SVH: A Lightweight Stream Cipher Based on Dual Pseudo-Random Transformation and OFB

Xuejun Dai[1,a], Yuhua Huang[1], Lu Chen[1], Tingting Lu[2], Sheng Zhao[3]

[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

[2]College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

[3] College of Energy and Power Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

[a]email: daixuemai@163.com

Keywords: Stream Cipher; Lightweight Cryptography; RFID

Abstract. A new lightweight stream cipher, SVH, is proposed. The design targets hardware environments where gate count, power consumption and memory is very limited. It is based on dual pseudo-random transformation and output feedback. The block of key size is 64 bits and SVH can achieve sufficient security margin against known attacks, such as linear cryptanalysis, differential cryptanalysis, impossible differential cryptanalysis. Hardware implementation of SVH is around 1171GE, which is comparable with the 1458 GE hardware implementation of Grain. The software implementation of SVH on 8-bit microcontroller is about 19.55Mb/s, and its efficiency is 30 times as much as that of Grain in RFID environment. The hardware complexity and throughput compares favourably to other hardware oriented stream ciphers like Grain.

Introduction

In recent years, a number of security and high performance stream cipher have been designed to promote the development of cryptography. However, with the development of wireless network technology, ordinary stream cipher is difficult to meet the mobile terminal resource-constrained, lightweight cryptographic algorithms required to meet hardware and software, computing power and energy consumption and other resource-constrained needs of the terminal. Grain [1] and WG-7 [2] as an outstanding representative of the lightweight stream cipher, provides us with a good opportunity based on the most advanced technologies designed for mobile terminals with limited resources. The stream cipher refers to a mathematical manipulation which the plain text is coded using the binary digital sequence $x_1, x_2, \ldots$. Each encryption is encrypted with the same length as the key stream, which is realized by the same key stream generated by the decryption, KS is the stream generator and KS_0 is the initial key. The blocks are then converted into the equal length digital sequences $y = (y_1, y_2, \ldots, y_n)$using the key of $k = (k_1, k_2, \ldots, k_t)$. This process can be represented with the model in Fig.1.

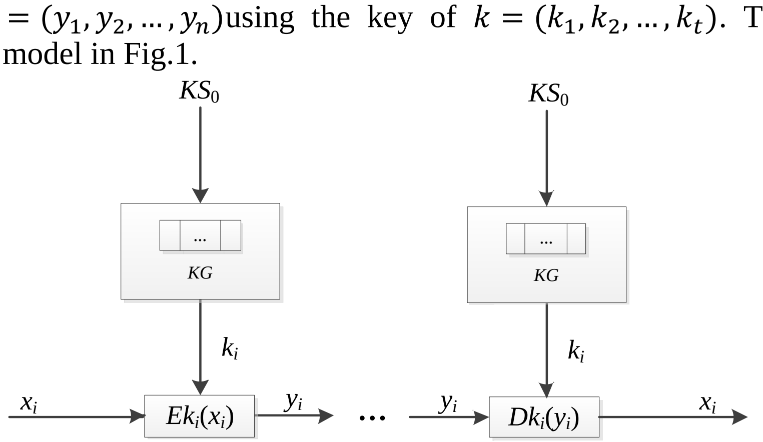

Fig.1. The mathematical model of stream cipher

In this paper, we propose a new lightweight stream cipher SVH, namely Stream Vertical Horizontal .It uses block cipher VH as the key stream generator, and it is used to encrypt and decrypt the OFB operating mode. The initial vector length of SVH is 64 bits and the block of key size is 64 bits. VH expressly refers to the 64 bits plaintext arranged in 8 bytes * 8 bits of matrix, pseudo random transformation for each row and column of the matrix, supporting key lengths of 64, 80, 96, 112 and 128 bits. One novel design methods of VH is the use of SP structure, with a pseudo-random transformation to construct 256-byte encryption transformation table and decryption transformation table, simplified *S*-box design, which fully reflects the realization of lightweight block cipher to achieve small space occupied. In addition, VH after *P* permutation, have been purged diagonal pseudo-random data transformation. SVH on currently known attack methods to achieve adequate immunity and exhibits efficiency in hardware and software on consumption. SVH's hardware implementation requires 1171 GE. The software efficiency of SVH is computed to be 19.55Mb/s. However, the hardware implementation of Grain reaches 1458 GE and the software efficiency is computed to be 0.61Mb/s. We believe that the software and hardware efficiency of SVH is higher than the stream cipher Grain. It can be considered that the SVH has achieved a superior balance in security and performance to meet the hardware usage needs of extreme environments with limited resources, such as RFID and to meet some software implementation needs from environment, such as embedded mode and MCU.

