

ORACLE®

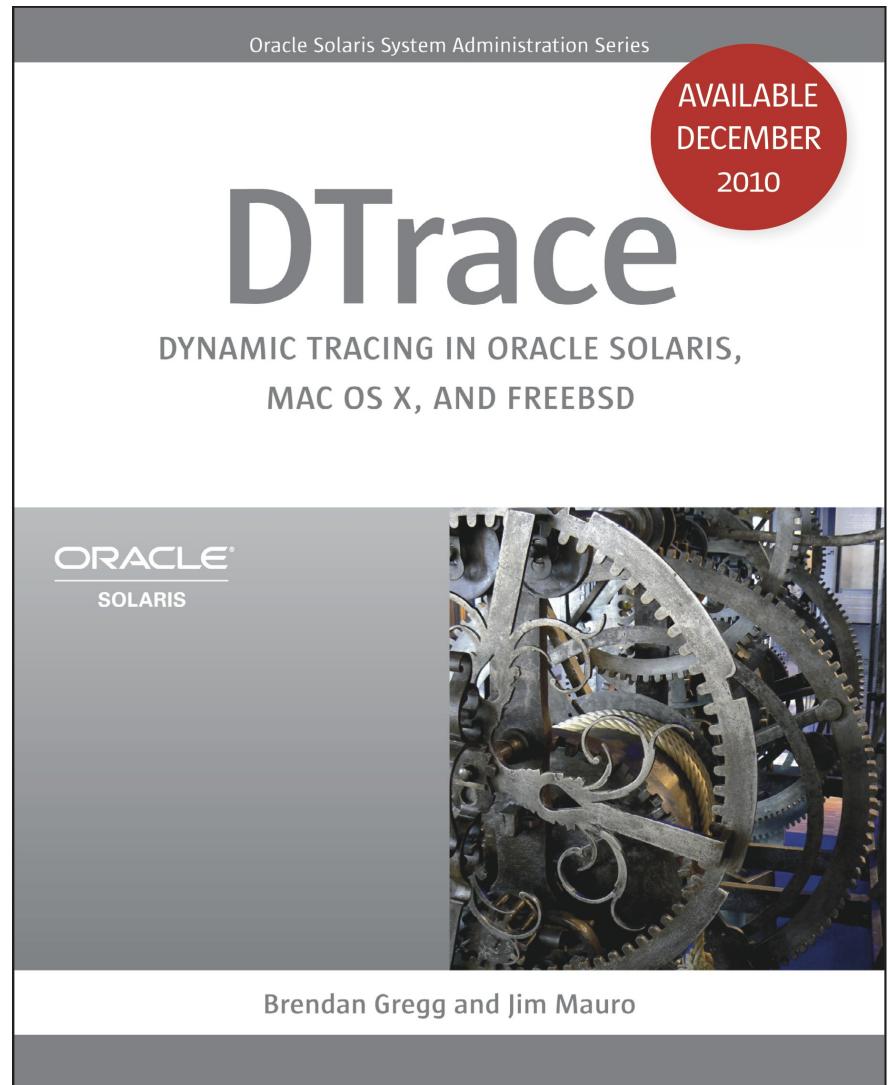
Oracle Solaris Dynamic Tracing – DTrace

Jim Mauro
Senior Software Engineer

Coming Soon!

- DTrace Cook Book
 - Over 230+ D scripts
 - Over 270+ one-liners
 - Shows the entire software stack, by example
 - Topics include:
CPU, Memory, Disk I/O,
Network I/O, Applications,
Databases, Languages,
System analysis, The kernel,
etc, ...

<http://www.dtracebook.com>



Solaris Dynamic Tracing - DTrace

*“ [expletive deleted] It's like they saw
inside my head and gave me The One
True Tool.”*

- A Slashdotter, in a post referring to DTrace

*“ With DTrace, I can walk into a room of
hardened technologists and get them
giggling”*

- Bryan Cantrill, Father of DTrace

Using DTrace

- DTrace is a powerful tool designed to observe complex systems
- The depth and breadth of your analysis with DTrace will depend in part on knowledge, experience and skill
- But you do not need to be a software engineer, or kernel engineer, or have a PhD in Computer Science to make practical use of DTrace
- It's relatively easy to observe easy-to-understand statistics with DTrace
- And...use DTrace as a learning tool. You can improve your knowledge of systems and software by using DTrace to observe systems.

