Chaskey: a MAC Algorithm for Microcontrollers – Status Update and Proposal of Chaskey-12 –

Nicky Mouha[1,2]

[1] Dept. Electrical Engineering-ESAT/COSIC, KU Leuven, Leuven and iMinds, Ghent, Belgium.
[2] Project-team SECRET, Inria, France.
nicky@mouha.be

Abstract. The Chaskey MAC algorithm was presented by Mouha et al. at SAC 2014. It is designed for real-world applications where 128-bit keys are required, but standard cryptographic algorithms cannot be implemented because of stringent requirements on speed, energy consumption, or code size. Shortly after its publication, Chaskey was considered for standardization by ISO/IEC JTC 1/SC 27/WG 2. At the October 2015 meeting, the ISO/IEC committee decided to terminate the study period on Chaskey, and to circulate a first working draft. Since Chaskey was introduced, many follow-up results were published, including improved cryptanalysis results, new security proofs and more efficient implementations. This paper gives a comprehensive overview of those results, and introduces a twelve-round variant of Chaskey: Chaskey-12. Although the original eight-round Chaskey remains unbroken, Chaskey-12 has a much more conservative design, while reducing the performance by only 15% to 30%, depending on the platform.

Keywords: Chaskey-12, Message Authentication Code, Microcontroller, Permutation-Based, ARX.

1 Introduction

Since the AES block cipher [3] was standardized by NIST in 2001, it has become the conventional choice for block-cipher-based constructions. However, AES may be too slow or too large in very constrained environments.

The focus of this paper is on Message Authentication Code (MAC) algorithms. MAC algorithms process a message and a secret key to produce a tag, in such a way that it is infeasible to construct a forgery: that is, to generate a valid message-tag pair without knowledge of the secret key.

At SAC 2014, Mouha et al. [12] proposed the Chaskey MAC algorithm for microcontrollers. Its main design requirement was to be an order of magnitude faster than AES-128-CMAC [5]. Recall that CMAC is a standardized variant of CBC-MAC that supports variable-length messages. It was also designed to be a drop-in replacement for AES-128-CMAC. More specifically, the key size should be 128 bits, speed for short inputs is critical, and nonce values should not be required.

On the ARM Cortex-M4, Chaskey reaches 7.0 cycles/byte, compared to 89.4 cycles/byte for AES-CMAC. Chaskey has a speed of 9.8 cycles/byte on the ARM-Cortex-M0, whereas AES-CMAC requires 136.5 cycles/byte.

The intention of this paper is to provide a status update on Chaskey, by giving an overview of the results since its publication. We will also propose Chaskey-12, a variant of Chaskey with twelve rounds instead of eight. Chaskey-12 has a much higher security margin than Chaskey, but is nevertheless only 15% to 30% slower, depending on the target platform.

2 Description of Chaskey

First, we provide a brief, yet complete description of Chaskey. For additional cryptanalysis and benchmark results, as well as a security proof based on an underlying Even-Mansour block cipher, we refer to the original paper [12].

2.1 Notation

Table 1 summarizes the notation used in this paper. Throughout, $n = 128$ is both the key size and the block size in bits.

We interchangeably consider an element a of $GF(2^n)$ as an n-bit string $a[n-1]a[n-2]\dots a[0]$ and as the polynomial $a(x) = a[n-1]x^{n-1} + a[n-2]x^{n-2} + \dots + a[0]$ with binary coefficients. Let $f(x)$ be an irreducible polynomial of degree n with binary coefficients. We choose $f(x) = x^{128} + x^7 + x^2 + x + 1$. Then to multiply two elements a and b, we represent them as two polynomials $a(x)$ and $b(x)$, and calculate $a(x)b(x) \bmod f(x)$. For example, we show how to multiply an element by x in Algorithm 1. Note that x corresponds to bit string $0^{126}10$, which is 2 in decimal notation.

When converting between bit strings and arrays of 32-bit words, we always use little endian byte ordering. Inside every byte, bit numbering starts with the least significant bit.

2.2 Mode of Operation

Chaskey takes a message m, and splits it into ℓ blocks $m_1, m_2, \dots, m_\ell$ of n bits each. Only the last block m_ℓ may be incomplete, or even empty

Table 1. Notation.

