Resource Saving AES-CCMP Design with Hybrid Counter Mode Block Chaining - MAC

M. Razvi Doomun*, K.M. Sunjiv Soyjaudah
Faculty of Engineering, University of Mauritius

Summary
IEEE 802.11i security standard is emerging as an essential security requirement to support the growth of a wide range of wireless data services and applications. However, with the advent of more battery powered wireless devices, efficient and robust cryptographic designs are needed that do not impose high computational overhead and avoid mismatch with limited battery resources and low processing capabilities inherent on these wireless devices. In this paper, we (a) apply a systematic approach to determine computational complexity and efficiency of AES-CCMP (Advance Encryption Standard - Counter Mode with Cipher Block Chaining Message Authentication Code Protocol) designed for IEEE 802.11i, (b) propose a resource saving AES-CCMP design with hybrid CBC-MAC variant merged with Counter Mode encryption, defined as Counter Mode Block Chaining-MAC (or CMBC–MAC), for faster and more efficient data encryption and decryption. A comparative analysis of CCMP with hybrid CMBC-MAC computational complexity is performed to show its energy economy.

Key words:
Security, AES-CCMP, Computational Complexity, Cipher Block Chaining MAC.

1. Introduction

IEEE 80211i security standard suite [1] has been developed as a replacement of the highly vulnerable Wired Equivalent Privacy (WEP) to provide the 'best' security for 802.11 wireless local area network (WLAN). IEEE 802.11i standard consists of the Counter Mode with Cipher Block Chaining –Message Authentication Code Protocol (Counter Mode + CBC MAC = CCMP) [2] and has been designed as a long term security solution. It is based on Advance Encryption Standard (AES) cipher and it offers robust encryption and message integrity as proved in ref [2][3]. Although CCMP is seen as an efficient algorithm which combines the Counter Mode for encryption and CBC MAC for message authentication, its encryption and authentication implementations are based on the relatively intensive Advance Encryption Standard (AES) operations that require support for more powerful hardware. Improvements in battery technology are easily offset by the increasing complexity of security mechanisms for WLAN and Ad Hoc networks. Thus, to guarantee a reasonable battery operation lifetime, designing innovative techniques to conserve power consumption for cryptographic algorithm are crucial. But, successfully designing resource-saving security mechanism needs good comprehension of the relationships between encryption parameters and power consumption. In the work [5], the computational complexity of AES is modeled based on its algorithmic operations to show how the degree of complexity varies with different key size and number of rounds of encryption. Indeed, AES is very robust, but at the expense of high computational operations. Furthermore, when IEEE 802.11i security protocol is used for certain real-time video and audio applications on WLAN or Ad Hoc network, the device CPU is overloaded severely causing noticeable performance hit. The battery power consumption of small wireless devices is severely affected by the computationally heavy cryptographic operations causing rapid power depletion which creates a new problem dimension between robust wireless security and energy efficiency. Moreover, the quest for efficient cryptographic mechanism is still research challenge for providing optimum security in resource constrained wireless networks which requires minimum memory, making optimal use of hardware processing capabilities and consuming the least energy or battery power [5] [4] [6].

In this paper, we first derive the complexity equation model of AES-CCMP to mathematically assess its computational cost in terms of processing cycles of different number of basic operations involved in the execution of the algorithm. The complexity study also unveils the algorithm's practicality in terms of encryption or decryption performance and speed. Eventually, the computational cost of AES-CCMP algorithm is correlated to its energy consumption when executed on any particular hardware platform. But, the mathematical complexity equations of AES-CCMP is independent of the actual platform they are implemented on and this gives us a standardized fairly accurate methodology to measure energy consumption of the security protocol. While currently there is no other standard way of measuring the computational complexity of security protocols, this creates a knowledge gap as it is difficult to compare different security protocols on an energy performance metric. In addition, for optimizing energy efficiency and

Manuscript received October 5, 2008
Manuscript revised October 20, 2008

security performance, we reengineer AES CCMP design by merging counter mode encryption and CBC-MAC to yield the benefit of a faster and lower complexity mechanism.

The motivation of this CCMP redesigned variant is also drawn form the fact that wireless transmission is frequently exposed to different interferences and other unfavorable conditions, thus having relatively high rate of errors and corrupted packets [7]. The whole encryption-transmission-decryption effort of packets is wasted if the wireless channel is error-prone. Adopting the authenticate-then-encrypt / decrypt-then-authenticate approach in CCMP is particularly inefficient for battery power limited wireless devices in noisy wireless channel because considerable energy may be spent for decrypting corrupted packets (error in the payload) and eventually rejecting them due to incorrect message code integrity (MIC) check. A better scheme is used such that the receiver can verify integrity of message to filter out corrupted packets before proceeding for decryption of those packets with correct MIC check only. Hence, our proposal is to revamp the AES-CCMP design making it computationally lightweight and energy-wise efficient by merging the encryption and authentication components. Therefore, optimizing the number of times AES block cipher is solicited in CCMP has been a fruitful avenue to explore.

The paper is organized as follows: We present related works in the field of efficient security in Section 2. The AES-CCMP protocol is discussed in detail in Section 3. Section 4 gives a mathematical analysis and interpretation of AES-CCMP. Section 5 discusses the strengths and imperfections of AES-CCMP. The resource-saving AES-CCMP design variant is proposed in Section 6 and its performance characteristics are analyzed and compared. This is followed by the conclusion in Section 7.

2. Related Work

Optimizing security mechanisms with resource-efficient block cipher implementations is of paramount importance for wireless devices with battery capacity constraint. In order to conserve battery power, new security protocols and optimization techniques [8][9][10][27] are being developed that are more lightweight and energy efficient. For the work in [4], the authors advocate that designing energy efficient security protocols can be realized by substituting the most energy consuming components of the existing security standard, modifying and optimizing the protocol message transactions and continuously adapting the security services to prevailing conditions based on a security policy. The major computational effort expended in security mechanisms is in computing the underlying block cipher supporting the security service. Existing methods to reduce the energy consumption of cryptographic primitives are mostly focused on: reduction in number of rounds, use of simpler operations (e.g. XORs and shifts), merging multiple operations, use of lookup tables, reduced block length, etc. The main cause of inefficiency criticism in wireless network security protocols is the extra energy consumed because of extra processing and overhead. For instance, in AES algorithm, the number of rounds determines the security strength of the algorithm and for each AES key size a minimum number of rounds for which the algorithm is considered to be secure as presented in ref [12]. While increasing the number of rounds increases the security margin but the overhead and energy consumption for each block will also increased. As such, for example, a variant CBC-MAC authentication mechanism can use a minimum number of AES invocations, one for each block of the message or fraction thereof, resulting in performance equivalent to classic CBC-MAC.

In the works [6][11], performance and energy characteristics of block ciphers are evaluated based on a set of metrics such as energy consumption, code size, or memory footprint, all of which are important for resource-constrained wireless systems. In ref [10], Chandramouli et al. use an experimental approach to estimate a mathematical model for the relationship between power consumption and security of block cipher. The optimal number of encryption rounds for a packet is computed and the vulnerability metric is minimized subject to a total power constraint. However, diverse experimental energy measurement will generally give different results mainly due to the hardware characteristics of the devices involved. Moreover, in such cases it is not possible to accurately compare energy performance measurements of security algorithms between heterogeneous hardware or software implementations. Therefore, we need to extend and normalize the analysis to evaluate the performance and energy characteristics of the mode of operations of block ciphers when used as a complete security package, such as CCMP in the IEEE 802.11i security protocol.

