FPGA Implementation of High Speed AES Based Authentication Algorithm

P. Vijayakumar[1], P.Lalit Kishore [2], K.Venkata Diwakar Reddy[3] ,Srujan Reddy[4], Rajashree R[5], Ananiah Durai[6]

[1,6]Associate Professor, Vellore Institute of Technology, Chennai, Tamilnadu, India
[2,3,4]Undergraduate Students, Vellore Institute of Technology, Chennai, Tamilnadu, India
[4]Research Scholar, Vellore Institute of Technology, Chennai, Tamilnadu, India

Abstract: Encryption and decryption is accomplished by means of software or hardware, has been in continuous usage since the 2000s for data encryption using any communication medium. This is primarily achieved so as to ensure that no personal material is revealed to unwelcome audiences. Block ciphers are critical for data transmission and also provide flexibility for the implementation of numerous key configurations and variable speeds. Rjindael's algorithm has been used as the Advanced Encryption Standard (AES) and it will remain the most popular algorithm used for data security in encryption and decryption and is used electronically for protected data transmission. Today, AES is the symmetric algorithm most widely used and does so through translation of user text by 'hex' values and then through the user-specified key execute specific steps. Although one of the most common algorithms, AES has its own inconveniences like a very simple algebraic structure, the same sequence of predetermined steps and are not very easy to use software. A modern authentication algorithm is introduced based on the Advanced Encryption Standard (AES) algorithmic program and Secured Hash Algorithm (SHA) algorithms to encrypt information for encrypted communication with improved security features. SHA shall be applied at the same time as the AES mechanism for secrecy, reliability and honesty checks. The core concept of algorithmic application is to achieve a high degree of data protection by integrating SHA and AES algorithmic programs in software and hardware.Both transmitter and receiver sections are included in the proposed model to safely send and receive data. It's been developed using Xilinx ISE 14.2, so that the parameters of the proposed algorithmic program can be compared with various different FPGA's (Field Programmable Gate Array) and then further comparison of parameters may lead to satisfactory results.

Keywords: Advanced Encryption Standard; Symmetric Key Cryptography; Block ciphers; Secure Hash Algorithm; Field Programmable Gate Array; Authentication.

I. INTRODUCTION

Design and deployment of an AES algorithm-based authentication that helps manage integrity checks between the sender and the recipient is proposed. The primary goal of the basic encryption and decryption approach was to reach a higher degree of data security with the aid of AES. The next step i.e. integrity test or authentication process between sender and receiver further increases protection and decreases the risks of data manipulation to almost null and, in the process of authentication between sender and receiver, a SHA algorithm has been used that uses padded input to produce hash codes. On the side of the transmitter, first AES encryption is done to the input message with the help of user given key to produce a cipher text which is illegible to a human. At the same time, an input message or plain text is also provided as an input to SHA 1 that generates a Message Authentication Code (MAC). The MAC which is given at the transmitter side can be termed as MAC_T. The output of SHA 1 and AES encryption is then appended to form 288 bits which are then sent to the receiver side. This 288 bit input is again separated into MAC and cipher text at the receiver side, and cipher text is decrypted into plaintext and this plaintext is again used to produce MAC code at the receiver side termed as MAC_R. Thus, if the hash code of the plaintext at the receiver side i.e. MAC_R is equal to the hash code obtained at transmitted side i.e. $MAC_{T,}$ then the plaintext has been authenticated and proves that no data-tampering has been done [1]. The encryption / decryption and validity verification process can be done at the same time with the aid of symmetric key cryptographic algorithms, so AES has been selected [2]. This architecture involves numerous security-related apps, such as ATM banking, online HTTPs, etc. In addition to the fore-mentioned applications and benefits, the AES-based authentication algorithm remains difficult for future attackers. Nowadays, much of the correspondence is done based on electronics and data-protection plays a very important role in communication [3]. Due to this, also an increasing need for numerous new ideas to avoid the loss of knowledge from numerous attacks has born [11].

Data transferred in an electronic way over the internet for example; messages over online messengers, emails etc. are vulnerable to attacks. The main purpose of the 'High-Speed AES Authentication Algorithm' is to pass data from the sender to the recipient by first utilizing a cryptographic algorithm that is AES Algorithmic program i.e. the user precedently provides input or plain text that is translated to 128 bits utilizing different algorithms and the user must then insert a key that is also translated to 128 bits to acquire a cipher text. The aim of this dissertation is to provide additional protection for the information transmitted from the transmitter to the receiver [13]. For the integrity check process, we require the production of hash codes, and for this process, due to time constraints, we have selected SHA-1.Simultaneously, while sending the plaintext for AES encryption, the user of the algorithm also sends the plaintext to the SHA as input, to receive a MAC code at the transmitter part (MAC_T). Thus, collectively, the user obtains a 128-bit cipher text and also a hash code (MAC_T) as output from the transmitter part of the algorithm [1].The cipher text obtained in the encryption step is given as an input to AES decryption. This step provides plaintext as the output. Since, the use of AES is employed, it is highly impossible for any unverified or malignant middleman to crack the cipher text and obtain the data. Now, the plaintext obtained at the end of decryption at the receiver side is again given as input to SHA1 algorithmic program to obtain a MAC code at the transmitter part (MAC_R). The obtained has codes in the transmitter and receiver part of the algorithm i.e. MAC_T and MAC_R are then compared and checked if equal.The reason the hash codes (MACs) are compared is because of integrity check between the sender and the recipient. Any intruder or opponent may tamper with data sent during transmission after encryption, and there is a loss of information due to this, because the receiver may not know if the recipient has received the correct information from the sender. However, since there is a comparison of hash codes (MACs) in the proposed algorithm, if there is any data tampering during transmission, the hash codes will not be the same. Therefore, the recipient must realize for sure that the data has been tampered with [1].

The availability of high-speed digital hardware has created the implementation of top-quality cryptographical devices attainable, in order that anyone can easily notice their application in industrial fields like ATMs and laptop terminals. In turn, such applications produce a requirement for few varieties of cryptographical algorithms that will increase the safety even more than the already existing techniques. It is necessary to secure the information’s secret, during this context and thus want to own a cryptosystem that is demonstrably secure and it ought to provide a whole lot of security [6]. Cryptography can be used to offer confidentiality using secret writing ways and might conjointly offer information integrity and authentication. Phishing and information thievery are a typical downside lately because of the immense impact of web and also the demand of a typical media or channel to share information. Due to this, there's the matter of playing few secret writing algorithms whereas conjointly keeping the information values hidden from curious parties. Thus, it is determined to figure on a project associated with secret writing that provides a really high level of information encryption, in order that the likelihood of any information thievery is alleviated. This can be achieved with the assistance of authentication between the transmitter and receiver with the assistance and with the usage of AES algorithmic program [7].