Design of the SVH

In this paper, the lightweight stream cipher SVH is based on the OFB mode of the block cipher, and the key generator is constructed by VH.

The Key Generator VH.

VH adopts the SP structure [3], has a block length of 64 bits, and supports the keys of 64, 80, 96, 112 and 128 bits. The number of rounds is r = 10, 11, 12, 13 and 14, respectively. VH has three parameters: 64-bit plain text P, key K and the 64-bit cipher text C. Let $C = E_K(P)$ denote the encryption transformation and $P = D_K(C)$ denote the decryption transformation. The encryption and decryption steps of VH are as follows:

(1) Encryption Transformation Table And Decryption Transformation Table

The encryption and decryption *S*-box is generated via pseudorandom transformation. We first compute $T(i) = \lfloor|256\sin(i)|\rfloor$, where $\lfloor\rfloor$denote the round-down operation; to generate 256 non-repetitious bytes, i ranges from 1 to 30000, eliminating repetitions that occur. The encryption transformation table $S[256]$ and decryption transformation table $S^{-1}[256]$ are a pseudorandom combination of 256 bytes, which is obtained by alternating bytes in *T*: $S[T(j)] = T(j+1)$, $S[T(255)] = T(0)$; $S^{-1}[T(j+1)] = T(j)$, $S^{-1}[T(0)] = T(255)$; where 0≤j≤254.

(2) Key Schedule

Key scheduling is achieved via iterations, so the L-byte key K is expanded to 8(r+1) bytes: the expanded key $\mathrm{Key} = K_0|K_1| \dots |K_i| \dots |K_r = k_0|k_1| \dots |k_i| \dots |k_{8r+7}$, where each K_i has 8 bytes, 0≤i ≤r, each k_i has a unit byte, 0≤i ≤8r+7. For the key K with 8, 10, 12, 14 and 16 bytes, the number of rounds is r =10, 11, 12, 13, and 14. The first L bytes of the expanded key is the key K :$K = k_0|k_1| \dots |k_{L-1}$. In the case of L≤i≤8r+7, k_i in the expanded key is obtained via recursion of k_{i-L} and k_{i-1}, i.e. $k_i = S[k_{i-1}] \oplus k_{i-L}$.

(3) Data Encryption Process

The encryption process of VH is shown in Figure 2. At the first, we get the results of the initial key K_0 and plaintext by XOR. Then, through further r-round to encrypt the result of the previous step, each round of encryption including *S*-box transformation, *P* permutation and round key XOR. The output is the next round of input. Finally we get final cipher text C.

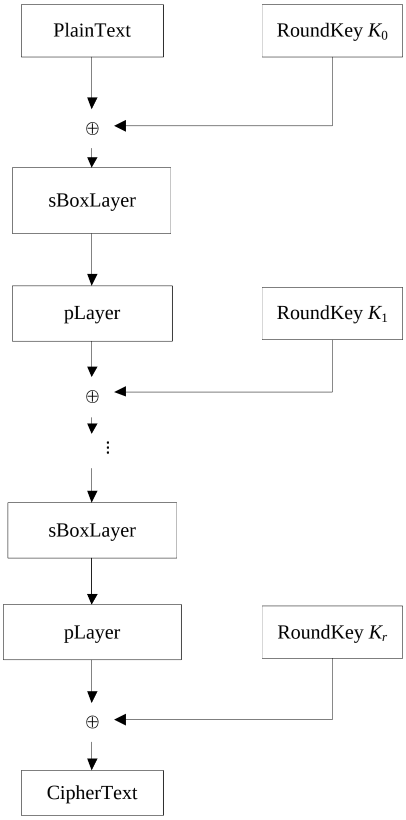

Fig.2. The VH encryption algorithm description

- *Initial Encryption*

The initial cipher text $C_0 = P \oplus K_0$, where P is the 64-bit initial plain text, K_0 is the first 8 bytes of the key *K*.

- *r-round Encryption*

For *i* within [1, *r*], each round of iteration has the following three steps:

Firstly, perform pseudorandom transformation on each byte of the data using the encryption *S*-box：$M_i(j) = S[C_{i-1}(j)]$， where *i* range from 1 to *r*, $X_i(j)$ denotes the *j* byte of X_i, 0≤*j*≤7.

