Error Masking Probability of 1's Complement Checksums

Changli Jiao
Dept. of Elec. and Computer Eng.
Wayne State University
Detroit, MI 48202
Email: changli@vajrasuchika.eng.wayne.edu

Loren Schwiebert
Dept. of Computer Science
Wayne State University
Detroit, MI 48202
Email: loren@cs.wayne.edu

Abstract—**In the transport layer of the TCP/IP protocol suite, both TCP and UDP use internet checksum to protect headers and data. Internet checksum uses 1's complement arithmetic to detect errors in the content delivered by the data-link layer. Both research and experience have shown that there are a wide variety of error sources which can not be detected by this lower layer. The error detecting performance of 1's complement checksum determines how many of these errors will be passed to higher layers, including the application layer. The performance analysis will also influence protocol design and improvement, for example, header compression. Unfortunately, previous work on this topic only determined the number of error passing patterns and the probability for 2 and 3 bit errors, and the method used for determining the probability is hard to extend to more bit errors. In this paper, we present a method to generate the formula of error passing probability. When too much calculation is needed to compute an exact result, we achieve a better estimation of the probability, which is around 3 percent of the upper bound achievable with previous techniques when 1's complement checksum is used in TCP/UDP.**

I. Introduction

When data are transmitted over a medium, it is possible that some bits will get corrupted. So we need error detection technology to prevent errors from getting passed to users. In Ethernet, Point-to-Point Protocol (PPP) and most wireless data networks [10], the link layer uses Cyclic Redundancy Check (CRC) to detect errors. In the transport layer, both TCP and UDP use internet checksum [8] [7] to do error detection on headers and data, although the checksum is optional with UDP. CRCs are based on polynomial arithmetic, base 2. It has long been known that CRCs are very powerful for error detection. CRC-32, the most commonly used CRC in the TCP/IP suite, can detect all bursty errors with length less than 32 bits and all 2-bit errors less than 2048 bits apart. For all other error patterns, the chance of not detecting is 1 in 2^{32}. Internet checksum [5] [9] is 1's complement value of the 16-bit 1's complement checksum of the header and data. Compared with CRC using the same length of bits, it is weaker in error detection, whereas less calculations are needed.

Given CRCs' strong error detection ability, one could argue that the TCP or UDP checksum is not necessary. Practically, this was tried in 1980s [12]. For some Network File Server (NFS) implementations, UDP checksum was disabled based on this argument. But this idea resulted in file corruption and soon was discarded as a bad idea. Recently, Stone and Patridge have shown for the Internet today, there are a wide variety of error sources which cannot be detected by link-level CRCs [12]. Defective hardware, buggy software, and problems in both end-systems and routers can all cause errors that will pass link-level error detection. Newly emerging software and hardware are especially vulnerable to these problems. In essence, any errors introduced at protocol layers above the link layer will not be detected by the link-layer CRC. In transport layer, this task can only be done by internet checksum. The property of internet checksum, i.e., 1's complement checksum, should be analyzed in order to understand the error detection performance and to guide protocol design and improvement.

This material is based upon work supported by the National Science Foundation under Grant No. 0086020.

A checksum is calculated over a block of data for use in error detection. When one or more bits in the data block are changed, usually the checksum value also changes. If bits are changed but the checksum remains the same, the errors in these bits cannot be detected via the checksum. The chance of the erroneous block matching the original checksum is called the *error masking probability* or *error passing probability*.

The error masking probability of only 2's complement checksum has been computed [1] before. For 1's complement checksum, the probability has not been computed fully. Desaki, et al. [3] partially solved this problem. They calculated the error passing probability of 2 or 3 errors. However, the method provided is hard to extend to more bits errors. And, extending the method might lead to wrong results since some error passing patterns are ignored. In addition, The upper and lower bounds on the total probability provided in [3] are not very tight.

In this paper, we give a brief review of the previous work in section II. We also correct the small oversight made by Desaki, et al. [3] and explain the group of error patterns not considered before. In section III, we prove several properties on error values and give an exact formula of the error passing probability for a block. We have not obtained a closed-form solution so the computation for large data blocks is infeasible. However, this formula does allow an exact result for smaller data blocks and tighter error bounds. In section IV, we give an upper bound and a way to calculate better upper and lower bounds. Based on observations from error passing probability on small data blocks, we provide evidence for a conjecture on the error passing probability. This conjecture, if true, provides even tighter bounds.

II. Previous Work

Suppose a block of data is composed of s words, where each word contains n bits and is treated as a binary integer. 2's complement checksum is the sum of these words modulo 2^n, which means the carry bits will be discarded. 1's complement checksum is calculated through a different way. If the sum grows

larger than n bits, the carry bits are added to the final sum. For either checksum, there are a number of block values that will give the same checksum result. As indicated in [11], the number of blocks that have the same 2's complement checksum is 2^{ns-n}. For 1's complement checksum, the checksum is 0 only when all the words in the block are 0. The possible ways of generating any other checksum value are $\frac{2^{ns}-1}{2^n-1}$.