This paper has consists of seven different chapters and it has been organized in such a way that each chapter deals with the major concepts and contributions involved in the research work. The first chapter deals with the introduction of ‘High-speed AES based authentication algorithm’ work and the objective of the dissertation work. The second chapter consists of the Literature Survey of various research papers and websites which helped throughout the formulation of this dissertation.The third chapter deals with the Mathematical background of AES and SHA algorithms with various examples of the same and also an overview of the project. This fourth chapter presents the proposed efficient high-speed AES based authentication algorithm. The fifth chapter presents the Hardware and FPGA Implementation of the proposed algorithm with the inclusion of synthesis options. This sixth chapter deals with the results obtained and finally ends with the conclusion.The chapter seven includes the project's end.

2. LITERATURE SURVEY

Ali Akbar Pammu [1] et al. have developed an AES algorithmic system that uses a CBC mac or CCM mode counter to obtain a very high degree of data protection, and this has been done on a multi-core / dual-read processor[1]. In addition, a mechanism of modifying the key for AddKeyRound is continuously updated in reference to completed or time-based rounds for more defensive capabilities against pattern-based assaults. There is also a very secure and complex implementation of AES-CCM method that is the key which is provided changes with the time interval provided. This removes the possibility of key-tamping from any third party and nullifies the major protection vulnerability of the AES algorithmic system. The central and important conclusion that can be taken from this journal paper is that this is a research work in which AES with integrity test algorithms has been used to gain more reliability. In fact, the whole application has been built on a multi-core processor.

Mr. Adnan Mohsin [14] et al., give a very informative and detailed explanation regarding 128-bit encryption and decryption of AES. Then also, the design or algorithm was implemented with the help VHDL software language then simulated by using Xilinx ISE [14]. The Keys were generated for all rounds before the data goes through the rounds. The S-box and Inv-Sbox array elements were already stored in the memory in order to be ready for substitute byte. Using pipeline technologies to increase performance. The parallel architecture has also been used. This provides a detailed explanation of the various things that were finally implemented on an FPGA, and speeds, LUT's and cores were written in a table format. Specific implementations for both the AES encryption method and the AES decryption method have since been introduced. The results obtained in the design summary were also compared with 5 different types of FPGA's with the different parameters and recorded the frequency and slices used for each device.

J.Balamurugan [17] et al. give a very informative and detailed explanation regarding a type of AES algorithm which is cost effective. In addition, also, a different attack which may be plausible on AES was briefed about [17]. The author makes it lucid regarding the Advanced Encryption Standard (AES) algorithm and this algorithmic method was applied in this paper using VHDL. Both the encryption and decryption algorithms used VHDL in the same way. It can be concluded that the algorithmic program consists of three main parts: Cipher, Inverse Cipher and Key Expansion, as indicated by the paper. Development and development of lightweight AES key which is used for encryption which is ideal for cost effective applications. The key thing was price stability and also power-efficient applications plus AES hardware implementation for a few specific FPGA types.

Liakot Ali [5] et al., have proposed the design which is useful to a very high degree and also can lead to further improved security of the already secure AES architecture. The authors have proposed an architecture that can be resistant to all cryptanalytic attacks on AES. Everything has been done, while maintaining high computing speed [5]. The proposed work tweaks the AES algorithm to reduce algebraic operations, which in turn increases the speed of architecture and also reduces latency. The key software used for Advanced Encryption Protocol is rendered or encoded with the aid of the Verilog language and is then dumped on the FPGA board to extract the results of the hardware. The efficiency of the proposed architecture has also been contrasted and tabulated with the research of several other previous dissertations in order to evaluate speed, throughput and latency, and the authors prove that the proposed architecture is stronger than the previous models. Ultra-high performance and AES processing speed can be reached and the delay can also be reduced.

Hassen Mestiri [10] et al., wrote about the numerous cryptanalytic attacks and shortcomings in achieving and implementing optimal and effective AES implementation, as well as how AES conventionally overcomes these problems [10]. The authors built a way to further improve AES 'data protection capabilities by improving the architecture to hold back against form of fault-injection attack. The authors also debated numerous solutions in detail. In embedded applications where the applications serve a specific purpose, it is of utmost importance that the system does not fail and AES has been the standard for data protection in these devices for many years. However, cryptanalytic attacks of inducing faults reduce security and may lead to serious cases of data leakage that may, in some cases, be considered to be sensitive. Thus, the authors of the particular thesis provide new ways to detect fault injections by varying or modifying the substation step of the SBox. The proposed algorithmic system allows AES safer to known fault or threats which can be avoided by making adjustments to the implementation process to the software. To this end, the transformations used in AES were broken into two sections with an intermediate stage between them in order to improve the protection and the possibility of cryptanalysis. Also, the authors have also shown that with the help of this implementation, the fault detection rate was increased by many fold in the architecture.

Harshali Zodpe and Ashok Sapkal [7] have written in a detailed format about the AES algorithm architecture, which can improve the security features of the standard, and this can be achieved by a new type of Sbox computation variation [7]. Through an in-depth analysis of AES, it is stated that the safety factor of this standard lies in the SBox step and, therefore, a different approach to the SBox step is proposed here. At the end of the day, the main inferences that could be drawn from this are that the modification of the existing design resulted in better speed. The main piece of knowledge from this paper is that different S-Box operations can increase or decrease the security of the Advanced Encryption Standard algorithmic program as a whole. The results obtained from this proposed work have been compared with previous dissertations in terms of speeds, slices used and total area occupied by the IO planner, as well as existing architectures, and the findings confirm that the proposed architecture is better. Muneer Bani Yassein Shadi Aljawarneh Ethar Qawasmeh [18] et al., on the various cryptographic algorithms that competed against the now standard Rjindael[18]. The main inference that could be drawn from this is that, although all of them were very secure, Rjindael proved to be the best in a variety of aspects, such as security and ease of implementation and efficiency. It also helps one to understand the difference between symmetric key and asymmetric key cryptography and also helps one to understand the different situations in which these two types of cryptographic algorithms can be used. Detailed information on private and public key cryptographic methods has also been written.

Sriperumbuduru Srilaya [25] et al. tested the two most widely used symmetrical algorithms to be similar DES (Data Encryption Standard) and AES [25]. The performance measurement metrics for each algorithm were used to do a comparative study dependent on the input scale.The parameters analyzed are encryption time and decryption period, performance, power usage, energy consumption, simulation period. This also illustrates the value of cryptography in providing better protection of knowledge through the communication channel and ensuring that the network is safe from attacks. Archana Mishra and Saurabh Sharma wrote about an AES (Advanced Encryption Standard) encryption architecture, which is focused primarily on Very Large Scale Implementation and notes that the suggested implementation is successful against attacks [13]. As is known, AES is the commonly used symmetric key cryptography algorithm, which encrypts input data into illegible text with the aid of a given user key due to various transformation measures. The major inspection here is just AES-128 bit architecture over VHDL-based transformation phases and higher speeds are also one of the core aspects of this research in the overall implementation of AES algorithmic program. Specific information is provided concerning the working of the 10 rounds in AES encryption and even reverse operations. Lookup tables were introduced for high speed, and few other improvements were made to the application portion of implementation.

