ECRYPT II

ICT-2007-216676

ECRYPT II

European Network of Excellence in Cryptology II

Network of Excellence

Information and Communication Technologies

D.SYM.10
The eSTREAM Portfolio in 2012

Due date of deliverable: 30 November 2011
Actual submission date: 16 January 2012

Start date of project: 1 August 2008 Duration: 4 years

Lead contractor: Royal Holloway (RHUL)

Version 1.0

Project co-funded by the European Commission within the 7th Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission services)	
RE	Restricted to a group specified by the consortium (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	

The eSTREAM Portfolio in 2012

Editors
Carlos Cid (RHUL) and Matt Robshaw (FTRD)

Contributors
Steve Babbage (Vodafone), Julia Borghoff (DTU), and
Vesselin Velichkov (KUL)

16 January 2012
Version 1.0

The work described in this report has in part been supported by the Commission of the European Communities through the ICT program under contract ICT-2007-216676.

Contents

1 The eSTREAM Portfolio in 2012 **1**

2 Profile 1 Algorithms **3**
2.1 HC-128 3
2.2 Rabbit 3
2.3 Salsa20/12 4
2.4 SOSEMANUK 5

3 Profile 2 Algorithms **6**
3.1 Grain v1 6
3.2 MICKEY 2.0 7
3.3 Trivium 7

Chapter 1

The eSTREAM Portfolio in 2012

eSTREAM, a multi-year project co-ordinated by ECRYPT, came to an end in April 2008. Dedicated to promoting the design of new stream ciphers, the project finished with the publication of a portfolio of new stream ciphers [7]. Four of the proposals in the final portfolio were suited to fast encryption in software (so-called Profile 1) while four offered particularly efficient implementation—in terms of physical resources—in hardware (so-called Profile 2).

The portfolio was revised in September 2008 [8], after the announcement of cryptanalytic results against one of the algorithms [30], and since then has been revisited periodically as the algorithms have matured. The current 2012 eSTREAM portfolio contains seven algorithms, presented in Table 1.1 in alphabetical order. Some of these algorithms also support some additional popular key lengths; these are presented in Table 1.2.

When the portfolio was published [7, 8], it was the opinion of the eSTREAM committee that all the algorithms in the portfolio would still benefit from some additional analysis. Several years have passed since then and, at some stage, the weight of expert opinion will shift from one of caution to one of acceptance. It is not clear that we have reached that point just yet, but certainly we cannot be far away. Indeed, while it might still be too early for most developers to consider deploying an eSTREAM cipher on a large scale, some early adopters might be interested in taking the plunge for some niche applications.

In this report we briefly survey the algorithms in the portfolio. Further, we highlight some cryptanalytic results that might apply to these portfolio ciphers, along with any results signaling structural features or impacting their use and deployment. The net result is that the eSTREAM portfolio remains unchanged, and while portfolio algorithms will benefit from more analysis over coming years, their design should still be considered sound.

eSTREAM continues to be maintained and any changes to the portfolio will be announced on `http://www.ecrypt.eu.org/stream/`.

Table 1.1: The eSTREAM portfolio in 2012.

Profile 1	*Profile 2*
128-bit key	*80-bit key*
HC-128 Rabbit Salsa20/12 Sosemanuk	Grain v1 MICKEY 2.0 Trivium -

Table 1.2: Versions of the eSTREAM portfolio ciphers that support extended key lengths. Note that the 128-bit version of Grain v1 is no longer supported by the designers and has been replaced by Grain-128a, see Sect. 3.1. We do not consider Grain-128a to be part of the eSTREAM portfolio, but we draw attention to this algorithm and encourage independent cryptanalysis.

Profile 1	*Profile 2*
256-bit key	*128-bit key*
HC-256 - Salsa20/12 -	- MICKEY-128 2.0 - -

Chapter 2

Profile 1 Algorithms

Algorithms in this profile were intended to give excellent throughput when implemented in software. These algorithms were required to support 128-bit keys though some, optionally, support a 256-bit key. The initialisation vector (IV) was required to be 64 or 128 bits.

