SWINBURNE UNIVERSITY OF TECHNOLOGY

TCP Forensics

Identifying and debugging issues with TCP in FreeBSD

Lawrence Stewart

lastewart@swin.edu.au

Centre for Advanced Internet Architectures (CAIA)
Swinburne University of Technology

Outline

1. Who is this guy?
2. TCP Congestion Control
3. Tools
4. Case studies
5. Wrapping Up

Detailed outline (section 1 of 5)

1 Who is this guy?

2 TCP Congestion Control

3 Tools

4 Case studies

5 Wrapping Up

1 Who is this guy?

Who is this guy (and who let him past security)?

- BEng (Telecomms and Internet Technologies) 1st class honours / BSci (Comp Sci and Software Eng) (2001-2006)
- Centre for Advanced Internet Architectures, Swinburne University (2003-2007)
  - Research assistant/engineer during/after studies
  - `http://caia.swin.edu.au/`
- Currently a PhD candidate in telecomms eng at CAIA
  - Strong focus on transport protocols
  - `http://caia.swin.edu.au/cv/lstewart/`
- Addicted to FreeBSD since 2003
  - Experimental research, software development, home networking, servers and personal desktops

Detailed outline (section 2 of 5)

1 Who is this guy?

2 TCP Congestion Control

3 Tools

4 Case studies

5 Wrapping Up

2 TCP Congestion Control

- TCP jargon
- Brief recap
- Where are we today
- Open issues

TCP jargon

cwnd	congestion window
MSS	maximum segment size
ssthresh	slow start threshold
ACK	TCP acknowledgment
RTT	round trip time
BDP	bandwidth-delay product
RFC	request for comment
CC	congestion control
tcpcb	TCP control block
RTO	Retransmit timeout

Brief recap

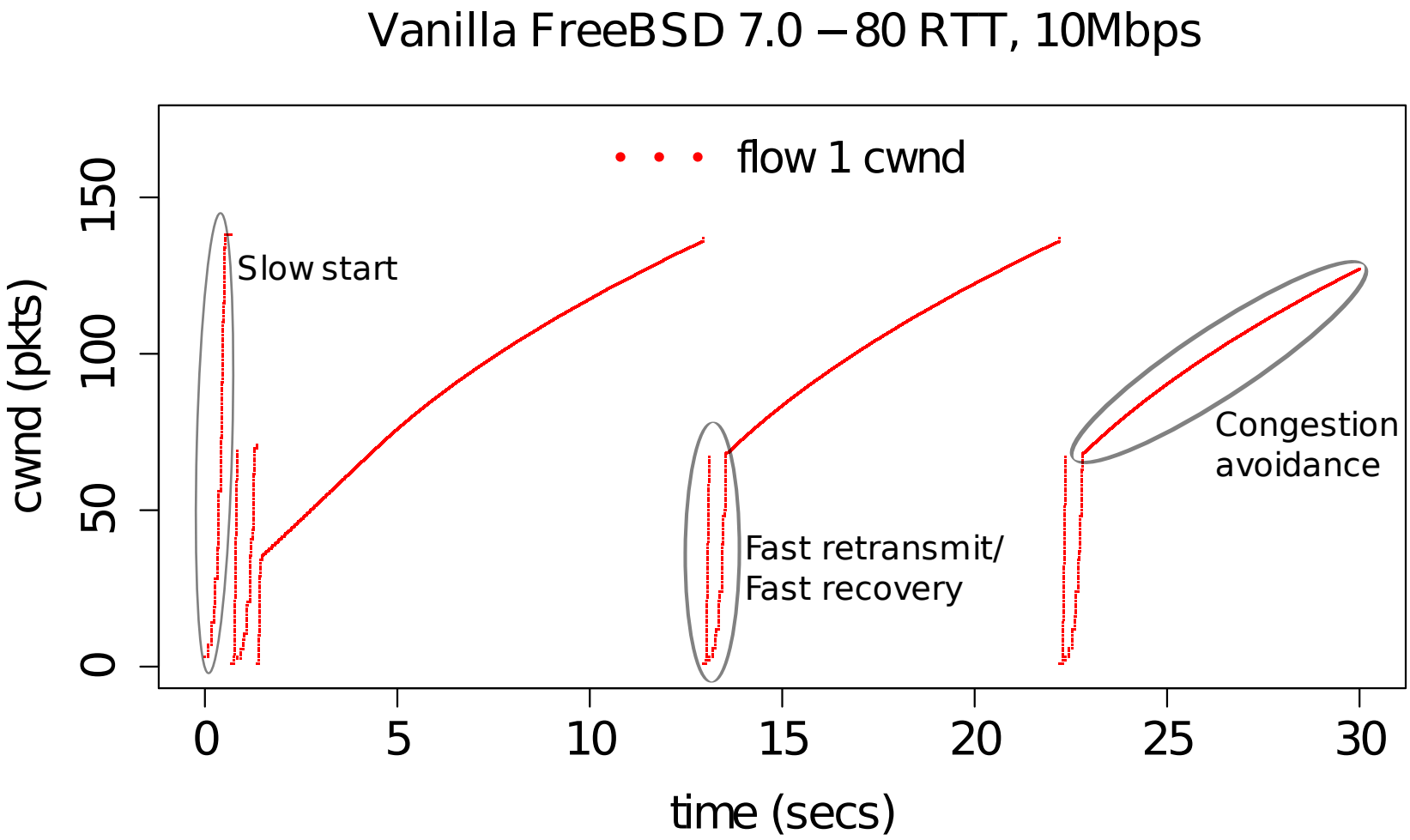
- “Congestion Avoidance and Control" (ACM SIGCOMM-88)
  - Seminal paper on TCP CC
- RFC 1122 (1989)
  - Required a TCP to use Van Jacobsen’s CC algorithms
- RFC 2582/3782 (1999/2004)
  - NewReno TCP
- RFC 4614 (2006)
  - A good summary of TCP related RFCs

