Secure Parameters for SWIFFT

Johannes Buchmann and Richard Lindner

Technische Universität Darmstadt, Department of Computer Science
Hochschulstraße 10, 64289 Darmstadt, Germany
buchmann,rlindner@cdc.informatik.tu-darmstadt.de

Abstract. The SWIFFT compression functions, proposed by Lyubashevsky *et al.* at FSE 2008, are very efficient instantiations of generalized compact knapsacks. They have the unique property, that *asymptotically* finding collisions for a random compression function implies being able to solve the *worst case* of computationally hard lattice problems.
We present two results. First, we present a parameter generation algorithm for SWIFFT, where n can be any integer in the image of Euler's totient function and not necessarily a power of 2 as before. Second, we give experimental evidence that finding *pseudo-collisions* for SWIFFT, is as hard as breaking a 87-bit symmetric cipher according to Lenstra's predictions. We then suggest conservative parameters, corresponding to 100-bit security.

Keywords: post-quantum cryptography, hash functions, lattices.

1 Introduction

On November 2nd 2007 the National Institute of Standards and Technology (NIST) announced a competition to develop a new cryptographic hash algorithm. The algorithm winning this competition will be called "SHA-3" and replace the standard hash functions in use today, namely MD5, SHA-1, and SHA-2. A main requirements for candidates in this competition is collision resistance. One such candidate is the SWIFFTX hash function [2], whose collision resistance relies on the collision resistance of the SWIFFT compression function family [8].

In this work we analyze the latter. Collisions in SWIFFT compression functions correspond naturally to vectors with ℓ_∞-norm bounded by 1 in certain lattices. We focus on attacks using lattice basis reduction algorithms. Since these algorithms are highly optimized to find small vectors in the Euclidean norm, it seems reasonable to analyze the computational problem of finding pseudo-collisions, i.e. vectors in the *smallest ball* which contains all vectors corresponding to collisions. We give experimental evidence that according to a well-known heuristic by Lenstra and Verheul [6] this problem as comparable to breaking a 87-bit symmetric cipher.

We also present a parameter generation algorithm for *efficient* SWIFFT compression function families, that works not only when the main parameter n is a power of 2, like in the original proposal, but also when $n = \varphi(k)$, where

$k > 0$ is some integer and φ is Euler's totient function. It was shown independently by Peikert and Rosen [10] as well as Lyubashevsky and Micciancio [7] that these compression functions enjoy the same asymptotic security argument as the orginal SWIFFT functions. Among all the resulting parameters, we suggest one, for which according to our experiments finding pseudo-collisions is at least as hard as breaking a 100-bit symmetric cipher.

1.1 Preliminaries

A lattice Λ is a discrete, additive subgroup of $\mathbb{R}^n$. It can always be described as $\Lambda = \{\sum_{i=1}^{d} x_i \, \mathbf{b}_i \,|\, x_i \in \mathbb{Z}\}$, where $\mathbf{b}_1, \ldots, \mathbf{b}_d \in \mathbb{R}^n$ are linearly independent. The matrix $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_d]$ is a *basis* of Λ and we write $\Lambda = \Lambda(\mathbf{B})$. The number of vectors in the basis is the dimension of the lattice.

2 SWIFFT compression functions

The SWIFFT compression function family was proposed by Lyubashevsky *et al.* at FSE 2008 [8]. They showed that its efficiency is comparable to SHA-2, while its collision-resistance is based on *worst-case* standard lattice problems asymptotically.

For a set of integer parameters (n, m, p), in their case $(64, 16, 257)$, they use the polynomial $f(x) = x^n + 1$, the ring $R_{p,n} = \mathbb{Z}_p[x]/(f(x))$, and the subset $D_n = \{0, 1\}[x]/(f(x))$ to define the family

$$\mathcal{H}_{n,m,p} = \left\{ h_{\widehat{\mathbf{a}}} \colon D_n^m \ni \widehat{\mathbf{x}} \longmapsto \sum_{i=1}^{m} \mathbf{a}_i \mathbf{x}_i \pmod{p} \;\middle|\; \widehat{\mathbf{a}} \in R_{p,n}^m \right\}.$$

The product $\mathbf{a}_i \mathbf{x}_i$ can be efficiently computed for all i, if an element ω of order $2n$, exists in $\mathbb{Z}_p$. This is guaranteed when $2n$ divides $p - 1$. For security reasons p is chosen to be prime, and n a power of two, making $x^n + 1$ irreducible over $\mathbb{Z}$.

