Rounding in lattices and its cryptographic applications

Dan Boneh
dabo@cs.princeton.edu
Department of Computer Science
Princeton University
Princeton, NJ 08544

Ramarathnam Venkatesan
venkie@bellcore.com
Bellcore Room-1C341B,
445 South Street,
Morristown NJ 07960

Abstract

We analyze a lattice rounding technique using a natural matrix norm. We present its application to proving in a non-uniform model the hardness of computing $2 \log\log p$ bits of the secret keys of Diffie-Hellman and related protocols from the public keys. Earlier in [2] it was shown that $\sqrt{\log p}$ bits are hard to compute.

1 Introduction

Lattice basis reduction techniques have proven to be very useful in diverse areas. Examples include cryptography, settling number theoretic conjectures, and diophantine approximation. Rounding a given vector to an approximately closest vector in a given lattice was first studied in this context by Babai [1]. Recently in [2] rounding in lattices was used to study the hardness of computing the most significant bits of secret keys obtained using the Diffie-Hellman protocol and related schemes. Motivated by this, we study a new lattice rounding technique which is used to improve on the results of [2] in a non-uniform model.

The Diffie-Hellman protocol [3] enables two parties Alice and Bob to exchange a secret. Alice and Bob choose private keys a and b respectively. Then, they exchange the values g^a and g^b and compute their joint secret key g^{ab} which they can both determine. To obtain the secret a passive eavesdropper must evaluate the function $\mathrm{DH}_g(g^a, g^b) = g^{ab}$, which is believed to be hard.

Typically Alice and Bob use their shared secret as the key in a fast block cipher such as DES or IDEA. A natural way to derive the key for the cipher would be to use a block of bits from g^{ab}. For example, if the prime p has a length of 1024 bits, one may use the 64 most significant bits of g^{ab}. An attacker, who may not be able to compute the whole g^{ab}, may nevertheless succeed in computing the part of the bits of g^{ab} and crack the session. Hence it is important to know if the most significant bits (MSB) of g^{ab} are secure from an adversary who knows both g^a and g^b.

In [2] the authors showed that the $\sqrt{\log p} + \log\log p$ bits of the Diffie-Hellman secret are as hard to compute as the entire secret. This was done by studying an abstract problem known as the *hidden number problem* or HNP for short. To define the problem we must first introduce some notation. Let p be a prime. For an integer x the notation $x \bmod p$ will refer to the unique integer $y \in [0, p-1]$ such that $x \equiv y \pmod p$. Given an integer $x \in [0, p-1]$ we denote by $\mathrm{MSB}_k(x)$ the unique integer t such that $(t-1) \cdot p/2^k \leq x < t \cdot p/2^k$. For example, $\mathrm{MSB}_1(x)$ is either 0 or 1 depending on whether x is smaller than or greater than $p/2$.

Let p be a prime and $g \in \mathbb{Z}_p^*$. For $\alpha \in \mathbb{Z}_p^*$ define the function $f_\alpha(x) = \mathrm{MSB}_k(\alpha \cdot g^x \bmod p)$. The *hidden number problem* is to find α in polynomial time given an oracle for f. Clearly one wishes to

solve this problem with a small a value of k as possible.

In [2] it was shown that the LLL algorithm enables one to solve HNP in polynomial time with $k = \sqrt{\log p} + \log\log p$. This fact was used to prove that computing the most significant bits of the Diffie-Hellman secret is as hard as computing the entire secret. This solution of HNP was sufficient to prove the same result for Okamoto's conference key sharing scheme [8], ElGamal's public key system [4] and Shamir's message passing scheme [6, pp. 96–97] all defined in the next section.

In this paper we improve the number of bits needed to solve HNP to $k = 2\log\log p$ in a non-uniform model. More precisely we present an algorithm which takes advice bits depending only on p and g and solves HNP in polynomial time for any α. Our results have applications to Okamoto's conference key sharing scheme and to a simple variant of ElGamal's public key system. We show that in a non-uniform model computing the $2\log\log p$ bits of the secret key in Okamoto's scheme is as hard as computing the entire key. The same result holds for the secret message in the ElGamal variant. Non-uniform reductions have been used previously to argue about the security of various encryption schemes (see e.g. Maurer [7]).