Denial of service, which is another severe threat in security protocols, causes a victim wireless device to waste enough resources keeping connections open so that the latter is unable to participate in any more instances of the security protocol and is thus effectively cut off from the network. Some adversaries use authentication as a means of launching denial of service attacks, since it is both computation and storage-intensive. At the end, the victim wireless device would squander its resources, namely battery power, verifying a series of incorrectly authenticated (corrupted) messages from the attacker. Sometimes, the error-prone nature of the wireless environment can cause burst of corrupted packets and this results in wastage of resources for decryption of packets

and verification of erroneous message authentication codes. Thus, there is a need for security schemes with cryptographic protocols that can prevent wasteful decryption and authentication by eliminating non-efficient functioning. But, ultimately, there are always tradeoffs among security strength, communication overhead, computational complexity, energy consumption and scalability that require in-depth security and performance analyses are studied to find a balance between a security protocol and other network protocols so that the overall security strength and network performance are not degraded [13]. Furthermore, without a systematic view, individual security protocols developed for different layers might provide redundant security services and unnecessarily consume the precious wireless network resource [14]. To some extent, a non-harmonious design of security protocol can use up device and network resources and therefore inadvertently lead a Denial of Service (DoS) attack. In this work, we propose a resource-saving AES-CCMP design for reducing the complexity and energy consumption of IEEE 802.11i security protocol in WLANs and it is supported by methodical complexity analysis of AES-CCMP.

3. AES-CCM Protocol

AES-CCMP is based on the Rijndael block cipher [12] algorithm which has a well-designed mathematical structure. The overhead in terms of basic operations for iterative block cipher encryption is relatively low compared to other block ciphers [11] as well as its comparatively better efficiency in terms of low memory requirements makes AES suitable to be deployed in wireless devices. The Specification for Enhanced Security over Wireless Networks, IEEE 802.11i, requires a strong encryption standard, and naturally, the use of AES is strongly desired. Therefore, a combination of Counter (CTR) mode encryption and CBC-MAC authentication is proposed in the standard [1]. While CTR-AES encrypts data transferred (i.e. achieves confidentiality) using an encryption key, CBC-MAC provides integrity of data and authentication of the sender by calculating the Message Integrity code (MIC) for message authentication with an authentication key. AES itself is a very strong cipher, but counter mode makes it difficult for an eavesdropper to spot patterns, and the CBC-MAC message integrity method ensures that messages have not been tampered with. If the correct MIC sizes, key sizes and MAC algorithms are used, then it is impossible to inject spoofed packets with a valid MAC into the WLAN. The overall AES-CCMP process is shown in Figure 1.

3.1 CBC-MAC and CTR Mode

For efficiency reason, CBC MAC allows for the creation of a message authentication code (MAC) using AES block cipher to check the integrity of a message in a secret key setting. CBC-MAC is encountered with the nice property of reusing the existing AES cipher block in low-end cryptographic devices to guarantee that any exchanged message was not altered while in transit. The plaintext message, in the form of 128-bit block of data, is 'encrypted' with AES block cipher algorithm in CBC mode to create a chain of blocks such that each block depends on the proper encryption of the block before it, as illustrated in **Figure 1**. This ensures that a change to any of the plaintext bits will cause the final encrypted block to change in a way that cannot be predicted without knowing the key to the block cipher because it uses XORing of ciphertext output in previous stage with the next one. In CCMP, CBC MAC provides data integrity only, but not data encryption, and it offers 32, 64 and 128 bits MIC tag length sizes. For the first CBC-MAC Initial Value (IV) block, special input values such as Flag, Data Length (***DLen***) and Nonce (***N***) are required. As CBC-MAC works on blocks of fixed length, both the CCMP Header and Plaintext data need to be padded to get it to the required length. The final MIC output is 128-bit size, but the CCMP takes only the upper 64-bit as truncated MIC [15]. The MIC eventually becomes an input to AES-CTR for encryption. Moreover, there are several variant of CBC-MAC described in [16][17][18]. For messages of fixed (non-zero) length, the simple CBC-MAC has been proved secure in [19]. In order to protect MACs against Birthday paradox attacks, a unique identifier can be added to each message to randomize the MACs [1]. Counter (CTR) mode encryption uses an arbitrary number (counter) that increments with each 128-bit block of data encrypted. The counter is first encrypted with AES, and the output is XORed with a 128-bit plaintext block to produce a ciphertext block. All counters used are unique and all the AES CTR mode encryption or decryption may be performed in parallel or pre-computed in advance for speed gain. The initial CTR is constructed from the flag field, length of payload and the nonce. The CCMP header contains information like ***PN*** value necessary to prevent replay attacks. The packet containing the encrypted message with its MIC, CCMP header and MAC header is sent over the insecure wireless channel. AES-CCMP decryption, since it is almost identical to encryption process. The receiver first decrypts the ciphertext blocks using AES-CTR mode and then proceeds with the MIC calculation using CBC-MAC to compare with received MIC results for data integrity.

4. Cipher Complexity Analysis

In general, most block ciphers share a number of elementary operations such as table lookups of varying size, bitwise Boolean operations, basic arithmetic operations, as well as bitwise shift and rotate operations by a fixed or variable number of position. The required confusion in an encryption algorithm is achieved by successively using different and "incompatible" group operations on blocks of data and mixing them (in such a way that at no point in the encryption process the same algebraic operation is used contiguously) while the structure of the cipher is carefully chosen to provide the necessary diffusion requirement (influence of each key, number of rounds and plaintext bit on every ciphertext bit). The cost of encryption and decryption depends on a number of parameters: the size of the plaintext and ciphertext; the implementation complexity of the algorithm; the cipher mode adopted; and the key scheduling. Specifically, key length is important and the longer the key, the higher the encryption time naturally. Also, the cost for decryption depends on the effort of the packet number and MIC checks needed to accept or validate the decryption. Most importantly, the intricacy of the cipher mode of operation can radically affect the cost of security services.

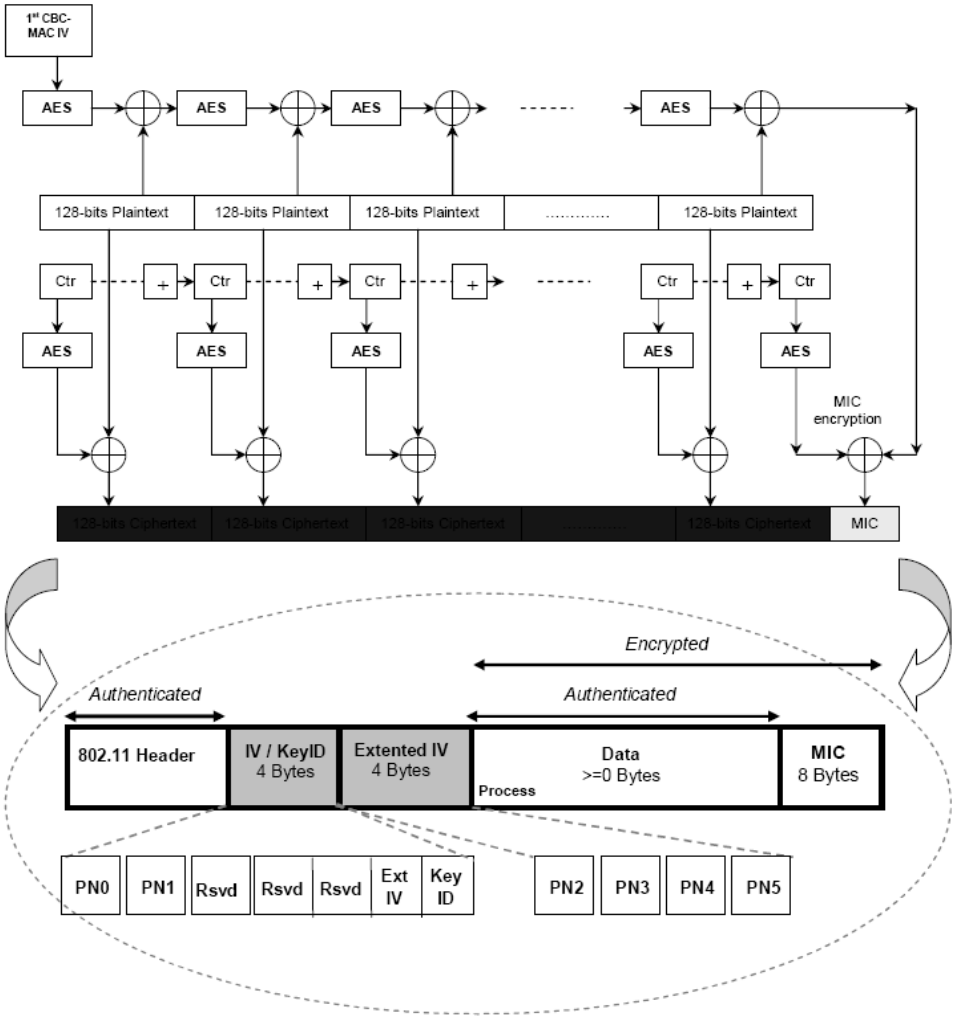

Figure 1. AES-CCMP structure and Output packet format

4.1 AES Computational Analysis

Commonly, CCMP uses AES 128-bit key, although there is provision for larger key size of 192 and 256 bits to implement AES. Being a block cipher, AES relies on the fact that it must encrypt and decrypt 128-bit blocks at a time. This means that even if the message contains a length that is not a factor of 128 bits, CCMP will perform padding on the block so that it meets the 128-bit requirement. In this section, we come up with a computational complexity model of the cipher function and it can be used to compute the energy consumption for the core encryption part. For example, for a given block data size and number of rounds using the AES cipher, we can obtain its computational complexity. However, mathematical analysis of operations in any algorithm overlooks the complexity of memory transfer, controls units and other software intricacies.

