UACEE International Journal of Advancements in Electronics and Electrical Engineering Volume 1: Issue 2
[ISSN: 2319 – 7498]

Hybrid AES-DES Block Cipher: Implementation using Xilinx ISE 9.1i

Anurhea Dutta, Prerna Bharti, Swati Agrawal, Surekha K S

Electronics and Telecommunication Department, Army Institute of Technology, Pune, India

Email: rhea712@gmail.com, prernabharti20@gmail.com, swati6790@gmail.com, surekhaks@yahoo.com

Abstract- **In this era of information, need for protection of data is more pronounced than ever. Secure communication is necessary to protect sensitive information in military and government institutions as well as private individuals. Current encryption standards are used to encrypt and protect data not only during transmission but storage as well. Data Encryption Standard was introduced in early 1970s as a standard cryptographic algorithm to protect data. However, due to its short 56-bit key length, simple brute force attacks cracked it in less than 10 hrs. Another disadvantage was also the possibility of weak and semi weak keys. In the year 2000, Rijndael Encryption algorithm or AES was chosen by National Institite of Standards and Technology(NIST) to be adopted by the U.S. Government as the new Encryption standard to replace the outdated and easily crackable DES. The major advantage lay in the non-linearity of the key-schedule which eliminated the possibility of weak and semi weak keys. This encryption algorithm is virtually crack-proof till date but research has concluded that side channel attacks can be a concern if the encryption and crack are running on the same server.**

In this paper we introduce the concept of hybridizing the AES and DES standards. The name "Hybrid" implies that this encryption algorithm has built in features which have been inherited from either of the constituent standards. The AES standard has been incorporated into each fiestal round of DES thereby reinforcing the DES architecture as well as giving rise to much more secure encryption algorithm. The 256-bit block cipher uses 128-bit cipher key adopted from the AES key schedule and incorporates 10 rounds. This design has been implemented in VHDL using Xilinx ISE 9.1i platform and targeted on a XILINX XC3S400 based FPGA technology.

Keywords- AES, DES, Hybrid algorithm, security enhancement, VHDL, FPGA implementation, Xilinx.

I. INTRODUCTION

The astounding growth of the Internet and computer systems has increased the dependence of both organizations and individuals on the information stored and communicated using these systems. However, at the same time, it also brought about a plethora of new issues and concerns, chief among them being the need to protect data and resources from disclosure during storage as well as transmission, guarantying the authenticity of data and messages, and protecting systems from network based attacks [9]. In the early 1970s, encryption algorithms such as DES had been sufficient to handle most security needs. but it became apparent that the time of DES had quickly approached its end. With the limitations of DES's 56-bit key and the advent of faster computers, DES could no longer be considered a secure algorithm. Simple brute force attacks could crack into the DES algorithm in less than 10 hours rendering it to be a less secure and outdated algorithm over time. With the eminent failure of DES, the National Institute of Standards and Technology (NIST) had issued a request for submissions of stronger encryption algorithms to replace the aging DES. These algorithms had to conform to several strict requirements: (a)it must be a block cipher;(b)longer key length;(c) larger block size;(d)fast in computation and; (e) greater flexibility. After several rounds of submissions and eliminations, the NIST had narrowed the applicant pool down to five finalists out of which Advanced Encryption Standard (AES) algorithm, also known as Rijndael algorithm, was selected. This cipher was developed by two Belgian cryptographers by the names of Vincent Rijmen and Joan Daemen.

Rijndael is the most popular symmetric key block cipher used today. It uses a block size of 128 bits with a variable key length of 128 bits,192 bits or 256 bits. While DES and many other ciphers used a Feistel network, Rijndael uses a substitution-permutation network. This substitution-permutation network allows Rijndael to perform fast in both software and hardware applications. Rijndael is simple to implement and uses very little system memory.

