TCP over Constrained-Node Networks
draft-gomez-core-tcp-constrained-node-networks-00

Abstract

This document provides a profile for the Transmission Control Protocol (TCP) over Constrained-Node Networks (CNNs). The overarching goal is to offer simple measures to allow for lightweight TCP implementation and suitable operation in such environments.

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."

This Internet-Draft will expire on December 12, 2016.

Copyright Notice

Copyright (c) 2016 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.
1. Introduction

The Internet Protocol suite is being used for connecting Constrained-Node Networks (CNNs) to the Internet, enabling the so-called Internet of Things (IoT) [RFC7228]. In order to meet the requirements that stem from CNNs, the IETF has produced a suite of protocols specifically designed for such environments [I-D.ietf-lwig-energy-efficient].

At the application layer, the Constrained Application Protocol (CoAP) was developed over UDP [RFC7252]. However, the integration of some CoAP deployments with existing infrastructure is being challenged by middleboxes such as firewalls, which may limit UDP-based communications. This is one of the main reasons why a CoAP over TCP specification is being developed [I-D.tschofenig-core-coap-tcp-tls].

On the other hand, other application layer protocols not specifically designed for CNNs are also being considered for the IoT space. Some examples include HTTP/2 and even HTTP/1.1, both of which run over TCP by default [RFC7540][RFC2616]. TCP is also used by non-IETF application-layer protocols in the IoT space such as MQTT and its lightweight variants [MQTTS].

This document provides a profile for TCP over CNNs. The overarching goal is to offer simple measures to allow for lightweight TCP implementation and suitable operation in such environments.
1.1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL","SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]

2. Characteristics of CNNs relevant for TCP

Constrained nodes are characterized by significant limitations on processing, memory, and energy resources [RFC7228]. The first two dimensions pose constraints on the complexity and on the memory footprint of the protocols that constrained nodes can support. The latter requires techniques to save energy, such as radio duty-cycling in wireless devices [I-D.ietf-lwig-energy-efficient], as well as minimization of the number of messages transmitted/received (and their size).

Constrained nodes often use physical/link layer technologies that have been characterized as ‘lossy’. Many such technologies are wireless, therefore exhibiting a relatively high bit error rate. However, some wired technologies used in the CNN space are also lossy (e.g. Power Line Communication).

Some CNNs follow the star topology, whereby one or several hosts are linked to a central device that acts as a router connecting the CNN to the Internet. CNNs may also follow the multihop topology [RFC6606].

3. TCP over CNNs

3.1. Maximum Segment Size (MSS)

Some link layer technologies in the CNN space are characterized by a short data unit payload size, e.g. up to a few tens or hundreds of bytes. For example, the maximum frame size in IEEE 802.15.4 is 127 bytes.

6LoWPAN defined an adaptation layer to support IPv6 over IEEE 802.15.4 networks. The adaptation layer includes a fragmentation mechanism, since IPv6 requires the layer below to support an MTU of 1280 bytes [RFC2460], while IEEE 802.15.4 lacked fragmentation mechanisms. 6LoWPAN defines an IEEE 802.15.4 link MTU of 1280 bytes [RFC4944]. Other technologies, such as Bluetooth LE [RFC7668], ITU-T G.9959 [RFC7428] or DECT-ULE [I-D.ietf-6lo-dect-ule], do support link layer fragmentation. By exploiting this functionality, the adaptation layers to enable IPv6 over such technologies also support an MTU of 1280 bytes.
In order to avoid IP layer fragmentation, the TCP MSS MUST NOT be set to a value greater than 1220 bytes in CNNs. (Note: IP version 6 is assumed.) In any case, the TCP MSS MUST NOT be set to a value leading to an IPv6 datagram size exceeding 1280 bytes.

3.2. Window Size

As per this document, the TCP window size MUST have a size of one segment. This value is appropriate for simple message exchanges in the CNN space, reduces implementation complexity and memory requirements, and reduces overhead (see section 3.6).

A TCP window size of one segment follows the same rationale as the default setting for NSTART in [RFC7252], leading to equivalent operation when CoAP is used over TCP.

