systems. Then we introduce two algorithms *i.e.,* Heiman-Odlyzko algorithm and Coppersmith's algorithm for the LHW-DLP.

## 2.1 The Discrete Logarithm Problem (DLP)

The discrete logarithm problem (DLP) is one of the most fundamental mathematical problem in contemporary cryptography. The definition of the DLP is as follows:

**Definition 2.1.1.** Let $G$ be a multiplicative cyclic group of order $m$ generated by $g$. Given $h \in G$, the DLP is to find the unique integer $x$, $0 \leq x \leq m-1$, such that $g^x = h$, which is denoted by $\log_g h$.

Since firstly being introduced in 1976 by Diffie and Hellman [DH76], the DLP plays an important role in public key cryptography. Numerous public key cryptosystems, such as the Diffie-Hellman key agreement protocol [DH76], the ElGamal encryption and signature schemes [ElG85], and its variants [Sch91, DSS, LL98], rely on the hardness of the DLP.

Algorithms for solving number-theoretic problems can be categorized into two main classes: generic attacks, applicable in any group, and specific attacks designed for particular groups. The generic attacks on the DLP include the baby-step giant-step (BSGS) attack [Sha71], Pollard's rho and lambda algorithms [Pol78] as well as their parallelized versions [vOW99, Pol00] and the Pohlig-Hellman Algorithm [PH78]. For the survey, refer to [Tes01]. On the other hand, the index calculus method, where the basic idea goes back to Kraitchik [Kra22] surveyed in [SWD96, Odl00], is a specific attack suitable for the multiplicative group of a finite field. The index calculus algorithm is the best solution known for solving the DLP.

Shoop showed that if $p$ is the largest prime divisor of the group order on which the DLP is defined, a generic attack has to perform at least $O(\sqrt{p})$ group operations to solve the DLP [Sho97]. The BSGS, Pollard's rho, lambda, and Pohlig-Hellman methods fit into this bound which are called the

square-root attack, while the index calculus method solves the problem in sub-exponential time.

Before completing this section we describe the BSGS algorithm in brief since all the solutions to the low Hamming weight (product) DLP, on which the thesis focuses, are based on the method.

Let $n := \lceil \sqrt{m} \rceil$ for the group order $m$. Given $g$ and $h = g^x$, $x$ can be represented as $x = in + j$ for some $0 \leq I, j \leq m$. From this fact the BSGS finds $x$ as follows: first the algorithm computes $hg^{-1}, hg^{-2}, \ldots hg^n$ and build a lookup table that support an efficient search. Then the algorithm computes $g^{I \cdot n}$ for $0 \leq I \leq n-1$ and use the lookup table to find a collision. If a collision occurs in $x_1$ of $hg^{-x_1}$ and $x_2$ of $g^{x_2 \cdot n}$, then outputs $x = x_2 n + x_1$. Thus the algorithm solves the DLP with $O(\sqrt{m})$ group operations using a $O(\sqrt{m})$ storage.

## 2.2 The Low Hamming Weight DLP (LHW-DLP)

In practical cryptosystems that are based on the intractability of the DLP, logarithms of special structure are sometimes used. The idea is to choose a subset $X \in \mathbb{Z}_m$ of some special structure, which makes the system more efficient. Note that for an integer $x$ of weight $t$ and $g$ of an element of a group $G$, computing $g^x$ requires

$$(\log x \text{ squarings}) + (t - 1 \text{ multiplications})$$

by using the text book binary method. So if squarings can be done very efficiently, for example, squarings in GF($2^n$) and doublings on Koblitz elliptic curves.

On the other hand, the use of LHW exponents may weaken the security of the scheme. More precisely, the use of LHW exponents gave a question for the hardness of the following variant of the DLP, which is called the LHW-DLP:

**Definition 2.2.1** (The LHW-DLP)**.** Let $G$ be a multiplicative cyclic group of order $m$ generated by $g$. Let $n := \lceil \log m \rceil$ and $S_t$ be the subset of $\mathbb{Z}_m$ which consists of $n$-bit integers of the Hamming weight $t$, where $t$ is chosen to be much less than $n$ typically. Suppose $h = g^x$ for some $x \in S_t$. Then the LHW-DLP is: given $h = g^x$ for some $x \in S_t$ and integers $n$ and $t$, find an integer $x$.

Theoretically, the generic computational complexity of the DLP constrained to a subset of $S \in \mathbb{Z}_m$ is known to be lower-bounded by the square root of the cardinality of $S$ [MMN06, Sho97].

## 2.3 Algorithms for The LHW-DLP

In this section we cover two algorithms for the LHW-DLP, Heiman-Odlyzko's algorithm [Hei92] and Coppersmith's algorithm [CS84, MvOV97]. Both algorithms are based on the BSGS approach and uses combinatorial techniques.

### 2.3.1 Heiman-Odlyzko's Algorithm

Given an integer $x$ of weight $t$, and a non-negative integer $t_s < t$ we want to express $x$ as the sum of two integers $x_1$ and $x_2$, with weights $t_s$ and $t - t_s$, respectively. Such $x_1$ is easily obtained by choosing $t_s$ positions among the nonzero coefficients of the binary representation of $x$. Then we have

$$h = g^x = g^{x_1+x_2}, \quad hg^{-x_2} = g^{x_1}.$$

Heiman-Odlyzko's algorithm deterministically finds $x$ as follows: first we compute $g^{x_1}$ for each $x_1 \in \mathbb{Z}_m$ of weight $t_s$, build a lookup table that contains all the pairs $(g^{x_1}, x_1)$, and support an efficient search on the first component. Then we compute $hg^{-x_2}$ for each $x_2 \in \mathbb{Z}_m$ of weight $t - t_s$ and use the lookup table to find a collision. This procedure is presented as a pseudo-code in Algorithm 1.

Note that the exponentiations can be performed incrementally so that each requires only a constant number of group operations. Neglecting logarithmic factors required to sort the table, the time complexity of Heiman-Odlyzko's algorithm is $O\left(\binom{n}{t_s} + \binom{n}{t-t_s}\right)$ group operations in $G$. Since we need to store only either the left- or right-hand side, the space complexity of Heiman-Odlyzko's algorithm is $O\left(\min\{\binom{n}{t_s}, \binom{n}{t-t_s}\}\right)$.

**Algorithm 1** Heiman-Odlyzko's Algorithm for the LHW-DLP

**Input:** $g$, $h \in G$ of order $m$, $n$, $t$ and $t_s (< t)$

**Output:** $\log_g h$

Initialize an easily searched structure table $T$

**for all** $X_1 \subset \mathbb{Z}_n$ such that $\#X_1 = t_s$ **do**

Compute $x_1 := \sum_{i \in X_1} 2^i$ and $g^{x_1}$

Store $(g^{x_1}, x_1)$ in $T$ ordered according to the 1st coordinate

**end for**

**for all** $X_2 \subset \mathbb{Z}_n - X$ such that $\#X_2 = t - t_s$ **do**

Compute $x_2 := \sum_{i \in X_2} 2^i$ and $hg^{-x_2}$

**if** $hg^{-x_2} = g^{x_1}$ for some $(g^{x_1}, x_1)$ **then**

**return** $x_1 + x_2$

**end if**

**end for**

### 2.3.2 Coppersmith's Algorithm

Coppersmith developed a time/memory tradeoff algorithm for the LHW-DLP. The algorithm was originally invented by Coppersmith. We follows the description presented by Stinson [Sti02] and Galbraith [Gal12, Section 13.6]. The algorithm finds a solution of the LHW-DLP nearly in time of square-root of the size of the key space.

The idea of the algorithm is to reduce solving $h = g^x$ where $x$ is $n$-bit with Hamming weight $t$ to solving $hg^{-x_2} = g^{x_1}$ where $x_1$ and $x_2$ are both $n/2$-bit and weight $t/2$. Coppersmith's algorithm comes from the following observation called *Coppersmith's Splitting System*.

**Theorem 2.3.1** (Coppersmith's Splitting System)**.** *Suppose $n$ and $t$ are both even integers. Let $I = [0, n)_n$ and $\mathcal{B} = \{B_i : 0 \leq i \leq \frac{n}{2} - 1\}$, where $B_i = [i, i + \frac{n}{2})_n$ is an interval called a block. Then for every $T \subseteq I$ such that $|T| = t$, there exists a block $B \in \mathcal{B}$ such that $|T \cap B| = \frac{t}{2}$.*

*Proof.* Fix any $Y \subset I$ of size $t/2$. Define

$$\nu(i) := \#(Y \cap B_i) - \#(Y \cap (I - B_i)).$$

Then $\nu(i)$ is always even and

$$\nu(n/2) = -\nu(0), \quad v(i+1) - v(i) = \{-2, 0, 2\}.$$

If $\nu(0) = 0$, we are done. Otherwise, the values $\nu(i)$ change sign at least once as $I$ goes from 0 to $n/2 - 1$. Thus there exists some integer $0 \leq I \leq n/2 - 1$ such that $\nu(i) = 0$. □

This system can be extended to odd integers $n$ and $t$ [Sti02], where $\frac{n}{2}$ and $\frac{t}{2}$ are replaced by the nearest integers to $\frac{n}{2}$ and $\frac{t}{2}$, respectively.

Coppersmith's algorithm works as follows: given a binary representation $\sum_{i=0}^{n-1} x_i 2^i$ of $x \in \mathbb{Z}_m$, we define

$$\begin{aligned} u_k &:= \sum_{j=0}^{\frac{n}{2}-1} x_{k+j \bmod n} 2^{k+j \bmod n}, \\ v_k &:= x - u_k \end{aligned}$$

for $k = 0, \ldots, \frac{n}{2} - 1$. By Theorem 2.3.1, there exists $i$ such that

$$x = \sum_{j=0}^{n-1} x_j 2^j = u_i + v_i,$$

where $wt(u_i) = wt(v_i) = \frac{t}{2}$. Then we can compute $x$ using

$$hg^{-u_i} = g^{v_i}.$$

This algorithm has a time complexity of $O\left(n\binom{\frac{n}{2}}{\frac{t}{2}}\right)$ and a space complexity of $O\left(\binom{\frac{n}{2}}{\frac{t}{2}}\right)$.

