Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 2016

H. Kitamura
NEC Corporation
S. Ata
Osaka City University
M. Murata
Osaka University
October 16, 2015

"Sharp Close": Elimination of TIME-WAIT state of TCP connections
<draft-kitamura-tcp-sharp-close-02.txt>

Abstract

This document describes an idea "Sharp Close" that eliminates or minimizes TIME-WAIT state of TCP connections.

In the current TCP specification ([RFC0793]), there are some inappropriate or not up-to-date functions. Here we focus and discuss on TCP TIME-WAIT state function.

TIME-WAIT is the last state of TCP connections of Active Close side nodes. After TCP connections are effectively closed, state of them move to TIME-WAIT state. After TIME-WAIT state is finished, resources of connections are released. This means that even if connections are effectively finished, resources of connections are NOT released. The TIME-WAIT state prevents from releasing them.

From the viewpoints of current high-speed and high-multiplicity communication styles, it is thought that TIME-WAIT state is one of evil functions.

In order to provide efficient communications that match current styles, an idea "Sharp Close" that eliminates or minimizes TIME-WAIT state of TCP connections is proposed.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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

Table of Contents

1. Introduction

This document describes an idea "Sharp Close" that eliminates or minimizes TIME-WAIT state of TCP connections.

In the current TCP specification ([RFC0793]), there are some inappropriate or not up-to-date functions. Here we focus and discuss on TCP TIME-WAIT state function.

TIME-WAIT is the last state of TCP connections of Active Close side nodes. After TCP connections are effectively closed, state of them move to TIME-WAIT state. [RFC0793] defines that the connections stay there 2MSL(Maximum Segment Lifetime) seconds. (2MSL = 240 sec.)

After TIME-WAIT state is finished, resources of connections are released. This means that even if connections are effectively finished, resources of connections are NOT released. The TIME-WAIT state prevents from releasing them.

From the viewpoints of current high-speed and high-multiplicity communication styles that require highly resource recycling, it is thought that TIME-WAIT state is one of evil functions.

In order to provide efficient communications that match current styles, an idea "Sharp Close" that eliminates or minimizes TIME-WAIT state of TCP connections is proposed.

In the following sections, analysis of current TIME-WAIT state and design of "Sharp Close" etc. are described.

