DOI: 10.21917/ijct.2012.0086

MULTIMEDIA DATA TRANSMISSION THROUGH TCP/IP USING HASH BASED FEC WITH AUTO-XOR SCHEME

R. Shalin[1] and D. Kesavaraja[2]
[1]*Department of Computer Science and Engineering, Dr. G.U. Pope College of Engineering, India*
E-mail: shalin.cse@gmail.com
[2]*Department of Computer Science and Engineering, Dr. Sivanthi Aditanar College of Engineering, India*
E-mail: dkesavaraja@gmail.com

Abstract
The most preferred mode for communication of multimedia data is through the TCP/IP protocol. But on the other hand the TCP/IP protocol produces huge packet loss unavoidable due to network traffic and congestion. In order to provide a efficient communication it is necessary to recover the loss of packets. The proposed scheme implements Hash based FEC with auto XOR scheme for this purpose. The scheme is implemented through Forward error correction, MD5 and XOR for providing efficient transmission of multimedia data. The proposed scheme provides transmission high accuracy, throughput and low latency and loss.

Keywords:
Encoding, FEC, Decoding, MD5, TCP/IP, XOR

1. INTRODUCTION

In TCP/IP, TCP is dependable for breaking data into IP packets before they are sent, and for assembling the packets when they arrive. IP is responsible for sending the packets to the correct destination. Packet loss occurs when one or more packets of data traveling across a computer network fail to reach their destination. Loss can occur for many reasons: transient congestion, degraded or dirty fiber, malfunctioning or misconfigured equipment, low receiver power, and burst switching contention are some reasons [5] – [9]. Loss occurs in different patterns, ranging from singleton drops to extensive bursts [17], [18]. Two methods can be used to deal with the transmission error [10] in the networks. One is Automatic Repeat Request (ARQ), and another is Forward Error Correction (FEC). TCP/IP most commonly used protocol uses ARQ to ask for retransmission of the lost data packets. However, in the case of distributing real-time multimedia data, the ARQ mechanism will result in considerable delays which are not allowed in such applications. While the traditional FEC methods mainly focus on the alteration of bit errors, on high-speed networks, particularly on fiber networks, bit errors rarely occur. For an example, on fiber networks, the Bit Error Rate (BER) is only 10^{-9}[10]. The main data loss comes from whole packet loss in the switch queue buffer [11].

The FEC method is introduced here to recover from packet loss with minimum overhead for multimedia data transmission. For long distance networks like international networks, latencies are high (on the order of hundreds of ms) [13]. This can rigorously impact real-time interactive applications. Hence a scheme is needed to transmit data reliably over long distances without requiring the acknowledgement typically used in protocols such as TCP. FEC provides a promising solution to the problem in that errors are corrected at the end point without the need to wait for the retransmission of a small package. The traditional reason for choosing ARQ as the main error correction used by many trustworthy protocols is that the FEC may introduce considerable computational overhead, and will also increase the bandwidth requirements [10]. Thus, it is important to choose an FEC method that can achieve loss recovery while minimizing computational overhead. The most suitable FEC scheme will depend on the nature of the data being transmitted [12].

These are several guidelines for generating FEC redundancy for real-time environments:

- Do not use very complex mathematic operations to generate the redundancy [10]. Make sure the computational time is less than the retransmission time. Here the operation used is exclusive OR it is very simple.
- Use the adjacent packets to generate the redundancy [10]. Using packets far away from each other will result in more delay, an increase in the requirements for the buffer both at the sender and receiver, and an increase in the complexity of buffer management

2. SYSTEM ARCHITECTURE

Senders sends the encoded file along with the hash value generated using MD5 algorithm hence if any loss occurs while transmitting self recovery is done by receiver while decoding with the help of encoded information. Once receiver receives the file it generates the hash value and checks with the sender hash value.

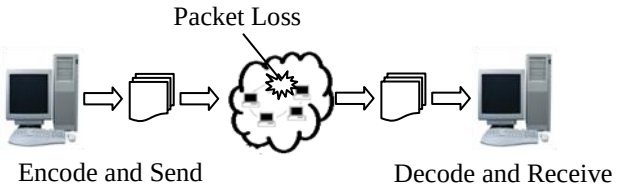


Fig.1. Architecture of proposed System

The encoded information used here is simple XOR. Hence if a packet is lost, with the help of encoded XOR we can decode and recover the lost packet. Fig.1 shows the proposed architecture. The architecture is simple and transparent.

