IP Checksum Covert Channels and Selected Hash Collision

Christopher Abad
aempirei@ucla.edu

A fundamental flaw in the design of the Internet checksum, the primary data checksum facility for network data, can allow a malicious user to embed covert channel data in the checksum field itself using a hash collision. What I will demonstrate is the two-way nature of this facility and a covert channel scheme for sending data through the Internet checksum.

The Internet checksum works on a set of 16-bit words (referred to as WORDS), for example a protocol header broken into two-octet chunks. We use the one-place $\Sigma(X)$ function symbol to mean the sum of all elements in the set X of WORDS. Because this sum may exceed the capacity of a WORD, $0 \leq w < 2^{16}$, we will express this sum as in terms of two WORDS, c, the carry bits, and m, which later will be our secret message. First we express the sum of all WORDS in W, a selected set of WORDS, in the form

$$\Sigma W = 2^{16}c + m \; ; \; 0 \leq m < 2^{16}$$

This holds true due to the division theorem. We use the one-place $\neg(x)$ function symbol to mean the one's-compliment of the integer value x with trimming to a maximum width of 16-bits. The Internet checksum is defined as the 16-bit one's-compliment of the sum of all WORDS plus the carry bits. To calculate the checksum we would take

$$S = \neg(c + m)$$

If we choose an insignificant member of W to be a pivotal value w^*, which will be dependant on our message m, and define

$$W_0 = W - \{w^*\}$$

This will allow us to work with w^*, adjusting it to fit our selected secret message. To facilitate this, we allow w^* to occur as a dependant variable in

$$\Sigma W = w^* + \Sigma W_0 = 2^{16}c + m \Rightarrow w^* = 2^{16}c + m - \Sigma W_0$$

We can now define m to be our message we would like to send over our covert channel. We know that $0 \leq m < 2^{16}$ and therefore if

$$0 \leq w^* < 2^{16} \Rightarrow 0 \leq 2^{16}c + m - \Sigma W_0 < 2^{16}$$

holds true, meaning a WORD sided w^* can be calculated, any arbitrary message may be sent and a hash collision will be guaranteed. We must only insure that c can always be chosen to meet the required restraints of the inequality above, due to the fact that m and ΣW_0 are known constants. Now $\Sigma W_0 - m$ is set as constant because m and ΣW_0 are known at time of calculation, so simply

$$let\ k = \Sigma W_0 - m = 2^{16}k_0 + r\ ;\ 0 \leq r < 2^{16}$$

which can be expressed in terms of k_0 and r due to the division theorem, therefore we can express the *w** inequality in the form

$$2^{16}k_0 + r \leq 2^{16}c < 2^{16} + 2^{16}k_0 + r$$

and show it to hold true for any value of *r* and k_0. The case where $r = 0$ would be

$$2^{16}k_0 \leq 2^{16}c < 2^{16} + 2^{16}k_0 \Rightarrow k_0 \leq c < k_0 + 1$$

which holds when $c = k_0$ for any value of k_0. Otherwise, if $r > 0$ then

$$let\ c = k_0 + 1$$
$$2^{16}k_0 \leq 2^{16}k_0 + 2^{16} - r < 2^{16}k_0 + 2^{16} \Rightarrow 0 \leq 2^{16} - r < 2^{16}$$

which holds when $0 < r < 2^{16}$ for any value of k_0, which agrees with the prior restraints on *r* for this case, therefore values for *c* and *w** can generated for any arbitrarily selected value of our message *m*.
QED

What follows is a method for message generation, and an example dataset.

W_0 = { 32531, 12431, 1421, 15236, 31511 }	
Select a message, m | $0 \leq m < 65536$	m = 6534
Calculate ΣW_0	$\Sigma W_0 = 93130$
Let $k = \Sigma W_0 - m$	k = 86596
Find k_0, r | $k = 2^{16}k_0 + r$	$k_0 = 1$, r = 21060
Solve for c | $k \leq 2^{16}c < 2^{16} + k$	c = 2
Let $w^* = 2^{16}c + m - \Sigma W_0$	w* = 44476
Let $S = \neg(c + m)$	S = 58999

A case-specific verification of our example dataset:

$W = W_0 \cup w^*$
Therefore W = { 32531, 12431, 1421, 15236, 31511, 44476 }
$\Sigma W = 137606 = 2^{16}(2) + 6534$
m = 6534
$S = \neg(2 + 6534) = \neg 6536 = 58999$

An example of how this can be used in the IP header would be the following: Set up an IP header with an additional 4 octets for IP options, set the first WORD of the options to 0 (end-of-options), and allow the second octet to be w*, which will be calculated later. Allow W_0 to be the set of WORDS in the IP header, not including w*. Allow for S to be the IP checksum, not yet calculated. Apply the method for message generation, selecting m to be our 16-bit message. Calculate w* and S.

This method can be used for any protocol that uses the Internet checksum, including ICMP, UDP, TCP, as well as many others. The most interesting use though comes from the IP header, because the fact that upon forwarding the packet to the gateway, and along each intermediate router, the TTL is decremented, and the checksum is recalculated, therefore losing the immediate covert-channel checksum. The end destination, in order to retrieve the original checksum, must replace the TTL with the original TTL and calculate the sum in the normal fashion, and then retrieve m. An extension to this would be to use the IP ID field as a 32-bit 'key', which the target node must also replace in order to retrieve the message.

In conclusion, this paper should have clearly demonstrated the fact that the internet checksum fails to be a secure method for validating data integrity because of the ability for a user to arbitrarily create a selected collision in the hashing mechanism in a trivial period of time, and because the fact that the original message can be retrieved from the hash, this demonstrates the two-way characteristic of the checksum function. As an alternative to the Internet checksum, a lightweight one-way hash function might want to be standardized during the integration of widespread IPv6.