# Comparative Study of DECIM-128 and DECIMV2 in relation to Compact Hash-based Message Authentication Code

P.Venkateswara Rao, K.Seetha Devi, CH.V. Phani Krishna, Dr. K.Subrahmanyam

*CSE Department, KL University, Guntur Dt., India.*

***Abstract*— constructing compact HMAC (Hash-based Message Authentication Code) is required to maintain integrity and authentication in computationally constrained environments like Wireless sensor networks and RFID. DECIM is a hardware oriented stream cipher submitted to the ECRYPT stream cipher project. It is highly scrutinized stream cipher and is portable to implement hashing for highly compact MAC, which is required to achieve efficiency, while not sacrificing security. In this paper, we present advantages of DECIM-128 when compared to DECIM$^{v2}$ in implementation as hashing in HMAC.**

***Keywords*— Stream cipher, DECIMv2, DECIM-128, eSTREAM, compact HMAC, Decimation, LFSR, ABSG.**

## I. INTRODUCTION

When we are talking about ”Authentication in computationally Constrained Environment”, it is very difficult to implement popular authentication techniques like MD5 and SHA-1. Because they require resources that cannot be provided in constrained environments like Wireless Sensor Networks and RFID [1]. One solution for this is ”COMPACT HMAC” implementation in constrained environment [4].

This Compact MAC is build by implementing “MAC based on Stream Cipher”. This offers high efficiency and possibly consuming minimal resource while being highly secured. These approaches concern ’stream - Cipher- Based Designs’ dedicated to MAC implementation combining hashing and encryption within an integrated solution [2], [6], [9], and [11].

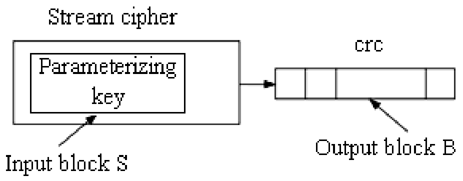

Figure 1: One-way block transformation based on stream cipher.

Benjamin Arazi already described the ways of interacting a one way block transformation based on stream ciphering, for the purpose of performing a general hash, shown in Figure.1. [4].

One of the best stream cipher algorithm that is suitable for constrained environments are DECIM$^{v2}$ which was developed by a team of thirteen researchers from various industrial and academic French Institutes.

Two main reasons selecting this topic are: Firstly, it implements a principle whereby the key consists of a secret part (k) and an initializing public value (IV). This suits very well with the hashing produce implemented with interacting one way block transformation and the security considerations like correlation, collision attacks and related key attacks. Secondly, the size of the parameters suits the practical applications intended to be served by the HMAC proposed with stream cipher. But DECIM$^{v2}$ has some draw backs that can be overcome with DECIM-128. This is submitted as ‘eSTREAM PHASE-3” [8].

This paper makes an attempt to give a comparative view of DECIM-128 and DECIM$^{v2}$.We will study this comparison with respect to DECIM keystream generation(shown in Figure 2) derived from DECIM-128 AND DECIM$^{v2}$ [7].

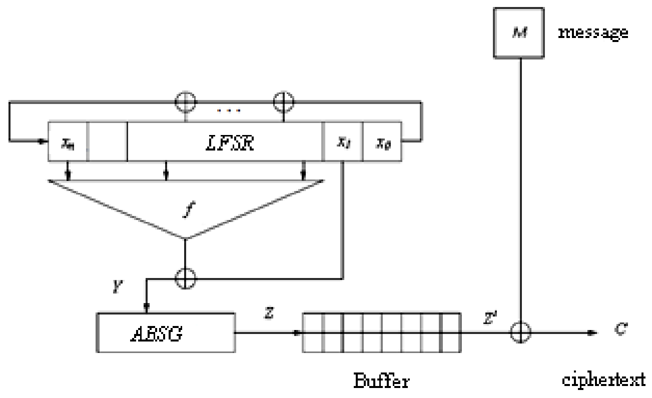

Figure 2. DECIM keystream generation

## II. DESCRIPTION OF DECIM-128 IN COMPARISON WITH DECIMv2.

### A. Key Stream Generation

Key stream Generation Mechanism is given below for DECIM. The difference comes with the bits of the internal state of the LFSR. DECIMv2 is numbered from 0 to 191 and they are denoted by ( $X_0,\ldots\ldots,X_{191}$) as well as DECIM-128 is numbered from 0 to 287 and they are denoted by ( $X_0,\ldots\ldots,X_{287}$).