3. PACKET TRANSMISSION

In this system has reduce delays and recover packet automatically handle the large size multimedia file in an efficient and effective way. It provides high through put.

3.1 RECOVERY OF PACKET DELAYS

Conventional TCP/IP uses positive acknowledgments and retransmissions to ensure trustworthiness. The sender packets until their receipt are acknowledged by the receiver and resends if an acknowledgment is not received within some time period. Hence, a lost packet is received in the form of a retransmission that arrives no earlier than 1.5 Round Trip Time after the original send occurrence [1]. The sender has to buffer each packet until it is acknowledged, which takes one Round Trip Time in lossless action, and it has to perform additional work to retransmit the packet if it does not receive the acceptance. Also, any arrived packet number is higher sequence numbers than that of a lost packet must be queued while the receiver waits for the lost packet to reach the destination.

3.2 MASSIVE FILE TRANSMISSION WITH HIGH THROUGHPUT FUNCTION

TCP/IP uses fixed-size buffer at receiver side to avoid overflow. The sender never pushes more unacknowledged information into network.. In other words, the size of the variable window at the sender is surrounded by the size of the buffer at the receiver [1]. In high-speed long-distance networks, the amount of unacknowledged data has to be very high for the pour to saturate the set of connections. Since the size of the receiver window limits the sending wrapper, it plays a major role in determining TCP/IP's throughput. The default receiver temporary memory sizes in many standard TCP/IP implementations are in the variety of data such as data, images and video, [1]. A normal resolution is to increase the size of the receiver buffers. However, in many cases, the receiving end host may not have the auxiliary memory capacity to buffer the entire bandwidth-delay [1].

4. HASH BASED FEC WITH AUTO XOR SCHEME

Hash based FEC with Auto XOR scheme in sender side packet separation and FEC Encoding is performed and in receiver side FEC Decoding and Packet Loss Determination and Error Correction is performed. Fig.2 explains the steps in this scheme. These operations are performed to obtain accurate output.

Process:

1. Read the input file
2. Packet Separation
3. FEC Encoding
4. Hash Generation
5. Hash Generation for Received File
6. FEC Decoding
7. Receiver Hash Compared with Sender Hash
8. Resultant File

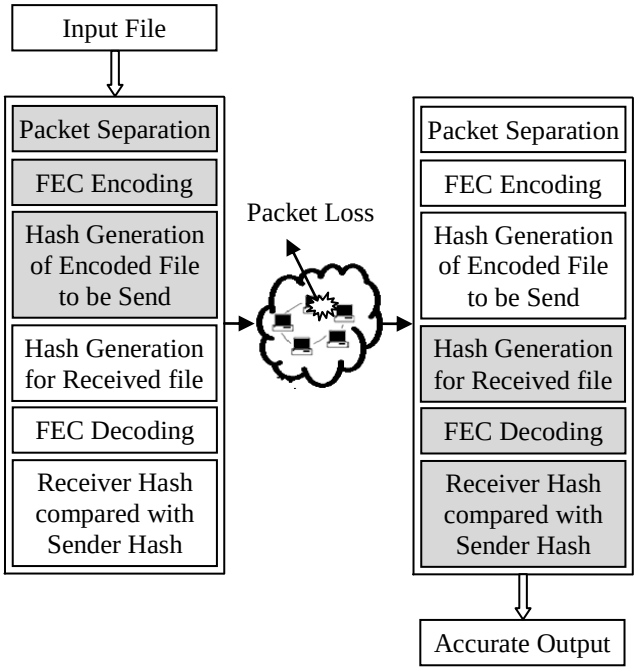


Fig.2. Hash Based FEC with Auto XOR Scheme

4.1 BASIC MECHANISM

A repair packet contains "R" list of data packet identifier and FEC information generated from these packets. At the receiving side it examines incoming repair packets and uses them to recover missing data packets. The basic operation of "Hash based FEC with auto XOR Scheme" is shown in Fig.3.

