On the implementation of TCP urgent data (draft-gont-tcpm-urgent-data)

Fernando Gont & A. Yourtchenko

UTN HAEDO
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL HAEDO

CISCO

73rd IETF meeting, November 16-21, 2008
Minneapolis, MN, USA

Problem statement (I) (where does the UP point to?)

- There was some ambiguity in RFC 793 with respect to the semantics of the TCP urgent pointer
  - UP points to the byte following the last byte of urgent data?
  - UP points to the last byte of urgent data?
- RFC 1122 clarified this ambiguity
  - “the UP points to the last byte of urgent data”
- However, virtually every implementation interprets the semantics of the UP as:
  - “the UP points to the byte following the last byte of urgent data”
- Result:
  - there’s a difference between what the specs state and what’s actually implemented

Problem statement (II) (OOB vs- in-line)

- RFC 793 explains that the UP simply represents a mark in the data stream where urgent data ends.
- Generally (but not actually specified in the RFCs), applications would skip (discard) all those data before the urgent mark. But all data would be in-band…
- However, virtually all stacks implement urgent data as follows:
  - The UP points to the byte following the last byte of urgent data
  - There can be only a single byte of urgent data at any time
- By default, this “single byte of urgent data” is typically delivered out-of-band, by means of the recv(2) call with the MSG_OOB flag
- Some implementations (e.g., BSD-derived) have a single byte for buffering urgent data. If you receive to urgent indications, the first byte is lot. Other implementations (Microsoft?) queue OOB data!

Problem statement (III) (let's make things worse)

- Some middle-boxes (e.g. Cisco PIX) clear the URG bit and set the Urgent Pointer to zero **by default**.
- This means that any application that currently depends on TCP urgent data may break.

What should we do about it?

- There are two different areas of work:
  - UP semantics
  - OOB vs. in-line processing of urgent data

What should we do about the UP? (I)

- Possible ways forward for UP semantics:
  - **Do nothing**: this would make the specs irrelevant with respect to urgent data
  - **Aim at having stacks implement the RFC 1122 semantics**: This would break any app that is currently using urgent data
  - **Update RFC 1122 to accomodate what implementations have been doing**: this would make the specs with what real implementations do

What should we do about the UP? (II)

- Let's be pragmatic:
  - However, we are in a situation in which the specifications and real implementations differ
  - We'd like to do RFC 1122, but if we tried to push the RFC 1122 semantics at this point in time, we'd probably break any application making use of urgent data.
  - So the question we should proably ask ourselves at this point is: as long as all implementations are consistent with how they send and how they receive urgent data, does it actually matter whether the UP points to the last byte of urgent data vs. the byte following the last byte of urgent data?
  - If the answer to this question is "No", then the way to go would be to update RFC 1122 in this respect to change the semantics of the UP.

What should we do about in-line vs. OOB? (I

- Possible ways forward for OOB vs. in-line processing of urgent data
  - **Do nothing**: apps would continue working, but with a broken semantics for the TCP urgent data
  - **Recommend apps to set the SO_OOBINLINE, so that urgent data is processed in-line**: apps would continue working, but they could migrate to the correct semantics of TCP urgent data.
  - **Deprecate urgent data**: might make sense, as some middle-boxes already break urgent data, but….

Moving forward

- We're planning to publish a revision (-01) of the urgent data draft that incorporates the feedback we got mainly from David Borman on-list, and a really fruitful discussion (today!) with David Borman and Joe Touch
- We believe that TCPM WG should do something about urgent data
- **Should this document be adopted as a TCPM WG item?**