Secure Data Transmission using Blowfish Algorithm

Panem.Charan Arur[1], M. Sai Chandrasekhar[2] , S.Sai Sreeram[3] , K.RamKishore[4] , S.Venu gopal[5] ,
[1]Assistant professor, [2]UG Students[B.Tech]
Dept. of ECE,
Priyadarshini Institute of Technology, Nellore.

***Abstract*—Data Security is primary concern for every communication system. There are many ways to provide security data that is being communicated. However, what if the security is assured irrespective of the hackers are from the noise. This Project describes a design of effective security for data communication by designing standard algorithm for encryption and decryption. The data transformation process for Pocket Brief uses the BLOWFISH algorithm for encryption and decryption.**

Keyword s : Encryption Algorithm, Performance, Analysis, AES, DES, Blowfish, Triple DES, Cryptography

I. INTRODUCTION

The purpose of this project (Zigbee Based Data EncriptionAndDecreption System) is to build an embedded system hardware which can encrypt and decrypt the text for secured wireless transmission using the blowfish algorithm. It is using the Zigbee technology for the transmission.

Existing system:

Currently, there are lot of encryption and decryption algorithms like RSA, AES, DES, etc. but among them the blow fish is the best and high secured algorithm.

Proposed system and its advantages:

In this system we are using the blowfish algorithm for data encryption and decryption to transmit the data wirelessly through the Zigbee module. In this system we are using the two PCs as terminals, the PCs are interfaced to the microcontrollers which accepts the data entered through the PC, encrypt them and transmit them through the Zigbee transceiver. On the other side the received data through the Zigbee transceiver is decrypted by the other controller and transmitted to the PC for the display.

Microcontroller (ATMEGA 328):

The microcontroller we used here is an 8-bit microcontroller, which means it can process the 8-bit data at a time. The controller we are using here is the ATMEGA328 which have the 14 digital pins through which we can perform the digital operations and 6analog pins through which we can perform analog read operation which is done by the inbuilt analog to digital converters in that controller which are of 10-bit resolution. The clock frequency we are using here is 16Mhz which gives us the good speed of execution for our applications.

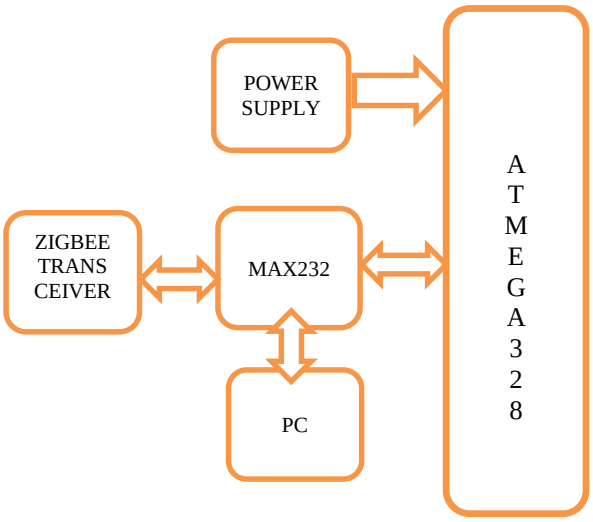
Zigbee Transceiver:

ZIGBEE is a specification for a suite of high level communication protocols using small, low-power digital radios based on the IEEE 802.15.4-2003 standard for Low-Rate Wireless Personal Area Networks (LR-WPANs). ZIGBEE is targeted at radio-frequency (RF) applications that require a low data rate, long battery life, and secure networking. ZIGBEE protocols are intended for use in embedded applications requiring low data rates and low power consumption. ZIGBEEs current focus is to define a general-purpose, inexpensive, self-organizing mesh network that can be used for industrial control, embedded sensing, medical data collection, smoke and intruder warning, building automation, home automation, etc. The resulting network will use very small amounts of power.

The project is designed in such a way that an X-BEE transceiver will be interfaced to a PC serially using a driver IC MAX232, to input the predefined data from PC to the X_BEE module, on the transmitter side. And on the receiver side the controller will be interfaced to another X-BEE transceiver which can receive the transmitted data from the remote transceiver, and to the ac loads through relays. Whenever a predefined data is transmitted from the PC to the transceiver on the transmitter side , the same data will be wirelessly transmitted to the remote transceiver and will be fed to the controller.