The randomized version of the above algorithm was invented by Coppersmith and is described in [Sti02]. In this version, a block $B$ consists of randomly chosen $\frac{n}{2}$ elements in $[0, n)_n$. The time and space complexities of the randomized version are $O\left(\sqrt{t}\binom{\frac{n}{2}}{\frac{t}{2}}\right)$ and $O\left(\binom{\frac{n}{2}}{\frac{t}{2}}\right)$, respectively. We present the randomized version of Coppersmith's algorithm in Algorithm 2.

**Algorithm 2** Coppersmith's Algorithm for the LHW-DLP

**Input:** $g$, $h \in G$ of order $m$, even $n$, even $t$

**Output:** $\log_g h$, or $\perp$

1: Choose $B \subset \mathbb{Z}_n$ such that $\#B = n/2$
2: Initialize an easily searched structure table $T$
3: **for all** $X_1 \subset B$ such that $\#X_1 = t/2$ **do**
4: Compute $x_1 := \sum_{i \in X_1} 2^i$ and $g^{x_1}$
5: Store $(g^{x_1}, x_1)$ in $T$ ordered according to the 1st coordinate
6: **end for**
7: **for all** $X_2 \subset \mathbb{Z}_n - B$ such that $\#X_2 = t/2$ **do**
8: Compute $x_2 := \sum_{i \in X_2} 2^i$ and $hg^{-x_2}$
9: **if** $hg^{-x_2} = g^{x_1}$ for some $(g^{x_1}, x_1)$ **then**
10: **return** $x_1 + x_2$
11: **end if**
12: **end for**
13: **return** $\perp$

# Chapter 3

# The Low Hamming Weight Product DLP

In this chapter, we introduce *the low Hamming weight product DLP* (LHWP-DLP) proposed by Hoffstein and Silverman and advantages from the use of LHWP exponents. Then we review existing algorithms for solving the LHWP-DLP: the method from Coppersmith's splitting system and the method from the notion of rotation-free elements. In particular, the approach of the last method plays an important role for constructing a new algorithm later.

## 3.1 The Low Hamming Weight Product DLP (LHWP-DLP)

As discussed in Chapter 2, the use of LHW exponents brings better computational efficiency to DL-based cryptosystems. However more efficient attack on the LHW-DLP have been proposed by Coppersmith, hence, the LHW-DLP appeared to have less complexity than suggested [Sti02, CLP05], and so does not give any significant advantage over the ordinary exponents.

To enhance the security and achieve a greater speed-up, Hoffstein and Silverman suggested the use of low Hamming weight product (LHWP) exponents [HS03]. They suggested to use an exponent $x$ which is a product $x_1x_2\cdots x_r$ of very low Hamming weight exponents and take advantage of the fact that the sample space of the product $x$ is more-or-less the product of the sample spaces for $x_1, x_2, \ldots, x_r$. This was then applied to the GPS identification scheme, recommended by the NESSIE project [GPS02], in which a secret key is taken as a product of two integers having low Hamming weights [GL04, CLP05].

### 3.1.1 The Efficiency of LHWP Exponents

Let $G$ be a group $g$ be an element of $G$. Suppose we want to compute $g^x$ where $x = x_1x_2$. Then the computation of $g^x$ can be done by

$$g^x = g^{x_1x_2} = (g^{x_1})^{x_2}.$$

Then the cost of computing is approximately

$$(\log x \text{ squarings}) + (wt(x_1) + wt(x_2) \text{ multiplications}),$$

when the text book binary method is used.

If squaring and multiplication take approximately the same amount of time, then the approach above will not be good. However if squaring is very fast, the method significantly accelerates the computation of $g^x$. Hoffstein and Silverman captured the idea and developed in three situations of cryptographic interest, namely exponentiation GF($2^n$), multiplication on Koblitz curves, and multiplication in NTRU convolution rings. We briefly take a look at the 1st two cases, which are our concern.

**Squarings over** GF($2^n$)

For an exponent $x$ of Hamming weight $t$ over a group GF($2^n$), only $t-1$ multiplications are required for exponentiation if a group element is represented with respect to a normal basis [AMOV91]. Note that in this case a squaring is just a shift operation.

**Doublings on Koblitz Curves**

Koblitz curve is an elliptic curve over GF($2^n$) defined by

$$E: \quad y^2 + xy = x^3 + ax^2 + 1, \quad a \in \mathrm{GF}(2).$$

Let $\tau$ be a Frobenius map on $E$:

$$\tau: \quad E(\mathrm{GF}(2^n) \to E(\mathrm{GF}(2^n)); \quad (x, y) \mapsto (x^2, y^2).$$

Then Frobenius map is efficiently computable on $E(\mathrm{GF}(2^n))$ and plays a similar role to squaring in binary fields. That is, suppose we want to compute $NP$ for an integer $N$ and a point $P$ on the curve. Then it is possible to write $N$ as a linear combination

$$N = N_0 + \tau N_1 + \cdots + \tau^n N_n,$$

with $N_I \in \{0, \pm 1\}$.

### 3.1.2 The Definition of LHWP-DLP

Using LHWP exponents we are faced with the following problem, so-called the *low Hamming weight product DLP* (LHWP-DLP).

**Definition 3.1.1** (The LHWP-DLP)**.** Let $G$ be a multiplicative cyclic group of order $m$ generated by $g$. For $i = 1, \ldots, r$, let $S_{t_i}$ be subsets of $\mathbb{Z}_m$ which consists of $n_i$-bit integers of the Hamming weight $t_i$, where $t_i$ is chosen to be much less than $n_i$ typically. Then the LHW-DLP is: given $h = g^x$, where $x = \prod_{i=1}^{r} x_i$ for some $x_i \in S_{t_i}$, find an integer $x$.

In practical applications such as Hoffstein and Silverman's suggestion and the GPS identification scheme, $r$ is chosen to be 2 or 3. In the rest of this chapter we review existing methods to solve this problem focusing on the case $r = 2$ or 3.

## 3.2 Existing Algorithms for LHWP-DLP

In a general manner, the LHWP-DLP is a form of $h = g^{xy}$, where $x$ is an integer of bit-length $n$ and Hamming weight $t$, and $y$ is an element of a set $Y$. More precisely, when $x = x_1x_2$ such that $x_1$ is of bit-length $n_1$ with weight $t_1$ and $x_2$ is of bit-length $n_2$ with weight $t_2$, the above $Y$ can be regarded as a set of $n$-bit number of weight $t_2$. When $x = x_1x_2x_3$ such that $x_I$ is of bit-length $n_I$ with weights $t_I$, the above $Y$ can be regarded as a product of a set of $n_2$-bit number of weight $t_2$ and a set of $n_3$-bit number of weight $t_3$.

### 3.2.1 Attack using Coppersmith's Splitting System

Coron, Lefranc, and Poupard presented a method for solving the LHWP-DLP using Coppersmith's algorithm to analyze the security of the GPS identification scheme [CLP05]. They gave methods in cases that the order of a group is both known and unknown when $r = 2$.

**The Known Order Case:** let $G$ be a group of prime order $m$ and $g$ be an element of $G$. Let $X$ and $Y$ subsets of $\mathbb{Z}_m$ and let $h := g^{xy}$ for some $x \in X$ and $y \in Y$. Then we have

$$h^{y^{-1}} = g^x,$$

where $y^{-1}$ is the inverse of $y$ modulo $m$.

Now we can find $xy$ as follows: first we compute $g^x$ for each $x \in X$, build a lookup table that contains all the pairs $(g^x, x)$, and support an efficient search on the first component. Then we compute $h^{y^{-1}}$ for each $y2 \in Y$ and use the lookup table to find a collision. Neglecting logarithmic factors required to sort the table, the time complexity of the method is $O(\#X + \#Y)$ group operations in $G$. Since we need to store only either the left- or right-hand

side, the space complexity of the method is $O(\min\{\#X, \#Y\})$.

**The Unknown Order Case:** we assume that the order of $G$ is prime. Recall the equation

$$h^{y^{-1}} = g^x. \tag{3.2.1}$$

If the order of $g$ is unknown, $y^{-1}$ can not be computed from $y$ and so we cannot use the above equation directly. However, authors in [CLP05] overcame this obstacle by the following trick, originally proposed by Shoup [Sho00]: let

$$\Upsilon := \prod_{y \in Y} y, \quad \hat{g} := g^{\Upsilon}.$$

Since the order of $g$ is prime, for any nonzero $x$ the order of $g^x$ is equal to that of $g$, which implies $\hat{g}$ is also a generator of $G$. By raising both sides of (3.2.1) to the power $\Upsilon$, we have

$$h^{\prod_{y' \in Y - \{y\}} y'} = \hat{g}^x. \tag{3.2.2}$$

With this new equation we can make use of the BSGS technique. That is, we compute the two following sets:

$$S_1 := \{h^{\prod_{y' \in Y - \{y\}} y'} : y \in Y\}, \quad S_2 := \{\hat{g}^x : x \in X\}.$$

Then in the two sets, two values meet for one value $h^{\prod_{y' \in Y - \{y_0\}} y'}$ and one value $\hat{g}^{x_0}$ such that $xy = x_0 y_0$.

The main bottleneck of this approach is to compute $S_1$. This can be done efficiently by using binary product tree method [Gal12, Section 2.15.1] (The details of a method will be presented in Section 5.2.). With this one can compute $S_1$ in $\#Y \log(\#Y)$ group exponentiations or $\#Y \log(\#Y) \log m$ group operations. Thus neglecting logarithmic factors required to sort the table, the time and space complexity of the method are $O(\#X + \#Y \log(\#Y))$ and $O(\min\{\#X, \#Y \log(\#Y)\})$.

### 3.2.2 Rotation-Free Elements

As discussed in Section 3.1, Hoffstein and Silverman considered the LHWP-DLP over a multiplicative group of GF($2^n$) with a generator $g$, *i.e.,* finding $x_1x_2x_3$, given $h = g^{x_1x_2x_3}$ where each $x_I$ is an integer of bit-length $n_I$ with weight $t_I$.