The sequence of the line feedback value of the LFSR is denoted by S= $(S_t)_{t\geq 0}$. The feed back polynomial of DECIM-128 has the same weight as that of DECIMv2.

## *B. The filtered LFSR*

This section describes the difference between DECIM-128 and DECIMv2 filtered LFSR that generates the sequence ‘y’ (i.e. is an input to ABSG mechanism).

*1) The LFSR:* LFSR of length 288 compared to DECIMv2 that is 192 over F2.This is shown by the following primitive feedback polynomial:

For DECIM$^{v2}$:

$$P(X) = X^{192} + X^{189} + X^{188} + X^{169} + X^{156} + X^{155} + X^{132} + X^{131} + X^{94} + X^{77} + X^{46}+X^{17} + X^{16} + X^{5} + 1$$

For DECIM-128:

$$P(X) = X^{288} + X^{285} + X^{284} + X^{247} + X^{204} + X^{185} + X^{154} + X^{125} + X^{124} + X^{123} +X^{82}+X^{35} + X^{18} + X^{5} + 1.$$

The recursion that corresponding to P for the LFSR is in DECIM-128

$$s_{t+288} =s_{t+283} \oplus S_{t+270}\oplus s_{t+253}\oplus s_{t+206}\oplus s_{t+165}\oplus s_{t+164} \oplus s_{t+163} \oplus s_{t+134} \oplus s_{t+103} \oplus s_{t+84}\oplus s_{t+41} \oplus s_{t+4} \oplus s_{t+3} \oplus s_{t}.$$

And in DECIM$^{v2}$

$$s_{t+192} = s_{t+187} \oplus s_{t+176} \oplus s_{t+175} \oplus s_{t+146} \oplus s_{t+115} \oplus s_{t+98} \oplus s_{t+61} \oplus s_{t+60} \oplus s_{t+37} \oplus s_{t+36} \oplus s_{t+23} \oplus s_{t+4} \oplus s_{t+3} \oplus s_{t}.$$

*2) The Filter:* The Filter function is the 14- Variable Boolean function defines by

$$F: F_2^{14} \longrightarrow F_2; a_1, a_{14} \longmapsto f(a_1, a_{13}) \oplus a_{14}$$

Where f is the symmetric quadratic Boolean function defined by:

$$f(a_1,\ldots,a_{13}) = \bigoplus_{1\le i<j\le 13} a_i a_j \bigoplus_{1\le i\le 13} a_i$$

The filter F is a 13-variable quadratic symmetric function which is balanced.

The only difference between DECIM$^{v2}$ and DECIM-128 regarding the filter is a different choice of tap positions.

For DECIM-128

287,276,263,244,227,203,187,159,120,73,51,39,21,1.

For DECIM$^{V2}$

191,186,178,172,162,144,111,104,65,54,45,28,13,1

And the input of the ABSG at the stage t is thus:

$$y_t = f(s_{t+287}, s_{t+276}, s_{t+263}, s_{t+244}, s_{t+227}, s_{t+203}, s_{t+187}, s_{t+159}, s_{t+120}, s_{t+73}, s_{t+51}, s_{t+39}, s_{t+21}) \oplus s_{t+1}.$$

$$y_t = f(s_{t+191}, s_{t+186}, s_{t+178}, s_{t+172}, s_{t+162}, s_{t+144}, s_{t+111}, s_{t+104}, s_{t+65}, s_{t+54}, s_{t+45}, s_{t+28}, s_{t+13}) \oplus s_{t+1}$$

The sequence `y’ is produced by the filter is of maximum non linear complexity, namely Equal to 288x289/2 =41616.

