Improved Timing Attacks on ECDSA

Vikram Singh*

Abstract

We improve the timing attack on ECDSA in [1] by Brumley and Tuveri. We use the Gaussian heuristic to analyse the length of error vectors in the lattice Close Vector Problem in order to determine the problems which are theoretically solvable. Then we cost each solution using a strengthened lattice reduction algorithm and Schnorr-Euchner enumeration to determine which problems are practically solvable. The original work by Brumley and Tuveri resulted in OpenSSL's ECDSA being updated to remove the timing information they exploited, so that application is not vulnerable to our improvements. However we publish this work as a general advance in side-channel recovery techniques which may be applicable in related scenarios.

Keywords: Cryptography, Lattice, ECDSA, OpenSSL, TLS, Side-channel, Timing Attack, Digital Signature Scheme, HKZ-reduction, Lattice Enumeration

1 Introduction

Digital signature schemes such as ECDSA underpin authentication in cryptographic protocols such as TLS. Traditional cryptanalysis has a good understanding of the strength of such schemes. However, increasingly, side-channel information is being used to demonstrate exploitability of these schemes due to weaknesses in implementations. In [4], Howgrave-Graham and Smart described how a theoretical side-channel giving a small number of bits of the ephemeral private keys used in DSA, could be used to construct a lattice

*vs77814@gmx.com. Principal Consultant, VS Communications.

for which solving a Close Vector Problem recovers the static private key. Later, Brumley and Tuveri in [1], identified such a side-channel which manifests itself as a timing correlation to the number of leading zero ephemeral private key bits occurring in the Montgomery's ladder algorithm for elliptic curve multiplication in the OpenSSL 0.9.8o implementation of ECDSA. They demonstrated this experimentally to recover the static private key from a remote TLS server after of the order of a thousand signatures. In accordance with their recommendation for a mitigating countermeasure, the OpenSSL implementation was patched to remove the timing vulnerability.

Our work presents improvements to the original timing attack. We acknowledge that the results are no longer applicable to the implementation of Montgomery's ladder in OpenSSL, but the results may be relevant in related scenarios and nonetheless represent a general advance in the applicability and efficacy of lattice techniques to such side-channel recoveries.

We shall summarise Brumley and Tuveri's approach in Section 2, and present the lattice attack in Section 3. Our contribution focuses on solving the lattice attack, we do not repeat the timing attack. Brumley and Tuveri use LLL-reduction and Babai rounding to recover the closest vector and we aim to improve on this.

The problem space is parameterized by the number of signatures and the number of leading zero bits given by the side-channel. We begin with an analysis of the length of the error vector in Section 4, and use the Gaussian heuristic to determine the region of the problem space where the closest vector is indeed likely to be the desired solution. This region has a theoretical solution by CVP provided we can afford to find the closest vector. We assess the practical cost in Section 5 by considering an improved but more expensive reduction than LLL, namely HKZ-reduction, which interpolates an exact method to find shortest vectors in low-dimensional subspaces with the approximate method of LLL. We consider the cost of Schnorr-Euchner enumeration on this reduced basis: first to find the closest vector; and then by extending the search radius to enumerate more distant lattice vectors and reach the desired solution, thus extending the theoretically solvable region. We use the size of the search tree as a bound on the enumeration cost.

In Section 6 we discuss the implications for the amount of work required to recover static private keys for the Brumley and Tuveri experiments. Then, in Section 7, look at the implications for reducing the number of signatures collected. We present where future work could go in Section 8, by exploit-

ing the side-channel information we have not used, and costing improved enumeration methods. Finally, Section 9 concludes.

2 ECDSA Side-Channel

Let E be an elliptic curve and G be a point on E of prime order p. To sign a message m with the static private key x, the signer chooses an ephemeral private key y and computes

$$b \equiv (m + xf)y^{-1} \pmod{p}$$

where $f \equiv ([y]G)_x \pmod{p}$.

The most expensive part of signature generation is the scalar multiplication in the elliptic curve so it is important that this is implemented as efficiently as possible. However, differences in the time taken to compute $[y]G$ can also leak information about the ephemeral private key y.

