Enhancing the FreeBSD TCP Implementation
An Update

Lawrence Stewart
lastewart@swin.edu.au
Centre for Advanced Internet Architectures (CAIA)
Swinburne University of Technology
Outline

1. Who is this guy?
2. Projects
3. Wrapping Up
Detailed outline (section 1 of 5)

1. Who is this guy?
2. Projects
3. Wrapping Up
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/](http://caia.swin.edu.au/)
- Currently a PhD candidate in telecomms eng at CAIA (2007-)
  - Main focus on transport protocols
- FreeBSD user since 2003, developer since 2008
  - Experimental research, software development, home networking, servers and personal desktops
Detailed outline (section 2 of 5)

1. Who is this guy?

2. Projects
   - Modular Congestion Control
   - SIFTR
   - DPD
   - ABC
   - TCP Reassembly Queue
   - ALQ

3. Wrapping Up
Modular Congestion Control

■ NEWS
  ■ Project moved into public svn repository: projects/tcp_cc_8.x
  ■ Completed CUBIC implementation (unlikely to be more from me)
  ■ Significant locking improvements
  ■ Maintaining both 7.x and 8.x patches
■ TODO for 8.x (roughly in order)
  ■ Commit ABI breaking parts
  ■ Finish ECN/ABC/VIMAGE integration
  ■ Complete documentation
  ■ Commit to 8.x with experimental status i.e. no ABI guarantees
■ ISSUES
  ■ Simple framework may be needed for CC-related algorithm-agnostic tasks
  ■ Should we consider moving more variables into a CC struct?
Modular Congestion Control

- Defined in `<netinet/cc.h>`

/* specify one of these structs per CC algorithm */
struct cc_algo {
    char name[TCP_CA_NAME_MAX];
    int (*init) (struct tcpcb *tp);
    void (*deinit) (struct tcpcb *tp);
    void (*cwnd_init) (struct tcpcb *tp);
    void (*ack_received) (struct tcpcb *tp, struct tcphdr *th);
    void (*pre_fr) (struct tcpcb *tp, struct tcphdr *th);
    void (*post_fr) (struct tcpcb *tp, struct tcphdr *th);
    void (*after_idle) (struct tcpcb *tp);
    void (*after_timeout) (struct tcpcb *tp);
    STAILQ_ENTRY(cc_algo) entries;
};
Modular Congestion Control

■ Housekeeping

/* called during TCP/IP stack initialisation on boot */
void cc_init(void);

/* dynamically registers a new CC algorithm */
int cc_register_algorithm(struct cc_algo *);

/* dynamically deregisters a CC algorithm */
int cc_deregister_algorithm(struct cc_algo *);
# Modular Congestion Control

- Minor ABI-breaking additions to struct tcpcb

```c
struct tcpcb {
    ....

    /* CC function pointers to use for this connection */
    struct cc_algo *cc_algo;

    /* connection specific CC algorithm data */
    void *cc_data;
};
```
SIFTR

- Statistical Information For TCP Research
- FreeBSD [6,7,8] kernel module
- BSD licenced source \(^1\)
- Similar base concept to Web100
- Event triggered (not poll based)
- Currently logs 25 different variables to file as CSV data \(^2\)
- Plan to integrate into base system for 8.x
- Work on v1.2.x sponsored by the FreeBSD Foundation

\(^1\) Available from: http://caia.swin.edu.au/urp/newtcp/tools.html
\(^2\) See README in SIFTR distribution for specific details
SIFTR

TCP In
TCP Out

IPv4/6 in IPv4/6 out

SIFTR

IPv4/6 in IPv4/6 out

L2 In
L2 Out

L2 In
L2 Out

TCP Control Block
src_port: 80
dst_port: 54677
cwnd: 4380
rtt: 100
...
query/update

tcp_input()
tcp_output()

ip_input()
ip_output()
Deterministic Packet Discard (DPD)

- Patch against FreeBSD 8.x IPFW/Dummynet
- BSD licenced source ³
- Useful for protocol (not just TCP!) verification and testing
- Adds 'pls' (packet loss set) option for dummynet pipes
  e.g. ipfw pipe 1 config pls 1,5-10,30 would drop packets 1, 5-10 inclusive and 30
- Need to catch up with Luigi’s work
- Lower priority, but hope to commit to 7.x and 8.x soon

Appropriate Byte Counting (ABC)

- Committed to FreeBSD 8.x as r187289
- Relatively straightforward patch
- Mostly a TCP bug fix
- Some interesting side effects...
- Sponsored by the FreeBSD Foundation
Appropriate Byte Counting (ABC)

100ms RTT, 10Mbps, 62500 byte queue

- noabc
- abc
TCP Reassembly Queue

- TCP reassembly queue tuning is inherently connection specific.
- Current method is wasteful and can severely damage TCP performance.
- Aim to do away with net.inet.tcp.reass.maxqlen.
- Adapt reassembly queue based on connection dynamics.
- Somewhat akin to socket buffer auto tuning.
- Currently WIP (building on Andre’s work).
- Sponsored by the FreeBSD Foundation.
TCP Reassembly Queue

Pic of reassembly queue badness here!
Asynchronous Logging Queues (ALQ)

- Jeff Roberson’s KPI for in-kernel file logging
- Made it build as a LKM
- Extended KPI to allow variable length message support
- Under-the-hood reworked to use a circular buffer
- Useful fallout from SIFTR work
- Would like to add high water mark triggered flushing
- Plan to commit in time for 8.x, also backportable

---

Asynchronous Logging Queues (ALQ)

/* unchanged. count=0 now means size arg specifies buffer size */
int alq_open(struct alq **, const char *file, struct ucred *cred,
            int cmode, int size, int count);

/* legacy fixed length write */
int alq_write(struct alq *alq, void *data, int flags);

/* new variable length write */
int alq_writen(struct alq *alq, void *data, int len, int flags);

/* legacy fixed length ale */
struct ale *alq_get(struct alq *alq, int flags);

/* new variable length ale */
struct ale *alq_getn(struct alq *alq, int len, int flags);
1. Who is this guy?
2. Projects
3. Wrapping Up

3. Wrapping Up
   - Ideas for future work
   - Towards a Network Testing Framework
   - Acknowledgements
   - Questions
Ideas for future work

- **TCP specific:**
  - RTT estimator
  - Share CC between TCP/SCTP (Randall et. al.)
  - Comprehensive RFC compliance check
  - Fix slow-start, FR/FR

- **TCP/IP stack in general:**
  - Framework for dealing with CSO/TSO/LRO/TOE
  - DTRACEesque instrumentation
  - Testing framework <- next project I want to tackle
Towards a Network Testing Framework

- Unit/blackbox testing
- Artificial fault injection
- Some level of automation... “cd /usr/src ; make testkernel”
  anyone?
- ... penny for your thoughts?
Acknowledgements

- The FreeBSD Foundation
- Dan Langille et. al.
- FreeBSD community
- Cisco Systems
As the two friends wandered through the snow on their way home, Piglet grinned to himself, thinking how lucky he was to have a best friend like Pooh.

Pooh thought to himself: ‘If the pig sneezes, he’s fucken dead.'