TCP 64-bit extension: "Modern Variation"
===========================================
for Internet Protocol - Five Fields

Abstract

This document attempts to modernize TCP protocol for new reality, faster bandwidth, encryption-optimization and optional checksums, which is required for Identity-Locator Network Protocol (ILNP) compatibility.

This extension is backwards compatible with the original TCP specification during session establishment, but not compatible during the rest of the session nor with deployed middleboxes.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
Introduction

TCP in IP-FF comes in several variations. The question is:

- Our operating systems and processors are 64-bit.
- Why not make TCP 64-bit also?
- Well, I decided to define what TCP 64-bit should look like.

The session beings with the good-old, time-tested "Classic variation", which looks familiar.

1. TCP Header: "Classic variation"

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Source Port</td>
<td>Destination Port</td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Sequence Number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Acknowledgment Number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Data</td>
<td>Reserved Flags</td>
<td>Window</td>
</tr>
<tr>
<td></td>
<td>Offset</td>
<td>R</td>
<td>E</td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Checksum</td>
<td>Urgent Pointer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Options</td>
<td>Padding</td>
<td></td>
</tr>
<tr>
<td></td>
<td>data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

and only during SYN/ACK, it MAY be moved to a different variation.

2. TCP Header: "Modern variation" a.k.a TCP.64

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Source Port</td>
<td>Destination Port</td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Reserved Flags</td>
<td>Window</td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Design note: Bloated for a good reason.

I realize the downside of making TCP bloated by a whopping extra 16 bytes, but I also realize this is a necessary evil at speeds over 100 Gigabits-per-Second.
If you’re on a slow link, just don’t advertise that you’re TCP 64-bit capable, and stay on the "classic variation".

64-bit Checksums:

16-bit checksums of "classic variation" may fail badly there. Today, only Data-Link layer checksum saves Internet from complete breakdown, as those checksums are fairly strong 32-bit CRCs.

But strong CRC64c checksums is an adequate protection for future huge amounts of unencrypted data.
Going encryption of-course renders checksums useless.

64-bit Sequence numbers and acknowledgements:

The problem with 32-bit SYN/ACK is TCP Reliability

Quote from RFC-7323:

"An especially serious kind of error may result from an accidental reuse of TCP sequence numbers in data segments. TCP reliability depends upon the existence of a bound on the lifetime of a segment: the "Maximum Segment Lifetime" or MSL.

Duplication of sequence numbers might happen in either of two ways:

(1) Sequence number wrap-around on the current connection

A TCP sequence number contains 32 bits. At a high enough transfer rate of large volumes of data (at least 4 G1B in the same session), the 32-bit sequence space may be "wrapped" (cycled) within the time that a segment is delayed in queues."
Earlier incarnation of the connection

Suppose that a connection terminates, either by a proper close sequence or due to a host crash, and the same connection (i.e., using the same pair of port numbers) is immediately reopened. A delayed segment from the terminated connection could fall within the current window for the new incarnation and be accepted as valid.

Duplicates from earlier incarnations, case (2), are avoided by enforcing the current fixed MSL of the TCP specification, as explained in Section 5.8 and Appendix B. In addition, the randomizing of ephemeral ports can also help to probabilistically reduce the chances of duplicates from earlier connections. However, case (1), avoiding the reuse of sequence numbers within the same connection, requires an upper bound on MSL that depends upon the transfer rate, and at high enough rates, a dedicated mechanism is required.

On a gigabit link, Sequence numbers are rotated every 17 seconds. On a 100-gigabit link, this is well under a second.

TCP originally was never designed for such speeds.

This is dangerous, because packets from older rotation might get stuck in queue, then released by a router, get through and corrupt user data if sent to destination, or cause a TCP reset, if sent as an ack to the sender.

And those old packets do have correct sequence number and correct checksum.

PAWS, a system designed to prevent such issues by using timers, may fail badly and produce bad errors with 32-bit Sequence number, due to inaccurate timing issues.

Data Offset: 5 bits; not including the first 28 bytes.

(this allows far more options in to be sent in TCP)

Compatibility Note: 64-bit TCP breaks compatibility with existing "middleboxes".

64-bit Initial sequence number:

Must be copied from the lower 32-bits to the upper 32-bits, if not specified during SYN.

The actual switching to "Modern variation" happens from SYN/ACK packet (3rd handshake), assuming the "ACK" advertises such capability.

2.1. TCP Header: "Modern variation without CRC" or TCP.64-NOCRC

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Source Port</td>
<td>Destination Port</td>
<td></td>
</tr>
<tr>
<td>+-------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Reserved Flags</td>
<td>Window</td>
<td></td>
</tr>
<tr>
<td>+-------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Data</td>
<td>Offset</td>
<td>Data Offset</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>C</td>
<td>E</td>
</tr>
</tbody>
</table>
Initiating a TCP.64 Session "Modern Variation"

This is the same signaling as for initiating a normal TCP connection, but the SYN, SYN/ACK, and ACK packets also carry the 64BIT_CAPABLE option.

Client                                  Server
------                                  ------
64BIT_CAPABLE            ->
[SYN flag]                      [ACK flag]
64BIT_CAPABLE            ->
[SYN+ACK flags]

TCP "Modern Variation" Option:

+--------+--------+--------+
| Kind=31 | Length  | Variation| or ISN |
+--------+--------+--------+

Kind: 31 (To be determined by IANA)

Length:

3 = For setting variation

6 = For upper 32-bits of initial sequence number.

Variations or codes:
0 = Checksum ignored (useful for ILNP);
   Takes effect from 1st packet. (initial SYN)

1 = Modern Variation (TCP.64-bit capable)

2 = Modern Variation, without CRC field; (useful for encryption)
   Takes effect after 64 KiB of data.

   If Len=6, this field is used to setup an upper 32-bits of
   the initial sequence number.

4. Checksum ignored option

   This option lets the receiver to ignore TCP-supplied checksum.
   Affects both classic 16-bit checksum as well as CRC.
   Affects TCP session from the initial SYN packet.

   This allows for Identity-Locator Network Protocol (ILNP) to
   function with TCP/IP networks.
   In this case ILNP or other protocol should provide their own
   checksums for both TCP and IP.

Acknowledgements:

   Influenced by the hard work of R. Ullmann "TP/IX: The Next Internet"
   [RFC-1475] and by "Identifier-Locator Network Protocol (ILNP)",
   [RFC-6740] written by "Randall J. Atkinson" and "SN Bhatti".

   Thanks to:
   for [RFC-1323] and [RFC-7323]; "TCP Extensions for High Performance".

   And big thanks to DARPA for the original specification of TCP, as
   defined in [RFC-793] !

Authors’ Contacts

   Alexey Eromenko
   Israel
   Skype: Fenix_NBK_
   EMail: al4321@gmail.com
   Facebook: https://www.facebook.com/technologov

INTERNET-DRAFT
Alexey
expiration date: 2016-06-16