Web10G

Joint Techs Tutorial (Part 1)

Andrew K. Adams, akadams@PSC.edu
John Estabrook, jestabro@ILLINOIS.edu
Chris Rapier, rapier@PSC.edu

July 15th, 2012

What is Web10G?

- Web10G is a follow on to Web100.
- Instrumentation of the Linux kernel to add TCP Extended Statistics as defined in RFC 4898.
  - Extensive per-connection metrics.
- Majority of kernel code contained in loadable kernel modules.
- Client tools for exploration of instruments.
- API for development of new applications and porting of existing applications.

Why Web10G?

- Web100 already does this so why should I care?
  - Web100 imposes unacceptable overhead due to the /proc interface.
    - It will never become part of the mainline linux kernel codebase.
- The Web100 KIS doesn't conform to RFC 4898.
- Web10G uses netlinks for the ABI allowing deployment on high volume production hosts.
  - Web100 was limited to ~30k connections.
  - Web10G should allow millions of concurrent connections.
- Web100 is no longer actively maintained.

Why adopt Web10G?

- The Web10G kernel ABI is efficient.
- The Web10G userspace API is simple and lightweight.
  - Currently 4 calls give access to all instruments in the stack.
- Web10G will be actively maintained.
  - Funded by NSF grant 1032813.
  - We are focused on mainline inclusion and will continue to work towards that goal.
- It opens up a new realm of exploration.
  - Full TCP metrics available on production servers can be a basis for research, application development, diagnostics, etc.

Kernel Instrument Set

Joint Techs Tutorial (Part 2)

RFC 4898

- The *TCP Extended Statistics MIB* exposed (or *instrumented*) in Web10G are defined in RFC 4898 (extends *TCP MIB*):
  - *TCP Extended Statistics MIB*, M. Mathis, J. Heffner, R. Raghunarayan, Request for Comments: 4898, May, 2007.
  - Referred to as the **Kernel Instrument Set** (**KIS**).
  - In effect, metrics for TCP/IP performance!
- Culmination of Web100 project.
  - Standardized Web100 and other TCP instruments.
- Web10G KIS contains *123* RFC 4898 variables.

Why is the KIS Important?

- The hour-glass shape of the OSI model hides the network from upper layers.
  - Perhaps really an artifact of the End2End Argument:
    - i.e., errors could be provided for *completely* and *correctly* in the end-hosts ...
- In any event, the OSI model is really good for scalability ...
  - ... but really bad for debugging!

- Almost all bugs have same symptom; less than expected performance.

TCP Tuning is Debugging

- All problems limit TCP/IP performance:
  - Sender/receiver buffer sizes
    - *Yeah! TCP Autotuning* fixed this!
  - Packet loss, corruption or congestion
  - Packet latency (long round trip times)
  - Packet reordering
  - Improper MSS negotiation or MTU discovery
  - Inefficient applications
- So, end-user sees less than expected performance ...
  - ... which means we must debug problem!

And Debugging Sucks

- It's all trial and error, and ...
- ... any one bug can mask any other bug(s).

- So, we need diagnostic tools.

- Web100 initially addressed this, now it's Web10G's job!

What can the KIS do?

- TCP/IP has a vantage point that we can leverage, and it *knows* how it's performing, e.g. the KIS records:
  - options and state (Window scale, SACK),
  - throughput (bytes in/out, etc.),
  - the RTT and MSS (needed for macroscopic congestion model),
  - flow and congestion control variables (rwin, cwnd, ssthresh, etc.),
  - and it knows when the sender is out of data, to name a few!

Path Diagnostics Instantiation

- TCP Macroscopic Congestion Model:
  - *The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm*, M. Mathis, J. Semke, J. Mahdavi, T. Ott, Computer Communication Review, volume 27, number 3, July 1997.

    Data Rate = **MSS** / **RTT** * 0.7 / *sqrt(**p**)*

    - Excessive RTT implies routing problem or congestion.
    - Excessive loss implies congestion or hardware issues.
    - Wrong MSS implies problem with MTU discovery.

Application Binary Interface

- Linux is moving away from /proc.
  - Web100 used the */proc interface* to expose the KIS.
- NetLink provides an ideal ABI.
- DLKM(s) expose the KIS via NetLink to userland!
- The KIS & ABI provide TCP/IP with a mechanism to *export* what TCP/IP knows!
- Web10G ultimately improves the networking experience of the end-user!

Userland API

Joint Techs Tutorial (Part 3)

Userland API; overview

- Init.
- Send (one of various) messages.
- Returned data encapsulated in easily used data structures.

and, perhaps,

- Monitor kernel events (say, of connection creation).
- Respond to events by user-defined callbacks.
- Close.

API; messages

```
enum nl_estats_msg_types {
    TCPE_CMD_LIST_CONNS,
    TCPE_CMD_READ_CONN,
        TCPE_CMD_WRITE_VAR,
    TCPE_CMD_READ_ALL,
    ...
    NLE_MSG_MAX
};
```

CMD_LIST_CONNS

- List all connections owned by requesting uid, in the form:
  - CID (connection ID; RFC 4898)
  - Local address
  - Local port
  - Remote address
  - Remote port

CMD_READ_CONN

- Request current values of all, or a subset of, MIB vars for a specified CID.
- Returned data is an array of values, encapsulated in tcpe-data struct.

CMD_READ_CONN, mask

- One has the option of sending a mask with the read_conn request, specifying a subset of MIB vars.
- This limits the time spent holding the socket locked.

CMD_WRITE_VAR

- There are a small number of writable MIB vars which can be set via this message.
- Limited, of course, to owned connections.

CMD_READ_ALL

- Read all (unmasked) vars for all (owned) connections.
- Walk the connection table, for each of which, walk the perf, path, stack, app, and tune tables.

Event notification

- Event notification delivered over GeNetlink multicast channels.
- Userland API allows to set callback to subscribe to a given channel (identified by unique string).
- Currently we only consider connection creation.

Netlink library

- The current release uses libmnl for genetlink support.
- Web10G hides the netlink client library with opaque types, so changes in this choice will not affect the API.

Porting to Web10G

- There are changes between the earlier, Web100, KIS names and RFC 4898 names.
- There are also significant differences in the API, tempered(?) by the path through the transitional API released last year.
- Both will be addressed in a porting document available in the Developers section of web10g.org
  - Available late next week.

Hands-On

Joint Techs Tutorial (Part 4)

Web10g host: frege.ncsa.illinois.edu
host: golf.psc.edu
Passwd: 3uph0rbu
listconns
watchconnmask cid -m mask
watchconnmask cid -m f,f,f,f,f
returns the first 4 entries of each of the MIB tables.
watchconnmask cid -m 0,0,0,,0
returns only the MIB app table, etc.