Forward error correction (FEC) is a method of obtaining error control in data broadcast in which the source sends redundant data and the destination know only the piece of the data that contains no obvious errors. FEC can be used for broadcasting of data to many destinations at the same time from a single source.

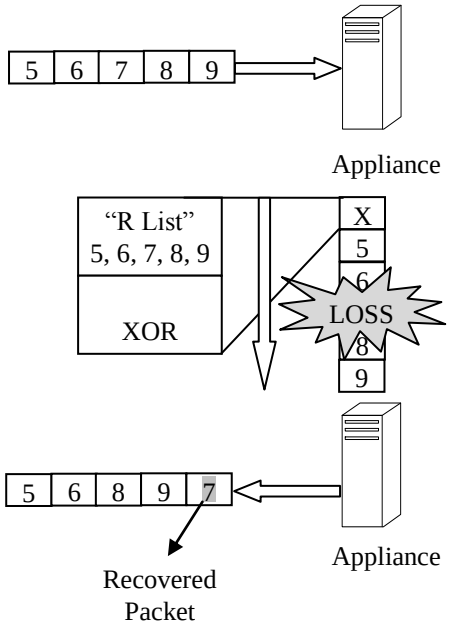


Fig.3. Basic Mechanism

In the above example the FEC information is a simple XOR. The repair packet contains the list of identifiers and encoded FEC Information that is the XOR of those 5 packets that is sent by the sender. The packets are sent along with the XOR and packet 8 is lost while broadcast. Hence using repair packet the lost data packet 8 is recovered. The self recovery is performed by performing XOR operation for the received packets and encoded XOR. The Fig.4 shows how XOR works to recover lost packet.

4.1.1 Packet Separation:

In the input file, this scheme reads all the characters, then separate the total characters in to equal number of blocks. This process is known as packet separation.

4.1.2 Interleaving:

Interleaving is a way of organize data in a non-contiguous way in order to increase performance. It is used in data transmission to protest against burst errors. In this module the data (shuffle) is set to avoid burst errors which are useful to increase the performance of FEC Encoding.

This process gets the input as blocks of bits from the FEC Encoder. In this module the bits inside a single block is shuffled in order to convert burst errors into random errors. This shuffling process is done for each and every block comes from the FEC Encoder.

4.1.3 FEC Encoding:

FEC is a scheme of fault control for data broadcast, where the sender adds redundant data to its messages. This allows the receiver to detect and correct errors, without the need to ask the sender for additional data. It reduces time and space for retransmission.

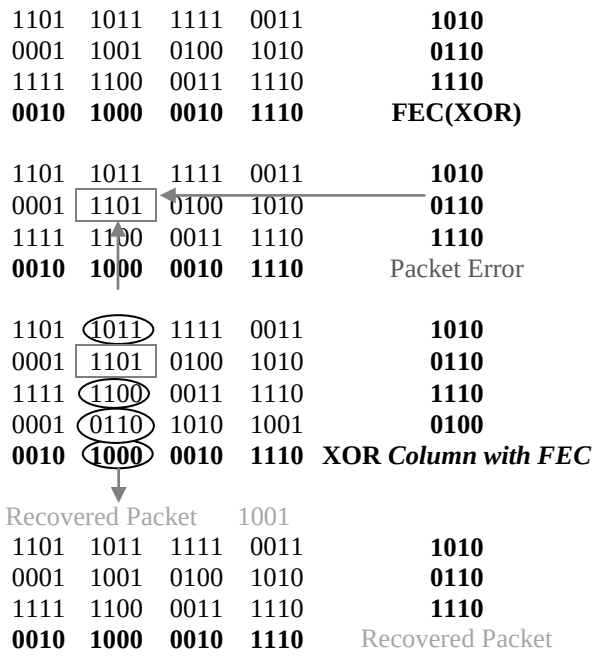


Fig.4. Steps in recovering packet using XOR operation

In this redundant data is added to the given input data, known as FEC Encoding. The text available in the input text file is converted into binary. The binary conversion is done for each and every character in the input file. Then we add the redundant data for each bit of the binary. After adding we have a block of packets for each character. In Fig.4 the redundant data is the simple XOR.