In [CK08], Cheon and Kim proposed an attack to Hoffstein and Silverman's exponents using the notion of *Rotation-Free elements*. The idea behind Cheon and Kim's attack is to reduce the key search space by considering only one element from each equivalent class. An equivalent relation $\sim$ on $\mathbb{Z}_{2^n-1}$ is defined an equivalent relation $\sim$ on $\mathbb{Z}_{2^n-1}$ as follows:

$a \sim b$ if and only if there exists a non-negative integer $I$ such that $a = 2^i b$.

However since there is no known algorithm to generate such representatives efficiently, they suggested the use of a set of rotation-free elements that contains at least one representative for each equivalent class. The set is only slightly larger than the number of equivalent classes and is easily generated, which defined as follows [CK08, Algorithm 1]:

**Definition 3.2.1.** [CK08, Definition 1] An element $z \in \mathbb{Z}_{2^n-1}$ is called a rotation-free element if there is a $t$-tuple $(a_1, a_2, \ldots, a_t)$ for a positive integer $t$ satisfying

1. $a_i \geq a_1$ for $1 \leq i \leq t$,
2. $\sum_{i=1}^{t} a_i = n$,
3. $z = 2^{n-1} + 2^{n-1-a_1} + \cdots + 2^{n-1-(a_1+a_2+\cdots+a_{t-1})}$.

Note that the corresponding $t$-tuple for a rotation-free element satisfies $ta_1 \leq a_1 + \cdots + a_t = n$. All the rotation-free elements of weight $t$ can be easily generated by the following procedures:

1. Input $n$ and $t$
2. Choose a positive integer $a_1 \leq n/t$
3. For $i = 2$ up to $t-1$, select an integer $a_i$ such that
$$a_1 \leq a_i \leq n - (t-i)a_1 - \sum_{j=1}^{i-1} a_j$$
4. Output $2^{n-1} + 2^{n-1-a_1} + \cdots + 2^{n-1-(a_1+a_2+\cdots+a_{t-1})}$

Note that the largest element of each equivalence class is a rotation-free element. Hence one can see that there is at least one rotation-free element in each equivalence class of $Z/(2^n-1)$ with respect to the relation $\sim$. Authors of [CK08] showed that this number is not far from the number of equivalence classes. We omit the proof.

**Lemma 3.2.1.** *[CK08, Theorem 1] Let $n, t$ be positive integers with $t < n$ and $\mathrm{RF}(n,t)$ be the number of rotation-free elements of weight $t$ in $\mathbb{Z}/(2^n-1)$. We have the followings:*

1. $\mathrm{RF}(n,t) = \sum_{i=0}^{\lfloor \frac{n}{t} \rfloor - 1} \binom{n-2-ti}{t-2}$.
2. *There is at least one rotation-free element in each equivalence class.*
3. *The difference $\mathcal{E}(n,t)$ between $\mathrm{RF}(n,t)$ and the number of equivalence classes on $\mathbb{Z}/(2^n-1)$ is at most*
$$\binom{n-2}{t-2} - \sum_{i=1}^{\lfloor \frac{n}{t} \rfloor - 1} \binom{n-2-ti}{t-1} + 1.$$

Now we are in a position to present the attack on Hoffstein and Silverman's exponent using rotation-free elements. The authors of [CK08] considered the case that $n = 1000$, $t_1 = t_2 = 2$, and $t_3 = 11$. First we convert the

equation

$$y = g^{x_1 x_2 x_3}$$

to

$$y^{2^k \bar{x}_1^{-1} \bar{x}_2^{-1}} = g^{x_3},$$

where $0 \leq k < n = 1000$ and each of $\bar{x}_1$ and $\bar{x}_2$ is a rotation-free element in $\mathbb{Z}_{2^n-1}$. Furthermore we rewrite $x_3$ as $x_3 = x_3' + \bar{x}_3'$ where $x_3'$ and $\bar{x}_3'$ are of weights 3 and 8 in $\mathbb{Z}/(2^n - 1)$ and $\bar{x}_3'$ is rotation-free. Then check the following equation:

$$y^{2^{-k}(\bar{x}_1 \bar{x}_2)^{-1}} g^{-x_3'} = g^{\bar{x}_3'}.$$

Then the complexity is

$$n \cdot \mathrm{RF}(n,2)^2 \cdot \binom{n-1}{3} + \mathrm{RF}(n,8) \approx 2^{55.2} + 2^{54.5} \approx 2^{55.2}, \quad n = 1000.$$

# Chapter 4

# Parameterized Splitting Systems

In this chapter, we propose *parameterized splitting systems*. A parameterized splitting system is a generalization of Coppersmith's splitting system for further applications. Given $T \subset I$, Coppersmith's splitting system gives $B \in \mathcal{B}$ such that $\#(T \cap B) = t/2$. Our parameterized splitting system, however, is flexible since it provides $T$ with $\#(T \cap B) = t_s$ and $\#B = \lfloor \frac{t_s n}{t} \rfloor$ for any $1 \leq t_s \leq t$. Furthermore, it has an additional property *i.e.,* it allows us to find a block $B$ whose starting element belonging to $T$.

## 4.1 Parameterized Splitting Systems

In this section, we propose a more flexible splitting system, called a parameterized splitting system. It can be regarded as a generalization of Coppersmith's splitting system: given a bit string of length $n$ and weight $t$ and any positive integer $t_1 < t$, there exists a part of the string of length $n_1$ and weight $t_1$ where $\frac{n_1}{t_1} \approx \frac{n}{t}$.

We start with the definition of parameterized splitting systems.

**Definition 4.1.1** (Parameterized Splitting Systems)**.** Let $n$ and $t$ be integers such that $0 < t < n$ and $I := [0, n)_n$. For any $t_s$ with $1 \leq t_s \leq t$, a subset $\mathcal{B}_n$ of $\{B \subset I : \#B = \lfloor \frac{t_s n}{t} \rfloor\}$ with cardinality $N$ is called an $(N; n, t, t_s)$-parameterized splitting system of $I$ if there exists a block $B \in \mathcal{B}$ such that $\#(T \cap B) = t_s$ for every $T \subseteq I$ with $\#T = t$.

### 4.1.1 The Construction

For any $n$, $t$, and $t_s$ such that $0 < t < n$ and $1 \leq t_s \leq t$, we construct $(n; n, t, t_s)$-parameterized splitting systems in the following theorem. Interestingly, though the motivation of parameterized splitting systems comes from Coppersmith's splitting system, our parameterized splitting systems have one nice additional property different from Coppersmith's.

**Theorem 4.1.1.** *Let $1 \leq t_s \leq t < n$ be integers and $n_s = \lfloor \frac{t_s n}{t} \rfloor$. Then*

$$\mathcal{B}_n = \{B_i = [i, i + n_s)_n : 0 \leq i \leq n - 1\}$$

*is an $(n; n, t, t_s)$-parameterized splitting system of $I = [0, n)_n$ with additional property: for any $T \subset I$ of cardinality $t$, there exists a block $B_i \in \mathcal{B}_n$ such that $i \in T$ and $\#(B_i \cap T) = t_s$.*

*Proof.* Let $T := \{y_0, y_1, \ldots, y_{t-1}\}$. For $0 \le i \le t-1$, we define

$$I_i := [y_{i \bmod t}, y_{i+1 \bmod t})_n$$

and

$$A_i := I_{i \bmod t} \cup \cdots \cup I_{i+t_s-1 \bmod t}.$$

Then $\#(T \cap A_i) = t_s$ for all $i$. Since $I_i = \bigcap_{j=0}^{t_s-1} A_{i-j \bmod t}$,

$$\#A_0 + \#A_1 + \cdots + \#A_{t-1} = t_s \sum_{i=0}^{t-1} \#I_i = t_s \#I = t_s n.$$

If $\#A_i = n_s$ for some $i$, then this block $A_i = [y_{i \bmod t}, y_{i+t_s \bmod t})_n$ is the desired one. Now suppose that $\#A_i \neq n_s$ for all $i$. If $\#A_i < n_s$ for all $i$, then

$$t_s n = \sum_{i=0}^{t-1} \#A_i < t n_s = t \left\lfloor \frac{t_s n}{t} \right\rfloor \le t_s n,$$

which is a contradiction.

If $\#A_i > n_s$ for all $i$, then

$$t_s n = \sum_{i=0}^{t-1} \#A_i \ge t(n_s + 1) > t_s n,$$

which is a contradiction. Thus there exists $i$ such that

$$\#A_i < n_s \text{ and } \#A_{i+1 \bmod t} > n_s,$$

which implies

$$\#[y_{i+1 \bmod t}, y_{i+t_s \bmod t})_n = \#(A_i \cap A_{i+1 \bmod t}) < n_s$$

and

$$\#\{(A_i \cap A_{i+1 \bmod t}) \cup [y_{i+t_s \bmod t}, y_{i+t_s+1 \bmod t})_n\} = \#A_{i+1 \bmod t} > n_s.$$

Therefore there exists

$$\ell \in [y_{i+t_s \bmod t}, y_{i+t_s+1 \bmod t})_n$$

such that

$$\#[y_{i+1 \bmod t}, \ell)_n = n_s.$$

This block $[y_{i+1 \bmod t}, \ell)_n$ is what we want to find because

$$\begin{aligned} T \cap [y_{i+1 \bmod t}, \ell)_n &= T \cap A_{i+1 \bmod t} \\ &= \{y_{i+1 \bmod t}, \ldots, y_{i+t_s \bmod t}\} \end{aligned}$$

whose cardinality is equal to $t_s$. □

The above $(n; n, t, t_s)$-parameterized splitting system guarantees that for any given target string $x$ of length $n$ and weight $t$, by trying at most $n$ blocks of $n_s$ consecutive elements, we can split $x$ into the sum of two strings, one of which is of length $n_s$ and weight $t_s$, and starts from one of the fixed $t$ positions. More precisely, by exploiting this property, given an $n$-bit integer $x$ one can find an $n_1$-bit integer $u$ and an $(n - n_1)$-bit integer $v$ of weight $t_1$ and $t - t_1$, respectively, such that $u + v2^{n_1} \equiv x2^k \pmod{2^n - 1}$ for some integer $k$.