For 2's complement checksum, the closed-form error masking probability was given in [1], under the assumption that every bit in the block has the same probability to be 0 or 1, and, each bit has the same probability to be corrupted. There is no similar result for 1's complement checksum. In [3], double and triple errors that could pass checksum error detection were analyzed, for $n \geq 3, s \geq 3$ and $n \geq 4, s \geq 5$, respectively. However, the analysis does not include all the error possibilities. For triple errors, it was indicated that for any kind of block, the errors could pass the error detection if and only if three errors occurred in two adjacent or contiguous columns. Two errors occur in the same column, both changing 0 to 1, or both 1 to 0. The third error falls on some row in the next higher column, changing 1 to 0, or 0 to 1, correspondingly. This conclusion does not hold for 3-bit word blocks. For example, taking three words when $n = 3$, the following three cases all have the same checksum value of 011.

$$B_1 = \begin{matrix}1\ 0\ 1\\0\ 0\ 1\\1\ 0\ 0\end{matrix} \quad B_2 = \begin{matrix}1\ 1\ 1\\0\ 1\ 1\\0\ 0\ 0\end{matrix} \quad B_3 = \begin{matrix}1\ 0\ 1\\0\ 1\ 1\\0\ 1\ 0\end{matrix} \quad B_4 = \begin{matrix}1\ 0\ 1\\1\ 1\ 1\\1\ 0\ 1\end{matrix}$$

Suppose the original block contains B_1. When three errors occur that change these three words to B_2 or B_3, checksum can not detect them. Checksum can not detect B_4 either, which is not included in [3] and which is also caused by 3 errors. This suggests that a group of error patterns may not have been considered before.

Previously, the method of getting the error probability was to analyze the patterns of error masking, count the number of packets falling into the pattern, and calculate the probabilities. The problem with this method is that finding all the error masking patterns is very difficult in some situations. First, when the number of bits corrupted, i, becomes bigger, the number of patterns, $\binom{ns}{i}$, increases until $i \geq ns/2$. Analyzing all these patterns and figuring out which packets fall into them and thus pass the checksum will become difficult. Second, like the case we pointed out, error patterns can be complex when the number of bit errors is greater than or equal to n, even when the value of n may not be too big. In the next section, we provide a new way of calculating the error masking probability. This procedure will count directly the number of packets passing the checksum when i bits are wrong without analyzing the patterns.

III. Error Masking Probability

A. Assumptions

We make the assumption that each bit has the same probability to be 0 or 1. We use this to simplify the analysis, even though for real TCP/UDP packets, the data distribution may not be uniform [6]. Moreover, we assume that each bit has the same probability, p, to be corrupted. These assumptions are also made in previous work [1] [3]. We also assume that the checksum word is independent of the data words, which is also assumed in [1]. If the checksum word can only be treated as dependent, the results can be used as errors occurring only in datum words.

B. Error Value Probability of One Word

For a block of $n \times s$ bits, where the checksum is applied, each row is an n-bit word. Let v_1 be the original decimal value of one word. After transmission, the word has a decimal value of v_2. The error value is defined as $v_2 - v_1$. Let $f(n,i)$ be the probability that error value is i for an n-bit word. In other words, given a possible value of an n-bit word, there is a certain probability that some errors happen and the error value is equal to i. $f(n,i)$ is the sum of this certain probability for all 2^n possible n-bit words. We have the following relations for $f(n,i)$.

Lemma 1: For an n-bit word with bit error probability p, $f(n,i)$, the probability that the error value is i has the following properties:

- $f(n,i) = f(n,-i)$, where $|i| \leq 2^n - 1$
- $f(n,2i) = (1-p)f(n-1,i)$, where $|i| \leq 2^{n-1} - 1$
- $f(n,2i+1) = \frac{p}{2}f(n-1,i) + \frac{p}{2}f(n-1,i+1)$, where $|i| \leq 2^{n-1} - 2$
- $f(1,0) = 1-p$
- $f(1,1) = \frac{1}{2}p$

Proof For one n-bit word, define the complement word as the one which contains the same number of bits, and every bit is the 1's complement of the bit which is in the same position of the original word. For any word that has an error value of i when certain bits are corrupted, its complement word will have an error value of $-i$ when the same bits are erroneous. So, $f(n,i) \leq f(n,-i)$. On the other hand, for every case that has an error value of $-i$, the same probability will cause the error value of i in the complement word, which means that $f(n,-i) \leq f(n,i)$. According to these two relations, $f(n,-i) = f(n,i)$.

For $f(n,2i)$, the only case that causes this error value is that no error happened on the lowest bit, whereas the higher $n-1$ bits have an error value of i. So, $f(n,2i) = (1-p)f(n-1,i)$.

In order to generate $f(n,2i+1)$, there must be an error on the lowest bit. The error value of the higher $n-1$ bits should be i when the error value of the last bit is 1, $i+1$ for -1. Because these two cases enumerate all the error patterns that will give an error value of $2i+1$, $f(n,2i+1)$ can be expressed as $\frac{p}{2}f(n-1,i) + \frac{p}{2}f(n-1,i+1)$.