Secondly, arrange the 64-bit data M_i into a 8*8 matrix, perform pseudorandom transformation on each diagonal line of M_i using the encryption *S*-box.

$$P_i(0) = S\{[M_i(0)\&128|[M_i(1)\&64]|[M_i(2)\&32]|[M_i(3)\&16]$$

$$|[M_i(4)\&8]|[M_i(5)\&4]|[M_i(6)\&2]|[M_i(7)\&1]\}$$

$$P_i(1) = S\{[M_i(1)\&128|[M_i(2)\&64]|[M_i(3)\&32]|[M_i(4)\&16]$$

$$|[M_i(5)\&8]|[M_i(6)\&4]|[M_i(7)\&2]|[M_i(8)\&1]\}$$

$$P_i(2) = S\{[M_i(2)\&128|[M_i(3)\&64]|[M_i(4)\&32]|[M_i(5)\&16]$$

$$|[M_i(6)\&8]|[M_i(7)\&4]|[M_i(0)\&2]|[M_i(1)\&1]\}$$

$$P_i(3) = S\{[M_i(3)\&128|[M_i(4)\&64]|[M_i(5)\&32]|[M_i(6)\&16]$$

$$|[M_i(7)\&8]|[M_i(0)\&4]|[M_i(1)\&2]|[M_i(2)\&1]\}$$

$$P_i(4) = S\{[M_i(4)\&128|[M_i(5)\&64]|[M_i(6)\&32]|[M_i(7)\&16]$$

$$|[M_i(0)\&8]|[M_i(1)\&4]|[M_i(2)\&2]|[M_i(3)\&1]\}$$

$$P_i(5) = S\{[M_i(5)\&128|[M_i(6)\&64]|[M_i(7)\&32]|[M_i(0)\&16]$$

$$|[M_i(1)\&8]|[M_i(2)\&4]|[M_i(3)\&2]|[M_i(4)\&1]\}$$

$$P_i(6) = S\{[M_i(6)\&128|[M_i(7)\&64]|[M_i(0)\&32]|[M_i(1)\&16]$$
$$|[M_i(2)\&8]|[M_i(3)\&4]|[M_i(4)\&2]|[M_i(5)\&1]\}$$
$$P_i(7) = S\{[M_i(7)\&128|[M_i(0)\&64]|[M_i(1)\&32]|[M_i(2)\&16]$$
$$|[M_i(3)\&8]|[M_i(4)\&4]|[M_i(5)\&2]|[M_i(6)\&1]\}$$

Finally, obtain the cipher text of the current iteration by performing XOR operation on the output P_i above and the sub-key K_i of the current iteration: $C_i = P_i \oplus K_i$, where *r* range from 1 to *r*. The result output in the last round is the final cipher text *C*.

The Lightweight Stream Cipher SVH.

SVH has 4 parameters: the plain sequence *x*, the key stream KS, the initial vector KS_0, and the cipher sequence *y*. The length of KS_0 is 64-bit, it is not repeated pseudo-random number. Its function is to resist replay attack. For the len-bit of the plaintext *x*, SVH based on block cipher VH by OFB mode: $n = \lceil(\text{len-1})/64\rfloor + 1$ group key stream element, $1 \le i \le n$, each key stream's length is 64-bit, as shown in Figure 3. For *i*=1 to *n*, $\mathrm{KS}_i = \mathrm{VH}(\mathrm{KS}_{i\text{-}1})$, where which VH (*X*) indicates that the block cipher VH uses the key to encrypt the data X. Encryption scheme for SVH:$y=x\oplus\mathrm{MSB}_{\mathrm{len}}(\mathrm{KS})$, Where $\mathrm{MSB}_{\mathrm{len}}$ (KS) represents the interception of all key stream KS len-bit. Decryption scheme for SVH is $x = y\oplus\mathrm{MSB}_{\mathrm{len}}(\mathrm{KS})$.