## 4.2 A Randomized Version

We may consider a faster algorithm by using probabilistic approaches. Given $n$, $t$, $t_s$, and $n_s = \lfloor \frac{t_s n}{t} \rfloor$ such that $1 \leq t_s \leq t < n$, we randomly choose $B \subset I$ such that $\#B = n_s$ and check whether $\#T \cap B = t_s$. Theorem 4.2.1 determines the expected running time in this case.

**Lemma 4.2.1.** *Let $I := [0, n)_n$. Given $t$, $t_s$, and $n_s = \lfloor \frac{t_s n}{t} \rfloor$ such that $1 \leq t_s \leq t < n$, fix a set $T \subset I$ such that $\#T = t$. The probability that a randomly chosen $B \subset I$ such that $\#B = n_s$ and $\#(T \cap B) = t_s$ is*

$$p = \frac{\binom{t}{t_s}\binom{n-t}{n_s-t_s}}{\binom{n}{n_s}} = \frac{\binom{n_s}{t_s}\binom{n-n_s}{t-t_s}}{\binom{n}{t}}.$$

*Proof.* The total number of blocks $B$ such that $|B| = n_s$ and $|T \cap B| = t_s$ is $\binom{t}{t_s}\binom{n-t}{n_s-t_s}$. Hence given $t_s$, the probability of success is

$$\begin{aligned}
p &= \frac{\binom{t}{t_s}\binom{n-t}{n_s-t_s}}{\binom{n}{n_s}} \\
&= \frac{\frac{t!}{t_s!(t-t_s)!} \cdot \frac{(n-t)!}{(n_s-t_s)!\{(n-t-(n_s-t_s)!)\}}}{\frac{n!}{n_s!(n-n_s)!}} \\
&= \frac{t! \cdot (n-t)! \cdot \frac{1}{t_s!(n_s-t_s)!} \cdot \frac{1}{(t-t_s)!\{n-n_s-(t-t_s)\}!}}{n! \cdot \frac{1}{n_s!(n-n_s)!}} \\
&= \frac{\frac{n_s!}{t_s!(n_s-t_s)!} \cdot \frac{(n-n_s)!}{(t-t_s)!\{n-n_s-(t-t_s)\}!}}{\frac{n!}{t!(n-t)!}} \\
&= \frac{\binom{n_s}{t_s}\binom{n-n_s}{t-t_s}}{\binom{n}{t}}.
\end{aligned}$$

□

In order to calculate the lower bound of $p$, we need Lemma 4.2.2. For the proof, refer to [MS77].

**Lemma 4.2.2.** *[MS77, p. 309, Lemma 7] Suppose that $n$ and $\lambda n$ are positive integers, where $0 < \lambda < 1$. Define*

$$H(\lambda) := -\lambda \log \lambda - (1-\lambda)\log(1-\lambda).$$

*Then*

$$\frac{2^{nH(\lambda)}}{\sqrt{8n\lambda(1-\lambda)}} \leq \binom{n}{\lambda n} \leq \frac{2^{nH(\lambda)}}{\sqrt{2\pi n\lambda(1-\lambda)}}.$$

The lower bound of $p$ can be easily obtained using Lemma 4.2.2 if $t$ divides $t_s n$, which includes the case that $n$ is even and $t_s = t/2$ [Sti02]. But if $t \nmid t_s n$, a more elaborate proof is required.

**Theorem 4.2.1.** *Suppose $2 \leq t \leq n/2$ and $1 \leq t_s \leq t/2$. Then*

$$p > \frac{1}{4}\sqrt{\frac{n\pi}{et(n-t)}} > \frac{\sqrt{\pi}}{4\sqrt{et}}.$$

*Furthermore, if $t \mid t_s n$, then*

$$p > \frac{1}{4}\sqrt{\frac{n\pi}{t(n-t)}} > \frac{\sqrt{\pi}}{4\sqrt{t}}.$$

*Proof.* Let

$$\lambda_1 := \frac{t_s}{t}, \quad \lambda_2 := \frac{n_s - t_s}{n-t}, \quad \lambda := \frac{n_s}{n}.$$

Then we can write

$$p = \frac{\binom{t}{\lambda_1 t}\binom{n-t}{\lambda_2(n-t)}}{\binom{n}{\lambda n}}.$$

From Lemma 4.2.2,

$$p \geq \frac{2^{tH(\lambda_1)+(n-t)H(\lambda_2)-nH(\lambda)} \cdot \sqrt{\lambda(1-\lambda)} \cdot \sqrt{2\pi n}}{\sqrt{\lambda_1(1-\lambda_1)\lambda_2(1-\lambda_2)} \cdot 8\sqrt{t(n-t)}}.$$

If $t \mid t_s n$, then $n_s = t_s n/t$ and so $\lambda_1 = \lambda_2 = \lambda$. Hence,

$$tH(\lambda_1) + (n-t)H(\lambda_2) - nH(\lambda) = 0.$$

If $t \nmid t_s n$, then $\lambda_2 < \lambda < \lambda_1 \leq \frac{1}{2}$. Since $H$ is a continuous function, by the mean value theorem there exist $\lambda < c_1 < \lambda_1$ and $\lambda_2 < c_2 < \lambda$ such that

$$H(\lambda_1) - H(\lambda) = H'(c_1)(\lambda_1 - \lambda)$$

and

$$H(\lambda_2) - H(\lambda) = H'(c_2)(\lambda_2 - \lambda).$$

Hence we have

$$\begin{aligned} & tH(\lambda_1) + (n-t)H(\lambda_2) - nH(\lambda) \\ = \; & t(H(\lambda_1) - H(\lambda)) + (n-t)(H(\lambda_2) - H(\lambda)) \\ = \; & tH'(c_1)(\lambda_1 - \lambda) + (n-t)H'(c_2)(\lambda_2 - \lambda) \\ = \; & \frac{t_s n - t n_s}{n}(H'(c_1) - H'(c_2)). \end{aligned}$$

Again by the mean value theorem, there exists $c_2 < c < c_1$ such that

$$H'(c_1) - H'(c_2) = H''(c)(c_1 - c_2)$$

since $H'$ is also continuous. From the inequality

$$t_s n - t n_s \leq t - 1,$$

we have

$$\begin{aligned} & \frac{t_s n - t n_s}{n}(H'(c_1) - H'(c_2)) \\ = \; & \frac{t_s n - t n_s}{n} H''(c)(c_1 - c_2) \\ \geq \; & \frac{t-1}{n} \cdot \frac{-\log e}{\lambda_2(1-\lambda_2)} \cdot \frac{t-1}{(n-t)t} \\ > \; & -\log \sqrt{e}, \end{aligned}$$

where the first inequality holds since $H''(x) = \frac{-\log e}{x(1-x)}$ is increasing for

$$0 < x < 1/2 \text{ and } c_1 - c_2 < \lambda_1 - \lambda_2,$$

and the second is obtained by using

$$1/(n-t) \leq \lambda_2 \text{ and } (t-1)/n < 1/2.$$

Since

$$\lambda_1(1-\lambda_2) \leq \lambda_1 = \frac{t_s}{t} \leq \frac{1}{2},$$

we have

$$\begin{aligned} \frac{\sqrt{\lambda(1-\lambda)}}{\sqrt{\lambda_1(1-\lambda_1)\lambda_2(1-\lambda_2)}} &\geq \frac{\sqrt{\lambda_2(1-\lambda_1)}}{\sqrt{\lambda_1(1-\lambda_1)\lambda_2(1-\lambda_2)}} \\ &= \frac{1}{\sqrt{\lambda_1(1-\lambda_2)}} > \sqrt{2}. \end{aligned}$$

Therefore,

$$p > 2^{-\log_2 \sqrt{e}} \cdot \sqrt{2} \cdot \frac{\sqrt{2\pi n}}{8\sqrt{t(n-t)}} > \frac{\sqrt{\pi}}{4\sqrt{et}}.$$

□

Theorem 4.2.1 implies that the expected value of trials to find an appropriate block $B$ such that $\#T \cap B = t_s$ is $O(\sqrt{t})$, regardless of $n$ and $t_s$. Note that a randomized version loses an additional property, that is, we know exactly one element of $B \cap T$ in a deterministic version of parameterized splitting systems of Theorem 4.1.1.

# Chapter 5

# A New Algorithm from Parameterized Splitting Systems

In this chapter we describe how parameterized splitting systems can be used to solve the DLP with LHWP exponents. We describe attacks for both cases when the order of the group is known and unknown. In virtue of the flexibility of parameterized splitting systems, a new algorithm shows more efficient performance that those of existing methods.

## 5.1 When The Order of a Group is Known

Let $G$ be a cyclic group of order $m$ generated by $g$. Given $h \in G$, recall that the LHWP-DLP is to fine $\log_g h$ when $h = g^z$, where $z := \prod_{i=1}^{r} x_i$ for $x_i$ of (known) bit-length $n_i$ and (known) Hamming weight $t_i$ (see Definition 3.1.1).

In this section we consider the LHWP-DLP when $z$ is a product of two elements $x \in X$ and $y \in Y$ for two subsets $X$ and $Y$ of $\mathbb{Z}_m$.

### 5.1.1 Motivations

If we apply the BSGS technique for the equation $h^{y^{-1}} = g^x$, $x$ and $y$ can be computed in $O(\#X + \#Y)$. This might not be the best approach when $\#X$ is greater than $\#Y$. In this unbalanced case, it might be better to split $x$ as $u + v$ for $u \in U$ and $v \in V$ where $U$ and $V$ are subsets of $\mathbb{Z}_m$ satisfying

$$X \subset U + V := \{u + v : u \in U, v \in V\}.$$

Then we check the following equality for each $y \in Y$ as in [CLP05]:

$$h\,(g^y)^{-u} = (g^y)^v\,.$$

Then the complexity becomes $O\,(\#Y \cdot (\#U + \#V))$. When $X$ is a set of LHW elements, the usable splitting systems include those of Heiman-Odlyzko [Hei92] and Coppersmith: the latter has a lower complexity.

