Dtrace for Java Developers

What Java developers should know about Dtrace

Jason Brazile and Stefan Tramm
Netcetera
4260

JAZOON08
THE INTERNATIONAL CONFERENCE ON JAVA TECHNOLOGY
JUNE 23 - 26, 2008 ZURICH

DTrace: What and Why?

- **What**?
  - It is a scriptable, unobtrusive, system observer with global (not just process) scope for visibility of system calls, application-level stack traces, etc.
- **Why**?
  - JMX can only monitor what someone wrote an MBean for
  - `truss(1)` can only snapshot system calls/signals/faults at process scope
  - `vmstat(1)` can only look at the vm subsystem
  - `iostat(1)` can only look at the io subsystem
  - `pstack(1)` can only look at a stack
  - …

Dtrace does these and more… and is scriptable…

Motivation I: Some problems observed by dtrace

- Where too much JVM garbage collection occurs and why?
- Where and how often Java monitor contention occurs?
- How reconfiguring an app's Oracle temp tables can win big
- How MacOSX system ticks are swallowed by iTunes
- How using tmpfs with ganglia/mysql can win big
- Why does my app have so many active threads?
- …

(answers to these found on urls on slide: "links: amazing use cases")

Motivation II: Example one-liners

- **System call count, by process**
  - dtrace -n 'syscall:::entry { @num[pid,execname] = count(); }'
- **Files opened by process (jboss, tomcat, etc)**
  - dtrace -n 'syscall::open*:entry { printf("%s %s",execname,copyinstr(arg0)); }'
- **Number of bytes read, by process**
  - dtrace -n 'sysinfo:::readch { @bytes[execname] = sum(arg0); }'
- **Write size distribution, by process**
  - dtrace -n 'sysinfo:::writech { @dist[execname] = quantize(arg0); }'
- **Sample java stack at 1001 Hertz (aka profile)**
  - dtrace -n 'profile-1001 /pid == $target/ { @num[jstack()] = count(); }' -p PID

Observability (before Dtrace)…

Level	What	How
Language Level:	.class	Eclipse/Visual Studio
Virtual Machine:	java/java.exe	jmx
System Runtime:	libc.so/.dll	sotruss,dbx/traceplus
System Call:	fork()/NtCreateProces()	strace,truss/traceplus
OS Kernel:	vm_page_alloc()/?	vmstat/Task Manager
Hardware:	disk/network controller	cat /proc/scsi/sg/debug

Dtrace covers much of these… and is scriptable…

Gregg's Dtrace toolkit (> 200 "canned" scripts)

> iotop	top disk I/O events by process
> tcpsnoop	snoop TCP packets by process
> prustat	%CPU, %Mem, %Disk,%Net by process
> dtruss	less burdensome version of truss(1)
> dapptrace	traces user/library function usage
> dappprof	profiles user/library function usage
> errinfo	monitor system call failures
> …	

…a good way to learn how to write custom D scripts

DTrace in Java (>=JSE6*)

- Probes that are enabled by default (no performance impact)
  - Garbage collection/Method compilation/Thread lifecycle/Class loading
- Probes that are disabled by default
  - Object allocation `java -XX:+DTraceAllocProbes`
  - Method invocation `java -XX:+DTraceMethodProbes`
  - Java monitor events `java -XX:+DTraceMonitorProbes`

Can also be enabled dynamically with jinfo

`jinfo –flag +ExtendedDTraceProbes <pid>`

(`ExtendedDTraceProbes` is a ***shortcut*** enabling the above disabled probes***)***

- ***New***: User-defined static probes (JSDT) in Java (>=JSE7)
  - Extend the **`Provider`** interface –> observability of that now scriptable via D

*(*note: JSE5 limited Solaris 10 support possible with dvm `dtrace-vm-agents` addon)*

A (Stolen) Simple Example

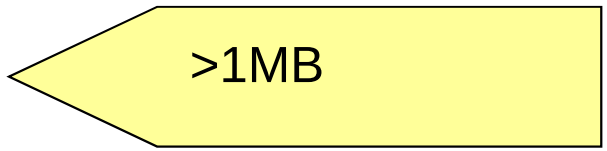
- **What: Sun's Java 2D demo program**
- **Problem: Excessive garbage collection**
  - What are largest (aggregate) object allocations?
  - What are the call stacks when a particular allocation type occurs?
  - Strategy: use **`object-alloc`** probe + sum/count aggregation
  - **Solution: 2 one-liners**

