J. Cryptology (2007) 20: 389–392
DOI: 10.1007/s00145-007-0564-4

Journal of
CRYPTOLOGY

Note

Theoretical Cryptanalysis of the Klimov–Shamir Number Generator TF-1*

Boaz Tsaban
Department of Mathematics, Weizmann Institute of Science,
Rehovot 76100, Israel
boaz.tsaban@weizmann.ac.il
and
Department of Mathematics, Bar-Ilan University,
Ramat-Gan 52900, Israel

Communicated by Eli Biham

Received 29 December 2005 and revised 25 September 2006
Online publication 25 June 2007

Abstract. The internal state of the Klimov–Shamir number generator TF-1 consists of four words of size w bits each, whereas its intended strength is 2^{2w}. We exploit an asymmetry in its output function to show that the internal state can be recovered after having 2^w outputs, using $2^{1.5w}$ operations. For $w = 32$ the attack is practical, but for their recommended $w = 64$ it is only of theoretical interest.

Key words. Pseudorandom number generators, T-functions, TF-1.

1. Generalized TF-1 Generators

The *Klimov–Shamir number generator TF-1* was introduced in [3] and is based on the methods developed in [2] and the references therein. This is an iterative pseudorandom number generator. Its internal state consists of four words a, b, c, d, of size w bits each. C_1, C_2, C_3, C are fixed constants chosen to optimize several properties (which are not relevant for our analysis). The update function of the generator is defined as follows:[1]

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} \mapsto \begin{pmatrix} a & \oplus & s & \oplus & 2c \cdot (b \vee C_1) \\ b & \oplus & s \wedge a & \oplus & 2c \cdot (d \vee C_3) \\ c & \oplus & s \wedge a \wedge b & \oplus & 2a \cdot (d \vee C_3) \\ d & \oplus & s \wedge a \wedge b \wedge c & \oplus & 2a \cdot (b \vee C_1) \end{pmatrix},$$

where

$$s = (C + (a \wedge b \wedge c \wedge d)) \oplus (a \wedge b \wedge c \wedge d).$$

* Supported by the Koshland Center for Basic Research.

[1] In the following description, $\wedge$, $\vee$, $\oplus$ denote bitwise logical *and*, *or*, and *xor*, respectively, and addition and multiplication are always carried modulo 2^w.

After each update, an output value

$$S(a + c) \cdot (S(b + d) \vee 1)$$

is extracted, where S is the function swapping the upper and lower halves of its input, i.e., $S(x) = x/2^{w/2} + x \cdot 2^{w/2}$ for each $x = 0, \ldots, 2^w - 1$ where "/" denotes integer division.

Earlier variants of this generator were cryptanalyzed in several works, see, for example, [4] and [1]. None of the earlier attacks applies to the present generator, though, since the present output function is more complicated. We present an attack on a generalized family of TF-1 generators, containing the Klimov–Shamir generator as a particular case.

Definition 1 [2]. $T\colon \{0, 1\}^{m\times w} \to \{0, 1\}^{n\times w}$ is a *T-function* if, for each $k = 1, \ldots, w$, the first k columns of $T(X)$ depend only on the first k columns of X.

Note that, using the convention that words from $\{0, 1\}^w$ are written such that the leftmost bit is the least significant one, the update function of a TF-1 generator is a T-function.

Following is a generalization of the family of TF-1 generators. The fact that we pose no restriction on its function F (and still are able to cryptanalyze it as shown below) seems to be of special interest.

Definition 2. A *generalized TF-1* generator consists of an update function $T_1\colon \{0, 1\}^{4\times w} \to \{0, 1\}^{4\times w}$ and output auxiliary functions $T_2, F\colon \{0, 1\}^{4\times w} \to \{0, 1\}^w$. T_1 and T_2 are T-functions, but F can be any efficiently computable function. Its internal state is a matrix $A \in \{0, 1\}^{4\times w}$. The update function is

$$A \mapsto T_1(A).$$

After each update, an output value

$$S(T_2(A)) \cdot (F(A) \vee 1)$$

is extracted.

2. Cryptanalysis

Generators with poor statistical properties are not suitable for cryptographic usage. We therefore restrict attention to the nondegenerate cases.

Lemma 3. *Assume that* $T\colon \{0, 1\}^{4\times w} \to \{0, 1\}^w$ *is a* (*mildly*) *random-looking T-function*, $k, l \in \{1, \ldots, w\}$, *and* $l \le k$. *If the first* $l - 1$ *columns of* X *are known and* $T(X) = 0$, *then the list of all possibilities for columns* $l, \ldots, k$ *of* X *can be enumerated in* (*roughly*) $2^{3(k-l)}$ *operations*.