Notation	Description
$x \| y$	concatenation of bit strings x and y
$\|x\|$	length of bit string x
$x + y$	addition of x and y modulo 2^{32} (in text)
$x \boxplus y$	addition of x and y modulo 2^{32} (in figures)
$x \lll s$	rotation of x to the left by s positions
$x \ll s$	shift of x to the left by s positions
$x \oplus y$	bitwise exclusive OR (XOR) of x and y
0^a	bit string consisting of a times 0
$\mathsf{right}_t(a)$	select the t least significant bits of a
$x[i]$	bit selection: bit at position i of word x, where $i = 0$ is the least significant bit

in case $|m| = 0$. It also takes a 128-bit key K, which must be chosen independently and uniformly at random from the entire key space. From K, two 128-bit subkeys K_1 and K_2 are derived, each by means of a 128-bit shift and a 128-bit conditional XOR. The generation of the subkeys is defined in Algorithm 2.

To generate a tag τ of at most n bits, Chaskey iterates an n-bit permutation π, as specified in Fig. 1.

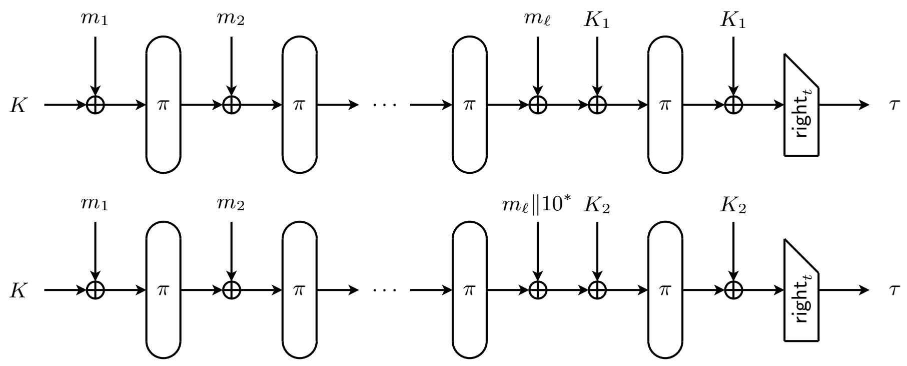

Fig. 1. The Chaskey mode of operation when $|m_\ell| = n$ (top), and when $0 \le |m_\ell| < n$ (bottom). The round function of permutation π is shown in Fig. 2, the subkeys K_1 and K_2 are generated according to Algorithm 2, and $m_\ell \| 10^*$ is shorthand for $m_\ell \| 10^{n-|m_\ell|-1}$.

Algorithm 1 TimesTwo

```
1: procedure TimesTwo($a$)
2:     if $a[127] = 0$ then
3:         return $(a \lll 1) \oplus 0^{128}$
4:     else
5:         return $(a \lll 1) \oplus 0^{120}10000111$
```

Algorithm 2 SubKeys

```
1: procedure SubKeys($K$)
2:     $K_1 \leftarrow$ TimesTwo($K$)
3:     $K_2 \leftarrow$ TimesTwo($K_1$)
4:     return $(K_1, K_2)$
```

2.3 Permutation π

In Chaskey, the permutation π consists of r applications of a round function. This round function is specified in Fig. 2.

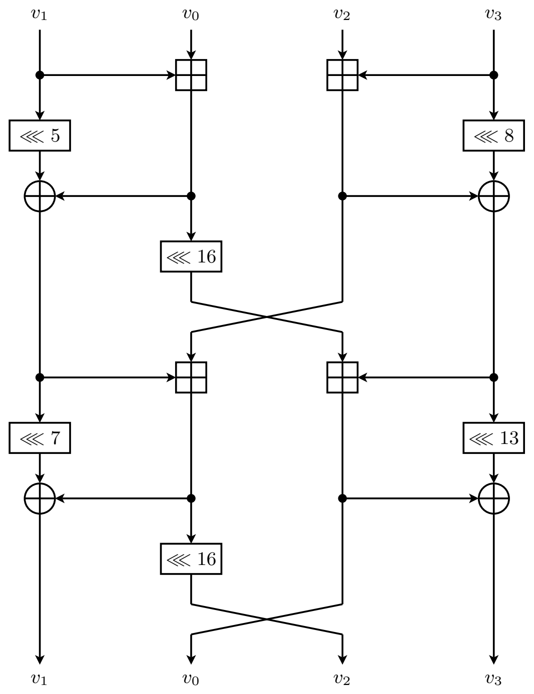

Fig. 2. A round of the Chaskey permutation π, defined as: $v_0 \| v_1 \| v_2 \| v_3 \leftarrow \pi(v_0 \| v_1 \| v_2 \| v_3)$. We intentionally swapped v_0 and v_1, as this reduces the number of crossing lines in the figure.