2.1 HC-128

The HC-128 algorithm was designed by Hongjun Wu. The cipher makes use of a 128-bit key and 128-bit initialisation vector; its secret state consists of two tables, each with 512 32-bit elements. At each step, one element of one of the tables is updated using a non-linear feedback function, while one 32-bit output is generated from the non-linear output filtering function. The cipher specification states that 2^{64} keystream bits can be generated from each key/IV pair. More details can be found via the eSTREAM web page [22] or the eSTREAM book [41]. HC-256, a companion version to HC-128, accommodates 256-bit keys.

Both HC-128 and HC-256 stream cipher offer a very impressive performance in software applications where one wishes to encrypt large streams of data. However, the impressive speed of HC-128 results from the fact it is a table-driven algorithm. The typical downside to such a design approach is the required time to initialise the cipher. As a result, for applications that encrypt small amounts of data but require frequent re-initialisation, there can be a performance penalty.

Despite the high profile of the algorithm there have been no significant cryptanalytic advances against HC-128. Further, there seems to be no indication that the security margin is in any way tight. The designer does not claim any intellectual property on HC-128 and the cipher is included in the latest release version of CyaSSL, a lightweight, open source embedded implementation of the SSL/TLS protocol [46].

2.2 Rabbit

Rabbit uses a 128-bit key and a 64-bit initialisation vector. A set of eight 32-bit state registers and eight 32-bit counters are used to provide an efficient encryption mechanism based on simple arithmetic and other basic operations such as rotation. More details can be found via the eSTREAM web page [22] or the eSTREAM book [41].

Testing during the eSTREAM process confirmed that the cipher was among the most efficient of the stream ciphers submitted, and this efficiency was visible on a wide range of

platforms. Like many stream ciphers there is some marginal cost incurred during initialisation. But this has a minor impact on the total cost of encryption even when encrypting short message streams.

A theoretical distinguisher for Rabbit was described by Aumasson [3] and later improved [35]. However the complexity of the distinguisher is greater than the complexity of exhaustive key search. There are no other cryptanalytic results on Rabbit though, like all other cryptographic algorithms, care needs to be taken with its implementation. In this regard work on the side-channel cryptanalysis of Rabbit can be found in [33, 12].

While Rabbit was patented by the designers, the designers have stated that the algorithm is nevertheless free for use. Rabbit is described in Internet draft RFC 4503 [14] and it is included in ISO/IEC 18033-4 [31]. The cipher is also included in the latest release version of CyaSSL, a lightweight, open source embedded implementation of the SSL/TLS protocol [46].

2.3 Salsa20/12

Salsa20/r is a software-oriented stream cipher by Bernstein. During the operation of the cipher the key, a 64-bit nonce (unique message number), a 64-bit counter, and four 32-bit constants are used to construct the 512-bit initial state of the cipher. After r iterations of the Salsa20/r round function, the updated state is used as a 512-bit output. Each such output block is an independent combination of the key, nonce, and counter and, since there is no chaining between blocks, the operation of Salsa20/r resembles the operation of a block cipher in counter mode. Salsa20/r therefore shares the very same implementation advantages, in particular the ability to generate output blocks in any order and in parallel.

The round transformation of Salsa20 uses a combination of three simple operations: addition modulo 2^{32}, bit rotation and bitwise exclusive-or (what has since become known as an ARX construction). The efficient implementation of these operations gives the good software performance of the cipher. More details can be found via the eSTREAM web page [22] or the eSTREAM book [41].

There are three variants of Salsa20 depending on the number of rounds. Each of them provides a different security vs. performance trade-off. Salsa20/20 has 20 rounds and is recommended by the designer for "encryption in typical cryptographic applications". The versions Salsa20/12 and Salsa20/8 have 12 and 8 rounds respectively and the designer recommends them for "users who value speed more highly than confidence" [11]. The eSTREAM committee suggested that Salsa20/12 would perhaps offer the most appealing trade-off.