Examples: Jarod Jenson, DTrace and Java: Exposing Performance Problems That Once Were Hidden

DTrace Java Example: Excessive GC (I)

> Show object allocations and total size of those allocations

```
$ dtrace -n hotspot116977:::object-alloc'{@[copyinstr(arg1, arg2)] = sum(arg3)},
dtrace: description 'hotspot116977:::object-alloc' matched 1 probe
^C
[…]
java/awt/geom/LineIterator                                  19608
java/util/HashMap$Entry                                     21936
java/awt/geom/Point2D$Double                                26160
sun/java2d/pipe/Region                                      39072
java/awt/geom/Path2D$Double                                 94368
java/awt/geom/Rectangle2D$Double                           106720
sun/java2d/SunGraphics2D                                   112200
[B                                                         154624
java/awt/Rectangle                                         166656
[F                                                         216824
java/awt/geom/RectIterator                                 242928
java/awt/geom/AffineTransform                              500224
[I                                                        1609512
[D                                                        2207120
```



Examples: Jarod Jenson, DTrace and Java: Exposing Performance Problems That Once Were Hidden

DTrace Java Example: Excessive GC (II)

> Dump the stack (20 frames) whenever an Integer array is allocated

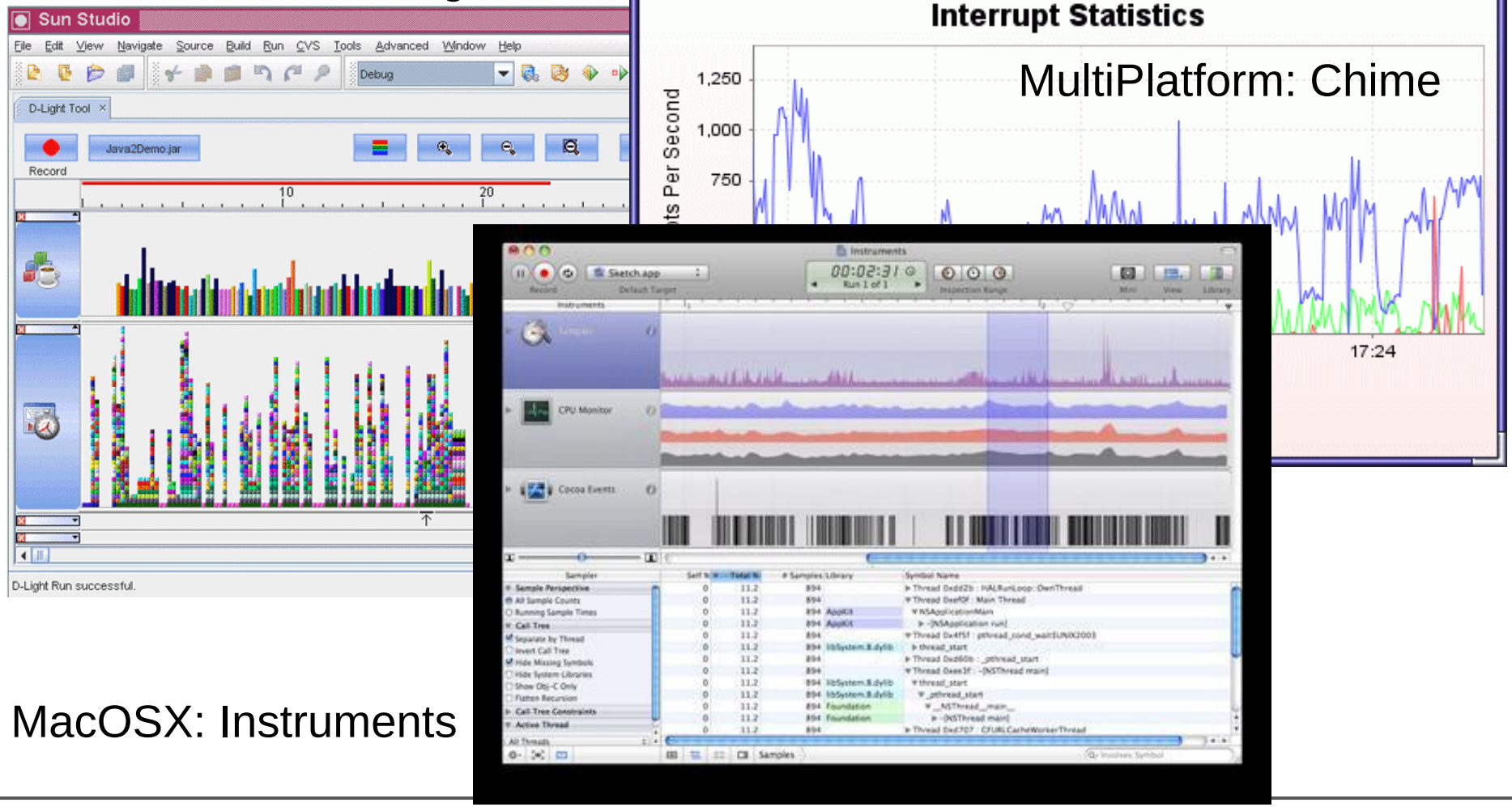
```
$ dtrace -n hotspot116977:::object-alloc'/copyinstr(arg1, arg2) == "[I"/{@[jstack(20,2048)] = count()}'
dtrace: description 'hotspot116977:::object-alloc' matched 1 probe
^C

libjvm.so`__1cNSharedRuntimeYdtrace_object_alloc_base6FpnGThread_pnHoopDesc__i_+0x7d
libjvm.so`__1cNSharedRuntimeTdtrace_object_alloc6FpnHoopDesc__i_+0x4f