Javier Herranz [13] discusses the design of attribute-based encoding methods that are not easy compared to the design of identity-based encoding methods. This shows how an attribute-based system and policy can be paired with a collision-resistant hash function to provide an identity-based encryption process. In 2016 and 2017, Odelu et al. proposed both an attribute-based encryption scheme in a separate logarithm environment, with no bilinear pairings, and an RSA framework attribute-based encryption scheme, both accepted and regulated. In both the RSA and the discrete logarithm environments, stable identity-based encryption schemes can be achieved without bilinear pairings, which would be a field development. Unfortunately, the author presents here the full attacks of the two schemes proposed by Odelu et al. K. Rahimunisa [16] et al presented another hardware-based implementation of the AES algorithmic program and the main objective was to improve throughput compared to previous iterations of similar architectures [16]. This introduces parallel type formulation in the folded AES architecture to achieve high throughput. The implementation of the FPGA architecture proves to consume less space. K. stated the proposed implementation. Rahimunisa, Karthigaikumar, Soumiya, Jayakumar and Suresh use 128-bit keys for encryption and decryption. The folded concept is introduced to reduce the overall area and parallel architecture is used to achieve better throughput..

Mahmoud Alfadel El-Sayed M [26] et al., introduced a rundown of a new exhibition assessment standards applied to encryption procedures [26]. The authors analyzed numerous AES algorithms and the efficiency of their modes is depending on time and distance. Various methods such as Anova were used for analysis.Text, image and video are some of the file types that were used to conduct the experiments. The modes OFB, CFB, CBC and ECB are fundamentally extraordinary as far as throughput while these modes have a confidence level of 95% time in case of encryption time. From this, it can be inferred that a great number of experimentations could be performed to compare parameters like memory usage, encryption and decryption implementation time using different key sizes and text files. Also, this experiment was tried and tested this experiment with various different setups. Soumya Nag K [20] et al., have depicted a new algorithm to overcome the disadvantages of single key setup and to increase the higher secure data transfer in a network [20]. A latest version of FPGA with multiple key encryptions has been proposed based on AES-192. The algorithm, which is also static, uses only one key throughout the conversation. If the key is used more, the hackers might figure out a way to discover the strategy of encryption. The algorithm is being used in VHDL in Xilinx ISE 9.2. The process are being synthesized and repeated in ModelSim simulator. This work features one of the novel methodologies led to the one of the highly secured algorithms, which is tolerant to data theft.

Akash Kumar Mandal [27] et al., have studied encryption algorithms AES and DES using MATLAB. If the execution is completed, these methods are evaluated for certain parameters such as the main avalanche effect induced by one-bit variation by keeping the plain text and vice versa fixed and the memory needed for the simulation and implementation of the code. It should also be noted that DES is widely used encryption method for financial services. AES has a strong avalanche impact and is suitable for encrypting messages transmitted from one entity to another, for transfers, etc. Also, it can be concluded that parametric characteristics such as memory, Processor speed, and processing time are being observed.

3. PROPOSED EFFICIENT HIGH-SPEED AES BASED AUTHENTICATION ALGORITHM

Data Encryption Standard **(DES)** the original predecessor of the present AES algorithm. Federal Information Processing Standard (FIPS) has initially chosen DES for the United States in 1976. It emerged to be generally used universal algorithm for numerous business applications, including many monetary exchanges. There were also questions posed by the National Security Department, which culminated in the algorithm staying controversial. NIST chose to select an algorithm that would serve as a replacement to DES, which could never again be considered as fragile in the light of its limited key scale and the improved efficiency of computing capacity. Therefore, NIST announced another computation in specific, and then approved Advanced Encryption Standard (AES) as a replacement, AES with 128 bit squares and 128 bit, 192 bit, and 256 bit keys to substitute DES. The symmetric-key estimation of the "delicate, unclassified" knowledge is sought by NIST.So that, the algorithm which is to be chosen should be accessible to worldwide for free. The algorithm that won would get admiration and significance and also get a huge consulting from across the world for its security applications. Additionally, AES would definitely get one of the most broadly utilized cryptographic calculations on the planet. In 1998, 21 industry and scholarly gatherings offered contenders; fifteen met NIST's accommodation models. This specification describes the Rijndael algorithmic system as a symmetric algorithm that can handle data blocks with a length of 128 bits and keys with a length of 128, 192 and 256 bits. Rijndael was referred to as the AES algorithm because it was able to handle extra block sizes and key lengths. The three separate main lengths as seen above can be used for the algorithm and thus such distinctive "flavors" should be referred to as "AES-128," "AES-192" and "AES-256." The distinctive key lengths need just a few additional changes, and AES is often effective and simpler to conduct.

The hidden key is considered secure in hand and known only to the sender and the recipient, two communicating parties in particular. If the knowledge communication is additionally appropriate to duplex, otherwise each side must have its own AES encryption and decryption FPGA-based processor. This would not necessarily be the case, though, because the key could be reached by an agent who may endanger the whole encryption process during transmission. The integrity test using the SHA algorithmic software is perhaps the most significant element in this method. With the help of SHA, one can perform integrity check of the acquired decrypted data and the sender and receiver side and if equal, is a proof that no tampering to the data has been done.

The method of the communication of information from the sender to receiver is as follows:

- The Sender designs the modules required for encryption in the FPGA board and then uses the already known plaintext and cipherkey for encrypting to obtain the ciphertext which will be transmitted along the channel. Additionally, the plain content is all the while sent to SHA which is inserted into encryption module to acquire the hash code on sender side i.e MAC_T
- After the above step, the 128-bit cipher text is annexed with the hash code and this is the collective part is sent from the sender part. At that point the output buffer gathers and sends the Cipher-text and hash code blend over the correspondence channel. All the customer in the center can see the figure content, yet the only individual who has Secret-key can utilize it or one could term him as the receiver.
- The cipher text and hash codes are separated from the data appended on the receiver side. The Receiver as built for the FPGA processor is fitted with some Decryption Module which is used to decode the Ciphertext received from the sender for the acquisition of plaintext. In fact, once the plaintext has been decrypted, this plaintext is submitted back to SHA, which is built in the encoding module on the receiver side to provide the hash code.
- Finally, on the receiver side, both the hash codes i.e. MAC_T and MAC_R are compared using XNOR function to check if equal [1].
- If the hash codes are equal, it indicates that the received cipher text wasn’t tampered with any third-party adversary during data transmission between sender and receiver.

