Dynamic Tracing
and the DTrace book

Brendan Gregg
Lead Performance Engineer, Joyent
BayLISA, May 2011
Agenda

• Dynamic Tracing
• DTrace
• Latency
• Performance Introspection of Cloud Computing
• DTrace Book

• Please ask questions
Dynamic Tracing is for Everyone

• DTrace is an implementation of Dynamic Tracing
  – One that has proven the concept
  – Has been used in large scale production for 5+ years
  – Provided examples of real world usage

• We may have confused the terms – they are separate

• If you never use DTrace, there is value from knowing what
  Dynamic Tracing is, to:
  – Learn what questions can conceivably be asked of the system
  – Rethink what performance observability is for – what is effective, ideal

• even if you end up not using DTrace

• Anecdote: “the most useful tool that never ran”
  – a blog post I’m writing (about har: hardware statistics tool)
Dynamic Tracing

• The ability to instrument running software
• Collect timestamps and other info (workload specifics)
• Lets you create ideal metrics. Imagine one that has:
  – Black/white answers: either there is an issue, or there isn’t
  – 100% reliability
  – No expert interpretation required
  – No time consuming analysis required
• Large scale environments usually have numerous possible issues
  – ideal metrics let you quickly identify the ones that matter, and the ones that don’t
• Given it is possible, it’s worth considering the metrics – and can lead you to design better ones to start with
  – Break the mentality of “making-do” with what you are given
• Prototype with dynamic tracing, then bake-into the software
DTrace

• Provides dynamic (and static) tracing
• Includes a rich set of actions, including data aggregations
• Currently is for:
  – Solaris-based OSes (Solaris 10+, Joyent SmartOS, Illumos, etc.)
  – Mac OS X 10.5+
  – FreeBSD 7.1+
  – Linux? http://www.crisp.demon.co.uk/blog/
    http://crtags.blogspot.com/
• Production safe; in use since 2005
• User-land and kernel-land tracing
• C & awk inspired language – easy to learn (the language)
Demo

- DTracing MySQL...
Latency

• A primary measure of application pain
  – When measured as a synchronous component of the workload
• Often not readily available when you want it: IOPS/throughput instead
• Easy to get with DTrace
• Either event by event, or summaries
  – Averages lose data; DTrace can provide the full distribution
• Locate the source of latency from the application down to the disk devices
Latency Drill Downs

Trace at each layer, for example, for disk I/O:

- Application
- Library
- System Calls
- Thread Scheduler
- VFS
- ZFS/UFS (and FS internals)
- sd
- SAS driver
- PCI driver

Any of these can be a source of latency
Performance Introspection of Cloud Computing

- Steps performed during a recent investigation of customer latency of a web app (as an example):
  1. Located latency in Apache/PHP: connect()s to MySQL (DTrace)
  2. Showed latency was not during the queries (DTrace)
     - Which saved time: investigating query-based latency was set aside for now
  3. Showed latency was not CPU dispatcher queue (DTrace)
  4. Found suspicious counters: tcpListenDrop on server (netstat)
  5. Showed Apache connects were dropped on server (DTrace)
  6. Showed TCP was retransmitting these on client (DTrace)
  7. Showed tcpListenDrops due to tcp_conn_req_max_q (DTrace)
  8. Showed connection queue length in realtime (DTrace)
  9. Showed application was not tuning accept() backlog (DTrace)
Demo

• DTracing TCP...
DTrace Changes Everything

• I spent most of my time using DTrace
• Other tools can provide clues and suggestions, but DTrace lets me confirm and move on
  – Not just for performance; also for confirming that applying patches are worthwhile, to avoid downtime when restarting apps
• If you told me before DTrace, that some new tool would be invented that I’d spend more time using than all other tools combined – I would have found it very hard to believe
DTrace Book

Aim:
• Help people use DTrace
• Share new performance metrics and ideas

Contains:
• > 270 scripts
• > 230 one-liners
• Examples
• Strategies
• Checklists
• Case Studies
# DTrace Book Chapters

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Introduction</td>
<td>18</td>
</tr>
<tr>
<td>2. The D Language</td>
<td>32</td>
</tr>
<tr>
<td>3. System View</td>
<td>100</td>
</tr>
<tr>
<td>4. Disk I/O</td>
<td>140</td>
</tr>
<tr>
<td>5. File Systems</td>
<td>108</td>
</tr>
<tr>
<td>6. Network Lower Level Protocols</td>
<td>158</td>
</tr>
<tr>
<td>7. Application Protocols</td>
<td>112</td>
</tr>
<tr>
<td>8. Languages</td>
<td>114</td>
</tr>
<tr>
<td>9. Applications</td>
<td>50</td>
</tr>
<tr>
<td>10. Databases</td>
<td>34</td>
</tr>
<tr>
<td>11. Security</td>
<td>26</td>
</tr>
<tr>
<td>12. Kernel</td>
<td>54</td>
</tr>
<tr>
<td>13. DTrace Tools</td>
<td>40</td>
</tr>
<tr>
<td>14. Tips and Tricks</td>
<td>18</td>
</tr>
</tbody>
</table>
Topics Include

- CPU, Memory, Disk, Network
- SCSI, SATA, IDE, SAS
- VFS, UFS, ZFS, HFS+, ...
- sockets, IP, TCP, UDP, ICMP, Ethernet
- NFS, CIFS, HTTP, DNS, ...
- C, Java, JavaScript, Perl, PHP, Python, Ruby
- Mysql, PostgreSQL, ...
- Kernel, Apps, ..

Shows examples of tracing each: getting started
DTraceToolkit

• In a way, the book’s scripts constitute version 2!
• They can be downloaded from: http://www.dtracebook.com
• Now that the book is done, and I have (some) spare time back, I can get back to updating the DTraceToolkit
More Info

•  http://www.dtracebook.com  All book scripts and sample chapter

•  http://dtrace.org  Many dtrace blogs

•  http://dtrace.org/blogs/brendan  My work blog

•  http://bdgregg.blogspot.com  My personal blog

•  http://www.brendangregg.com/dtrace.html  My DTrace page

•  http://www.youtube.com/joyent  Short perf talks

•  http://www.joyent.com  Company homepage

@brendangregg on twitter
Extra Info – DTrace-Based Latency Heat Map

- “Visualizing System Latency”, CACM, July 2010

http://cacm.acm.org/magazines/2010/7/95062-visualizing-system-latency/pdf
Extra Info – DTrace-Based Cloud Analytics