During the years following its publication, Salsa20 has undergone significant cryptographic analysis. Although several attacks have been found on reduced versions of the cipher, there is no attack better than exhaustive key search on either Salsa20/12 or Salsa20/20. Crowley presented a key-recovery attack on Salsa20/5 [19] while Fischer *et al.* [28] observe non-randomness after four rounds of Salsa20, using it to construct a key-recovery attack on Salsa20/6. The authors also report a related-key attack on Salsa20/7. An attack on Salsa20/7 without related keys appears in [44] while an improvement [4] gives the first key-recovery attack on Salsa20/8. The designer makes no intellectual property claim on Salsa20.

2.4 SOSEMANUK

SOSEMANUK has a variable key length, ranging from 128 to 256 bits, and takes an initial value of 128 bits. However, for any key length the cipher is only claimed to offer 128-bit security. SOSEMANUK uses similar design principles to the stream cipher SNOW 2.0 [31] and the block cipher SERPENT [13]. SOSEMANUK aims to fix some potential structural weaknesses in SNOW 2.0 while providing better performance by decreasing the size of the internal state.

As for SNOW 2.0, SOSEMANUK has two main components: a linear feedback shift register (LFSR) and a finite state machine (FSM). The LFSR operates on 32-bit words and has length 10. At every clock a new 32-bit word is computed. The FSM has two 32-bit memory registers. At each step the FSM takes as input some words from the LFSR, updates the memory registers, and produces a 32-bit output. On every four consecutive output words from the FSM an output transformation, based on SERPENT, is applied. The resulting four 32-bit output words are exclusive-ored with four outputs from the LFSR to produce four 32-bit words of keystream. More details can be found via the eSTREAM web page [22] or the eSTREAM book [41].

SOSEMANUK has received a great deal of cryptographic analysis and several attacks have been published in the literature. However none breaks the claimed 128-bit security of the cipher. A guess-and-determine attack presented at SASC 2006 recovers all 384 bits of the internal state of the cipher after initialisation [43]. It has time complexity 2^{224} and requires 24 words of key stream. This type of attack was improved [25] to give a time complexity of 2^{176}. An attack using the linear masking method [34] is able to recover the internal state with time, memory and data complexity less than 2^{148}. This was reduced by a factor of 2^{10} to $2^{135.7}$ [18].

The designers state that SOSEMANUK is free to use for any purpose.

Chapter 3

Profile 2 Algorithms

Algorithms in this profile were intended to be efficient—in terms of the physical resources required—when implemented in hardware. These algorithms were required to support 80-bit keys though some, optionally, also support a 128-bit key. Support for an initialisation vector (IV) of 32 or 64 bits was also required.

3.1 Grain v1

Grain is best described as a family of hardware-efficient synchronous stream ciphers. The initial version used an 80-bit key and a 64-bit initialisation vector but analysis in the early stages of the eSTREAM effort compromised its security [9]. The revised specifications, Grain v1, described two stream ciphers which supported 80-bit keys (with 64-bit initialisation vector) and 128-bit keys (with 80-bit initialisation vector). Elegant and simple, Grain v1 has been an attractive choice for cryptanalysts and implementors alike with two shift registers, one with linear feedback and the second with non-linear feedback, being the essential feature of the algorithm family. These registers, and the bits that are output, are coupled by means of very lightweight, but judiciously-chosen boolean functions. More details can be found via the eSTREAM web page [22] or the eSTREAM book [41].

For the version of the cipher that takes 80-bit keys, the specifications given by Grain v1 are the currently recommended version. However, cryptanalysis [21] of the 128-bit version of Grain v1 has lead to the proposal of a new cipher called Grain-128a [1]. This variant also specifies some additional registers to enable the calculation of a *message authentication code* in addition to generating a keystream. While Grain-128a retains the elegance of earlier versions of the cipher, in its fastest implementation it now occupies more space and runs at half the speed of Grain v1. However, the design of the Grain family allows for an ingenious multiplication of throughput speed, though at the cost of a minor increase in the space consumed.