Alternatively, if the new Recipient has to transfer confidential details to the initial Sender, at this stage the above-mentioned protocols should be reshaped for the corresponding Sender and Receiver work. Accordingly, the goal of the study, the FPGA implementation of safe data transmission utilizing AES-based authentication, can be achieved very efficiently, for all purposes , by the efforts of this dissertation research, the by using FPGA boards, on both sender and reciever of the data transfer, given that both the sender and reciever have knowledge about the Cipherkey used. The goal of the project is to recognize and undertsand the basic cryptographic specification known as AES, which is one of the most widely implemented and versatile symmetrical algorithms that could be easily adapted in the future to the strongest protection structures, which has been opened up by the tremendous development and key breakthroughs in the recent history [18][20]. And provide message integrity between sender and receiver using Hash codes generated using SHA-1 algorithm. Also, SHA is implemented concurrently with AES for confidentiality and security and also integrity check [1]. Later, the AES is discussed in detail with the steps required in algorithm, and mathematical analysis of the proposed algorithmic program. Futhermore, the software implementation including few drawbacks are examined and the different approaches related to hardware architecture and implementation is studied comprehensively. This study also includes very specific details on the SHA algorithm. The ultimate aim is to implement an algorithmic program which is highly reliable and a proper sufficient answer for the much prevailing security concerns in implementation of AES. Also, we are striving to make the proposed algorithhm's implementation cost effective. Thus, finally implement a high speed AES based authentication algorithm by combining VLSI for reliable communication of data with adequate security features The below fig.1 shows a very detailed block diagram of the proposed High-speed AES based authentication algorithm. The figure represents both parts, i.e. the transmitter part which inputs and sends the encrypted plain text and a hash code and the receiver part which receives the cipher text and hash code from the message transmitter side.

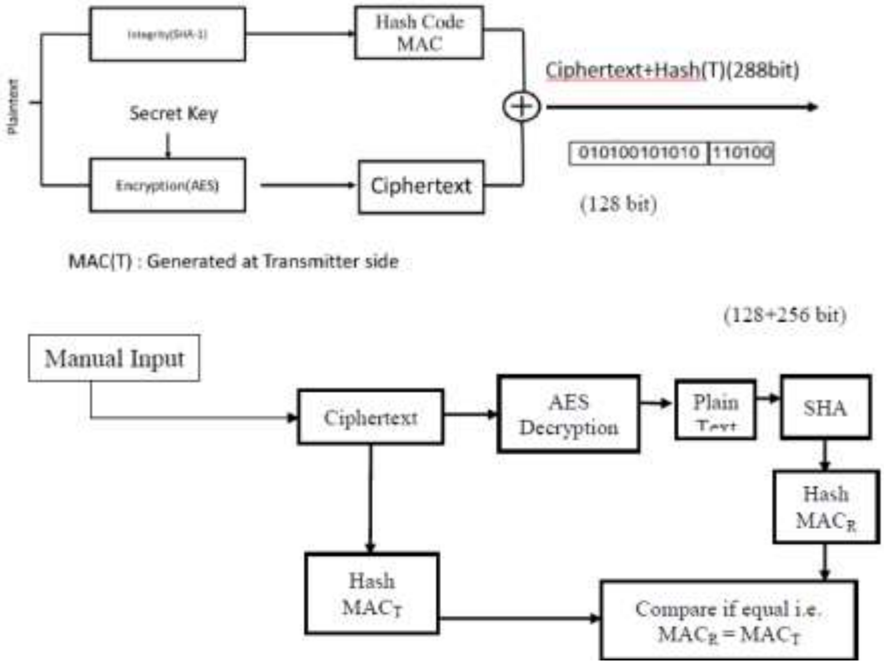

Fig. 1 AES and SHA based authentication block diagram

The following steps is performed to provide authentication by using AES and SHA Algorithm

Step 1 : Choose a plaintext (Pm) of size 128 bit and key (K) which are mostly couple of words.
Step 2 : Convert the plaintext (Pm) and key (K) into hexadecimal values
Step 3 : The plaintext (Pm) and key (K) are given as input for AES encryption and also for SHA algorithm to obtain hash.
Step 4: As part of AES encryption, derive the set of round keys from the cipher key (in Hex)
w[4] = w[0] XOR g(w[3],, where g(w[3]) is circular shift of w[3] then byte Substitution (S-Box) and finally adding round constant , w[i] is round key of i th round and w[n] = w[n-1] XOR w[n-4] for i>4.
Step 5 : Adding Round key, Round 0.
XOR of Round key and Round 0 gives us new State matrix
Step 6 : Round 1, Sub Bytes.
Four rows are shifted cyclically to the left by offsets of 0,1,2, and 3.
Step 7: Round1, Mix Column.
Mix Column multiplies predefined fixed matrix against current State Matrix.
Step 8: Add Round key, Round 1.
XOR of Round key and Round 1 gives us a new matrix.
Step 9 : Below operations are done similarly for next rounds
Sub Bytes, Shift Rows, Mix Columns, Roundkey. For last round, mix column is not performed.
Step 10 : Now, finally we obtain CipherText from AES encryption.
Step 11 : As part of SHA algorithm plaintext (Pm) and key (K) are taken as input.

Step 12 : Initialize five strings of random Hexadecimal numbers which represent as a part of the Hash function (in Hex ,128 bits).
Step 13 : padding of message is done .padding is done until the message is of 512 bits 1 is followed by enough zeroes upto 448 bits and then message of 64 bits is added at the end.
Step 14 : The padded message is then divided into 512 bit blocks, and each block is again divided into sixteen 32-bit words, W_0 … W_{15} . In our message 'abc', there's only one block because our message is less than 512-bits total.

$W(i)=C1(W(i-3)\oplus W(i-8)\oplus W(i-14)\oplus W(i-16))$, where W(i) is bitword.

Step 15 : Hash values are stored in differengt variables.
Step 16 : For each chunk, 80 iterations are done, i, which are necessary for hashing , and the following steps on each chunk must be executed, Mn.
For iteration where $16 \le i \le 79$, operation described below is performed $W(i)=C1(W(i-3)\oplus W(i-8)\oplus W(i-14)\oplus W(i-16))$,Where $Cn(X)=(X<<n)OR(X>>32-n)$.
where X << n is the **left-shift** operation and X>> 32-n is the **right-shift** operation
Step 17 : For 80 iterations, $0 \le i \le 79$, it is required compute,
$TEMP= C5*(A)+f(i;B,C,D)+E+W(i)+K(i)$.
Step 18 : Reassigning the following variables.

E=D
D=C
C=S^30 (B)
B=A
A=Temp

Step 19 : Store the result of the chunk's hash to the overall hash value of all chunks, and proceed to execute the next chunk. And then we obtain the Message digest or Hash value (h0).
Step 20 : Appending both the ciphertext and Message digest (128+160 bits) and then communicated through the channel as output.
Step 21 : Now first 128 bits (ciphertext) and next 160 bits (Hash code) are seperated and then respective process is done.
Step 22 : First 128 bits undergo AES decryption process which is similar to AES encryption and finally we obtain plain text as the output of AES decryption.
Step 23 : Now the plaintext obtained after decryption is taken as input message for SHA.
Step 24 : Now SHA algorithm is performed as mentioned in above steps.
Step 25 : Finally another message digest or hash code (h1) is obtained.
Step 26 : Now compare h0 and h1 , if both are equal the message is considered to be untampered and thus message integrity is obtained.

Typical Example:

As, it was already stated earlier the proposed algorithm has two parts : Transmitter part and Receiver part.

