Detecting TCP regressions with tcpdiff

BSDCan 2009

Mike Silbersack
silby@silby.com

http://www.silby.com/bsdcan09/

Presentation Overview

- What is tcpdiff
- The tcpdiff test apparatus
- How tcpdiff analysis works
- 6 kernel pairs tested
- Systems that can not be tested
- Future Work

What is tcpdiff?

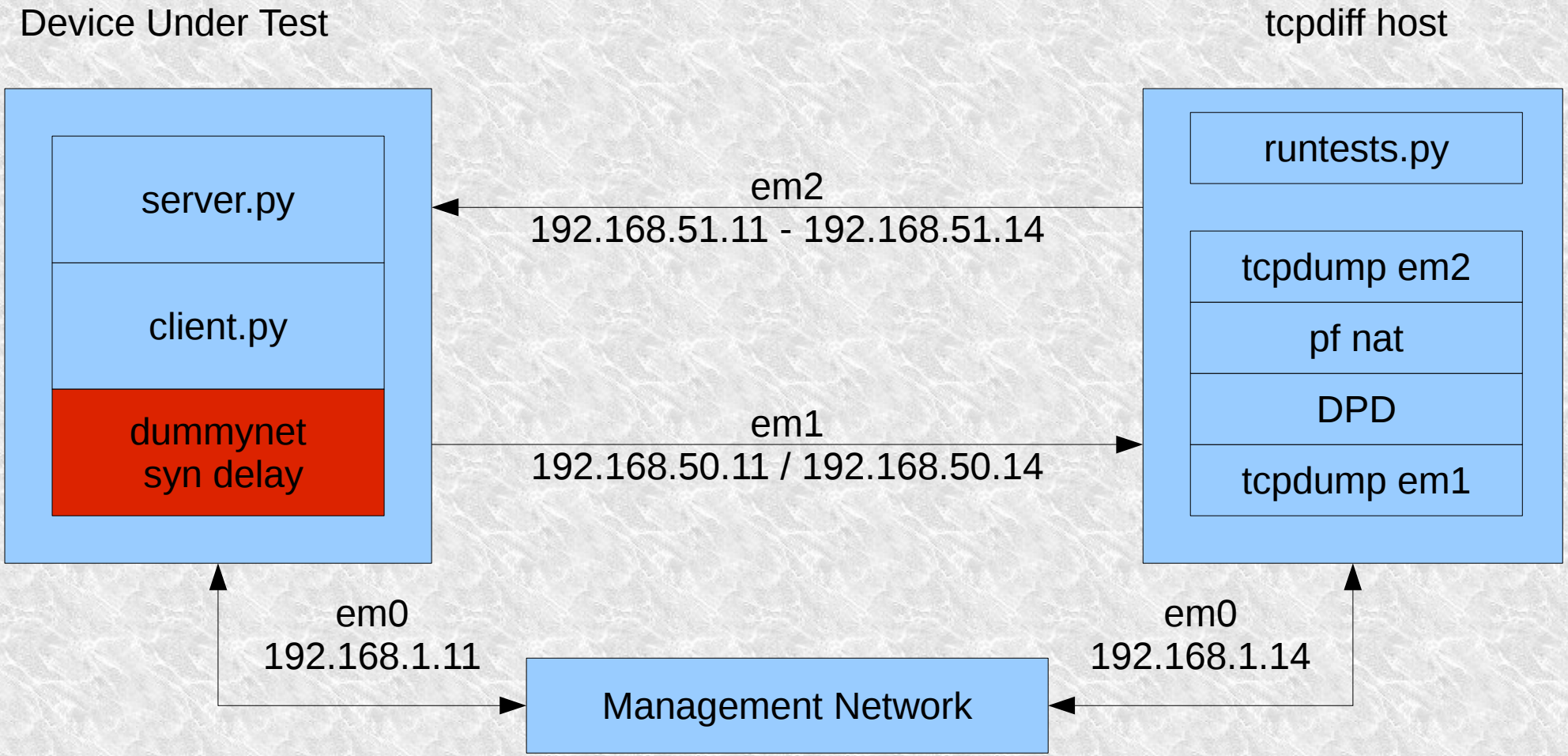
tcpdiff is designed to automatically detect differences in TCP behavior between different versions of an operating system and display those differences in an easy to understand format.

tcpdiff looks only at on the wire behavior of the network stack. The kernel under test requires no modifications; all faults are injected on the wire.

