Improvement of Error Correction in the Networks using Generative Programming

Senthil J.
Assistant Professor
VIT University
Vellore, India

Ishan Gupta
3rd Year B.E. CSE
VIT University
Vellore, India

Snehil Modani
3rd Year B.E. CSE
VIT University
Vellore, India

Apoorv K. Agarwal
3rd Year B.E. CSE
VIT University
Vellore, India

ABSTRACT

This paper focuses on impelling the FEC towards TCP transmission through generative frameworks based on existing error correction codes. It categorizes existing error correction algorithms on various bases and uses them as an outline for a generative program that monitors the present network parameters and generates a correction code accordingly. Required modifications in TCP packets are depicted and discussed with the challenges involved. Also need for smarter nodes has been emphasized based on discussed topics, considering the increasing load on servers.

General Terms

Forward Error Correction, Generative Programming, Checksum, TCP packets.

Keywords

FEC (Forward Error Correction), ECC (Error Correcting Code), Subtype Selection Bit, Broad-type Selection Bit, Error Correction Bits Array.

1. INTRODUCTION

Forward Error Correction[1][9] is used to control and correct errors in data transmission that occurs due to noisy channels. The Sender encodes their message in recurring fashion using an Error-Correcting Code (ECC). However, only a limited amount of error can be detected, but the ability of correcting these errors without requesting for retransmission makes F.E.C a topic of interest for the paper.Although this is achieved at the cost of increasing the channel bandwidth, it saves the time lost in retransmission, thus making it the choice for data recovery in mass storages and modems, analog-to-digital conversions, etc. The viability and efficiency of FEC depends on the ECC used, thus it become very important to use the right codes for an effective result.

The present TCP based packet consists of a checksum field used to detect accidental errors in header and data, based on a checksum algorithm. If the checksums matches at both the ends, it signifies that the data received is same as the one transmitted. Present checksum is a fixed-sized datum and is constrained by its definition. The proposed replacement bits for checksum will be defined dynamically at the time of transmission by the size of packet being sent and the choice of encryption.

Generative programming[6] is used to create source codes through generic models, templates, etc. for improving theprogram's efficiency. Many robust error correction codes exist today which can be exploited to design a framework for implementing the FEC effectively over TCP for a much speedy and secure data transmission.

The present TCP header is as follows. (See Figure 1)

Offsets Octet		0								1								2								3							
Octet	**Bit**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
0	**0**	Source port																Destination port															
4	**32**	Sequence number																															
8	**64**	Acknowledgment number (if ACK set)																															
12	**96**	Data offset				Reserved **0 0 0**				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	**128**	Checksum																Urgent pointer (if URG set)															
20	**160**	Options (if Data Offset > 5, padded at the end with "0" bytes if necessary																															

Figure 1: Present TCP Header

2. PROPOSAL

The proposed technique offers a solution of the problem of error detection as well as correction keeping in mind the Network parameters. The proposed technique does this using generative programming based on already exiting algorithms which detect and correct the errors in the packet.

2.1 Broad Type Selection Bits

The Network administrator is given an upper hand in selecting what kind of network the system belongs to, i.e. whether the system is in a network that requires more security, speed or reliability during Packet Transmission. If the network administrator chooses nothing then the default mode is selected in the implementation of the proposed techniques, which assesses the network by itself and proceeds thereafter.

This choice or assessment is stored in a 2-bit space in the TCP packet header as shown in Fig 2 as Class also called the Broad type Selection Bits. Table1 shows the option available to the Network administrator for choice given by the proposed technique; the following table shows the working of these bits:

Table1. Broad Type Selection Table

Type-bit	Broad-Type Selection
00	Default
01	High Speed Requirement Mode
10	More Security Requirement Mode
11	Increased Reliability Requirement Mode

2.2 Sub Type Bits

The error detection and correction is done using different techniques for different requirements. There are different algorithms present which offer very good error detection and correction in terms of security ad some algorithms do it without risking the efficiency of the packet transmission. The Proposed method involves the assessment of the network parameters by various means to know how the network is behaving at the moment just before the packet transmission. The network parameters include network congestion, network load, etc. which might hamper the packet transmission. Thus in this proposed technique the application layer sends a request to the network layer and gets these parameters in return.