At the transmitter side

- First a plain text and key is word is given as an input to the transmitter part.
  Let the plain text sentence be: "Two One Nine Two".
  Let the key word sentence be: "Thats my Kung Fu".
- Initially, both are converted to hex values:
  Hex values of Message - "54 77 6F 20 4F 6E 65 20 4E 69 6E 65 20 54 77 6F".
  Hex values of Key - "54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75".
- The RoundKey function is used to generate the expanded round key. After the next following steps circular shift, byte substitution, adding round constant and XOR'ing the required columns first round key is generated.
  First round key: E2 32 FC F1 91 12 91 88 B1 59 E4 E6 D6 79 A2 93.
- As the key is of 128 bits, 10 round keys are generated. Similarly, it follows the above same steps to generate 10 roundkeys.
- Then, these round keys are used along with plain text to generate the cipher text.
- The algorithm then undergoes 10 rounds with following functions in each round to generate a state matrix.
  After Sub Bytes = [63 EB 9F A0 , 2F 93 92 C0 , AF C7 AB 30 , A2 20 CB 2B]

After Shift Rows = [63 EB 9F A0 , 93 92 C0 2F , AB 30 AF C7 , 2B A2 20 CB]
After Mix Columns = [BA 84 E8 1B , 75 A4 8D 40 , F4 8D 06 7D , 7A 32 0E 5D]
After Roundkey = [58 47 08 8B , 15 B6 1C BA , 59 D4 E2 E8 , CD 39 DF CE]

- Finally, after 10 rounds, we get cipher text: 29 C3 50 5F 57 14 20 F6 40 22 99 B3 1A 02 D7 3A
- Simultaneously, the SHA part of the proposed algorithm does its work. It generates MAC code [MAC_T] with the help of plain text.
- After following the required functions in SHA algorithm, the MAC code of 160 bit generated is: 67 45 23 01 EF CD AB 89 98 BA DC FE 10 32 54 76 C3 D2 E1 F0
- The ciphertext and MAC_T code are appended and sent to the receiver part.
  29 C3 50 5F 57 14 20 F6 40 22 99 B3 1A 02 D7 3A 67 45 23 01 EF CD AB 89 98 BA DC FE 10 32 54 76 C3 D2 E1 F0

At the receiver side

First, the encrypted input and the key word are given to the receiver part and it breaks it down into two parts with the first being 128 bits and the second being 160 bits.

- The 128 bits text is decrypted with the help of key word and it gives us the initial plain text in hex value.
  Plain Text: 54 77 6F 20 4F 6E 65 20 4E 69 6E 65 20 54 77 6F
- Then the plain text generated here is used to perform SHA algorithm to form MAC at receiver part [MAC_R].
  MAC_R : 67 45 23 01 EF CD AB 89 98 BA DC FE 10 32 54 76 C3 D2 E1 F0
- Then, MAC_T and MAC_R are compared and if the both are equal then it follows data integrity.

4. FPGA IMPLEMENTATION OF HIGH-SPEED AES BASED AUTHENTICATION ALGORITHM

The layered architecture of combined AES and SHA has been implemented to achieve a very high level of security with integrity check at both transmitter side and receiver side too. The transmitter side consists of AES encryption with SHA to produce a hash code and the receiver side includes AES decryption with SHA to receive another hash code. Both these hash codes are verified to perform integrity check. This proposed architecture has been implemented on Xilinx software 14.7 analysis and performance has been compared for three FPGAs namely Virtex-4, Virtex-5 and Spartan-6 FPGAs. The overall schematic, synthesis, software implementation has also been explained. The proposed architecture of AES algorithm and SHA architecture on the transmitter side and receiver side has been implemented on multiple FPGAs namely Virtex-4, Virtex-5, Spartan-6, Artix-7 and Virtex-7 FPGAs. For optimum performance the FPGA versions were selected such that the selected FPGA boards have the maximum number of Input-Output bonds, internal LUTs, FFs to support the proposed architecture. The results obtained have been verified and are in accordance with the desired output. The RTL Schematic and the corresponding outputs have been presented. A comparison table which compares the synthesis and properties of the various FPGA's which are being put into use is also presented.

4.1 RTL Schematics

The figure below i.e. Fig.2 shows the RTL schematic of AES encryption module which performs the encryption part of the whole process and is present in the transmitter part of the proposed algorithm.

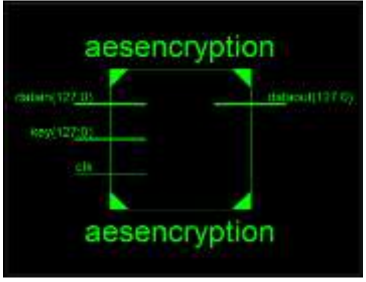

Fig.2 RTL schematic of AES encryption

The Fig.3 shows the zoomed in RTL schematic picture of the AES encryption module which was represented in the Fig.2. This zoomed in picture allows us to further understand the working of AES encryption.

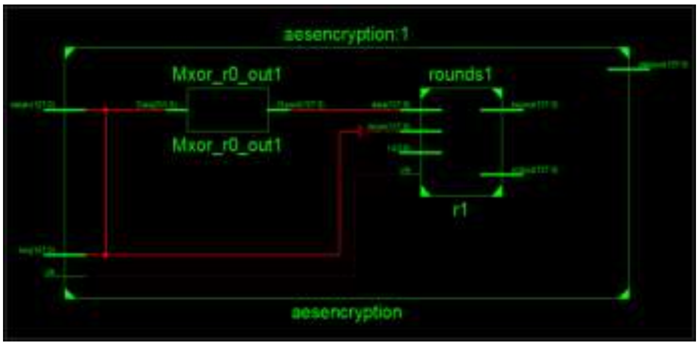

Fig.3 RTL schematic of AES encryption-zoomed in

The figure below i.e. Fig.4 shows the RTL schematic of AES decryption module which performs the decryption part of the whole process and is present in the receiver part of the proposed project.

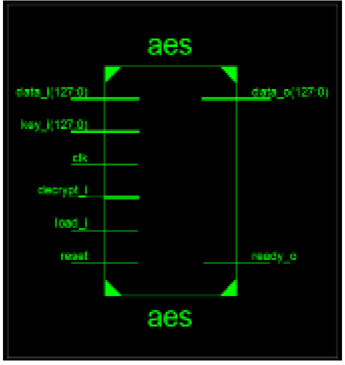

Fig.4 RTL Schematic of AES decryption

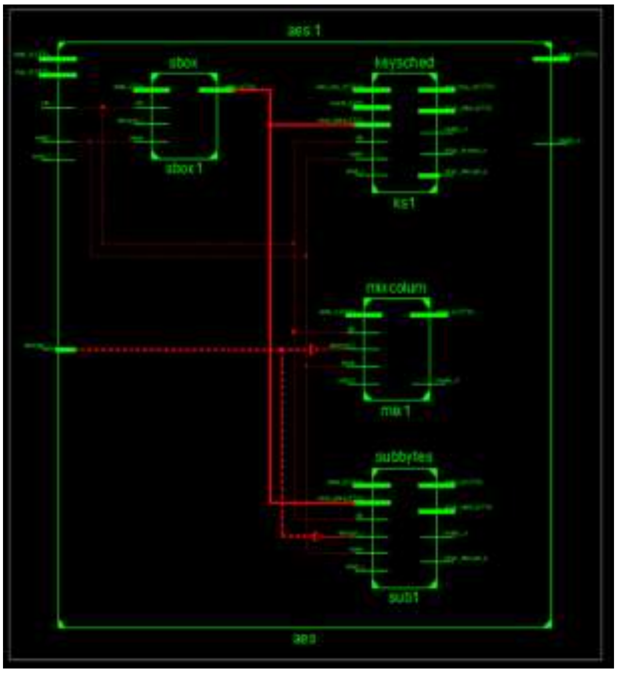

Fig.5 RTL schematic of AES decryption-zoomed in

The Fig.5 shows the zoomed in RTL schematic picture of the AES decryption module which was represented in the Fig.4. This zoomed in picture allows for further understanding of the working of AES decryption. The below figures namely Fig.6 and Fig.7 depict the RTL schematic of SHA used. It has been split as SHA-top module and SHA core in the code thus, the two schematics.