So, extending and applying previous works on block cipher analysis [5] [20] and by Granelli et *al.* [21], the computational complexity of AES-CCMP is derived in terms of the algorithm's basic operations like bytewise-AND, bytewise-OR and shift of bytes. Thus, the total number of processing cycles for **encrypting a data block** using AES-Irondale, is given as follows in terms of basic operations of byte-wise AND, byte-wise OR, and a byte-wise Shift :

$\mathbf{T_{AES\text{-}ENCRYPT}} = (8\mathbf{N_b}\mathbf{T_{and}} + 4\mathbf{N_b}\mathbf{T_{or}}) + [46\mathbf{N_b}\mathbf{T_{and}} + (31\mathbf{N_b} + 12)\,\mathbf{T_{or}} + (31\mathbf{N_b} + 12)\,\mathbf{T_{shift}}]\,(\mathbf{N_r} - 1) + (8\mathbf{N_b}\,\mathbf{T_{and}} + 7\mathbf{N_b} + 3\mathbf{N_b}\mathbf{T_{shift}})$,

where:

- T_{and}, T_{or}, and T_{shift} denote the numbers of processing cycles required for performing basic operations of a byte-wise AND, a byte-wise OR, and a byte-wise Shift respectively.
- N_b = blocklength / 32 (here $N_b = 4$ since the size of a data block will be 128-bit)
- N_r = number of rounds of block cipher

The details mathematical proof for deriving the above equation can be referred in [5] and the equation is presented in simplified form as follows:

$\mathbf{T_{AES\text{-}ENCRYPT}} = (46\mathbf{N_b}\,\mathbf{N_r} - 30\mathbf{N_b})\,\mathbf{T_{and}} + [31\mathbf{N_b}\,\mathbf{N_r} + 12(\mathbf{N_r} - 1) - 20\mathbf{N_b}]\,\mathbf{T_{or}} + [64\mathbf{N_b}\,\mathbf{N_r} + 96(\mathbf{N_r} - 1) - 61\,\mathbf{N_b}]\,\mathbf{T_{shift}}$,

Using another result in [5], the total number of processing cycles in computational effort required for AES **decryption of one block of data** is expressed as:

$\mathbf{T_{AES\text{-}DECRYPT}} = \mathbf{T_{AES\text{-}ENCRYPT}} + \{[96\,\mathbf{N_b}\mathbf{T_{and}} + 72\,\mathbf{N_b}\mathbf{T_{or}} - 32\,\mathbf{N_b}\mathbf{T_{shift}}] \times (\mathbf{N_r} - 1)\}$.

The computational effort, in terms of $\mathbf{T_{AES\text{-}ENCRYPT}}$ equation, is now used to investigate the overall computational complexity of CCMP. However, the CCMP encryption and CCMP decryption, both utilize the forward AES block cipher only, which circumvent the more complex inverse AES cipher. In the next section, the computational complexity of CBC-MAC, which is a key component of CCMP, is determined.

4.2 CBC-MAC Computational Analysis

The CBC-MAC algorithm to compute MIC computed is expressed as follows:

For each message $\boldsymbol{M} = \mathbf{M_1}, \mathbf{M_2}, ..., \mathbf{M_n}$

$\mathbf{O} = \mathbf{E_k}\,(\mathbf{B_o})$

For i = 1...n do

$\mathbf{MIC_i} = \mathbf{E_k}\,(\mathbf{O} \oplus \mathbf{M_i}),\ \mathbf{O} \leftarrow \mathbf{MIC_i}$

$\mathbf{MIC} = \mathbf{MIC_n}$,

where:

$\boldsymbol{n}$ is the number of 128-bit data blocks

$\mathbf{B_o}$ = 1st starting CBC-MAC block,

O is the output block,

$\mathbf{E_k}$ is the AES encryption algorithm and

$\oplus$ represents the bitwise XOR operation.

Therefore, the total number of processing cycles to generate the MIC is:

$\mathbf{T_{CBC\text{-}MAC}} = 1 * \mathbf{T_{AES\text{-}ENCRYPT}} + \boldsymbol{n}* (1 * \mathbf{XOR} + 1 * \mathbf{T_{AES\text{-}ENCRYPT}})$

In this case, the XOR operation between the 128-bit output block **O** and the 128-bit plaintext block $\mathbf{M_i}$ is similar to the AddRoundKey operation present in AES cipher operation. We employ the fundamental logical expression that for a simple bitwise-XOR it is equivalent to the sum of 2 bitwise-ANDs and 1 bitwise-OR. Therefore, it can be deduced that in the case of the XOR with 128-bit (or 16 bytes) data blocks, **1 block XOR** = $8\mathbf{N_b}\mathbf{T_{and}} + 4\mathbf{N_b}\mathbf{T_{or}}$, where is $\mathbf{N_b}$ = block length/32.

Consequently, considering a message of $\boldsymbol{n}$ such data blocks, the total number of processing cycles for CBC-MAC is:

$\mathbf{T_{CBC\text{-}MAC}} = 1 * \mathbf{T_{AES\text{-}ENCRYPT}} + \boldsymbol{n} * \mathbf{T_{AES\text{-}ENCRYPT}} + \boldsymbol{n} * (8\mathbf{N_b}\mathbf{T_{and}} + 4\mathbf{N_b}\mathbf{T_{or}})$

$\mathbf{T_{CBC\text{-}MAC}} = (\boldsymbol{n} + \boldsymbol{1}) * \mathbf{T_{AES\text{-}ENCRYPT}} + \boldsymbol{n} * (8\mathbf{N_b}\mathbf{T_{and}} + 4\mathbf{N_b}\mathbf{T_{or}})$

$\mathbf{T_{CBC\text{-}MAC}} = (\boldsymbol{n} + \boldsymbol{1}) * \{(46\mathbf{N_b}\,\mathbf{N_r} - 30\mathbf{N_b})\,\mathbf{T_{and}} + [31\mathbf{N_b}\,\mathbf{N_r} + 12(\mathbf{N_r} - 1) - 20\mathbf{N_b}]\,\mathbf{T_{or}} + [64\mathbf{N_b}\,\mathbf{N_r} + 96(\mathbf{N_r} - 1) - 61\,\mathbf{N_b}]\,\mathbf{T_{shift}}\} + \boldsymbol{n} * (8\mathbf{N_b}\mathbf{T_{and}} + 4\mathbf{N_b}\mathbf{T_{or}})$

The extra $\mathbf{T_{AES\text{-}ENCRYPT}}$ in CBC-MAC computation results from the encryption of the 1st CBC-MAC IV block.