In order to lower the complexity, we may consider the following the equation, as suggested in [HS03],

$$h^{y^{-1}} g^{-u} \;=\; g^v. \tag{5.1.1}$$

The BSGS attack using the above equation has the complexity $O(\#Y \cdot \#U + \#V)$, which is smaller than the previous when $\#U < \#V$.

For the above $X$ consisting of LHW elements, it is obtained by Heiman-Odlyzko's algorithm, but not by Coppersmith's algorithm, which supports only symmetric splitting with $\#U \approx \#V$.

### 5.1.2 Using The Parameterized Splitting Systems

Let us consider the subset $X$ of $\mathbb{Z}_m$

$$X := \left\{ x = \sum_{j=0}^{n-1} x_j 2^j : x_j = 0 \text{ or } 1, wt(x) = t \right\}.$$

We explain how to apply our parameterized splitting systems of Theorem 4.1.1 in more detail. Define

$$T = \{j : x_j = 1\} \subset I = [0, n)_n.$$

Given $t_s \in \left[0, \lceil \frac{t}{2} \rceil\right]$, there exists an $(n; n, t, t_s)$-parameterized splitting system $(I, \mathcal{B})$ by Theorem 4.1.1. Hence, for $n_s := \left\lfloor \frac{t_s n}{t} \right\rfloor$, there is a block

$$B_i := [i, i + n_s \bmod n)_n \in \mathcal{B}$$

such that $\#(T \cap B_i) = t_s$. For this $i$, we set

$$u = \sum_{j=0}^{n_s-1} x_{i+j \bmod n} 2^{i+j \bmod n}.$$

Then we have $wt(u) = t_s$ and $wt(v) = t - t_s$ for $v := x - u$. Furthermore we can force the first nonzero bit of $u$ to be $x_i$ thanks to an additional property.

The algorithm works as follows: for each $i$ with $0 \le i \le n-1$, we define

$$U_i := \left\{ u = \sum_{j=i}^{n_s+i-1} u_{j \bmod n} 2^{j \bmod n} : u_i = 1, wt(u) = t_s \right\}$$

and

$$V_i := \left\{ v = \sum_{j=0}^{n-1} v_j 2^j : v_i = v_{i+1 \bmod n} = \cdots = v_{n_s+i-1 \bmod n} = 0, wt(v) = t - t_s \right\}.$$

Then we compute the left-hand side of Eq. (5.1.1) for all $u \in U_i$ and $y \in Y$, and store them after sorting by the value $h^{y^{-1}}g^{-u}$. Second, we compute the right-hand side of Eq. (5.1.1) for each $v \in V_i$ and check if it is in the list from the first part.

We have to compute $\#Y \cdot \binom{n_s-1}{t_s-1}$ exponentiations in the first step, $\binom{n-n_s}{t-t_s}$ exponentiations in the second step, and repeat these two steps $n$ times. Hence the time complexity is

$$O\left(n\left(\#Y \cdot \binom{n_s-1}{t_s-1} + \binom{n-n_s}{t-t_s}\right)\right).$$

Since we can store the smaller set among the sets from the first and the second step, the space complexity is

$$O\left(\min\left\{\#Y \cdot \binom{n_s-1}{t_s-1}, \binom{n-n_s}{t-t_s}\right\}\right).$$

The randomized version of this algorithm uses randomly chosen blocks that do not need to be sets of consecutive numbers. Theorem 4.2.1 guarantees that we can find an appropriate block in at most $\frac{4\sqrt{et}}{\sqrt{\pi}}$ trials. Thus, the running time of the randomized version is

$$O\left(\sqrt{t}\left(\#Y \cdot \binom{n_s}{t_s} + \binom{n-n_s}{t-t_s}\right)\right).$$

The space requirement is the same as that of the deterministic version.

We present pseudo-codes of deterministic and randomized versions in Algorithm 3 and Algorithm 4, respectively.

**Algorithm 3** Solving the LHWP-DLP of known order case with parameterize splitting systems (deterministic)

**Input:** $g$, $h \in G$ of order $m$, two subsets $X$ and $Y$ of $\mathbb{Z}_m$ with descriptions $n$ and $t$ such that $0 < t < n$

**Output:** $\log_g h$

1: Choose appropriate $1 \leq t_s \leq t$ and set $n_s := \left\lfloor \frac{t_s n}{t} \right\rfloor$
2: **for** $i = 0$ to $n - 1$ **do**
3: Initialize an easily searched structure table $T$
4: Set $B_i := [i, i + n_s)_n \subset I\,(:= [0, n)_n)$
5: **for all** $y \in Y$ **do**
6: **for all** $U \subset B_i - \{i\}$ such that $\#U = t_s - 1$ **do**
7: Compute $u := 2^i + \sum_{j \in U} 2^j$
8: Store $\left(h^{y^{-1}} g^{-u}, y, u\right)$ in $T$ ordered according to the 1st coordinate
9: **end for**
10: **end for**
11: **for all** $V \subset I - B_i$ such that $\#V = t - t_s$ **do**
12: Compute $v := \sum_{j \in V} 2^j$
13: **if** $h^{y^{-1}} g^{-u} = g^v$ for some $(g^v, v)$ **then**
14: **return** $y(u + v)$
15: **end if**
16: **end for**
17: **end for**

**Algorithm 4** Solving the LHWP-DLP of known order case with parameterize splitting systems (randomized)

**Input:** $g$, $h \in G$ of order $m$, two subsets $X$ and $Y$ of $\mathbb{Z}_m$ with descriptions $n$ and $t$ such that $0 < t < n$

**Output:** $\log_g h$, or $\perp$

1: Choose appropriate $1 \leq t_s \leq t$ and set $n_s := \left\lfloor \frac{t_s n}{t} \right\rfloor$
2: Choose $B \subset I\,(:= [0, n)_n)$ such that $\#B = n_s$
3: Initialize an easily searched structure table $T$
4: **for all** $U \subset B$ such that $\#U = t_s$ **do**
5: Compute $u := \sum_{j \in U} 2^j$
6: Store $\left(h^{y^{-1}} g^{-u}, y, u\right)$ in $T$ ordered according to the 1st coordinate
7: **end for**
8: **for all** $V \subset I - B$ such that $\#V = t - t_s$ **do**
9: Compute $v := \sum_{j \in V} 2^j$
10: **if** $h^{y^{-1}} g^{-u} = g^v$ for some $(g^v, v)$ **then**
11: **return** $y(u + v)$
12: **end if**
13: **end for**
14: **return** $\perp$

## 5.2 When the Order of a Group is Unknown

### 5.2.1 The Basic Approach

We consider the DLP of LHWP exponents when the order of $G$ is unknown. We assume it is known that the order of $G$ is prime. Recall Eq. (5.1.1)

$$h^{y^{-1}} g^{-u} \quad = \quad g^{v}.$$

If the order of $g$ is unknown, $y^{-1}$ can not be computed from $y$ and so we cannot use Eq. (5.1.1) directly. However we can overcome this obstacle by the trick discussed in Section 3.2. We recall it compactly. By letting

$$\Upsilon := \prod_{y \in Y} y, \quad \hat{g} := g^{\Upsilon},$$

and raising both sides of Eq. (5.1.1) to the power $\Upsilon$, we have we have

$$h^{\prod_{y' \in Y - \{y\}} y'} \cdot \hat{g}^{-u} \quad = \quad \hat{g}^{v}. \tag{5.2.2}$$

Once we precompute and store $\hat{g}$, $\hat{g}^{-1}$, and $h^{\prod_{y' \in Y - \{y\}} y'}$, we can solve the DLP using parameterized splitting systems and a technique similar to that in the known-order case. Then the main bottleneck of this approach is to compute $h^{\prod_{y' \in Y - \{y\}} y'}$ for all $y' \in Y$. We come up with binary product tree method for this problem.

### 5.2.2 Precomputation

For a better overview of the construction, we consider in the following a set $Y$ of $2^n$ elements denoted by $x_i$ for $1 \leq i \leq 2^n$. What we want to compute is the set of values

$$S := \left\{ h^{\prod_{y' \in Y - \{y\}} y'} \right\}.$$

The method relies on an implicit binary tree structure. The algorithm starts from the root equal to $g$ and it ends with $2^n$ leaves equal to the elements of $S$. The tree consists of $n$ level, *i.e.,* the depth of the tree is $n$ (we ignore the root level) and each level $L_i$ consists of $2^i$ elements in $G$. We represent $L_i$ as $\{h_{i,1}, \ldots, \}$

We define some notations as follows:

- for $A \subset \mathbb{Z}_n$, $g^A := g^{\prod_{i \in A} x_i}$,
- $L_i$ is identified with $\{h_{i,1}, \ldots, h_{i,2^i}\}$,
- from each element $h_{i,j}$ of $L_i$ we compute two elements of $L_{i+1}$. Hence we can denote these two elements by $h_{i+1,2j-1}$ and $h_{i+1,2j}$
- let $h_{i,j} := g^A$ for some $A \subset \mathbb{Z}_n$. Then $\text{idx}(h_{i,j}) := A$

Now we describe the algorithm.

1. compute $h_{1,1} := g^{\{2^{n-1}+1 \ldots, 2^n\}}$ and $h_{1,2} := g^{\{1,2,\ldots,2^{n-1}\}}$.
2. for $1 \leq i \leq n-1$, compute $L_{i+1}$ from $L_i$ as follows:
   (a) for each $h_{i,j}$, set $A := \mathbb{Z}_n - \text{idx}(h_{i,j})$.
   (b) according to numerical order set $A_1$ and $A_2$ as the last and first half elements of A, respectively.
   (c) compute $h_{i+1,2j-1} := h_{i,j}^{A_1}$ and $h_{i+1,2j} := h_{i,j}^{A_2}$ and discard $h_{i,j}$.

**Example:** let $n = 8$. We first compute $h_{1,1} = g^{x_5x_6x_7x_8}$ and $h_{1,2} = g^{x_1x_2x_3x_4}$. And then we compute the level 2 elements from $h_{1,1}$ and $h_1, 2$, *i.e.,* we compute $h_{2,1} = h_{1,1}^{x_3x_4} = g^{x_3x_4 \cdot x_5x_6x_7x_8}$, $h_{2,2} = h_{1,1}^{x_1x_2} = g^{x_1x_2 \cdot x_5x_6x_7x_8}$, $h_{2,3} = h_{1,2}^{x_7x_8} = g^{x_7x_8 \cdot x_1x_2x_3x_4}$, and $h_{2,4} = h_{1,2}^{x_5x_6} = g^{x_5x_6 \cdot x_1x_2x_3x_4}$. The rest of computation is presented in Fig. 5.1.