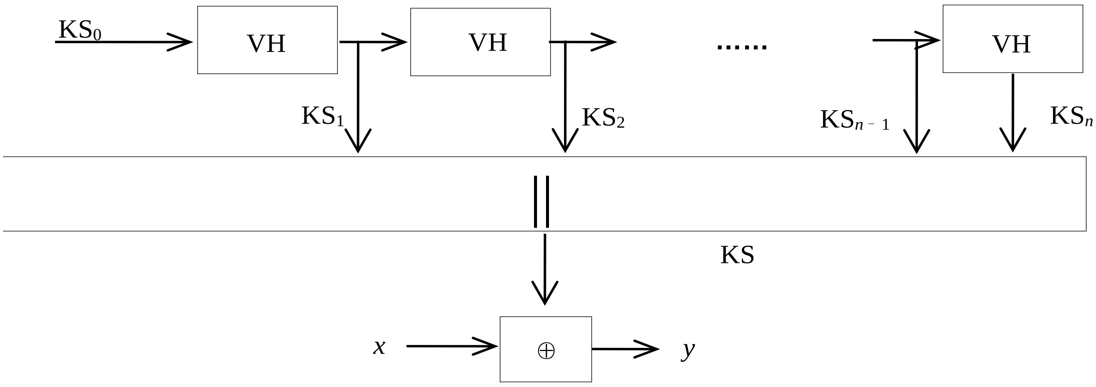

Fig.3. The SVH encryption algorithm description

Performance Evaluation

Software Implementations.

The implementation performance of SVH-80 is tested by using the C language in the environment of Intel(R) Core(TM) i7-3610QM 2.3GHz CPU and 8 GB memory. From the comparison with existing lightweight stream cipher Grain, WG-7 and A2U2, it can be seen that the software implementation of SVH consumes much less time than other algorithms. The software efficiency of SVH-80, Grain, WG-7 and A2U2 is 19.95Mb/s, 0.61 Mb/s, 13.43Mb/s and 0.31Mb/s respectively, which are shown in Tab 1. It can be observed that SVH-80 outperforms other stream cipher in terms of software efficiency.

Table 1. Comparison of lightweight stream cipher implementations

Lightweight stream cipher	Software efficiency (Mb/s)	Hardware efficiency (GE)
SVH-128	11.31	1171
Grain-128	0.61	1458
WG-7	13.43	2194
A2U2	0.31	254

Hardware Implementations.

Efficiency of hardware implementation is usually measured by the equivalent number of GE. The number of GE needed to achieve hardware implementation of CLEFIA-128 was estimated in reference [4]. The authors reported that 5979 GE were required to implement CLEFIA-128, neglecting the effect of different ASIC libraries. The authors in reference stated that 6 GE were needed to store 1 bit, and 2.67 GE were required for a XOR operation. The number of gate circuits needed by SVH and other lightweight stream cipher is given in Tab.2. Compared with Grain which needs 1458 GE for hardware implementation, SVH only demands 1171 GE. It demonstrates that hardware efficiency of SVH is higher than Grain.

Security Evaluation

Differential Cryptanalysis and Linear Cryptanalysis.

Differential cryptanalysis [5] and linear cryptanalysis [6] have been the most effective attack against stream cipher. The authors in reference detailed the method for evaluating a cipher's resistance against differential and linear cryptanalysis, and proposed to compute the maximum differential and linear probability by counting the number of dynamic S-boxes [5-6]. This method was used by reference to evaluate CLEFIA [4].

The maximum differential probability of SVH's *S*-box is computed to be $2^{-4.415}$. The number DS of dynamic *S*-boxes for the first ten rounds of SVH-80, is computed by using the program, which is shown in Table 3. It can be observed that the 4-round maximum differential probability of SVH is $\mathrm{DCP}_{\max}^{4d} \le 2^{21*(-4.415)} = 2^{-67.92} \le 2^{-64}$. When the number round is larger than 4, no effective differential characteristic is found for cryptanalysis. So the full-round SVH can resist differential cryptanalysis.

The maximum linear probability of SVH's *S*-box is computed to be $2^{-2.83}$. The number LS of dynamic *S*-boxes for the first ten rounds of SVH-80 is computed by using the program, which is shown in Table 3. It can be observed that the 4-round maximum linear probability of SVH is $\mathrm{LCP}_{\max}^{4d} \le 2^{24*(-2.83)} = 2^{-67.62} \le 2^{-64}$. When the number of round is larger than 4, no effective linear characteristic is found for cryptanalysis. So the full-round SVH can resist linear cryptanalysis.

Table 2. Guaranteed number of active S-boxes of SVH

Rounds	**DS**	**LS**	**Rounds**	**DS**	**LS**
1	0	0	**6**	35	40
2	7	8	**7**	42	48
3	14	16	**8**	49	56
4	21	24	**9**	56	64
5	28	32	**10**	63	72

Impossible Differential Cryptanalysis.

