IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735. Volume 7, Issue 6 (Sep. - Oct. 2013), PP 63-70
www.iosrjournals.org

Wireless Network Security Architecture with Blowfish Encryption Model

Subrahmanyeswararao Maradani[1], Seetharamanjaneyulu Babburi[2]
[1]*Dept. of E.C.E, VignanUniversity, Andhra Pradesh, India* [2]*Dept. of E.C.E, VignanUniversity, Andhra Pradesh, India*

Abstract: *In this research paper ,we developed a model for a large network, wireless nodes are interconnected and each can be considered as a node processor that offer services to other node processors connected to a specific network. A very high proportion of the nodes that offer services need to carry out an authentication process so as to make an access request to the node offering the service. In this context, an integrated reconfigurable network security architecture moved to the application layer has become the need of the day for secure wireless data sharing. The security schemes of the seven layer OSI architecture need to be placed intrinsically in the wireless node itself and should be capable of supporting the MAC layer, IP address based layer and the routing protocols of the network layer. This work focuses on the use of emulator and embedded hardware architectures for wireless network security. In this work, the individual nodes can have a unique security signature pattern maintained by respective wireless nodes using an encryption algorithm and this is made dynamic. The metrics includes latency, throughput, Scalability, Effects of data transfer operation on node processor and application data located in the processor*

Keywords:*Wireless Network security, Embedded hardware, Reconfigurable architecture, blowfish algorithm*

I. Introduction

The researchers have agreed that security is very significant issue for network where difficult to manage whole network at a time with all users. Security can be implemented at various levels of the intercommunications which are established on a physical layer and goes up via the data link, network, and transport layers up to the topmost application layer. In this research work, the nodes can be dynamic and can join or leave a network at any time. In order to withstand malicious attacks, the end-to-end communication is secured using cryptographically strong authentication. The attacks can transmit malicious information from a node to the destination to intercept information and unwanted attacks can record packets at a location in the network to secure data

1.1 Application Layer Security with Embedded System

An application layer security application is secure shell, which allows a secure login for administration and monitoring purposes. A possible security approach at the application level is to authenticate and encrypt the information packages from secure system, but be aware that additional protection avoids replay attacks. An embedded system based wireless node architecture is shown in Figure 1.

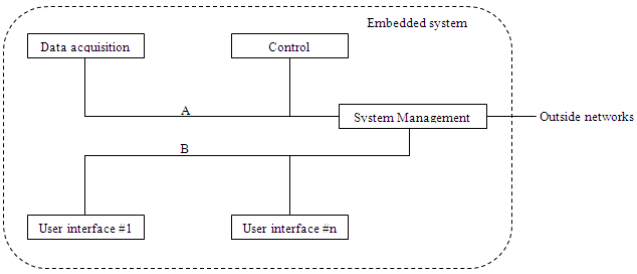


Fig. 1. Example embedded system based wireless node architecture

II. Data Structure

2.1 Communication in MAC layer

Table.1. Data Structure in Mac Layer

Source MAC Authenticating	Destination MAC	Beacon	Raw Data	32-Bit
(6 bytes)	(6 bytes)	(1 byte)	(N bytes)	(4 bytes)

The data structure for nodes communicating in the MAC layer i.e., either Physical (or) Data link layer consists of the header MAC info and authenticating 32-bit code at the end of frame. The total size is 6+6+2+N+4bytes (refer Table I). This data structure is maintained for the network layer also

2.2 Communication in Upper layer [i.e. above MAC layer

Layer 4-7 services, sometimes referred to as the Upper layers, support end-to-end communication between a source and destination application and are used whenever a message passes from or to a user the data structure for nodes communicating in the Upper layer i.e., either UDP (or) session (or) Presentation layer consists of the header IP address info, MAC address info and subnet Mask address (refer Table 2).

Table. 2. Data Structure in Upper Layer

Source IP	Destination IP	source MAC	Destination MAC	Subnet Mask
(6 bytes)	(6 bytes)	(6 bytes)	(6 bytes)	

The Upper layer Communication is made asynchronous and is hence faster. The following metrics have been studied: (i) throughput (ii) packet size (iii) Input file size (iv) Delay between packets etc.