For a single bit word, the error value is 0 if and only if no error happens. The probability of this situation is $1-p$. If and only if bit 0 is changed to 1 the error value is 1, therefore the probability is $\frac{1}{2}p$. □

C. 1's Complement Error Value Probability of One Word

From $f(n,i)$ we can define $g(n,i)$, the probability of a 1's complement error value, $v_2 - v_1$, where $0 \leq i \leq 2^n - 2$. In other words, $g(n,i)$ is the sum of $f(n,j)$ for all js with the same 1's complement value of i. Now, we assume both error values 0 and $2^n - 1$ will contribute to error masking. The fact that a word composed of all 0s is the only case that generates a checksum of 0 will be considered in section III-D.

Define the sum of one word to be sum. As long as $sum = A \times (2^n - 1) + i$, where A is an integer and $0 \leq i \leq 2^n - 2$, the 1's complement error value is i. So, we have $g(n,i) = f(n,i) + f(n,-(2^n-1)+i)$, where $0 < i \leq 2^n - 2$ and $g(n,0) = f(n,0) + f(n,2^n-1) + f(n,1-2^n)$. For $g(n,i)$, we have the following lemma.

Lemma 2: The following two properties hold for $g(n,i)$, the probability that the 1's complement error value is i for an n-bit word:

- $g(n,i) = g(n, 2^n - 1 - i)$, where $0 < i \leq 2^n - 2$
- $g(n,2i) = g(n,i)$, where $0 < 2i \leq 2^n - 2$

Proof Using the definition of $g(n,i)$ and Lemma 1, we can get $g(n,i) = f(n,i) + f(n,-(2^n-1)+i) = f(n,-i) + f(n,(2^n-1)-i) = f(n,-(2^n-1)+((2^n-1)-i))) + f(n,(2^n-1)-i) = g(n,2^n-1-i)$.

Define the 1-to-left shift operation as shift a word one bit to the left, and the highest bit be moved to the lowest bit. For one word, which has a 1's complement error value of i after transmission, where $0 < i \leq 2^{n-1} - 2$, assume the original value is v_1 and the value is v_2 after transmission. Consider the 1-to-left shift operation on both the words. The original value will be $2v_1$ or $2v_1 - 2^n + 1$. And the value after errors is $2v_2$ or $2v_2 - 2^n + 1$. So the 1's complement error value of the shifted word will be $2i$, which means $g(n,2i) \geq g(n,i)$. Similarly, from the 1-to-right shift operation we can get $g(n,i) \geq g(n,2i)$. Thus, we have $g(n,i) = g(n,2i)$. □

D. 1's Complement Checksum of a Block

Given $g(n,i)$ for one n-bit word, we can calculate the probability of any checksum for an s-word block. We only consider the block where not all the bits are 0s.

The probability that the sum of the error values in the whole block is a multiplication of $2^n - 1$ can be expressed as:

$$\sum_{x_1=0}^{2^n-2} \sum_{x_2=0}^{2^n-2} \cdots \sum_{x_{s-1}=0}^{2^n-2} g(n,x_1)g(n,x_2)\cdots g(n,x_{s-1})g(n,x) \qquad (1)$$

where $0 \leq x \leq 2^n - 2$, and $x_1 + x_2 + \cdots + x_{s-1} + x = i \times (2^n - 1)$, i is an integer.

Remember that checksum of 0 can only be generated when all the bits in one block are 0. But, this expression includes the probability that the checksum of a block is changed from 0 to $2^n - 1$ and vice versa. So we need to remove these two events from expression (1). Define $h(n,i)$ as the probability that the 1's complement error value is i for an n-bit word composed of 0s. Obviously, $h(n,i)$ has the following values:

- $h(n,0) = \frac{1}{2^n}[(1-p)^n + p^n]$
- $h(n,i) = \frac{1}{2^n} p^j (1-p)^{n-j}$, where $0 < i \leq 2^n - 2$ and j is the number of 1 bits in the binary expression of value i.

Since the probability that the checksum of a block is changed from 0 to $2^n - 1$ is the same as that changed from $2^n - 1$ to 0, we can get the probability of errors that can pass the checksum, i.e., the errors resulting in no change to the original non-zero checksum value:

$$\sum_{x_1=0}^{2^n-2} \sum_{x_2=0}^{2^n-2} \cdots \sum_{x_{s-1}=0}^{2^n-2} g(n,x_1)g(n,x_2)\cdots g(n,x_{s-1})g(n,x) \; - \; 2 \times \sum_{x_1=0}^{2^n-2} \sum_{x_2=0}^{2^n-2} \cdots \sum_{x_{s-1}=0}^{2^n-2} h(n,x_1)h(n,x_2)\cdots h(n,x_{s-1})h(n,x) \qquad (2)$$

where $0 \leq x \leq 2^n - 2$, and $x_1 + x_2 + \cdots + x_{s-1} + x = i \times (2^n - 1)$, i is an integer.

This expression is the complete error passing probability, since it enumerates all the cases that give the same non-zero checksum. Following this expression, we can calculate and get the exact probability, whereas the analysis provided in [3] is hard to extend to an arbitrary number of error bits.