4.1.3 De-Interleaving:

This process receives the blocks of data from the Queue through the socket connection. In this process the data packets is rearranged inside a block in the order in which it is before Interleaving. This process of Interleaving and De-Interleaving is done to convert burst errors into random errors. After De-Interleaving the blocks are arranged in the original order. Then the data blocks are sent to the FEC Decoder.

4.1.4 FEC Decoding:

The received packets are processed to remove the redundant bits from it. Thus we recover the original bits of a character by decoding. After retrieving the original bits, it converts this to characters and writes it inside a text file. If any of the packets is being lost it can be retrieved by using the redundant data.

4.2 PACKET LOSS DETERMINATION

MD5 hash is classically expressed as a 32 digit Hexadecimal number. MD5 consists of 64 of these operations, grouped in four rounds of 16 operations. F is a nonlinear function; one function is used in each round. M_i denotes a 32-bit block of the message input, and K_i denotes a 32-bit constant, different for each operation. $<<<_s$ denotes a left bit rotation by s places; s varies for each operation. $\boxplus$ denotes addition modulo 232[22].

The Server sends the file along with the hash value generated by using MD5 [20] and the client receive the file and generate hash value for the received file. If both the hash value matches it displays absence of packet loss else it displays presence of packet loss. The property of combining forward error correction (FEC) and MD5 with TCP is discussed and concluded that FEC and MD5 reduces the retransmissions rate and it is useful for efficiently running the network at a very huge load.

5. EXPERIMENTAL RESULTS

The investigational result shows the original file, the encoded and decoded file. The text, video and image files are chosen for example that is shown in Fig.5. The original file is being separated into packets and encoded and stored. For this encoded file hash value is computed using MD5 algorithm and sent to receiver along with the file.

Receiver receives the encoded file and computes hash value with MD5 algorithm for the received file and compares the hash value with the sender hash value. The comparison is done to ensure trustworthiness. Then decoding is performed to receive the proper original file and if a packet is recovered by decoding it will surely improve the overall throughput.

The efficiency of this “Hash based FEC with auto XOR Scheme” is estimated using the block size, code rate and it is found that this scheme is 98.00% to 100% efficient data transmission. The Fig.7 shows the efficiency of “Hash based FEC with auto XOR Scheme” and it adds 10% of extra information to attain our goal in an efficient way.

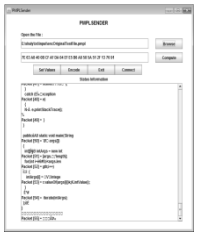
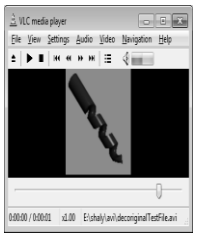
Original File	Encoded File	Decoded File
Text File (.txt) Size: 2.58 KB 	Size: 2.27 KB 	Size : 2.27KB 
Image File (.jpg) Size: 25KB 	Size: 23KB 	Size: 23KB 
Video File (.avi) Size: 30KB 	Size: 34KB 	Size: 30KB 

Fig.5. Original, Encoded and Decoded Files

The process of transmitting data until positive acknowledgement is received. And if negative acknowledgement is received the process of retransmitting it takes additional time (Fig.6). But by using this scheme self restoration can be done by using the encoded information and packets. Hence retransmission can often be avoided. The proposed scheme represents a way for improving the trustworthiness of transmitted or stored data. To ensure trustworthiness and to detect if any loss of packet is found during transmission the hash based MD5 algorithm is used.

6. PERFORMANCE EVALUATION

Hash based FEC represents the most efficient, economical, and predictable way of improving the reliability of transmitted or stored data. The process of buffering until positive acknowledgement is received. And if negative acknowledgement is received the process of retransmitting it takes extra time. But by using Hash based FEC self recovery can be done using the encoded information.

