

TCP Maintenance and Minor
Extensions (tcpm)
Internet-Draft
Intended status: Standards Track
Expires: May 1, 2009

F. Gont
UTN/FRH
A. Yourtchenko
Cisco
October 28, 2008

On the implementation of TCP urgent data
draft-gont-tcpm-urgent-data-00.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of 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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on May 1, 2009.

Abstract

This document analyzes the current practices for handling TCP urgent data in the current Internet. It describes how popular TCP implementations process urgent data, and also describes how a number of middle-boxes affect how urgent data is processed by end systems. Additionally, it includes a survey of the processing of urgent data by popular TCP implementations. This document updates the relevant specifications such that they accommodate current practice in processing TCP urgent data.

Table of Contents

- 1. Introduction 3
- 2. Specification of TCP urgent data 3
 - 2.1. Semantics of the Urgent Pointer 4
 - 2.2. Allowed length of urgent data 4
- 3. Current implementation practice of TCP urgent data 4
 - 3.1. Semantics of the Urgent Pointer 4
 - 3.2. Allowed length of urgent data 4
- 4. Updating the specifications to reflect implementations behavior 5
- 5. Interaction of middle-boxes with urgent data 5
- 6. Security Considerations 5
- 7. IANA Considerations 6
- 8. Acknowledgements 6
- 9. References 6
 - 9.1. Normative References 6
 - 9.2. Informative References 6
- Appendix A. Survey of the processing of urgent data by some popular implementations 7
 - A.1. FreeBSD 7
 - A.2. Linux 7
 - A.3. NetBSD 8
 - A.4. OpenBSD 8
 - A.5. Cisco IOS, versions 12.2(18)SXF7, 12.4(15)T7 8
 - A.6. Microsoft Windows 2000, Service Pack 4 8
 - A.7. Microsoft Windows 2008 8
 - A.8. Microsoft Windows 95 9
- Authors' Addresses 9
- Intellectual Property and Copyright Statements 10

1. Introduction

TCP incorporates a "urgent mechanism" that allows the sending user to stimulate the receiving user to accept some urgent data and to permit the receiving TCP to indicate to the receiving user when all the currently known urgent data has been received by the user. This 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]

The URG control flag indicates that the urgent 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 "urgent mechanism" was originally specified in RFC 793 [RFC0793]. Later, RFC 1122 [RFC1122] corrected some errors found in that specification. However, these "corrections" never made into actual implementations, and thus the current specifications do not reflect the way in which virtually all TCP implementations process urgent data.

This document analyzes the current practices for handling TCP urgent data in the current Internet. It describes how popular TCP implementations process urgent data, and also describes how a number of middle-boxes affect how urgent data is processed by end systems. Additionally, it includes a survey of the processing of urgent data by popular TCP implementations. This document updates the relevant specifications such that they accommodate current practice in processing TCP urgent data.

Section 2 describes what the current IETF specifications state with respect to TCP urgent data. Section 3 describes how current TCP implementations actually process TCP urgent data. Section 4 updates RFC 793 [RFC0793] and RFC 1122 [RFC1122] such that they accommodate current practice in processing TCP urgent data.

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 TCP urgent data

2.1. Semantics of the Urgent Pointer

Section 3.1 (page 17) of RFC 793 [RFC0793] originally stated 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."

Section 4.2.2.4 (page 84) of RFC 1122 [RFC1122] "corrects" RFC793 stating that "the urgent pointer points to the sequence number of the LAST octet (not LAST+1) in a sequence of urgent data."

2.2. Allowed length of urgent data

RFC 793 [RFC0793] allows TCP peers to send urgent data of any length. Section 4.2.2.4 (page 84) of RFC 1122 explicitly states that "A TCP MUST support a sequence of urgent data of any length".

3. Current implementation practice of TCP urgent data

3.1. 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 stated in Section 3.1 of RFC 793. This means that while RFC 1122 officially "corrected" RFC 793, this change has never reflected to become a default behavior in popular TCP implementations.