Proof. First check all 2^4 possibilities for the lth column of X. Only about 2^3 should give 0 at the lth bit of $T(A)$. For each of them, check all 2^4 possibilities for the $(l + 1)$th

bit. Again about 2^3 of which will survive. Continue in this manner. The total number of operations is roughly

$$2^4 + 2^3 \cdot 2^4 + (2^3)^2 \cdot 2^4 + \cdots + (2^3)^{k-l-1} \cdot 2^4 \approx 2 \cdot 2^{3(k-l)}.$$

Note that there is no need to store the resulting tree in memory, since the search in the tree could be of "depth-first" type, i.e., follow each branch up to its end before moving to the next branch. □

Remark 4. For the function $T((a, b, c, d)^t) = a + c$ used in TF-1, the enumeration as in Lemma 3 is trivial: just enumerate $(a, b, -a, d)^t$ where $a, b, d \in \{0, 1\}^k$. Note further that 0 plays no special role in the proof of Lemma 3 and it can be replaced by any constant.

Theorem 5. *Assume that G is a generalized TF-1 generator which is (mildly) random-looking. Then the internal state of G can be recovered from roughly 2^w output words, using roughly $2^{1.5w}$ operations.*

Proof. Scan the output sequence until an output word 0 is found (this requires roughly 2^w output words). Denote the internal state at this point by A. Then

$$S(T_2(A)) \cdot (F(A) \vee 1) = 0.$$

As $F(A) \vee 1$ is relatively prime to 2^w, we have that $S(T_2(A)) = 0$, and therefore $T_2(A) = 0$.

Use Lemma 3 with $l = 1$ and $k = w/2 + 1$ to enumerate the 2^{3k} possibilities for the first k columns of A. During the enumeration, compute for each possibility the first k columns of $A' = T_1(A)$ and of $T_2(A')$. The kth bit of $T_2(A')$ should be equal to the least significant bit of the next output word. This rules out about half of the suggested solutions. Checking about one more step will rule out about half of the remaining solutions, etc. Algorithmically, continue updating and checking until a contradiction is found (or until a solution survives more than $3k$ steps) and then move to the next suggested solution. On average this requires two steps per suggested solution.

Having completed the above 2^{3k+1} operations, the first k columns of A are known. Use Lemma 3 again to go over all possibilities for columns $k + 1, \ldots, w$ of A. Now there are only 2^{3k-6} possibilities, and each of them gives a complete knowledge of the internal state and can thus be checked by computation of one or two output words. The total amount of operations is roughly

$$2^{3k+1} + 2^{3k-6} \approx 2^{3k+1} = 2^{1.5w+4} = 16 \cdot 2^{1.5w}.$$ □

3. Examples

Any generalized TF-1 generator for words of 32 bits has an internal state of size 128 bits and intended strength 2^{64}. By Theorem 5, the whole internal state can be recovered from 2^{32} output words (i.e., 16 gigabytes) using $16 \cdot 2^{1.5 \cdot 32} = 2^{52}$ operations. These parameters are practical.

Any generalized TF-1 generator for words of 64 bits has an internal state of size 256 bits and intended strength 2^{128}. By Theorem 5, the internal state can be recovered from 2^{64} output words using $16 \cdot 2^{1.5 \cdot 64} = 2^{100}$ operations. In this setting, our attack is only of theoretical interest.

Acknowledgments

We thank Alexander Klimov and the referees for their comments.

References

[1] V. Benony, F. Recher, E. Wegrzynoski, and C. Fontaine, Cryptanalysis of a particular case of Klimov–Shamir pseudo-random generator, in: *SETA* 2004, pp. 313–322, LNCS 3486, Springer-Verlag, Berlin, 2005.

[2] A. Klimov and A. Shamir, New cryptographic primitives based on multiword T-functions, in: *Fast Software Encryption*: 11*th International Workshop*, pp. 1–15, LNCS 3017, Springer-Verlag, Berlin, 2004.

[3] A. Klimov and A. Shamir, The TF-i family of stream ciphers, Handout distributed at The State of the Art of Stream Ciphers – SASC 2004.

[4] J. Mitra and P. Sarkar, Time-memory trade-off attacks on multiplication and T-functions, in: *ASIACRYPT* 2004, pp. 468–482, LNCS 3329, Springer-Verlag, Berlin, 2004.