II. SYSTEM MODEL

TRANSMISSION SECTION:



RECEIVER SECTION:

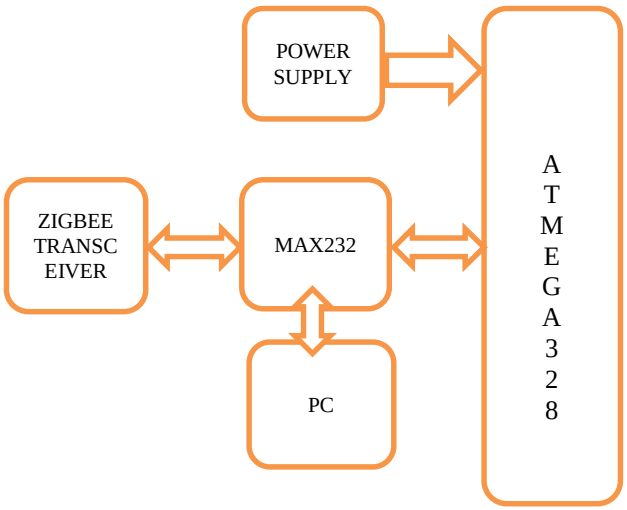


Fig1: Proposed system block diagram

.

The Zigbee modules interface to a host device through a logic-level asynchronous serial port. Through its serial port, the module can communicate with any logic and voltage compatible UART or through a level transistor to any serial device (For example : RS-232 or USB interface board).These Zigbee modules are in the form of Zigbee or X-bee or Tharang.

The project is designed in such a way that one ZIGBEE transceiver will be interfaced to the PC through serial communication, so that we can input the data to the controller using the hyper terminal of PC. Here we will use a serial line driver IC MAX232 to interface the PC with controller.

This project uses regulated 5V,500mA power supply. 7805 three terminal voltage regulator is used for voltage regulation. Full wave bridge rectifier is used to rectify the ac output of secondary 230/12V step down transformer.

III. BLOWFISH ALGORITHM

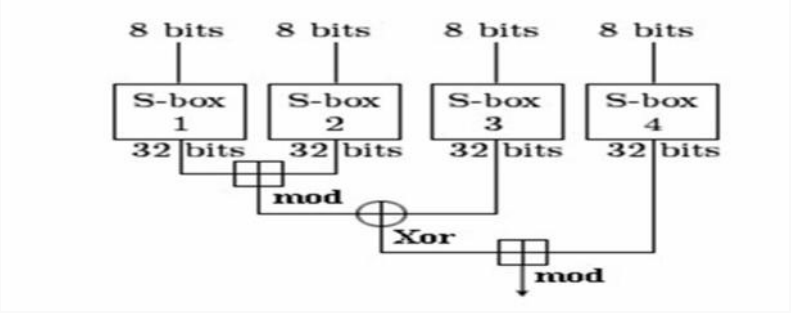
Blowfish is a symmetric-key block cipher, designed in 1993 by Bruce Schneier and included in a large number of cipher suites and encryption products. Blowfish provides a good encryption rate in software and no effective cryptanalysis of it has been found to date. However, the Advanced Encryption Standard (AES) now receives more attention.

Schneier designed Blowfish as a general-purpose algorithm, intended as an alternative to the aging DES and free of the problems and constraints associated with other algorithms. At the time Blowfish was released, many other designs were proprietary, encumbered by patents or were commercial or government secrets. Schneier has stated that, "Blowfish is unpatented, and will remain so in all countries. The algorithm is hereby placed in the public domain, and can be freely used by anyone."

Notable features of the design include key-dependent S-boxes and a highly complex key schedule.

The Algorithm:

Blowfish has a 64-bit block size and a variable key length from 32 bits up to 448 bits.It is a 16-round Feistel cipher and uses large key-dependent S-boxes. In structure it resembles CAST-128, which uses fixed S-boxes.