4.3 CTR Computational Analysis

In the CTR mode, the counter blocks are encrypted with AES to produce a sequence of output blocks that are XORed with the plaintext blocks to produce the ciphertext. All counters must be different, i.e. no reuse allowed, for all of the messages that are encrypted under the given key. If we again represent a message $M = M_1, M_2, ..., M_n$, and AES is applied to input block counters (128 bits) to produce output blocks (***O***) which are then XORed with the plaintext blocks (*P*) to produce the encrypted data or ciphertext (*C*), then the full counter mode algorithm is expressed as:

Initially, counter ← 0
For each message $M = M_1, M_2, ..., M_n$
Initial-counter ← counter
For i = 1 to n do
$C_i \leftarrow M_i \oplus E_K$ (counter), counter ← counter + 1

As a result, the total number of processing cycles to encrypt only the ***n*** 128-bit plaintext blocks is:

$$\mathbf{T'_{CTR}} = *\boldsymbol{n} *(1*\mathbf{XOR} + 1 * \mathbf{T_{AES\text{-}ENCRYPT}})$$
$$= \boldsymbol{n} * (8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}}) + \boldsymbol{n} * \mathbf{T_{AES\text{-}ENCRYPT}}$$

Note that we are not considering the counter increment operation as it is negligible in the computation analysis. Also, the MIC tag needs to be encrypted along with the data payload. By encrypting the MIC, we avoid all the collision attacks on CBC-MAC mode. The total number of processing cycles to encrypt the MIC only is simply:

$$\mathbf{T'_{MIC}} = 1*\mathbf{XOR} + 1 * \mathbf{T_{AES\text{-}ENCRYPT}}$$
$$= 8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}} + \mathbf{T_{AES\text{-}ENCRYPT}}$$

Finally, the total number of processing cycles to encrypt both the message M and its MIC Tag is:

$$\mathbf{T_{CTR}} = \mathbf{T'_{CTR}} + \mathbf{T'_{MIC}}$$

$$= (\boldsymbol{n+1})*\mathbf{XOR} + (\boldsymbol{n+1})*\mathbf{T_{AES\text{-}ENCRYPT}}$$
$$= (\boldsymbol{n+1})*(8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}}) + (\boldsymbol{n+1})*\mathbf{T_{AES\text{-}ENCRYPT}}$$
$$= (\boldsymbol{n+1})*(8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}}) + (\boldsymbol{n+1})* \{(46\mathbf{N_b N_r} - 30\mathbf{N_b})\ \mathbf{T_{and}} + [31\mathbf{N_b N_r} + 12(\mathbf{N_r} - 1) - 20\mathbf{N_b}]\ \mathbf{T_{or}} + [64\mathbf{N_b N_r} + 96(\mathbf{N_r} - 1) - 61\ \mathbf{N_b}]\ \mathbf{T_{shift}}\ \}$$

4.4 AES-CCMP Computational Analysis

Now, we can formulate the AES-CCMP computational operating cost model in terms of basic operations from the equations derived for the CBC-MAC and CTR-mode AES operation. The resultant **computational complexity for the AES-CCMP** is written as:

$$\mathbf{T_{AES\text{-}CCM}} = \mathbf{T_{CBC\text{-}MAC}} + \mathbf{T_{CTR}}$$
$$= (\boldsymbol{n+1})*\mathbf{T_{AES\text{-}ENCRYPT}} + \boldsymbol{n}*(8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}}) + (\boldsymbol{n+1})*(8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}}) + (\boldsymbol{n+1})*\mathbf{T_{AES\text{-}ENCRYPT}}$$

$$\mathbf{T_{AES\text{-}CCM}} = 2(\boldsymbol{n+1})\ \mathbf{T_{AES\text{-}ENCRYPT}} + (\boldsymbol{2n+1})\ (8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}}),$$

where, ***n*** is the number of plaintext blocks obtained after splitting the data into multiple of 128 bits.

The full complexity equation as the total number of processing cycles is thus:

$$\mathbf{T_{AES\text{-}CCM}} = 2(\boldsymbol{n+1})\ \{(46\mathbf{N_b N_r} - 30\mathbf{N_b})\ \mathbf{T_{and}} + [31\mathbf{N_b N_r} + 12(\mathbf{N_r} - 1) - 20\mathbf{N_b}]\ \mathbf{T_{or}} + [64\mathbf{N_b N_r} + 96(\mathbf{N_r} - 1) - 61\ \mathbf{N_b}]\ \mathbf{T_{shift}}\ \} + (\boldsymbol{2n+1})\ (8\mathbf{N_b T_{and}} + 4\mathbf{N_b T_{or}}),$$

We do not have a benchmark tool that can measure exactly how much CPU cycles is used for each basic AND, OR & SHIFT. We assumed fixed unit cycle for all basic operation, but it could be changed easily in the equation if the CPU cycles for the different basic instructions are known.

CBC-MAC and CTR mode contribute equally to the overall complexity of CCMP, as shown in Table 1. Complexity of CCMP increase with increasing number of encryption rounds, as well as increasing message size.

Number of 128 bit blocks (n) & Complexity Component		AES-CCMP Computational Complexity		
		Key size 128 bits		
		6 Rnd	8 Rnd	10 Rnd
1	$T_{CBC\text{-}MAC}$	7008	9696	12384
	T_{CTR}	7056	9744	12432
	$T_{AES\text{-}CCM}$	14064	19440	24816
2	$T_{CBC\text{-}MAC}$	10536	14568	18600
	T_{CTR}	10584	14616	18648
	$T_{AES\text{-}CCM}$	21120	29184	37248
3	$T_{CBC\text{-}MAC}$	14064	19440	24816
	T_{CTR}	14112	19488	24864
	$T_{AES\text{-}CCM}$	28176	38928	49680
10	$T_{CBC\text{-}MAC}$	38760	53544	68328
	T_{CTR}	38808	53592	68376
	$T_{AES\text{-}CCM}$	77568	107136	136704
20	$T_{CBC\text{-}MAC}$	74040	102264	130488
	T_{CTR}	74088	102312	130536
	$T_{AES\text{-}CCM}$	148128	204576	261024
30	$T_{CBC\text{-}MAC}$	109320	150984	192648
	T_{CTR}	109368	151032	192696
	$T_{AES\text{-}CCM}$	218688	302016	385344
40	$T_{CBC\text{-}MAC}$	144600	199704	254808
	T_{CTR}	144648	199752	254856
	$T_{AES\text{-}CCM}$	289248	399456	509664

Table 1: AES-CCMP Computational Complexity

5. Further Discussion on AES-CCMP

AES-CCMP decryption will use the same number of processing cycles as the AES-CCMP encryption because both, CCMP encryption and CCMP decryption operations require only the forward AES block cipher function. Moreover, Xiao *et al.* [22] analyzed the security overhead

of AES-CCMP in IEEE 802.15.4 specification and observed that processing cycles per block increases as key length increases, payload increases or MIPS (millions of instructions per second) decreases. The important observation made is that the increase of processing cycles of AES-CCMP over the key length and the payload size tends to be **linear** [22][20]. In the next section, we discuss the strength and imperfections in the AES-CCMP and address why, how and what to optimize in it.

5.1 Strengths of AES-CCMP

Although AES-CCMP is a well-known and reliable security ensuring both authentication and integrity of the data, having been widely scrutinized and documented to avoid potential implementation loopholes, we believe there is no perfect security algorithm. First, the following points summarize the key advantages of the security protocol:

(i) AES-CCMP readily handles messages in which certain parts are intended to be authenticated only and not encrypted, and this is done without any additional ciphertext overhead. It can use a single key to provide authentication and integrity. Thus, it reduces key management overhead and minimizes the time spent computing AES key schedules.

(ii) AES-CCMP encryption and decryption use only the forward AES block cipher function rather than the more costly and processing-intensive inverse AES cipher. Using only the AES forward cipher leads to significant savings in code and hardware size.

(iii) AES-CCMP is powerful and offers greater data privacy by encrypting parts of the 802.11 header. It computes the CBC-MAC over the IEEE 802.11 header length, selected parts of the IEEE 802.11 MAC Payload Data Unit (MPDU) header, and the plaintext MPDU data, whereas the old IEEE 802.11 WEP mechanism provided no protection to the MPDU header.

(iv) CCMP implementation allows parallel implementation and further streamlining of AES-CCMP in hardware or software. CTR mode offers several advantages, since there is no computational dependency between successive cipher blocks, C_i and C_j. This enables effective utilization of the software and hardware efficiency by enabling parallelized computation, i.e. one can be computing blocks C_1, C_2, … all at the same time, limited only by the amount of hardware. Hence, CTR mode encryption is fast and bulk data can be encrypted quickly due to parallelized computing. Pipelining CBC-MAC and CTR-mode can be used to increase throughput [26]. CTR mode is simple, as both encryption and decryption depend on a single cipher function.