Chaskey was originally defined with $r = 8$ rounds. At this time of writing, there are no attacks on the full-round Chaskey, and we are confident that this construction will remain secure. However, Leurent [8] has discovered an attack on a variant of Chaskey with $r = 7$. Due to concerns that Chaskey may have a small margin of security, we now propose Chaskey-12, a variant of Chaskey with $r = 12$.

The advantage of Chaskey-12 is that the security margin is greatly increased, but it is only about 15% slower on ARM Cortex-M0 or -M3, and about 30% slower on 16-bit Texas Instruments MSP430 and 8-bit Atmel AVR ATmega128. These Cortex-M benchmarks were communicated to us by Björn Haase, the MSP430 and AVR implementations were constributions of Jason Smith for the FELICS project [4].

2.4 Security Claim

We refer to [12] for a security proof of Chaskey, based on the security of an underlying Even-Mansour block cipher. It states that for any adversary that queries D message blocks in total, attacking Chaskey with a non-negligible probability requires about $2^n/D$ queries to the permutation π.

We should clarify that D is total number of message blocks under any currently-used key. It is assumed that $2^n/D$ off-line evaluations of π is infeasible. To ensure this, Chaskey should be rekeyed before the security drops below an acceptable level.

Chaskey should never use the same key for more than $D = 2^{48}$ message blocks, which corresponds to 4 petabytes of data. It is assumed that only a small number of devices can be attacked at the same time, as this will reduce the security of Chaskey (and many other MAC algorithms [2]), even when the key is refreshed regularly.

To avoid tag guessing attacks, we recommend that the tag size $|\tau| \geq 64$. Changing $|\tau|$ always requires selecting a new key K uniformly at random. Although Chaskey with short tags is vulnerable to efficient tag guessing attacks, it remains secure against all other attacks.[3] Therefore, shorter tags can be used for applications that can tolerate occasionally accepting an inauthentic message.

3 Updates on Chaskey

In this section, we give an overview of Chaskey-related activity that is known at the time of writing (December 2015).

[3] This is a major advantage of Chaskey over the GCM authenticator. For GCM, there is an efficient attack to recover the authentication key when short tags are used [6].

- **August 15, 2014:** Chaskey is introduced at the SAC 2014 conference in Montreal, Canada. [12]
- **November 6, 2014:** ISO/IEC JTC 1/SC 27/WG 2 conducts a preliminary study on the standardization of Chaskey, and ITU-T SG17 has added new work items related to IoT and ITS security, for which Chaskey seems to be well-suited.
- **January 15, 2015:** At ESC 2015, Leurent presents the first external cryptanalysis results on Chaskey [7]. He demonstrates a practically verified attack on 6 rounds with $D = 2^{25}$ and $T = 2^{29}$, and sketches an attack on 7 rounds with $D = 2^{45} - 2^{48}$. His attack is based on differential-linear cryptanalysis with several state-of-the-art improvements. Several novel ideas are introduced, which also lead to an improved attack on FEAL [13].
- **April 21, 2015:** The underlying block cipher of Chaskey was benchmarked by the FELICS project [4] of the University of Luxembourg on a variety of microcontrollers. As the implementation results show, the Chaskey block cipher performs very well on 8-bit, 16-bit and 32-bit microcontrollers. It is nearly always the fastest block cipher, even when compared to other ciphers with significantly smaller block and key sizes.
- **May 19, 2015:** Mennink [10] gives a new security proof for Chaskey, this time considering a strictly stronger fsetting: PRF instead of MAC security. He also shows how Chaskey can easily be made secure against related-key attacks.
- **May 21, 2015:** Bhaumik et al. [1] announce two new observations on Chaskey. Their results do not have any practical impact, as they are only small speed-ups of exhaustive search. However, interesting about these observations is that their query complexity goes down when the number of rounds of Chaskey is increased.
- **July 16, 2015:** An optimized implementation by Björn Haase reaches 9.8 cycles/byte on an ARM Cortex-M0.
- **July 20, 2015:** Chaskey is presented at the NIST Lightweight Cryptography Workshop 2015 in Gaithersburg, Maryland, USA.
- **August 13, 2015:** At SAC 2015, Mavromati [9] presents several new key-recovery attacks on Chaskey. The attacks consider both the single-user and multi-user settings. As the attacks assume that Chaskey uses an ideal permutation, they do not violate the security proof of Chaskey, which can be extended to the multi-user setting using the techniques of Mouha and Luykx [11].
- **September 14, 2015:** The FELICS benchmark results [4] of the Chaskey block cipher have been improved significantly, in particular

for 8-bit and 16-bit microcontrollers. The improved implementations are due to Jason Smith.
- **October 8, 2015:** The research paper of the currently best-known cryptanalysis attack on Chaskey by Leurent is made available [8].
- **October 31, 2015:** ISO/IEC JTC 1/SC 27/WG 2 has decided to terminate the study period, and to circulate a first working draft.