What is tcpdiff not?

tcpdiff is not designed to know whether a TCP stack operates correctly or not. All it can do is compare two TCP stacks. The value judgment of whether a certain change between version X and Y of a TCP stack is good or bad is left to human eyes.

The tcpdiff test apparatus



Tools used in the apparatus

- Vmware ESXi
  - Physical hardware would be better, of course.
- DPD (Deterministic Packet Discard)
  - Lawrence Stewart's modification to dummynet that allows specific packets to be dropped
  - http://caia.swin.edu.au/urp/newtcp/tools.html
- OpenBSD's pf
  - Performs the NAT magic to send packets back to the Device Under Test.

Tools used in the apparatus, continued

- Dummynet
  - Used to delay the initial SYN packet so that the SYN and SYN-ACK retransmission timers are not synchronized. (Both ends have a 1 second retransmission timer.)
  - This should be resident on the tcpdiff host, but stacking multiple dummynet pipes caused buggy results.
- Tcpdump
  - This is run on both interfaces so that the correct behavior of DPD can be verified.

Tools used in the apparatus, continued

- server.py
  - Accepts a connection, sends 64K of data, closes the connection.
- client.py
  - Initiates a connection, does a MSG_WAITALL recv to read all data at once, sleeps one second, closes the connection. Waits for the FIN_WAIT/etc socket to clear before exiting.
    - The MSG_WAITALL read is important as it prevents window updates from showing up on the wire.

Tools used in the apparatus, continued

- runtests.py
  - Set up various sysctls
  - Start server.py
  - Loop for each drop pattern
    - Reset the hostcache
    - Reset the DPD rules
    - Reset tcpdumps
    - Run client.py
  - Disable delayed acks and run a subset of the drop pattern

Drop patterns

- 64K of application data sent by server
  - 27 packets, 1416 bytes of packet data from client to server
  - 49 packets, 68096 bytes of packet data from server to client
- Drop patterns tested:
  - A mixture of single and double packet drops were tested, focusing on the start and end of each connection.
  - A few tests with massive packet loss in the middle of the connection
  - 81 packet drop patterns tested with delayed acks, 8 packet drop patterns tested without delayed acks.

How tcpdiff analysis works

- Stage 1: process.py / normalize.py
  - Normalize output
    - Normalize IP/port into “client > server” and “server > client”
    - Zero out TCP timestamps
    - Round off inter-packet time differences
    - Normalize ISNs
  - Split output into directional flows
  - Add notation as to which packets were dropped by DPD.

How tcpdiff analysis works, continued

- Stage 2: compare.py
  - Runs diff on two processed directories.

Test Results: 09/12/08 to 4/26/09

- SVN rev 182970 to 191512
- Show 182970to191512.txt

Test Explanation: 9/12/08 to 4/26/09

- There are no detectable TCP changes other than noise.
- An ideal automated nightly run would be able to say “no change”

Test Results: 3/13/08

- SVN rev 177138 to 177139
- Show 177139to177138.txt

Test Explanation: 3/13/08

- Note: This could not have been detected – tcpdump did not display the padding differences in March of '08!

Revision 177139:

Author: bz
Date: Thu Mar 13 10:09:12 2008 UTC (13 months, 3 weeks ago)
Log Message: MFC rev. 1.146 tcp_output.c

Padding after EOL option must be zeros according to RFC793 but
the NOPs used are 0x01.
While we could simply pad with EOLs (which are 0x00), rather use an
explicit 0x00 constant there to not confuse poeple with 'EOL padding'.
Put in a comment saying just that.

Early MFC requested by: silby
because of more people reporting problems on net@

Test Results: 12/5/07 to 12/3/07

- Show 20071205to20071203.txt

Test Explanation: 12/5/07 to 12/3/07

- SACK negotiation was not working correctly
  - Break
    - tcp_syncache.c 1.105 Mar 15, 2007
  - Fix
    - tcp_syncache.c 1.136 Dec 4, 2007

Test Results: 9/8/08 to 9/6/07

- Show 20070908to20070906.txt

Test Explanation: 9/8/08 to 9/6/07

- Tcp timer merge/unmerge – sockets could emit packets after shutdown
  - Break – Apr 11, 2007
    - tcp_timer.c 1.90
  - Fix – Sep 7, 2007
    - tcp_timer.c 1.96
- What improvement needs to be made to tcpdiff to detect this?
  - A program that holds a socket open after doing a shutdown would be necessary.