2 A Lattice Rounding Technique

We show that there exist a polynomial time algorithm which given advice bits depending only on p and g can solve the hidden number problem using an oracle returning only $k = 2\log\log p$ bits. This is stated in the following theorem:

Theorem 2.1 *Let p be a prime and g a generator of $\mathbb{Z}_p^*$. Set $k = \lceil 2\log\log p \rceil$. For a hidden α define the function $f(r) = \mathrm{MSB}_k(\alpha \cdot g^r \bmod p)$. Then there exists a polynomial time algorithm which takes a polynomial number of advice bits depending only on p, g and recovers α given an oracle for the function $f(r)$.*

The proof relies on a new analysis of a lattice rounding technique to solve the hidden number problem. We intend to query the oracle $f(r)$ at uniformly and independently chosen inputs $r_1, \ldots, r_d$. The value of d will be specified later. As a convention we assume that the output of the oracle $f(r)$ is automatically converted to a number a satisfying

$$|(\alpha \cdot g^r \bmod p) - a| < p/2^{k+1}$$

This can easily be done by applying the transformation $f(r) \longmapsto f(r) \cdot \frac{p}{2^k} + \frac{p}{2^{k+1}}$.

Set $t_i = g^{r_i} \bmod p$ then the oracle's answers will be converted to integers $a_1, \ldots, a_d, a_{d+1}$ satisfying

$$|(\alpha t_i \bmod p) - a_r| < R \qquad \text{and} \qquad |(\alpha \bmod p) - a_{d+1}| < R$$

where $R = p/2^{k+1}$. The value a_{d+1} is obtained by querying the oracle at the point $r = 0$. Our goal is to recover α from this information. Consider the lattice L spanned by the rows of the matrix:

$$L = \begin{pmatrix} p & 0 & \ldots & 0 & 0 \\ 0 & p & \ldots & 0 & 0 \\ & \vdots & & & \vdots \\ 0 & 0 & \ldots & p & 0 \\ t_0 & t_1 & \ldots & t_d & 1 \end{pmatrix}$$

Set $u = (a_1, \ldots, a_d, a_{d+1})$ and define the lattice vector $v_\alpha \in L$ by

$$v_\alpha = (t_1\alpha \bmod p,\ t_2\alpha \bmod p, \ldots, t_d\alpha \bmod p,\ \alpha)$$

Notice that by definition $\| u - v_\alpha \| \leq R\sqrt{d+1}$. In other words, the vector u is "close" to a lattice point v_α. We describe an algorithm to find the lattice point v_α given the vector u. Clearly α can be recovered once v_α is found.

To construct the vector v_α from u we intend to use a different basis of the lattice L. Let $b_1, \ldots, b_{d+1}$ be some basis of L. Given u the algorithm performs two steps.

1. Write $u = \sum_{i=1}^{d+1} y_i b_i$ for some $y_i \in \mathbb{R}$.

2. Set $v = \sum_{i=1}^{d+1} \lfloor y_i \rceil b_i$ where $\lfloor y_i \rceil$ is the integer closest to y_i.

The question is when does this type of rounding produce the right answer. In other words, when does $v = v_\alpha$? The next lemma gives a tight criteria for when this occurs. We need the following notation: for a matrix W whose columns are the vectors $w_1, \ldots, w_d$ define

$$L_{1,\infty}(W) \stackrel{def}{=} \max_i L_1(w_i)$$

where $L_1(w_i)$ is the L_1 norm of w_i.

Lemma 2.2 *Let A be the matrix whose rows are the basis vectors b_i. If $L_{1,\infty}(A^{-1}) < 1/(2R)$ then the vector v constructed above satisfies $v = v_\alpha$.*

Proof Since the b_i form a basis of L we know that there exists a vector $x \in \mathbb{Z}^{d+1}$ such that $xA = v_\alpha$. Let $y \in \mathbb{R}^{d+1}$ be the vector such that $yA = u$. We need to prove that the theorem's hypothesis implies $\lfloor y_i \rceil = x_i$ for every coordinate i. Here x_i, y_i are the i'th coordinates of the vectors x, y respectively. Since x_i is an integer it suffices to prove that $|x_i - y_i| < 1/2$ for all i.