The blow figure i.e. Fig.6 shows RTL schematic of the module SHA.

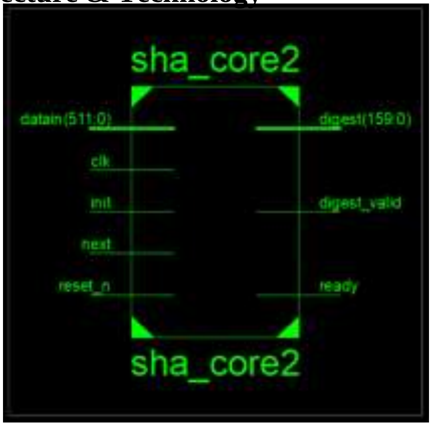

Fig.6 RTL schematic of SHA top module

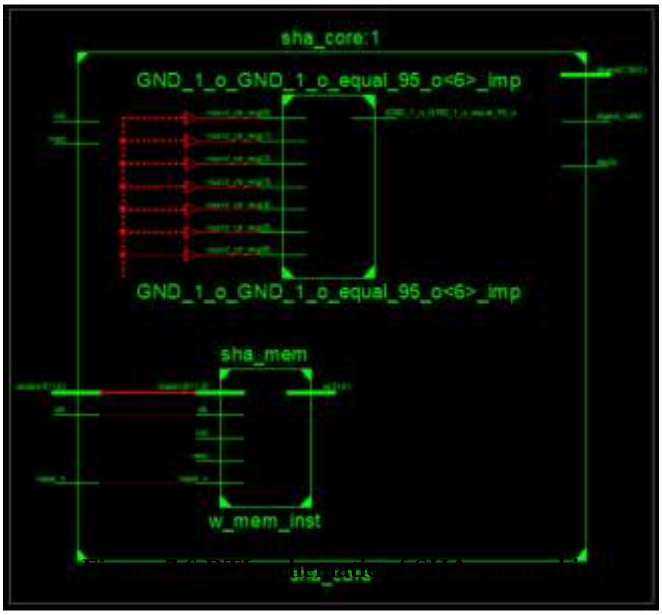
sha_core:1
GND_1_o_GND_1_o_equal_95_o<6>_imp
GND_1_o_GND_1_o_equal_95_o<6>_imp
sha_mem
w_mem_inst
sha_core

Fig.7 RTL schematic of SHA-zoomed in

Finally, the below two pictures shows the RTL schematic of Transmitter side i.e. AES encryption with SHA and receiver side i.e. AES decryption with SHA. Fig. 8 shows the Transmitter side which includes AES encryption with SHA to produce hash code. In the schematic it can be clearly seen that, three different modules i.e. aesencryption, append and sha_core. ‘The ‘appendp’ module is just a module to append the outputs of encryption and sha to send to the next step.

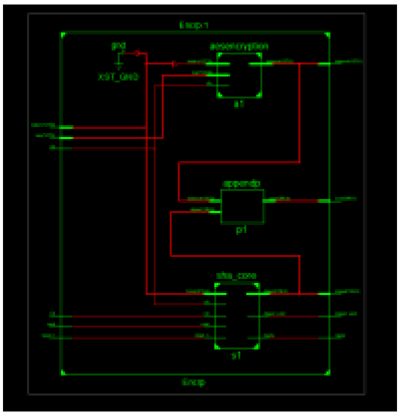

Fig.8 RTL schematic of Transmitter side

Fig.9 shows the Transmitter side which includes AES decryption with SHA to produce hash code. In the schematic we can see, two different modules i.e. aes, append and sha_core2.

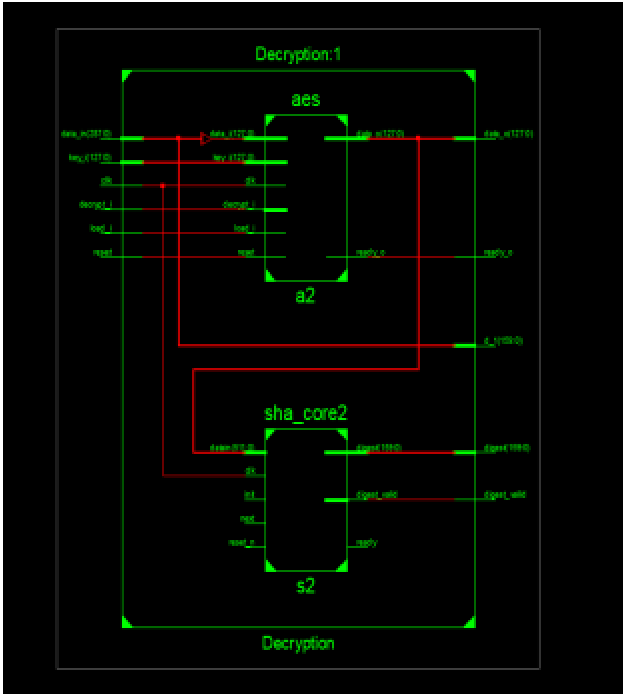

Fig.9 RTL schematic of Receiver side

In the below Fig.10, the final output in XILINX is shown for the transmitter part of AES and SHA where AES encryption is done with SHA hashing.

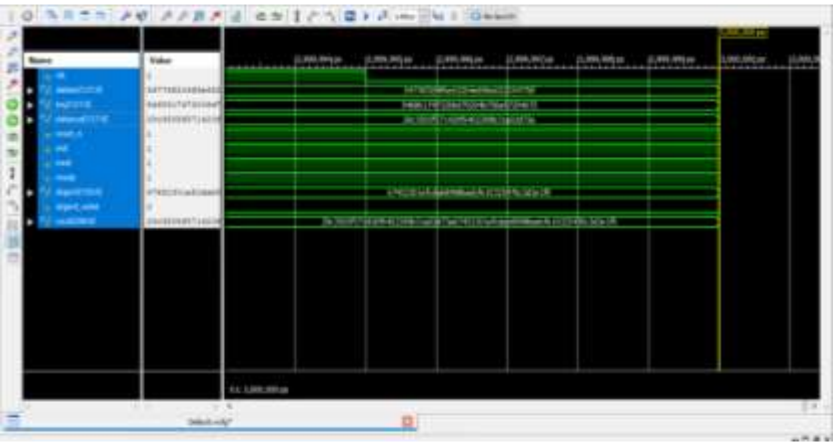

Fig.10 Output using ISIM for Transmitter part

Here, datain is plaintext, key is the key given by user, dataout is the cipher test, Digest is the hash code or MAC_T and cout is the appended output that is dataout appended with digest. In the below Fig.11, the final output in XILINX is shown for the receiver part of AES and SHA where AES decryption is done with SHA-1 hashing.