Some operating systems provide a system-wide toggle to override this behavior, and interpret the semantics of the Urgent Pointer as specified in RFC 1122. However, this system-wide toggle has been found to be inconsistent. For example, Linux provides a the `sysctl` "tcp_stdurg" (e.g., `net.ipv4.tcp_stdurg`) that, when set, supposedly changes the system behavior to interpret the semantics of the TCP Urgent Pointer as described in RFC 1122. However, this `sysctl` only changes the semantics of the Urgent Pointer of incoming segments, and not of all segments. This means that if this `sysctl` is set, the host might be unable to interoperate with itself.

3.2. Allowed length of urgent data

While Section 4.2.2.4 (page 84) of RFC 1122 explicitly states that "A TCP MUST support a sequence of urgent data of any length", in practice a lot of implementations support only a single byte of urgent data. [UNPv1]

4. Updating the specifications to reflect implementations behavior

We believe the relevant specifications should be updated to reflect what is the de-facto standard for urgent data: interpret the semantics of the Urgent Pointer as described in RFC 793, and allow only a single byte of urgent data. This has been the behavior implemented by most popular TCP implementations, including the one Linux and in BSD-derived systems.

A TCP MUST support a sequence of a single byte urgent data, and MAY support a sequence of urgent data of any length.

The TCP Urgent Pointer MUST point to the byte following the last byte of urgent data.

(Future revisions of this document will include a list of the specific text in RFC 793 and RFC 1122 that should be "patched" to produce the proposed specification update).

5. Interaction of middle-boxes with urgent data

As a result of the publication [phrack] Network Intrusion Detection (NIDs) evasion techniques based on urgent data, some middle-boxes modify the TCP data stream such that urgent data is put "in band", that is, they are accessible by the read(2) or recv(2) calls without the MSG_OOB flag. This should probably discourage applications to depend on urgent data for their current operation, as urgent data may not be not reliable in the current Internet. Examples of such middle-boxes are Cisco PIX firewall [Cisco-PIX].

6. Security Considerations

Given that there are two different interpretations (RFC793 vs. RFC 1122) of semantics of the Urgent Pointer in current implementations, and that either middle-boxes (such as packet scrubbers) or the end-systems themselves could cause the urgent data to be processed "in band", there exists ambiguity in how TCP urgent data sent by a TCP will be processed by the intended recipient. This might make it difficult for a Network Intrusion Detection System (NIDS) to track the application-layer data transferred to the destination system, and thus lead to false negatives or false positives in the NIDS. [CPNI-TCP].

Probably the best way to avoid the security implications of TCP urgent data is to avoid having application protocols depend on the use of TCP urgent data altogether. Packet scrubbers could probably

be configured to clear the URG bit, and set the Urgent Pointer to zero. This would basically cause the urgent data to be put "in band". However, this might cause interoperability problems or undesired behavior in the applications running on top of TCP.

7. IANA Considerations

This document has no actions for IANA.

8. Acknowledgements

The authors of this document would like to thank Carlos Pignataro, Anantha Ramaiah and Dan Wing for providing valuable feedback on an earlier version of this document.

9. References

9.1. Normative References

- [RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.
- [RFC1122] Braden, R., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

- [CPNI-TCP] CPNI, "Security Assessment of the Transmission Control Protocol (TCP)", (to be published) .
- [Cisco-PIX] Cisco PIX, "<http://www.cisco.com/en/US/docs/security/asa/asa70/command/reference/tz.html#wp1288756>".
- [FreeBSD] The FreeBSD project, "<http://www.freebsd.org>".
- [Linux] The Linux Project, "<http://www.kernel.org>".
- [NetBSD] The NetBSD project, "<http://www.netbsd.org>".
- [OpenBSD] The OpenBSD project, "<http://www.openbsd.org>".

- [UNPv1] Stevens, W., "UNIX Network Programming, Volume 1. Networking APIs: Sockets and XTI", Prentice Hall PTR , 1997.
- [Windows2000] Microsoft Windows 2000, "[http://technet.microsoft.com/en-us/library/bb726981\(printer\).aspx](http://technet.microsoft.com/en-us/library/bb726981(printer).aspx)".
- [Windows95] Microsoft Windows 95, "<ftp://ftp.demon.co.uk/pub/mirrors/win95netfaq/faq-c.html>".
- [phrack] Ko, Y., Ko, S., and M. Ko, "NIDS Evasion Method named "SeolMa"", Phrack Magazine, Volume 0x0b, Issue 0x39, Phile #0x03 of 0x12 <http://www.phrack.org/issues.html?issue=57&id=3#article>, 2001.