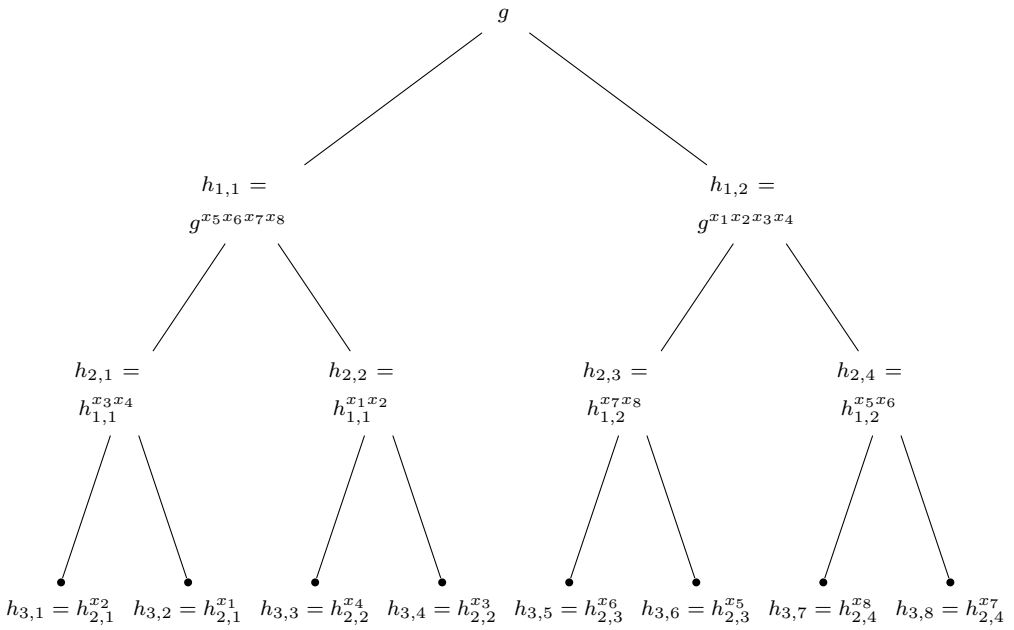


Figure 5.1: Product Tree when #Y=8

**The cost for precomputation:** at each level we need to perform $2^n = \#Y$ exponentiations. Since the total number of levels is $\log(\#Y)$, the algorithm requires $\#Y \log(\#Y)$ group exponentiations or $\#Y \log(\#Y) \log\log(\#Y)$ group operations. The space requirement during the algorithm execution is equal to the space required for the storage of the set $S$, *i.e.,* $\#Y \log(\#Y)$ group elements.

### 5.2.3 Complexity of the Full Algorithm

Recall Eq. (5.2.2),

$$h^{\prod_{y' \in Y - \{y\}} y'} \cdot \hat{g}^{-u} = \hat{g}^{v}.$$

If we have the set

$$\left\{ (y', h^{\prod_{y' \in Y - \{y\}} y'}) : y' \in Y \right\},$$

we can compute $(u, v, y')$ satisfying the above equation as the known order case. Then we have $\log_g h = y'(u + v)$.

The only difference from the known order case, we need to precompute the above set. This can be computed efficiently using the product tree method with $\#Y \log(\#Y)$ group exponentiations. Thus the total time and space complexities for solving the DLP increase by $\#Y \log(\#Y)$ both in the deterministic or randomized versions. However, this increment is almost negligible because $\log(\#Y) \leq n$ when $\#X \geq \#Y$.

We present pseudo-codes of procedures of unknown order case in Algorithm 5 (deterministic) and Algorithm 6 (randomized), respectively.

**Algorithm 5** Solving the LHWP-DLP of unknown order case with parameterize splitting systems (deterministic)

**Input:** $g$, $h \in G$, two subsets $X$ and $Y$ of $\mathbb{Z}_m$ with descriptions $n$ and $t$ such that $0 < t < n$

**Output:** $\log_g h$

1: Set $\Upsilon := \prod_{y \in Y} y$, $\hat{g} := g^{\Upsilon}$, $\hat{h} := h^{\Upsilon}$ and compute $\hat{g}^{-1}$
2: Initialize a table for precomputation $T'$
3: Store $\left\{ (y', h^{\prod_{y' \in Y - \{y\}} y'} = \hat{h}^{y'^{-1}}) : y' \in Y \right\}$ in $T'$
4: Substituting $\hat{g}$ for $g$ and $\hat{h}$ for $h$, execute Algorithm 3 (during execution, identify $\hat{h}^{y'^{-2}}$ with $h^{\prod_{y' \in Y - \{y\}} y'}$ in $T'$)

**Algorithm 6** Solving the LHWP-DLP of unknown order case with parameterize splitting systems (randomized)

**Input:** $g$, $h \in G$, two subsets $X$ and $Y$ of $\mathbb{Z}_m$ with descriptions $n$ and $t$ such that $0 < t < n$

**Output:** $\log_g h$

1: Set $\Upsilon := \prod_{y \in Y} y$, $\hat{g} := g^{\Upsilon}$, $\hat{h} := h^{\Upsilon}$ and compute $\hat{g}^{-1}$
2: Initialize a table for precomputation $T'$
3: Store $\left\{ (y', h^{\prod_{y' \in Y - \{y\}} y'} = \hat{h}^{y'^{-1}}) : y' \in Y \right\}$ in $T'$
4: Substituting $\hat{g}$ for $g$ and $\hat{h}$ for $h$, execute Algorithm 4 (during execution, identify $\hat{h}^{y'^{-2}}$ with $h^{\prod_{y' \in Y - \{y\}} y'}$ in $T'$)

# Chapter 6

# Cryptanalysis

In this chapter, we apply our algorithms to the private key of the GPS identification scheme and Hoffstein and Silverman's (2,2,11)-exponents. Girault and Lefranc suggested the use of LHWP exponents for the private key of the GPS identification scheme at CHES 2004. Coron, Lefranc, and Poupard gave an attack to this private key and proposed new parameters at CHES 2005. We give more efficient attack to both parameters using our algorithm. We also propose an attack to Hoffstein and Silverman's (2,2,11)-exponents. Our attack takes the smaller time and space complexity over the attack proposed by Cheon and Kim in 2008.

## 6.1 GPS identification Scheme

The GPS identification scheme, the only identification scheme in the recommended portfolio of the NESSIE project [GPS02], is an interactive protocol between a prover and a verifier which contains one or several rounds of three passes [GL04].

The GPS identification scheme is a (statistically) zero-knowledge protocol based on both discrete logarithm and integer factorization. As in many other DL-based schemes, the GPS scheme can be used in on-line/off-line manner [EGM89]: almost all the computations can be performed by the prover before the interaction with the verifier. But contrary to all the other DL-based schemes, it can be used in an on the fly manner [PS98]: the prover only has one multiplication and one addition to do, without any modular reduction, after the prover received the challenge from the verifier [GPS02].

### 6.1.1 The Scheme

We describe the GPS identification scheme briefly.

**Public parameters:**

- $N$: $N$ be a product of two primes that is hard to factorize,
- $g$: an element of ${\mathbb{Z}_N}^*$ of maximal order $m$,
- $S$: the upper bound of the binary size of secret keys. Typically S=160,
- $k$: the binary size of the challenges sent to the prover and determines the level of security of the scheme,
- $R$: the binary size of the exponents used in the commitment computation. Typically $R = S + k + 80$,

- $e$: the number of rounds the scheme is iterated. Theoretically, $e$ is a polynomial in the size of the security parameter. But, in practice, $e$ is often chosen equal to 1.

**Public/Private keys:**

- Private key: a non-negative integer $x$, whose binary size is at most $S$,
- Public key: $h = g^{-x} \bmod N$.

**Protocol:**

**step 1.** [**from the prover to the verifier**]
a round of identification consists for the prover in randomly choosing an integer $r$ in $[0, 2^R)$ and computing the commitment $W := g^r \bmod N$,

**step 2.** [**from the verifier to the prover**]
the prover sends $W$ to the verifier who answers a challenge $c$ randomly chosen in $[0, 2^k)$,

**step 3.** [**from the prover to the verifier**]
the prover computes $z := r + x \times c$ and send it to the verifier who checks $W = g^z h^c \bmod N$.

A complete identification consists in repeating $e$ times the elementary round. We present the scheme in Fig. 6.1. It was reported that the order of $g$ is kept secret and the answer $z$ is computed in $\mathbb{Z}$ in the NESSIE submission of the GPS scheme, "*GPS - An Asymmetric Identification Scheme for on the fly Authentication of Low Cost Smart Cards.*"

Parameters: $N$ a composite modulus, $g \in \mathbb{Z}_N^*$

Private key: $x$ non-negative $k$-bit integer

Public key: $y = g^{-x} \bmod N$

| **Prover** | | **Verifier** |
|---|---|---|
| choose $r \in [0,\ 2^R)$ | | |
| compute $W = g^r \bmod N$ | $\xrightarrow{W}$ | |
| | | choose $c \in [0,\ 2^k)$ |
| | $\xleftarrow{c}$ | |
| check $c \in [0,\ 2^k)$ | | |
| compute $z = r + x \times c$ | $\xrightarrow{z}$ | check $z \in [0,\ 2^R + 2^{k+S})$ |
| | | verify $g^z h^c = W$ |

Figure 6.1: The GPS identification scheme

### 6.1.2 The LHWP Private Keys

Assuming the commitment is precomputed, the efficiency of the protocol from the prover side depends on the computation cost of $z = r + x \times c$ carried by the prover in Fig. 6.1. For fast computation of the response, Girault and Lefranc suggested the use of a LHWP secret key [GL04]; that is, given a $S$-bit secret key $x$, we choose $\ell$ numbers, $x_1, \ldots, x_\ell$, where $x_i$ has bit-length $n_i$ and Hamming weight $t_i$. Here $S = \sum_{i=1}^{\ell} n_i$. If $c$ is a $k$-bit number, computing $z = r + x \times c$ requires $S + k + \sum_{i=1}^{\ell} t_i \times (k + \sum_{j=1}^{i-1} n_j)$ bit additions.