III. Hardware

In this work, self powered nodes are used: either one as the server and the other as the client connected to each other via the Ethernet cable. The hardware used is ARM based architecture clocked at 120 MHz and has special feature of SDRAM to access the external memory. The features available are a LCD panel controller, an Ethernet MAC, a USB device interface, UART, SD card with the sample file for transfer residing in it.

IV. Performance Metrics

The metrics studied includes: The packets sent, packets received, packets lost and throughput for variable data size. The UDP senders will not get any feedback, and the data transfer is asynchronous. The UDP tests reports the received packets, throughput and the transfer time. These traces are used to calculate the packet lost and the efficiency. The UDP tests procedure involves the packets transfer in both the directions; therefore the measurements are significant in both the sides. The difference between the bytes sent and received across both the sides illustrates the actual volume of loss on the link.

4.1 Dynamic Parameters in Internode Communication

The user defined dynamic variables assigned includes MAC id user configurable, Ip address user configurable, Subnet mask details user configurable, Variable packet size (upper layer), Session time out and Delay or latency among the packets (throughput)

V. Data Encryption Model

Blowfish is a keyed, symmetric block cipher, designed in 1993 by Bruce Schneier and included in a large number of cipher suites and encryption products. 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 commercials /government secrets. Blowfish is unpatented. The algorithm is placed in the public domain, and can be freely used by anyone. Blowfish has a 64-bit block size and a variable key length from 1 bit up to 448 bits. It is a 16-round Feistel cipher and uses large key-dependent S-boxes. 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. 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 cipher text replaces P_1 and P_2. The same cipher text is then encrypted again with the new sub

keys, and P_3 and P_4 are replaced by the new cipher text. 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 sub keys - about 4KB of data is processed.

5.1. Encryption Algorithm on Chip

In this research, the encryption algorithm discussed in section V is implemented in HY-LPC1788 ARM processor on the UDP layer. The dynamic parameters (refer table 1and table 2) are declared in a separate header file. The 'c' file is written in keil compiler; a hex file is created and downloaded into the hardware. The communication results among the two mobile hardware nodes are shown in appendix -1 and appendix -2. The pseudo code of the key box selection function, encryption and decryption functions are presented below:

5.1 Error detection mechanism First match for MAC id If

{ yes
Then
Recalculate authentication code at receiver and compare with received code If yes
Accept data } Else
Give –ve ack.

5.2 Key box selection pseudo code

```
unsigned long F(BLOWFISH_CTX *ctx, unsigned long x) {
unsigned short a, b, c, d; unsigned long y;
d = x & 0x00FF; x >>= 8;
c = x & 0x00FF; x >>= 8;
b = x & 0x00FF; x >>= 8;
a = x & 0x00FF;
y = ctx->S[0][a] + ctx->S[1][b]; y = y ^ ctx->S[2][c];
y = y + ctx->S[3][d]; return y;

}
```

5.3 Encryption pseudo code

```
void Blowfish_Encrypt(BLOWFISH_CTX *ctx, unsigned long *xl, unsigned long *xr) {
unsigned long  Xl; unsigned long  Xr; unsigned long  temp; short i;
                                                    Xl = *xl; Xr = *xr;
for (i = 0; i < N; ++i) {
Xl = Xl ^ ctx->P[i]; Xr = F(ctx, Xl) ^ Xr; temp = Xl;
Xl = Xr;  Xr = temp; }

temp = Xl; Xl = Xr;  Xr = temp;
Xr = Xr ^ ctx->P[N];
Xl = Xl ^ ctx->P[N + 1]; *xl = Xl;
*xr = Xr;

}
```