Brumley and Tuveri [1] noted that for binary curves OpenSSL 0.9.8o used a Montgomery ladder whose length depended on $log_2(y)$. This meant that scalar multiplication, and so signature generation, was faster for ephemeral private keys that had a greater number of leading zeroes. Turning this around, by timing a large number of signatures they could assume that the fastest ones corresponded to ephemeral private keys with a certain number of leading zeroes. Given enough of these it was then possible to use the lattice attack from [4] to recover the ephemeral private keys and thus the static private key.

Brumley and Tuveri perform three experiments to gather timing information, all using OpenSSL with the NIST binary elliptic curve B-163, whose ephemeral private keys have 163 bits. The first is a local experiment whereby timings are taken in ideal conditions by directly invoking the ECDSA functions on a local machine. The second and third are remote experiments whereby a TLS handshake is performed over a network to a server, and the time between the TLS ClientHello and ServerKeyExchange is used as an approximation to the side-channel timing. For the second experiment the server is actually on the same host machine, connected to the client by a loopback interface. For the third experiment, the server is on a remote host machine but is connected to the same network switch. From all experiments, they form lattices as described in Section 3, using the signatures with the

fastest timings as an approximation to those whose ephemeral keys have a particular number, a, of leading zeroes.

The lattice CVP is solved using Sage's implementation of LLL-reduction to produce what is hopefully a nearly orthogonal lattice, then using Gram-Schmidt orthoganalization and Babai rounding to estimate the closest vector. In case the filtered signatures contain too many false positives - that erroneously have fewer than a leading zeroes - then a subset is randomly re-sampled from the fastest signatures until few enough false positives are present. "Few enough" is preferably zero, but they acknowledge the CVP sometimes solves in the presence of small errors because the desired solution can still be the closest vector.

We shall try to extract comparable results from their paper by focusing on 7 leading zeroes. They find that 43 correctly filtered signatures gives a lattice that solves with good probability in a short time. For each experiment they collect 8192 signatures and re-sample a subset of 43 from the fastest 64 until their CVP succeeds. We'll use their estimate of the number of random re-samplings until they select a filtered set with no false positives to estimate the work required. This is presented in Table 1. They find that recovery of the static private key is feasible for all their experimental scenarios, and performing 8192 signatures only takes a few minutes.

Experiment	Collected signatures	Work
Local attack	8192	5.6
Loopback attack	8192	100
Switched attack	8192	100000

Table 1: Average work (number of size 43 lattice attacks) to recover static private key given that the filtered set is selected from the top 64 timings

3 Lattice attack

Recall that, to sign a message m with the static private key x, the signer chooses an ephemeral private key y and computes

$$b \equiv (m + xf)y^{-1} \pmod{p}$$

where $f \equiv ([y]G)_x \pmod{p}$.

If we have n signatures, then we obtain a set of n equations

$$b_i \equiv (m_i + xf_i)y_i^{-1} \pmod{p}.$$

We can rearrange these to give a set of equations of the form

$$y_i + C_i x + D_i \equiv 0 \pmod{p}$$

where $C_i \equiv -f_i b_i^{-1} \pmod{p}$ and $D_i \equiv -m_i b_i^{-1} \pmod{p}$. Further, using the equation for $i = 0$ to cancel x we obtain

$$y_i + A_i y_0 + B_i \equiv 0 \pmod{p}$$

where $A_i \equiv -C_i C_0^{-1} \pmod{p}$ and $B_i \equiv D_i - D_0 C_i C_0^{-1} \pmod{p}$.

Suppose now that we know the leading bits of y_i; that is, for some μ_i we have

$$y_i = z_i + 2^{\mu_i} z_i''$$

where $0 \leq z_i < 2^{\mu_i}$ and z_i'' is known. This then gives

$$z_i + s_i z_0 + t_i \equiv 0 \pmod{p}$$

where $s_i = A_i$ and $t_i = 2^{\mu_i} z_i'' + A_i 2^{\mu_0} z_0'' + B_i$.

The key point is that the solutions z_i to the equations above are shorter than would be expected for random equations of this form. We can therefore attempt to recover the ephemeral private keys via a Close Vector Problem. Consider the lattice

$$A = \begin{bmatrix} -1 & s_1 & s_2 & \dots & s_{n-1} \\ 0 & p & 0 & \dots & 0 \\ 0 & 0 & p & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & p \end{bmatrix}.$$

Then we see that

$$(-z_0, k_1, \dots, k_{n-1})A - (0, t_1, \dots, t_{n-1}) = (z_0, \dots, z_{n-1})$$

where $k_i \in Z$ are such that $z_i + s_i z_0 + t_i = k_i p$. Thus, we want to find the lattice vector closest to the target vector $t = (0, t_1, \dots, t_{n-1})$.