For example, when we calculate the probability of error passing for $n = 3$, we can first get the error value probability of 3-bit word following Lemma 1, then $g(3,i)$ according to the definition. Finally we can calculate the error passing probability for a 3-bit word block from expression (2). From Lemma 2, we can get the result that when $i \neq 0$, $g(3,i)$ has the same value, which eases the final calculation a lot. Unfortunately, when $n \neq 3$, $g(n,i)$ lacks such relationships to ease calculating, so too many calculations could be involved. When n is a fixed number, we can determine how much calculation is needed as the block becomes bigger. Only expression (1) itself contains $(2^n - 1)^{s-1}$ terms to be added together, which grows exponentially with s. The number of multiplications needed for each term is $\frac{n^2 s(s-1)}{2}$. The number of additions for each term is $\frac{n^2 s(s-1)}{2} - n$. These two numbers increase almost linearly with s^2. The total number of arithmetic operations grows even faster than an exponential function as s becomes infinite. So as s becomes bigger, the calculation becomes more and more computationally intensive and eventually computationally infeasible.

IV. Estimation of Error Passing Probability

A. Estimation Method

Since the calculation needed for the exact probability computation grows too fast with s, the number of words in a block, estimation of the probability becomes necessary.

There is one method of estimation provided in [3]. The estimation uses the actual error masking probability caused by 2 and 3 bit errors as the lower bound. For an upper bound, all i-bit errors, where $i > 3$, are assumed to result in error passing. The probability of all the i-bit errors is $\binom{ns}{i} p^i (1-p)^{ns-i}$. This probability can be omitted only when $ns \ll \frac{1}{p}$. Under this situation only, the upper bound is almost the same as the lower bound, which means the bounds provide a good estimation.

In order to get a better estimate, we analyze how the error passing probability changes when s is increased for a fixed value of n. Define x_{ab} as the probability that a 1's complement checksum of the error value equals a in an s word block when b errors occur. Define y_{ab} as the probability that the 1's complement error value of one n-bit word is a if b bits get corrupted. y_{ab} is actually a certain term in $g(n,a)$. When one more word is added to the block, $x_{0b}\prime$, the new error passing probability caused by b errors ($b \leq n$), can be calculated as:

$$\begin{aligned} x_{0b}\prime < {} & x_{0b}y_{00} + x_{1b}y_{(2^n-2)0} + \cdots + x_{(2^n-2)b}y_{10} \\ & + x_{0(b-1)}y_{01} + x_{1(b-1)}y_{(2^n-2)1} + \cdots + x_{(2^n-2)(b-1)}y_{11} \\ & + \cdots \\ & + x_{00}y_{0b} + x_{10}y_{(2^n-2)b} + \cdots + x_{(2^n-2)0}y_{1b} \end{aligned} \qquad (3)$$

Since one more word is added, the right part of the inequality includes cases that the checksum of the block is changed from 0 to $2^n - 1$ and vice versa, which should be excluded to get $x_{0b}\prime$.

And the probability that b bits got corrupted, $\binom{ns+n}{i} p^i (1-p)^{ns+n-i}$, can be calculated as

$$\sum_{a=0}^{2^n-2} x_{ab} \sum_{a=0}^{2^n-2} y_{a0} + \sum_{a=0}^{2^n-2} x_{a(b-1)} \sum_{a=0}^{2^n-2} y_{a1} + \cdots + \sum_{a=0}^{2^n-2} x_{a0} \sum_{a=0}^{2^n-2} y_{ab} \qquad (4)$$

If we can get the ratio of expression (3) to expression (4), we can calculate the value of $x_{0b}\prime$. If $\frac{y_{i0}}{\sum_{a=0}^{2^n-2} y_{a0}}$, $\frac{y_{i1}}{\sum_{a=0}^{2^n-2} y_{a1}}$,...,

$\frac{y_{in}}{\sum_{a=0}^{2^n-2} y_{an}}$ all fall in one range for any $0 \le i \le 2^n - 2$, then this ratio should have the same upper bound. Consider the case when $b > n$, we can get the same result. And we will use this idea to estimate the ratio.

B. Upper Limit of Error Passing Probability

Still consider the ratio of expression (3) to expression (4). When $n = 5$, the range of $\frac{y_{ib}}{\sum_{a=0}^{2^n-2} y_{ab}}$, where $0 \le i \le 2^n - 2$, is as follows:

$errors$	0	1	2	3	4	5
$range$	$0-1$	$0-\frac{1}{10}$	$0-\frac{1}{20}$	$0-\frac{1}{20}$	$0-\frac{1}{20}$	$0-\frac{2}{33}$

From the list, it is clear that the common range is $[0, 1]$, which means that the portion of erroneous packets which could pass checksum detection is $[0, 1]$. But, further analysis makes this portion $[0, \frac{1}{10}]$, i.e., the range of non-0 errors.