(v) AES-CCMP mechanism protects users from replay attacks because it uses packet sequence numbers, while it uses temporal key which is derived from the robust 4-way handshake scheme. IEEE802.11i standard specifies that 1st CBC-MAC IV and Counter value (Ctr) of CTR mode are never repeated with the same TK as keystream reuse must not occur.

(vi) Lastly, there are no patent issues regarding the use of AES-CCMP and all intellectual property rights to CCMP have been released into the public domain [15].

5.2 Limitations in AES-CCMP

In this section we discuss possible disadvantages of AES-CCMP implementation in certain contexts:

(i) AES-CCMP is relatively more complex and consumes more energy compared to its predecessors, namely WEP and TKIP. This poses a liability when providing security services to devices with limited battery power. Recently, in the literature [6][11], important studies of energy requirements of cryptographic algorithms has been tackled which shows a strong motivation to investigate techniques which lead to energy-efficient execution of security protocols. However, to our knowledge, a comprehensive analysis of the computational complexity and energy requirement of AES-CCMP has not been addressed before.

(ii) AES-CCMP is not backwards compatible with legacy Wi-Fi hardware. This means AES-CCMP deployments may require a firmware or hardware upgrade. Most drivers do not support the co-existence of RC4-based encryption and AES-CCMP on the radio link [7]. It results in more complex configurations to support two networks in parallel, for example RC4 for broadcast frames and CCMP for Unicast frames transmission.

(iii) Next, AES-CCMP suffers from performance problems when implemented on software compared to other modes of operations. This is because AES used in CCMP is inherently a block cipher used in stream CTR-mode. Consequently, the majority of time of the protocol is spent on computing the AES algorithm. From both a technical and economic aspects, CCMP uses a lot more resources due to the additional overhead required for encryption and decryption. If the wireless device does not have the hardware capability to run the AES and is forced to do it in software, then the wireless network would become unusable.

(iv) CCMP makes use of a single-key AES key. It uses this key both for encryption and for computing MIC. Using the same key for two different purposes is normally considered questionable. However, the construction of counter mode and CBC-MAC IV from packet sequence counter provides the key separation needed to use the same key both for encryption and the MIC. In [18] it has been found that it not possible to extract much parallelism

when computing CBC-MAC, but it could become a potential drawback in the future.

(v) In CCMP it is possible to begin encryption before completing calculation of the message authentication code (MIC tag). However, the negative side of this characteristic is that decryption should be completed before verifying message authentication code.

(vi) Concerning vulnerability of its Nonce construction, in the paper [23] it is described that the initial counter value used in the CCMP of 802.11 Wireless LANs can be predicted. Since the nonce value can be pre-computed, the only thing required to predict the counter value is length of payload. The length of the payload can be obtained through a priori information e.g. 802.11 maximum Payload length is 2296 bytes (2312 bytes total payload length – 8 bytes MIC – 8 bytes CCMP Header) and if the data is more than maximum length of Payload then MSDU is fragmented into MPDUs. If larger data than the maximum payload length is to be transmitted, then the first fragment's (MPDU) Payload length will be 2296 bytes. In [24], it is iterated that if initial counter value is predictable, then attacks using pre-computation can be used to lower the security level of AES-128-CM below the recommended strength for block ciphers.

5.3 AES-CCMP Optimization Avenues

Our main objective is to devise methods to conserve resources, namely the precious battery power of devices in wireless LANs or Ad Hoc networks, by implementing a more efficient AES-CCMP. We focus on two aspects of the protocol: First, we reduce possible recursive function redundancies in the AES-CCMP structure and secondly, we increase the decryption or de-encapsulation efficiency of CCMP under error-prone or hostile wireless channel conditions in order to avoid unnecessary wastage of processing time. We therefore adopt a fusion of the CTR mode and CBC-MAC, to form a hybrid CCMP design which is more computationally efficient without compromising security services and perform early MIC validation filtering before decryption of any packet.

6. AES-CCMP Design with CBC-MAC Variant

For CCMP, if the AES block cipher encrypted 128-bits plaintext blocks at a rate of ***r*** processor cycles per byte, then the joint sequential process of Counter mode encryption plus CBC-MAC authentication would require at least ***2r*** cycles per byte. We propose a hybrid Counter-Mode Block Chaining MAC (CMBC-MAC) mechanism, as shown in Figure **2**, which can lower the combined authenticated-encryption of CCMP down to nearly ***r*** cycles per byte processing. The resulting construction is provably secure and has peak efficiency and speed close to the sum of counter mode encryption.

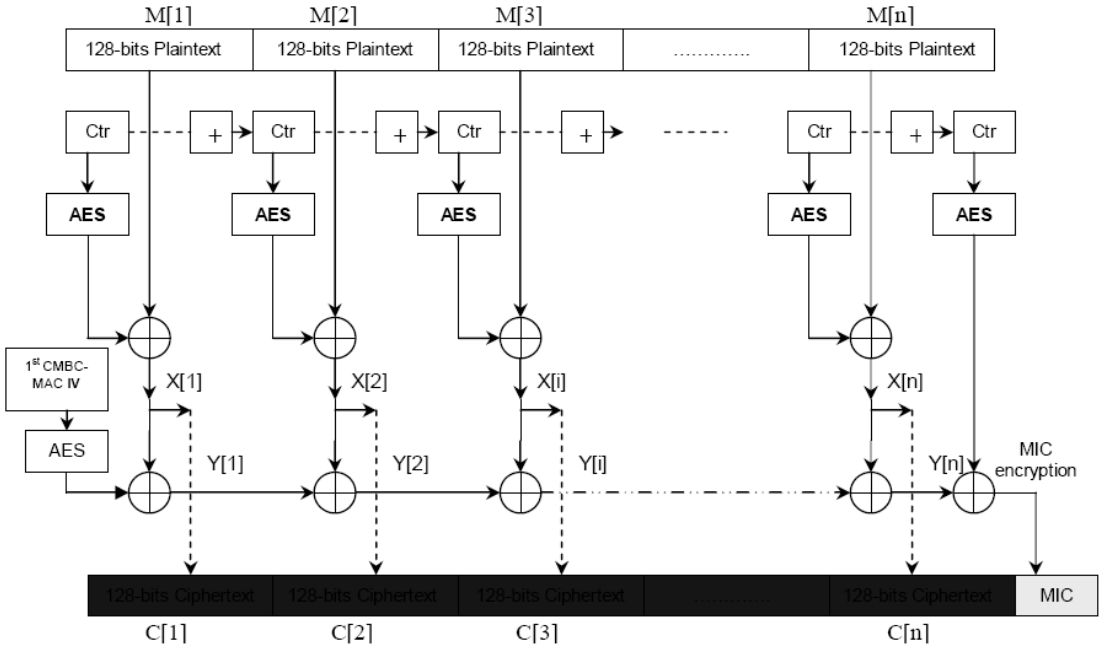

Figure 2: CCMP design with merged CTR data Encryption and lightweight CBC-MAC

6.1 CCMP with CMBC-MAC Operation

Let message M = M[1], M[2], M[3], … M[n] be a sequence of 128-bit plaintext block. A nonce ***N*** of ***L*** bits is chosen to derive both the re-initial value of CMBC-MAC and the Counter encrypted blocks. Using X[i] and Y[i] as dummy intermediate results in **Figure 2** algorithm execution path, we suppose:

X[i] = $\mathbf{M[i] \oplus E_K(Ctr + i)}$, *for i = 1, 2, ..., n.*

Y[1] = $\mathbf{X[1] \oplus E_K(IV)}$

Y[i] = $\mathbf{X[i] \oplus Y[i-1]}$, *for i= 2, 3, ..., n.*

Finally, Encrypted MIC Tag**,** T = $\mathbf{Y[n] \oplus E_K(Ctr + n+1)}$

The Counter Mode Block Chaining MAC operation, illustrated in **Figure 2**, is summarized as follows:

1. Initialization: A unique counter block (Ctr) is encrypted with AES cipher using key K_c.
2. The result (1) is XORed with the first message block (M[1])
3. The first CBC-MAC block IV is encrypted with AES cipher using key K_i and XORed with the previous output (2) to get MAC_1
4. An incremented Ctr is encrypted with AES and XORed with the second message block (M[2]), followed by XOR with MAC_1 to give MAC_2.
5. The next incremented Ctr is encrypted with AES and XORed with the third message (M[3]) block, followed by XOR with MAC_2 to give MAC_3, and so on.
6. The n^{th} incremented Ctr is encrypted with AES and XORed with the n^{th} message block (M[n]), followed by XOR with MAC_{n-1} to give MAC_n.
7. Finally, the $(n+1)^{th}$ incremented Ctr is encrypted with AES and XORed with MAC_n to give the Counter mode encrypted MAC_n.