After the parameters are analyzed the proposed technique uses a Generative approach to select and create a code based on an algorithm for which the packet transmission in unhindered , error detection and generation is more efficient, speedy and secure depending on the type of the sub-type bit involved. In case when no sub-type bit is selected i.e. the sub-type bit is ’000000’ or ‘Default’ then the program chooses by itself anAlgorithm according to the network parameter and the code is generated respectively. Following are some of the algorithms that can be used for Error Detection and Correction. (See Table2)

Table 2. Sub-Type Selection Table

Sub-Type Bits (in decimal)	Algorithm Selected
1	Hamming Code[4]
2	Raptor Code[3]
3	Golay-Binary
4	Latin-Square[5]
5	Lexicographic Code[2]
6	Low Density Parity Check Code
7	LT Codes
8	Tornado Code
9	AN Codes
10	Constant- Weight Codes
11	Goray-Ternery
12	Hadamard Code
13	Online Codes[7]
14	BCH Code
15	Convolutional Code[8]
16	Reed-Muller Code

2.3 Division of Algorithms on the Basis of Functionality

These algorithms can be divided according to its support to the packet transmission and its error Discovery and retrieval. The following classification is done for the above mentioned algorithms:

2.3.1 High Speed

i. Hamming Code
ii. Raptor Code
iii. Golay-Binary
iv. Latin-Square
v. Lexicodes
vi. Low Density Parity Check Code
vii. LT Codes
viii. Tornado Code

2.3.2 More Security

i. AN Codes
ii. Constant-Weight Code
iii. Golay-Ternary
iv. Hadamard
v. LT Codes
vi. Online Codes
vii. Tornado Code

2.3.3 Increased Reliability

i. BCH Code
ii. Convolution
iii. Reed-Muller Code

Offsets	Octet	0								1								2								3							
Octet	**Bit**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
0	**0**	Source port																Destination port															
4	**32**	Sequence number																															
8	**64**	Acknowledgment number (if ACK set)																															
12	**96**	Data offset				Reserved **0 0 0**			NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size															
16	**128**	Class	Type of Algorithm (Sub Type Bit)															Urgent pointer (if URG set)															
20	**160**	Error Correction Bit Array's Length																Correction Bits															
24	**192**	Correction Bits (as per the size of Correction Bit Array) followed by data bits Options (if Data Offset > 5, padded at the end with "0" bytes if necessary																															

Fig 2: Modified TCP Header (Changes are highlighted)

2.4 Changes made in the packet frame

Modifications in TCP packet frame as mentioned in Fig.1 are depicted in Fig.2. The Checksum Bits are removed and replaced with the Broad-Type Bits and the Sub-Type Bits. Then after the URG bits, the proposed method adds a field into the TCP header for storing the length of the Error Correction Bit Array generated after the algorithm is applied to the Packet after the Generative Selection of the algorithm.

The method adds the Error Correction Bit Array in the data field according to the algorithm selected during the Generative decision. Length of this array will be dynamic and is stored with the frame. These changes have to be taken care off by the network layer in case of Error Detection and Correction. The application layer will finally use the error correcting bits and regenerate the information lost in the packet.

2.5 Reasons for adding Error Correction Bit Array Length in the packet frame

At present the packet size is limited over a network with average packet size being 6kB to a maximum of 16kB. The Error Correcting Bits Array generated depends on the algorithm chosen as well as the size of data, hence it's important to understand that for varied size data, packet size could be varied and bandwidth could be saved. A larger data requires larger number of bits for correction; hence the Correcting array must be of varying length, which is depicted by Array Length field as shown in Figure 2. Thus the Data transmission will be more reliable, even if the Packet Size increases.

2.6 Need for Generative Programming[6]

The proposed theory states that the Error Correction cannot be done independently ignoring the factors affecting the network and for which self-generating codes out of different individual error correction algorithms have been generated for optimum results. The idea behind the implementation of the generative programming is to include the network attributes at the time of packet distribution. This accountability of the packet transmission can only be done using self-generating codes which will implement the needed algorithm at the right time so as to increase the speed, efficiency and security of the packet transmission.

Here generative Programming is used to determine the type of algorithm needed to maximize the speed, efficiency or security of the Packet transmitted, takinginto consideration the Network parameters like network congestion, or network bottle-neck present at that time when the packet is transmitted. This real-time assessment is veryhelpful in generating theright algorithm for the error correction of transmitting packets. This is done with a simple communication of the application layer with the network layer and collecting the information about the network parameters. A carefully judged algorithm is dynamically picked for which the code is generated; this code is used to encode the packets into the error correction bit array and also store this type of algorithm no in the respective bit block of the Error-Correction-Byte-Tag.The rest of the places will contain the length of the error correction Bit Array which varies, depending on the size of the packet and the algorithm used to calculate the same. These bits will be accessed by the decoder end-system to regenerate the data in case of error detection in the data. Also it will indicate the marker between the data bits and the Error Correction Bit Array.