The Feistel structure of Blowfish

The diagram to the left shows the action of Blowfish. Each line represents 32 bits. The algorithm keeps two subkey arrays: the 18-entry P-array and four 256-entry S-boxes. The S-boxes accept 8-bit input and produce 32-bit output. One entry of the P-array is used every round, and after the final round, each half of the data block is XORed with one of the two remaining unused P-entries.

The diagram to the upper right shows Blowfish's F-function. The function splits the 32-bit input into four eight-bit quarters, and uses the quarters as input to the S-boxes. The outputs are added modulo 232 and XORed to produce the final 32-bit output.

Blowfish's key schedule starts by initializing the P-array and S-boxes with values derived from the hexadecimal digits of pi, which contain no obvious pattern (see nothing up my sleeve number). The secret key is then, byte by byte, cycling the key if necessary, XORed with all the P-entries in order. A 64-bit all-zero block is then encrypted with the algorithm as it stands. The resultant ciphertext replaces P1 and P2. The same ciphertext is then encrypted again with the new subkeys, and the new ciphertext replaces P3 and P4. This continues, replacing the entire P-array and all the S-box entries. In all, the Blowfish encryption algorithm will run 521 times to generate all the subkeys - about 4KB of data is processed.

Because the P-array is 576 bits long, and the key bytes are XORed through all these 576 bits during the initialization, many implementations support key sizes up to 576 bits. While this is certainly possible, the 448 bits limit is here to ensure that every bit of every subkey depends on every bit of the key, as the last four values of the P-array don't affect every bit of the ciphertext. This point should be taken in consideration for implementations with a different number of rounds, as even though it increases security against an exhaustive attack, it weakens the security guaranteed by the algorithm. And given the slow initialization of the cipher with each change of key, it is granted a natural protection against brute-force attacks, which doesn't really justify key sizes longer than 448 bits.

IV. EFFICIENT COMMUNICATION

An application may consist of communicating objects which cooperate to carry out the desired tasks. The focus of ZigBee is to distribute work among many different devices which reside within individual ZigBee nodes which in turn form a network (said work will typically be largely local to each device, for instance the control of each individual household appliance).

The collection of objects that form the network communicate using the facilities provided by APS, supervised by ZDO interfaces. The application layer data service follows a typical request-confirm/indication-response structure. Within a single device, up to 240 application objects can exist, numbered in the range 1-240. 0 is reserved for the ZDO data interface and 255 for broadcast; the 241-254 range is not currently in use but may be in the future.Two services are available for application objects to use (in ZigBee 1.0):

The key-value pair service (KVP) is meant for configuration purposes. It enables description, request and modification of object attributes through a simple interface based on get/set and event primitives, some allowing a request for response. Configuration uses compressed XML (full XML can be used) to provide an adaptable and elegant solution.

The message service is designed to offer a general approach to information treatment, avoiding the necessity to adapt application protocols and potential overhead incurred on by KVP. It allows arbitrary payloads to be transmitted over APS frames.

Addressing is also part of the application layer. A network node consists of an 802.15.4-conformant radio transceiver and one or more device descriptions (basically collections of attributes which can be polled or set, or which can be monitored through events). The transceiver is the base for addressing, and devices within a node are specified by an endpoint identifier in the range 1-240.

Communication and device discovery:

In order for applications to communicate, their comprising devices must use a common application protocol (types of messages, formats and so on); these sets of conventions are grouped in profiles. Furthermore, binding is decided upon by matching input and output cluster identifiers, unique within the context of a given profile and associated to an incoming or outgoing data flow in a device. Binding tables contain source and destination pairs.

This extended discovery protocol permits external devices to find out about devices in a network and the services that they offer, which endpoints can report when queried by the discovering device (which has previously obtained their addresses). Matching services can also be used.

The use of cluster identifiers enforces the binding of complementary entities by means of the binding tables, which are maintained by ZigBee coordinators, as the table must be always available within a network and coordinators are most likely to have a permanent power supply. Backups, managed by higher-level layers, may be needed by some applications. Binding requires an established communication link; after it exists, whether to add a new node to the network is decided, according to the application and security policies.