Set $W = A^{-1}$ and $W = (w_{i,j})$. Then

$$\left\{ \begin{array}{c} xA = v_\alpha \\ yA = u \end{array} \right. \quad \Longrightarrow \quad (x-y)A = v_\alpha - u \quad \Longrightarrow \quad x - y = (v_\alpha - u)A^{-1} \quad \Longrightarrow$$

$$|x_i - y_i| \leq \sum_{j=1}^{d+1} |w_{j,i}| \cdot |(v_\alpha)_j - u_j| \leq R \sum_{j=1}^{d+1} |w_{j,i}| \leq R \cdot L_{1,\infty}(W) < \frac{1}{2}$$

This completes the proof of the lemma. □

At this point we need to recall the notion of the dual lattice. Let L be a lattice spanned by the vectors $b_1, \ldots, b_d \in \mathbb{R}^d$. The dual lattice of L, denoted L^*, is the set of vectors

$$L^* = \left\{ w \in \mathbb{R}^d \quad \text{s.t.} \quad \forall u \in L \ : \ (u, w) \in \mathbb{Z} \right\}$$

One can readily show that if A is the matrix whose rows are $b_1, \ldots, b_d$ then the dual lattice is spanned by the columns of A^{-1}. Using this terminology Lemma 2.2 implies that we must find a basis of the dual lattice whose $L_{1,\infty}$ norm is as small as possible.

By inverting (and transposing) the matrix L we obtain that the dual lattice of our lattice L is spanned by the rows of the matrix:

$$L^* = \frac{1}{p} \begin{pmatrix} 1 & 0 & \ldots & 0 & -t_1 \\ 0 & 1 & \ldots & 0 & -t_2 \\ & \vdots & & & \vdots \\ 0 & 0 & \ldots & 1 & -t_d \\ 0 & 0 & \ldots & 0 & p \end{pmatrix}$$

We prove that with high probability (over the choice of the random t_i) the lattice L^* has a basis with low $L_{1,\infty}$ norm.

Theorem 2.3 *Let p be a prime and $d > 4+\log p+\log\log p$. Let $t_1, \ldots, t_d$ be integers chosen uniformly and independently at random in the range $[0, p-1]$. Consider the corresponding lattice L^*. Then with probability $1/2$ there exists a basis of this lattice with $L_{1,\infty} < 3d\log p/p$.*

The proof relies on the following lemma.

Lemma 2.4 *Let $p, d, t_1, \ldots, t_d$ as above. Set $m = \lceil \log\log p \rceil + 4$ and $n = m + \lfloor \log p \rfloor$. By assumption $d > n$. Then with probability $1/2$ for all $k = m+1, m+2, \ldots, n$ there exist subsets $S_k \subseteq \{1, \ldots, k-1\}$ satisfying:*

$$2^{n-k} \leq t_k + \sum_{i \in S_k} t_i \bmod p < 2 \cdot 2^{n-k}$$

Proof Let $k \in [m, n]$ be an integer. For each non empty set $S \subseteq \{1, \ldots, k-1\}$ define the random variable $T_S = t_k + \sum_{i \in S} t_i \bmod p$. The set $\{T_S\}_{S \in \subseteq \{1,\ldots,k-1\}}$ is a collection of pair wise independent random variables each uniformly distributed in the range $[0, p-1]$. Similarly, define the indicator random variable X_S as

$$X_S = \begin{cases} 1 & \text{if} \quad 2^{n-k} \leq T_S < 2 \cdot 2^{n-k} \\ 0 & \text{otherwise} \end{cases}$$

For notational convenience we define $X_\emptyset$ to be the fixed value $1/2^{k-m+1}$. The set $\{X_S\}$ is also a collection of pair wise independent random variables with $E[X_S] = 2^{n-k}/p \geq 1/2^{k-m+1}$. We set $Z = \sum_S X_S$. Then by pair wise independence we obtain

$$E[Z] \geq 2^{k-1} \cdot \frac{1}{2^{k-m+1}} = 2^{m-2} \quad \text{and} \quad \mathrm{Var}[Z] = \sum_S \mathrm{Var}[X_S] < 2^{m-1}$$

Chebyshev's inequality now gives:

$$\Pr[Z < 1] \leq \Pr\left[|Z - E[Z]| > 2^{(m-3)/2}\sqrt{\mathrm{Var}(Z)}\right] < \frac{1}{2^{m-3}}$$