DTrace Components

- Probes
 - A point of instrumentation and data generation
- Providers
 - A major component of DTrace, Providers manage probes of specific types, and for a specific area of the system
 - syscall, io, sched, proc, vminfo, etc
- Consumers
 - Users of the framework

`dtrace(1), lockstat(1), plockstat(1), intrstat(1)`

DTrace User Components

- Predicates
 - User-defined conditional statements evaluated when probes fire
 - Provides a control flow mechanism for your D programs – data pruning at the source
- Actions
 - What to do when the probe fires
 - Data to gather
 - Timestamps for profiling
 - Many other actions supported

A D Program

```
probe
/ optional predicate /
{
    clause
        what to do when the probe(s) fire, and the predicate,
        if present, evaluates true
}
```

Example:

```
syscall::read:entry
/ execname == "java" /
{
    @reads[pid, fds[arg0].fi.pathname] = count();
}
```

Or, via the command line;

```
#dtrace -n 'syscall::read:entry / execname == "java" /
{ @reads[pid, fds[arg0].fi.pathname] = count(); }'
```

Getting Started

- It's always best to start at the beginning
- Examine system resources using conventional tools
 - CPUs (vmstat, mpstat)
 - Memory (vmstat)
 - Disks (iostat)
 - Network (netstat)
- Examine running processes
 - top, prstat
- Don't forget the basics
 - What problem are you trying to solve?
 - What do you need to measure?

Before You Begin...

“Would you tell me, please, which way I ought to go from here?” asked Alice

“That depends a good deal on where you want to get to” said the Cat

“I don’t much care where...” said Alice

“Then it doesn’t matter which way you go” said the Cat



Lewis Carroll
Alice’s Adventures in Wonderland

Performance Metrics

- **Throughput – *how fast***

- Typically a bandwidth measurement
 - MB/sec, Gb/sec
- An application/workload business metric
 - Transactions per second

- **Latency – *how long***

- How much time to read a disk (milliseconds), network packet roundtrip (microseconds), read from memory (nanoseconds)

- **IOPS/OPS – *how many***

- Disk IOPS – reads/writes per second
- Network packets – transmits/receives per second
- Process/thread creates

- **Utilization – *how much***

- CPU - %busy (usr + sys) versus %idle
- IO – available IOPS or bandwidth versus used

DTrace – Getting the Big Picture

- Once you've learned what you can from the "stat" tools
- **syscall provider**
 - An useful place to start
 - Observe how application threads are using the kernel
 - Determine which process threads are issuing which calls
- **sched provider**
 - Which threads are getting on CPU
 - What are threads sleeping on
- **sysinfo**
 - Which sys events are occurring (which probes are firing)
- **profile/tick**
 - Where's the kernel spending time
 - Which user threads are running

Looking at CPUs

- Metrics important to CPUs
 - Utilization
 - How busy?
 - Usr (user) time versus sys (kernel) time
 - Run queues
 - Scheduling latency
 - Are runnable threads waiting for a CPU?
 - How long are they waiting?
 - Are run queues balanced across CPUs?
 - Scheduling and resource management options
 - Scheduling classes
 - Resource capping
 - Processor sets/Resource pools

DTrace Providers for CPUs

- profile/tick
 - time based profiling
- sched
 - kernel scheduler activity
- proc
 - process/thread activity
- fbt
 - kernel functions
- lockstat
 - kernel profile/kernel lock statistics
- plockstat
 - Application lock staticics
- syscall
 - system calls
- pid
 - Looking are user processes

Getting Started – One-liners looking at CPU

Which processes are on CPU?

```
dtrace -n 'profile-997hz { @[pid, execname] = count(); }'
```

Which processes are on CPU, running user code?

```
dtrace -n 'profile-997hz /arg1/ { @[pid, execname] = count(); }'
```

What are the top user functions running on CPU (% usr time)?

```
dtrace -n 'profile-997hz /arg1/  
{ @[execname, ufunc(arg1)] = count(); }'
```