Lyubashevsky and Micciancio showed in [7] that *asymptotically* these compression functions are collision-resistant, as long as standard lattice problem in lattices corresponding to ideals of $\mathbb{Z}[x]/(f(x))$ are hard in the *worst-case*.

2.1 More parameters

Let $n = \varphi(k)$, for some integer $k > 0$ with φ being Euler's totient function. Furthermore, let f be the kth cyclotomic polynomial, which is monic, irreducible over the integers, and has degree equal to n. Using the same structures as above, i.e. the ring $R_{p,n} = \mathbb{Z}_p[x]/(f(x))$, and subset $D_n = \{0, 1\}[x]/(f(x))$ with this new f, we can construct the same compression function family as above and the asymptotic security argument given in [10,7] still holds.

Now, if we chose a prime p, such that k divides $p - 1$, then similarly to the case above an element ω of order k exists in $\mathbb{Z}_p$ and the products $\mathbf{a}_i \mathbf{x}_i$ for all i can be efficiently computed using speedups described by the SWIFFT inventors

in [8]. Further *implementation-level* optimizations similar to the ones available for SWIFFT for these new sets of parameters are still an area of investigation.

We believe these optimizations are easiest to find when in the cases where n is either predecessor of a prime or a power of two. Focusing on these two special cases, we can show that this construction offers much more variety in the choice of parameters. See Table 1 for comparison of parameters where n is between 64 and 128.

2.2 SWIFFT Lattice

Let $\widehat{\mathbf{a}} \in R_{p,n}$. Consider the function $h_{\widehat{\mathbf{a}}}$ and extended the domain to $R_n = \mathbb{Z}[x]/(f(x))$. The coefficient vectors of the *periods* of this function form the set

$$\Lambda_p^{\perp}(\widehat{\mathbf{a}}) = \left\{ (x_1, \ldots, x_{nm}) \in \mathbb{Z}^{nm} \;\middle|\; h_{\widehat{\mathbf{a}}}\left(\sum_{i=0}^{n-1} x_{i+1} x^i, \ldots, \sum_{i=0}^{n-1} x_{m(i+1)} x^i \right) = \mathbf{0} \right\}.$$

This is a lattice of dimension nm, since the extended $h_{\widehat{\mathbf{a}}}$ is R_n-linear. A basis for this lattice can be found efficiently using a method described by Buchmann *et al.* [3]. Collisions in the original (unextended) function $h_{\widehat{\mathbf{a}}}$ correspond exactly to vectors in this lattice with ℓ_∞-norm bounded by 1. Therefore we refer to these lattices as SWIFFT lattices. A pseudo-collision is a vector in this lattice with Euclidean norm less than $\sqrt{nm}$. So every collision is a pseudo-collision, but not vice versa.

3 Parameter generation

We now describe an algorithm for generating parameter sets (n, m, p) for the SWIFFT compression function families in Section 2. For the polynomial f we will use the kth cyclotomic polynomial, such that $n = \varphi(k)$. If multiple polynomials are possible, we choose the one, where the resulting bitlength of the output is shorter, i.e. the one with smaller p. For example, if the first parameter n is a predecessor of a prime, we will use the polynomial $f(x) = x^n + x^{n-1} + \cdots + 1$, and if n is a power of two, we will use the polynomial $f(x) = x^n + 1$.

3.1 Algorithm

All parameter sets can be generated with Algorithm 1.

