VOL. 9, NO. 1, JANUARY 2014 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

REVIEW OF ERROR DETECTION OF DATA LINK LAYER IN COMPUTER NETWORK

Afiqah Azahari[1], Raed Alsaqour[1], Mueen Uddin[2, 3] and Mohammed Al-Hubaishi[4, 5]
[1]School of Computer Science, Faculty of Information Science and Technology, University Kebangsaan Malaysia, Bangi, Selangor, Malaysia
[2]Kulliah of Information and Communication Technology, International Islamic University Malaysia, Malaysia
[3]Asia Pacific University of Technology and Innovation, Kuala Lumpur, Malaysia
[4]Faculty of Computer Science and Information System, Thamar University, Thamar, Republic of Yemen
[5]LAB, FCT-DEEI, Universidade Algarve Portugal, Faro, Portugal
E-Mail: raed.ftsm@gmail.com

ABSTRACT

Error control describes how the network handles and detects errors especially in the data link layer. In this paper, we present on an overview of error control regarding error detection and error correction. Error control happens in data link layer. We mainly discuss the type of error detection mechanisms that is used to detect the errors and how the errors will be corrected so the receiver can extract the real data. At the end of this paper, the conclusion and the future work are presented.

Keywords: data link layer, error control, parity check, checksum, cyclic redundancy check.

INTRODUCTION

As the second layer in TCP/IP model, data link layer provides services to the network layer (layer 3) using the services of the physical layer (layer 1) [1]. One of the services that are provided by the data link layer is the error detection and correction codes. Any connection that is based on the network is considered to have two channels: one used for the traffic and other used for signaling and control. In the traffic channel the information bits are encoded with robust error detection and correction codes to form the transmit data stream.

Error control that happens in data link layer of TCP/IP model detects error in received frames and retransmission requests of frames, while flow control determines the amount of data which can be transmitted in a given period of time [2]. Not all the network devices can run in the same speed, so we need flow control to control the amount of the data sent by the devices so the receiving device is able to accept data and handle it. The sliding window method and the stop-and-wait method are used in the flow control [3].

In this paper, we review the error detection methods for data link layer in TCP/IP model. Also, we discuss the performance of each method.

TYPE OF ERROR CONTROL

The information of data is transfer from one hop to another hop. In TCP/IP model, the physical layer, the final layer of TCP/IP model transforms the data into stream of bits and transfers them into a signal toward the receiver device. Meanwhile those bits flow from one hop to another, they are exposing to channels interference [4], for example electrical interference or thermal noise that subject to unpredictable change [5]. These channel interferences can change the shape of the transmitted signal leading into errors in the signal. There are two types of error, single-bit error and burst error [6]. In single bit error, only single bit in the stream is changed a zero changes to one or one changes to 0. In burst error, multiple bits in the stream are changed.

Single-bit: Single-bit error means that only one bit of data been change through transportation of data [7]. It changes either from 0 to 1 or 1 to 0. This one bit changed cannot be ignored since one bit change can change the whole meaning of the data that is transmitted. Figure-1 shows an example of this type of error. as shown in the figure, there are 8-bit stream 00000010 represent a start of text, but after one bit change to 00001010 due to the interference when the bits are transfer to the receiver, this corrupted bit stream will be mistakenly interpreted as a line feed, which is away different from the original transmitted data by the sender.

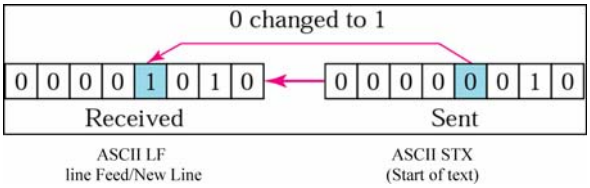

Figure-1. Single-bit error.

In the single-bit error, it was least likely type of error in serial data transmission. For example, imagine that the data was sent at 1 Mbps. This means that each bit last only 1/1000000 s. For the single-bit error to occur, the noise must have duration of 1 µs, which is very rare because noise normally last much longer than 1 µs.

Burst Error: Burst error means that two or more bits are changed when the transmitting data from the sender to the receiver the data units have change from 0 to 1 or 1 to 0 because of the channel interference [7].

Figure-2 shows the burst error of 8-bits where the length of bits error is measure from the first bit the error

occurs until the last bit of the corrupted bits, although if some bits between the errors are not corrupted.

Burst errors are likely to occur rather that the single-bit error. The duration of the error was longer that duration of 1 bit, which means the data is affected by the noise usually affect a set of bits. The number of bits that corrupted always depends on the data rate and duration of noise.