What are the top kernel functions running on CPU (%sys time)?

```
dtrace -n 'profile-997hz /arg0/ { @[func(arg0)] = count(); }'
```

What are the top 5 kernel stack traces on CPU (shows why)?

```
dtrace -n 'profile-997hz { @[stack()] = count(); }  
END { trunc(@, 5); }'
```

What are the top 5 user stack traces on CPU (shows why)?

```
dtrace -n 'profile-997hz { @[ustack()] = count(); }  
END { trunc(@, 5); }'
```

Getting Started – One-liners – Looking at CPU

What threads are on CPU, counted by their thread name? (FreeBSD)

```
dtrace -n 'profile-997 { @[stringof(curthread->td_name)] = count(); }'
```

Which processes are getting placed on CPUs? (sched provider – event based)

```
dtrace -n 'sched:::on-cpu { @[pid, execname] = count(); }'
```

Which processes are getting charged with CPU time when tick accounting is performed?

```
dtrace -n 'sched:::tick { @[stringof(args[1]->pr_fname)] = count(); }'
```

What system calls are being executed by the CPUs?

```
dtrace -n 'syscall:::entry { @[probefunc] = count(); }'
```

Which processes are executing the most system calls?

```
dtrace -n 'syscall:::entry { @[pid, execname] = count(); }'
```

Performance Metrics

- **Throughput – *how fast***

- Typically a bandwidth measurement
 - MB/sec, Gb/sec
- An application/workload business metric
 - Transactions per second

- **Latency – *how long***

- How much time to read a disk (milliseconds), network packet roundtrip (microseconds), read from memory (nanoseconds)

- **IOPS/OPS – *how many***

- Disk IOPS – reads/writes per second
- Network packets – transmits/receives per second
- Process/thread creates

- **Utilization – *how much***

- CPU - %busy (usr + sys) versus %idle
- IO – available IOPS or bandwidth versus used

CPU - Example

```
solaris# dtrace -n 'profile:::profile-997hz /arg1/{ @[pid, execname] = count(); }'
^C
[...output truncated...]
 2735 oracle.orig          4088
 2580 oracle.orig          4090
 2746 oracle.orig          4093
 2652 oracle.orig          4100
 2748 oracle.orig          4108
 2822 oracle.orig          4111
 2644 oracle.orig          4112
 2660 oracle.orig          4122
 2554 oracle.orig          4123
 2668 oracle.orig          4123
 2560 oracle.orig          4131
 2826 oracle.orig          4218
 2568 oracle.orig          4229
 2836 oracle.orig          4244
 2736 oracle.orig          4277
 2654 oracle.orig          4290
 2816 oracle.orig          4320
 2814 oracle.orig          4353
 2658 oracle.orig          4380
 2674 oracle.orig          7892
```

CPU - Example

```
solaris# ps -efcL | grep 2674
```

oracle	2674	1	1	19	TS	0	May	27	?	51:21	ora_lgwr_BTRW
oracle	2674	1	2	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	3	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	4	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	5	19	TS	46	May	27	?	10:51	ora_lgwr_BTRW
oracle	2674	1	6	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	7	19	TS	19	May	27	?	10:50	ora_lgwr_BTRW
oracle	2674	1	8	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	9	19	TS	50	May	27	?	10:49	ora_lgwr_BTRW
oracle	2674	1	10	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	11	19	TS	41	May	27	?	10:50	ora_lgwr_BTRW
oracle	2674	1	12	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	13	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	14	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	15	19	TS	59	May	27	?	0:00	ora_lgwr_BTRW
oracle	2674	1	16	19	TS	25	May	27	?	10:48	ora_lgwr_BTRW
oracle	2674	1	17	19	TS	60	May	27	?	10:50	ora_lgwr_BTRW
oracle	2674	1	18	19	TS	60	May	27	?	10:50	ora_lgwr_BTRW
oracle	2674	1	19	19	TS	3	May	27	?	10:49	ora_lgwr_BTRW