3.3. RTO estimation

Traditionally, TCP has used the well known RTO estimation algorithm defined in [RFC6298]. However, experimental studies have shown that another algorithm such as the RTO estimator defined in [I-D.bormann-core-cocoa] (hereinafter, CoCoA RTO) outperforms state-of-art algorithms designed as improvements to RFC 6298 for TCP, in terms of packet delivery ratio, settling time after a burst of messages, and fairness (the latter is specially relevant in multihop networks connected to the Internet through a single device, such as a 6LoWPAN Border Router (6LBR) configured as a RPL root) [Commag]. In fact, CoCoA RTO has been designed specifically considering the challenges of CNNs, in contrast with the RFC 6298 RTO. Therefore, as per this document, CoCoA RTO SHOULD be used in TCP over CNNs. Alternatively, implementors MAY choose the RTO estimation algorithm defined in RFC 6298. One of the two RTO algorithms MUST be implemented.

3.4. Keep-alive and TCP connection lifetime

In CNNs, a TCP connection SHOULD be kept open as long as the two TCP endpoints have more data to exchange or it is envisaged that further segment exchanges will take place within an interval of two hours since the last segment has been sent. A greater interval MAY be used in scenarios where applications exchange data infrequently.

TCP keep-alive messages [RFC1122] MAY be supported by a server, to check whether a TCP connection is active, in order to release state of inactive connections. This may be useful for servers running on memory-constrained devices.
Since the keep-alive timer may not be set to a value lower than two hours [RFC1122], TCP keep-alive messages are not useful to guarantee that filter state records in middleboxes such as firewalls will not be deleted after an inactivity interval typically in the order of a few minutes [RFC6092]. In scenarios where such middleboxes are present, alternative measures to avoid early deletion of filter state records (which might lead to frequent establishment of new TCP connections between the two involved endpoints) include increasing the initial value for the filter state inactivity timers (if possible), and using application layer heartbeat messages.

3.5. Explicit congestion notification

Explicit Congestion Notification (ECN) [RFC3168] MAY be used in CNNs. ECN allows a router to signal in the IP header of a packet that congestion is arising, for example when queue size reaches a certain threshold. If such a packet encapsulates a TCP data packet, an ECN-enabled TCP receiver will echo back the congestion signal to the TCP sender by setting a flag in its next TCP ACK. The sender triggers congestion control measures as if a packet loss had happened. In that case, when the congestion window of a TCP sender has a size of one segment, the TCP sender resets the retransmit timer, and will only be able to send a new packet when the retransmit timer expires [RFC3168]. Effectively, the TCP sender reduces at that moment its sending rate from 1 segment per RTT to 1 segment per default RTO.

ECN can reduce packet losses, since congestion control measures can be applied earlier than after the reception of three duplicate ACKs (if the TCP sender window is large enough, which will not happen as per section 3.2 of this document) or upon TCP sender RTO expiration [RFC2884]. Therefore, the number of retries decreases, which is particularly beneficial in CNNs, where energy and bandwidth resources are typically limited. Furthermore, latency and jitter are also reduced.

ECN is also appropriate in CNNs, since in these environments transactional type interactions are a dominant traffic pattern. Exploiting other possible congestion signals such as the reception of three duplicate ACKs would require the use of greater TCP window sizes than the one specified in this document.

3.6. TCP options

Because this specification mandates a TCP window size of one segment, the following TCP options MUST NOT be supported in CNNs: Window scale [RFC1323], TCP Timestamps [RFC1323], and Selective Acknowledgements (SACK) [RFC2018]. Other TCP options SHOULD NOT be used, in keeping with the principle of lightweight operation.
3.7. Explicit loss notifications

There has been a significant body of research on solutions capable of explicitly indicating whether a TCP segment loss is due to corruption, in order to avoid activation of congestion control mechanisms [ETEN] [RFC2757]. While such solutions may provide significant improvement, they have not been widely deployed and remain as experimental work. In fact, as of today, the IETF has not standardized any such solution.

4. Security Considerations

TBD

5. Acknowledgments

Carles Gomez has been funded in part by the Spanish Government (Ministerio de Educacion, Cultura y Deporte) through the Jose Castillejo grant CAS15/00336. His contribution to this work has been carried out during his stay as a visiting scholar at the Computer Laboratory of the University of Cambridge.

6. References

6.1. Normative References


Gomez & Crowcroft Expires December 12, 2016 [Page 6]


6.2. Informative References


[I-D.tschofenig-core-coap-tcp-tls]


Authors’ Addresses

Carles Gomez
UPC/i2CAT
C/Esteve Terradas, 7
Castelldefels 08860
Spain
Email: carlesgo@entel.upc.edu

Jon Crowcroft
University of Cambridge
JJ Thomson Avenue
Cambridge, CB3 0FD
United Kingdom
Email: jon.crowcroft@cl.cam.ac.uk