Hence, $Z \geq 1$ with probability at least $1 - \frac{1}{2^{m-3}}$. When $Z \geq 1$ there exists at least one set S_k satisfying the required condition. Therefore, the required S_k exists with probability at least $1 - \frac{1}{2^{m-3}}$. We conclude that the probability that S_k exists for all $k = m+1, m+2, \ldots, n$ satisfies

$$\Pr[\forall k \ : \ S_k \text{ exists}] \geq 1 - \frac{n-m}{2^{m-3}} > \frac{1}{2}$$

This concludes the proof of the lemma. □

Proof of Theorem 2.3 The proof follows by multiplying the matrix L^* above on the left by two unimodular matrices. Let n, m be defined as in Lemma 2.4. Since $d > n$ it follows that with probability at least $1/2$ there exist sets $S_{m+1}, S_{m+2}, \ldots, S_n$ satisfying the conditions of Lemma 2.4. For all $k \leq m$ and $k > n$ we define $S_k = \emptyset$. Define the $d+1 \times d+1$ matrix $U = (u_{i,j})$ by

$$u_{i,j} = \begin{cases} 1 & \text{if} \quad j \in S_i \ \text{ or } \ i = j \\ 0 & \text{otherwise} \end{cases}$$

All the elements on the diagonal of U are 1. Furthermore, since the sets S_k satisfy $S_k \subseteq \{1, \ldots, k-1\}$ for all k it follows that U is a lower diagonal matrix and hence unimodular. The rows of the matrix UL^* form a basis of the dual lattice L^*. Multiplying L^* by U on the left has the effect of replacing each basis vector b_k by the vector $b_k + \sum_{i \in S_k} b_i$.

Notice that the elements in the last column of UL^* may be reduced modulo p since the vector $(0, 0, \ldots, 0, p)$ is a basis vector. After rearranging the rows of UL^* and reducing the elements in the last column modulo p the resulting basis has the following structure:

$$\frac{1}{p}\begin{pmatrix} & & u_1 \\ & & \vdots \\ & & u_{n-m} \\ & B & t_1 \\ & & \vdots \\ & & t_m \\ & & p \end{pmatrix}$$

where B is a $d+1 \times d$ matrix with $0, 1$ entries. By the choice of the sets S_k we can arrange things so that the elements $u_1, \ldots, u_{n-m}$ satisfy $2^{k-1} \leq u_k < 2^k$. This property of the u_k enables one to write any integer x in the range $[1, 2^k]$ as $x = \sum_{i=1}^{k-1} x_i u_i$ where all $x_i = 0, 1, 2, 3$. Specifically, any integer x in the range $[0, p]$ can be written as $x = \sum_{i=1}^{n-m} x_i u_i$ where $x_i = 0, 1, 2, 3$. We refer to the x_i's as the digits of x to the base $u_1, \ldots, u_{n-m}$.

We now transform the above basis into a basis with low $L_{1,\infty}$ norm. Let $b_1, \ldots, b_{d+1}$ be the basis vectors. Let $x^{(k)}$ be the right most entry in the vector b_k, e.g. $x^{(1)} = u_1$, $x^{(2)} = u_2$, $x^{(d+1)} = p$, etc. Let $x_i^{(k)}$ be the digits of $x^{(k)}$ to the base $u_1, \ldots, u_{n-m}$. That is,

$$x^{(k)} = \sum_{i=1}^{k-1} x_i^{(k)} x^{(i)} \quad \text{with} \quad x_i^{(k)} = 0, 1, 2, 3$$

We replace the basis vector b_k by the vector $b_k - \sum_{i=1}^{k-1} x_i^{(k)} b_i$. The unimodular matrix which performs this transformation is:

$$V = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ -x_1^{(2)} & 1 & 0 & \ldots & 0 \\ -x_1^{(3)} & -x_2^{(3)} & 1 & \ldots & 0 \\ & \vdots & & \vdots & \\ -x_1^{(d+1)} & -x_2^{(d+1)} & -x_3^{(d+1)} & \ldots & 1 \end{pmatrix}$$

The resulting basis has the structure

$$\frac{1}{p}\begin{pmatrix} & & 1 \\ & & 0 \\ & B' & \vdots \\ & & 0 \end{pmatrix}$$

where B' is a $d+1 \times d$ matrix whose entries are all less than $3(n-m) < 3 \log p$ in absolute value. Thus, the $L_{1,\infty}$ norm of the resulting basis is at most $3d \log p/p$. This completes the proof of Theorem 2.3. □