## *C. Decimation:*

Here there is no change in DECIM-128 compared to DECIMv2.This part describes how the key stream sequence` Z’ is obtained from the sequence `y’. One more interesting fact that the security of DECIM rely more on the length of the involved LFSR than on the ABSG algorithm [12].

## *D. Buffer Mechanism*

The rate of the ABSG mechanism is irregular. Therefore, we use a buffer in order to guarantee a constant throughput. For DECIM-128, we choose a buffer of 64 bits, instead of 32 in DECIMv2. With these parameters the probability that the buffer is empty while it has to out put one bit less than 2-178 at each step (this is 2-89 in DECIMv2).

If the ABSG outputs one bit when the buffer is full, then the newly computed bit is not added into the queue, i.e. it is dropped.

## *E. key/ IV setup*

This subsection describes the computation of the initial inner-state for starting the key stream generation.

*1) Initial filling of the LFSR:* The secret key `k’ is a 128-bit and IV is a 128 bit in DECIM-128, where as in DECIM$^{v2}$ 80 bit and 64bit.

In DECIM-128 the number of possible initial values of the LFSR state is $2^{256}$ this size is large compared to DECIM$^{v2}$. i.e.,$2^{80+64}=2^{144}$.

*2) Update of the LFSR state:* (non-linear feedback)This step consists in updating the LFSR at each clock using the same nonlinear feedback as in DECIM.

If we denote by `$y_t$‘ the output of `f’ at time `t’ and by `$\ell v_t$’ the linear feedback value t > 0, the feedback bit $s_{t+288}$ is given by :

$$S_{t+288} = \ell v_t \oplus y_t.$$

In DECIM-128 the LFSR is clocked 4x288=1152 times in this step. This improves 384 clocks for LFSR compared to DECIM$^{v2}$. (In this LFSR clocks 768 times).

*3) Initial filling of the buffer:* The buffer mechanism guarantees a constant throughput for the keystream.

In DECIM-128, 64 bits chosen as buffer intend of 32 bits in DECIM$^{v2}$.Like in DECIM, the buffer outputs one bit, exactly for every 4 bits that are input into the ABSG. With these parameters, the probability that the buffer is empty while it has to output one bit is less than $2^{-178}$ at each step compared to $2^{-89}$ from DECIM$^{v2}$.

## III. ADAPTING DECIM TO DIFFERENT SECURITY LEVELS

The design of DECIM can be easily adapted to different key and IV sizes.

Changes in the size of the key & IV impact the following design choices in DECIM:

1. Linear feedback shift register and buffer lengths.
2. Subsequent choice of the feedback polynomial and of the filtering function taps.
3. Key and IV injection in the initialization phase [5].

### *A. Length of the LFSR :*

We denote it by `ℓ' the security parameter. Notice first that, in order to obtain the desired security with a secret key of length `ℓ' exactly, then the initialization vector should be at least l-bit long in order to thwart time memory trade-off attacks [10].

In this case, the length of the LFSR is chosen at least as being twice the security parameter.

We propose a register length of $L = (2 + \frac{1}{4})\ell$.

### *B. Length of the Buffer:*

The length of the buffer must be such that the probability that it becomes empty when it has to output one bit is less than $\frac{1}{2^\ell}$.

In order to ensure that, we check that the sum is less than $\frac{1}{2^\ell}$.

