CRYPTANALYSIS OF A HASH FUNCTION BASED ON NORM FORM EQUATIONS

JEAN-PHILIPPE AUMASSON

Abstract. The cryptographic hash function Codefish is an outcome of a research project led by the company KRIPTO Research and the University of Debrecen in Hungary, and sponsored by the European Union and the Hungarian Government. It is commercialised by KRIPTO Research, for use in security protocols like digital signature, authentication, or message checksum. Codefish benefits of strong mathematical foundations, since it grounds its security on the difficulty of solving norm form equations. This paper demonstrates that it is insecure for cryptographic applications, by presenting practical attacks for solving the problems Second Preimage and Collision.

1. Introduction

Hash functions play a fundamental role in modern cryptography. They appear in a variety of protocols, be it for encryption, authentication, or message integrity; it is commonplace to refer to them as the Swiss Army knife of cryptographers.

A hash function maps a *message* of arbitrary length to some small fixed length bit string, called *hash-value, fingerprint, or digest*. For a hash function h to be *secure*, the three following problems should be hard to solve:

- Preimage: given a random image y, find x such that $h(x) = y$
- Second Preimage: given a random message x, find $x' \neq x$ such that $h(x) = h(x')$.
- Collision: find x and x' such that $h(x) = h(x')$ and $x \neq x'$.

Codefish is the product name of a cryptographic hash function introduced in 2004 by Bérczes, Ködmön, and Pethő [1]; it is an outcome of a project led by the company KRIPTO Research[1] and the University of Debrecen in Hungary, and sponsored by the European Union and the Hungarian Government.

The author is supported by the Swiss National Science Foundation, under project number 113329.

[1]See `http://www.kripto.hu/`.

Codefish grounds its security on the difficulty of solving norm form equations, claiming mathematical proofs that it is preimage and collision resistant—in other words, that the problems PREIMAGE and COLLISION are hard to solve, which implies that SECOND PREIMAGE is hard as well.

This paper shows that, despite the mathematical guarantees given, it is easy to find many preimages of zero for Codefish (i.e. solving COLLISION), and presents how to solve efficiently SECOND PREIMAGE. These results do not contradict the proofs of the designers, because they considered a non-standard notion of collision-resistance, similar to the notion of universal hashing.

Our attacks are practical and represent concrete threats when Codefish is used, e.g. , for digital signature, checksum, or authentication protocols. For example it allows an adversary to forge valid DSA signatures, and HMAC message authentication codes. The software commercialised should thus urgently be patched.

2. Description of Codefish

Let s be the product of two 512-bit primes, and n be a small integer. Define a *word* X_i as a an element of $\mathbb{Z}_s$, and a *block* as a n-tuple of words.

Codefish follows a rather unusual iterated scheme: it hashes $X_1 \dots X_\ell$, $\ell > n$, by computing $H_1 = \mathcal{N}(X_1, \dots, X_n)$, then

$$H_2 = \mathcal{N}(H_1, X_{n+1}, \dots, X_{2n-1}), H_3 = \mathcal{N}(H_1, X_{2n}, \dots, X_{3n-2})$$

and so on until X_ℓ is reached. Null inputs are added to fill the last block, if needed. The digest returned is the last H_i computed. A general formula of the recursion is

$$H_{i+1} = \mathcal{N}(H_i, X_{n+i(n-1)}, \dots, X_{n+(i+1)(n-1)}).$$

The *compression function* $\mathcal{N}$ returns the determinant in $\mathbb{Z}_s$ of the $n \times n$ circulant matrix constructed from the input words: e.g. to compress $X_1 \dots X_n$ it computes

$$\mathcal{N}(X_1, \dots, X_n) = \det \begin{pmatrix} X_1 & X_2 & \dots & X_n \\ X_n & X_1 & \dots & X_{n-1} \\ \dots & \dots & \dots & \dots \\ X_2 & X_3 & \dots & X_1 \end{pmatrix}.$$

A nice property of circulant matrices is that their eigenvalues are of the form $\sum_{i=1}^{n} X_i \omega^{i-1}$, where ω is here an n-th root of unity in $\mathbb{Z}_s$ (see [2,3] for more about circulant matrices).

On the KRIPTO company's website[2], the datasheet of Codefish specifies an additional parameter m, that allows $\mathcal{N}$ to take a reduced number of input words, i.e. redefining

$$\mathcal{N}(X_1, \ldots, X_m) = \det \begin{pmatrix} X_1 & X_2 & \ldots & X_m & 0 & \ldots & 0 \\ 0 & X_1 & \ldots & X_{m-1} & X_m & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ X_2 & X_3 & \ldots & 0 & 0 & \ldots & X_1 \end{pmatrix}.$$

This clearly speeds up the computation of determinants, but increases the number of calls to $\mathcal{N}$.

3. Homomorphic Property

It is well known that the inverse of a circulant matrix is circulant, and that the product of two circulant matrices is circulant. Furthermore, recall that for any $n \times n$ matrices A and B over the same abelian ring, we have $\det(A) \times \det(B) = \det(AB)$.