Impossible differential cryptanalysis is a very effective attack against SVH. J. Kim proposed a moment algorithm $\mu - method$ to perform impossible differential cryptanalysis of the structure of the stream cipher [7]. Their method can find different impossible differential paths. Impossible differential cryptanalysis of SVH is carried out by using this method, determining the maximum number of rounds is 6, finding 8 non-differential paths.

$$(0,0,0,\alpha,0,\alpha,\alpha,\alpha)\overset{6r}{\nrightarrow}(0,0,0,0,\alpha,\alpha,\alpha,\alpha) \qquad p=1$$

$$(0,0,\alpha,0,\alpha,\alpha,\alpha,0)\overset{6r}{\nrightarrow}(0,0,0,0,\alpha,\alpha,\alpha,\alpha) \qquad p=1$$

$$(0,\alpha,0,\alpha,\alpha,\alpha,0,0)\overset{6r}{\nrightarrow}(0,0,0,\alpha,\alpha,\alpha,\alpha,0) \qquad p=1$$

$$(0,\alpha,\alpha,\alpha,0,0,0,\alpha)\overset{6r}{\nrightarrow}(0,\alpha,\alpha,\alpha,\alpha,0,0,0) \qquad p=1$$

$$(\alpha,0,0,0,\alpha,0,\alpha,\alpha)\overset{6r}{\nrightarrow}(\alpha,\alpha,0,0,0,0,\alpha,\alpha) \qquad p=1$$

$$(\alpha,0,\alpha,\alpha,\alpha,0,0,0)\overset{6r}{\nrightarrow}(0,0,\alpha,\alpha,0,0,0,0) \qquad p=1$$

$$(\alpha,\alpha,0,0,0,\alpha,0,\alpha)\overset{6r}{\nrightarrow}(\alpha,\alpha,\alpha,0,0,0,0,\alpha) \qquad p=1$$

$$(\alpha,\alpha,\alpha,0,0,0,\alpha,0)\overset{6r}{\nrightarrow}(\alpha,\alpha,\alpha,\alpha,0,0,0,0) \qquad p=1$$

Where $\alpha \in \mathrm{GF}(2^8)$ denotes the non-zero differential. Therefore, impossible differential cryptanalysis is invalid for SVH.

Conclusion

A new lightweight stream cipher, SVH, has been introduced. It is designed with small hardware and software implementation in mind. A complete description of the algorithm as well as a security analysis based on known attacks have been given. The construction is based on dual pseudo-random transformation and output feedback. The block of key size is 64 bits and SVH can achieve sufficient security margin against known attacks, such as linear cryptanalysis, differential cryptanalysis, impossible differential cryptanalysis. Comparison with other lightweight stream cipher in hardware implementation indicates that SVH needs 1171 GE, less than 1458 GE of the international standard Grain. .It can be considered that the SVH has achieved a superior balance in security and performance to meet the hardware usage needs of extreme environments with limited resources, such as RFID and to meet some software implementation needs from environment, such as embedded mode and MCU.

Acknowledgement

In this paper, the research was sponsored by the Science & Technology Support Plan of Jiangsu Province under Grant No.BE2013879 and the NUAA Research Funding under Grant No.NS2010097.

References

[1] Hell M, Johansson T, Meier W. Grain: a stream cipher for constrained environments. International Journal of Wireless and Mobile Computing [J], 2007, 2(1): 86-93.

[2] Luo Y, Chai Q, Gong G, et al. A Lightweight Stream Cipher WG-7 for RFID Encryption and Authentication [C]. //IEEE Global Telecommunications Conference. IEEE, 2010:1 - 6.

[3] Wu W., Feng D., Zhang W.: Design and analysis of Block cipher (In Chinese) [C]. TsingHua University Press, Beijing,2009.

[4] Shirai T., Shibutani K., Akishita T.: The 128-bit Block cipher CLEFIA. Fast software encryption 2007[J]. LNCS, vol.4593, pp. 181-195. Springer, Heidelberg, 2007.

[5] Su B., Wu W., Zhang W.: Differential Cryptanalysis of SMS4 Block Cipher. IACR. Cryptology Eprint Archive [J], 2010.

[6] Matsui M.: Linear Cryptanalysis Method for DES Cipher. Advances in Cryptology-EUROCRYPT 1993 [J]. LNCS, vol.765, pp. 386-397. Springer, Heidelberg, 1994.

[7] Kim J., Hong S., Sung J.: Impossible Differential Cryptanalysis for Block Cipher Structures. INDOCRYPT 2003 [J], LNCS, vol.2904, pp.82-96. Springer, Heidelberg, 2003.