Communication can happen right after the association. Direct addressing uses both radio address and endpoint identifier, whereas indirect addressing uses every relevant field (address, endpoint, cluster and attribute) and requires that they be sent to the network coordinator, which maintains associations and translates requests for communication.Indirect addressing is particularly useful to keep some devices very simple and minimize their need for storage. Besides these two methods, broadcast to all endpoints in a device is available, and group addressing is used to communicate with groups of endpoints belonging to a set of devices

A. SECURITY

ZigBee provides facilities for carrying out secure communications, protecting establishment and transport of cryptographic keys, ciphering frames and controlling devices. It builds on the basic security framework defined in IEEE 802.15.4. This part of the architecture relies on the correct management of symmetric keys and the correct implementation of methods and security policies. Within the protocol stack, different network layers are not cryptographically separated, so access policies are needed and correct design assumed.

The open trust model within a device allows for key sharing, which notably decreases potential cost. Nevertheless, the layer which creates a frame is responsible for its security. If malicious devices may exist, every network layer payload must be ciphered, so unauthorized traffic can be immediately cut off. The exception, again, is the transmission of the network key, which confers a unified security layer to the network, to a new connecting device.

B. PERFORMANCE RESULTS

The first set of experiments were conducted using ECB mode, the results are shown in figure 8 below. The results show the superiority of Blowfish algorithm over other algorithms in terms of the processing time. It shows also that AES consumes more resources when the data block size is relatively big. The results shown here are different from the results obtained by since the data block sizes used here are much larger than the ones used in their experiment.

Another point can be noticed here that 3DES requires always more time than DES because of its triple phase encryption characteristic. Blowfish ,although it has a long key (448 bit) , outperformed other encryption algorithms. DES and 3DES are known to have worm holes in their security mechanism, Blowfish and AES, on the other hand, do not have any so far. These results have nothing to do with the other loads on the computer

since each single experiment was conducted multiple times resulting in almost the same expected result. DES, 3DES and AES implementation in .NET is considered to be the best in the market.

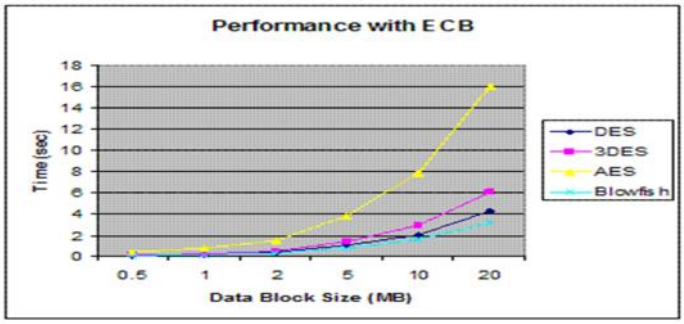


Fig .Performance Results with ECB Mode

As expected CBC requires more processing time than ECB because of its key-chaining nature. The results show in Fig. 9 indicates also that the extra time added is not significant for many applications, knowing that CBC is much better than ECB in terms of protection. The difference between the two modes is hard to see by the naked eye, the results showed that the average difference between ECB and CBC is 0.059896 second, which is relatively small.

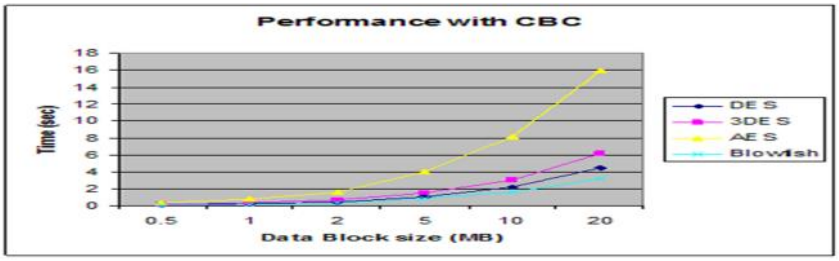


Fig. Performance Results with CBC Mode

This section showed the simulation results obtained by running the four compared encryption algorithms using different Cipher Modes. Different load have been used to determine the processing power and performance of the compared algorithms.