Rijndael is used for both classified and non-classified government information today and is seen as being practically crack-proof. While the algorithm is seen as being theoretically able to be cracked, it is not a realistic threat with today's level of technology. Brute force attacks against Rijndael have proven ineffective to date. Side channel attacks, which work to attack the implementations of the cipher rather than the cipher itself, have proven that a crack of Rijndael is possible but not a practical concern unless the crack is running on the same server as the encryption is happening on. Despite the fact that AES represents the current recommended standard by NIST for symmetrical based encryptions; issues were raised on possible attacks against the algorithm as described above. This is due to the fact that the core structure of the AES itself renders a clean, simple algebraic method, hence yielding the algorithm susceptible towards algebraic based cryptanalysis attacks.

This paper presents an idea of integrating AES into the fiestal architecture of DES, embracing advantages from either of the constituent standards. This results in a much more efficient and crack resistant hybrid encryption algorithm. It utilizes a 128-bit cipher key on a 256-bit plaintext to give rise to a ciphertext of 256 bits.

II. HYBRID AES-DES CONCEPT

The basic idea of the proposed hybrid model is to integrate AES into each iteration of the fiestalnetwork of DES. Mathematically, each round of the model can be expressed as:

$$L_n = R_{n-1} \quad (1)$$

$$R_n = AES\ (L_{n-1}\ \text{xor}\ \ R_{n-1}\ \text{xor}\ K_n) \quad (2)$$

The above set of equations is repeated over each of the 10 rounds.

The input block of 256 bit data is split into two halves, the left and the right. For each round *n*, XOR function is carried out between three variables- the left and the right halves of the previous round (L_{n-1},R_{n-1})and the key generated for that round(K_n). The result of this is channelled as input for the AES algorithm. The output of the AES process represents the right half R_n.The AES operations include the byte substitution, shift row, mix columns and add round key operations. The left half of the round L_n is simply the previous right half R_{n-1}. The above mentioned equations are iterated over the 10 rounds in compliance with the keys generated from the key schedule process.

The key schedule for the proposed hybrid algorithm has been directly adapted from AES. This has been done in order to reduce the computational complexity which would have otherwise occurred if multiple sets of keys were employed for each of the individual standards. Another contributing factor was also that the cipher key of DES has disadvantages of producing weak, semi weak and complement keys. The cipher design of DES is also flawed as the '*f*' function with its s-boxes and p-boxes have inherent weaknesses. The initial and final permutation as stated in the standard also has no security benefits. Thus, they have been eliminated in our hybrid model. The proposed number of iterations of this hybrid system is set at 10. This has been done because the mathematical modelling for the key expansion of the AES limits its generation at 10 keys based on the number or rounds as applied for a 128 bit based AES encryption process. However, the iterations can also be varied to suit the level of security implementation for the Hybrid AES-DES. Fig. 1 below displays the basic architecture of a Hybrid AES-DES algorithm.