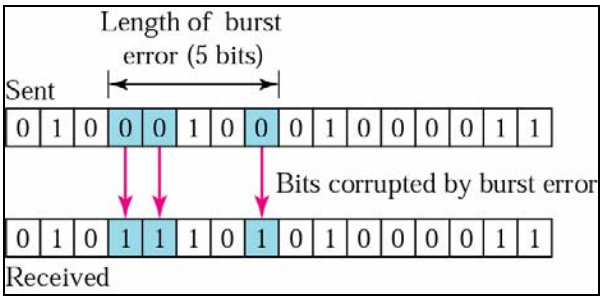

Figure-2. Burst error of length 8.

ERROR DETECTION

When transmitting a bit stream over a transmission line or channel a scheme is normally incorporated into the transmission control circuit to enable the presence of bits or transmission error in the receiving block to be detected. In general, this is done by the transmitter which computes a set of additional bits based on the contents of blocks of bits to be transmitted. This is known as error detection which is based on a block of extra bits which is transmitted together with the original bits in the block. The receiver uses the complete sets of received bits to determine whether the block contains any error to the high probability [8].

The two factors that determine the type of error detection scheme used are the bit error rate (BER) probability of the line and the type of error, that whether the errors occur as random single-bit errors or as groups of continuous of bit errors (burst error).

The different type of errors detection schemes detects different type of errors. Also the number of bits used in some schemes determines the burst lengths that are detected. The three most widely used schemes are parity, cyclic redundancy checks (CRC) and checksum.

PARITY CHECK

The most common method for detecting bits error with asynchronous character and character-oriented synchronous transmission is parity bit method. There are two types of parity check schemes: even and odd parity checks [9]. With the even parity check, the redundant bit is chosen so that an even number of bits are set to one in the transmitted bit string of N+r bits, where r is the bit that used to be the even parity check and N is the bit that is transmitted by the transmitter of the network. The receiver re-computes the parity of each received bits from the transmitter and discard the strings with the invalid parity. The parity scheme is always used if 7-bits character is exchanged. If there are 7-bits that are transmitted by the transmitter and parity check are used to detect the error, the eight bit is often the parity bit.

Table-1 shows a table contains 3 bits string. The transmitter will add 0 or 1 to the bits string according to the parity check mechanism (even or odd). When the receiver receives the bits string, the receiver will use the same mechanism to count the 1's in the bit string to determine whether it matches the counted parity from the transmitter or not. For example if the bits string transmitted 000 for odd parity check the transmitter will add 1 at the end of bits string so it will transmitted 0001 to the receiver. Then the receiver will count every single bit using the same mechanism for parity check. If the 1's in the receive bits string is odd, the bits string will be accept, otherwise the bits string will be rejected.

Table-1. Parity bits that are compute for bits string

3 bits string	Odd parity	Even parity
000	1	0
001	0	1
010	0	1
100	0	1
111	0	1
110	1	0

Performance of parity check: Parity check mechanism can detect all single-bit errors. It can also detect burst errors only if the total number of errors in each data unit is odd/even (based on parity check used). For example, even parity check mechanism cannot detect errors where the total number of hits changed is even. If any two bits change in transmission, the changes cancel each other and the data unit will pass a parity check even though the data unit is damaged. The same holds true for any further even number of errors.

CYCLIC REDUNDANCY CHECK (CRC)

The second method in error detection in data link layer is cyclic redundant check. Unlike parity check which is based on the submission of the binary, CRC is based on the binary division. In CRC, instead of adding bits to achieve a desired parity, a sequence of redundant bits, called the CRC or the CRC remainder, is appended to the end of a data unit so that the resulting data unit becomes exactly divisible by a second. On the destination side, the incoming data the binary data is divided by the same number to be compared on the source side. Means that, if the remainder of the division is same as the value on the added CRC when the data was transmit, the data will be accepted, otherwise the unmatched reminder produced on the destination after the CRC is indicates the data unit has been damage during the transmission of data.

The redundancy bits used by CRC are derived by dividing the data unit by a predetermined divisor; the remainder is the CRC. To be valid, a CRC must satisfy two conditions: It must have exactly one less bit than the divisor, and appending it to the end of the data string

must make the resulting bit sequence exactly divisible by the divisor.