Brief recap

- Core CC algorithms [1]
  - Slow start
  - Congestion avoidance
  - Fast retransmit
  - Fast recovery
- Tweaks and additions along the way
  - SACK, ABC, ECN, window scaling, timestamps, etc.

[1] See RFC2001

Brief recap


Vanilla FreeBSD 7.0 −80 RTT, 10Mbps
flow 1 cwnd
Slow start
Fast retransmit/
Fast recovery
Congestion
avoidance
cwnd (pkts)
0
50
100
150
0
5
10
15
20
25
30
time (secs)

Where are we today

- NewReno is defacto standard with warts e.g.
  - Fat pipes
  - Wireless
- State of the CC union
  - BSD uses NewReno
  - Linux uses CUBIC
  - Windows Vista uses CTCP
- TCP/IP stack enhancements
  - CSO/TSO/LRO/TOE
  - Socket buffer autotuning
- Are we going to hell in a handbasket?

Open issues

- High-speed CC algorithms [2]
  - FAST, HS-TCP, H-TCP, CTCP, CUBIC, etc.
- Delay based CC algorithms
- How do we compare and evaluate CC algorithms?
- CSO/TSO/LRO/TOE obscure behaviours
- Testing/verification of TCP/IP stack behaviour

[2] Nice summary:
http://kb.pert.geant2.net/PERTKB/TcpHighSpeedVariants

Detailed outline (section 3 of 5)