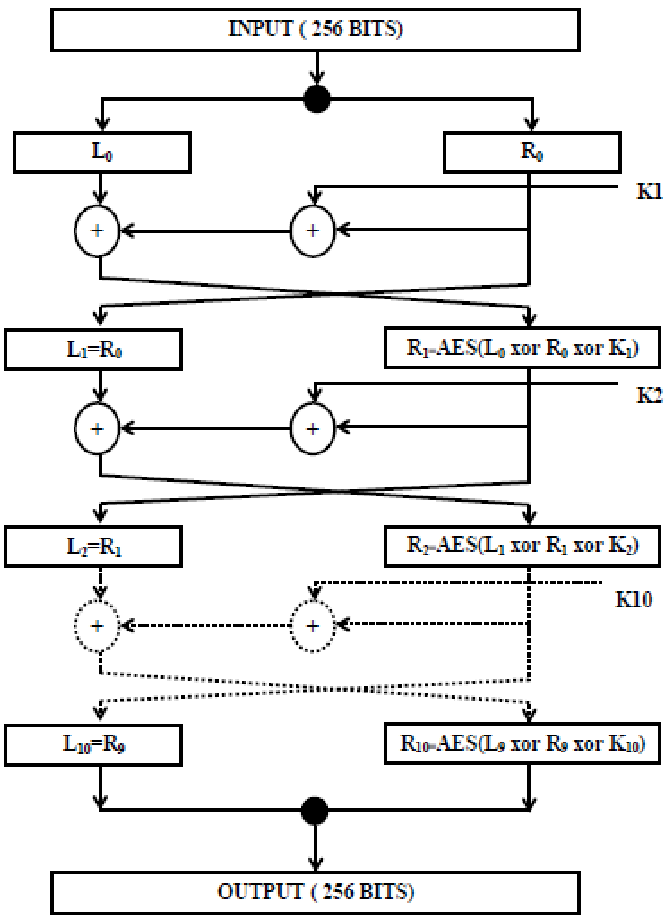


Fig 1: Hybrid algorithm block diagram

A complete Hybrid AES-DES operation performs 10 layers of Feistel calculations, utilizing the non linear Equations (1) and (2) as well incorporating

10 sets of AES. This increases the strain onto the computational density needed to complete the encryption and decryption processes. Consequently, the number of layers as applied for the hybrid structure is set as a varying parameter as controlled by the user to comply with the required level of security.

In organizations that require increased standard of security for data and multimedia transmission and storage, a higher number of layers of the algorithm can be employed. An inverse policy can be applied to organizations with lower security requirements in which the number of layers are reduced.

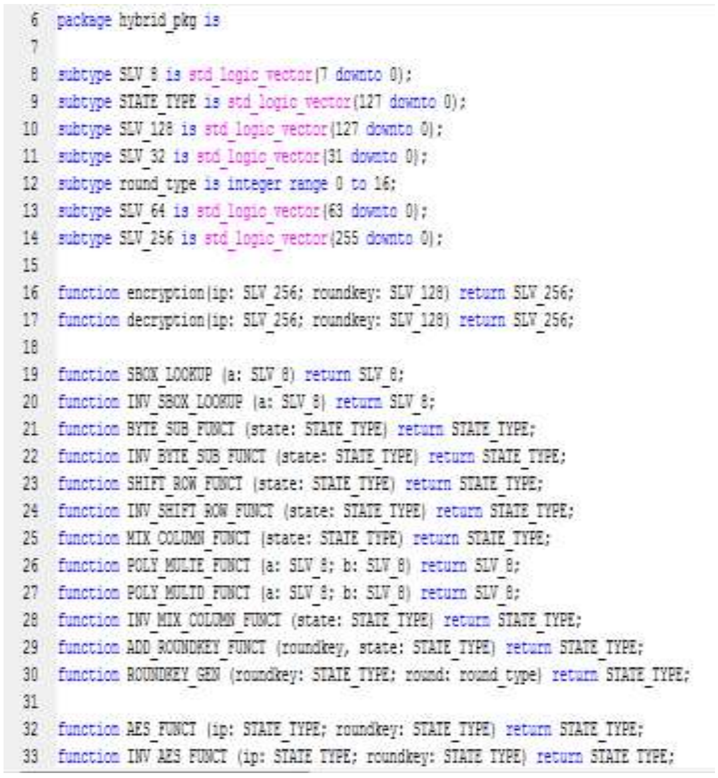
III. APPROACH TO IMPLEMENTATION AND RESULTS

Firstly, both the encryption algorithms were studied in detail. The Advanced Encryption Standard was implemented module by module. These separate functions were then incorporated into a main code thereby implementing the AES standard. This was followed by the integration of this standard in the proposed fiestal network of DES resulting in the implementation of the 256-bit hybrid block cipher.The coding of the above algorithms was done in VHDL. The platform used was Xilinx ISE 9.1i.

AES Encryption standard was taken up as the first algorithm to be implemented. Modules listed below were coded and implemented using Xilinx ISE 9.1i and tested successfully using Spartan 3 FPGA Kits.

a) S-box Lookup and its inverse,
b) Byte substitution and its inverse,
c)Shift row transformation and its inverse,
d)Mix column transformation and its inverse,
e) Polynomial multiplication and its inverse,
f) Add round key function,
g) Round key generation function.