Appendix A. Survey of the processing of urgent data by some popular implementations

A.1. FreeBSD

FreeBSD [FreeBSD] interprets the semantics of the urgent pointer as specified in RFC 793. It does not provide any `sysctl` to override this behavior. However, it provides the `SO_OOBINLINE` that when set causes TCP urgent data to be put "in band". That is, it will be accessible by the `read(2)` or `recv(2)` calls without the `MSG_OOB` flag.

FreeBSD supports only one byte of urgent data. That is, only the byte preceding the Urgent Pointer is considered as "urgent data".

A.2. Linux

Linux [Linux] interprets the semantics of the urgent pointer as specified in RFC 793. It provides the `net.ipv4.tcp_stdurg` `sysctl` to override this behavior to interpret the Urgent Pointer as specified by RFC 1122. However, this `sysctl` only affects the processing of incoming segments (the Urgent Pointer in outgoing segments will still be set as specified in RFC 793).

Linux supports only one byte of urgent data. That is, only the byte preceding the Urgent Pointer is considered as "urgent data".

A.3. NetBSD

NetBSD [NetBSD] interprets the semantics of the urgent pointer as specified in RFC 793. It does not provide any `sysctl` to override this behavior. However, it provides the `SO_OOBINLINE` that when set causes TCP urgent data to be put "in band". That is, it will be accessible by the `read(2)` or `recv(2)` calls without the `MSG_OOB` flag.

NetBSD supports only one byte of urgent data. That is, only the byte preceding the Urgent Pointer is considered as "urgent data".

A.4. OpenBSD

OpenBSD [OpenBSD] interprets the semantics of the urgent pointer as specified in RFC 793. It does not provide any `sysctl` to override this behavior. However, it provides the `SO_OOBINLINE` that when set causes TCP urgent data to be put "in band". That is, it will be accessible by the `read(2)` or `recv(2)` calls without the `MSG_OOB` flag.

OpenBSD supports only one byte of urgent data. That is, only the byte preceding the Urgent Pointer is considered as "urgent data".

A.5. Cisco IOS, versions 12.2(18)SXF7, 12.4(15)T7

Cisco IOS, versions 12.2(18)SXF7, 12.4(15)T7 interpret the semantics of the urgent pointer as specified in RFC 793. However, tests performed with an HTTP server running on Cisco IOS version 12.2(18)SXF7 and 12.4(15)T7 suggest that urgent data is processed "in band". That is, they are accessible together with "normal" data. The TCP debugs on the Cisco IOS device do explicitly mention the presence of urgent data, and thus we infer that the behavior is different depending on the application.

A.6. Microsoft Windows 2000, Service Pack 4

Microsoft Windows 2000 [Windows2000] interprets the semantics of the urgent pointer as specified in RFC 793. It provides the `TcpUserRFC1122UrgentPointer` system-wide variable to override this behavior to interpret the Urgent Pointer as specified by RFC 1122. However, the tests performed with the sample server application compiled using the `cygwin` environment, has shown that the default behavior is to return the urgent data "in band".

A.7. Microsoft Windows 2008

Microsoft Windows 2008 interprets the semantics of the urgent pointer as specified in RFC 793.

A.8. Microsoft Windows 95

Microsoft Windows 95 interprets the semantics of the urgent pointer as specified in RFC 793. It provides the BSDUrgent system-wide variable to override this behavior to interpret the Urgent Pointer as specified by RFC 1122. Windows 95 supports only one byte of urgent data. That is, only the byte preceding the Urgent Pointer is considered as "urgent data". [Windows95]

Authors' Addresses

Fernando Gont
Universidad Tecnologica Nacional / Facultad Regional Haedo
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires 1706
Argentina

Phone: +54 11 4650 8472
Email: fernando@gont.com.ar
URI: <http://www.gont.com.ar>

Andrew Yourtchenko
Cisco
De Kleetlaan, 7
Diegem B-1831
Belgium

Phone: +32 2 704 5494
Email: ayourtch@cisco.com

Full Copyright Statement

Copyright (C) The IETF Trust (2008).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