4 Conclusion

Chaskey is a MAC algorithm that is optimized for microcontrollers. It reaches 9.8 cycles/byte on ARM Cortex-M0, and 7.0 cycles/byte on ARM Cortex-M4, which is about order of magnitude faster than AES-CMAC.

Chaskey comes with a proof of security, based on the security of an underlying Even-Mansour block cipher. The proof states that if an attacker obtains D message blocks in total (under any currently-used key), any attack with a non-negligible probability of success requires about $2^{128}/D$ evaluations of the underlying block cipher. Originally only MAC security was proven, but later Mennink provided a PRF security proof as well. Mavromati introduced several key-recovery attacks that match the security bound.

As of today, the full eight-round Chaskey remains secure. However, Leurent showed how up to seven rounds can be attacked. To address concerns about a seemingly small security margin of Chaskey, this paper proposes Chaskey-12. This twelve-round variant has a very comfortable security margin against the best-known attacks, but is only about 15% slower on the 32-bit ARM Cortex-M microcontrollers, and 30% slower on the 16-bit TI MSP430 and the 8-bit Atmel AVR ATmega128.

Acknowledgments. This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007), by Research Fund KU Leuven, OT/13/071, and by the French Agence Nationale de la Recherche through the BLOC project under Contract ANR-11-INS-011, and the BRUTUS project under Contract ANR-14-CE28-0015. Nicky Mouha is supported by a Postdoctoral Fellowship from the Flemish Research Foundation (FWO-Vlaanderen), and by FWO travel grant 12F9714N.

References

1. Bhaumik, R., Dutta, A., Guo, J., Jean, J., Mouha, N., Nikolić, I.: More Rounds, Less Security? Cryptology ePrint Archive, Report 2015/484 (2015), `http://eprint.iacr.org/`

2. Chatterjee, S., Menezes, A., Sarkar, P.: Another Look at Tightness. In: Miri, A., Vaudenay, S. (eds.) Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers. LNCS, vol. 7118, pp. 293–319. Springer (2011), `http://dx.doi.org/10.1007/978-3-642-28496-0_18`
3. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer (2002)
4. Dinu, D., Biryukov, A., Großschädl, J., Khovratovich, D., Corre, Y.L., Perrin, L.: FELICS – Fair Evaluation of Lightweight Cryptographic Systems. `https://www.cryptolux.org/index.php/FELICS_Block_Ciphers_Brief_Results` (2015)
5. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. NIST special publication 800-38b, National Institute of Standards and Technology (NIST) (May 2005), `http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf`
6. Ferguson, N.: Authentication weaknesses in GCM. Comments submitted to NIST Modes of Operation Process (May 2005)
7. Leurent, G.: On Chaskey - Work in progress... Talk at Early Symmetric Cryptography - ESC 2015 (2015)
8. Leurent, G.: Differential and Linear Cryptanalysis of ARX with Partitioning – Application to FEAL and Chaskey. Cryptology ePrint Archive, Report 2015/968 (2015), `http://eprint.iacr.org/`
9. Mavromati, C.: Key-recovery attacks against the MAC algorithm Chaskey. Cryptology ePrint Archive, Report 2015/811 (2015), `http://eprint.iacr.org/`
10. Mennink, B.: XPX: Generalized Tweakable Even-Mansour with Improved Security Guarantees. Cryptology ePrint Archive, Report 2015/476 (2015), `http://eprint.iacr.org/`
11. Mouha, N., Luykx, A.: Multi-key Security: The Even-Mansour Construction Revisited. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I. LNCS, vol. 9215, pp. 209–223. Springer (2015), `http://dx.doi.org/10.1007/978-3-662-47989-6_10`
12. Mouha, N., Mennink, B., Herrewege, A.V., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers. In: Joux, A., Youssef, A. (eds.) Selected Areas in Cryptography - SAC 2014 - 21th International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers. LNCS, vol. 8781, pp. 1–18. Springer (2014), full version: `http://eprint.iacr.org/2014/386`
13. Shimizu, A., Miyaguchi, S.: Fast Data Encipherment Algorithm FEAL. In: Chaum, D., Price, W.L. (eds.) Advances in Cryptology - EUROCRYPT '87, Workshop on the Theory and Application of of Cryptographic Techniques, Amsterdam, The Netherlands, April 13-15, 1987, Proceedings. LNCS, vol. 304, pp. 267–278. Springer (1987), `http://dx.doi.org/10.1007/3-540-39118-5_24`