Let arbitrary blocks $X_1 \ldots X_n$ and $Y_1, \ldots, Y_n$. It follows that

$$\mathcal{N}(X_1, \ldots, X_n) \times \mathcal{N}(Y_1, \ldots, Y_n) = \mathcal{N}(Z_1, \ldots, Z_n)$$

with $Z_i = \sum_{j=1}^{n} X_j Y_{n-j+1+i}$, $i = 1, \ldots, n$, where indexes are reduced modulo n. We can thus find the digest of $Z_1 \ldots Z_n$ without computing explicitly $\mathcal{N}$, provided that $h(X_1 \ldots X_n) = \mathcal{N}(X_1 \ldots X_n)$ and $h(Y_1 \ldots Y_n) = \mathcal{N}(Y_1 \ldots Y_n)$ are known. This property is generally perceived as undesirable, though does not help to find collisions.

4. Finding Preimages of Zero

Observe that $\mathcal{N}(0, \ldots, 0) = 0$; hence, for any $\ell > 0$, if $X_1 = X_2 = \cdots = X_\ell = 0$ then $h(X_1 \ldots X_\ell) = 0$. This is because all intermediate hash values $H_1, \ldots, H_{\ell-1}$ are null, and the last block is padded with zeros. More generally, $\mathcal{N}(X_1, \ldots, X_n) = 0$ whenever the circulant matrix formed from $X_1 \ldots X_n$ has determinant 0 in $\mathbb{Z}_s$.

For example, if the additional parameter m is used, and set equal to n, then a one-block message $X_1 \ldots X_n$ has the zero hash value if $X_1 = X_2 = \cdots = X_n$.

If $m < n$, choosing $X_1 \ldots X_m$ with $X_1 = -X_2 \mod s$, X_1 and X_2 coprime with s, and $X_3 = \cdots = X_m = 0$, also leads to a matrix with null determinant.

The observation above can be used to find a preimage of zero from any given prefix: Let $X_1 \ldots X_\ell$ be an arbitrary message, and suppose ℓ is of the form $m + k(m-1)$; append the block $Y_1 \ldots Y_m$ with $Y_1 =$

[2]See `http://www.kripto.hu/kripto/codefish.html`.

$\cdots = Y_m = h(X_1 \dots X_\ell)$: $\mathcal{N}$ then finally computes the determinant of a constant matrix, so we get $h(X_1 \dots X_\ell Y_1 \dots Y_m) = 0$.

We demonstrated the easiness of finding preimages of the zero digest for Codefish; however, this does not break preimage resistance, because we are still not able to find preimages of a random image. Nevertheless, it clearly breaks its collision resistance.

5. Solving Second Preimage

Recall that for any matrix A the equality $\det(A^T) = \det(A)$ holds. For Codefish, we can exploit this property to solve the problem Second Preimage, as follows: Let $X_1, \dots, X_n$ be the first n words of the message received; write A the corresponding circulant matrix; compute A^T, and denote its first line $(Y_1, \dots, Y_n)$. We can now replace $X_1 \dots X_n$ by $Y_1 \dots Y_n = X_1 X_n X_{n-1} \dots X_2$ to form a second message with same hash value as the original one.

Why does this work? (1) since $X_1 \dots X_n$ is a random block, we have $X_1 \dots X_n \neq Y_1 \dots Y_n$ that holds with high probability, and (2) the circulant matrices constructed from these entries have the same determinant, from our preliminary observation. It follows $\mathcal{N}(X_1, \dots, X_n) = \mathcal{N}(Y_1, \dots, Y_n)$, and thus the second message created will have same digest as the received one, which solves the problem Second Preimage.

Another attack exploits a symmetry of the algebra of circulant matrices (see [3]). Denoting $\mathrm{circ}(X_1, \dots, X_n)$ the circulant matrix whose first row is $(X_1, \dots, X_n)$, we have for $0 < k < n$

$$\det \mathrm{circ}(X_0, \dots, X_{n-1}) = (-1)^{k(n-1)} \det \mathrm{circ}(X_k, X_{k+1}, \dots, \dots, X_{k-1}),$$

where $k+1, k-1$, etc. are reduced modulo n.

From the first block of a message it is thus easy to find a distinct block that has the same image by the compression function. For example, for an odd n and an arbitrary $X_1, \dots, X_n$ for which exist i, j such that $X_i \neq X_j$, we have

$$\mathcal{N}(X_1, \dots, X_n) = (X_2, \dots, X_n, X_1).$$

6. Conclusion

We have shown that the hash function Codefish is not collision-resistant, by presenting a simple method that finds many preimages of the null hash value. Our attack is practical and applies to all instances of Codefish.

We have then shown that Codefish is not second-preimage resistant. We presented two attacks that exploit algebraic properties of circulant

matrices. However, the parameters proposed for practical use make these attacks unfeasible for long messages. But they remain feasible for short messages; for example for the parameters propose $m = 5$, $n = 7$, the second-preimage attacks apply for message of at most 4 Kbits (because in this case using the symmetric property leads to a valid message block, i.e. with null blocks at the end).

References

[1] Bérczes, A., Ködmön, J., and Pethő, A. A one-way function based on norm form equations. *Periodica Mathematica Hungarica 49*, 1 (2004), 1–13.

[2] Bini, D., Corso, G. M. D., Manzini, G., and Margara, L. Inversion of circulant matrices over $\mathbf{Z}_m$. *Mathematics of Computation 70*, 235 (2001), 1169–1182.

[3] Geller, D., Kra, I., Popescu, S., and Simanca, S. On circulant matrices. `http://www.math.sunysb.edu/ sorin/eprints/circulant.dvi`.