

The Frogbit Cipher Submission Facts Sheet

Thierry Moreau

April 2005

CONNOTECH Experts-conseils Inc.
9130 Place de Montgolfier
Montreal, Qc
Canada H2M 2A1

Tel.: +1-514-385-5691
Fax: +1-514-385-5900
e-mail: thierry.moreau@connotech.com

Table of contents

1.	Background and Status of this Proposal	- 2 -
2.	Contents of the Frogbit Cipher Submission	- 2 -
3.	No Hidden Weaknesses	- 3 -
4.	Intellectual Property Statement	- 3 -
5.	The Frogbit ECRYPT NoE Reference Implementation	- 3 -
5.1	Overview	- 3 -
5.2	The Frogbit PRNG Design	- 4 -
5.3	The Key Schedule	- 4 -
5.4	The MAC Computation	- 6 -
5.5	Test Vectors	- 6 -
6.	The Frogbit Challenge Program	- 8 -
7.	References	- 8 -

1. Background and Status of this Proposal

The Frogbit cipher was developed in 1996. Although it led to no scientific publication, the design was not kept secret. The ECRYPT NoE terms of reference for stream ciphers provides the opportunity to submit the Frogbit cipher in the context of a well-defined cipher application scheme.

The Frogbit cipher is deemed relevant to the ECRYPT NoE stream cipher efforts for the following reasons:

- it is a unique stream cipher arrangement with an intrinsic integrity property,
- seemingly, there is a need for hardware-based stream cipher with very high data throughput,
- the ECRYPT NoE call for stream cipher allows research contributions, and
- although we do not claim that the Frogbit cipher is secure, we have no indication to the contrary.

The Frogbit cipher is presented as a research result. While it does meet the formalities required for an ECRYPT NoE stream cipher submission, there is no claim that it can go through the second phase of the stream cipher project without a significant study effort. The submitting organization is currently *not* committed to support the Frogbit development project to a point where the ECRYPT NoE evaluation criteria are comprehensively addressed (e.g. we did no statistical testing of Frogbit ciphertext with an all-zero plaintext). Accordingly, the submitting organization welcomes further independent research work on the Frogbit cipher and its underlying concepts.

2. Contents of the Frogbit Cipher Submission

This submission contains the following major components:

- A documentation package comprising
 - a description of the Frogbit cipher algorithm ([1]),
 - a description of the PRNG (Pseudo-Random Number Generator) design used in the reference Frogbit implementation ([2]),
 - the present document that reconciles the various components and addresses the ECRYPT NoE remaining requirements.
- A reference Frogbit implementation, tailored to the ECRYPT NoE Profile 1A specifications, containing the following original files:
 - ecrypt-sync-ae.h
 - ecrypt-frogbit.c
 - ecrypt-frogbit-prng.c
 - ecrypt-frogbit-prng.h
 - makefile

The following API files are taken unmodified:

ecrypt-sync-ae.c
ecrypt-portable.h
ecrypt-machine.h
ecrypt-config.h

A main program is created if the preprocessor symbol TESTVECTOR_FP is defined in the file ecrypt-frogbit.c; it merely prints the Frogbit test vectors.

- A test vector file:
frogbit-test-vectors.txt
- A self-contained “Frogbit demonstration and challenge program” that may be used as a starting point for (crypt-)analysis of the integrity property built-in the Frogbit cipher:
frogbit.c

3. No Hidden Weaknesses

There are no hidden weaknesses inserted by the designers in the Frogbit cipher design or its ECRYPT NoE Reference Implementation.

4. Intellectual Property Statement

The Frogbit cipher is covered by the United States patent 6,069,954 ([3]). It was also disclosed in a Canadian patent application that was since abandoned. There are no Intellectual Property restriction outside of the United States jurisdiction that we are aware of. The US patent 6,069,954 contains fairly broad claims for the core Frogbit cipher concept, consistent with the novel integrity arrangement.

Since the ECRYPT NoE submission is a research result contribution and since the ECRYPT NoE is not a standardization body, a commitment to offer licenses on a reasonable and non-discriminatory basis would be premature. Parties interested in the Frogbit intellectual property rights in the United States jurisdictions should contact the submitting organization.

5. The Frogbit ECRYPT NoE Reference Implementation

5.1 Overview

The reference implementation is a Frogbit cipher *application*. The Frogbit cipher itself is very generic, being a single bit processing specification. When turning into a Frogbit application, the core specification needs further provisions in many aspects, e.g.

- the ten pseudo-random sources (PRNG) feeding the Frogbit cipher,
- an initialization procedure,
- applicable byte and block processing rules, and

- API specifications for providing the integrity functionality to the application (formalities and semantic aspects).