As a solution to authenticating encryption, CMBC-MAC encrypts a message and authenticates the ciphertext, using either the single key or separate keys for each operation. The complexity of CCMP is affected directly by the AES block cipher call for computing message authentication code. In the analysis by Rogaway et al. [3], it is concluded that the authentication tag of the raw CBC-MAC is computed by a method that is seen connected to the entire CCMP mode and there is no alternative to use another more secure MAC. Reducing the number of times the AES block cipher is used in the process by merging the CTR mode data encryption with CBC-MAC variant gives a lower complexity and a secure AES-CCMP design that inherits all of CCMP positive features in addition to lower computational cost. This CBC-MAC variant or Counter Mode Block Chaining-MAC (CMBC-MAC) is economical in the sense that it is embedded in the CTR mode procedure and it uses the output of the CTR mode process to construct the MIC, hence eliminating the need to reprocess the plaintext separately during MIC computation. The advantage of CMBC-MAC is also significant during decryption process, as we can see it allows message integrity check prior to decrypting the whole cipher message. At the receiver, the MIC is recomputed based on the ***n*** ciphertext blocks received using the 1^{st} CBC-MAC IV block and the block cipher key. This recomputed MIC is encrypted with the ***(n + 1)*** incremented counter (Ctr) and compared with the received encrypted MIC value to check authenticity of received ciphertext, by extension verifying the integrity of plaintext message from sender. The early quick MIC check avoids decrypting corrupted ciphertext messages, hence saving considerable battery power when operating in noisy or interference-prone wireless network.

6.2 Security Analysis

Counter mode is known to be secure against chosen-plaintext attacks [26], as the ciphertext hides all partial information about plaintext, even if some a priori information about the plaintext is know. But, security is valid under the assumption that the primary block cipher, i.e. in this case AES, is a pseudo-random function family and that a unique counter value is used at every step. Hence, CCMP is secure as long as the triple (key K_i, CBC-MAC IV) is not reused and a pair (key K_c, counter value (Ctr)) is not reused. Moreover, the inclusion of MAC addresses in the IV prevent the sender and receiver from applying the same (Key, CBC-MAC IV) pair. It is also confirmed that security of CCMP with embedded CMBC-MAC is guaranteed by the security analysis provided in [16] because the counter mode encryption scheme parameters are not modified, but rather showed how the counter mode can be also extended and merged for authentication using a chaining procedure in the new approach. Considering, the following MAC expression deduced from enhanced CCMP with CMBC-MAC, we obtain:

$\mathbf{C[1]} = M[1] \oplus E_K(Ctr)$

$\mathbf{MAC[1]} = C[1] \oplus E_K(IV) = M[1] \oplus E_K(Ctr) \oplus E_K(IV)$

$\mathbf{C[2]} = M[2] \oplus E_K(Ctr + 1)$

$\mathbf{MAC[2]} = MAC[1] \oplus C[2]$

$\mathbf{MAC\ [2]} = M[1] \oplus E_K(Ctr) \oplus E_K(IV) \oplus M[2] \oplus E_K(Ctr + 1)$

$\mathbf{MAC\ [n]} = M[1] \oplus M[2] \oplus \ldots \oplus M[n] \oplus E_K(IV) \oplus E_K(Ctr) \oplus E_K(Ctr + 1) \oplus \ldots \oplus E_K(Ctr + n)$

where, each symbol in the above expressions has same meaning as used in previous explanation in this paper.

The MAC expression also suggests that the security of the composite AES-CCMP with CMBC-MAC is based on more than just the security of the individual components. The cross-relationship and dependencies between CTR mode and CMBC-MAC also influence the actual security composition. The initialization phase has been assessed with respect to the Strict Avalanche Criterion (SAC) [25]. This has been done not only for the key sensitivity but also for the first CBC-MAC IV sensitivity. If the full AES encryption rounds (N_r) are performed during the initialization, the security level is assumed to be so high, that only exhaustive search can find the correct key or IV value from known plaintext / cipher text pairs. Note that for a key of size **n** bits, require 2^n operations for key searching attack. Hence, to provide an equivalent security of MAC to the cipher, we would theoretically require a MAC size twice as long as the cipher key text. However, we cannot afford this extra MAC overhead as the performance will suffer in terms of lower throughput, more processing, higher memory and transmission cost.

6.3 Complexity Analysis of AES-CCMP with Hybrid CMBC-MAC

Considering a message $M = M_1, M_2, ..., M_n$, where n is the number of 128-bit plaintext blocks, then the counter mode is described as:

Initially, counter (Ctr) ← 0
For each message $M = M_1, M_2, ..., M_n$
Initial-counter ← counter (Ctr)
For i = 1 to n do
$\mathbf{C_i \leftarrow M_i \oplus E_K (Ctr), Ctr \leftarrow Ctr + 1}$

The total number of processing cycles to encrypt n blocks of 128-bit plaintext is:

$\mathbf{T'_{CTR} = n^* (1\ XOR + 1 * T_{AES\text{-}ENCRYPT})}$

Therefore, the total number of processing cycles to construct the MIC and encrypt both the message and its MIC for the improved CCMP design with hybrid CMBC-MAC is:

Improved $\mathbf{T_{AES\text{-}CCM}} = (\boldsymbol{n + 2})^* \mathbf{T_{AES\text{-}ENCRYPT}} + (2\boldsymbol{n}+1)^*(\text{XOR})$,

Improved $\mathbf{T_{AES\text{-}CCM}} = (\boldsymbol{n + 2})^* \mathbf{T_{AES\text{-}ENCRYPT}} + (2\boldsymbol{n}+1)^*(8N_bT_{and} + 4N_bT_{or})$,

Improved $\mathbf{T_{AES\text{-}CCM}} = (\boldsymbol{n + 2})^*\{(46N_b\ N_r - 30N_b)\ T_{and} + [31N_b\ N_r + 12(N_r - 1) - 20N_b]\ T_{or} + [64N_b\ N_r + 96(N_r - 1) - 61\ N_b]\ T_{shift}\ \} + (\boldsymbol{2n + 1})^*(8N_bT_{and} + 4N_bT_{or})$,

Conversely, for comparison, recall that in the case of generic CCMP,

Original $\mathbf{T_{AES\text{-}CCM}} = (\boldsymbol{2n + 2})^* \mathbf{T_{AES\text{-}ENCRYPT}} + (\boldsymbol{2n+1})^*(8N_bT_{and} + 4N_bT_{or})$,

Original $\mathbf{T_{AES\text{-}CCM}} = (\boldsymbol{2n + 2})^*\{(46N_b\ N_r - 30N_b)\ T_{and} + [31N_b\ N_r + 12(N_r - 1) - 20N_b]\ T_{or} + [64N_b\ N_r + 96(N_r - 1) - 61\ N_b]\ T_{shift}\ \} + (\boldsymbol{2n+1})^*(8N_bT_{and} + 4N_bT_{or})$,

The difference in computational complexity between **Original** $T_{AES\text{-}CCM}$ and **improved** $T_{AES\text{-}CCM}$ variant is $(n)^*\{(46N_b\ N_r - 30N_b)\ T_{and} + [31N_b\ N_r + 12(N_r - 1) - 20N_b]\ T_{or} + [64N_b\ N_r + 96(N_r - 1) - 61\ N_b]\ T_{shift}\ \}$

From tabulated results, **Table 2**, it is noted that the overall speedup factor increases when the message size increases, but then almost stabilizes for large message size, typically above n=30. This is in accordance with Amdahl's principle because the fraction of times the CMBC-MAC for authentication component being used has an upper bound, which is 50%.