The above mentioned modules were then compiled into a VHDL package. A new VHDL module for AES was implemented which called this package from the work library. AES encryption and decryption, was then made as a function and updated in this package.The VHDL module of the hybrid encryption algorithm was coded so as to integrate the fiestal network of DES with AES implemented in the package. Our final step was to code our hybrid algorithm as a function in the VHDL package. Sample code of the VHDL package incorporating the above mentioned modules have been shown below in Fig. 2.

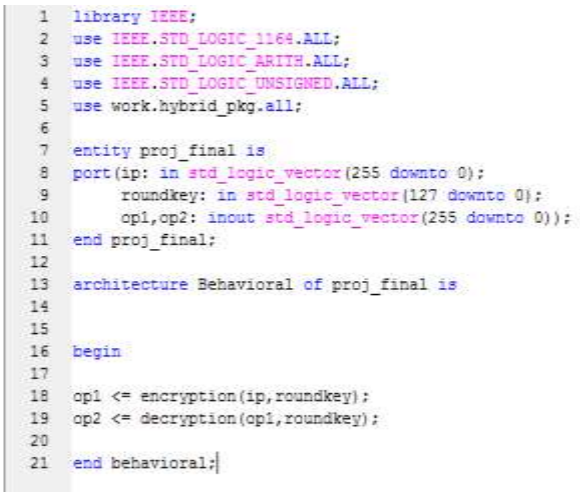


```
package hybrid_pkg is

subtype SLV_8 is std_logic_vector(7 downto 0);
subtype STATE_TYPE is std_logic_vector(127 downto 0);
subtype SLV_128 is std_logic_vector(127 downto 0);
subtype SLV_32 is std_logic_vector(31 downto 0);
subtype round_type is integer range 0 to 16;
subtype SLV_64 is std_logic_vector(63 downto 0);
subtype SLV_256 is std_logic_vector(255 downto 0);

function encryption(ip: SLV_256; roundkey: SLV_128) return SLV_256;
function decryption(ip: SLV_256; roundkey: SLV_128) return SLV_256;

function SBOX_LOOKUP (a: SLV_8) return SLV_8;
function INV_SBOX_LOOKUP (a: SLV_8) return SLV_8;
function BYTE_SUB_FUNCT (state: STATE_TYPE) return STATE_TYPE;
function INV_BYTE_SUB_FUNCT (state: STATE_TYPE) return STATE_TYPE;
function SHIFT_ROW_FUNCT (state: STATE_TYPE) return STATE_TYPE;
function INV_SHIFT_ROW_FUNCT (state: STATE_TYPE) return STATE_TYPE;
function MIX_COLUMN_FUNCT (state: STATE_TYPE) return STATE_TYPE;
function POLY_MULTE_FUNCT (a: SLV_8; b: SLV_8) return SLV_8;
function POLY_MULTD_FUNCT (a: SLV_8; b: SLV_8) return SLV_8;
function INV_MIX_COLUMN_FUNCT (state: STATE_TYPE) return STATE_TYPE;
function ADD_ROUNDKEY_FUNCT (roundkey, state: STATE_TYPE) return STATE_TYPE;
function ROUNDKEY_GEN (roundkey: STATE_TYPE; round: round_type) return STATE_TYPE;

function AES_FUNCT (ip: STATE_TYPE; roundkey: STATE_TYPE) return STATE_TYPE;
function INV_AES_FUNCT (ip: STATE_TYPE; roundkey: STATE_TYPE) return STATE_TYPE;
```