1 Who is this guy?

2 TCP Congestion Control

3 Tools

4 Case studies

5 Wrapping Up

3 Tools

- SIFTR
- SIFTR demo
- Dummynet
- Others

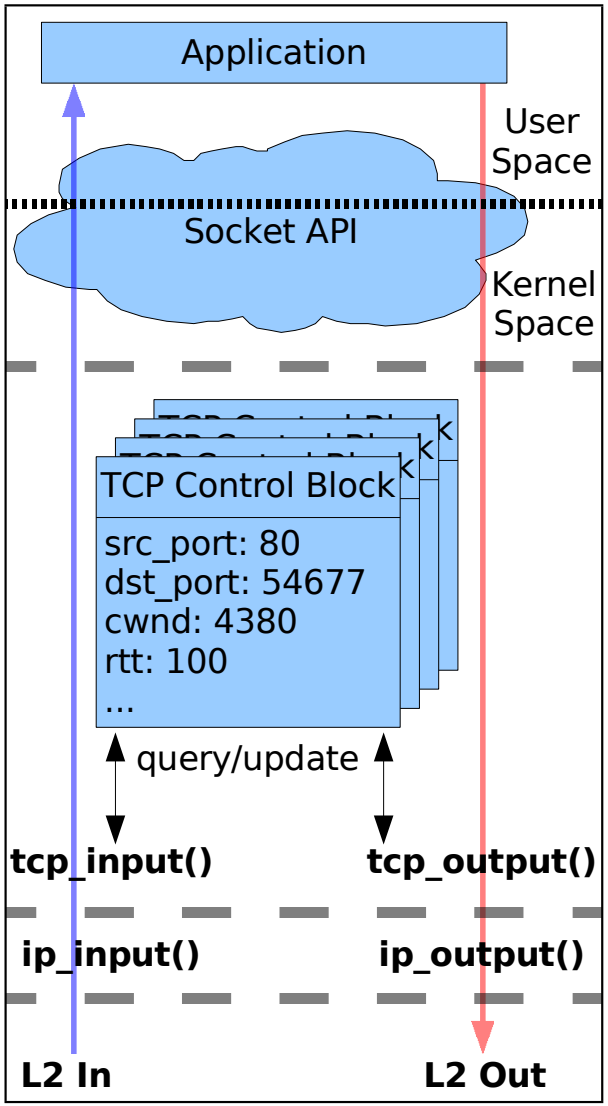
SIFTR

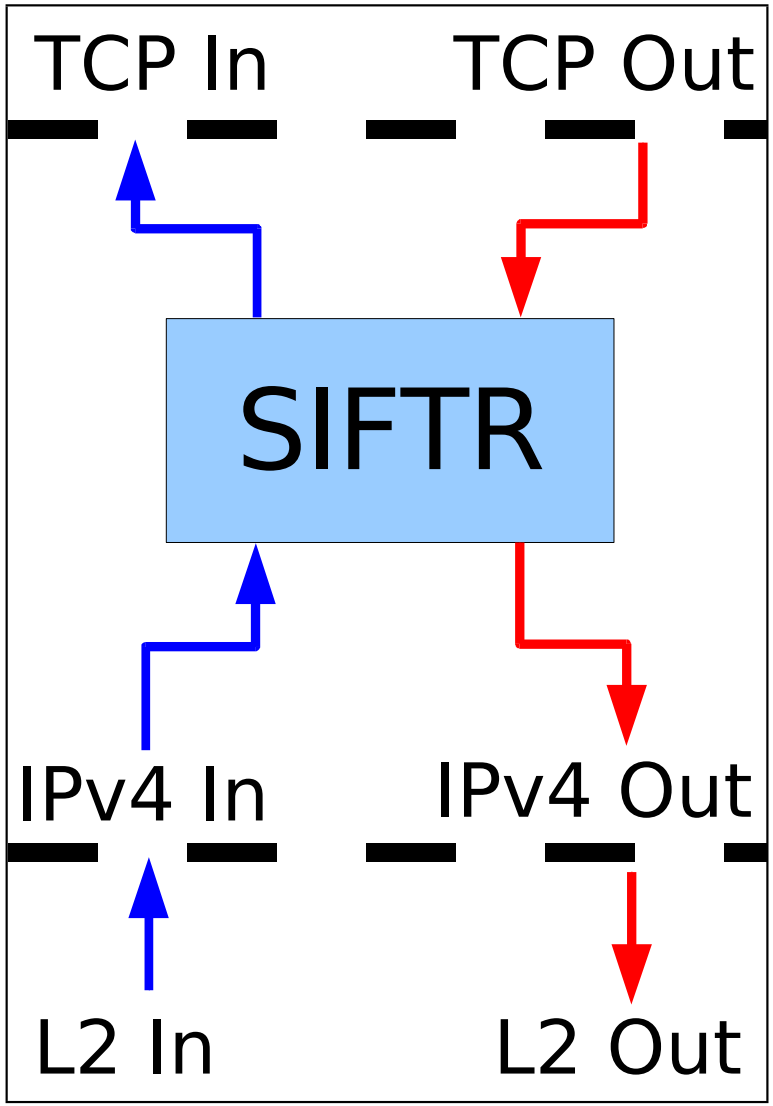
- Statistical Information For TCP Research
- FreeBSD [5,6,7,8] kernel module
- BSD licenced source [3]
- Similar base concept to Web100
- Event triggered (not poll based)
- Currently logs 25 different variables to file as CSV data [4]