2. Analysis of current TIME-WAIT state

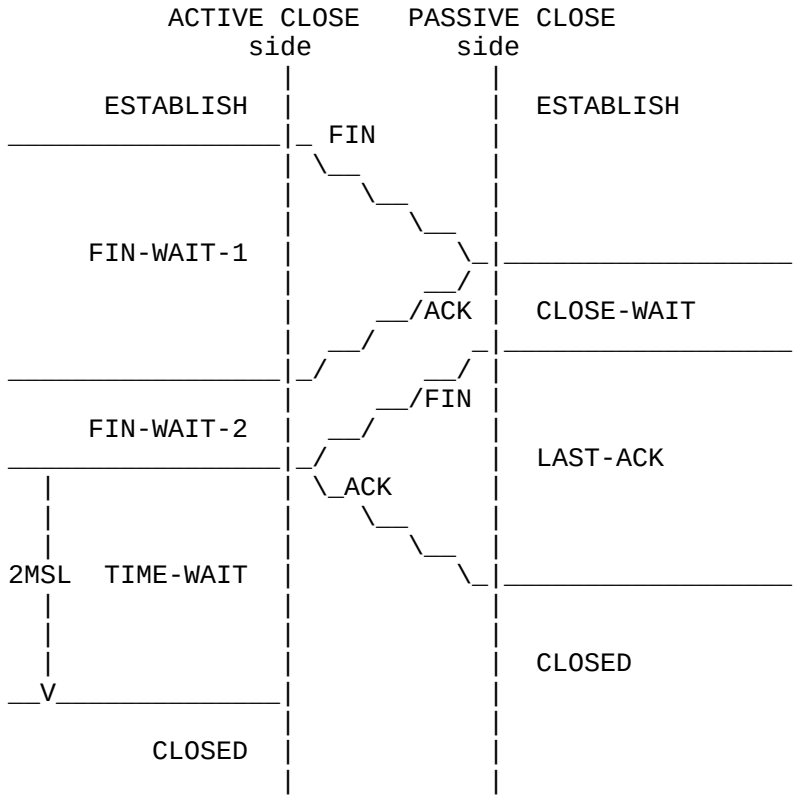

Fig. 1 Current ACTIVE-PASSIVE Close Sequence

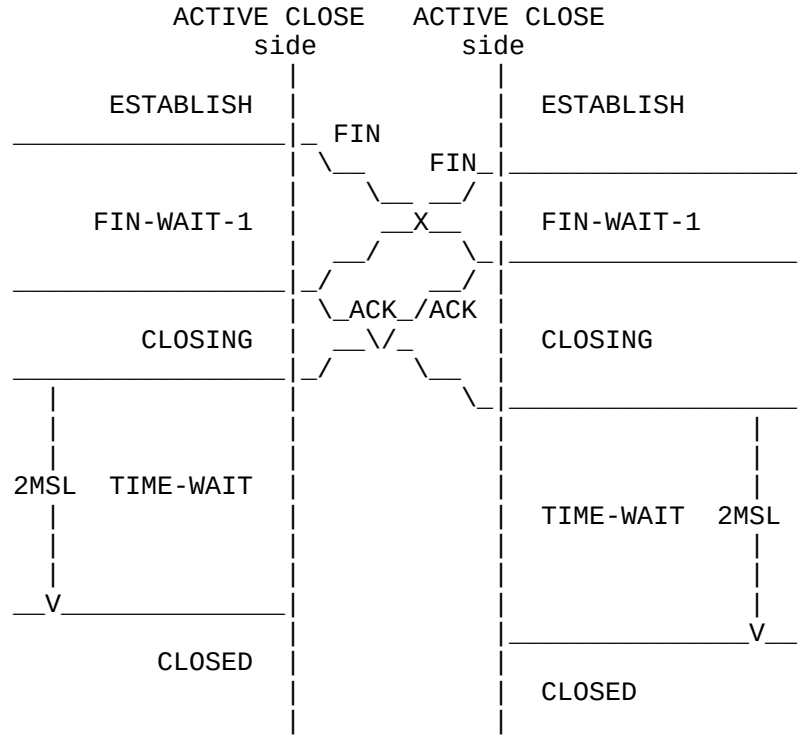

Fig. 2 Current ACTIVE-ACTIVE Close Sequence

Fig. 1 and Fig. 2 show Close Sequence that is defined by current specification [RFC0793]. TCP connections on ACTIVE CLOSE node (that initiates sending FIN) side reach TIME-WAIT as a last state. They stay there 2MSL seconds.

Table 1 Actual 2MSL values used by major OS implementation.

RFC/OS	2MSL value
[RFC0793]	240 sec.
Windows2000	240 sec.
Windows (after Win2K)	120 sec.
Unix/Linux	60 sec.

Table 1 shows actual 2MSL values that are surveyed by authors.

[RFC0793] says "For this specification the MSL is taken to be 2 minutes."

Since 240 sec. ([RFC0793]) is long time, recent major OSes adopt rather shorter time.

However, from the viewpoints of current communication styles that require highly resource recycling, TIME-WAIT time is still too long.

Now, it is almost thought that staying at TIME-WAIT state is waste of time.

3. Why TIME-WAIT state is needed?

Basically, TIME-WAIT state is designed for !fail-safe! purpose.

If it is assumed that packets transferring order is not changed, all of !data! packets from a corresponding node are received when FIN-WAIT-2 state is finished (responding FIN packet is received) and no !data! packets will not be received after that.

At TIME-WAIT state, an ACTIVE CLOSE node waits for a ’resending’ !control! packet FIN only from the corresponding node for the case of the sent ACK (for the FIN) is lost. (No !data! packets are waited for.)

Only when the last sent ACK from the ACTIVE CLOSE node is lost, ’resending’ control packet FIN from the corresponding node is issued.

It is rare case to happen this event at current stable network environment.

Since all data from the corresponding node is received by the ACTIVE CLOSE node, it is less significant issue to wait for ’resending’ FIN packet.

If ’resending’ FIN is NOT waited at ACTIVE CLOSE node and ’resending’ FIN is issued from the corresponding node, significant problem will NOT be happened, only RST packet (to notify receiving unexpected packet) will be issued from the ACTIVE CLOSE node.