For each set of parameters, we may additionally compute the output bitlength $out = \lceil n \log_2(p) \rceil$, the compression rate $cr = m \log_2(p)$, the Hermite factor δ required for finding pseudo-collisions, and the minimal dimension d where we can expect to find pseudo-collisions. These values are listed in Table 1.

The two latter values δ and d are computed in the following fashion. Consider the function $len(d) = p^{n/d}\delta^d$. According to an analysis by Gama and Nguyen

Input: Integer n, s.t. $n = \varphi(k), k > 0$
Output: Parameters (n, m, p)

$l \leftarrow 1$
$p \leftarrow k + 1$
while *not isPrime(p)* **do**
    $l \leftarrow l + 1$
    $p \leftarrow l \cdot k + 1$
end
$m \leftarrow \lceil 1.99 \cdot \log_2(p) \rceil$

Algorithm 1: Parameter generation for $n = \varphi(k), k > 0$.

n	m	p	out	cr	δ	d
64	16	257	513	1.999	1.0085	205
66	17	269	533	2.106	1.0084	211
70	19	569	641	2.076	1.0073	247
72	17	293	591	2.074	1.0078	231
78	17	317	649	2.046	1.0072	250
82	15	167	606	2.032	1.0076	236
88	15	179	659	2.004	1.0071	255
96	18	389	826	2.092	1.0061	308
100	19	607	925	2.055	1.0056	340
102	19	619	946	2.049	1.0055	347
106	19	643	989	2.037	1.0053	361
108	21	1091	1090	2.081	1.0050	392
112	16	227	877	2.044	1.0058	325
126	18	509	1133	2.002	1.0048	407
128	20	769	1228	2.086	1.0045	434

Table 1. Parameters for $64 \leq n \leq 128$.

[5][1] this is the Euclidean size of the smallest vector we are likely to find when reducing a sublattice with dimension d of any SWIFFT lattice $\Lambda_p^\perp(\widehat{\mathbf{a}})$. Micciancio and Regev observed in [9] that this function takes its minimal value

$$len(d_{min}) = \delta^{2\sqrt{n\log(p)/\log\delta}} \qquad \text{for} \qquad d_{min} = \sqrt{n\log(p)/\log(\delta)}.$$

A pseudo-collision is a vector in $\Lambda_p^\perp(\widehat{\mathbf{a}})$ with Euclidean norm $\sqrt{nm}$. In order to find such a vector, we need a δ, s.t. $len(d_{min}) = \sqrt{nm}$. We say this is the Hermite factor required for finding pseudo-collisions, and the corresponding d_{min} is the minimal dimension, where we can expect to find a pseudo-collision. Note that these minimal dimensions, which we will work in are about 5 *times* smaller than the corresponding dimensions of the SWIFFT lattices.

3.2 Recommended parameters

We will give arguments in Section 4.2 that parameters with $d \geq 220$ correspond to SWIFFT instances, where finding pseudo-collisions is at least as hard as breaking a 100-bit symmetric cipher. Such a parameter set is given in the 4th row of Table 1, i.e. $(n, m, p) = (72, 17, 293)$. Concerning all attacks these parameters are more secure than the standard ones, and we recommend to use them when pseudo-collisions should be hard to find.

4 Security Analysis

In their original proposal of SWIFFT, Lyubashevsky *et al.* provide a first analysis of all standard attacks. However, attacks using lattice basis reduction algorithms like LLL/BKZ/RSR often behave much better in *practice* then their theoretical analysis suggests. We believe this is the case concerning SWIFFT lattices (cf. Section 2.2).

We will focus on this particular attack and give experimental evidence that the computational problem of finding pseudo-collisions corresponds to breaking a 87-bit symmetric cipher according to the predictions given by Lenstra and Verheul in [6].

In this section we will only consider the standard SWIFFT parameters

$$(n, m, p) = (64, 16, 257).$$

All SWIFFT lattices have dimension $nm = 1024$, but a sublattice of dimension $d = 205$ is sufficient to find pseudo-collisions (cf. Table 1).

[1] Their experiments were performed on random lattices following a different distribution, but experimentally their results apply here as well.