As a concrete example, in [GL04], a private key $x$ was proposed to be $x = x_1 x_2$, where $x_1$ is a 19-bit number with 5 random bits equal to 1, chosen from among the 16 least significant ones and $x_2$ is a 142-bit number with 16 random bits equal to 1, chosen from among the 138 least significant ones. With this private key, the prover should perform 1168 bit additions for computing $z = r + x \times c$. Later, in order to strengthen the security, $x_1$ and $x_2$ were proposed to be a 30-bit number with 12 nonzero bits and a 130-bit number with 26 nonzero bits, respectively [CLP05]. With this private key, the prover should perform 2188 bit additions.

### 6.1.3 Cryptanalysis

We attack the above parameters.

**Parameter settings:** we set

$$\#X_1 = \binom{16}{5}, \quad n_2 = 138, \quad t_2 = 16$$

for the private keys from [GL04] and

$$\#X_1 = \binom{30}{12}, \quad n_2 = 130, \quad t_2 = 26$$

| Method | Exponentiations | Storage |
|---|---|---|
| [GL04] | $2^{52}$ | $2^{33}$ |
| *Deterministic*, $t_s = 7$ | $2^{45.57}$ | $2^{37.41}$ |
| *Probabilistic*, $t_s = 7$ | $2^{44.57}$ | $2^{37.41}$ |

Table 6.1: Private keys from [GL04]

| Method | Exponentiations | Storage |
|---|---|---|
| [CLP05] | $2^{77.3}$ | $2^{43.9}$ |
| *Deterministic*, $t_s = 10$ | $2^{64.53}$ | $2^{54.58}$ |
| *Probabilistic*, $t_s = 9$ | $2^{61.82}$ | $2^{56.09}$ |

Table 6.2: Private keys from [CLP05]

for the private keys from [CLP05]. Since $N$ is public, we can easily compute $\hat{g}^{-1}$ of Eq. (5.2.2) using the extended Euclidean algorithm. We note that $t_s$ is chosen to minimize the time complexity.

**Results:** Table 6.1 and 6.2 compare the complexities of the processes of recovering the private keys for the scheme suggested in [GL04] and [CLP05], respectively.

For the private key suggested [GL04], Coron, Lefranc, and Poupard presented an attack requiring $2^{52}$ group exponentiations [CLP05]. But the parameterized splitting system and its randomized version reduce this further to $2^{45.57}$ and $2^{44.57}$, respectively.

Table 6.2 shows that a parameterized splitting system and its randomized version reduce the complexity of the DLP with the private key proposed in [CLP05] from $2^{77.3}$ to $2^{64.53}$ and $2^{61.82}$, respectively. We can use the better bound of $p$ in Theorem 4.2.1 because $t_2 \mid n_2$.

**A search for another key candidates:** we note that the private keys with $n_1 + n_2 = 160$ and $t_1 + t_2 \leq 44$ can be revealed in $2^{70}$ group exponentiations. Under these condition the strongest private key, whose security level is $2^{69.92}$, is obtained when $n_1 = 3$, $t_1 = 1$, $n_2 = 157$ and $t_2 = 43$. And the private keys with $t_1 + t_2 \leq 52$ can be revealed in $2^{75}$ group exponentiations. Under these condition the strongest private key is obtained when $n_1 = 3$, $t_1 = 1$, $n_2 = 157$ and $t_2 = 51$. This private key achieves a security level of $2^{74.94}$. We get the above results by applying a randomized version to all keys under a given condition.

## 6.2 Hoffstein and Silverman's Exponents

### 6.2.1 Silverman and Hoffstein's exponents

Hoffstein and Silverman proposed the use of exponent $x = x_1x_2x_3 \in \mathbb{Z}_{2^{1000}-1}$, where $x_1$, $x_2$ and $x_3$ are integers of $wt(x_1) = 6$, $wt(x_2) = 7$ and $wt(x_3) = 7$, called a (6,7,7)-exponent, or $wt(x_1) = 2$, $wt(x_2) = 2$ and $wt(x_3) = 11$ [HS03], called a (2,2,11)-exponent. When ignoring squaring, which is much faster than a multiplication in binary fields, the computation of $g^x$ requires 5+6+6=17 multiplications for a (6,7,7)-exponent and 1+1+10=12 multiplications for a (2,2,11)-exponent. For a (6,7,7)-exponent, all values of the Hamming weights are similar. Hence, splitting one of $x_i$ does not afford an advantage. Therefore, we focus on a (2,2,11)-exponent.

### 6.2.2 Cryptanalysis

As discussed in Section 3.2.2, authors of [CK08] proposed an attack with $2^{55.9}$ group exponentiations by storing $2^{54.5}$ elements. The key idea is to reduce the key space by giving an equivalence class over the space, called the rotation-free elements.

Our attack to a (2,2,11)-exponent also exploits the technique of [CK08]. According to the trick of [CK08], we convert the equation $y = g^{x_1x_2x_3}$ to $y^{2^t\bar{x}_1^{-1}\bar{x}_2^{-1}} = g^{x_3}$, where $0 \leq t < n = 1000$ and each of $\bar{x}_1$ and $\bar{x}_2$ is a rotation-free element in $\mathbb{Z}_{2^n-1}$. Then we split $x_3$ into $x_3 = x_4 + x_5$ using our parameterized splitting system with $wt(x_4) = t_s$ and $wt(x_5) = 11 - t_s$. We then have

$$y^{2^t\bar{x}_1^{-1}\bar{x}_2^{-1}} g^{-x_4} = g^{x_5}. \tag{6.2.1}$$

We take more operations to Eq. (6.2.1). By repeating squaring both sides

| Method | Exponentiations | Storage |
|---|---|---|
| [CK08] | $2^{55.9}$ | $2^{54.5}$ |
| *Ours* , $t_s = 4$ | $2^{53.02}$ | $2^{49.80}$ |

Table 6.3: Hoffstein and Silverman's (2,2,11)-exponent

of Eq. (6.2.1), we may assume that $x_4$ is just the first $n_s$ bits of $2^{t'}x_3$ for some $t'$. Then the complexity of the splitting systems is reduced by $n$. That is, it is sufficient to consider a string of length $n_s$ with weight $t_s$ and starting from 1 for $x_4$. Therefore the total time complexity for $t_s = 4$ is equal to

$$n \cdot \binom{\mathrm{RF}(n,2)+1}{2} \cdot \binom{n_s - 1}{t_s - 1} + \binom{n - n_s}{t - t_s} \approx 2^{53.02}$$

group exponentiations and the space complexity is equal to $2^{49.80}$. The second term of the left-hand side is obtained from a combination with repetition of $\mathrm{RF}(n, 2)$ elements choose 2. It is a deterministic algorithm, but has no less complexity than our randomized algorithm. We summarize the results in Table 6.3.

## 6.3 Implementation

The full implementation of the proposed attacks is not easy due to huge memory requirements. For example, the proposed attack in $\mathrm{GF}(2^{1000})$ for $(2, 2, 11)$-exponents requires $2^{49.80}$ memory, which amounts to about $2^{16}$ TBytes. It is too huge to store.

To verify the effectiveness of our attack and estimate the attack time in practice, we may try an implementation of our attacks for modified parameters requiring smaller time and storage complexity. We have chosen (2,2,11)-exponents because the change of the size of the base field is enough to reduce the complexity within practical bound. On $\mathrm{GF}(2^{61})$, we take $t_s = 4$ and the lookup table for right-hand side of Eq. (6.2.1) consists of about $2^{23.87}$ elements, which requires 0.25 GBytes memory. The number of $y^{2^t \bar{x}_1^{-1} \bar{x}_2^{-1}}$ of Eq. (6.2.1) is about $2^{25.17}$. Hence the time complexity is about

$$2^{23.87} + 2^{25.17} \approx 2^{25.66}.$$

The experiment was performed using the NTL [Sho] on a machine with a dual-core AMD Opteron 2.6 GHz CPU and 4 GBytes RAM. We have tested the attack for 200 number of randomly chosen $h$. The discrete log of each $h$ was computed in 219.64 seconds on average. More precisely, computing exponentiations for $2^{23.87}$ exponents and constructing the lookup table took 103.6 seconds. And computing on-the-fly and finding a match on the lookup table took 116.04 seconds.

A multiplication in $\mathrm{GF}(2^{1000})$ could be $16^2$ times slower than $\mathrm{GF}(2^{61})$ using a schoolbook multiplication method. Using a fast arithmetic, however, a multiplication in $\mathrm{GF}(2^{1000})$ is about 5 times slower than $\mathrm{GF}(2^{61})$ by our experiment. Hence the attack time on $(2, 2, 11)$-exponents in $\mathrm{GF}(2^{1000})$ is

estimated to be about

$$219.64 \cdot 5 \cdot 2^{53.02-25.66} \approx 2^{37.6} \text{ sec.}$$

We note that the attack on real parameters are possible only with sufficiently large memory allowing efficient read and write.

# Chapter 7

# Conclusion and Open Problems

In the thesis we have proposed parameterized splitting systems, which is a generalization and refinement of Coppersmith's splitting system. The flexibility in the choice of the size of a block allows easier control of the trade-off between time and space complexity for solving the DLP with LHWP exponents. Moreover, the property that such a block starts with one reduces the time complexity further.

In the generic group model, the computational complexity of the DLP constrained to a subset $S$ of a group $G$ is known to be lower-bounded by the square root of the cardinality of $S$ [Sho97, MMN06]. In [EN77], Erdös and Newman asked for finding a set that is resistant to the baby-step giant-step algorithm, *i.e.,* the computational complexity of the DLP on $S$ is larger than the square root of the cardinality of $S$.