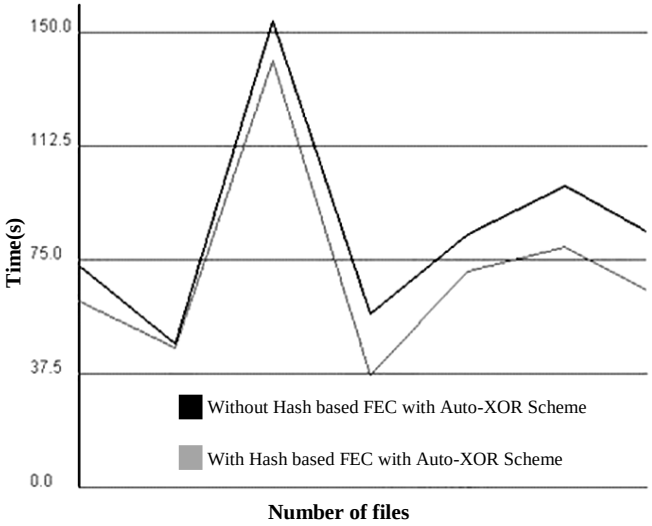


Fig.6. Time Analysis

Table.1. Time Analysis with and without Hash based FEC

File Name	Time in ms	
	With Hash based FEC	Without Hash based FEC
a.txt	74	62
b.txt	48	47
c.txt	154	141
d.txt	58	38
e.txt	84	72
f.txt	100	80
g.txt	82	63
h.txt	60	47
i.txt	84	62

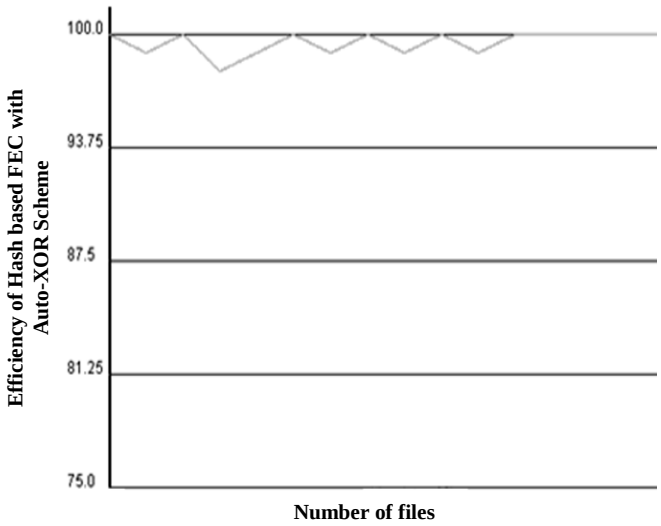


Fig.7. Efficiency of Hash based FEC

Table.2. Hash based FEC Efficiency

File Name	Efficiency of Hash based FEC(100-Packet Loss Rate)
a.txt	100
b.txt	98
c.txt	99
d.txt	100
e.txt	99
f.txt	100
g.txt	99
h.txt	100
i.txt	99

The Fig.6 shows the Time analysis with and without Hash based FEC and Fig.7 shows the efficiency of Hash based FEC. The efficiency of Hash based FEC is evaluated and it is found that it is highly efficient.

7. CONCLUSION

Loss of data in a network based communication system might hinder the proper functioning of the system. It is very important to cover packet loss transparently in a fast paced manner. The system proposed above is a edge piece of a software that uses Forward error correction for covering packet loss and improving TCP/IP throughput and latency by orders of scale when loss occurs. It was observed that using the proposed system the server only sent 10% more data to achieve the goal without acknowledgement traffic. This scheme will find a wide application in areas were transfer of multimedia documents is involved. The scheme is easy to install and transparent thereby improving efficiency.

REFERENCES

[1] M Balakrishnan, T Marian, K P Birman, H Weatherspoon and L Ganesh, "Maelstrom: Transparent Error Correction for Communication between Data Centers", *IEEE/ACM Transactions on Networking*, Vol. 19, No. 3, pp. 617- 629, 2011.

[2] D Wei, C Jin, S Low and S Hegde, "FAST TCP: Motivation, architecture, algorithms, performance", *IEEE/ACM Transactions on Networking*, Vol. 14, No. 6, pp. 1246–1259, 2006.