4 Close vectors

There is an estimate in [4] of the number of signatures needed to recover the ephemeral private keys from the lattice under the assumption that the Close Vector Problem is being solved using Babai's nearest plane algorithm with an LLL-reduced basis. Similarly, [1] provides experimental evidence of success rates when Babai's rounding algorithm is used with an LLL-reduced basis. We will take a slightly different approach and consider the number of signatures required to guarantee that the solution to the Close Vector Problem gives the desired answer regardless of the method of solving it.

To be confident of recovering z from the Close Vector Problem we need that $||z||$ is at most half the length of the shortest vector in L. We shall use the standard notation of $\lambda_1(L)$ for the length of the shortest vector. The Gaussian heuristic estimates this for a random lattice of dimension n as

$$\lambda_1(L) \simeq \sqrt{\frac{n}{2\pi e}} \det(L)^{1/n}$$

where $\det(L)$ is the determinant of the lattice. In our case it is clear that $\det(L) = p^{n-1}$, so we have

$$\lambda_1(L) \simeq \sqrt{\frac{n}{2\pi e}} p^{(n-1)/n}.$$

Now we need to estimate $||z||$. For simplicity we will assume that $\mu_0 = \mu_1 = \cdots = \mu_{n-1} = \mu$. We then have an upper bound

$$||z|| \leq \sqrt{n} 2^{\mu}.$$

However, viewing the z_i as independent random variables chosen uniformly from the interval $[0, 2^{\mu} - 1]$, we see that the expected squared-length of z is

$$E(||z||^2) = \frac{n(2^{\mu+1} - 1)(2^{\mu} - 1)}{6}.$$

Consequently, we can bound the expected length of z by

$$E(||z||) \leq \sqrt{\frac{n(2^{\mu+1} - 1)(2^{\mu} - 1)}{6}} \simeq \sqrt{\frac{n}{3}} 2^{\mu} \tag{1}$$

and so we should expect z to be recoverable from the Close Vector Problem whenever

$$2^{\mu} \leq \sqrt{\frac{3}{8\pi e}} p^{(n-1)/n}.$$

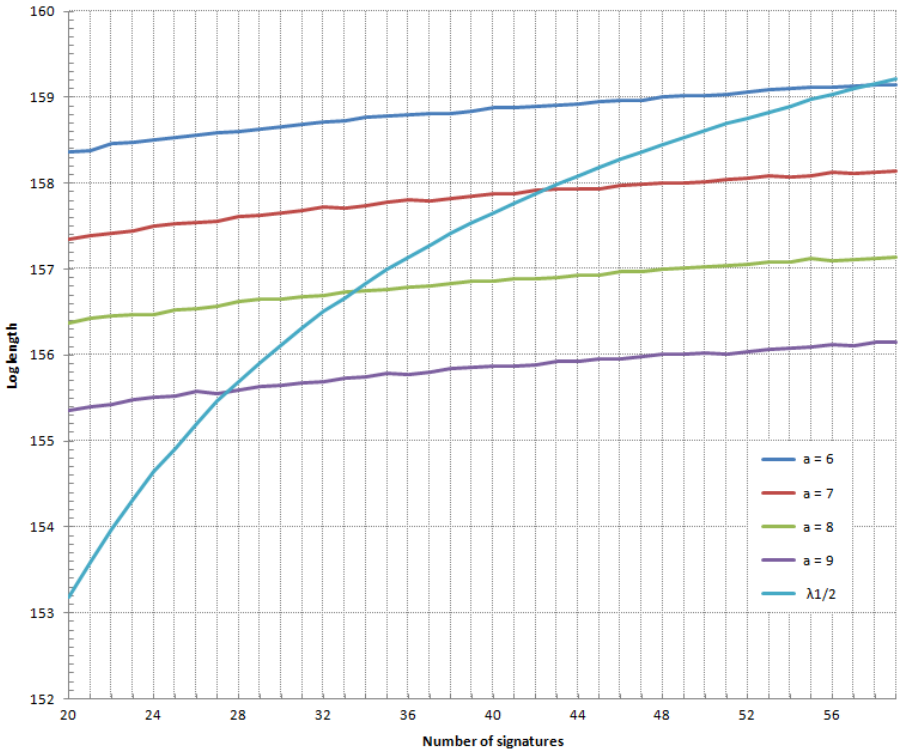

Figure 1: Log_2 length of error vector z, and half short vector, for B-163

In the case of the binary curve B-163 where we know $a = 163 - \mu$ leading bits of the ephemeral private keys this corresponds to the values in Table 2.

