DTrace

get your game on

OmniTI / Seeing the whole picture

What’s this DTrace thing?

- What it isn’t?
  - A simple metrics observer:
    top, prstat, mpstat, iostat, vmstat, etc.
  - A firehose through a magnifying glass:
    strace, ltrace, ktrace, truss
- It is a surgical tool for asking questions that span all layers within a single system.

How’s it work?

- It’s kernel enabled...
- It uses instruction-level instrumentation leveraging fast-trap.
  - causes the “point of interest” to jump into a kernel-level register-based virtal machine that executes DOF (compiled D code).
  - The VM ~~is~~ can be “safe” (not stack based and limited in resource consumption by implementation).
  - once the VM is complete, it runs the instructions that were “replaced” and returns the the “point after.”
  - static probes can be defined that place noop in the right place in the code so that the instructions being replaced aren’t “important.” (think placeholders)

That sounds evil

- Yes. yes it is.
- It is designed to be provably safe.
  - it’s a good start… implementation leaves room for errors
  - I’ve had a good experience… in fact, I’m going to log into a live **production** system in a few minutes and demonstrate.

Prerequisites.

- In order to use DTrace, you need:
  - An operating system that support DTrace:
    Solaris, OpenSolaris, Mac OS X, FreeBSD, Linux (almost)
- What you need to make DTrace useful:
  - DTrace is not a firehose.
  - You need to ask questions.
  - The value of the answers is limited by the
    clarity and **intelligence** of your questions.

Deep deep deep understanding.

- You should know:
  - All the systems calls, what they do, when they are used.
  - The kernel structure (internal kernel implementation)
  - System call parameters and internal kernel structures.
  - Virtual memory system theory and implementation.
  - Virtual FileSystem (VFS) implementation.
  - IO subsystems (hard disk have heads, they move to read data)
  - C, stacks, reading machine instructions (or disassembling)
    - the more you know about the apps running,
      the more intelligent questions you can ask, and
      the more the answers mean.

DTrace providers

- syscall
- sysinfo
- vminfo
- sched
- io
- mib
- profile
- fbt
- fasttrap
- fpuinfo
- lockstat
- proc
- pid
- plockstat
- ip
- iscsi
- nfsv4
- nfsv3
- sdt

Safe... safe... boom.

- DTrace is proven safe.

- DTrace is empirically unsafe.
  (I've had it crash things, albeit rarely; more rarely than strace)

- DTrace when things go wrong:

  ```
  dtrace -q -n '............'
  dtrace: processing aborted: Abort due to systemic unresponsiveness
  ```

- Due to some bugs,
  some of which have been fixed,
  this can happen when it shouldn't;
  I need my script to run...
  what now?

-
  ```
  dtrace -w
          -w  permit destructive actions
  ```

Enough is enough... break something Theo

- Begin logging into live systems.
  - First a tour of DTrace
  - Then applied to httpd