[3] C Parsa and J J Garcia-Luna-Aceves, "Differentiating congestion vs. random loss: A method for improving TCP performance over wireless links", *Proceedings of IEEE Wireless Communications and Networking Conference*, Vol. 1, pp. 90–93, 2000.

[4] R Krishnan, J Sterbenz, W Eddy, C Partridge and M Allman, "Explicit transport error notification (ETEN) for error-prone wireless and satellite networks", *Computer Networks: The International Journal of Computer and Telecommunications Networking – Special Issue: Networking for the earth science*, Vol. 46, No. 3, pp. 343–362, 2004.

[5] R Habel, K Roberts, A Solheim and J Harley, "Optical domain performance monitoring", *Optical Fiber Communication Conference*, Vol. 2, pp. 174-175, 2000.

[6] Internet2, "End-to-end performance initiative: When 99% isn't quite enough – educause bad connection", Accessed on 2011[Online], Available:http://e2epi.internet2.edu/casestudies/EDUCAUSE/index.html.

[7] Internet2, "End-to-end performance initiative: Hey! Where did my performance go? Rate limiting rears its ugly head", Accessed 2011[Online], Available: http://e2epi.internet2.edu/case-studies/UMich/index.html.

[8] A Kimsas, H Overby, S Bjornstad and V L Tuft, "A cross layer study of packet loss in all-optical networks", *Proceedings of International Conference on Internet and Web Applications and Services / Advanced Telecommunications*, pp. 65, 2006.

[9] L B James, A W Moore, M Glick and J Bulpin, "Physical layer impact upon packet errors", *Passive and Active Measurement Workshop*, 2006.

[10] Ray Fang, Dan Schonfeld, Rashid Ansari, Jason Leigh, "Forward Error Correction for Multimedia and Teleimmersion Data Streams", *Internal Technical report, Electronic Visualization Laboratory*, University of Illinois at Chicago, 2000.

[11] E W Biersack, "Performance Evaluation of Forward Error Correction in ATM networks", *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 4, pp. 631-640, 1993.

[12] E W Biersack, "A simulation Study of Forward Error Correction in ATM networks", *Computer Communications Review*, Vol. 22, No. 1, pp. 36-47, 1992.

[13] J C Bolot, "End-to-End Packet Delay and Loss Behavior in the Internet", *Proceedings on Communications architectures, protocols and applications*, pp. 289-298, 1993.

[14] T V Lakshman and U Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss", *IEEE/ACM Transactions on Networking*, Vol. 5, No. 3, pp. 336–350, 1997.

[15] J Padhye, V Firoiu, D Towsley and J Krusoe, "Modeling TCP throughput: A simple model and its empirical validation", *SIGCOMM Conference on Applications, technologies, architectures and protocols for Computer Communication*, pp. 303–314, 1998.

[16] D Katabi, M Handley and C Rohrs, "Congestion control for high bandwidth-delay product networks", *Proceedings of SIGCOMM Computer Communication Review*, Vol. 32, No. 4, pp. 89–102, 2002.

[17] D C Kilper, R Bach, D J Blumenthal, D Einstein, T Landolsi, L Ostar, M Preiss and A E Willner, "Optical performance monitoring", *Journal of Lightwave Technology*, Vol. 22, No. 1, pp. 294-304, 2004.

[18] T J Hacker, B D Noble and B D Athey, "The effects of systemic packet loss on aggregate TCP flows", *Proceedings of ACM/IEEE Conference on Supercomputing*, pp. 1–15, 2002.

[19] William Stallings, "*Cryptography and Networks Security*", Prentice Hall, 2011.

[20] Douglas E. Comer, "*Computer Networks and Internet*", Prentice Hall, 2001.

[21] D Kesavaraja, R Balasubramanian and D Sasireka, "Implementation of a Cloud Data Server (CDS) for Providing Secure Service in E-Business", *International Journal of Database Management Systems*, Vol. 2, No. 2, pp. 44-55, 2010.

[22] A Saidane, V Nicomette and Y Deswarte, "The Design of a Generic Intrusion Tolerant Architecture for Web Servers", *IEEE Transactions on Dependable and Secure Computing*, Vol. 6, No. 1, pp. 45-58, 2009.