5.4 *Decryption pseudo code*

```
void Blowfish_Decrypt(BLOWFISH_CTX *ctx, unsigned long *xl, unsigned long *xr) {
unsigned long  Xl; unsigned long  Xr; unsigned long  temp; short        i;
                                                    Xl = *xl; Xr = *xr;

for (i = N + 1; i > 1; --i) {
Xl = Xl ^ ctx->P[i]; Xr = F(ctx, Xl) ^ Xr;
/* Exchange Xl and Xr */ temp = Xl;
Xl = Xr;  Xr = temp; }

/* Exchange Xl and Xr */ temp = Xl;
Xl = Xr;  Xr = temp;
Xr = Xr ^ ctx->P[1]; Xl = Xl ^ ctx->P[0]; *xl = Xl;
*xr = Xr; }
```

VI. Results and Discussion

6.1 Variable file size, fixed delay and packet length

Case 1: The link, file and network parameters chosen for study in this work are listed in Table 3.

Table. 3. Link, File and Network Parameters

Packet Length	Delay	File Size
64 bytes	600 ms	$F_1 < F_2 < F_3$

The parameters studied are the throughput, packets sent, packets received and the transfer time are shown in figure 2 to figure 5. The packets sent and received are the same for the respective data size. Ex: for the file size of 104KB the packets sent and received is 108191 for the fixed delay of 600ms i.e. packets sent from the server and that received at the client end are the same : no packets lost for the delay of 600ms. From the graph it can be inferred that the throughput (has smaller values) becomes smaller as the file size increases. The packets sent, packets received and the transfer time increases as the file size increases.

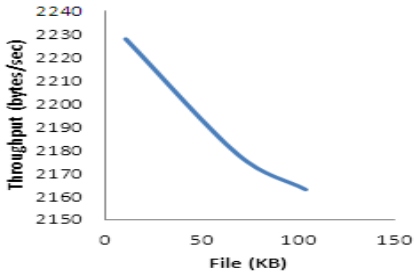


Fig. 2. File size (Kilo bytes) Vs throughput sent (bytes/sec); (bytes); Delay = 600ms, Packet length = 64 bytes

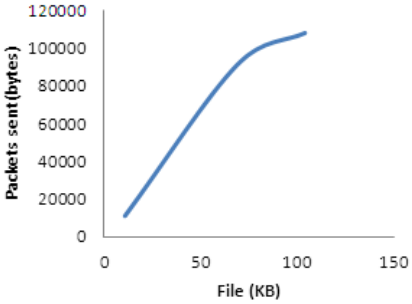


Fig. 3. File size (Kilo bytes) Vs packets Delay = 600ms, Packet length = 64 bytes

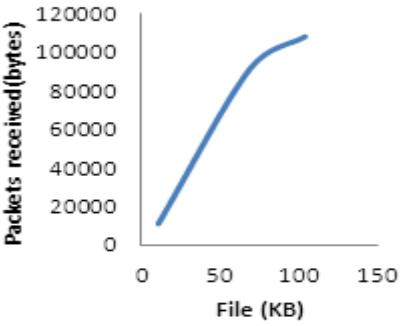


Fig. 4. File size (Kilo bytes) Vs packets received (bytes); Delay = 600ms, Packet length = 64 bytes

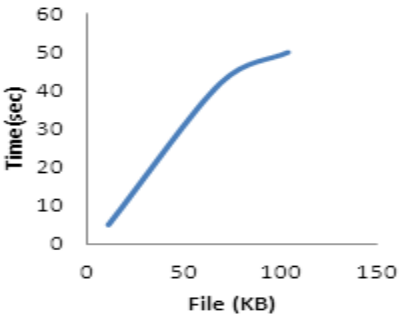


Fig. 5. File size (Kilo bytes) Vs transfer time (seconds); Delay = 600ms, Packet length = 64 bytes

Case 2: The second set of link, file and network parameters chosen for study in this work are listed in Table 4.