a	5	6	7	8	9
n	93	60	44	35	29

Table 2: Expected number of signatures required for B-163

Experiments show that the mean length of the observed error is close to the bound on the expected length, but that the Gaussian heuristic slightly underestimates the length of the actual shortest vector in the lattices. This means that the crossover points, see Figure 1, are marginally lower in practice, as shown by Table 3.

Note that for $a = 7$ this agrees with the observation in [1] that 43 signatures

a	6	7	8	9
n	59	43	34	28
prob	0.76	0.65	0.74	0.75

Table 3: Observed number of signatures required for B-163 and the probability of success

are needed to solve the lattice problem with a reasonable probability and for $a = 6$ this is a significant improvement on the results in [1].

5 Lattice enumeration

The dimension of the lattices we are using is low enough that we can consider lattice enumeration to solve the Close Vector Problem. In its simplest form, this is a depth-first search through a tree whose leaves are all the lattice vectors within a given distance R of the target vector. If R is chosen correctly then enumeration (without any form of pruning) is guaranteed to find the closest vector.

The cost of lattice enumeration is exponential and depends on the quality of the basis. Following the complexity analysis in [3], the number of nodes visited at level k by Schnorr-Euchner enumeration [5] can be estimated as

$$H_k = \frac{1}{2} \frac{V_k(R)}{\prod_{i=n+1-k}^{n} ||b_i^*||} \tag{2}$$

where b_i^* is the i-th basis vector after Gram-Schmidt orthogonalization, and $V_k(R) = R^k \pi^{k/2} \Gamma(k/2+1)^{-1}$ is the volume of the k-dimensional sphere of radius R. In our case, we will take R to be the bound on the expected length of the error vector from (1).

The ratio $||b_i^*||/||b_{i+1}^*||$ is approximately constant for a reduced basis, where this constant depends on the reduction algorithm. For example, [2] suggests that in practice LLL will produce a basis with $||b_i^*||/||b_{i+1}^*|| \simeq 1.0439$. The argument in [3] would then give an estimate of the number of nodes visited at level k as

$$H_k \simeq (1.0439)^{(n-k)k/2} 2^{O(n)}.$$

However, this is an asymptotic estimate and is not particularly accurate for the small dimensional lattices that we will be using.

Instead we will estimate the average number H of nodes visited by reducing the lattice for the curve B-163 and applying equation (2) directly, see Table 4. Again, as the dimension of our lattices is relatively low, we will use the stronger HKZ-reduction rather than the standard LLL-reduction.

a	6	7	8	9
n	59	43	34	28
$\log_2(H)$	12.6	7.44	4.92	3.55

Table 4: Estimated $\log_2$ number of nodes visited by enumeration for B-163 using HKZ-reduced bases

In fact, we can go further and estimate the cost of enumerating up to the bound given by (1) as we reduce the dimension of the lattices, see Figure 2. When the number of signatures drops below the crossover bounds specified in Table 3, we cannot guarantee that the ephemeral private keys are given by the shortest error vector. However, lattice enumeration will find all error vectors whose length is at most the expected length of z, so we can use secondary testing to identify the desired solution. This allows us to reduce the dimension of the lattices used while still limiting the total number of nodes visited by enumeration to a reasonable level, say 2^{16}. The number of signatures necessary are extracted into Table 5.

a	6	7	8	9
n	50	34	27	22
$\log_2(H)$	15.6	14.6	13.5	14.9

Table 5: Number of signatures required for B-163 with HKZ-reduced bases and lattice enumeration limited to 2^{16} nodes

In particular, we see that for $a = 7$ the lattice attack will succeed with 34 signatures, if we are willing to increase the number of nodes visited in lattice enumeration by a factor of around 2^7.