[3] Available from: `http://caia.swin.edu.au/urp/newtcp/tools.html`

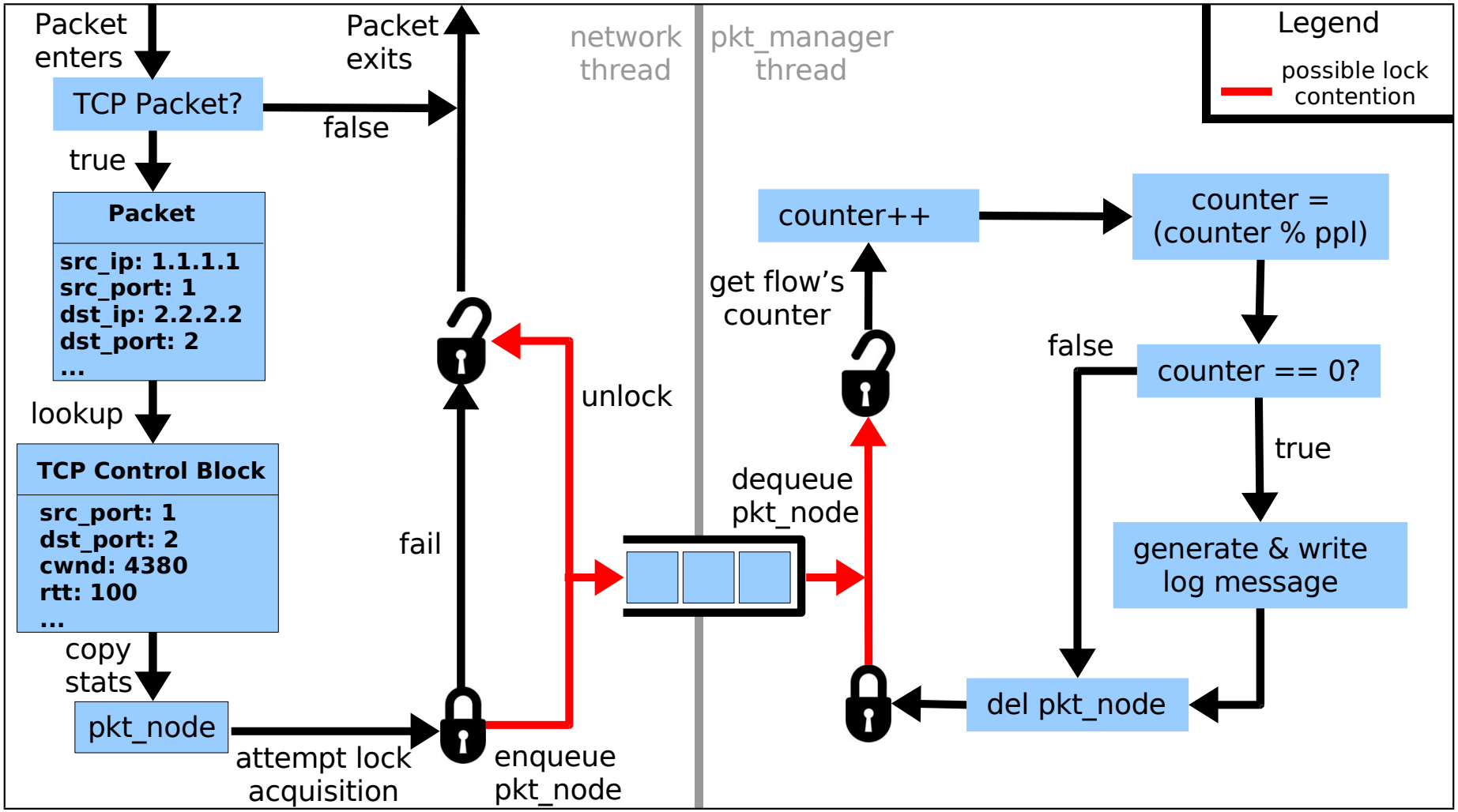
[4] See README in SIFTR distribution for specific details

SIFTR


Application
User Space
Socket API
Kernel Space
TCP Control Block
src_port: 80
dst_port: 54677
cwnd: 4380
rtt: 100
...
query/update
tcp_input()
tcp_output()
ip_input()
ip_output()
L2 In
L2 Out


TCP In
TCP Out
SIFTR
IPv4 In
IPv4 Out
L2 In
L2 Out

SIFTR


Packet enters
TCP Packet?
false
Packet exits
true
Packet
src_ip: 1.1.1.1
src_port: 1
dst_ip: 2.2.2.2
dst_port: 2
...
lookup
TCP Control Block
src_port: 1
dst_port: 2
cwnd: 4380
rtt: 100
...
copy stats
pkt_node
attempt lock acquisition
fail
unlock
enqueue pkt_node
network thread
pkt_manager thread
dequeue pkt_node
get flow's counter
counter++
counter = (counter % ppl)
false
counter == 0?
true
generate & write log message
del pkt_node
Legend
possible lock contention

SIFTR demo

Let's see what we can see!