Like many stream ciphers, there is some cost incurred during initialisation and the impact of this will depend on the intended application and the likely size of the messages being encrypted. The designers make no intellectual property claim on the Grain family. There is anecdotal evidence that some early adopters have been considering versions of the Grain family for deployment.

3.2 MICKEY 2.0

The name MICKEY is derived from "Mutual Irregular Clocking KEYstream generator" which describes the essential behavior of the cipher. The state consists of two 100-bit shift registers, one linear and one non-linear, each of which is irregularly clocked under control of the other. The specific clocking mechanisms contribute to the cipher's cryptographic strength while allowing guarantees on period and pseudorandomness. The cipher specification states that each key can be used with up to 2^{40} different IVs of the same length, and that 2^{40} keystream bits can be generated from each key/IV pair. The designers have also specified a scaled-up version of the cipher called MICKEY-128 2.0, which takes a 128-bit key and an initialisation vector of up to 128 bits. More details can be found via the eSTREAM web page [22] or the eSTREAM book [41].

MICKEY 2.0 can be implemented with a particularly small hardware footprint, making it a good candidate where low gate count or low power are the primary requirements. However the irregular clocking mechanism means that it cannot be readily parallelised.

It has been noted, *e.g.* Gierlichs *et al* [29], that straightforward implementations of the MICKEY ciphers are likely to be susceptible to timing or power analysis attacks. Otherwise there have been no known cryptanalytic advances against MICKEY 2.0 or MICKEY-128 2.0.

The designers state that MICKEY 2.0 and MICKEY-128 2.0 are free for any use.

3.3 Trivium

Trivium is a synchronous stream cipher that is designed to generate up to 2^{64} bits of keystream from an 80-bit key and 80-bit IV. Like many synchronous stream ciphers, the algorithm requires initialisation and the internal state of the keystream generator is initialized using the secret key and the IV. Initialisation is very similar to keystream generation and requires 1152 steps of the clocking procedure of Trivium. Then keystream is generated by repeatedly clocking the cipher, where in each clocking cycle one bit of keystream is produced and output. The design is such that this basic throughput can be improved by several multiples without an undue increase to the area required for an implementation. Details can be found via the eSTREAM web page [22] or the eSTREAM book [41].

The elegant and simple structure of Trivium has attracted the attention of many cryptanalysts; however there is no attack faster than exhaustive key search. The authors [17] describe Trivium as being " [...] designed as an exercise in exploring how far a stream cipher can be simplified without sacrificing its security, speed or flexibility."

In [17] it is claimed that the probability of a given key and IV pair yielding a keystream with a period smaller than 2^{80} is 2^{-208}. It is expected that the period would be at least 2^{128} bits. Furthermore, it is stated that each state bit depends on each key and IV bit in a non-linear way after 576 iterations of the cipher. As initialisation consists of 1152 iterations it is believed that resynchronization attacks are not possible. There have been several guess-and-determine attacks on Trivium. The designers estimated the complexity of a naive guess-and-determine attack to be 2^{195} operations while a reformulation of Trivium [32] might be used to give a more intelligent attack with complexity 2^{135}. Maximov and Biryukov [37] describe an attack with complexity around $2^{96.5}$ operations.

Other proposed attacks are based on the algebraic description of Trivium. Raddum showed that the internal state of Trivium can be described as a sparse, fully-determined equation

system [40]. After observing 288 keystream bits, the system consists of 288 linear and 666 quadratic Boolean equations in 954 variables where each equation contains precisely six variables. Different techniques have been considered to solve this equation system [40, 38, 15], though without success.

