

## Short description of the Internet checksum

### IP checksum definition

The **IP checksum** is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header.

One question many people may ask is "What is the 1's complement sum ?". This is because all computers utilize the 2's complement representation and the 1's complement is not used. The following gives a short introduction.

#### 2's complement fixed point integers (8-bit)

Binary	Decimal	Hex
0000 0000	0	00
0000 0001	1	01
0000 0010	2	02
0000 0011	3	03
1111 1111	-1	FF
1111 1110	-2	FE
1111 1101	-3	FD

Let's add two integers:

$$-3 + 5 = 2$$

$$FD + 05 = 01 02$$

**Discarding** the carry (01) gives the correct result.

#### 1's complement fixed point integers (8-bit)

Binary	Decimal	Hex
0000 0000	0	00
0000 0001	1	01
0000 0010	2	02
0000 0011	3	03
1111 1111	-0	FF
1111 1110	-1	FE
1111 1101	-2	FD
1111 1100	-3	FC

Add the same numbers:

$$-3 + 5 = 2$$

$$FC + 05 = 01 01$$

**Adding** the carry (01) to the LSB (01) gives the correct result:

$$01 + 01 = 02$$

So, the 1's complement sum is done by summing the numbers and adding the carry (or carries) to the result..

#### Simple Internet checksum example

Suppose we have an 8-bit, 2's complement, machine and send the packet

FE 05 00

where 00 is the checksum field.

Let's calculate and verify the Internet checksum.

$$FE + 05 = 01\ 03$$

This is the result of the normal (2's complement) addition. The 1's complement sum requires the addition of the carry to the 8-bit word (even though we will not get the same result)

$$03 + 01 = 04$$

so the 1's complement sum of FE + 05 is 04.

The 1's complement of the 1's complement sum (Internet checksum) will be

$$\sim 04 = FB \qquad \text{1's complement of 04 (0000 0100) = 1111 1011 (FB)}$$

and the packet will be sent as

FE 05 FB

Now, at the receiving end we add all the received bytes, including the checksum (again using the 2's complement representation)

$$FE + 05 + FB = 01\ FE$$

The 1's complement sum is

$$FE + 01 = FF = -0$$

which checks that the transmission was OK (see below).

### **A more complex example (32-bit machine)**

As shown in **RFC 1071**, the checksum calculation is done in the following way:

(1) Adjacent octets to be checksummed are paired to form 16-bit integers, and the 1's complement sum of these 16-bit integers is formed.

(2) To generate a checksum, the checksum field itself is cleared, the 16-bit 1's complement sum is computed over the octets concerned, and the 1's complement of this sum is placed in the checksum field.

(3) To check a checksum, the 1's complement sum is computed over the same set of octets, including the checksum field. If the result is all 1 bits (-0 in 1's complement arithmetic), the check succeeds.

Packet

01 00 F2 03 F4 F5 F6 F7 00 00

(00 00 is the checksum field)

Form the 16-bit words

0100 F203 F4F5 F6F7

Calculate ~~2's complement~~ sum

$0100 + F203 + F4F5 + F6F7 = 0002\ DEEF$  (store the sum in a 32-bit word)

Add the carries (0002) to get the 16-bit 1's complement sum

$DEEF + 002 = DEF1$

Calculate 1's complement of the 1's complement sum

$\sim DEF1 = 210E$

We send the packet including the checksum 21 0E

01 00 F2 03 F4 F5 F6 F7 21 0E

At the receiving

$0100 + F203 + F4F5 + F6F7 + 210E = 0002\ FFFD$

$FFFD + 0002 = FFFF$

which checks OK.

### **Comments**

It may look awkward to use a 1's complement addition on 2's complement machines. This method however has its own benefits.

Probably the most important is that it is endian independent. Little Endian computers store hex numbers with the LSB last (Intel processors for example). Big Endian computers put the LSB first (IBM mainframes for example). When carry is added to the LSB to form the 1's complement sum (see the example) it doesn't matter if we add  $03 + 01$  or  $01 + 03$ . The result is the same.

Other benefits include the easiness of checking the transmission and the checksum calculation plus a variety of ways to speed up the calculation by updating only IP fields that have changed.

More TCP/IP Software Development Info at [www.netfor2.com](http://www.netfor2.com).

Alex Urich

[alex@netfor2.com](mailto:alex@netfor2.com)