Internet Engineering Task Force (IETF)
Request for Comments: 6093
Updates: 793, 1011, 1122
Category: Standards Track
ISSN: 2070-1721

F. Gont
UTN/FRH
A. Yourtchenko
Cisco

January 2011

On the Implementation of the TCP Urgent Mechanism

Abstract

This document analyzes how current TCP implementations process TCP urgent indications and how the behavior of some widely deployed middleboxes affects how end systems process urgent indications. This document updates the relevant specifications such that they accommodate current practice in processing TCP urgent indications, raises awareness about the reliability of TCP urgent indications in the Internet, and recommends against the use of urgent indications (but provides advice to applications that do).

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6093.

Table of Contents

1. Introduction 3
2. Specification of the TCP Urgent Mechanism 3
2.1. Semantics of Urgent Indications 3
2.2. Semantics of the Urgent Pointer 4
2.3. Allowed Length of "Urgent Data" 4
3. Current Implementation Practice of the TCP Urgent Mechanism 5
3.1. Semantics of Urgent Indications 5
3.2. Semantics of the Urgent Pointer 5
3.3. Allowed Length of "Urgent Data" 6
3.4. Interaction of Middleboxes with TCP Urgent Indications 6
4. Updating RFC 793, RFC 1011, and RFC 1122 6
5. Advice to New Applications Employing TCP 7
6. Advice to Applications That Make Use of the Urgent Mechanism 7
7. Security Considerations 7
8. Acknowledgements 8
9. References 8
9.1. Normative References 8
9.2. Informative References 8
Appendix A. Survey of the Processing of TCP Urgent Indications by Some Popular TCP Implementations 10
A.1. FreeBSD 10
A.2. Linux 10
A.3. NetBSD 10
A.4. OpenBSD 11
A.5. Cisco IOS software 11
A.6. Microsoft Windows 2000, Service Pack 4 11
A.7. Microsoft Windows 2008 11
A.8. Microsoft Windows 95 11

1. Introduction

This document analyzes how some current TCP implementations process TCP urgent indications, and how the behavior of some widely deployed middleboxes affects the processing of urgent indications by hosts. This document updates RFC 793 [RFC0793], RFC 1011 [RFC1011], and RFC 1122 [RFC1122] such that they accommodate current practice in processing TCP urgent indications. It also provides advice to applications using the urgent mechanism and raises awareness about the reliability of TCP urgent indications in the current Internet.

Given the above issues and potential interoperability issues with respect to the currently common default mode operation, it is strongly recommended that applications do not employ urgent indications. Nevertheless, urgent indications are still retained as a mandatory part of the TCP protocol to support the few legacy applications that employ them. However, it is expected that even these applications will have difficulties in environments with middleboxes.

Section 2 describes what the current IETF specifications state with respect to TCP urgent indications. Section 3 describes how current TCP implementations actually process TCP urgent indications. Section 4 updates RFC 793 [RFC0793], RFC 1011 [RFC1011], and RFC 1122 [RFC1122], such that they accommodate current practice in processing TCP urgent indications. Section 5 provides advice to new applications employing TCP, with respect to the TCP urgent mechanism. Section 6 provides advice to existing applications that use or rely on the TCP urgent mechanism.

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 RFC 2119 [RFC2119].

2. Specification of the TCP Urgent Mechanism

2.1. Semantics of Urgent Indications

TCP implements an "urgent mechanism" that allows the sending user to stimulate the receiving user to accept some "urgent data" and that permits the receiving TCP to indicate to the receiving user when all the currently known "urgent data" have been read.

The TCP urgent mechanism permits a point in the data stream to be designated as the end of urgent information. Whenever this point is in advance of the receive sequence number (RCV.NXT) at the receiving TCP, that TCP must tell the user to go into "urgent mode"; when the receive sequence number catches up to the urgent pointer, the TCP

must tell user to go into "normal mode" [RFC0793]. This means, for example, that data that was received as "normal data" might become "urgent data" if an urgent indication is received in some successive TCP segment before that data is consumed by the TCP user.