4. Design of "Sharp Close" (elimination of TIME-WAIT state)

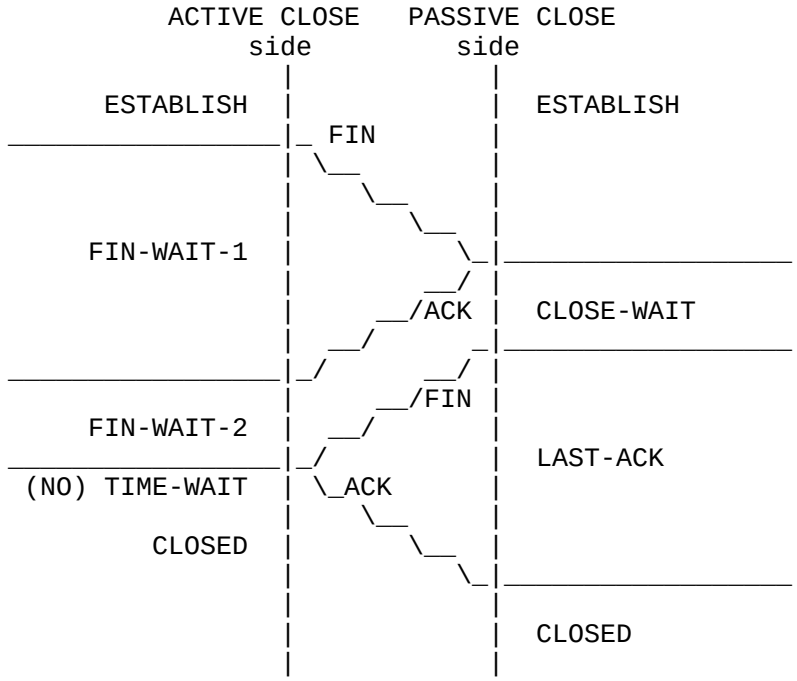

Fig. 3 (Proposed) Sharp ACTIVE-PASSIVE Close Sequence

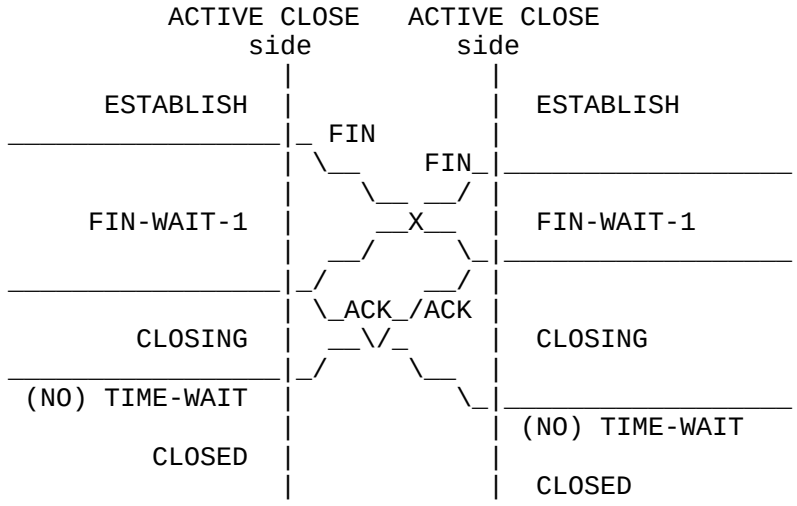

Fig. 4 (Proposed) Sharp ACTIVE-ACTIVE Close Sequence

It is easy to design "Sharp Close" function. "Sharp Close" function is achieved by eliminating or minimizing TIME-WAIT state of TCP connections.

Fig. 3 and Fig. 4. show Close Sequence that is defined by "Sharp Close" function.

5. Eliminate TIME-WAIT state by setsockopt()

Under current implementation, TIME-WAIT (close()) action can be controlled by setsockopt() function.

SO_LINGER option of setsockopt() can eliminate TIME-WAIT state and close connections immediately.

Concrete procedures how to eliminate TIME-WAIT:

Fig. 5 shows struct socket in <sys/socket.h>

```
struct linger {
       int l_onoff;    /* linger active */
       int l_linger;   /* how many seconds to linger for */
};
```

Fig. 5. struct linger

By using the following shown procedures, TIME-WAIT state is eliminated and connections are closed immediately.

1: makes linger active(on)
   l_onoff = on;

2: sets linger time to 0
   l_linger = 0 ;

It is possible to eliminate TIME-WAIT state by these procedures. However, this behavior is "NOT default" operation. In order to utilize this feature, it is necessary to modify huge number of communication applications.

Furthermore, this feature is not implemented on every existing OSes and it is not always possible to eliminate TIME-WAIT state on every OSes.

6. Security Considerations

Goals of the proposed idea ("Sharp Close") are to eliminate or minimize TIME-WAIT state by default on OS kernel level. From functional viewpoints, the same concept to eliminate TIME-WAIT state is already implemented by using LINGER option of setsockopt() function. It is not default operation, however it has already implemented and worked.

So, there are no new Security Consideration issues that should be discussed here.

7. IANA Considerations

This document does not require any resource assignments to IANA.

Acknowledgment

A part of this work is supported by the program: SCOPE (Strategic Information and Communications R&D Promotion Programme) operated by Ministry of Internal Affairs and Communications of JAPAN.

Appendix A. Implementations

Currently, above described "Sharp Close" functions have been implemented and verified under the following OS.

Ubuntu 13.04 (kernel 3.8.13.8)

References

Normative References

[RFC0793] "Transmission Control Protocol", RFC 0793, September 1981

[RFC3513] R. Hinden and S.Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, April 2003

[RFC3493] R. Gilligan, S. Thomson, J. Bound, J. McCann and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC3493, February 2003

Informative References

[RFC3542] W. Stevens, M. Thomas, E. Nordmark and T. Jinmei, "Advanced Sockets Application Program Interface (API) for IPv6", RFC 3542, May 2003

Authors' Addresses

Hiroshi Kitamura
Cyber Security Strategy Division / Cloud System Research Laboratories,
NEC Corporation
7-1, Shiba 5-chome, Minato-ku, Tokyo 108-8001, JAPAN
Phone: +81 3 3798 0563
Email: kitamura@da.jp.nec.com

Shingo Ata
Graduate School of Engineering, Osaka City University
3-3-138, Sugimoto, Sumiyoshi-Ku, Osaka 558-8585, JAPAN
Phone: +81 6 6605 2191
Fax: +81 6 6605 2191
Email: ata@info.eng.osaka-cu.ac.jp

Masayuki Murata
Graduate School of Information Science and Technology, Osaka Univ.
1-5 Yamadaoka, Suita, Osaka 565-0871, JAPAN
Phone: +81 6 6879 4542
Fax: +81 6 6879 4544
Email: murata@ist.osaka-u.ac.jp