3. RUN THROUGH WORKING EXPLANATION

We need an established connection and an application which wants to share data with another device over the network. The application will send the data in the form of Packets over TCP protocol; as this process is run the proposed method invokes the function to know the network parameters and comes to know the network condition, now a code is generated depending on the algorithm chosen by the method which is decided by the network analysis done and the Sub-Type bit selected by the Network administrator. With the generation of the code for the selected algorithm, the Error Correction Bit Array is generated. The length of this generated array is taken and stored in the Error Correction Length Bit as mentioned above in the explanation. The Error Correction Bit Array is stored in the Data field of the Packet to be transmitted, all executed by the Application layer. Packet is then transmitted using the existing protocols. The Packet which is transmitted may have an error, which will be detected by the end user at the time of receiving of the packet. The Error Detection is done by the end user after the Sub-type and the Type of Algorithm is detected. Only if the packet is detected of an error, the corresponding Correcting Algorithm is generated which is different for each type of Error Detection and Correcting Algorithm. The Error is then resolved depending on the type of algorithm used, if the reliability is to be guaranteed the Error Correction is much higher in that case. Or else if the error bits are too many, the packet itself is asked to be re-sent, which is already done in the present protocol. Thus the proposed method can guarantee a better and more reliable Network Transmission.

4. ADVANTAGES OF PROPOSED METHOD

The proposed system has many advantages over the present one, as the status of network is never static but affected by innumerous dynamics factors of environment, requirements, bandwidth, reliability, security, speed, traffic, medium, congestion etc. Deploying a single error correction algorithm may appease many, but it is in itself obsolete. Although choosing and implementing these algorithms may involve overheads in themself, but when compared to the time loss occurring due to retransmission of packet or losses occurring due to security faults or packet dropping turns out to be much more uneconomical and destructive than the generated overheads.

Hence a dynamic system based on generative program will not only be able to choose the best error correction algorithm among them as per the then present limitations and requirements of the network, but will also ensure speedier, efficient and more reliable transfer of information. With the ability to be overridden by advanced users, very clear dictating codes of conduct will be followed, which will ensure the fulfillment of the needs of the user.The existing size of the packet have may have an extension in their overall packet size, the proposed method helps in the reliable data transmission even for Packets with varied as well as packets which are bigger in size.

The proposed method also takes care of the over-head of the network when the Network is in Congestion, the algorithm ischosen in such a way that the network will not further deteriorate the loss in packet data, and the Network thus will not be further congested.

5. CONCLUSION

In this paper, we discussed the limitations of present TCP protocols and restrictions imposed by these bottlenecks on the much efficient Forward Error Correction Concept. We proposed modifications in the basic TCP packet architecture which will facilitate in implementation of a dynamic code generation technique based on present algorithms for error correction. We also discussed the current techniques used in various networking systems ranging from NASA's Space Missions to general file transfer techniques, and proposed a modular and combined version of all of these algorithms, where the requirements of the user are priorities. However, the system is robust enough to consider the limitations of the network and will find out the best solution catering to all confines and needs.

Considering the increased use of internet in past decade and upsurge in network oriented devices, it's important to balance the load on limited servers that are present today. Hence ensuring that data reaches its destination in minimum possible time will be more than ever necessary in upcoming time. Retransmission will greatly limit the potentials of present systems and hence implementation of smart nodes needs to be considered. This approach may lead to a more efficient and secure browsing scenario and a speedy transmission of data for even the home users, at a very low cost, if implemented it can act as a momentum in continuum of the ever increasing giant known as the Internet.

6. FUTURE WORK

Once implemented, this method of error correction will save a lot of resources and time that is being wasted in retransmission of packets. Also it will encourage development of many new FEC algorithms which shall make the system even more robust. It will open a new domain of data transmission where the nodes will not remain simple machines but will work as advanced intelligent routers which can communicate with each other and take decisions as a team for the interest of the whole connection. As the transmitted packets will take in consideration the network's limitations, a large amount of packets could be sent with much promising transmission rate. This method could possibly be implemented on the whole network to recover from failures in form of self-healing networks in future.

7. AKNOWLEDGEMENT

We thank Dr. S Arumugam, CEO, Nandha Educational Institution for his very helpful inputs and guiding us in our work.

8. REFERENCES

[1] Moon, Todd K. (2005). *Error Correction Coding*. New Jersey: John Wiley & Sons. ISBN 978-0-471-64800-0.

[2] G. Cohen, I. Honkala, S. Litsyn, A. Lobstein, *Covering Codes*, Elsevier (1997) ISBN 0-444-82511-8

[3] Bailey, R.A. (2008). "6 Row-Column designs and 9 More about Latin squares". *Design of Comparative Experiments*. Cambridge University Press. ISBN 978-0-521-68357-9.

[4] RFC5053 Raptor Forward Error Correction Scheme for Object Deliver.

[5] Latin Squares: New Developments in the Theory and Applications (Annals of Discrete Mathematics)by J. Denes, A.D. Keedwell

[6] Generative Programming: Methods, Tools, and Applications byKrzysztof Czarnecki and Ulrich Eisenecker ISBN 0201309777

[7] Hadamard Matrices and Their Applications, K. J. HoradamISBN069111921

[8] Convulational Codes by Philippe Piret

[9] Fundamentals of Error-Correcting Codes byW. C. (William Cary) Huffman, Vera S. Pless ISBN 978-0521131704