libjvm.so`__1cNCollectedHeapbCpost_allocation_setup_common6FnLKlassHandle_pnIHeapWord_I_v_+0x121
libjvm.so`__1cOtypeArrayKlassIallocate6MipnGThread__pnQtypeArrayOopDesc__+0x19b
libjvm.so`__1cKoopFactoryNnew_typeArray6FnJBasicType_ipnGThread__pnQtypeArrayOopDesc__+0x2f
libjvm.so`__1cSInterpreterRuntimeInewarray6FpnKJavaThread_nJBasicType_i_v_+0x33
sun/java2d/pipe/DuctusShapeRenderer.renderPath(Lsun/java2d/SunGraphics2D;Ljava/awt/Shape;Ljava/awt/BasicStroke;)V
sun/java2d/pipe/DuctusShapeRenderer.fill(Lsun/java2d/SunGraphics2D;Ljava/awt/Shape;)V
sun/java2d/pipe/PixelToShapeConverter.fillRect(Lsun/java2d/SunGraphics2D;IIII)V sun/java2d/SunGraphics2D.fillRect(IIII)V
sun/java2d/SunGraphics2D.clearRect(IIII)V java2d/Intro$Surface.paint(Ljava/awt/Graphics;)V
javax/swing/JComponent.paintToOffscreen(Ljava/awt/Graphics;IIIII)V
javax/swing/BufferStrategyPaintManager.paint(Ljavax/swing/JComponent;Ljavax/swing/JComponent;Ljava/awt/Graphics;IIII)Z
javax/swing/RepaintManager.paint(Ljavax/swing/JComponent;Ljavax/swing/JComponent;Ljava/awt/Graphics;IIII)V
javax/swing/JComponent._paintImmediately(IIII)V javax/swing/JComponent.paintImmediately(IIII)V
javax/swing/RepaintManager.paintDirtyRegions(Ljava/util/Map;)V
javax/swing/RepaintManager.paintDirtyRegions()V javax/swing/RepaintManager.seqPaintDirtyRegions()V
94
```

Examples: Jarod Jenson, DTrace and Java: Exposing Performance Problems That Once Were Hidden

New: Dtrace GUIs (For limited use cases)

Solaris: D-Light



MacOSX: Instruments

What to remember…

- For: app developers, db admins, security admins, kernel developers
- Java findings portable (observe on one system -> apply to others)
- not just Java -> e.g. interaction with DB/network/filesystem/etc
- Java programmers can add user-defined probes (JSDT) in Java SE7
- higher-impact JVM observers can be switched on/off at runtime via jinfo
- GUI-based interfaces available (for limited use cases)