Notice $y_{00} = 1$, and $y_{i0} = 0$, when $i \neq 0$. Thus, the ratio of the first line in expression (3) to the first term in expression (4) becomes $\frac{x_{0b}}{\sum_{a=0}^{2^n-2} x_{ab}}$. So $\frac{x_{0b'}}{\sum_{a=0}^{2^n-2} x_{ab'}}$ is revised to fall in the common range of $\frac{x_{0b}}{\sum_{a=0}^{2^n-2} x_{ab}}$, $\frac{y_{i1}}{\sum_{a=0}^{2^n-2} y_{a1}}$, $\frac{y_{i2}}{\sum_{a=0}^{2^n-2} y_{a2}}$, $\cdots$, $\frac{y_{ib}}{\sum_{a=0}^{2^n-2} y_{ab}}$. The range of all these ratios except the first one is $[0, \frac{1}{10}]$ for $n = 5$. For a one word block, error masking probability is 0, which also falls in the range $[0, \frac{1}{10}]$. So, no matter how many words are in the block, $\frac{x_{0b'}}{\sum_{a=0}^{2^n-2} x_{ab'}}$ will always fall in $[0, \frac{1}{10}]$.

Similarly, we can get the same result for $b > n$. Actually, we have the following lemma for the coefficient of $g(n, i)$.

Lemma 3: For an n-bit word ($n \ge 4$), $\frac{y_{ib}}{\sum_{a=0}^{2^n-2} y_{ab}} \le \frac{1}{2n}$ where $0 \le i \le 2^n - 2$ and $0 < b \le n$.

Proof The proof is given in Appendix A . □

From this Lemma, we can get directly that for any n-bit word block, $n \ge 4$, less than $\frac{1}{2n}$ of all the i-bit erroneous packets can pass the checksum.

C. Toward Better Estimation of Error Masking Probability

Notice that $y_{0i} = 0$ for $0 < i < n$. Except for an error value of 0, the range of $\frac{y_{ib}}{\sum_{a=0}^{2^n-2} y_{ab}}$, where $0 < i \le 2^n - 2$, may be different from the range we calculated before. For example, the new range for $n = 5$ is as follows:

$errors$	1	2	3	4	5
$range$	$0-\frac{1}{10}$	$\frac{1}{40}-\frac{1}{20}$	$\frac{1}{80}-\frac{1}{20}$	$\frac{1}{40}-\frac{1}{20}$	$\frac{1}{33}-\frac{2}{33}$

Since y_{0i} can be multiplied with only x_{0j}, we can use the new range to calculate the upper bound and lower bound of the probability, which will yield tighter bounds compared to the range of $[0, \frac{1}{2n}]$. For example, the passing probability range with 6 errors for $n = 5$ and $s = 4$ is $[\frac{1}{85.2747}, \frac{1}{14.255}]$, which is tighter than $[0, \frac{1}{10}]$.

Furthermore, we compute some exact values for error passing probability following section III. In the result, the coefficient of item $p^i(1-p)^{ns-i}$ is the number of error patterns that can pass the checksum. Since the total number of i-bit error patterns is $\binom{ns}{i}$, we can calculate the ratio between these two values. The reciprocal value of this ratio is actually the ratio of the total number of erroneous packets to the actual number of passing packets. The results for a 4-bit word block and also a 7-bit word block are given. In table I and table II, the leftmost column gives the possible number of error bits. Every other column presents the result for a block composed of a certain number of words. Each value in these columns represents the reciprocal ratio of erroneous packets which can pass the checksum when the corresponding number of errors happen. For example, a value of 10 means $\frac{1}{10}$ of the erroneous packets pass the checksum.

TABLE I
RECIPROCAL RATIO OF ERRONEOUS PACKETS PASSING CHECKSUM, $n = 4$

number of error bits	number of words in the block							
	2	3	4	5	6	7	8	9
2	14	11	10	9.5	9.2	9	8.85714	8.75
3	28	24.4444	23.3333	22.8	22.4889	22.2857	22.1429	22.037
4	13.0233	13.3476	13.4325	13.4351	13.4167	13.3944	13.373	13.3537
5	14	15.3095	15.6564	15.8003	15.8745	15.9184	15.9468	15.9665
6	14	14.3995	14.4748	14.5067	14.5225	14.531	14.5358	14.5387
7	16	15.4778	15.3098	15.2655	15.2497	15.2431	15.24	15.2385
8	16	15.0716	14.9273	14.8995	14.8908	14.8874	14.8858	14.885
9		15.0267	15.0468	15.0558	15.0582	15.0589	15.059	15.0591
10		14.8995	14.9476	14.9645	14.9686	14.9699	14.9703	14.9706
11		15.0588	15.0107	15.0136	15.0142	15.0144	15.0145	15.0146
12		15.0588	15.0016	14.995	14.9933	14.9929	14.9927	14.9927
13			15.0052	15.0051	15.0041	15.0038	15.0037	15.0037
14			14.9927	14.9978	14.9981	14.9981	14.9982	14.9982
15			15.0037	15.0004	15.0007	15.0009	15.0009	15.0009
16			15.0037	14.9998	14.9995	14.9995	14.9995	14.9995
17				15.0005	15.0003	15.0003	15.0002	15.0002
18				14.9995	14.9999	14.9999	14.9999	14.9999
19				15.0002	15	15.0001	15.0001	15.0001
20				15.0002	15	15	15	15
21					15	15	15	15
22					15	15	15	15
23					15	15	15	15
24					15	15	15	15
25						15	15	15
26						15	15	15
27						15	15	15
28						15	15	15
29							15	15
30							15	15
31							15	15
32							15	15
33								15
34								15
35								15
36								15