Number of 128 bit blocks (n) & Complexity Component		AES-CCMP with CBC-MAC Computational Complexity Key size 128 bits			AES-CCMP with CMBC-MAC Computational Complexity Key size 128 bits			Overall Speedup factor		
		N_r = 6 rd	N_r = 8 rd	N_r = 10 rd	N_r = 6 rd	N_r = 8 rd	N_r = 10 rd	N_r = 6 rd	N_r = 8 rd	N_r = 10 rd
n = 1	$T_{AES\text{-}CCM}$	14064	19440	24816	10584	14616	18648	1.33	1.33	1.33
2	$T_{AES\text{-}CCM}$	21120	29184	37248	14160	19536	24912	1.49	1.49	1.50
3	$T_{AES\text{-}CCM}$	28176	38928	49680	17736	24456	31176	1.59	1.59	1.59
10	$T_{AES\text{-}CCM}$	77568	107136	136704	42768	58896	75024	1.81	1.82	1.82
20	$T_{AES\text{-}CCM}$	148128	204576	261024	78528	108096	137664	1.89	1.89	1.90
30	$T_{AES\text{-}CCM}$	218688	302016	385344	114288	157296	200304	1.91	1.92	1.92
40	$T_{AES\text{-}CCM}$	289248	399456	509664	150048	206496	262944	1.93	1.93	1.94

Table 2: Theoretical Complexity of AES-CCMP with CMBC-MAC

From the AES-CCMP mathematical computational analysis, we note that the increase in amount of processing cycles over payload/message size is linear as shown in **Figure 3**, and this outcome is inline with previous research that were carried out by experimental approach in [22] [10]. The number of processing cycles for CCMP also

varies linearly with the number of rounds of the core encryption, as illustrated in **Figure 3**. The energy expenditure is a function of number of computations and can be easily calculated by multiplying the number of computations of AES-CCMP for a given AES key size times the energy consumed in Joules by a single computation.

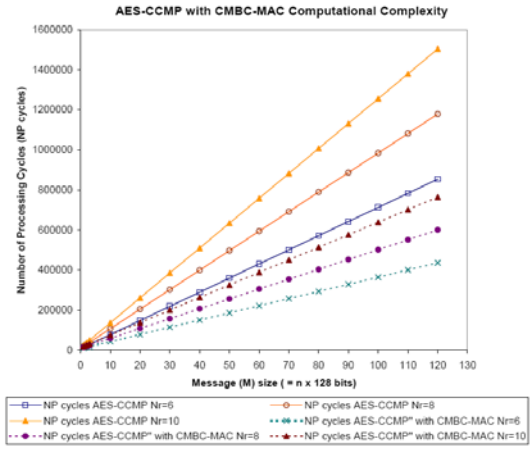

Figure 3: AES-CCMP complexity comparison

From **Figure 4**, we note that AES-CCMP with CMBC-MAC has a lower gradient for the rate of change of processing cycles (indirectly, the power) with respect to the number of rounds is smaller compared to AES-CCMP with CBC-MAC. With a fixed key size 128-bit, it is more efficient to encrypt larger messages since the security overhead is smaller. The execution time of the security protocol depends on the number of operations executed and the computational power required by a security protocol is determined by measuring the time required to complete the all security operations. The amount of energy consumed by the cryptographic function is directly proportional to the amount of computation. In general, by doing more computations in a cryptographic algorithm it results in stronger security level, assuming the extra computations for a specific cipher can come from increasing the key size or number of rounds parameters. However, for a fixed cipher parameter set, AES-CCMP with CMBC-MAC gives approximately 25% reduction in number of computations compared to AES-CCMP with CBC-MAC, but this is a result of the new CMBC-MAC design that optimizes the recursive call of AES cipher function for combined authentication and encryption process. This result is shown in **Figure 5**.

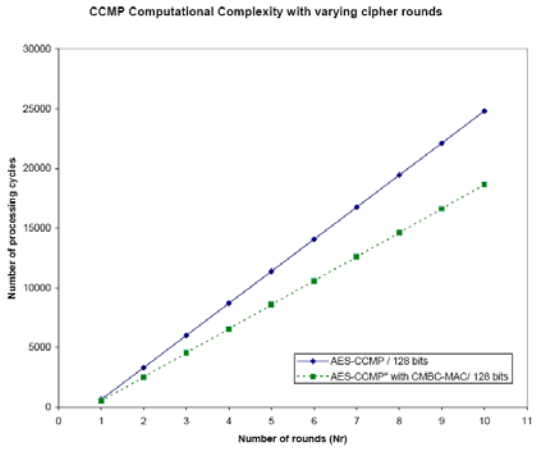

Figure 4: Relationship between AES-CCMP complexity and different cipher rounds

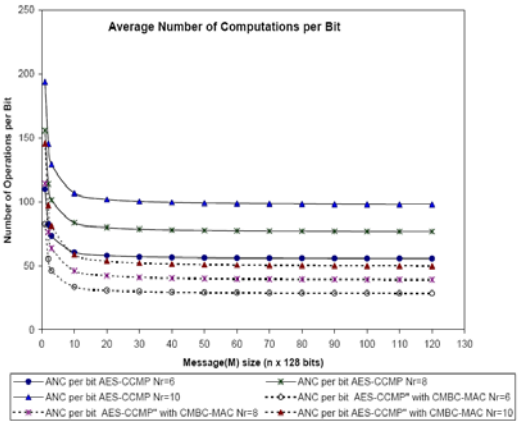

Figure 5: Average number of computation (ANC) per bit encrypted for both CCMPs

A logical statement is that lower computational complexity security component has faster execution time than more complex security modules. In fact, Amdahl's law [28] states that the performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used. A program X is N% faster than program Y if, the ratio CPU Execution time of x / CPU Execution time of Y is equal (1 + N/100). Then, accordingly to **Amdahl's Law:**

Overall Speedup = $1 / [(1-f) + f/s]$

where, f is fraction of a program that is enhanced and s is speedup of the enhanced portion.

For example, prior to optimization, for AES cipher with N_r = 6 rounds, key size is 128 bits, $T_{AES\text{-}CCM}$ = 14064 processing cycles, $T_{CBC\text{-}MAC}$ = 7008 processing cycles, and

$\mathbf{T_{CTR}}$ = 7056 processing cycles. We infer that a CCMP module (hardware or software) consists of 49.83% of the time spent by CBC-MAC authentication element and 50.17% time expend for CTR mode encryption part. Next, with CMBC-MAC optimization, improved $\mathbf{T_{AES\text{-}CCM}}$ = 10584 processing cycles. The overall Speed-up is thus 1.33 due to performance of authentication element being enhanced by a factor of 1.5, i.e. $\boldsymbol{s}$ = 1.5.

Usually, the relationship between efficiency or performance cost and security strength is that, the higher security we want, the less efficient is the method providing that security level. Each operation executed by the processor takes a finite time and some operations are more complex and more resource demanding than others. The ideal case is to use simple operations, but to combine these operations in a way to achieve a high level of security. Moreover, CMBC-MAC can be parallelized similar to Parallelizable MAC (PMAC) that was introduced by Black and Rogaway [9] where all the incoming data blocks are passed through block ciphers in parallel, essentially reducing the processing time further. Another way to gain in speed, is to interleave certain repetitive sequential functions in CCMP, but in our work we do not investigate the impact of interleaving on AES-CCMP with hybrid CMBC-MAC.

The proposed hybrid CMBC-MAC has a valuable energy-saving characteristic as it allows pre-decryption MIC validation or MIC filtered decryption (MFD). Only the received MIC tag needs to be extracted and decrypted for this purpose which is directly done in Counter mode. The CTR mode already allows decryption with "random access". It means that the receiver need not decrypt all the *C[n]* ciphertext blocks before re-computing the MIC for checking message authentication. So, under error channel conditions, any received packet is first filtered with MIC checking, and only then the whole message with valid MIC is decrypted with the cipher.