Fig 2: VHDL Package of modules

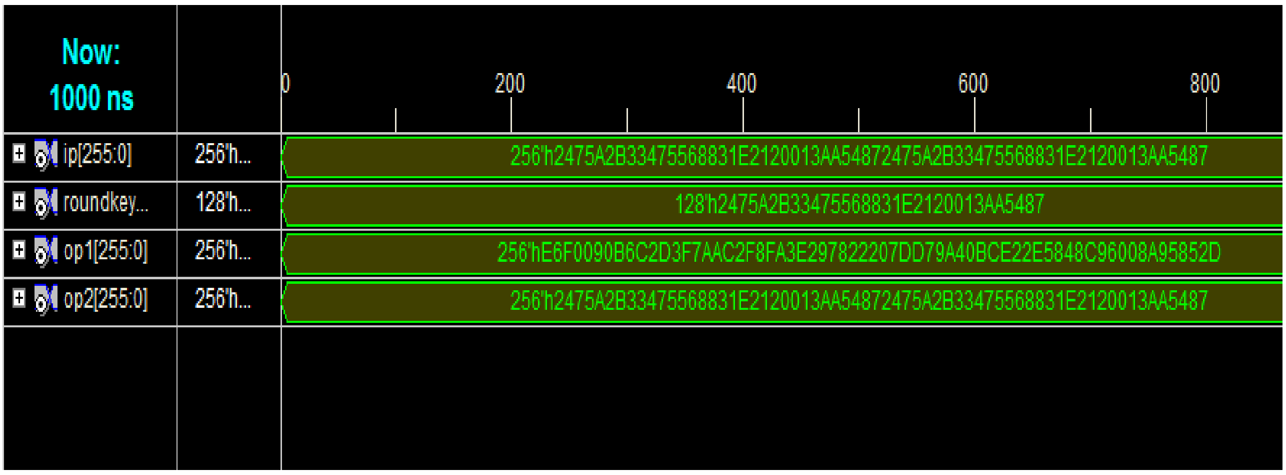
The main code of the proposed hybrid algorithm is shown in Fig 3. The function "encryption" is called which encrypts the 256 bit input data using the cipher key to give us "op1" which is the 256 bit ciphertext. The "decryption"function is called on this "op1" and the same cipher key is used decrypt the data which is channelled into "op2" giving the same plaintext.

The behavioural simulation of the main VHDL module was then carried out and the test bench waveforms validated the correctness of the code as shown in Fig 4.



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use work.hybrid_pkg.all;

entity proj_final is
port(ip: in std_logic_vector(255 downto 0);
     roundkey: in std_logic_vector(127 downto 0);
     op1,op2: inout std_logic_vector(255 downto 0));
end proj_final;

architecture Behavioral of proj_final is


begin

op1 <= encryption(ip,roundkey);
op2 <= decryption(op1,roundkey);

end behavioral;
```

Fig 3: VHDL module of Hybrid algorithm being tested



```
Now:
1000 ns                    0        200        400        600        800