Proof of Theorem 2.1 The non-uniform result of Theorem 2.1 readily follows by combining Lemma 2.2 and Theorem 2.3. Given a prime p and a generator g the advice bits is a set of integers $t_1, \ldots, t_d$ with $d = 2 \log p$ along with a basis of L^* whose $L_{1,\infty}$ norm is less than $3d \log p/p = 6 \log^2 p/p$. Theorem 2.3 shows that indeed there exist integers $t_1, \ldots, t_d$ for which the required basis exists[1]. Furthermore, the advice bits include the discrete-log of $t_1, \ldots, t_d$ to the base g.

[1]Theorem 2.3 shows even more; it shows that for most tuples $(t_1, \ldots, t_d)$ such a basis is known to exist.

Since the $L_{1,\infty}$ norm of the given basis of L^* is less than $6\log^2 p/p$, Lemma 2.2 shows that the rounding algorithm will find the correct α if the oracle returns $2\log\log p + O(1)$ bits. More precisely, the value of R in Lemma 2.2 must satisfy $R < p/12\log^2 p$ implying that the oracle must return at least $2\log\log p + 4$ bits. Notice that since the discrete-log of t_i base g is known, the oracle can indeed be queried at the appropriate points to return the most significant bits of $\alpha \cdot t_i \bmod p$. This completes the proof of Theorem 2.1. □

We note that constructing a basis satisfying the conditions of Lemma 2.3 is likely to be hard. It requires one to solve certain instances of the modular subset sum problem. Impagliazzo and Naor [5] note that these instances are likely to be hard on average.

3 An application to Diffie-Hellman related schemes

3.1 Encryption schemes related to the Diffie-Hellman protocol

Several encryption schemes related to the Diffie-Hellman protocol have been proposed in the literature. Here we describe three of these schemes. For convenience, we assume that the generator g and the prime p have been agreed upon by Bob and Alice and all the required inverses exist.

Shamir message passing scheme: To send a message m Alice picks a random r, and sends $y = m^r$ to Bob. Bob picks a random s and sends $z = y^s$ back to Alice. Alice sends $w = z^{r^{-1}}$ to Bob who computes $m = w^{s^{-1}}$. Here r^{-1}, s^{-1} denote inverses modulo $p-1$. Breaking this scheme requires computing $SH(g^{rs}, g^r, g^s) = g$.

Okamoto conference key sharing: Bob picks r at random and sends to Alice $x = g^r$. Alice picks a random s and sends $y = x^s$ back. Bob computes $y^{r^{-1}} = g^s$ which is the conference key they use. Since the conference key is determined by Alice's bits alone she can distribute the same key to all members of the conference. Cracking this scheme needs computing the function $OK_g(g^{rs}, g^r) = g^s$.

ElGamal public key encryption: Bob picks a random x and publishes $y = g^x$ as his public key. To send a message m to Bob, Alice picks a random r and sends g^r, my^r. Bob can decode the message by computing $my^r/(g^r)^x$. To break the scheme one has to compute the function $EL_g(g^x, g^r, mg^{xr}) = m$.

Modified ElGamal public key encryption: Bob picks a random x and publishes $y = g^x$ as his public key. To send a message m to Bob, Alice picks a random r and sends y^r, mg^r. Bob can decode the message by computing $mg^r/(y^r)^{x^{-1}}$. To break the scheme one has to compute the function $EL'_g(g^x, g^{xr}, mg^r) = m$.

The equivalence of computing the first three functions above to breaking Diffie-Hellman was studied in [9]. The methods of [9] can also be used to show that breaking the modified ElGamal public key system is equivalent to breaking the original ElGamal system.

Our non-uniform results can be applied to the modified ElGamal public key system and Okamoto's conference key scheme. It is still an open problem whether the same results apply to Shamir's message passing scheme. To apply our results to the above schemes one must first derive several relations satisfied by the corresponding functions. These relations are stated in the following simple lemma taken from [2]. The fourth relation is easily proved in a similar manner to the third.