From table I and table II, we observe two phenomenon, which also hold for all other values we calculated. Let i be the number of errors in the block. The first feature is that when $1 < i < n$, the larger the block, the greater the proportion of all possible i-bit error packets that can pass checksum. The second feature is that when $i > 2n$, the proportion passing the checksum trends to fall in $[\frac{1}{2^n+1}, \frac{1}{2^n-2}]$. If the first feature really holds, we can achieve a better estimated lower bound. If the second feature holds, we can achieve a better estimate for error passing probabilities with many error bits.

We give the estimated error passing bounds for a 16-bit word block, the same situation as TCP/UDP header checksum. Previous upper bound and previous lower bound are those used in [3], as described in section IV-A. We present our calculated upper bound described at the beginning of section IV-C. For the lower bound, we use the calculated lower bound and the features of the observed phenomenon. For all these bounds, 2 and 3 errors passing probabilities use the exact values from [3]. First, the plot of error bounds for a 10 word block is shown. This is the minimum size of the TCP header. In Figure 1, the comparison of the

TABLE II

RECIPROCAL RATIO OF ERRONEOUS PACKETS PASSING CHECKSUM, $n = 7$

number of error bits	number of words in the block			
	2	3	4	5
2	26	20	18	17
3	104	84.4444	78	74.8
4	114.4	84.4444	74.2857	69.1792
5	163.429	129.2	117.244	111.163
6	163.429	135.777	124.457	118.297
7	138.754	135.822	131.262	128.05
8	120.421	130.3	129.69	128.263
9	108.952	125.698	127.815	127.917
10	106.419	124.402	126.783	127.285
11	110.933	125.038	126.452	126.933
12	118.857	126.639	126.632	126.862
13	128	128.269	126.939	126.911
14	128	129.156	127.151	126.982
15		129.152	127.205	127.026
16		128.54	127.13	127.033
17		127.744	126.999	127.019
18		127.148	126.88	126.999
19		126.909	126.812	126.986
20		127.008	126.807	126.984
21		127.008	126.85	126.991
22			126.912	127.002
23			126.966	127.012
24			126.996	127.017
25			127.003	127.018
26			126.999	127.014
27			127	127.009
28			127	127.004
29				127.001
30				127
31				127
32				127
33				127
34				127
35				127

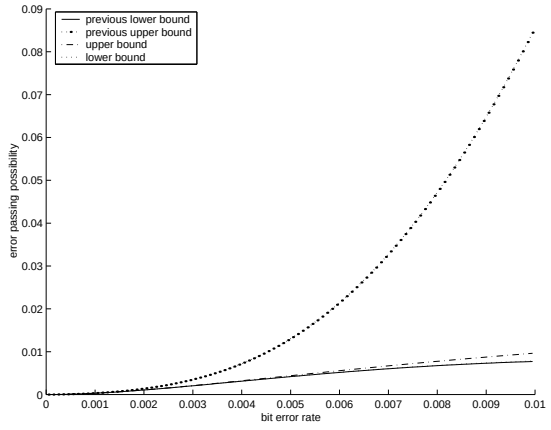

Fig. 1. Error Masking Probability For 10 16-bit Word Block

estimated bounds is given. In order to give a better understanding of how the various bounds differ for higher bit error rates, we also present Figure 2, which displays the error probability using a logarithmic scale on the Y-axis. It is clear that the previous bounds are good approximations for bit error rates less than 10^{-3}, whereas our estimated bounds are good until 10^{-2}. We also give the figures for a 256 16-bit word block, Figure 3 and Figure 4. 512 bytes is a reasonable size for TCP packets in Ethernet. This length could also be proper for wireless links with high Bit Error Rate (BER), since a smaller packet size increases the chance of successful transmission. As we analyzed in section IV-A, the previous bounds are good with a lower BER when