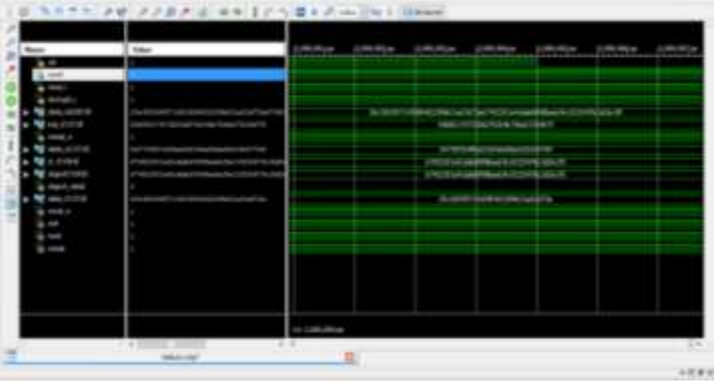

Fig.11 Output using ISIM for Receiver part

Here, data_in is the cout from the previous step, key_i is the user given key, data_o is the plain text after decryption of cipher text, d_1 is the hash code at transmitter side i.e. MAC_R , digest is the hash in transmitter side.

4.2 HDL SYNTHESIS REPORT

Macro Statistics

Adders/Subtractors used = 5

Table 1 Adders and Subtractor used

4-bit adder	2
4-bit subtractor	1
5-bit adder	1
6-bit subtractor	1

Registers used = 673

Comparators used = 2

Table .2 Comparators used

32-bit comparator lessequal	1
4-bit comparator equal	1

FSM's used = 2

Multiplexer's used = 275

Table .3 Comparators used

1-bit 2-to-1 multiplexer	214
128-bit 2-to-1 multiplexer	15
32-bit 2-to-1 multiplexer	5
32-bit 4-to-1 multiplexer	1
4-bit 2-to-1 multiplexer	16
5-bit 2-to-1 multiplexer	1
8-bit 16-to-1 multiplexer	1
8-bit 2-to-1 multiplexer	21

XOR's used = 125

Table 4 XOR's used

1-bit xor2	29
1-bit xor3	12
1-bit xor4	2
1-bit xor5	1
1-bit xor6	2
1-bit xor7	1
128-bit xor2	2
32-bit xor2	5
4-bit xor2	37
4-bit xor3	1
4-bit xor4	12
4-bit xor5	1
8-bit xor2	12
8-bit xor3	4
8-bit xor4	4

4.3 COMPARISON OF PARAMETERS AMONG VARIOUS FPGA's

FPGA Type	Total no. of slice registers	Total no. of slices in LUTs	Total no. of pairs of LUT-FF completely used	Total no. of IOBs utilized	Total no .of BUFG/BUFGCTRLs
Spartan-6Q	854	10772	749	837	1
Spartan-6	857	10773	844	837	1
Artix-7	670	1468	573	870	1
Virtex-7	670	1469	573	870	1
Virtex-6	849	10769	848	836	1

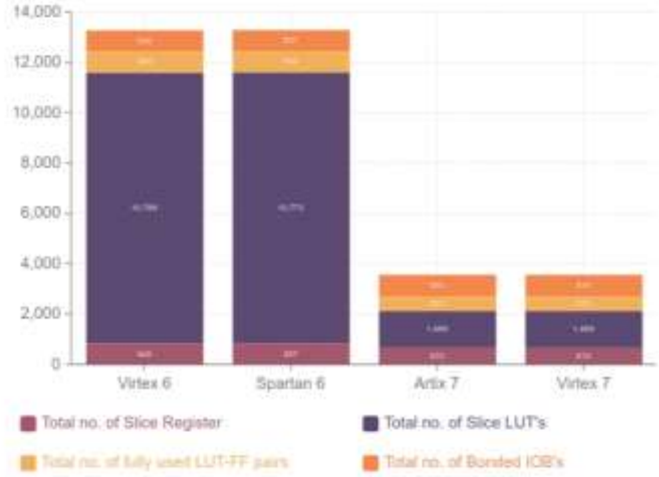

Fig.12 Graph depicting the various parameters in different FPGA

CONCLUSION AND FUTURE WORK

A combination of AES encryption algorithm and SHA 1 algorithm has been implemented in both software and hardware, i.e. both in Xilinx and on the FPGA board. The proposed idea to create a special algorithm in which AES algorithmic program is used for encryption and decryption of data while Secure Hash Algorithm is used for message integrity has been presented. At transmitted side plain text is given to both AES and SHA algorithms, then the cipher text from AES and hash code from SHA are appended and transmitted. At receiver side cipher text and hash code are separated, then cipher text is sent to decryption and then clear-text is given to SHA where it provides a hash code. Now both hash codes are compared and by doing that message integrity is checked at receiver side. This algorithm has been implemented in Xilinx ISE 14.7 and on various FPGA's namely Virtex-6, Virtex-7, Artix, Spartan 6, Spartan 6Q boards and the various parameters have been noted and graphed. As AES encryption is a symmetric algorithm (private-key cryptography). It involves key which is shared between sender and recipient in secret, so whole algorithm is in jeopardy if the key is known to third party. So, it must be made sure that private key is not tampered with during the transmission. Performance of the encryption algorithms can be improved. High throughput AES can be developed using techniques like parallelism. Also, the implementation of AES-CCM for further improving security can be implemented, but it has hardware requirements which exceed the number of required ports on FPGA's. Thus, requiring the use of multicore processors. The implementation will further benefit by implementing it on a faster CPU rather than FPGA's which provide higher speed of process completion than FPGA's.

REFERENCES

[1] Pammu, A. A., Ho, W.-G., Lwin, N. K. Z., Chong, K.-S., & Gwee, B.-H. (2019). A High Throughput and Secure Authentication-Encryption AES-CCM Algorithm on Asynchronous Multicore Processor. *IEEE Trans.Inform.Forensic Security*. *14*(4), 1023–1036. Retrieved April 4, 2020, from 10.1109/tifs.2018.2869344

[2] Paul, A. Soltani and S. Sharifian, "An ultra-high throughput and fully pipelined implementation of AES algorithm on FPGA," Microprocessors and Microsystems, vol. 39, no. 7, pp. 480–493, 2015, doi: 10.1016/j.micpro.2015.07.005.

[3] Advantages Of AES | Disadvantages Of AES." [Online]. Available: https://www.rfwireless-world.com/Terminology/Advantages-and-disadvantages-of-AES.html.

[4] Harshali Zodpe and Ashok Sapkal, " An Efficient AES Implementation Using FPGA With Enhanced Security Features ", Journal Of King Saud University - Engineering Sciences, Volume 10.1016, July 2018.