Smaller versions of the cipher exist: for instance Bivium [40] and variants of Trivium with a round-reduced key setup. For both there have been successful attacks, however none of the attacks can be extended to the full cipher. Chosen IV-attacks have been proposed on variants of Trivium with a round-reduced key setup. A chosen IV statistical analysis [27] can recover a few key bits of Trivium when only 672 out of 1152 initialisation rounds are used. Cube attacks [20] were able to recover the secret key when the attacker can obtain several keystreams under chosen IVs when only 767 initialisation rounds are used. This attack is similar to the AIDA attack [45]. Furthermore, for 736 initialisation rounds statistical weaknesses in the keystream have been observed [24].

The designers make no intellectual property claim on Trivium.

Bibliography

[1] M. Ågren, M. Hell, T. Johansson, and W. Meier. A New Version of Grain-128 with Authentication. Presented at SKEW 2011, available via `skew2011.mat.dtu.dk/proceedings/`.

[2] M. Afzal and A. Masood. Resistance of stream ciphers to algebraic recovery of internal secret states. *Third International Conference on Convergence and Hybrid Information Technology*, ICCIT, vol. 2, pages 625–630, 2008.

[3] J.-P. Aumasson. On a bias of Rabbit. Proceedings of SASC 2007, available via `http://www.ecrypt.eu.org/stvl/sasc2007/`.

[4] J-P. Aumasson, S. Fischer, S. Khazaei, W. Meier and C. Rechberger. New Features of Latin Dances: Analysis of Salsa, ChaCha, and Rumba. In K. Nyberg, editor, *Proceedings of FSE 2008*, LNCS 5086, pp. 470–488, Springer 2008.

[5] J-P. Aumasson, I. Dinur, W. Meier and A. Shamir. Cube testers and key recovery attacks on reduced-round MD6 and Trivium. In O. Dunkelman, editor, *Proceedings of FSE 2009*, LNCS 5665, pp. 1–22, Springer, 2009.

[6] J-P. Aumasson, I. Dinur, L. Henzen, W. Meier and A. Shamir. Efficient FPGA Implementations of High-Dimensional Cube Testers on the Stream Cipher Grain-128. Cryptology ePrint Archive, Report 2009/218. Available via `http://eprint.iacr.org/2009/218`.

[7] S. Babbage, C. De Cannière, A. Canteaut, C. Cid, H. Gilbert, T. Johansson, M. Parker, B. Preneel, V. Rijmen, and M.J.B. Robshaw. The eSTREAM Portfolio. April 2008. Available via `http://www.ecrypt.eu.org/stream/`.

[8] S. Babbage, C. De Cannière, A. Canteaut, C. Cid, H. Gilbert, T. Johansson, M. Parker, B. Preneel, V. Rijmen, and M.J.B. Robshaw. The eSTREAM Portfolio (rev. 1). September 2008. Available via `http://www.ecrypt.eu.org/stream/`.

[9] C. Berbain, H. Gilbert, and A. Maximov. Cryptanalysis of Grain. In M. Robshaw, editor, *Proceedings of FSE '06*, volume 4047 of LNCS, pages 15–29. Springer-Verlag, 2006.

[10] D.J. Bernstein. Notes on the ECRYPT Stream Cipher project (eSTREAM). `http://cr.yp.to/streamciphers.html`.

[11] D.J. Bernstein. Salsa20 page. `http://cr.yp.to/snuffle.html`.

[12] A. Berzati, C. Canovas-Dumas, and L. Goubin. Fault analysis of Rabbit: Toward a secret key leakage. In B. Roy and N. Sendrier, editors, *Proceedings of Indocrypt 2009*, LNCS 5922, pp. 72–87, Springer 2009.

[13] E. Biham, L.R. Knudsen, and R.J. Anderson. Serpent: A New Block Cipher Proposal. In S. Vaudenay, editors, *Proceedings of FSE 1998*, LNCS, volume 1372, pp. 222–238, Springer Verlag.

[14] M. Boesgaard, M. Vesterager, and E. Zenner. A Description of the Rabbit Stream Cipher Algorithm. Network Working Group, Request for Comments 4503. Available via `http://tools.ietf.org/html/rfc4503`.