CPU - Example

```
solaris# dtrace -n 'profile-997hz /arg0 && curthread->t_pri != -1/
{ @[stack()] = count(); } tick-10sec { trunc(@, 10); printa(@); exit(0); }'
[...]
```

```
FJSV,SPARC64-VII`cpu_smt_pause+0x4
unix`current_thread+0x164
platmod`plat_lock_delay+0x78
unix`mutex_vector_enter+0x460
genunix`cv_timedwait_sig_hires+0x1c0
genunix`cv_waituntil_sig+0xb0
semsys`semop+0x564
unix`syscall_trap+0xac
1208
```

```
unix`disp_getwork+0xa0
genunix`disp_lock_exit+0x58
unix`disp+0x1b4
unix`swtch+0x8c
genunix`cv_wait_sig+0x114
genunix`str_cv_wait+0x28
genunix`strwaitq+0x238
genunix`kstrgetmsg+0xdcc
sockfs`sotpi_recvmsg+0x2ac
sockfs`socktpi_read+0x44
genunix`fop_read+0x20
genunix`read+0x274
unix`syscall_trap+0xac
1757
```

The D From The Previous Example

```
1  #!/usr/sbin/dtrace -s
2
3  profile-997hz
4  /arg0 && curthread->t_pri != -1/
5  {
6      @[stack()] = count();
7  }
8  tick-10sec
9  {
10     trunc(@,10);
11     printa(@);
12     exit(0);
13 }
```

Another Method for Profiling the Kernel

```
1  #!/usr/sbin/dtrace -s
2  #pragma D option quiet
3
4  profile-997hz
5  /arg0 && curthread->t_pri != -1 /
6  {
7      @ [func(caller), func(arg0)] = count();
8  }
9  tick-10sec
10 {
11     trunc(@,20);
12     printf("%-24s %-32s %-8s\n", "CALLER", "FUNCTION", "COUNT");
13     printa("%-24a %-32a %-*d\n", @);
14     exit(0);
15 }
```

Example Output

CALLER	FUNCTION	COUNT
unix`syscall_trap	genunix`sleepq_wakeall_chan	1607
genunix`fop_rwlock	genunix`fop_rwlock	1652
unix`current_thread	unix`mutex_delay_default	1667
genunix`fop_rwunlock	genunix`fop_rwunlock	1691
ip`tcp_fuse_output	ip`tcp_fuse_output	1777
genunix`str_cv_wait	genunix`rwnext	1797
unix`current_thread	ip`tcp_loopback_needs_ip	1927
0x28cdf48	unix`disp_getwork	1962
unix`_resume_from_idle	unix`_resume_from_idle	2041
unix`current_thread	unix`lock_set	2120
0x1400	ip`tcp_fuse_output	2157
unix`current_thread	unix`lock_set_spl	2292
unix`current_thread	unix`mutex_exit	2313
unix`current_thread	FJSV, SPARC64-VII`cpu_smt_pause	2656
genunix`kstrgetmsg	unix`fp_restore	2978
0x0	genunix`kstrgetmsg	3102
unix`current_thread	unix`ut10	3782
genunix`disp_lock_exit	FJSV, SPARC64-VII`copyout	4957
unix`current_thread	unix`disp_getwork	5110
	unix`mutex_enter	17420

Digging Deeper

```
solaris# dtrace -n 'fbt:unix:mutex_enter:entry'
dtrace: invalid probe specifier fbt:unix:mutex_enter:entry:
probe description fbt:unix:mutex_enter:entry does not match any probes

solaris# dtrace -l | grep mutex_enter
60103    lockstat          genunix                         mutex_enter adaptive-acquire
60104    lockstat          genunix                         mutex_enter adaptive-block
60105    lockstat          genunix                         mutex_enter adaptive-spin

solaris# dtrace -n 'lockstat:genunix:mutex_enter: { @[stack()] = count(); }'
[...]

genunix`cv_wait_sig+0x13c
genunix`str_cv_wait+0x28
genunix`strwaitq+0x238
genunix`kstrgetmsg+0xdcc
sockfs`sotpi_recvmsg+0x2ac
sockfs`socktpi_read+0x44
genunix`fop_read+0x20
genunix`read+0x274
unix`syscall_trap+0xac
1868297

ip`squeue_enter+0x10
sockfs`sostream_direct+0x194
genunix`fop_write+0x20
genunix`write+0x268
unix`syscall_trap+0xac
1893532
```