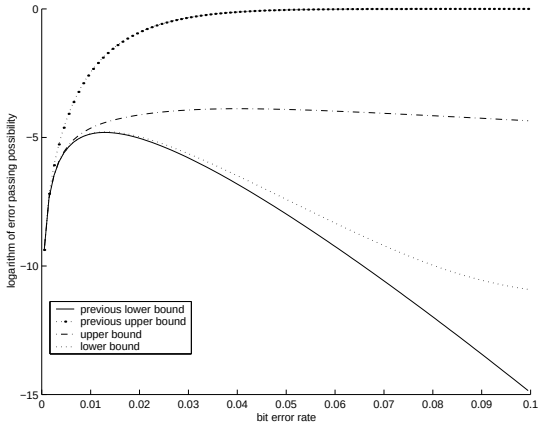

Fig. 2. Error Masking Probability For 10 16-bit Word Block, Logarithm Result

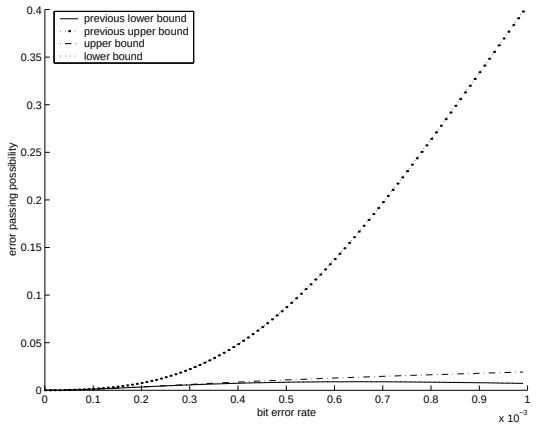

Fig. 3. Error Masking Probability For 256 16-bit Word Block

there are more words in a block. It is shown that the bounds are tight for BER less than 10^{-4}. Again, the new estimations give better bounds until around 10^{-3}. With higher BER, the distance between the bounds remains almost the same, which could be acceptable for many research and analysis purposes.

V. CONCLUSION

So far, we have provided an exact formula for error passing probability for 1's complement checksum. Since too much calculation is needed for bigger data blocks, we also make estimations on the probability. We achieved an upper bound for n-bit word block, $\frac{1}{2n}$, when $n \geq 4$, which is much better than previous work. We can get better bounds through calculation. Even better bounds can be achieved using the observed phenomenon from exact probability of small blocks. The lower bounds achieved through this approach match the calculated upper bounds pretty well.

Further work includes more discussion on the observed features. The influence on error detecting ability caused by some protocol proposals [4] [2] also deserves considerations. In VJ TCP/IP header compression protocol [4], some header fields are transferred using only the difference between those of the previous packet, instead of the original values. The influence on error detecting probability needs further consideration. For the

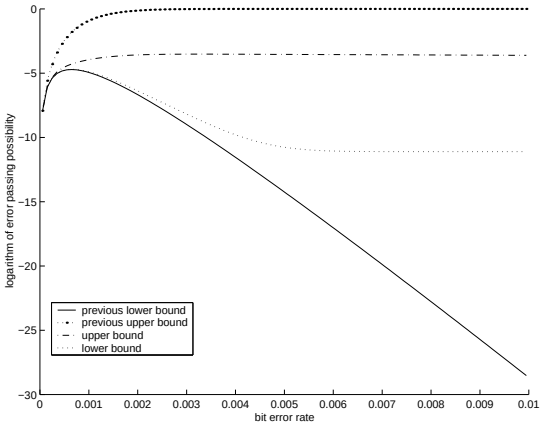

Fig. 4. Error Masking Probability For 256 16-bit Word Block, Logarithm Result

"twice" header compression [2], the TCP header checksum is also used to help recover the value of these fields if some compressed packets are lost. The error masking probability of this algorithm is definitely higher than that of VJ compression, since errors passing CRC could contribute to generating the wrong values of these fields. Thus, this issue needs more careful consideration.

REFERENCES

[1] C. Tzou Chen and G.S. Fang. A Closed-Form Expression for the Probability of Checksum Violation. *IEEE Trans. On Systems, Man, and Cybernetics*, pages Vol. SMC–10, No. 7, July, 1980.

[2] M. Degermark, B. Nordgren, and S. Pink. IP header compression. RFC 2507, Internet Engineering Task Force, February 1999.

[3] Y. Desaki, K. Iwasaki, Y. Miura, and D. Yokota. Double and triple error detecting capability of Internet checksum and estimation of probability of undetectable error. In *Pacific Rim International Symposium on Fault-Tolerant Systems*, pages 47 –52, 1997.

[4] V. Jacobson. IP headers for low-speed serial links. RFC 1144, Internet Engineering Task Force, February 1990.

[5] T. Mallory and A. Kullberg. Incremental Updating of the Internet Checksum. RFC 1141, Internet Engineering Task Force, January 1990.

[6] C. Partridge, J. Hughes, and J. Stone. Performance of checksums and CRCs over real data. *SIGCOMM '95*, page October, vol. 25, no. 4, 1995.

[7] J. Postel. User Datagram Protocol. RFC 768, Internet Engineering Task Force, August 1980.

[8] J. Postel. Transmission Control Protocol. RFC 793, Internet Engineering Task Force, September 1981.

[9] A. Rijsinghani. Computation of the Internet Checksum via Incremental Update. RFC 1624, Internet Engineering Task Force, May 1994.

[10] A. K. Salkintzis. A Survey of Mobile Data Networks. *IEEE Communications Surveys*, pages Vol 2, No.3, Third Quarter, 1999.

[11] Nirmal R. Saxena and Edward J. McCluskey. Analysis of Checksums, Extended-Precision Checksums, and Cyclic Redundancy Checks. *IEEE Trans. on Computers*, pages Vol. 39, No. 7, July, 1990.

[12] J. Stone and C. Partridge. When The CRC and TCP Checksum Disagree. *ACM SIGCOMM*, September 2000.

APPENDIX

I. PROOF OF LEMMA 3

First, for the coefficient of $f(n,x)$, we have the following lemma.