… and …

…if you only remember 2 things…

- dtrace is *always available(*)*, without restarting application; *low impact* when running, *zero impact* when not…
- *dtrace toolkit* scripts are very helpful – but *custom* scripts can be *amazing…*

(*) Solaris 10/OpenSolaris, MacOS X Leopard, FreeBSD (soon); (>= JSE6 for Java-specific probes)
NOT AVAILABLE ON: Linux (see SystemTap) or Microsoft Windows operating systems

Jason Brazile
Netcetera
http://netcetera.ch/
jason.brazile@netcetera.ch

Stefan Tramm
Netcetera
http://netcetera.ch/
stefan.tramm@netcetera.ch

Extra Slides

Links

- DTrace
  - McDougall et al, *Solaris Performance and Tools*, Prentice Hall, 2006
  - http://www.brendangregg.com/dtrace.html (D scripts)
  - http://www.opensolaris.org/os/community/dtrace/dtracetoolkit/
  - https://solaris10-dtrace-vm-agents.dev.java.net/ (dvm for JSE<6)
  - http://blogs.sun.com/kamg/entry/adding_user_defined_dtrace_probes (JSE>6)
- Custom use of Dtrace (Amazing Use-Cases!)
  - http://www.devx.com/Java/Article/33943
  - http://www.forsythesunsolutions.com/node/34
  - http://blogs.sun.com/ahl/entry/mac_os_x_and_the
  - http://www.joyeur.com/2008/04/24/dtrace-mysql-ganglia-and-digging-for-solutions
  - http://blogs.sun.com/jonh/entry/dtrace_javaone

DTrace: History

- 1996: conceived of by Bryan Cantrill while an undergrad at Brown
- 2001: Started work on Dtrace with Michael W. Shapiro
- 2002: Early prototype, Adam H. Leventhal joined development
- 2004: Appeared in Solaris
- 2005: Sun released DTrace source code, initial java support (dvm)
- 2006: Ported to FreeBSD by John Birrell
- 2007: Released in MacOSX 10.5 “Leopard” and basis of “Instruments”
- 2008: Native Java Static Probes (JSDT) (similar to USDT)

DTrace: Design features

- Provides (>30k) instrumentation points…
  - …in the kernel
  - …in the C runtime library
  - …in the Java VM
  - …in Java itself
- Safe (probes not allowed to crash system)
- Extensible (users can implement their own instrumentation points)
- Predicates avoid retaining (copying, and storing) unneeded data
- Provides scalable aggregation (sum, min, avg, quantize, count, etc.)
- High level awk-like language, called D

A Second Example…

- **Problem: Contended monitor**
  - List (quantized) time between requested entrance and actual entrance
  - Strategy: use **`monitor`** probe + **`timestamp`** and **`quantize`**
  - **Solution: a 10-liner** using **`monitor`** + quantize

Examples: Jarod Jenson, DTrace and Java: Exposing Performance Problems That Once Were Hidden

DTrace Java Example: Monitor Contention (I)

> Bucketed distribution of monitor acquisition wait times, per-thread

```
$./monitor-contend.d `pgrep java`
dtrace: script './monitor-contend.d' matched 2 probes
^C
    30
    value ------------- Distribution ------------- count
[…]
    14
    value ------------- Distribution ------------- count
      1024 |                                         0
      2048 |@@@@                                     12
      4096 |@                                        4
      8192 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ 84
     16384 |                                         0
     32768 |@                                        2
     65536 |@                                        2
    131072 |@@@                                      7
    262144 |                                         1
    524288 |                                         0
```

8 times >= 131ms

Examples: Jarod Jenson, DTrace and Java: Exposing Performance Problems That Once Were Hidden

DTrace Java Example: Monitor Contention (II)

> And the D script used in the previous slide…

```
$ cat monitor-wait.d
#!/usr/sbin/dtrace -s
hotspot$1:::monitor-contended-enter
{
    self->ts = timestamp;
}
hotspot$1:::monitor-contended-entered
/ self->ts /
{
    @[tid] = quantize(timestamp - self->ts); self->ts = 0;
}
```

Examples: Jarod Jenson, DTrace and Java: Exposing Performance Problems That Once Were Hidden