Digging Deeper

```
solaris# dtrace -n 'syscall::read:entry,syscall::write:entry  
/fds[arg0].fi_fs == "sockfs"/ { @[execname,pid] = count(); }'  
[...]  
 oracle.orig 8868 57888  
 oracle.orig 8772 57892  
 rwdoit 8724 57894  
 rwdoit 8914 57918  
 oracle.orig 8956 57920  
 oracle.orig 8872 58434  
 rwdoit 8849 58434  
 oracle.orig 9030 58511  
 rwdoit 8982 58512  
 oracle.orig 8862 58770  
 rwdoit 8816 58772  
 oracle.orig 8884 59068  
 rwdoit 8846 59068  
 oracle.orig 8778 59616  
 rwdoit 8735 59616  
 rwdoit 8909 62624  
 oracle.orig 8954 62626  
 rwdoit 8844 62776  
 oracle.orig 8874 62778
```

Digging Deeper

```
solaris# dtrace -qn 'syscall::read:entry,syscall::write:entry  
/fds[arg0].fi_fs == "sockfs"/ { @[probefunc] = sum(arg2); }  
tick-1sec { printa(@); trunc(@); }'  
  
write 34141396  
read 1832682240  
  
write 33994014  
read 1822898304  
  
write 33950736  
read 1824884640  
  
write 33877395  
read 1820879136
```

Script Version of Previous Example

```
1 #! /usr/sbin/dtrace -qs
2
3  syscall::read:entry,syscall::write:entry
4  /fds[arg0].fi_fs == "sockfs"/
5  {
6      self->flag = 1
7  }
8  syscall::read:return,syscall::write:return
9  /(int)arg0 != -1 && self->flag/
10 {
11     @[probefunc] = sum(arg0);
12 }
13 syscall::read:return,syscall::write:return
14 {
15     self->flag = 0;
16 }
```

System Metrics and DTrace

- Several DTrace providers can be used to correlate system metrics as reported by other utilities to the workload
- Notably, sysinfo for system information statistics, and vminfo for virtual memory statistics

System Metrics - Example

```
solaris# dtrace -qn 'sysinfo::: { @[probename] = count(); }  
tick-1sec { printa(@); trunc(@); }'
```

```
[...]  
rawch 1  
bwrite 2  
lwrite 3  
namei 20  
outch 20  
rw_rdfails 46  
rw_wrfails 117  
intrblk 955  
trap 1157  
lread 2140  
inv_swtch 3042  
sema 4271  
mutex_adenters 5405  
idlethread 116231  
xcalls 158512  
readch 176097  
sysread 176098  
syswrite 176882  
writech 176882  
pswitch 305054
```

System Metrics – reads and writes

```
solaris# dtrace -qn 'syscall::*read:entry,syscall::*write:entry
{ @[probefunc] = count(); } tick-1sec { printa(@); trunc(@); }'
```

pread	1
pwrite	946
read	228018
write	228043
pwrite	962
read	225105
write	225142

```
solaris# dtrace -qn 'syscall::read:entry,syscall::write:entry
{ @[probefunc, fds[arg0].fi_fs] = count(); } tick-1sec { printa(@); trunc(@); }'
```

read	proc	16
write	ufs	30
read	sockfs	225418
write	sockfs	225418

System Metrics – Another Example

```
solaris# mpstat 1
CPU minf mjf xcal  intr ithr  csw icsw migr smtx srw syscl usr sys wt idl
 0    0   0 2746  1707   48 4759  100 2464  130    7 17181   49  27  0  24
 1    0   0 2968  1808   49 5053   84 2560  115    8 18426   50  24  0  26
 2    0   0 2911  1742   50 5040   93 2610  116    7 18115   51  24  0  25
 3    0   0 2861  1729   41 4986   89 2498  110    6 18171   51  23  0  26
 4    0   0 2821  1786   50 5010   99 2527  117    7 17708   51  24  0  25
 5    0   0 2982  1958   276 4782   73 2443  104    5 17245   50  25  0  25
[...] 
```