A set of LHWP exponents is a good candidate for this problem. The attack on LHWP exponents using a parameterized splitting system is the most efficient of any previously known algorithms, but is still larger than the square root bound of the key space. In particular, when the secret exponent is the product of three integers with almost-equal Hamming weights, our

algorithm is far from the bound. It still remains open whether a set of LHWP exponents is an answer to the Erdös and Newman question.

So far today, all known efficient algorithms for the LHWP-DLP require the space complexity comparable to the time complexity while the ordinary DLP has space-efficient algorithms such as Pollard rho or kangaroo. It would be an important future research question to find a space-efficient algorithm for this problem.

# Bibliography

[AMOV91] G. Agnew, R. Mullin, I. Onyszchuk, and S. Vanstone, "An Implementation for a Fast Public-Key Cryptosytem," *J. Cryptology*, vol. 3, no. 2, pp. 63–79, 1991.

[CK08] J. Cheon and H. Kim, "Analysis of Low Hamming Weight Products," *Discrete Appl. Math.*, vol. 156, no. 12, pp. 2264–2269, Jun. 2008.

[CS84] D. Coppersmith and G. Seroussi, "On the Minimum Distance of Some Quadratic Residue Codes," *IEEE Trans. Inf. Theory*, vol. IT-30, no. 2, pp. 407–411, Mar. 1984.

[CLP05] J. Coron, D. Lefranc, and G. Poupard, "A New Baby-Step Giant-Step Algorithm and Some Application to Cryptanalysis," in *Proc. Cryptographic Hardware and Embedded Systems – CHES 2005*, vol. 3656, Lecture Notes in Computer Science, pp. 47–60, 2005.

[DH76] W. Diffie, M. Hellman, "New Directions in Cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[DSS] NIST, "Digital Signature Standard," FIPS PUB 186, 1994.

[ElG85] T. ElGamal, "A Public Key Cryptosystem and a Signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 4, pp. 469–472, Jul. 1985.

[EGM89] S. Even, O. Goldreich, and S. Micali, "On-line/off-line Digital Signatures," in *Proc. Advances in Cryptology – Crypto 1989*, vol. 435, Lecture Notes in Computer Science, pp. 263–277, 1990.

[EN77] P. Erdös and D. Newman, "Bases for Sets of Integers," *J. Number Theory*, vol. 9, no. 4, pp. 420–425, 1977.

[Gal12] D. Galbraith, "Mathematics of Public Key Cryptography ," Cambridge University Press; 1 edition, 2012.

[Gir91] M. Girault, "Self-Certified Public Keys," in *Proc. Advances in Cryptology – Eurocrypt 1991*, vol. 547, Lecture Notes in Computer Science, pp. 490–497, 1991.

[GL04] M. Girault and D. Lefranc, "Public Key Authentication with One Single (on-line) Addition," in *Proc. Cryptographic Hardware and Embedded Systems – CHES 2004*, vol. 3156, Lecture Notes in Computer Science, pp. 413–427, 2004.

[GPS02] M. Girault, G. Poupard, and D. Lefranc, "Some Modes of Use of the GPS Identification Scheme," presented at the 3rd NESSIE Workshop, Nov. 2002.

[Hei92] R. Heiman, "A Note on Discrete Logarithms with Special Structure," in *Proc. Advances in Cryptology – Eurocrypt 1992*, vol.658, Lecture Notes in Computer Science, pp. 454–457, 1992.

[HS03] J. Hoffstein and J. Silverman, "Random Small Hamming Weight Products with Application to Cryptography," *Discrete Appl. Math.*, vol. 130, no.1, pp. 37–49, 2003.

[Kra22] M. Kraitchik, "Théorie des nombres," *Gauthier–Villards*, 1922.

[KC08] S. Kim and J. Cheon, "A Parameterized Splitting System and its Application to the Discrete Logarithm Problem with Low Hamming Weight Product Exponents," in *Proc. Public Key Cryptography – PKC 2008*, vol. 4939, Lecture Notes in Computer Science, pp. 328–343, 2008.

[KC10] S. Kim and J. Cheon, "Parameterized Splitting Systems for the Discrete Logarithm," *IEEE Trans. Inf. Theory*, vol. IT-56, no. 5, pp. 2528–2535, May 2010.

[LL98] C. Lim and P. Lee, "A Study on the Proposed Korean Digital Signature Algorithm," *Proc. Advances in Cryptology – Asiacrypt 1998*, vol. 1514, Lecture Notes in Computer Science, pp. 175–186, 1998.

[MMN06] I. Mironov, A. Mityagin, and K. Nissim, "Hard Instances of the Constrained Discrete Logarithm Problem," in *Proc. Algorithm Number Theory Symposium – ANTS 2006*, vol. 4076, Lecture Notes in Computer Science, pp. 582–598, 2008.

[MS77] F. J. MacWilliams and N. J. A. Sloane, "The Theory of Error-Correcting Codes," Amsterdam, The Netherands: North-Holland, vol. MR 57:5408a; MR 57:5408b, p. 309, 1977.

[MvOV97] A. Menezes, P. van Oorschot, and S. Vanstone, "Handbook of Applied Cryptography," Boca Raton, FL: CRC, p. 128, 1997.

[Odl00] A. Odlyzko, "Discrete logarithms: The past and the future," *Des. Codes Cryptography*, vol. 19, no. 2, pp. 129–145, Mar. 2000.

[Pol78] J. Pollard, "Monte Carlo Methods for Index Computation (mod $p$)," *Math. Comput.*, vol. 32, pp. 918–924, 1978.

[Pol00] J. Pollard, "Kangaroos, Monopoly and Discrete Logarithms," *J. Cryptology*, vol. 13, no. 4, pp. 437–447, 2000.

[PH78] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over GF($p$) and its cryptographic significance," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 1, pp. 106–110, Jan. 1978.

[PS98] G. Poupard and J. Stern, "Security Analysis of a Practical "on the fly" Authentication and Signature Generation," in *Proc. Advances in Cryptology – Eurocrypt 1998*, vol. 1403, Lecture Notes in Computer Science, pp. 422–436, 1998.

[Sch91] C. Schnorr, "Efficient Signature Generation by Smart Cards," *J. Cryptology*, vol. 4, no. 3, pp. 161–174, 1991.

[Sha71] D. Shanks, "Class Number, a Theory of Factorization and Genera," *Proc. Symp. Pure Math.*, vol. 20, pp 415–440, 1971.

[Sho] V. Shoup, NTL: A Library for doing Number Theory [Online]. Available: http://www.shoup.net/ntl/

[Sho97] V. Shoup, "Lower Bounds for Discrete Logarithms and Related Problems," in *Proc. Advances in Cryptology – Eurocrypt 1997*, vol. 1233, Lecture Notes in Computer Science, pp. 256–266, 1997.

[Sho00] V. Shoup, "Practical Threshold Signatures," in *Proc. Advances in Cryptology – Eurocrypt 2000*, vol. 1807, Lecture Notes in Computer Science, pp. 207–220, 2000.

[Sti02] D. Stinson, "Some Baby-Step Giant-Step Algorithms for the Low Hamming Weight Discrete Logarithm Problem," *Math. Comput*, vol. 71, no. 237, pp. 379–391, 2002.

[SWD96] O. Schirokauer, D. Weber, and T. Denny, "Discrete logarithms: The effectiveness of the index calculus method," in *Proc. Algorithmic Number Theory Symposium – ANTS 1996*, vol. 1122, Lecture Notes in Computer Science, pp. 337–361, 1996.

[Tes01] E. Teske, "Square-root Algorithms for The Discrete Logarithm Problem (a survey)," In *Public-Key Cryptography and Computational Number Theory*, pp. 283–301, 2001.

[vOW99] P. van Oorschot and M. Wiener, " Parallel Collision Search with Cryptanalytic Applications," *J. Cryptology*, vol. 12, no. 1, pp.1–28, 1999.*J. Cryptology*, vol. 13, no. 4, pp. 437–447, 2000.

# 국문초록

이산대수 문제는 현대 공개키 암호에 있어 가장 중요한 수학적 기반 문제의 하나이다. 수많은 암호 시스템과 프로토콜들이 이산대수가 어렵다는 가정하게 설계 및 제안되고 있으며 이러한 연구는 활발하게 진행되고 있다.

이산대수 기반 암호 시스템의 효율성은 지수승 연산 속도에 직결된다. Hoffstein과 Silverman은 이산대수 문제가 정의된 군에서 빠른 지수승과 안전성을 보장하기 위해 해밍 웨이트가 작은 지수들의 곱(성긴 지수 곱)을 사용할 것을 제안하였다. 특히 GF($2^n$)에서의 제곱연산 그리고 Koblitz 타운 곡선에서의 두 배 연산은 각각의 군 연산보다 훨씬 빠르기 때문에 성긴 지수 곱을 사용하면 연산을 매우 가속화시킬 수 있다.

본 학위 논문에서는 성긴 지수 곱 이산대수 문제의 안전성을 분석한다. 현재의 성긴 지수 곱 이산대수 문제의 안전성 분석은 성긴 지수 이산대수 문제의 분석 기법에 의존하고 있는데 이로부터는 본래 문제의 정확한 안전성을 측정할 수 없다.

본 논문에서는 성긴 지수 곱 이산대수 문제의 안전성을 분석하기 위해 매개화된 분할 시스템을 이용하여성긴 지수 곱 이산대수 문제를 공격하는 효율적인 알고리즘을 제안한다. 제안 알고리즘은 현재까지 알려진 알고리즘 중 가장 빠른 시간 안에 성긴 지수 곱 이산대수 문제의 해를 찾는다. 실증적인 예로써 Coron, Lefranc 그리고 Poupard가 CHES 2005에서 제안한 GPS 인증 스킴의 비밀키와 Hoffstein과 Silverman이 제안한 (2,2,11)-지수에 대해 제안 알고리즘을 적용하여 각각에 대해 $2^{61.82}$ 그리고 $2^{53.02}$ 번의 군 연산을 사용하여 비밀키를 복구할 수 있음을 보인다.

**주요어휘:** 성긴 지수 이산대수 문제, 성긴 지수 곱 이산대수 문제, 매개화 분할 시스템

**학번:** No. 2005-20314