6 Implications

Brumley and Tuveri perform their attacks by collecting a large number of signatures and using the timing information to filter down a set of 64 signatures which they believe should correspond to ephemeral private keys with

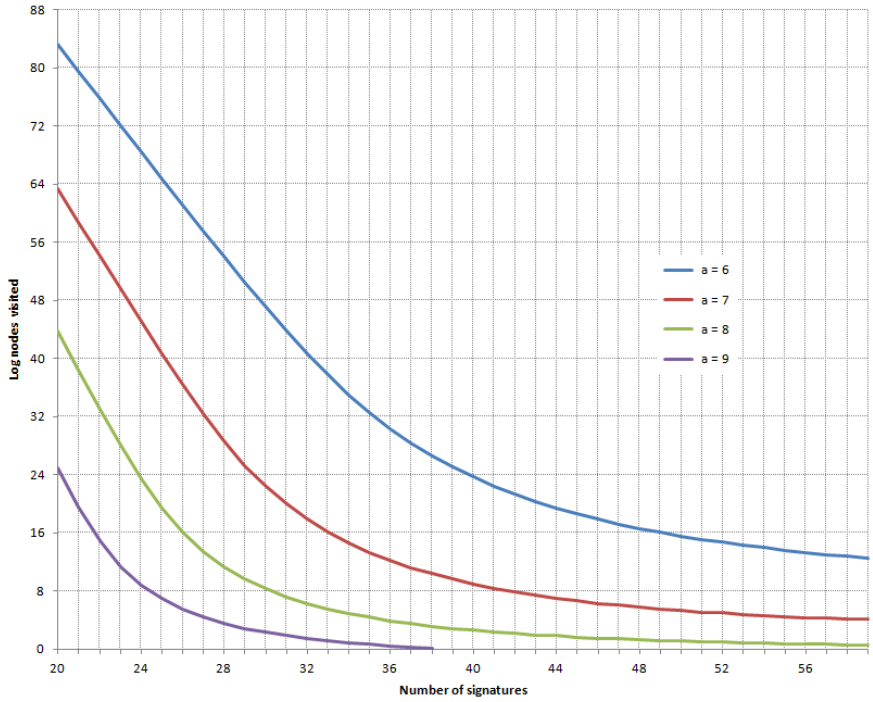

Figure 2: Estimated $\log_2$ cost of lattice enumeration up to the bound (1) using HKZ-reduced bases for B-163

the leading $a = 7$ bits set to zero. However, noise in the timing information means that their filtered set contains a number of false positives, as shown in Table 6.

N	4096	8192	16384
Local attack	17.92	1.48	0.05
Loopback attack	17.06	4.01	0.90
Switched attack	19.40	8.96	11.81

Table 6: Average false positive counts for B-163 with $a = 7$ from [1]

Their solution to the problem of false positives is to note that they only need 43 signatures for the lattice attack so they can select subsets of size 43 from the filtered set until they find one for which the lattice attack works. The cost of the attack can then be viewed as the number of attempts they expect to have to make before choosing a subset with no false positives, as shown

in Table 7.

N	4096	8192	16384
Local attack	41.0	2.40	0.08
Loopback attack	38.0	6.75	1.44
Switched attack	47.0	16.4	22.9

Table 7: $\log_2$ CVP attempts for B-163 with $a = 7$ and $n = 43$

We can improve on this in two ways. Firstly, from the previous section we see that the lattice attack can succeed with only 34 signatures if we are willing to use lattice enumeration with secondary testing. The lattice attack will be more expensive by a factor of 2^7, but for the remote attack over a switched network this is more than offset by the reduction in the number of times the lattice attack needs to be performed as shown in Table 8.

N	4096	8192	16384
Local attack	25.1	1.63	0.05
Loopback attack	23.6	4.55	0.98
Switched attack	28.0	10.9	14.9

Table 8: $\log_2$ lattice enumeration attempts for B-163 with $a = 7$ and $n = 34$

Secondly, we can weaken the assumption on the leading bits and instead perform the lattice attack for $a = 6$. Unfortunately, Brumley and Tuveri do not indicate how many of the false positives in their filtered set correspond to ephemeral private keys where the leading 6 bits are still all zero. We will estimate the number of "doubly false positives" by assuming it is proportional to the number of false positives in the filtered set; e.g. if there are 17.92 false positives in a filtered set of 64 then $5.02 = \frac{17.92}{64} \times 17.92$ of these will be doubly false positive. In this case, Table 5 tells us we need to select a set of 50 signatures with no doubly false positives for the lattice enumeration attack to succeed. We give these results in Table 9, presenting a considerable further improvement.