The URG control flag indicates that the "Urgent Pointer" field is meaningful and must be added to the segment sequence number to yield the urgent pointer. The absence of this flag indicates that there is no "urgent data" outstanding [RFC0793].

The TCP urgent mechanism is NOT a mechanism for sending "out-of-band" data: the so-called "urgent data" should be delivered "in-line" to the TCP user.

2.2. Semantics of the Urgent Pointer

There is some ambiguity in RFC 793 [RFC0793] with respect to the semantics of the Urgent Pointer. Section 3.1 (page 17) of RFC 793 [RFC0793] states that the Urgent Pointer "communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment. The urgent pointer points to the sequence number of the octet following the urgent data. This field is only be interpreted in segments with the URG control bit set" (sic). However, Section 3.9 (page 56) of RFC 793 [RFC0793] states, when describing the processing of the SEND call in the ESTABLISHED and CLOSE-WAIT states, that "If the urgent flag is set, then SND.UP <- SND.NXT-1 and set the urgent pointer in the outgoing segments".

RFC 1011 [RFC1011] clarified this ambiguity in RFC 793 stating that "Page 17 is wrong. The urgent pointer points to the last octet of urgent data (not to the first octet of non-urgent data)". RFC 1122 [RFC1122] formally updated RFC 793 by stating, in Section 4.2.2.4 (page 84), that "the urgent pointer points to the sequence number of the LAST octet (not LAST+1) in a sequence of urgent data".

2.3. Allowed Length of "Urgent Data"

RFC 793 [RFC0793] allows TCP peers to send "urgent data" of any length, as the TCP urgent mechanism simply provides a pointer to an interesting point in the data stream. In this respect, Section 4.2.2.4 (page 84) of RFC 1122 [RFC1122] explicitly states that "A TCP MUST support a sequence of urgent data of any length".

3. Current Implementation Practice of the TCP Urgent Mechanism

3.1. Semantics of Urgent Indications

As discussed in Section 2, the TCP urgent mechanism simply permits a point in the data stream to be designated as the end of urgent information but does NOT provide a mechanism for sending "out-of-band" data.

Unfortunately, virtually all TCP implementations process TCP urgent indications differently. By default, the last byte of "urgent data" is delivered "out of band" to the application. That is, it is not delivered as part of the normal data stream [UNPv1]. For example, the "out-of-band" byte is read by an application when a recv(2) system call with the MSG_OOB flag set is issued.

Most implementations provide a socket option (SO_OOBINLINE) that allows an application to override the (broken) default processing of urgent indications, so that "urgent data" is delivered "in line" to the application, thus providing the semantics intended by the IETF specifications.

3.2. Semantics of the Urgent Pointer

All the popular implementations that the authors of this document have been able to test interpret the semantics of the TCP Urgent Pointer as specified in Section 3.1 of RFC 793. This means that even when RFC 1122 formally updated RFC 793 to clarify the ambiguity in the semantics of the Urgent Pointer, this clarification was never reflected in actual implementations (i.e., virtually all implementations default to the semantics of the urgent pointer specified in Section 3.1 of RFC 793).

Some operating systems provide a system-wide toggle to override this behavior and interpret the semantics of the Urgent Pointer as clarified in RFC 1122. However, this system-wide toggle has been found to be inconsistent. For example, Linux provides the sysctl "tcp_stdurg" (i.e., net.ipv4.tcp_stdurg) that, when set, supposedly changes the system behavior to interpret the semantics of the TCP Urgent Pointer as specified in RFC 1122. However, this sysctl changes the semantics of the Urgent Pointer only for incoming segments (i.e., not for outgoing segments). This means that if this sysctl is set, an application might be unable to interoperate with itself if both the TCP sender and the TCP receiver are running on the same host.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.