draft-heffner-frag-harmful-02
IETF 66, July 2006

Fragmentation Considered Very Harmful

John Heffner <jheffner@psc.edu>
Matt Mathis <mathis@psc.edu>
Ben Chandler <bchandle@psc.edu>

Overview

- Draft history and status
- Review of the problem
- Some possible workarounds (and why they’re not included in the draft)

History and status

- Draft title is a reference to Kent/Mogul SIGCOMM ‘87 “Fragmentation Considered Harmful”
- We document additional problem which can result in corrupted datagrams (“very” harmful)
- Problem has long been in the lore, but not well known or published. It’s time to fix that.
- Wrote the draft a couple years ago, didn’t know exactly where it belonged. Lars Eggert currently shepherding through the AD-sponsored draft process.
- Received and incorporated some feedback from tsvwg and int-area lists, no major items.
- We consider it mostly done.

Mis-association

- IPv4 fragments are associated with each other by a 16-bit identification (IP ID) field.
- If we send 2^{16} datagrams in less than the timeout for a fragment reassembly buffer, we wrap the IP ID field and can *mis-associate* fragments. Some call these “frankengrams.” :-)
- With common hardware (100 Mbps) and most OS default settings, this easily happens today.

Cyclical mis-association

- If you lose the first fragment, the rest of the datagram sits in the reassembly buffer.
- When the IP ID is wrapped, the first new fragment will be mis-associated with the old fragments. The rest of the new fragments will sit in the reassembly buffer until the next IP ID wrap, forming a self-propogating cycle.
- You can have a number of concurrent cycles.

Effects

- Packets get dropped when the checksum test fails.
- With such high corruption rate, 16-bit checksum isn't strong enough. Streams get corrupted.
  - UDP checksum is especially weak, likely to have "hot spots"
- If you're running UDP without a checksum, you've got trouble!

Who's affected

- Protocols using fragmentation
  - Doing MTU discovery eliminates the problem.
- High rate *per protocol* (not per flow) per address pair
  - NAT makes the situation worse (surprise)
- Low rate (DNS) is probably okay
- Fixed rate (streaming media) - unclear

Experimental observations

- Moved 10 TB of random data with a UDP bulk transport tool (Reliable Blast UDP) with 100 Mbps NIC, Linux box
- Induced intermittent loss with small cross-traffic flows
- Observed 8847668 checksum errors, 121 corruptions

Work-arounds (1)

- Adjust fragment boundaries on wraps of the IP ID
  - No matter what, you always end up having some wraps that overlap
  - Practically, it's expensive and difficult to coordinate this
  - Doesn't work if fragmentation occurs in the network

Work-arounds (2)

- Shorten the timeout
  - Some peers may be too fast while others simultaneously too slow.
  - Doesn’t work with classical global timeout.

Work-arounds (3)

- Per-peer adaptive timeout
  - Best way is to use packet count rather than actual timer
  - Recently implemented in Linux
  - Mostly works, little reason not to do it
    - Still some issues, for example NAT, and possibly multi-path
  - Does require per-peer state
  - Main difficulty from a standards perspective: work-around implemented on receiver, but sender has no way of knowing if it's safe or not.

Informational only

- This draft only documents a known problem, and is strictly informational.
- We didn't want to prescribe a fix because:
  - Each solution has some known problems
  - Under the cases where the problem occurs, it is usually best to avoid fragmentation anyway (for reasons stated in the Kent/Mogul paper)

IPv6

- Uses a 32-bit ID field (instead of 16 bits)
- So, IPv6 is safe — for now. :-) We only have a few orders of magnitude to go.