7 Reducing number of signatures

In practical terms, we can use lattice enumeration with secondary testing, or reduction of the number of assumed leading bits a, to decrease the work

N	4096	8192	16384
Local attack	11.9	0.07	0.00
Loopback attack	10.7	0.54	0.03
Switched attack	14.3	2.76	4.89

Table 9: Estimate $\log_2$ lattice enumeration attempts for B-163 with $a = 6$ and $n = 50$

required to recover the static private key or reduce the number of signatures that need to be collected, N. The probability of a B-163 signature having a leading zero bits is 2^{-a+1} because the top bit is (almost) always 0 due to the group size being only just larger than 2^{162}. However, timing noise means we cannot perfectly identify the signatures with a (or more) leading zeroes. *If* we could identify perfectly then the number of signatures for which we can expect to recover the static private key is given by Table 10 where we solve by closest vector, and by Table 11 where we do more work by solving by enumeration up to the nearest 2^{16} vectors.

a	6	7	8	9
n	59	43	34	28
N	1888	2752	4352	7168

Table 10: Expected total number of signatures required if no timing noise and solve by closest vector

a	6	7	8	9
n	50	34	27	22
N	1600	2176	3456	5632

Table 11: Expected total number of signatures collected if no timing noise and solve by enumeration up to 2^{16}

Thus, it's actually better to assume fewer leading zero bits.

8 Future Work

The timing attack lets us predict the number of lead zeroes, a_i, for a signature. Crudely we threshold by time and estimate that all surviving signatures have equally many lead zeroes, i.e. a_i are constant, whereas in practice,

shorter times have more lead zeroes. We could imagine scaling each column of the lattice by our best estimate of 2^{a_i}, and this would, in effect, re-balance for the varying a_i in the error norm. Where we are confident of a_i we could also include the information that the next bit down is set, reducing the number of unknown ephemeral private key bits by one. Further work thus first requires an analysis of the accuracy of the mapping from time to number of lead zeroes, and how this affects the lattice problem. The work should then subsequently exploit the full information given varying a_i for each signature.

We consider the size of the search tree as an upper bound on the lattice enumeration work. However, improvements which prune branches of the tree with a low probability of containing the solution, should be considered. Most notably, this should include the technique of Extreme Pruning defined by Gama, Nguyen and Regev in [3], which gives an order of magnitude complexity speedup compared to Schnorr-Euchner enumeration.

9 Conclusion

We have performed an analysis of the length of error vectors in the Brumley and Tuveri attack, and used it to determine the cost of recovering the static private key by Schnorr-Euchner lattice enumeration with secondary testing, even when the desired solution is not the closest lattice point to the target vector. We've shown this reduces the number of signatures required and allows us to solve the higher-dimensional problems arising where we assume fewer lead zero bits are set.

References

[1] B.B. Brumley and N. Tuveri. Remote Timings Attacks are Still Practical. *Proceedings of the 16th European conference on Research in computer security*, pages 355-371, Springer-Verlag, 2011.

[2] N. Gama and P.Q. Nguyen. Predicting lattice reduction. *EUROCRYPT'08*, pages 31-51, Springer-Verlag, 2008.

[3] N. Gama, P.Q. Nguyen, O. Regev. Lattice enumeration using extreme pruning. *EUROCRYPT'10*, pages 257-278, Springer-Verlag, 2010.

[4] N.A. Howgrave-Graham and N.P. Smart. Lattice Attacks on Digital Signature Schemes. *Designs, Codes and Cryptography*, Volume 63, Issue 3, pages 283-290, Kluwer Academic Publishers, 2001.

[5] C.P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematical Programming: Series A and B*, Volume 66, Issue 2, pages 181-199, Springer-Verlag, 1994.

[6] G. Hanrot and D. Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm. *CRYPTO'07*, pages 170-186, Springer-Verlag, 2007.