4.1 Existence of (pseudo-)collisions in d-dimensional sublattices

The method we have given in 3.1 for choosing the dimension of the sublattice we attack with lattice-basis reduction algorithms is a heuristic, because it is based on extensive experiments by Gama and Nguyen. We will now give a related result independent of experiments but dependent on the construction of SWIFFT lattices and other lattices arising from Ajtai's original proposal [1].

Let h be a random SWIFFT compression function with parameters (n, m, p). The range of this function has size $|R| = p^n$. We change the domain of h to all vectors in a d-dimensional subspace of $\mathbb{Z}^{nm}$ that have Euclidean norm less than $r = \sqrt{nm}/2$. The size of this space is approximately equal to the volume of a d-dimensional ball with radius r, i.e. $|D| = r^d \pi^{d/2} / \Gamma(d/2 + 1)$.

Now any collision in the modified h corresponds to a pseudo-collision of the corresponding SWIFFT function by the triangle inequality. These collisions exist for certain by the pigeonhole principle for all $d \geq 251$. For smaller d, we know the probability of a collision is

$$\Pr[\text{“pseudo-collision”}] = 1 - \frac{|R|!}{|D|^{|R|}(|R| - |D|)!} \geq 1 - e^{-|D|(|D|-1)/(2|R|)}$$

by the birthday argument. Some exemplary values of this expression are

d	94	95	$\cdots$	205	$\cdots$	251
Pr	0.46	0.99		$1 - 2^{-1.77 \cdot 10^{115}}$		1

So the dimension $d = 205$ suggested by the heuristic may be too pessimistic.

The situation for proper collisions is similar. Here, we shrink the input to all vectors in a d-dimensional subspace that have coefficients in $\{0, 1\}$. The size of this input space is $|D| = 2^d$. Again, collisions exist by the pigeonhole principle for all $d \geq 513$ and some exemplary probabilities are.

d	256	257	258	$\cdots$	513
Pr	0.32	0.79	0.99		1

This suggests that proper collisions are much harder to find.

4.2 Experiments

For our experiments we did not choose the lowest possible sublattice dimension described in the last subsection, but rather the dimension where lattice basis reduction algorithms like LLL/BKZ behave optimal in practice (see Section 3.1).

For our experiments, we fixed $n = 64$, $m = 16$ to their standard values and chose the third parameter p variable. This results in a steady decrease in the Hermite factor and increase in the dimension required to find pseudo-collisions (see Table 2). We found that for smaller values of p, corresponding to smaller values of d, pseudo-collisions were found too fast to make sensible measurements.

For each of these 9 parameter sets, we created 10 random SWIFFT lattices using the PRNG which is part of the “Number Theory Library” 5.4.2 (NTL) by

n	m	p	δ	d
64	16	29	1.0140	125
64	16	33	1.0135	130
64	16	37	1.0131	134
64	16	41	1.0127	138
64	16	45	1.0124	141
64	16	49	1.0121	144
64	16	53	1.0119	147
64	16	57	1.0117	150
64	16	61	1.0115	152

Table 2. Parameters used for our experiments.

Shoup [12]. We then proceeded to break all instances with the NTL floating-point variant of BKZ (bkzfp), by increasing the BKZ parameter β until a pseudo-collision was found and recording the total time taken in each case. We also broke all instances with a floating-point variant of Schnorr's RSR algorithm [11] (rsrfp) implemented by Ludwig [4][2] using the parameters $\delta = 0.9, u = 22$ and again increasing β until a pseudo-collision was found.

For each parameter set we computed the average runtime of both algorithms and plotted the $\log_2$ of this value relative to the dimension d. We also plotted a conservative extrapolation for the average runtime of rsrfp using the steepest observed slope fixed to the last known data point (see Figure 1, left).

