Encryption Mechanism for System-on-Chip using miniDES Algorithm with Digital Signature for Secure Communication of Embedded Real Time Applications

KURRA MALLAIAH

Advanced Numerical Research & Analysis Group
(ANURAG),
DRDO, Hyderabad, Andhra Pradesh, india-500058

Abstract-- **System-on-Chip applications have been widen to various public, private, government and social communication systems for handling security and trust as a systems for mission critical and e-government. Most mission critical and embedded real time applications run on System -on -Chip. SoCs will have limited resources, processing ability and rigid power constraints therefore, SoCs becomes a holdup when it comes to handling the complex and large footprint software [3]. saRTL is standalone Real Time Linux operating system with small footprint and having hard real time features. The bare kernel of saRTL is approximately 200KB only. Therefore using saRTL is a better option for SoC based applications. miniDES is a symmetric key encryption algorithm with very small footprint. Approximately, the size of algorithm is 3KB only and it has the power and elegance of Data Encryption Standard algorithm (DES). Since, SoCs used in critical and Real Time embedded applications, security has to be enforced in the SoCs and there is a need to provide confidentiality, integrity and authentication for SoC based applications, especially, in the area of Embedded Real Time and critical communication applications.**

In this paper, I explore a concept of TCP/IP packet IP level encryption mechanism using miniDES with Digital Signature for securing information, while transmitting using saRTL running on SoCs. This work aims to show that encryption mechanism of IP payload provides confidentiality, authentication and integrity of applications running on SoCs using miniDES with Digital Signature.

Keywords: **Digital Signature, miniDES, standalone Real Time Linux, System -on-Chip, TCP/IP**

1. saRTL

saRTL offer an environment fully attuned with Real Time Linux that fit memory requirements for small and medium embedded systems. The same application code runs faster on a sa-RTL system than on standard RTLinux, because Linux kernel and applications do not cause processor cache pollution and access to main memory is faster since no address conversions is done due to paging is not enabled[1]. Stand-Alone RTLinux will be a Linux independent implementation of RTLinux. Stand-Alone RTLinux will be a bootable executive with the following main features:

- It has the small footprint
- Low memory overhead
- open source software
- Microkernel
- Has the features of Real-time Operating system
- Scalability
- Porting will be possible to systems without hardware for virtual memory support.
- Less TLB and memory cache misses.

We are not longer limited by the Linux memory manager. That is, it is possible to implement memory protection schemes

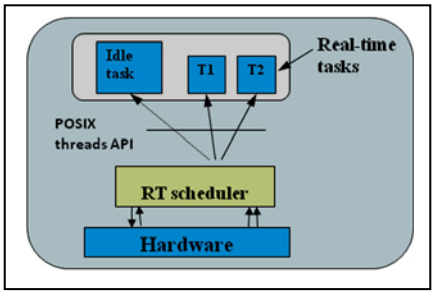


Figure 1.1 Architecture of saRTL

saRTL supports the POSIX standard I/O, RATE MONOTONIC ALGORITHM for real-time task scheduling. Architecture of saRTL is shown in fig.1.1. It has limited device driver support. Ex: display, keyboard, serial, timer and some network drivers. The approximate size of the bare saRTL kernel is 200KB only, which can be easily

Proc. of the Intl. Conf. on Advances in Computer Science and Electronics Engineering
Editor In Chief Sahil Seth.
ISBN: 978-981-07-1403-1

accommodated in embedded systems for executing Real Time applications on the SoCs.

2. miniDES

miniDES is a symmetric key encryption algorithm. Time complexity of this algorithm is O(n), where n is the size of the input characters. In symmetric key encryption, you and your friend share a "secret" key. Using this key, you can encrypt a message into "cipher text". Cipher text looks like a random sequence of characters and is completely meaningless to anyone unless they also have the secret key, in which case they can decrypt the cipher text back into the original message and read it. Fig 2 illustrates symmetric encryption method. Symmetric cryptography also provides a degree of authentication because data encrypted with one symmetric key cannot be decrypted with any other symmetric key. Therefore, as long as the symmetric key is kept secret by the two parties using it to encrypt communications, each party can be sure that it is communicating with the other as long as the decrypted messages continue to make sense. miniDES algorithm takes 8bit plain text as input and this input is divided into two groups each of four bits, such as upper four bits into L0 and lower four bits into R0. This algorithm takes three rounds with a swap to encrypt a character.