Lemma 3.1 *The functions* $SH(a,b,c), OK_g(a,b), EL_g(a,b,c)$ *and* $EL'_g(a,b,c)$ *satisfy the following relations:*

$$\begin{aligned} SH(g^{xy}, g^{x+rxy}, g^y) &= g \cdot (g^y)^r \qquad (when\ \ \gcd(yr+1, p-1) = 1\ ) \\ OK_g(g^{y(x+r)}, g^y) &= g^x \cdot g^r \\ EL_g(g^{x+r}, g^y, mg^{xy}) &= m \cdot (g^{-y})^r \\ EL'_g(g^x, g^{x(y-r)}, mg^y) &= m \cdot g^r \end{aligned}$$

Unfortunately the non-uniform result of Theorem 2.1 can not be applied to the Diffie-Hellman protocol. However it can be applied to the Okamoto conference key sharing scheme and the modified ElGamal scheme described in Section 3.1. We obtain the following theorem which is much better than the results of [2] in terms of the number of bits needed.

Theorem 3.2 *Let* p *be a prime and* g *a generator of* $\mathbb{Z}_p^*$.

1. *There exist advice bits which depend only on the prime* p *and the generator* g *such that given an efficient algorithm to compute* $2\log\log p$ *MSB's of the conference key in the Okamoto system there is an algorithm to efficiently (in polynomial time) compute the entire key.*

2. *There exist advice bits which depend only on the prime* p *and the generator* g *such that given an efficient algorithm to compute* $2\log\log p$ *MSB's of the message in the modified ElGamal scheme there is an algorithm to efficiently (in polynomial time) compute the entire message.*

Proof We prove statement (2) of the theorem. Statement (1) follows using a similar argument. Let A be an efficient algorithm computing $\mathrm{MSB}_k(m)$ given g^x, g^{rx}, mg^r. Define $f(r)$ to be the function $f(r) = \mathrm{MSB}_k(m \cdot g^r \bmod p)$. Then

$$f(r) = \mathrm{MSB}_k(m \cdot g^r \bmod p) = \mathrm{MSB}_k\big(EL'_g(g^x, g^{x(y-r)}, mg^y)\big) = A(g^x, g^{x(y-r)}, mg^y)$$

The second equality follows by part (4) of Lemma 3.1. It follows that given g^x, g^{yx}, mg^y the algorithm A can be used to evaluate the function $f(r)$ for arbitrary r. Hence, according to Theorem 2.1 there exist advice bits depending only on p and g such that the hidden number problem can be solved in polynomial time. This shows that in a non-uniform model the secret message m can be found if the algorithm A exists. □

The theorem can not be applied to the Diffie-Hellman protocol or the other related schemes since the generator g in the associated hidden number problem is not fixed. For instance, the relation $\mathrm{DH}_g(g^{x+r}, g^y) = g^{xy} \cdot (g^y)^r$ shows that the generator $h = g^y$ in the associated hidden number problem is changed in every invocation of the protocol. The same applies to Shamir's message passing protocol.

3.2 The small generator case

We show that when the generator g is small HNP can be solved with an oracle which provides very few bits. Unlike the results of the previous section this result is uniform. It provides a "real" algorithm for solving HNP.

A similar theorem was presented in [2] however there the proof made use of a more refined oracle than an MSB oracle. Here we are able to prove a slightly weaker result, but we make use of the standard MSB oracle. The proof is based on the lattice rounding technique described in the previous section.

Theorem 3.3 *For fixed p and $g, \alpha \in \mathbb{Z}_p^*$ define $f(x)$ to be the function $f(x) = \mathrm{MSB}_k(\alpha \cdot g^x \bmod p)$ where $k = \log g + \log\log p + 2$. There exists a polynomial time algorithm (in $\log p$) which given access to an oracle computing the function $f(x)$ finds α.*