System Metrics – mpstat(1M) xcalls

```
solaris# dtrace -n 'sysinfo:::xcalls { @[stack()] = count(); }'  
[...]  
FJSV,SPARC64-VII`send_one_mondo+0x20  
unix`xt_one_unchecked+0xc8  
genunix`sleepq_wakeall_chan+0x48  
genunix`cv_broadcast+0x4c  
ip`tcp_fuse_output+0x7f0  
ip`tcp_output+0x74  
ip`squeue_drain+0x130  
ip`squeue_enter+0x348  
sockfs`sostream_direct+0x194  
genunix`fop_write+0x20  
genunix`write+0x268  
unix`syscall_trap+0xac  
177428  
  
FJSV,SPARC64-VII`send_one_mondo+0x20  
unix`xt_one_unchecked+0xc8  
genunix`sleepq_wakeall_chan+0x48  
genunix`cv_broadcast+0x4c  
ip`tcp_fuse_output+0x7f0  
ip`tcp_output+0x74  
ip`squeue_enter+0x74  
sockfs`sostream_direct+0x194  
genunix`fop_write+0x20  
genunix`write+0x268  
unix`syscall_trap+0xac  
1177168
```

Observing Memory

- Memory consumers
 - The kernel
 - The file system cache (which is technically part of the kernel)
 - User processes
- Bundled utilities (vmstat) for getting the big picture
- With DTrace, we can
 - Track which user processes are allocating/freeing memory
 - Track kernel memory allocation and correlate to a kernel subsystem
 - Correlate VM statistics (e.g. page-in, page-out) to the workload
 - Measure memory-related latencies

Memory One-liners

Tracking memory page faults by process name:

```
dtrace -n 'vminfo:::as_fault
{ @mem[execname] = sum(arg0); }'
```

Tracking pages paged-in by process name;

```
dtrace -n 'vminfo:::pgpgin
{ @pg[execname] = sum(arg0); }'
```

Tracking pages paged-out by process name;

```
dtrace -n 'vminfo:::pgpgout
{ @pg[execname] = sum(arg0); }'
```

Tracking process user stack sizes;

```
dtrace -n 'sched:::on-cpu
{ @_ [execname] = max(curthread->t_procp->p_stksize); }'
```

Tracking which processes are growing their address space heap segment:

```
dtrace -n 'fbt:::brk:entry
{ @_ [execname] = count(); }'
```

Memory One-liners

Process allocation (via malloc()) counting requested size:

```
dtrace -n 'pid$target::malloc:entry  
{ @_[arg0] = count(); }' -p PID
```

Process allocation (via malloc()) requested size distribution plot:

```
dtrace -n 'pid$target::malloc:entry  
{ @_ = quantize(arg0); }' -p PID
```

Process allocation (via malloc()) by user stack trace and total requested size:

```
dtrace -n 'pid$target::malloc:entry  
{ @_@[ustack()] = sum(arg0); }' -p PID
```

Process allocation (via malloc()) by java stack trace and total requested size:

```
dtrace -n 'pid$target::malloc:entry  
{ @_@[jstack()] = sum(arg0); }' -p PID
```

Tracing User Memory Allocation – Mac OS X

```
macosx# dtrace -n 'syscall:::entry /pid == $target/
{ @[probefunc]=count(); }' -c ./mm
malloc of 104857600 done, touching pages...
dtrace: description 'syscall:::entry ' matched 434 probes
malloc of 104960000 done, touching pages...
malloc of 105062400 done, touching pages...
malloc of 105164800 done, touching pages...
malloc of 105267200 done, touching pages...
malloc of 105369600 done, touching pages...
malloc of 105472000 done, touching pages...
malloc of 105574400 done, touching pages...
dtrace: pid 6283 has exited
```

exit	1
mmap	7
write_nocancel	7
madvice	16

Tracing User Memory Allocation – Mac OS X

```
macosx# dtrace -n 'syscall::mmap:entry /pid == $target/
{ @[arg1] = count(); }' -c ./mm
malloc of 104857600 done, touching pages...
dtrace: description 'syscall::mmap:entry ' matched 1 probe
malloc of 104960000 done, touching pages...
malloc of 105062400 done, touching pages...
malloc of 105164800 done, touching pages...
malloc of 105267200 done, touching pages...
malloc of 105369600 done, touching pages...
malloc of 105472000 done, touching pages...
malloc of 105574400 done, touching pages...
dtrace: pid 6292 has exited
```