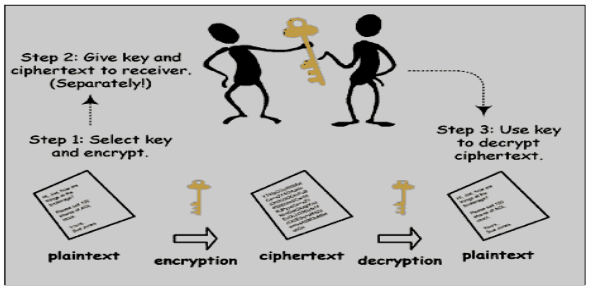


Figure 2: Symmetric key Encryption

miniDES Encryption

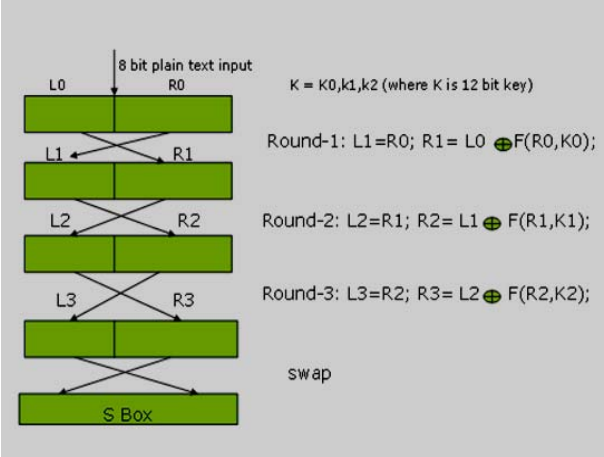
Received 8-bit plain text is sub divided into two groups, each of with four bits. Move upper four bits into L0 and lower four bits into R0. A 12bit key has also sub divided into three groups, such as k0, k1, k2, each of with four bits. Consider the example, k=101010101010, this key sub divided into k0=1010; k1=1010; K2=1010. These sub keys are being used in encryption of 8-bit plain text.

Round 1: Copy four bits of R0 into L1 and R1=L0 exclusive or with F(R0,k0).In round 1, key k0 is used.

Round 2: Copy four bits of R1 into L2 and R2=L1 exclusive or with F(R1,k1).In round 2, key k1 is used.

Round 3: Copy four bits of R2 into L3 and R3=L2 exclusive or with F(R2,k2). In round 3, key k2 is used

Swap: swap L3, R3 contents which is the cipher text of 8-bit input



Calculation of F(R0,K0) is as follows:

i) Let us assume tha, content of R0 four bits are b1b2b3b4, then left shift or right shift R0 contents i.e. R0=b2b3b4b1(left shift). Now find Exclusive or of R0,K0 i.e. W=R0 ⊕ K0

ii) Now, let us assume that the four bits of W are 1010 (assume this as ABCD)from the 4x4 two dimensional table select the number corresponding to [AD][BC] i.e. [10][01]. The table contains the 16 hexadecimal numbers in any order starting from 0 to F as shown in table 1. The [10][01] corresponding number in the table 1 is **8.**

iv) Therefore F(R0,K0) is 8

v) Now, calculate R1=L0⊕F(R0,K0)

vi) Similarly, calculate R2 & R3 in round 2 and round 3 and then swap the L3 and R3 which gives the cipher text.

miniDES Decryption

miniDES, decryption processes takes an 8 bit cipher text as input and does operations same as encryption process, but key is taken in reverse order. i.e. from k2 to k0.

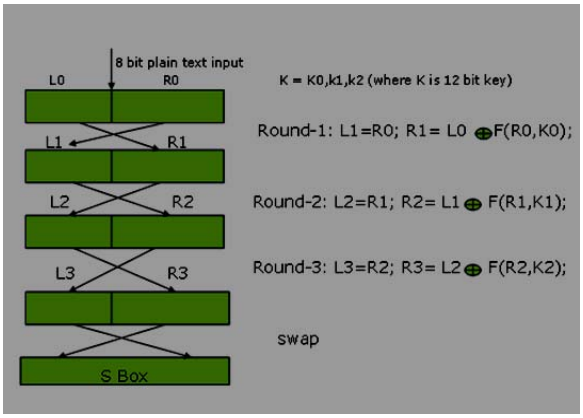
Round 1: Copy four bits of R0 into L1 and R1=L0 exclusive or with F(R0,k2), in round 1, the key k2 is used.