Figure-3 shows the outline basic operation of the CRC. First, the string of *n* bits is added to the data unit. Second, the newly binary data unit is divided by the divisor *p*, combination of *n + 1* bits, called binary division. The remainders result from this division is the CRC (*n* bits). Third, the CRC value resulting from the second step is replaced with the value of *n* string. Note that the CRC may consist of all 0's. The binary data unit arrives to the receiver followed by the CRC. The receiver treats the incoming binary data as one data block and divides it with the same divisor that is used to get the CRC value. If the data was arrived without an error, the CRC checks return all the value which is zero and the binary data unit is passed [7].

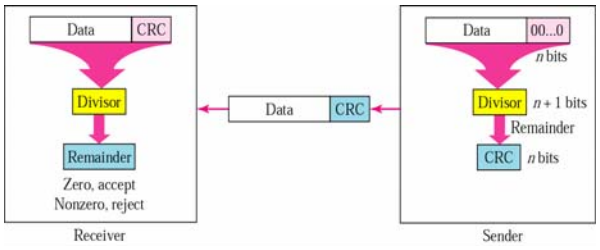

Figure-3. CRC operation.

Figure-4 shows the calculation for the CRC in the sender, as shown, the added of the data plus extra zero that is added to the data string and divided with the divisor. The remainder of the division will be the value of CRC that will replace the data plus extra zeros at the receiver side. Figure-5 shows the calculation for the CRC in the receiver side. At the receiver side, the data string and the CRC value is divided by the same value of divisor in the sender part. Then the remainder of this division determines either the received data bit string that to be accepted or not. If the remainder is zero, then the data will be accepted or else it will be rejected.

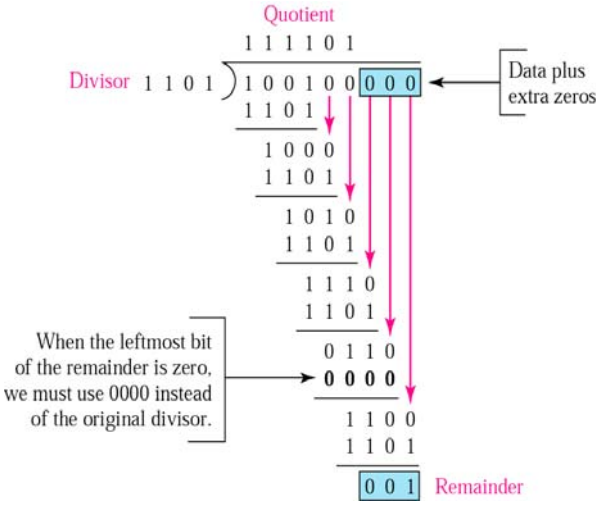

Figure-4. CRC in sender side.

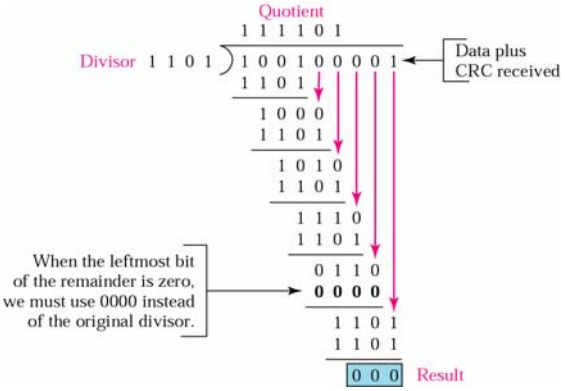

Figure-5. CRC in receiver side.

Performance of CRC: CRC has a very good performance in detecting single-bit errors, double errors, an odd number of errors, and burst errors. They can easily be implemented in hardware and software. They are especially fast when implemented in hardware. This has made CRC a good candidate for many networks.

CHECKSUM

A third approach to determine the error detection. The checksum method which is a very simple method based on adding up all the words that are transmitted and then transmit them including the complement result of that sum. Like the parity check and CRC, the checksum is based on the concept of redundancy. As shown in Figure-6, in the sender, the checksum generator subdivides the data unit into equal segments of n bits (usually 16). These segments are added using ones complement arithmetic in such a way that the total is also n bits long. That total (sum) is then complemented and appended to the end of the original data unit as redundancy bits, called the checksum field. The extended data unit is transmitted across the network. So if the sum of the data segments is T, the checksum will be -T. The receiver performs the same calculation on the received data and compares the result with the received checksum. If the result is 0, the receiver keeps the transmitted data; otherwise, the receiver knows that an error occurred discards the transmitted data [5].

The checksum detects all errors involving an odd number of bits as well as most errors involving an even number of bits.