104960000	1
105062400	1
105164800	1
105267200	1
105369600	1
105472000	1
105574400	1

Tracing User Memory Allocation – Mac OS X

```
macosx# dtrace -qn 'syscall::mmap::entry  
/pid == $target/  
{ printf("FLAG: %x\n", arg3); }' -c ./mm  
malloc of 104857600 done, touching pages...  
[ ... ]  
FLAG: 1002  
FLAG: 1002  
[ ... ]
```

```
macosx# grep MAP_ANON /usr/include/sys/mman.h  
#define MAP_ANON 0x1000  
/* allocated from memory, swap space */
```

Tracing User Memory Allocation – Mac OS X

What Does The Kernel Do?

```
1  #!/usr/sbin/dtrace -s
2
3 #pragma D option flowindent
4
5 syscall::mmap:entry
6 {
7     self->flag = 1;
8 }
9 fbt:::
10 /self->flag/
11 {
12 }
13 syscall::mmap:return
14 /self->flag/
15 {
16     self->flag = 0;
17     exit(0);
18 }
```

Tracing User Memory Allocation – Mac OS X

What Does The Kernel Do?

```
macosx# ./mmap.d -c ./mm
malloc of 104857600 done, touching pages...
dtrace: script './mmap.d' matched 18393 probes
malloc of 104960000 done, touching pages...
CPU FUNCTION
1  -> mmap
1      -> current_map
1      <- current_map
1      -> vm_map_enter_mem_object
1          -> vm_map_enter
1              -> lock_write
1              <- lock_write
1              -> vm_map_entry_insert
1                  -> zalloc
1                      -> zalloc_canblock
1                          -> lck_mtx_lock_spin
1                          <- lck_mtx_lock_spin
1                          -> lck_mtx_unlock_darwin10
1                          <- lck_mtx_unlock_darwin10
1                              <- zalloc_canblock
1                                  <- zalloc
1                                  <- vm_map_entry_insert
1          -> lock_done
1          <- lock_done
1          -> lck_rw_done_gen
1          <- lck_rw_done_gen
1          <- vm_map_enter
1      <- vm_map_enter_mem_object
1  <- mmap
1  <= mmap
```

Tracing User Memory Allocation – Mac OS X

```
macosx# dtrace -qn 'syscall::mmap:entry /arg3 & 0x1002/ { @m[execname, arg1] = count(); }'  
^C
```

Terminal	4096	1
dtrace	4096	1
dtrace	266240	1
dtrace	4194304	2
Dock	23040	21
Dock	90112	21
WindowServer	16384	21

Tracing User Memory Allocation – Solaris

```
solaris# dtrace -n 'syscall::brk:entry { @[pid,execname] = count(); }'  
^C  
[...]  
 21008 arch 28  
 21013 arch 28  
 20958 oracle 48  
 21034 m2loader 125  
 20827 java 246  
 21035 java 332  
 21033 java 340
```

Tracing User Memory Allocation – Solaris

```
1  #!/usr/sbin/dtrace -qs
2
3  self int endds;
4
5  syscall::brk:entry
6  /pid == $target && !self->endds/
7  {
8      self->endds = arg0;
9  }
10
11 syscall::brk:entry
12 /pid == $target && self->endds != arg0/
13 {
14     printf("Allocated %d\n", arg0 - self->endds);
15     self->endds = arg0;
16 }
```

Tracing User Memory Allocation – Solaris

```
solaris# ./brk.d -c ./a.out
Allocated 16384
Allocated 8192
Allocated 8192
Allocated 8192
Allocated 8192
Allocated 8192
Allocated 8192
^C
```