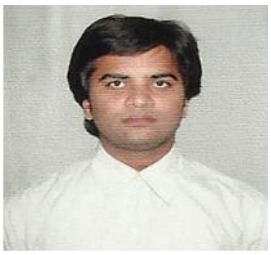
IV. CONCLUSION

The presented simulation results showed that Blowfish has a better performance than other common encryption algorithms used. Since Blowfish has not any known security weak points so far, which makes it an excellent candidate to be considered as a standard encryption algorithm. AES showed poor performance results compared to other algorithms since it requires more processing power. Using CBC mode has added extra processing time, but overall it was relatively negligible especially for certain application that requires more secure encryption to a relatively large data blocks

REFERENCES

[1] [RFC2828],"Internet Security Glossary", http://www.faqs.org/rfcs/rfc2828.html
[2] [Nadeem2005]Aamer Nadeem et al, "A Performance Comparison of Data Encryption Algorithms", IEEE 2005
[3] [Earle2005] "Wireless Security Handbook,". Auerbach Publications 2005
[4] [Dhawan2002] Priya Dhawan., "Performance Comparison: Security Design Choices," Microsoft Developer Network October 2002. http://msdn2.microsoft.com/en-us/library/ms978415.aspx
[5] [Edney2003]," Real 802.11 Security: Wi-Fi Protected Access and 802.11i ,". Addison Wesley 2003
[6] [Wikipedia-BC] "Block Cipher", http://en.wikipedia.org/wiki/Block_cipher

[7] [Hardjono2005]," Security In Wireless LANS And MANS ,". Artech House Publishers 2005
[8] [TropSoft] "DES Overview", [Explains how DES works in details, features and weaknesses]
[9] [Bruce1996] BRUCE SCHNEIER, "Applied Cryptography" , John Wiley & Sons, Inc 1996
[10] [Crypto++]"Crypto++ benchmark", http://www.eskimo.com/~weidai/benchmarks.html [Results of comparing tens of encryption algorithms using different settings].
[11] [BlowFish.NET] "Coder's Lagoon",http://www.hotpixel.net/software.html [List of resources to be used under GNU]
[12] D. I. Inc. (2013, accessed on 6 October, 2012). ZigBee Wireless Standard. Available: http://www.digi.com/technology/rf-articles/wireless- zigbee
[13] "Issues in Wireless Sensor Networks," in World Congress on Engineering, London, U.K, 2008



Panem Charan Arur.He did M.Tech (VLSI System Design) and B.Tech(ECE).Now working as a Assistant Professor in ECE department at Priyadarshini Institute of Technology(PINN),SPSR NelloreAP,India.Doing Research Work on Low Power VLSI. Published Three InterNational Journal,Attended one InterNational conference and Three national level conference and two national level technical seminars,two national level workshops.Professional Association member ships IAENG,CSIT,IACSIT. He has a review committee member in three International Journals.Now he doing research on advanced technologies in VLSI and Embedded systems.Email:panem.charan@gmail.com.

M.Sai Chandra Sekhar Studying B.Tech(ECE) at Priyadarshini Institute Of Technology(PINN),SPSR Nellore,AP,and doing projectwork on SECURE DATA TRANSMISSION USING BLOWFISH ALGORITHM Email: saichandrasekhar@outlook.com

S,Sai Sreeram Studying B.Tech(ECE) at Priyadarshini Institute Of Technology(PINN),SPSR Nellore,AP,and doing projectwork on SECURE DATA TRANSMISSION USING BLOWFISH ALGORITHM.. Email: sreeram.sudepalli@gmail.com

K.Ramkishore Studying B.Tech(ECE) at Priyadarshini Institute Of Technology(PINN),SPSR Nellore,AP,and doing projectwork on SECURE DATA TRANSMISSION USING BLOWFISH ALGORITHM Email: kalahastriramkishore@gmail.com

S.Venugopal Studying B.Tech(ECE) at Priyadarshini Institute Of Technology(PINN),SPSR Nellore,AP,and doing projectwork on SECURE DATA TRANSMISSION USING BLOWFISH ALGORITHM.

Email:venugopal.s117@gmail.com