Table. 4. Second set of link, file and network parameters

Packet Length	Delay	File Size
64 bytes	300 ms	$F_1 < F_2 < F_3$

It is observed that the packets sent, packets received and the transfer time increases as the file size increases and is shown in figure 6-9. The throughput decreases as the file size increases. In this scenario it is observed that the no. of packets sent from server are not correctly received at the client end i.e. there is packet loss but the packets loss is constant for variable file size**.**

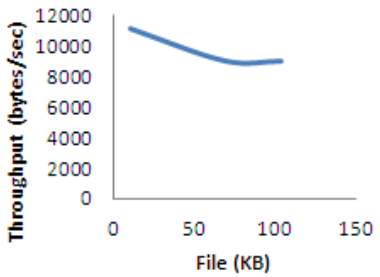


Fig. 6. File size (Kilo bytes) Vs throughput sent (bytes/seconds); Delay = 300ms, Packet length = 64 bytes

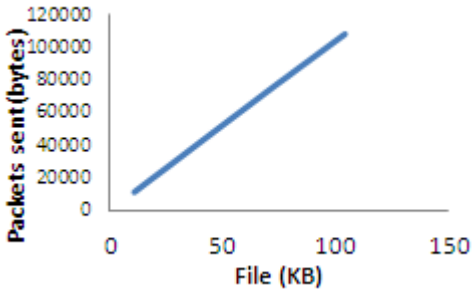


Fig. 7. File size (Kilo bytes) Vs packets (bytes); Delay = 300ms, Packet length = 64 bytes

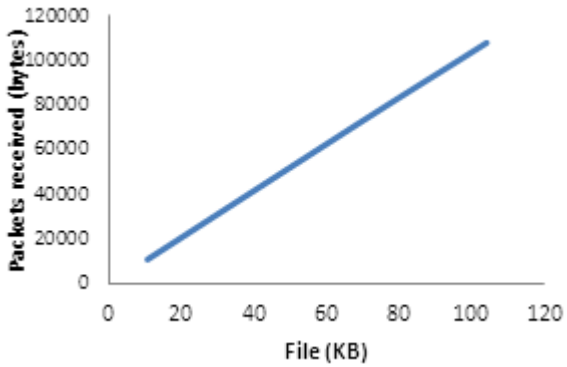


Fig. 8. File size (Kilo bytes) Vs packets received (bytes)Delay = 300ms, Packet length= 64 bytes

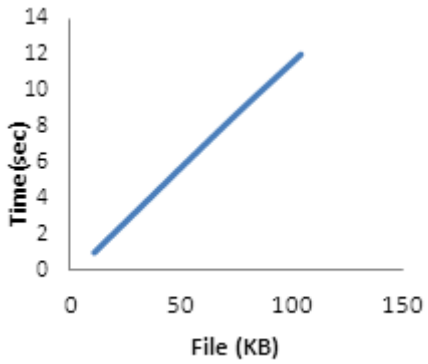


Fig. 9. File size (Kilo bytes) Vs transfer time(seconds); Delay = 300ms, Packet length= 64 bytes

Case 3: Encrypted Module Output

The data encryption and decryption module implemented in Netwinz Emulator is shown in figure 10 for trusted node figure 11 for untrusted node.

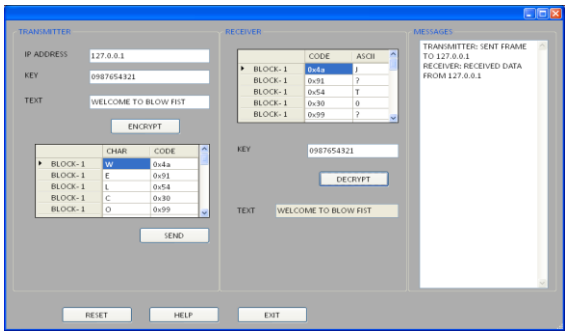


Fig. 10. Trusted Node Encrypted data transfer

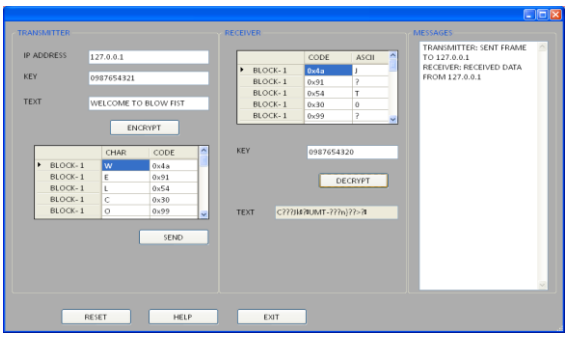


Fig. 11.untrusted Node Encrypted data transfer

Case 4: Chip implementation of encryption data transfer

The hardware nodes with the encryption algorithm (blowfish based) is set up in the two hardware nodes and is shown in appendix -1. In appendix -2 the transfer of data from node 1 (source) to node 2 (receiver) is shown. Similarly data can be transferred from node2 (as Source) to node1 (as receiver)