Lastly, assuming the number of encrypted packets is **P** and the number of packets discard by MIC filter at receiver is **d,** then, the number of packet decrypted by AES-CCMP with CMBC-MAC is (P – d). Further, if a CBC-MAC MIC module consumes μ Joule/packet and CTR mode takes β Joule/packet, we deduce the original CCMP decryption process energy consumption will be , $\mathbf{E = P(\mu) + P(\beta) = P(\mu + \beta)}$. However, comparatively for CCMP with hybrid CMBC-MAC (i.e. MIC filtered decryption), the decryption energy is $\mathbf{E' = P(\mu') + (P\text{-}d)\beta}$, where μ' is the energy cost associated with CMBC-MAC part and μ > μ' . The expected energy saving is $\mathbf{P(\mu\text{-}\mu') + (d)\beta}$ Joule. Hence, the MFD technique is efficient in the sense that CPU cycles and computation energy are not wasted on fully decrypting corrupted received packets.

7. Conclusion

In this paper, a concise theoretical analysis of the complexity and computational cost of AES-CCMP has been presented which can be used to study its energy consumption. CCMP is confirmed to be computationally expensive with CBC-MAC authentication element and Counter mode encryption part contributing equally to the overall computational cost. We proposed a minimalist modified AES-CCMP design with hybrid CMBC-MAC to demonstrate the potential for high energy saving at both the encryption side and decryption side. The encrypt-then-authenticate composition first encrypts with AES-CTR, and then computes MAC on the ciphertext with separate keys for encryption and authentication. The resource-saving AES-CCMP with CMBC-MAC enables faster message authentication and eliminates AES-CTR-mode decryption of corrupted packets with invalid MIC received from error-prone wireless channel. As part of our ongoing work, we are investigating better methods to optimize IEEE 802.11i security components from a resource-efficient implementation perspective, both in software and hardware, to be attuned with growing number of heterogeneous battery limited wireless devices.

References

[1] Institute of Electrical and Electronics Engineers, Inc., IEEE Std. 802.11i-2004, Amendment to Standard for Telecommunications and Information Exchange Between Systems – LAN/MAN Specific Requirements – Part 11: "Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC) Security Enhancements", July, 2004.

[2] D. Whiting, R. Housley, and N. Ferguson. "IEEE 802.11-02/001r2: AES Encryption and Authentication Using CTR Mode and CBC-MAC", March 2002.

[3] P. Rogaway and D. Wagner, "A Critique of CCM" Eprint cryptology archive, February 2003. Available at http://eprint.iacr.org.

[4] P. Prasithsangaree and P. Krishnamurthy, "On a framework for energy efficient security protocols in wireless networks", Elsevier Computer Communications, 27(17), pp. 1716 – 1729, 2004.

[5] M. Razvi Doomun and KMS Soyjaudah, "Analytical Comparison of Cryptographic Techniques for Resource-Constrained Wireless Security", International Journal of Network Security, 2008. http://ijns.femto.com.tw:8088/prepare_to_publish.jsp

[6] N. Potlapally, S. Ravi, A. Raghunathan and N. K. Jha, " A Study of the energy consumption characteristics of cryptographic algorithms and security protocols" IEEE Transactions on Mobile Computing, vol. 5, No. 2, February 2006.

[7] Matthew Gast and Loukides Mike, 802.11 Wireless Networks: The Definitive Guide By, Ed. Safari Tech Books Online Published by O'Reilly (ISBN 0596100523, 9780596100520), pp. 496, 2005.

[8] S. Michel and K. Srinivasan. “State based key hop protocol: A lightweight security protocol for wireless networks”, In Proceedings of 1st ACM International Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor and Ubiquitous Network, PE-WASUN 2004, pp. 112-118, October 2004.
[9] J. Black and P. Rogaway. “A Block-Cipher Mode of operations for Parallelizable Message Authentication”. Advances in Cryptology – Eurocrypt 2002, Lecture Notes in Computer Science, Volume 2332, pp 384-397, Springer-Verlag, 2002.
[10] R. Chandramouli, S. Bapatla, K.P. Subbalakshmi, and R.N. Uma, “Battery Power-Aware Encryption” ACM Transactions on Information ad System Security, Vol. 9, No. 2, pp. 162-180, May 2006.
[11] J. Grobschadl, S. Tillich, and C. Rechberger, “Energy evaluation of software implementations of block ciphers under memory constraints” In Proceedings of the 10th Conference on Design, Automation and Test in Europe, pp. 1110–1115, 2007.
[12] J. Daemen and V. Rijmen, “The design of Rijndael: The Advanced Encryption Standard”, Springer-Verlag, 2002.
[13] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, “ Security in Mobile Ad Hoc Networks: Challenges and Solutions” IEEE Wireless Communications, 11(1), pp. 38-47. 2004
[14] J.F. Kurose and K.W. Ross, “Computer Networking - A Top-Down Approach Featuring the Internet” Addison Wesley, 2002.
[15] J Walker, “802.11 Security Series. Part III: AES-based Encapsulations of 802.11 Data”, Network Security Architect, Platform Networking Group Intel Corporation, 2005.
[16] S. Vaudenay, “Security Flaws Induced by CBC padding,” Proceeding International Conference Theory and Applications of Cryptographic Techniques”, Springer-Verlag, pp.534-546, 2002.
[17] H. Krawczyk “The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?)” Proceedings 21st Annual International Cryptology Conference Advances in Cryptology, CRYPTO 2001, pp. 310 - 331, Springer Verlag, 2001.
[18] J. Black and P.Rogaway, “A Suggestion for Handling Arbitrary-Length Messages with the CBC MAC”. NIST Second Modes of Operation Workshop, August 2001.
[19] M. Bellare, J. Killian, and P. Rogaway, “The Security of the Cipher Block Chaining Message Authentication Code”, In Advances in Cryptology, CRYPTO ’94 Vol. 839 of Lecture Notes in Computer Science, pp. 341 – 358, Springer 1994.
[20] M. Razvi Doomun, KMS Soyjaudah, and D. Bundhoo, “Energy Consumption and Computational Analysis of Rijndael-AES”, Third IEEE International Conference in Central Asia on Internet The Next Generation of Mobile, Wireless and Optical Communications Networks, ICI 2007, 2007.
[21] F. Granelli and G. Boato, “A novel methodology for analysis of the computational complexity of block ciphers: Rijndael, Camellia and Shacal-2 compared”, In Third Conference on Security and Network Architectures (SAR’04), 2004.
[22] Y. Xiao, H. Chen, B. Sun, R. Wang and S. Sethi. “MAC Security and Security Overhead Analysis in IEEE 802.15.4 Wireless Sensor Networks” EURASIP Journal on Wireless Communication and Networking, pp.1-12. Volume 2006.
[23] M. Junaid , M. Mufti, and M.Umar Ilyas, “Vulnerabilities of IEEE 802.11i Wireless LAN CCMP Protocol” Proceedings Of World Academy Of Science, Engineering And Technology Vol. 11, pp. 228-233, 2006.
[24] David A. McGrew, “Counter Mode Security: Analysis and Recommendations”, Cisco Systems, November, 2002.
[25] M. Zeghid, M. Machhout, L. Khriji, A. Baganne, and R. Tourki “A Modified AES Based Algorithm for Image Encryption”, International Journal of Computer Science and Engineering, Vol. 1 No. 1, pp. 70 – 75. 2007.
[26] M. Dworkin, “Recommendation for Block Cipher Modes of Operations: Methods and Techniques”, Special publication 800-38B, US National Institute Standards and Technology, 2005.
[27] A. Samiah, A. Aziz, and N. Ikram, “An Efficient Software Implementation of AES-CCM for IEEE 802.11i Wireless Standard”, 31st Annual International Computer Software and Applications Conference, COMPSAC 2007, Vol. 2, pp. 689-694, 2007
[28] M. Annaratone, “MPPs, Amdahl's law, and comparing computers” Fourth Symposium on the Frontiers of Massively Parallel Computation, McLean, VA, USA Publication, pp. 465-470. 1992.

M. Razvi Doomun holds a B.Eng (Hons.) in Electronic and Communication Engineering from University of Mauritius and an MSc in Multimedia Communications from University of Surrey, UK in 2002. He is presently pursuing his PhD in wireless network security.

K.M. Sunjiv Soyjaudah received his BSc. (Hons.) degree in Physics from Queen Mary College, University of London in 1982, his M.Sc. degree in digital electronics from King’s College, University of London in 1991 and his PhD degree from University of Mauritius in 1998. He is presently Professor at the department of electrical and electronic engineering of the University of Mauritius. His interests are communication theory, cryptography and security.