[15] J. Borghoff, L. Knudsen, and K. Matusiewicz. Hill climbing algorithms and Trivium. In A. Biryukov, G. Gong, D. Stinson, editors, *Proceedings of SAC 2010*, LNCS, volume 6544, pp. 57–73, Springer Verlag.

[16] J. Borghoff, L. Knudsen, and M. Stolpe. Bivium as a mixed-integer linear programming problem. In M. Parker, editor, *Proceedings of Cryptography and Coding 2009*, LNCS 5921 pp. 133–152, Springer 2009.

[17] C. De Cannière and B. Preneel. Trivium. M. Robshaw and O. Billet, editors. *New Stream Cipher Designs: The eSTREAM Finalists*. LNCS 4986, pp. 244–266. Springer 2008.

[18] J. Cho and M. Hermelin. Improved linear cryptanalysis of SOSEMANUK. In D. Lee and S. Hong, editors, *Proceedings of ICISC '09*, LNCS 5984, pp. 101–106. Springer-Verlag, 2009.

[19] P. Crowley. Truncated differential cryptanalysis of five rounds of Salsa20. In Proceedings of SASC 2006, available via `http://www.ecrypt.eu.org/stvl/sasc2006/`. Also available via `http://eprint.iacr.org/2005/375`.

[20] I. Dinur A. Shamir. Cube attacks on tweakable black box polynomials. In A. Joux, editor, *Proceedings of Eurocrypt 2009*, LNCS 5479 pp. 278–299, Springer 2009.

[21] I. Dinur and A. Shamir. Breaking Grain-128 with Dynamic Cube Attacks. In A. Joux, editor, *Proceedings of FSE 2011*, LNCS 6733, pp. 167–187, Springer, 2011.

[22] ECRYPT Network of Excellence. The eSTREAM project, available via `http://www.ecrypt.eu.org/stream/`.

[23] T. Eibach, E. Pilz, and G. Völkel. Attacking Bivium using SAT solvers. In H. Kleine Büning and X. Zhao, editors, *Theory and applications of satisfiability testing - SAT '08*, LNCS 4996, pp. 63–76, Springer 2008.

[24] H. Englund, T. Johansson, and M. Turan. A framework for chosen IV statistical analysis of stream ciphers. In K. Srinathan, C. Pandu Rangan, and M. Yung, editors, *Proceedings of Indocrypt 2007*, LNCS 4859, pp. 268–281, Springer 2007.

[25] X. Feng, J. Liu, Z. Zhou, C. Wu, and D. Feng. A byte-based guess and determine attack on SOSEMANUK. In M. Abe, editor, *Proceedings of Asiacrypt '10*, LNCS 6477, pages 146–157. Springer-Verlag, 2010.

[26] S. Fischer, S. Khazaei and W. Meier. Chosen IV Statistical Analysis for Key Recovery Attacks on Stream Ciphers. In S. Vaudenay, editor, *Proceedings of Africacrypt 2008*, LNCS 5023, pp. 236–245, Springer 2008.

[27] S. Fischer, S. Khazaei, and W. Meier. Chosen IV statistical analysis for key recovery attacks on stream ciphers. In *Proceedings of Africacrypt 2008*, LNCS 5023, pp. 236–245, Springer 2008.

[28] S. Fischer, W. Meier, C. Berbain, J.-F. Biasse, and M. Robshaw. Non-randomness in eSTREAM Candidates Salsa20 and TSC-4. In R. Barua and T. Lange, editors, *Proceedings of Indocrypt 2006*, LNCS 4329, pp. 2–16, Springer 2006.

[29] B. Gierlichs, L. Batina, C. Clavier, T. Eisenbarth, A. Gouget, H. Handschuh, T. Kasper, K. Lemke-Rust, S. Mangard, A. Moradi, and E. Oswald. Susceptibility of eSTREAM Candidates towards Side Channel Analysis. In Proceedings of SASC 2008, available via `http://www.ecrypt.eu.org/stvl/sasc2008/`.