### *C. Feedback Polynomial and Filter :*

Both are same in DECIM. We advise that the filtering function remains exactly the same, except for the positions of the taps, for which criteria appear in [10] .We can also check that the sequence produced by the filter is of maximal nonlinear complexity (using the Berlekamp-Massey algorithm [3]), that is equal to Where ` L' is the length of the LFSR which is shown as below [7]:

$$\sum_{i\geq 1} \frac{1 - \sum_{j=i-\ell}^{2i-1} 2^{j+1}\binom{4i-j-1}{j} - \sum_{j=i-\ell}^{2i} 2^{j}\binom{4i-j-1}{j-1}}{2^{4i}}$$

### *D. Initialization Process:*

In DECIM-128 key and IV injection remains same. The Buffer has to be filled before key stream generation starts. There are several possibilities, depending on the implementation:

The usual key stream generation round (without buffer shifting) can be performed until the buffer is full, with an upper bound on the number of rounds.

Another possibility is to perform exactly a fixed number of rounds. Both the upper bound and the fixed numbers are to be chosen such that the buffer is full with probability at least at the end of the process.

This is ensured by choosing a number N of rounds such that we have

$$1 - 2^{L_B} \sum_{i=2L_B}^{N} \frac{\binom{i-L_B-1}{i-2L_B}}{2^{-i}} < \frac{1}{2^\ell},$$

Where $L_B$ is the length of the buffer.

## IV. CONCLUSION:

Implementing Hash transformation for HMAC using DECIM-128 stream cipher in place of DECIM$^{v2}$ will provide more integrity and authentication in devices they have computationally constrained environment. The required number of gates is increased in DECIM-128 implementation compare to DECIM$^{v2}$. If we compromise with little extra size for our devices then this is advisable to get more benefit with respect to integrity and authentication for message transformation.

## REFERENCES

[1] A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M. Robshaw, and Y.Seurin, "Hash Functions and RFID Tags: Mind the Gap," Proc. Workshop Cryptographic Hardware and Embedded Systems (CHES'08), 2008.

[2] B. Zoltak, "VMPC-MAC: A Stream Cipher Based Authenticated Encryption Scheme," Cryptology ePrint Archive, Report 2004/301,2004.

[3] Berlekamp, Elwyn R. (1967). "Factoring Polynomials over Finite Fields". Bell Systems Technical Journal 46: 1853–1859. Later republished in: Berlekamp, Elwyn R. Algebraic Coding Theory. McGraw Hill.

[4] Benjamin Arazi, "Message Authentication in Computationally Constrained Environments," IEEE Transactions on Mobile Computing, vol. 8, no. 7, pp. 968-974, July, 2009.

[5] C. Berbain, O. Billet, A. Canteaut, N. Courtois, B. Debraize, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin, and H. Sibert. Decimv2.In SASC 2006 - Stream Ciphers Revisited: Workshop Record, Leuven, Belgium, 2006.

[6] D. Whiting, B. Schneier, S. Lucks, and F. Muller, "Phelix—Fast Encryption and Authentication in a Single Cryptographic Primitive," Ecrypt Stream Cipher Project, Report 2005/020, 2005.

[7] eSTREAM Phase 3 Candidates. eSTREAM, ECRYPT Stream Cipher Project. http://www.ecrypt.eu.org/stream/decimp3.html.

[8]. eStream, Stream cipher project of the European Network of Excellence in Cryptology ECRYPT, http://www.ecrypt.eu.org/stream.

[9] H. Krawczyk, "LFSR-Based Hashing and Authentication," Proc. Ann. Int'l Cryptology Conf. (CRYPTO 94), pp. 129-139, 1994.

[10] Jin Hong and Palash Sarkar. New Applications of Time Memory Data Tradeoffs. In Ad-vances in Cryptology - ASIACRYPT 2005, Lecture Notes in Computer Science. Springer-Verlag, 2005.

[11] K. Wirt, "ASC a Stream Cipher with Built in MAC Functionality," Int'l J. Computer Science, vol. 2, pp. 131-136, 2007.

[12] Zhang (1)bin.zhang@uni.lu. New Cryptanalysis of Irregularly Decimated Stream Ciphers. DOI: 10.1007/978-3-642-05445-7_28; Springer Link Date: Tuesday, November 03, 2009.