System Call Tracing

What's that program doing?

Adam Thompson

athompso@athompso.net

2013-Feb-12 MUUG General Meeting

This work is licensed under the Creative Commons Attribution-ShareAlike 3.0 Unported License. To view a copy of this license, visit http://creativecommons.org/licenses/by-sa/3.0/.

usermod –e 2013-02-10 dshewfelt

- This presentation is dedicated to Doug Shewfelt, in remembrance of his 20 years of service to MUUG.
- Doug wrote articles for MUUG Lines starting in 1993, when he began collaborating with Arne Grimstrup.
  - "Complaining that C code is too difficult to read, Kenneth Iverson has ported the Unix kernel into a single line of APL." –Sound Bytes, April 1994
- Doug served on the committee organizing the 1993 MUUG/CIPS seminar.
- Doug was then elected MUUG's treasurer in 1994 and remained so until 2013.

(Note that this is all based on our digitized archives to date. If you have other information, please let me know so I can update this.)

Debugging Program Behaviour

- With source code:
  - Symbolic Debuggers
  - Profiling Tools
- Without source code:
  - Symbolic Debuggers
  - Call-stack tracing
  - Assembly-language inspection
  - ***System call tracing***

System Call Tracing Options

Frameworks

- DTrace
- SystemTap
- CTF (LTTng)
- ProbeVue

Individual Tools

- /[a-z]*trace/
- /[a-z]*truss/
- par
- tusc
- etc., etc., etc.

Frameworks

Who, When, Where, Why (but not How)

DTrace

- Originated at Sun, for Solaris
- Ported from OpenSolaris to FreeBSD, NetBSD, Mac OS X, Linux, QNX
- Oracle ported it (again) to Oracle Unbreakable Enterprise Linux
- The “Gold Standard” for traceability of both userland and kernel
- Vast amounts of documentation
- Requires vast amounts of knowledge to use
- Must write scripts in “D” (a DSL to define dtrace behaviour)
- Main web site shuts down in Q1’13 as Oracle retreats further from Open Source

SystemTap

- Originated at Red Hat, for Red Hat Enterprise Linux
- Now supported in almost all Linux kernels
- Originally designed to trace kernel activity, now includes userland
- Large amounts of documentation, much of it outdated
- Requires moderately large amounts of knowledge to use
- Must write scripts in a DSL that is not “D”
- Not as stable, still very useful, still very complex.

CTF / LTTng

- Originated at ?
- Now supported in almost all Linux kernels
- Originally designed to trace kernel activity, now includes userland
- Broad industry and tool support
- Requires moderately large amounts of knowledge to use
- No scripts, AFAIK
- Still very complex.

ProbeVue

- Originated at IBM, for AIX
- Has been ported to… nothing else
- IBM's answer to DTrace
- Similar features to DTrace
- Similar complexity to Dtrace
- Also appears to use a DSL

Individual Tools

Where, When, Why
(and 2 examples of How)

Individual tool coverage (*not* exhaustive!)

Tool/OS	Linux	OUEL	Solaris	FreeBSD	Mac OS X	NetBSD	OpenBSD	AIX	HPUX	IRIX	QNX
dtrace(1)	Y	Y	Y	Y	Y	Y					Y
dtruss(1)			Y	Y	Y						
ftrace(1)	Y	Y									
ktrace(1)				Y	Y	Y	Y				
latrace(1)	Y	Y	~	~		~					
ltrace(1)	Y	Y									
par(1)										Y	
ptrace(2)	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
strace(1)	Y	Y									
systrace(1)							Y				
trace(1)	Y	Y									
truss(1)			Y	Y	[*dtruss*]			Y	*[tusc]*		
tusc(1)									Y		

strace(1) on Linux

- Displays system (i.e. kernel) calls only
- Can run as a harness or attach to running process
- Many options, but default is still useful
- *[demo]*

ltrace(1) on Linux

- Displays libc calls by default, can also display system calls
- Can run as a harness or attach to existing process
- Many options, default is still useful
  - Suggest using --demangle, to decode symbol names
- *[demo]*

OK, one more

KTRACE(1) ON *BSD

ktrace(1) & kdump(1) on *BSD

- Available on all BSDs, including MacOS X.
  - dtrace(1) replaces ktrace(1) on newer versions of OS X
- ktrace(1) only records to a file, does not display output
- kdump(1) reads trace file, outputs human-readable(!) text
- Can run as a harness or attach to running process
- *[demo]*

Needle in a Haystack

What am I looking for?

Finding the Needle in the Haystack

- Key problem: sorting wheat from chaff
- Know your syscalls:
  - connect(2)
  - fopen(2)
  - etc.
- Know your syserrors:
  - ENOPERM
  - EINVAL
  - etc.