In the case of the ECRYPT NoE submission, the details of the call for stream cipher primitives shaped the Frogbit application in many aspects. This gave the opportunity to remove speculative text from the document ([1]).

The reference implementation follows the ECRYPT NoE API conventions. We do not repeat them. Here are some technical details:

- The processing order for bits is from least significant to most significant.
- The Frogbit cipher does not support unauthenticated encryption.
- The Frogbit cipher can authenticate associated data (an optional feature in the ECRYPT NoE), without the restriction that this feature can be called only once.
- We met the minimal requirements for parameter sizes, that is a secret key length of 128 bits, an initial value length of either 64 or 128 bits, and a MAC size of either 32, 64, 96, or 128 bits.

5.2 The Frogbit PRNG Design

The Frogbit PRNG design is described in a separate document ([2]). The use of MTGFSR generators for a Frogbit application has been envisioned for some time, and finalized for the present ECRYPT NoE submission. During this design finalization, an overall state size above 512 bits was deemed adequate.

It remains to be seen whether this PRNG arrangement is convenient for hardware implementation. More critically, no statistical testing has been performed on the 10 PRNGs selected in the reference source code.

5.3 The Key Schedule

The present document section describes the key schedule for the Frogbit reference design submitted to the ECRYPT NoE call for stream cipher primitives in April 2005. The key schedule provides minimal support for the Profile 1A, i.e. a key length of 128 bits and an initial value (IV) length of either 64 or 128 bits. When bit or nibble ordering is relevant in this specification, the least significant bit of a byte comes first.

Each of the 10 Frogbit PRNGs has three state words (nominally 26 bits), the first one being $X_{[0]}$ in the notation used to specify the Frogbit PRNGs. For the key schedule specifications, we use the notation $X0_{[0]}$, $X1_{[0]}$ and so on to $X9_{[0]}$ for the ten PRNGs to be initialized.

A stream cipher key is thus created by the following processing sequence:

Bit Interleaving

The key and IV are interleaved, at the individual bit level. That is, one bit (in the case of 128 bits IV) or two bits (in the case of 64 bits IV) are taken from the key, followed by one bit of the IV, starting with the first bit in the respective byte strings.

Nibble Distribution to First State Word in Each PRNG

The interleaved byte string is taken as a nibble string (4 bits each). Four nibbles (in the case of 128 bits IV) or five nibbles (in the case of 64 bits IV) are taken from the interleaved byte string to initialize the state word $X_{i[0]}$, the first nibble in the least significant position in $X_{i[0]}$, and the most significant bits set to zero. This is repeated for $X_{i[1]}$ and so on in sequence for the 10 Frogbit PRNGs.

Other State Words in Each PRNG

For each Frogbit PRNG i , the other two state words are set according to

$$X_{i[1]} = X_{i[0]} \oplus 3333333_8$$

$$X_{i[2]} = X_{i[0]} + 2626262_8$$

The Frogbit PRNGs are thus expeditiously initialized.

Frogbit Encryption of Unused Interleaved Bits

The core Frogbit cipher state is initialized according to the Frogbit specification, and then the interleaved bits that were not used (in the nibble distribution step above) are passed to the single bit encryption function, discarding the ciphertext bits. That represents 32 bit encryption steps (in the case of 128 bits IV) or 56 encryption steps (in the case of 64 bits IV).

The above key schedule specification has been designed quickly to adapt the core Frogbit algorithm to the ECRYPT NoE stream cipher formal submission requirements. It is definitely weak. We believe that the research directions on the Frogbit stream cipher arrangement should focus on its integrity properties, i.e. finding collisions, so that the key schedule specifications is not so important at this stage.

5.4 The MAC Computation

The MAC value computation proceeds as follows: at the end of the message processing (encryption or decryption), the first 32 bits from the secret key are encrypted, and the ciphertext bits are discarded. Then, the MAC computation proceeds with the encryption of constant zero bits for the requested MAC length, this time preserving the ciphertext bits as the final MAC value. This creates a binary MAC value of the required length, based on the final Frogbit state and some secret key bits.

5.5 Test Vectors

The test vector is based on authenticating "FROGBIT" and encrypting "HYDROCHARIS MORSUS-RANAE" (the botanical frogbit) using a secret key based on the initial fractional part of pi (i.e. $\pi=3.1415926535\dots_{10}=3.243F6A8885A308\dots_{16}$). The test vector initial values are made of further hexadecimal digits of pi. Four tests are provided, with two different initial value lengths and two MAC lengths.