[5] Liakot Ali , Ishak Aris , Fakir Sharif Hossain and Niranjan Roy, " Design of an Ultra High Speed AES Processor for Next Generation IT Security", Computers and Electrical

[6] M.Kundalakesi MS(IT&M),M.Phil, An Overview of Modern Cryptography, 31–51. Retrieved April 4, 2020, from 10.1007/0-387-26090-0_3

[7] Aes Document 1 Final | Cryptography | Field Programmable Gate Array. *Scribd*. Retrieved Feb 2, 2020, from https://www.scribd.com/document/375361742/Aes-Document-1-Final

[8] Philemon, M. P. (2012, March 3). Implementation Of Advanced Encryption Standard Algorithm. *International Journal of Scientific & Engineering Research Volume 3, Issue 3*. Retrieved April 4, 2020, from https://www.ijser.org/paper/Implementation-of-Advanced-Encryption-Standard-Algorithm.html

[9] Riadi, I. (2018, September). Analysis of Secure Hash Algorithm (SHA) 512 for Encryption Process on Web Based Application. *ResearchGate*. Retrieved April 4, 2020, from https://www.researchgate.net/publication/327392778_Analysis_of_Secure_Hash_Algorithm_SHA_512_for_Encryption_Process_on_Web_Based_Application

[10] Hassen Mestiri, Fatma Kahri, Belgacem Bouallegue and Mohsen Machhout, “A High-Speed AES Design Resistant To Fault Injection Attacks”, Microprocessors And Microsystems, Volume 41, March 2016, Pages 47-55, Dec 2015.

[11] Ali, S., Ramji, P. V. S., Harsha, M., & Reddy, Y. (2017, October). VDAAESA: VLSI Based Design And Analysis Of Advanced Encryption Standard Algorithm. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*.

[12] Guduri, M., & Rajesh, V. J. (n.d.). Design of High Performance 32-bit Cryptographic processor. International Journal of Scientific & Engineering Research.

[13] Archana Mishra and Saurabh Sharma, “Design And Implementation of High Speed AES Algorithm for Data Security”, International Journal of Engineering Sciences & Research Technology, Volume 10.5281, August 2016

[14] Adnan Mohsin Abdulazeez and Ari Shawkat Tahir, “Design And Implementation of Advanced Encryption Standard Security Algorithm Using FPGA”, International Journal of Scientific & Engineering Research, Volume 4, Issue 9, September-2013.

[15] Rafidah Ahmad, Asrulnizam Abd and Widad Ismail, “Development of an Improved Power-Throughput Blowfish Algorithm on FPGA”, IEEE 12th International Colloquium on Signal Processing & its Applications, vol. , PP., march 2016.

[16] K. Rahimunnisa, P. Karthigaikumar, Soumiya Rasheed, J. Jayakumar and S. Suresh Kumar “FPGA implementation of AES algorithm for high throughput using folded parallel architecture”, Wiley Online Library, vol 10.1002, PP. 2225-2236 , October 2012.

[17] J. Balamurugan and E. Logashanmugam, “High-Speed Low-Cost Implementation of Advanced Encryption Standard on FPGA”, 2nd International Conference on Current Trends in Engineering and Technology, vol. ICCTET’14, PP.371, July 2014.

[18] Muneer Bani Yassein Shadi Aljawarneh Ethar Qawasmeh, Wail Mardini and Yaser Khamayseh, “Comprehensive Study Of Symmetric Key And Asymmetric Key Encryption Algorithms”, The International Conference On Engineering & Technology, July 2017.

[19] (2009, May 16). Simple Hex Multiplication Question From AES Algorithm. *Physics Forums | Science Articles, Homework Help, Discussion*. Retrieved April 4, 2020, from https://www.physicsforums.com/threads/simple-hex-multiplication-question-from-aes-algorithm.314497/

[20] K. Sowmya Nag, A. C. Nuthan, and H. B. Bhuvaneswari, “Implementation of advanced encryption standard-192 bit using multiple keys,” 2013, doi: 10.1049/cp.2013.2514.

[21] P. Kakarountas, G. Theodoridis, T. Laopoulos, and C. E. Goutis, “High-Speed FPGA Implementation of the SHA-1 Hash Function,” 2005, doi: 10.1109/idaacs.2005.282972.

[22] F. Kahri, H. Mestiri, B. Bouallegue, and M. Machhout, “Efficient FPGA hardware implementation of secure hash function SHA-256/Blake-256,” 2015, doi: 10.1109/ssd.2015.7348105.

[23] G. Wang, “An Efficient Implementation of SHA-1 Hash Function,” 2006, doi: 10.1109/eit.2006.252210.

[24] S. Chandra, S. Paira, S. S. Alam, and G. Sanyal, “A comparative survey of Symmetric and Asymmetric Key Cryptography,” 2014, doi: 10.1109/icecce.2014.7086640.

[25] S. Srilaya and S. Velampalli, “Performance Evaluation for DES and AES Algorithms- An Comprehensive Overview,” 2018, doi: 10.1109/rteict42901.2018.9012536.

[26] M. Alfadel, E.-S. M. El-Alfy, and K. M. A. Kamal, “Evaluating time and throughput at different modes of operation in AES algorithm,” 2017, doi: 10.1109/icitech.2017.8079948.

[27] K. Mandal, C. Parakash, and A. Tiwari, “Performance evaluation of cryptographic algorithms: DES and AES,” 2012, doi: 10.1109/sceecs.2012.6184991.

[28] S. Chandra, S. Bhattacharyya, S. Paira, and S. S. Alam, “A study and analysis on symmetric cryptography,” 2014, doi: 10.1109/icsemr.2014.7043664.

[29] Nadeem and M. Y. Javed, “A Performance Comparison of Data Encryption Algorithms,” doi: 10.1109/icict.2005.1598556.

[30] S. Srilaya and S. Velampalli, “Performance Evaluation for DES and AES Algorithms- An Comprehensive Overview,” 2018, doi: 10.1109/rteict42901.2018.9012536.

[31] J. Yenuguvanilanka and O. Elkeelany, "Performance evaluation of hardware models of Advanced Encryption Standard (AES) algorithm," IEEE SoutheastCon 2008, Huntsville, AL, 2008, pp. 222-225.

[32] R. Martino and A. Cilardo, "A Flexible Framework for Exploring, Evaluating, and Comparing SHA-2 Designs," in IEEE Access, vol. 7, pp. 72443-72456, 2019.

[33] Y. Yang, D. He, N. Kumar and S. Zeadally, "Compact Hardware Implementation of a SHA-3 Core for Wireless Body Sensor Networks," in IEEE Access, vol. 6, pp. 40128-40136, 2018.

[34] On the Exploitation of a High-Throughput SHA-256 FPGA Design for HMAC, Harris E. Michail, George S. Anthanasiou, Vasilis Kelefouras, George Theodordis and Costas E. Goutis

[35] Chen, S., Hu, W., & Li, Z. (2019). High Performance Data Encryption with AES Implementation on FPGA. Retrieved from http://dx.doi.org/10.1109/bigdatasecurity-hpsc-ids.2019.00036