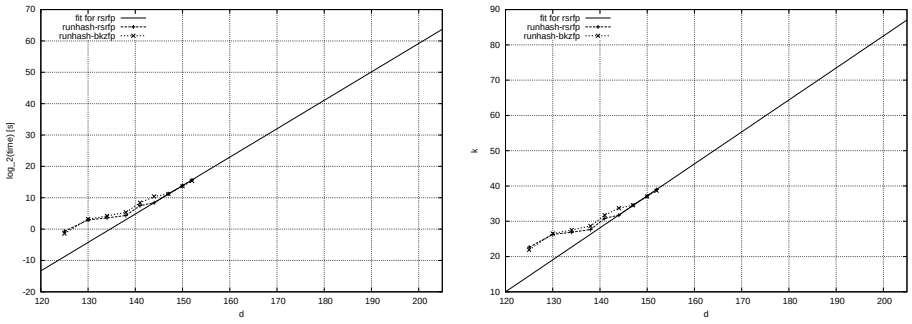

Fig. 1. Average runtime and corresponding symmetric bit-security of our experiments.

All our experiments where run on a single 2.3 Ghz AMD Opteron processor. According to the predictions of Lenstra and Verheul [6] the computational hardness of a problem solved after t seconds on such a machine is comparable

[2] Available on request from Ludwig.

to breaking a k-bit symmetric cipher, where

$$k = \log_2(t) + \log_2(2300) - \log_2(60 \cdot 60 \cdot 24 \cdot 365.25) - \log_2(5 \cdot 10^5) + 56.$$

We have plotted these k corresponding to the average runtime of each algorithm relative to the dimension d for each parameter set. Again, we also included the same conservative extrapolation (see Figure 1, right).

The rightmost side of both graphs correspond to $p = 257$, i.e. a real SWIFFT lattice. The extrapolated symmetric bit security for finding pseudo-collisions on these lattices is $k = 87.03$. Any parameter set, where $d \geq 220$ would correspond to a cipher with symmetric bit-security at least 100 according to our extrapolation. Parameters realizing this paradigm are given in Section 3.2.

4.3 Acknowledgments

We would like to thank Chris Peikert and Alon Rosen for helpful advice and encouragement.

References

1. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Annual Symposium on the Theory of Computing (STOC) 1996*, pages 99–108. ACM Press, 1996.
2. Yuriy Arbitman, Gil Dogon, Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. SWIFFTX: A proposal for the SHA-3 standard. `http://www.eecs.harvard.edu/~alon/PAPERS/lattices/swifftx.pdf`, 2008.
3. Johannes Buchmann, Richard Lindner, and Markus Rückert. Explicit hard instances of the shortest vector problem. In Johannes Buchmann and Jintai Ding, editors, *PQCrypto*, volume 5299 of *Lecture Notes in Computer Science*, pages 79–94. Springer, 2008.
4. Johannes Buchmann and Christoph Ludwig. Practical lattice basis sampling reduction. In Florian Hess, Sebastian Pauli, and Michael E. Pohst, editors, *ANTS*, volume 4076 of *Lecture Notes in Computer Science*, pages 222–237. Springer, 2006.
5. Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 31–51. Springer, 2008.
6. Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key sizes. *J. Cryptology*, 14(4):255–293, 2001.
7. Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In *International Colloquium on Automata, Languages and Programming (ICALP) 2006*, Lecture Notes in Computer Science, pages 144–155. Springer-Verlag, 2006.
8. Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. Swifft: A modest proposal for fft hashing. In *Fast Software Encryption (FSE) 2008*, Lecture Notes in Computer Science, pages 54–72. Springer-Verlag, 2008.
9. Daniele Micciancio and Oded Regev. *Post Quantum Cryptography*, chapter Lattice-based Cryptography. Springer-Verlag, 2009.

10. Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *Theory of Cryptography Conference (TCC) 2006*, Lecture Notes in Computer Science, pages 145–166. Springer-Verlag, 2006.
11. Claus-Peter Schnorr. Lattice reduction by random sampling and birthday methods. In Helmut Alt and Michel Habib, editors, *STACS*, volume 2607 of *Lecture Notes in Computer Science*, pages 145–156. Springer, 2003.
12. Victor Shoup. Number theory library (NTL) for C++. `http://www.shoup.net/ntl/`.