Frogbit cipher test vectors, version 1.0

Secret key: 243F6A8885A308D313198A2E03707344

Authenticated text: "FROGBIT"

Authenticated text: 46524F47424954

Plaintext: "HYDROCHARIS MORSUS-RANAE"

Plaintext: 485944524F434841524953204D4F525355532D52414E4145

Test 1, Authenticate-encrypt test, key length 128, IV length 64, MAC length 32.

Initial value: A4093822299F31D0

Ciphertext: 20A0A38B05D62EBDE7E28E8F17F5EB78882A5B28B79B9088

MAC value: B4B4CC9F

Decrypt-verify done.

Test 2, Authenticate-encrypt test, key length 128, IV length 128, MAC length 32.

Initial value: A4093822299F31D0082EFA98EC4E6C89

Ciphertext: 2F62E90DFBEE03D38DF6788FE61687E4A2E1FF6DB71DFAF1

MAC value: CBB4503A

Decrypt-verify done.

Test 3, Authenticate-encrypt test, key length 128, IV length 64, MAC length 128.

Initial value: 452821E638D01377

Ciphertext: 2BE4E7ED84EC97AC99231C96181204008F6AD8FB1EBB8AEE

MAC value: 57DB85BFA9C88267CACDDEE3B23483A0

Decrypt-verify done.

Test 4, Authenticate-encrypt test, key length 128, IV length 128, MAC length 128.

Initial value: 452821E638D01377BE5466CF34E90C6C

Ciphertext: 85AFF3FFA16F17CA9FDF0468FAF3EF28283CD7AEFCE97F76

MAC value: 1DE36E6731ED764BE9A480B3C682D1F7

Decrypt-verify done.

The test vector file contains a summary of test vectors, followed by a detailed listing of Frogbit state variables during the processing of the same test values. The reported Frogbit state does not contain the PRNG indexes, i.e. the cumulative number of bit pairs extracted from each of the 10 Frogbit PRNGs. This detailed listing looks like:

```
A100 (10) 00- [9728654130] 6
A000 (00) 010 [9728654130] 6
C000 (00) 00- [1659038472] 4
C010 (11) 10- [7165894023] 6
C010 (11) 111 [7165894023] 6
C110 (01) 10- [5379462810] 3
```

The listing columns show the current data, and the Frogbit state following the processing of this data. The current data columns are

- a letter indicating the type of bit processing:
 - I for initial value processing (part of Frogbit key schedule),
 - A encryption for authenticated data processing,
 - C encryption,
 - D decryption,
 - Z encryption of secret key bits in preparation of MAC value computation, and
 - M encryption for MAC value computation;
- three bits, respectively a plaintext bit, an inner sequence bit, and a ciphertext bit;
- two bits between parentheses, respectively first and second keystream bits (**k1**, **k2**).

The Frogbit state after processing this data consists of the following columns:

- a binary digit for the run length processing;
- a binary digit for the bounded run count;
- the accumulated binary representation from the **k2** keystream, being either empty (“-”) or a single bit;
- ten digits between brackets ([]), the current table of permuted indexes;
- a single digit, the current key stream number.

See the reference [1], end of section 4, for the precise meaning of these report column descriptions.

6. The Frogbit Challenge Program

A Frogbit challenge program is provided. It is a self-contained source code file. It is intended to facilitate cryptanalysis attempts at the core Frogbit algorithm.

This challenge would be fully met if someone shows

- 1) a starting Frogbit state file,
 - 2) two non-empty binary files, identical in size but differing in contents,
 - 3) optionally, a specification for the "-o[1-7]" command line option to the challenge program (specifying a number of bits to be ignored at the end of both binary files),
- that produce the same final Frogbit state file when submitted to the challenge program.

Other cryptanalysis attempts may target a modified versions of the challenge program. Notably, the Frogbit PRNG design may be modified, or even the nominally pseudo-random sequences may be replaced by arbitrary constant sequences chosen by the cryptanalysis. In theory, for any limited length binary sequence one can find a PRNG design that would produce the binary sequence in a portion of its output.

7. References

- [1] Thierry Moreau, *The Frogbit cipher, a data integrity algorithm*, CONNOTECH Experts-conseils Inc., January 1997, Updated, i.e. shortened, April 2005
- [2] Thierry Moreau, *The "Multiplexed and Twisted" GFSR, a Flexible Scheme for the Creation of Pseudo-random Generators*, CONNOTECH Experts-conseils Inc., April 2000, Revised April 2005
- [3] US patent document 6,069,954, Moreau, Thierry, *Cryptographic Data Integrity Apparatus and Method Based on Pseudo-Random Bit Generators*, May 30, 2000 (application number 08/853,455, May 9, 1997, priority based on Canadian patent application number 2,177,622, filed on May 29, 1996).