VII. Conclusion

In this work, the metrics for the wireless based data transfer among nodes is studied. For authenticated data transfer, an encryption algorithm using blowfish algorithm is discussed. Stand alone Hardware implementation of individual nodes using ARM processor is also done. From the study, it can be concluded that for the delay of 600ms the packets received and sent are identical i.e. no loss. The packets loss decreases as the delay increases and is zero for a delay of 600ms. The throughput also decreases as the delay increases. Even for the larger file size (> 104KB) the throughput is smaller but packets loss is less.

References

[1] A. F. Dana, R. Gowaikar, R. Palanki, B. Hassibi and M. Effros, "Capacity of wireless erasure networks", *IEEE Transaction Information Theory, vol. 52, no. 3*, pp. 789–804, 2006.

[2] A. Shokrollahi and R. Storn, "Design of efficient erasure codes with differential evolution", *in Proc. International Symposium Information Theory*, Sorrento, Italy, June 2000.

[3] C. Cachin and M. Geisler, "Integrity protection for revision control", in M. Abdalla and D. Pointcheval, editors, *Proc. Applied Cryptography and Network Security (ACNS)*, vol. 5536 of Lecture Notes in Computer Science, pages 382–399, 2009.

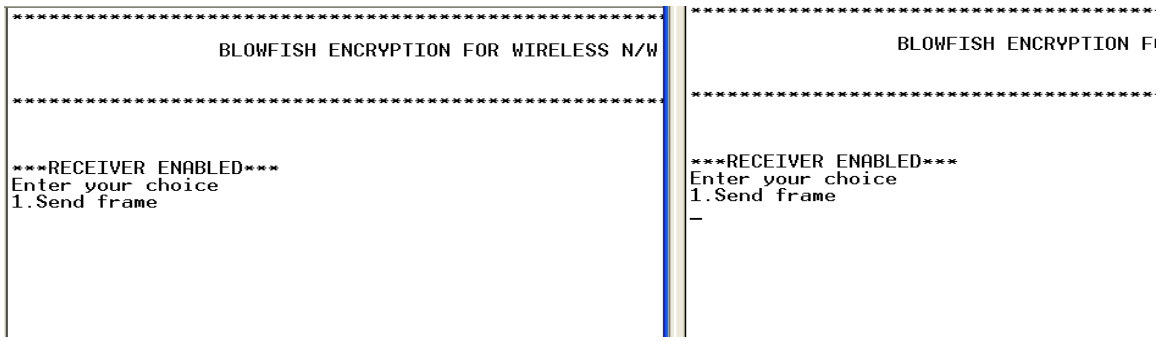
[4] C. Cachin, A. Shelat, and A. Shraer, "Efficient fork-linearizable access to untrusted shared memory", in *Proc. 26th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 129–138, 2007.

[5] D. L. Rosenband, "*Synthesis of multi-cycle operation-centric descriptions*", Ph.D. Dissertation Proposal, Massachusetts Institute of Technology, June 2000.

[6] D. J. C. MacKay, "Good error correcting codes based on very sparse matrices", *IEEE Transactions Information Theory, vol. 45*, pp. 399–431, Mar. 1999

[7J. Wang, L. Dolecek and R.D. Wesel, "Controlling LDPC absorbing sets via the null space of the cycle consistency matrix", in Proc. *IEEE International Conference on Communication (ICC)*, Kyoto, Japan, Jun. 2011.

APPENDIX-1 (Communication established for encrypted data transfer between two nodes in hardware



APPENDIX -2(Hardware node 1 acts as transmitter and hardware node 2 as receiver)