Test Results: 8/1/07 to 7/30/07

- Show 20070801to20070730.txt

Test Explanation: 8/1/07 to 7/30/07

- FreeBSD 6-current – kern.hz 100 -> 1000
  - Break - ?
  - Fix – tcp_timer.h 1.37 Jul 31, 2007
- net.inet.tcp.rexmit_min was being initialized to 3ms instead of 30ms; until tcp_timer.h was fixed it had been a constant that was not dynamically adjusted to hz.
- The fast timeout behavior is only evident when delayed acks are disabled in the test environment.

Test Results: 7/29/07 to 7/27/07

- Show 20070729to20070727.txt

Test Explanation: 7/29/07 to 7/27/07

- FreeBSD 7-current – SYNcache retransmission was not working.
  - Break
    - tcp_syncache.c 1.87 Jun 17, 2006
  - Fix
    - tcp_syncache.c 1.126/1.127 Jul 28, 2007

Revisions I'd like to detect

Revision 182970:
Author: bz
Date: Fri Sep 12 19:28:57 2008 UTC (7 months, 3 weeks ago)
Log Message:
MFC: svn r182885, cvs rev. 1.381 tcp_input.c

Work around an integer division resulting in 0 and thus the
congestion window not being incremented, if cwnd > maxseg^2.
As suggested in RFC2581 increment the cwnd by 1 in this case.

See http://caia.swin.edu.au/reports/080829A/CAIA-TR-080829A.pdf
for more details.

Submitted by: Alana Huebner, Lawrence Stewart,
Grenville Armitage (caia.swin.edu.au)

More revisions I'd like to detect

Revision 181339
Author: jhb
Date: Tue Aug 5 22:08:04 2008 UTC (9 months ago)
Log Message:
MFC: Fix a check in the SYN cache expansion to accept packets that arrive in the receive window instead of just on the left edge of the receive window.

Revision 181340
Author: jhb
Date: Tue Aug 5 22:22:47 2008 UTC (9 months ago)
Log Message:
MFC: Change incorrect stale cookie detection in syncookie_lookup() that prematurely declared a cookie as expired.

Systems that can not be tested

- FreeBSD < 4.0.
  - Delayed acks in these versions of FreeBSD are run from a global timer, and as such are unpredictable.
  - FreeBSD 4+ use per-connection timers; delayed acks are predictable
- FreeBSD < 7.0.
  - The socket copyin routine hands off data from userspace to socket buffers in mbuf cluster (2k) sized chunks, leading to a condition where less than mtu sized packets are sent if the tcp output routine gets ahead.

A run with FreeBSD 6.3

IP server > client: . ack 1 win 64074 <nop,nop,timestamp 0 0>
IP server > client: . 1:1449(1448) ack 1 win 64074 <nop,nop,timestamp 0 0>
IP server > client: . 1449:2897(1448) ack 1 win 64074 <nop,nop,timestamp 0 0>
IP server > client: P 2897:4097(1200) ack 1 win 64074 <nop,nop,timestamp 0 0>
IP server > client: . 4097:5545(1448) ack 1 win 64074 <nop,nop,timestamp 0 0>
IP server > client: P 5545:6145(600) ack 1 win 64074 <nop,nop,timestamp 0 0>
IP server > client: . 6145:7593(1448) ack 1 win 64074 <nop,nop,timestamp 0 0>
IP server > client: P 7593:8193(600) ack 1 win 64074 <nop,nop,timestamp 0 0>

The pattern of full vs less than full sized frames differs nearly each time the test is run.

Future Work

- Run tcpdiff nightly!
  - I have kernel builds running, but no testing or reporting
- Move the SYN delay off of the device under test and on to the tcpdiff host.
  - This requires tracking down and fixing a bug in dummynet
- Normalize RSTs
- Improve time difference comparison to be more relaxed

Future Work continued

- Add more test cases
  - Client connects, sends data to server.
  - Client connects to echo server, sends strings at .1 second intervals. Disconnects once final string has been echoed.
  - Http fetch of a file
- Add more drop patterns
- Determine which other OSes can be tested

Future Work continued

- Fix the processing of the 20070727 to 20070729 comparison
- Make the report friendlier (colorful HTML diff in the style of cvsweb, perhaps?)
- Try creating a three VM setup so that the device under test is actually ipfw or pf rather than the host TCP stack.
  - Based on early tests, I have concerns that this may introduce timing irregularities.

NYCBSDCon suggestions

- Create a virtual appliance with DPD
- Add packet delay in addition to packet drop
- Repeat packets

Questions?