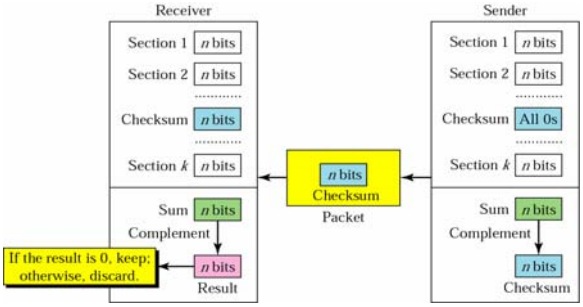

Figure-6. Checksum operation.

Suppose the following block of 16 bits is to be sent using a checksum of 8 bits.

10101001 00111001

The numbers are added using one's complement

```
          10101001
          00111001
          ----------
Sum:      11100010
Checksum: 00011101
```

The pattern sent is:

10101001 00111001 00011101

Now suppose the receiver receives the pattern: 10101001, 00111001, 00011101 and there is no error. When the receiver adds the three sections, it will get all 1s, which, after complementing, is all 0s and shows that there is no error.

```
                10101001
                00111001
                00011101
Sum:            11111111
Complement:     00000000 means that the
pattern is OK.
```

Performance of checksum: The traditional checksum uses a small number of bits (16) to detect errors in a message of any size (sometimes thousands of bits). However, it is not as strong as the CRC in error-checking capability. For example, if the value of one word is incremented and the value of another word is decremented by the same amount, the two errors cannot be detected because the sum and checksum remain the same. Also if the values of several words are incremented but the total change is a multiple of 65535, the sum and the checksum does not change, which means the errors are not detected. Fletcher and Adler [10, 11] have proposed some weighted checksums, in which each word is multiplied by a number (its weight) that is related to its position in the text. This will eliminate the first problem we mentioned. However, the tendency in the Internet, particularly in designing new protocols, is to replace the checksum with a CRC. The Fletcher checksum and the later Adler checksum are both designed to give error detection properties almost as good as CRCs with significantly reduced computational cost.

CONCLUSIONS

There are different ways to detect error in the data link layer. But not all the methods of error detection can detect error accurately and effectively. Every method has its own specialty, advantage and its own mechanism to detect error. Parity check is simple and can detect all single-bit error. CRC has a very good performance in detecting single-bit errors, double errors, an odd number of errors, and burst errors while checksum is not efficient as the CRC in error detection when the two words are incremented with the same amount, the two errors cannot be detected because the sum and checksum remain the same.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support of this work by the Centre for Research and Instrumentation Management (CRIM), University Kebangsaan Malaysia (UKM), Malaysia. Grant numbers: UKM-GGPM-ICT-035-2011 and UKM-GUP-2012-089.

REFERENCES

[1] A. Jasin, R. Alsaqour, M. Abdelhaq, O. Alsukour and R. Saeed. 2012. Review on Current Transport Layer Protocols for TCP/IP Model. International Journal of Digital Content Technology and its Applications. 6: 495-503.

[2] J. Chellis, C. Perkins and M. Strebe. 1999. MCSE: Networking Essentials Study Guide with CDROM, 2nd Ed. New Riders Publishing.

[3] S. Karris. 2009. Networks: Design and Management, 2nd Ed. Orchard Publications.

[4] K. Wu, H. Tan, Y. Liu, J. Zhang, Q. Zhang and L. M. Ni. 2012. Side channel: bits over interference. Mobile Computing, IEEE Transactions on. 11: 1317-1330.

[5] L. L. Peterson and B. S. Davie. 2007. Computer networks: a systems approach, 3rd Ed. Elsevier.

[6] T. Kaise and M. Kitakami. 2002. Single-bit error correcting and burst error locating codes. In: Information Theory, 2002. Proceedings 2002 IEEE International Symposium on. p. 117.

[7] F. Behrouz and M. Firouz. 2012. Computer Networks: A Top Down Approach, 1st International Ed. McGraw-Hill.

[8] F. Halsall. 2006. Computer Networking and the Internet, 5th Ed. Pearson Education India.

[9] O. Bonaventure. 2011. Computer Networking: Principles, Protocols and Practice: The Saylor Foundation.

[10] J. Fletcher. 1982. An arithmetic checksum for serial transmissions, Communications. IEEE Transactions on. 30: 247-252.

[11] J. Gailly and P. Deutsch. 1996. Zlib compressed data format specification version 3.3. Network Working Group Request for Comments (RFC), http://tools.ietf.org/html/rfc1950.