Dummynet

- Repeatable tests makes things easier
- Patched dummynet to:
  - Include a new pipe option “pls” for predicatable packet loss
  - Provide detailed per-packet logging functionality
- New variables added to the `net.inet.ip.dummynet` sysctl tree:
  - `log_enable`: Enable/disable detailed logging
  - `dn_logfile`: Set path to logfile
  - `max_queue_size_pkts`: Set the maximum queue size for packet-based pipes
  - `max_queue_size_bytes`: Set the maximum queue size for byte-based pipes

Others

- I've personally found these invaluable:
  - Iperf
  - tcpstat
  - R
  - ns-2

Detailed outline (section 4 of 5)

1 Who is this guy?

2 TCP Congestion Control

3 Tools

4 Case studies

5 Wrapping Up

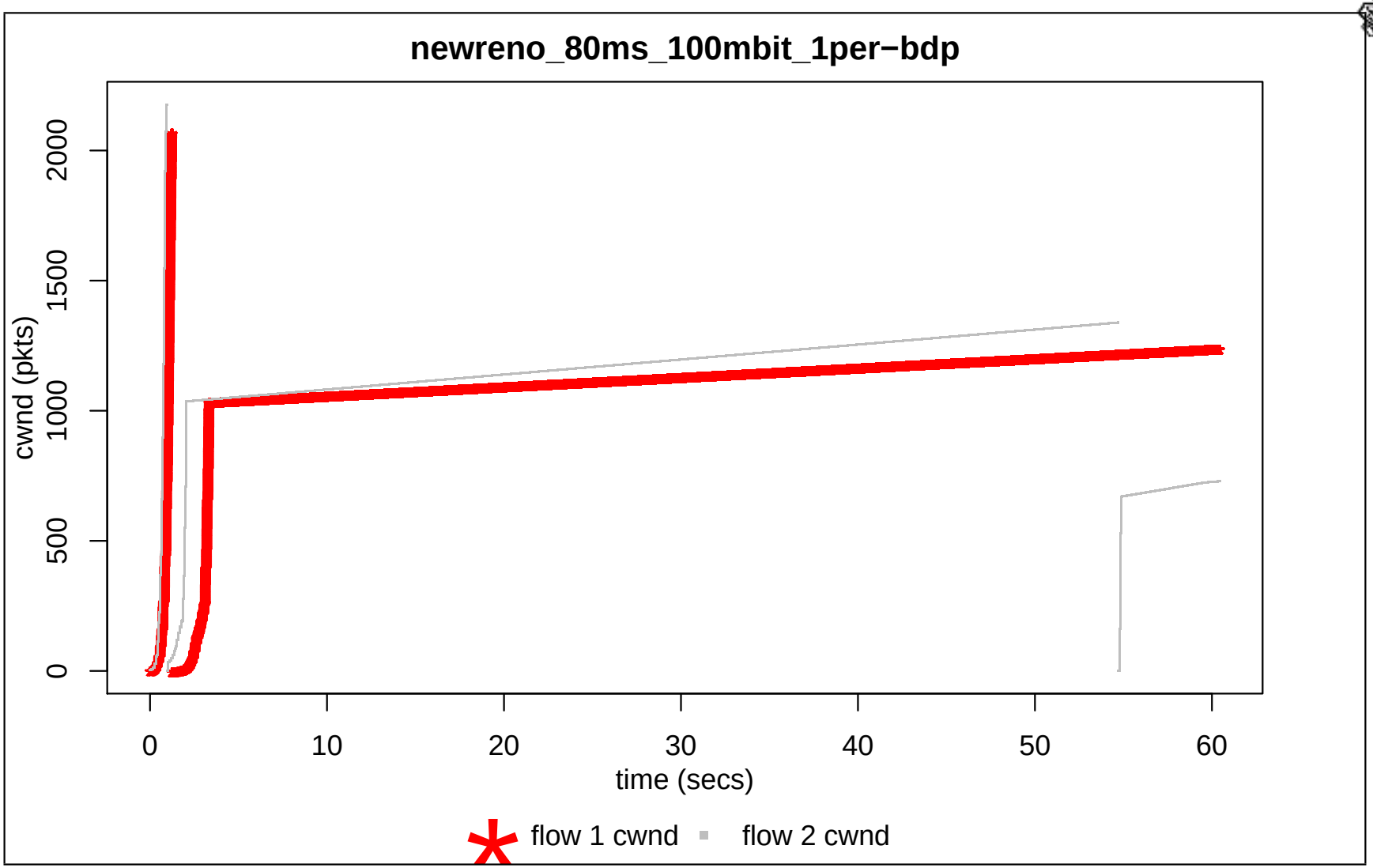
4 Case studies

- The case for ABC
- Skeletons in the (New?)Reno closet
- Holey SACK

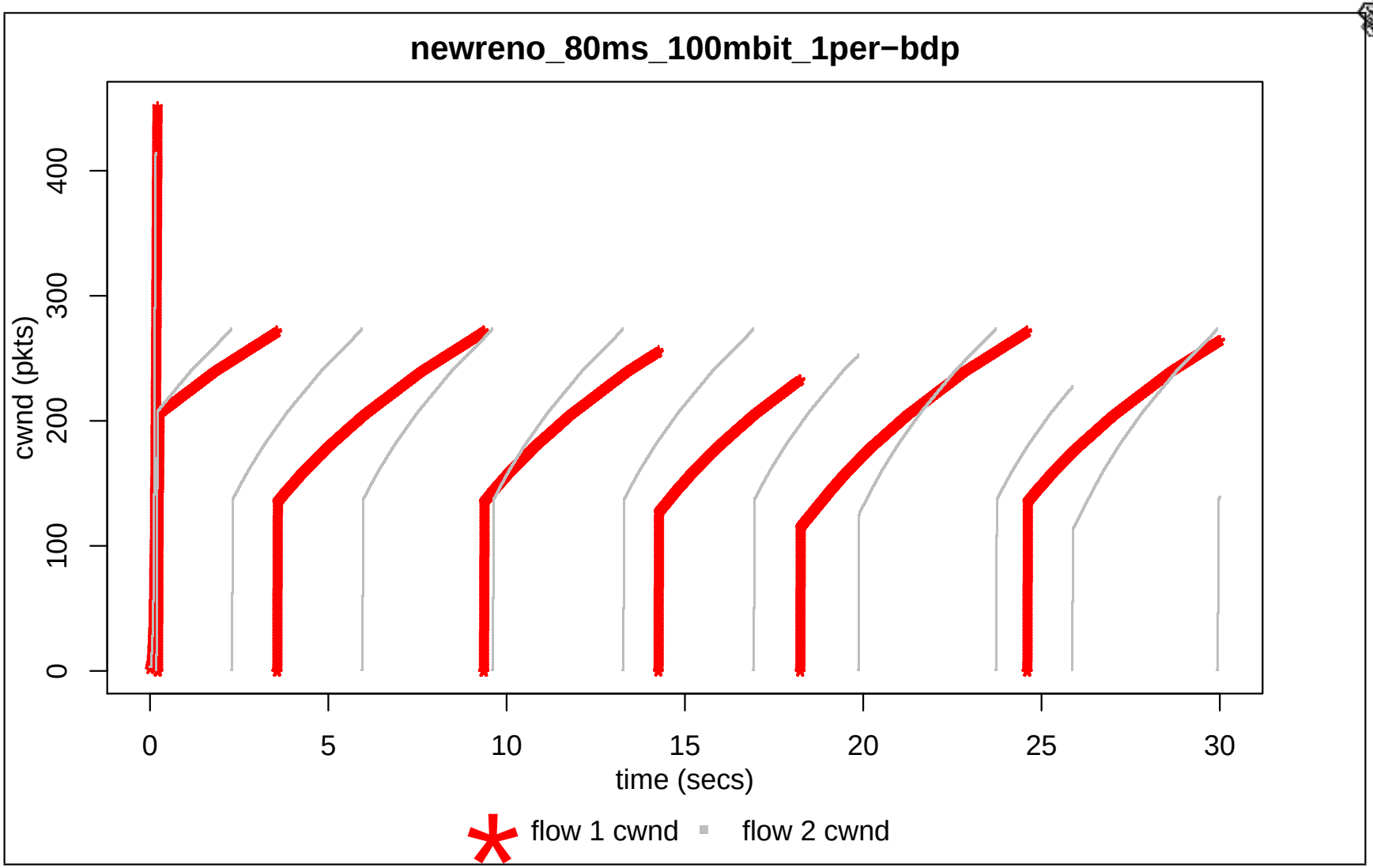
The case for ABC

- Appropriate Byte Counting (RFC 3465)
- Change cwnd increase to be based on # ACK'ed bytes
- Removes error introduced by current cwnd increase approximation:
  - For each ack, $cwnd += \alpha/cwnd$, where $\alpha = 1$ for (New)Reno