Lemma 4: Define $0 < |i| \leq 2^n - 2$,

- the coefficient of p^n in $f(n,i)$ is $\frac{1}{2^n}$, if i is an odd value
- the coefficient of p^n in $f(n,i)$ is 0, if i is an even value
- the coefficient of $p^{n-1}(1-p)$ in $f(n,i)$ is less than or equal to $\frac{2(n-1)}{2^n}$, if i is an odd value
- the coefficient of $p^{n-1}(1-p)$ in $f(n,i)$ is less than or equal to $\frac{2}{2^n}$, if i is an even value
- the coefficient of $p^x(1-p)^{n-x}$ in $f(n,i)$ is less than or equal to $\frac{1}{2}$, for $1 < x < n-1$

Proof (by induction) From the expression of $f(n,1)$ and $f(n,2^n-1)$, the relation holds for $i=1$ and $i=2^n-1$ for any value of n. Calculate $f(1,i)$ and $f(2,i)$ directly, we know that the relation holds for $n=1$ and $n=2$.

Suppose that the relation also holds for any value of n.

Consider the situation for $n+1$. For an odd value of i, where $i = 2 \times j + 1$, from Lemma 1, we have $f(n+1, 2j+1) = \frac{p}{2}f(n,j) + \frac{p}{2}f(n,j+1)$. From the assumption,the coefficient of p^{n+1} is $\frac{1}{2^{n+1}}$. Similarly, the maximum value of the coefficient of $p^n(1-p)$ is $\frac{1}{2} \times (\frac{2(n-1)}{2^n} + \frac{2}{2^n}) = \frac{2n}{2^{n+1}}$. For other values of x, the coefficient is less than or equal to $\frac{1}{2}$.

For an even value of i, where $i = 2 \times j$, from Lemma 1, we have $f(n+1, 2j) = (1-p)f(n,j)$. So the coefficient of p^{n+1} is 0. The coefficient of $p^n(1-p)$ is less than or equal to $\frac{1}{2^n}$. For term $p^{n-1}(1-p)^2$, the maximum coefficient is either $\frac{2(n-1)}{2^n} \leq \frac{1}{2}$ or $\frac{2}{2^n} \leq \frac{1}{2}$. For other values of x, the maximum value of the coefficient is $\frac{1}{2}$.

So, the relationship holds for $n+1$. Hence, the relation holds for any value of n. □

Proof of Lemma 3

y_{ab} is actually a certain term in $g(n,a)$. And, $\sum_{a=0}^{2^n-2} y_{ab} = \binom{n}{b}p^b(1-p)^{n-b}$. By definition, $g(n,0) = (1-p)^n + \frac{2}{2^n}p^n$. For $0 < a < 2^n - 1$, $g(n,a) = f(n,a) + f(n,-(2^n-1)+a)$. We can analyze the coefficients according to Lemma 4.

When $b = 1$, the error value can only be power of 2. The maximum coefficient of $g(n,i)$ is $\frac{2^{n-1}}{2^n}$ or 0. So, $\frac{y_{i1}}{\sum_{a=0}^{2^n-2} y_{a1}} \leq \frac{\frac{2^{n-1}}{2^n}}{\binom{n}{1}} = \frac{1}{2n}$.

When $b = n$, the coefficient of $g(n,0)$ is $\frac{2}{2^n}$; the coefficient of $g(n,i)$, when $i \neq 0$ is $\frac{1}{2^n}$. Plus, $\frac{2}{2^n} \leq \frac{1}{2n}$ when $i \geq 4$. So $\frac{y_{in}}{\sum_{a=0}^{2^n-2} y_{an}} \leq \frac{\frac{2}{2^n}}{\binom{n}{n}} = \frac{2}{2^n} \leq \frac{1}{2n}$.

When $b = n-1$, the coefficient of $g(n,0)$ is 0. The maximum value of (g,i), where $i \neq 0$, is $\frac{2(n-1)}{2^n} + \frac{2}{2^n} = \frac{2n}{2^n}$. So $\frac{y_{i(n-1)}}{\sum_{a=0}^{2^n-2} y_{a(n-1)}} \leq \frac{\frac{2n}{2^n}}{\binom{n}{n-1}} = \frac{2}{2^n} \leq \frac{1}{2n}$

For all other values of b, the coefficient of $g(n,b)$ is equal to or less than $\frac{1}{2} + \frac{1}{2} = 1$. And $\frac{y_{ib}}{\sum_{a=0}^{2^n-2} y_{ab}} \leq \frac{1}{\binom{n}{b}} \leq \frac{1}{\binom{n}{2}}$. When $n > 4$, $\binom{n}{2} \geq 2n$, so $\frac{y_{ib}}{\sum_{a=0}^{2^n-2} y_{ab}} \leq \frac{1}{2n}$. When $n = 4$, calculating $g(4,i)$ directly shows the maximum coefficient of term $p^2(1-p)^2$ is $\frac{12}{1}$, so $\frac{y_{i2}}{\sum_{a=0}^{15} y_{a2}} \leq \frac{12/1}{\binom{4}{2}} = \frac{1}{8} = \frac{1}{2n}$. □