Round 2: Copy four bits of R1 into L2 and R2=L1 exclusive or with F(R1,k1), in round 2, the key k1 is used.

Round 3: Copy four bits of R2 into L3 and R3=L2

exclusive or with F(R2,k0), in round 3, key k0 is used.

Swap: swap L3, R3 contents, which is the decrypted input of 8-bits



Calculation of F(R0,K2) is as follows:

i) Let us assume that content of R0 four bits are b1b2b3b4 then left shift or right shift the R0 contents i.e. R0=b2b3b4b1(left shift).

ii) Now, find the Exclusive or of R0, K2 i.e. W=R0⊕K2

iii) Now let us assume that the four bits of W are 1100 (assume this as ABCD) from the 4x4 two dimensional table select the number corresponding to [AD][BC] i.e. [10][10]. The table contains the 16 hexadecimal numbers in any order stating from 0 to F as shown in table 1. [10][10], Corresponding number in table 1 is E

iv) Therefore F(R0,K2) is E

v) Now, calculate R1=L0⊕F(R0,K2)

vi) Similarly, calculate R2 & R3 in round 2 and round 3 and swap L3 and R3, which gives decrypted 8-bits plaintext.

TABLE 1. 0 TO F 16 HEXADECIMAL DIGITS

	00	01	11	10
00	0	1	3	4
01	7	9	B	D
11	5	A	C	6
10	2	8	F	E

3. The Digital Signature approach

Digital signature is an authentication mechanism that enables the creator of a message to attach a code that acts as a signature [2]. The signature is created by taking the hash of the message and encrypting the message with miniDES with shared key of sender. The signature guarantees the source and integrity of the message.

The message to be signed is input to the hash function that provides a secure hash code of fixed length. This code is then encrypted using the shared symmetric key to form a digital signature. Both encrypted message using shared symmetric key and Digital Signature are then transmitted. The recipient decrypts the message with symmetric key and produces a hash code of it. The recipient also decrypts Digital Signature using the symmetric key. If the calculated hash code matches the decrypted signature, the signature is accepted as valid. Because only the sender knows the shared key, only the sender could have produced a valid signature [2]. Here message is also secured because it only could be possible decrypting it with the shared key.

4. Security requirements of System-on-Chip

Information is vital in any organization, especially, in Embedded Real time and critical communication applications. Therefore information has to be communicated in a secure manner with zero tolerance of assault. Having limited resources, SoCs may not afford to run very complex or large footprint software. Software like, Linux operating may not be an option in this type of environment. saRTL in such environment is a better option, because of its rich set of features mostly suites for the embedded kind of systems. Application can be integrated along with kernel, and it has the hard real time operating system features, which are essential for the Real Time embedded applications. Secure data communication is also achieved for SoCs based applications by providing encryption mechanism at the IP level using miniDES with Digital Signature.

5. Encryption mechanism using miniDES with Digital Signature

In this paper, proposing a mechanism called miniDES with Digital signature for encryption of the IP packet payload for the saRTL based SoCs, which ensures Authentication of the sender, integrity of IP payload and confidentiality. miniDES combined with Digital Signature provides the Authentication and integrity of IP payload and miniDES encryption provides the confidentiality.

Encryption IP payload at sender

Fig. 5.1 illustrates encryption of IP payload using miniDES with Digital signature. Secure Hash Algorithm (SHA) is applied to the payload of IP

packet to produce the hashed Payload **HP. HP** is then encrypted using shared key using miniDES algorithm, which results DS. The payload of the IP packet is next encrypted using shared key, then produced DS is attached to it, to complete encryption process. From Internet layer this message is passed to the Data link layer with IP header.