The case for ABC



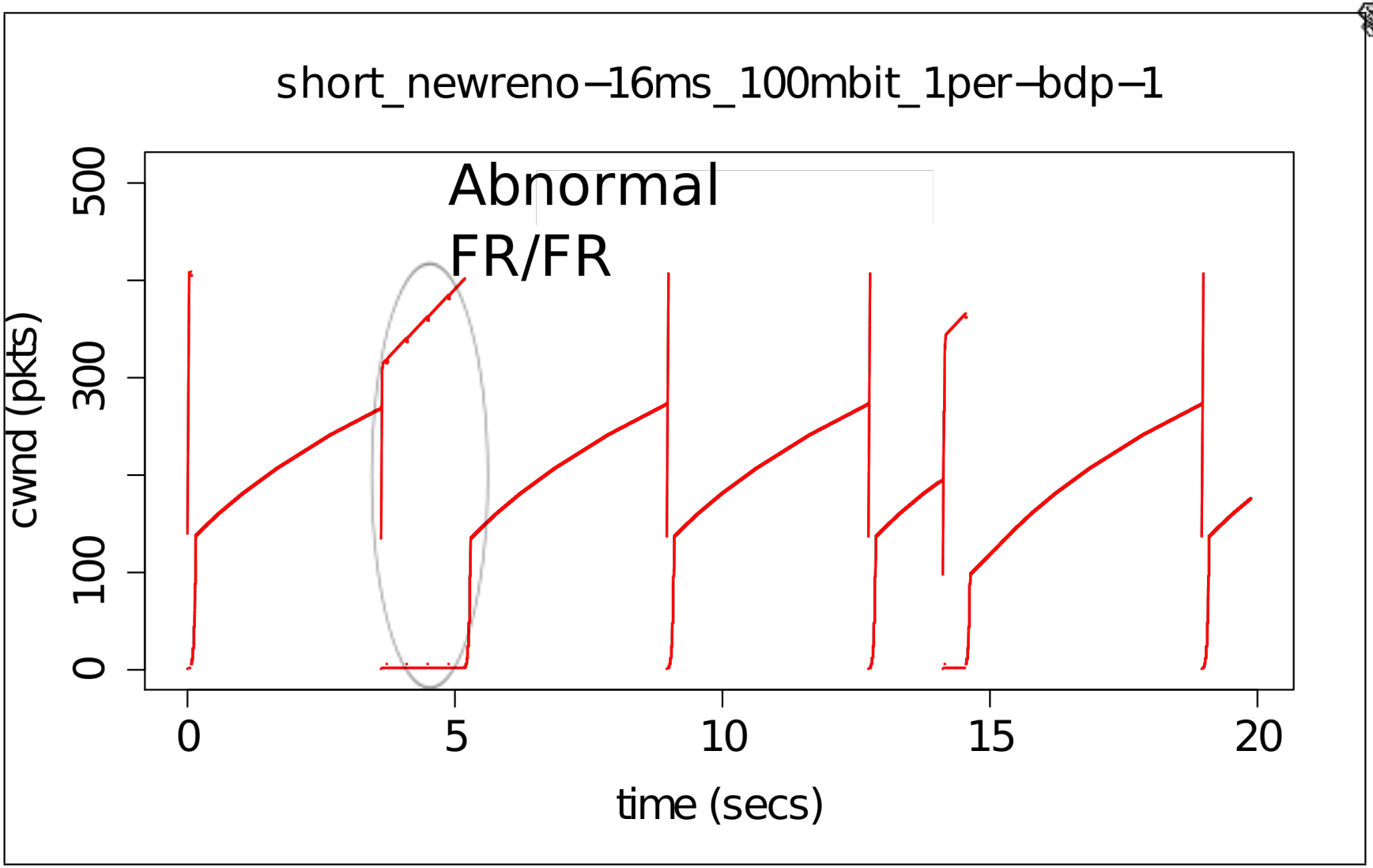
The case for ABC



Skeletons in the (New?)Reno closet

- Bug suspected to be hiding in `tcp_output()`
- Characterised by sender going quiet for an RTT during FR/FR
- Causes very laboured FR/FR which can significantly impact performance