ip[255:0]      256'h...   256'h2475A2B33475568831E2120013AA54872475A2B33475568831E2120013AA5487
roundkey...    128'h...   128'h2475A2B33475568831E2120013AA5487
op1[255:0]     256'h...   256'hE6F0090B6C2D3F7AAC2F8FA3E297822207DD79A40BCE22E5848C96008A95852D
op2[255:0]     256'h...   256'h2475A2B33475568831E2120013AA54872475A2B33475568831E2120013AA5487
```

Fig 4: Behavioural Simulation of main Hybrid VHDL module

IV. FPGA IMPLEMENTATION

The FPGA technology used to implement the proposed hybrid algorithm was Spartan 3 XC3S400 device with package PQ208. The limitation of using this kit was that the LEDs used to indicate the result were 16 in number and could only depict the last 2 bytes of the encrypted and decrypted data.

Fig 5. represents the last two bytes of the encrypted data “852Dh”. Fig 6. shows the last two bytes of the decrypted data “5487h”.

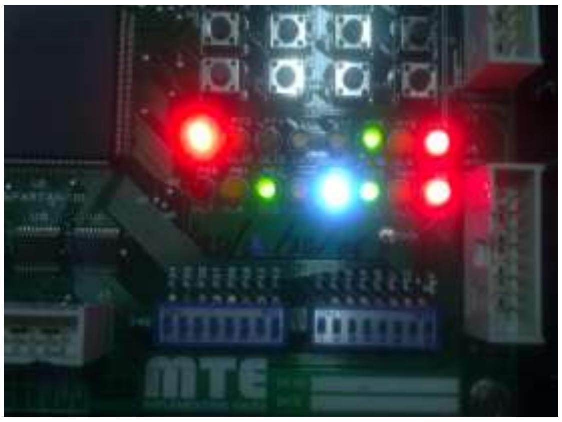


Fig 5: LEDs showing last 2 bytes of encrypted data- 852Dh

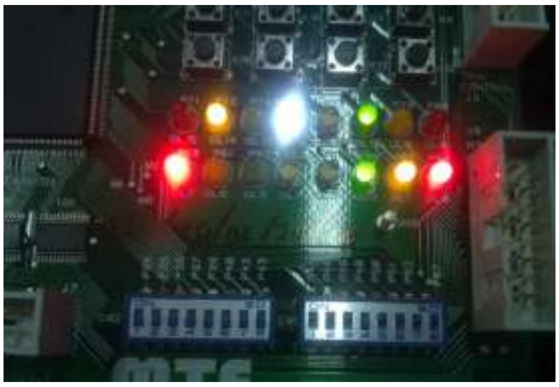


Fig 6: LEDs showing last 2 bytes ofdecrypted data- 5487h

V. FUTURE SCOPE

This proposedhybrid algorithm can be made much more powerful and secure by increasing the number of iterations in the encryption algorithm to suit the level of security required. An inverse policy of reducing the number of iterations for lower security can also be employed.

VI. CONCLUSION

In this paper, we have introduced the concept of an improved hybrid AES-DES as means of strengthening the current AES architecture. Implementation of such an algorithm results in a much more secure and attack resistant encryption algorithm which can be employed in varied fields such as wireless communications, electronic financial transactions, smart cards and video surveillance systems. We have also implemented the proposed algorithm in Spartan 3 FPGA kits thereby verifying the working of the algorithm in hardware.

VII. ACKNOWLEDGMENT

We would like to especially thank our project guide Prof. Surekha K S whose valuable suggestions helped shape the basis of our project idea.We would also thank our Director, Joint Director, Principal and HOD(E&TC) for supporting us during the research time. Our research is totally attributed to their guidance and support. We would also like to extend our gratitude to all the people and organizations directly or indirectly involved in this work. All these factors gave directions to our ideas.

VIII.REFERENCES

[1] Behrouz A. Forouzan, Cryptography and Network security, TMH

[2]M.B Vishnu, S.K. Tiong, Zaini M, Koh S P, “Security Enhancement of Digital Motion Image Transmission usingHybridAES-DESalgorithm,” 14th Asia-Pacific Conference onCommunications, APCC 2008, pp 1-5,2008

[3] Maire McLoone, John V. McCanny, ”HighPerformance Single Chip FPGA Rijndael Algorithm Implementations,” Proceedings of the Third International Workshop on Cryptographic Hardware & Embedded Systems , Springer-Verlag London UK, ISBN:3-54

[4]Sanchez-Avila, C.; Sanchez-Reillol R, “The Rijndael block cipher: A comparison with DES,” 35th IEEE International Carnahan conference on Security Technology, pp229-234, 2001

[5]McLoone, M. McCanny, J.V, “A high performance FPGA implementation of DES ,” IEEE Workshop on Signal Processing Systems, SiPS 2000,pp 374-383,2000

[6]Standaert, F.-X, Rouvroy G, Quisquater, J.-J,“ FPGA Implementations of the DES and Triple-DES Masked Against Power Analysis Attacks,” International conference on Field Programmable Logic and Applications, FPL '06. pp 1-4, 2006

[7]Saeid Taherkhani, Enver EveOrhanGemikonaklir, “Implementation of Non-Pipelined and PipelinedData Encryption Standard (DES) Using Xilinx Virtex -6 FPGA Technology,” 10thComputer & Information Technology(CIT 2010), pp 1257-1262, 2010

[8] Design of VLSI system by Dr Danial J. Miynek

[9] William Stallings, Cryptography and NetworkSecurity: Principles and Practice, 2nd ed., Prentice-Hall, Inc. 2000.

[10] http://www.xilinx.com