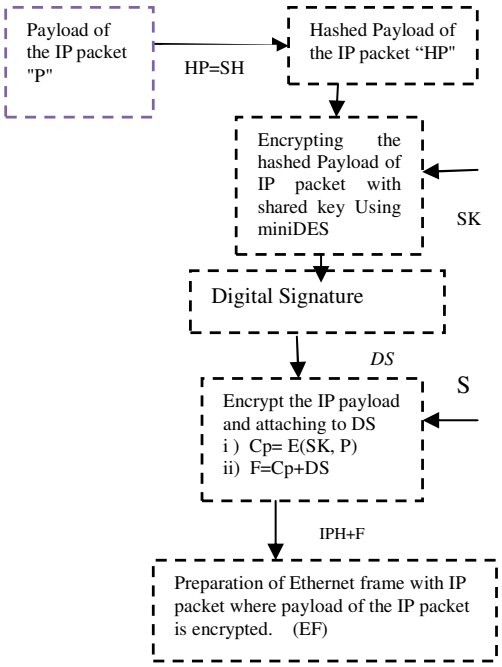


Figure 5.1 Encryption at sender

Operational procedure for Encryption of IP payload at sender.

i) **P:** Payload of the IP Packet

ii) **HP**: Result of the Secure Hash Algorithm
HP=SHA(P)

iii) **DS:** Digital Signature
DS=E(SK,HP)
Where **SK** is the shared key

iv) Cp = E(SK,**P**)

v) **Cp :** Result of Encryption of the IP payload

vi) **F :** Combining the encrypted payload with Digital signature
F= Cp + DS

vii) **IPH+F:** IP header with the encrypted digital signature

viii) **EF:** Ethernet Frame

Decryption IP payload at receiver

Fig 5.2 illustrates decryption of the IP payload using miniDES with Digital Signature. On receiving IP packet at internet layer of TCP/IP protocol, the message is segregated into encrypted IP payload message and DS. The DS is decrypted using the shared key and encrypted IP payload message is also decrypted using shared key. This is resulted to "Y" and "P" respectively. Now, the message "P" is hashed using SHA to produce "X". Finally, "X" is compared with "Y" if both are same, and then source and integrity of the message is maintained and authentication of the sender are valid and accepted.

Operational Procedure of the Decryption at receiver:

i) Segregation of received data from data link layer into payload message and DS.

ii) **EIP**: Encrypted IP payload message

iii) **P=D(SK,EIP)**. Decrypting the IP payload , where SK is the shared key.

iv) **Y= D(SK,DS)**: Decrypting the DS

v) **X**=SHA(P); Applying Secure Hashing Algorithm on P

vi) IF(X=Y) then source and integrity of the message is maintained authentication is valid and accepted else Authentication is Failed

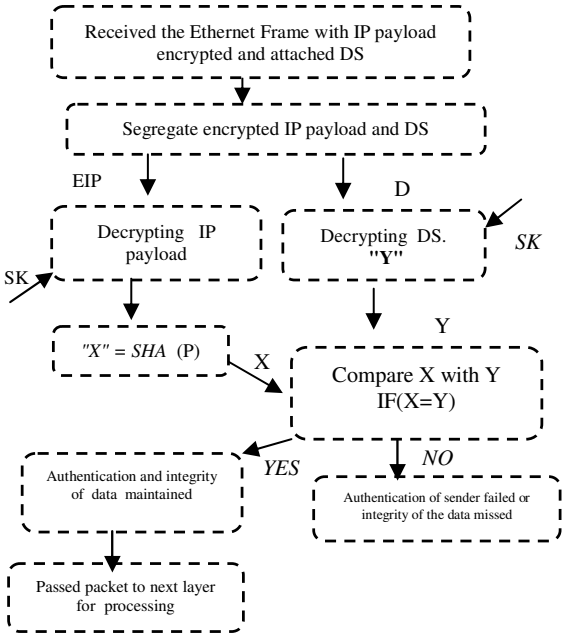


Figure 5.2 Decryption at receiver

Fig. 5.3 illustrates Encryption and decryption of the IP payload using miniDES with Digital Signature. The miniDES with DS is integrated into network layer of the TCP/IP stack at transmitter and receiver ends.