Skeletons in the (New?)Reno closet

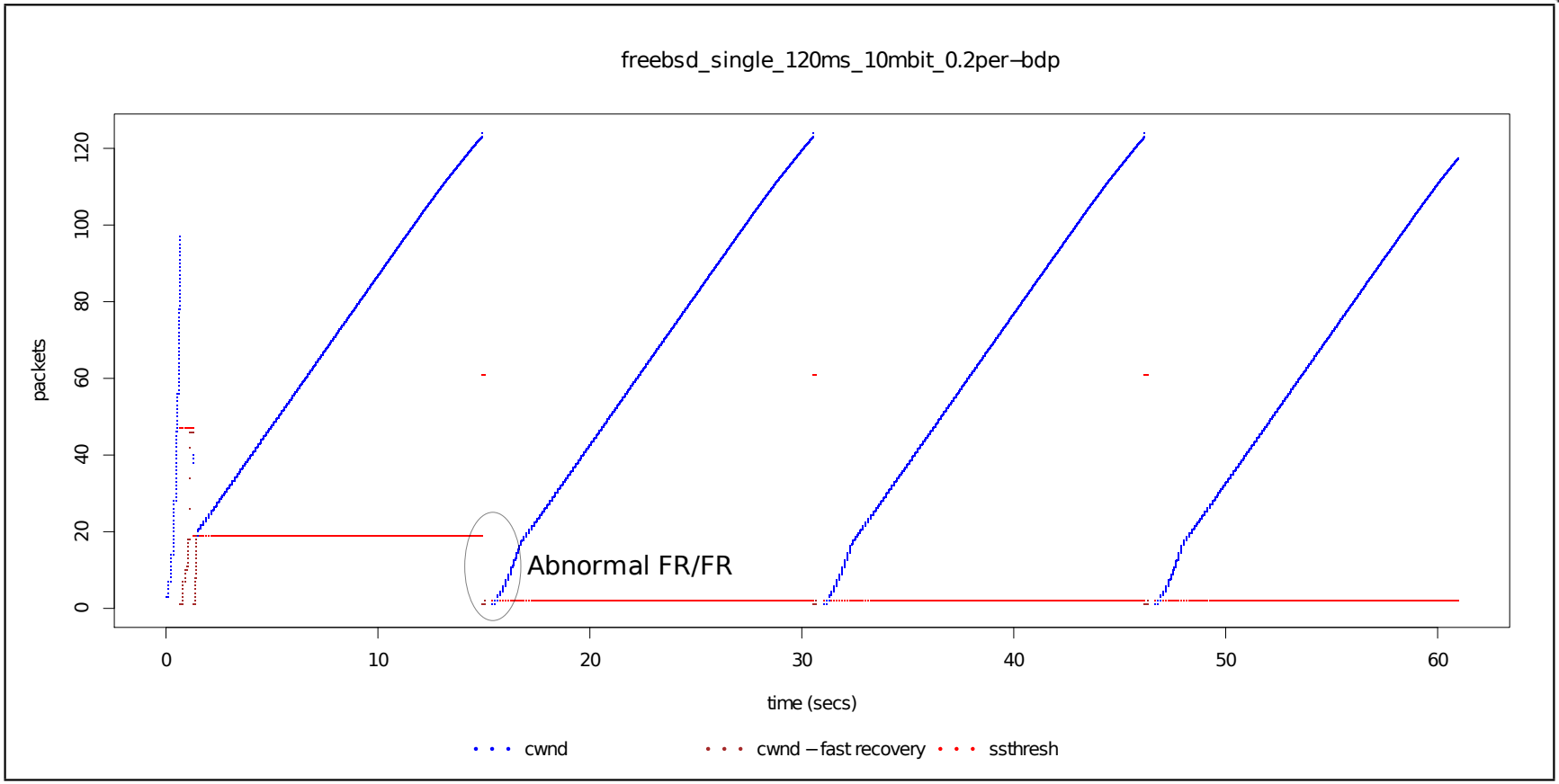


Holey SACK

- Bug in SACK implementation [5]
- Characterised by emission of 0 byte data packet by sender
- Causes an unnecessary RTO
- Triggered by packet loss to a receiver with small reassembly queue when cwnd is large
- May possibly be resolved by fixing the more general (New?)Reno bug

[5] Package of data demonstrating problem: `http://caia.swin.edu.au/urp/newtcp/freebsd_sack_issue-1.0.tar.gz`

Holey SACK


freebsd_single_120ms_10mbit_0.2per–bdp
packets
0
20
40
60
80
100
120
0
10
20
30
40
50
60
time (secs)
Abnormal FR/FR
cwnd
cwnd – fast recovery
ssthresh

Detailed outline (section 5 of 5)

1 Who is this guy?

2 TCP Congestion Control

3 Tools

4 Case studies

5 Wrapping Up

5 Wrapping Up

- Ideas for future work
- Acknowledgements
- Questions

Ideas for future work

- TCP specific:
  - RTT estimator
  - Reassembly queue autotuning (Andre)
  - Share CC between TCP/SCTP (Randall et. al.)
  - Implement more CC algorithms
  - Comprehensive RFC compliance check
- TCP/IP stack in general:
  - Framework for dealing with CSO/TSO/LRO/TOE
  - Testing framework

Acknowledgements

- The FreeBSD Foundation

- Cisco Systems

- Dan Langille et. al.
- FreeBSD community

Fin

Questions?