Proof Let L^* be the lattice spanned by the rows of the matrix:

$$A = \frac{1}{p}\begin{pmatrix} 1 & 0 & \dots & 0 & -g \\ 0 & 1 & \dots & 0 & -g^2 \\ & \vdots & & & \vdots \\ 0 & 0 & \dots & 1 & -g^d \\ 0 & 0 & \dots & 0 & p \end{pmatrix}$$

where $d > \log_g p$. Lemma 2.2 shows that to prove Theorem 3.3 it suffices to construct a basis of L^* with low $L_{1,\infty}$ norm. Write $p = \sum_{i=0}^{d} b_i g^i$ where all b_i satisfy $0 \le b_i < g$. This is possible since $d > \log_g p$. We multiply the basis A by a unimodular matrix to obtain a new basis B as follows:

$$B = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ -g & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & -g & 1 & \dots & 0 & 0 & 0 \\ & \vdots & & & \vdots & & \\ 0 & 0 & 0 & \dots & -g & 1 & 0 \\ b_1 & b_2 & b_3 & \dots & b_{d-1} & b_d & 1 \end{pmatrix} \cdot A = \frac{1}{p}\begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 & -g \\ -g & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & -g & 1 & \dots & 0 & 0 & 0 \\ & \vdots & & & \vdots & & \\ 0 & 0 & 0 & \dots & -g & 1 & 0 \\ b_1 & b_2 & b_3 & \dots & b_{d-1} & b_d & b_0 \end{pmatrix}$$

For this basis we have $L_{1,\infty}(B) = gd/p$. Therefore, taking $R = \frac{p}{3gd}$ in Lemma 2.2 guarantees that the rounding algorithm of Section 2 will recover α. Taking $R = \frac{p}{3dg}$ corresponds to an oracle which returns more than $\log 3dg$ of the most significant bits of αg^k. Since $\log 3dg \le \log g + \log\log p + 2$ we see that the hidden number problem can be solved using this many bits. □

We are hopeful that the above proof can be generalized to show that a small number of bits suffices to solve the hidden number problem for an arbitrary generator g. To do so one must construct a basis of L^* with low $L_{1,\infty}$ norm for a general g.

The implications of Theorem 3.3 to the Diffie-Hellman protocol are discussed in [2]. Based on Theorem 3.3 one can modify the Diffie-Hellman protocol to obtain a new protocol for which the $O(\log g + \log\log p)$ bits of the secret key are as hard to compute as the entire secret.

4 Conclusions

We presented an analysis of a lattice rounding technique which enabled us to solve the hidden number problem with $k = \log\log p$ bits in a non-uniform model. To prove the result we analyzed a class of random lattices and proved they contain a basis with low $L_{1,\infty}$ norm with high probability.

Our solution to HNP enabled us to give a non-uniform reduction from computing the most significant bits of the key in the Okamoto conference key sharing scheme to computing the entire key. We proved a similar result for the modified ElGamal public key system. It is still an open problem whether such results hold for the Diffie-Hellman protocol or the other schemes discussed in the introduction. Non-uniform reductions are of interest in cryptography since they apply to an adversary capble of long preprocessing. We also demonstrated an algorithm for solving HNP in the case of a small generator using an MSB oracle returning few bits.

It is our hope that the techniques described in this paper will be helpful in improving "real" algorithms for solving HNP. Eventually we hope that HNP can be solved using an MSB oracle returning one bit ($k = 1$). Such an algorithm will prove the hardness of computing the single most significant bit of the Diffie-Hellman secret.

Acknowledgments

We thank David Shmoys and Andrew Odlyzko for some helpful discussions.

References

[1] L. Babai. On Lovasz' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6:1–13, 1986.

[2] D. Boneh, Venkatesan R., "Hardness of Computing the Most Significant Bits of Secret Keys in Diffie-Hellman and Related Schemes", Proc. of Crypto, 1996.

[3] W. Diffie, Hellman E.M., "New directions in cryptography", IEEE. Trans. Inform. Theory, IT-22, No. 6, pp 644-654. (nov 1976)

[4] T. Elgamal, "A public-key cryptosystem and a signature scheme based on Discrete Logarithms", IEEE Trans. on Info. Theory, IT-31, No 4, pp 469-472. (july 1985).

[5] R. Impagliazzo, Naor M., "Efficient cryptographic schemes provably as secure as subset sum", Proc. FOCS 1989, pp. 236–241.

[6] N. Koblitz, "A course in Number Theory and Cryptography", Springer Verlag 1987.

[7] U. Maurer, "Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms", Proc. of Crypto, 1994, pp. 271–281.

[8] T. Okamoto, "Encryption and Authentication Schemes Based on Public Key Systems, Ph. D Thesis, The Univ. of Tokyo, (1988).

[9] K. Sakurai, Shizuya H, "Relationships among Computational powers of Breaking Discrete Log Cryptosystems", Eurocrypt 95, pp 341-351.