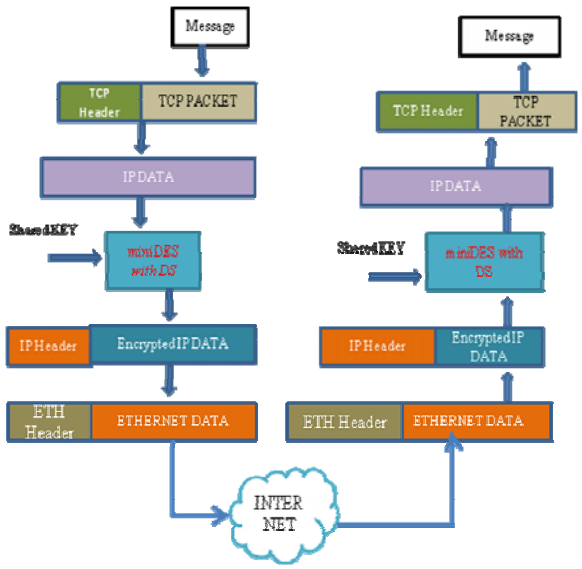


Figure 5.3: Encryption and Decryption integration of miniDES with DS in TCP/IP

Authentication, confidentiality, Integrity with miniDES and DS

For data communication in mission critical and Real Time embedded applications, CIA (Confidentiality, integrity, Authentication) has to be ensured. Where miniDES alone provides only Confidentiality, and miniDES with Digital signature provides CIA as shown in the table 2

miniDES: *Confidentiality* is achieved with miniDES. Let us assume that P is the IP payload. Now P is encrypted with the secrete key which is shared by the both sender and receiver.

C= E(SK,P); P is encrypted with SK; SK is the shared key. It is possible to only decrypt the cipher text C only with key is used for the encryption. Even attacker gains the access to the cipher text, he cannot able be decrypt it because he does not have the shared key. This way it ensures the confidentiality of the IP packet payload.

P= D(C, SK); decrypting cipher text C with shared key, which retrieves original IP payload P.

With miniDES integrity is not possible because it does not check for any modifications in the received cipher

text. Authentication is also not possible because the receiver has no way of validating the sender.

miniDES with DS*: Confidentiality, integrity and authentication* is possible using miniDES with Digital signature.

DS = E(SK,(SHA(P))); → Digital Signature for the message P

C= E(SK,P); → Encrypting Message P into Cipher Text C

F= DS+C; where F is the frame

P= D(SK, C); Decryption of Cipher text C

S = D(SK,DS); Decryption of Digital Signature.

R= SHA(P);

if(R=S), i.e., the hash of the message generated at the receiver matches the hashed message sent by the transmitter. This ensures integrity and authentication of IP payload and sender respectively. This is because the DS is decrypted using the shared key to retrieve the hash of the original message, and this retrieved hash is matched against the hash of the decrypted IP payload. IP payload is encrypted with shared key therefore confidentiality is ensured, because valid receiver can only decrypt with shared key. Therefore with help of miniDES with Digital signature CIA is possible as shown in the table 2.

TABLE 2: CIA

	miniDES	**miniDES + DS**
Confidentiality	Yes	Yes
Integrity	No	Yes
Authentication	No	Yes

6. Conclusion and future scope

System-on-Chip are playing a crucial role in various public, private and social communication systems, for handling security and trust as a systems for mission critical and e-government. Since SoCs are having limited resources, a light weight operating system and a small footprint crypto algorithm are better option for providing secure communication. saRTL is a light weight operating system with hard real time features and miniDES is symmetric key encryption algorithm having very small footprint of 3KB only and it has the power and elegance of DES. Since, SoC are used in embedded Real time and critical communication applications, integrity, confidentiality and authentication of these SoCs has to be ensured. The proposed encryption mechanism using miniDES with Digital Signature ensures the Confidentiality with symmetric key encryption called miniDES and Authentication and integration using miniDES in combination with Digital signature. This work can be further enhanced by exploring other proprietary/ indigenous encryption mechanisms for enhancing CIA of applications running on SoCs. RSA with Digital signature also can be used for the CIA of network security for applications running on SoCs.

8. References

[1] Stand-Alone RTLinux Internals Vicente Aurelio Esteve Lloret viesllo@inf.upv.es

[2] William Stallings,“Cryptography and Network Security Principles and practices" fourth Edition 'pgno. 378 & 390 Prentice Hall, PEARSON

[3] A Low-Complexity Heterogeneous Multi-Core Platform for Security SoC, Wei Huang, Jun Han, Shuai Wang and Xiaoyang Zeng, IEEE Asian Solid-State Circuits Conference November 8-10,2010 / Beijing, China