[30] M. Hell and T. Johansson. Breaking the F-FCSR-H stream cipher in Real Time. In J. Pieprzyk, editor, *Proceedings of Asiacrypt 2008*, LNCS 5350, pp. 557–569, Springer 2008.

[31] ISO/IEC 18033-4. Information technology – Security techniques – Encryption algorithms – Part 4: Stream ciphers. 2005. Available via `www.iso.org`.

[32] S. Khazaei Re: A reformulation of Trivium. Posted on the eSTREAM Forum, 2006. Available via `http://www.ecrypt.eu.org/stream/phorum/read.php?1,448`.

[33] A. Kircanski and A. Youssef. Differential fault analysis of Rabbit. In M. Jacobson Jr., V. Rijmen, and R. Safavi-Naini, editors, *Proceedings of SAC 2009*, LNCS 5867, pp. 197–214, Springer 2009.

[34] J-K. Lee, D.H. Lee and S. Park. Cryptanalysis of Sosemanuk and SNOW 2.0 Using Linear Masks. In J. Pieprzyk, editor, *Proceedings of Asiacrypt 2008*, LNCS 5350, pp. 524–538, Springer 2008.

[35] Y. Lu, H. Wang and S. Ling. Cryptanalysis of Rabbit. In T.-C. Wu, C.-L. Lei, V. Rijmen, and D.-T. Lee, editors, *Proceedings of ISC 2008*. LNCS 5222, pp. 204–214, Springer 2008.

[36] S. Maitra, G. Paul and S. Raizada. Some Observations on HC-128. *Proceedings of the International Workshop on Coding and Cryptography (WCC)*, May 10-15, 2009, Ullensvang, Norway, pp. 527–539.

[37] A. Maximov and A. Biryukov. Two trivial attacks on Trivium. In C. Adams and M. Wiener, editors, *Proceedings of SAC 2007*, LNCS 4876 pp. 36–55, Springer 2007.

[38] C. McDonald, C. Charnes, and J. Pieprzyk. An algebraic analysis of Trivium ciphers based on the boolean satisfiability problem. Cryptology ePrint Archive, Report 2007/129, 2007. Available via `http://eprint.iacr.org/2007/129`.

[39] D. Priemuth-Schmid and A. Biryukov. Slid Pairs in Salsa20 and Trivium. In D.R. Chowdhury, V. Rijmen, and A. Das, editors, *Proceedings of Indocrypt 2008*, LNCS 5365, pp. 1–14, Springer 2008.

[40] H. Raddum. Cryptanalytic results on Trivium. eSTREAM report 2006/039, 2006. Available via `http://www.ecrypt.eu.org/stream/triviump3.html`.

[41] M. Robshaw and O. Billet, editors. *New Stream Cipher Designs: The eSTREAM Finalists*. LNCS 4986, pp. 267–293. Springer 2008.

[42] A. Röck. Stream Ciphers Using a Random Update Function: Study of the Entropy of the Inner State. In S. Vaudenay, editor, *Proceedings of Africacrypt 2008*, LNCS 5023, pp. 258–275, Springer 2008.

[43] Y. Tsunoo, T. Saito, M. Shigeri, T. Suzaki, H. Ahmadi, T. Eghlidos, and S. Khazaei. Guess-and-Determine Attacks against SOSEMANUK Stream Cipher. In Proceedings of SASC 2006, available via `http://www.ecrypt.eu.org/stvl/sasc2006/`.

[44] Y. Tsunoo, T. Saito, H. Kubo, T. Suzaki, and H. Nakashima. Differential cryptanalysis of Salsa20/8. In Proceedings of SASC 2007, available via `http://www.ecrypt.eu.org/stvl/sasc2007/`.

[45] M. Vielhaber. Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack. Cryptology ePrint Archive, Report 2007/413. Available via `http://eprint.iacr.org/2007/413`.

[46] yaSSL. Yet Another SSL. Available via `www.yassl.com`.