Tracing User Memory Allocation – Solaris

```
solaris# dtrace -n 'pid$target::malloc:entry
{ @[pid, arg0] = count(); }' -p `pgrep -nx java`
dtrace: description 'pid$target::malloc:entry' matched 2 probes
dtrace: pid 11089 has exited
```

11089	552	1
11089	1480	1
11089	256	2

Tracing User Memory Allocation – Solaris

```
solaris# dtrace -n 'pid$target::malloc:entry
{ @[jstack()] = count(); }' -p `pgrep java | tail -1`
dtrace: description 'pid$target::malloc:entry' matched 2 probes
^C
```

```
    libc.so.1`lmalloc
    libc.so.1`fdopendir+0x22
    libc.so.1`opendir+0x3e
    libjava.so`Java_java_io_UnixFileSystem_list+0x65
0xf82bfccf8
0xeff0dc18
0xebefeaaf8
    6
```

```
    libc.so.1`lmalloc
    libc.so.1`fdopendir+0x30
    libc.so.1`opendir+0x3e
    libjava.so`Java_java_io_UnixFileSystem_list+0x65
0xf82bfccf8
0xeff0dc18
0xebefeaaf8
```

Tracing User Memory Allocation – Solaris

```
solaris# dtrace -qn 'vminfo:genunix:pageio_setup:*pgin
{ @[execname,probename] = count(); }
END { printa("%-12s %-12s %:@12d\n", @) ; } '
^C
zsched          fspgin      75
zsched          pgin        75
zsched          pgpgin     75
m2loader        fspgin     686
m2loader        pgin       686
m2loader        pgpgin     686
```

Tracing User Memory Allocation – Solaris

```
solaris# dtrace -qn 'vmunix:genunix:pageio_setup:*pgin
/ execname == "m2loader" / { @[stack(),ustack()] = count(); }'
^C

. . .

nfs`nfs4_getapage+0x1c1
nfs`nfs4_getpage+0xe2
genunix`fop_getpage+0x47
genunix`segvn_fault+0x8b0
genunix`as_fault+0x205
unix`pagefault+0x8b
unix`trap+0x3d7
unix`_cmntrap+0x140

libmysqlclient.so.16.0.0`adler32+0x685
libmysqlclient.so.16.0.0`read_buf+0x62
libmysqlclient.so.16.0.0`fill_window+0x969
libmysqlclient.so.16.0.0`deflate_slow+0x1ff
libmysqlclient.so.16.0.0`deflate+0x82d
m2loader`read_zstream+0x1a3
m2loader`get_filecont+0x2a0
m2loader`store_files_data+0xb9c
m2loader`main+0x463
m2loader`0x403b3c
198
```

Tracing User Memory Allocation – Solaris

```
1  #!/usr/sbin/dtrace -s
2
3  #pragma D option quiet
4
5  fbt:nfs:nfs4_getapage:entry
6  /execname == "m2loader"/
7  {
8      self->st = timestamp;
9      @calls = count();
10 }
11 fbt:nfs:nfs4_getapage:return
12 /self->st/
13 {
14     @mint = min(timestamp - self->st);
15     @maxt = max(timestamp - self->st);
16     @avgt = avg(timestamp - self->st);
17     @t["ns"] = quantize(timestamp - self->st);
18     self->st = 0;
19 }
20 END
21 {
22     normalize(@mint, 1000);
23     normalize(@maxt, 1000);
24     normalize(@avgt, 1000);
25     printf("%-8s %-8s %-8s %-8s\n", "CALLS", "MIN(us)", "MAX(us)", "AVG(us)");
26     printa("%-@8d %-@8d %-@8d %-@8d\n", @calls, @mint, @maxt, @avgt);
27     printf("\n");
28     printa(@t);
29 }
```

Tracing User Memory Allocation – Solaris

```
solaris# ./nfs.d
^C
CALLS      MIN(us)    MAX(us)    AVG(us)
79742        1          21454       42
```

Tracking VM Events - Solaris

```
1 #!/usr/sbin/dtrace -s
2
3 #pragma D option quiet
4
5 vminfo:::
6 {
7     @[execname, probefunc, probename] = count();
8 }
9 tick-1sec
10 {
11     trunc(@, 10);
12     printf("%-16s %-16s %-16s %-8s\n", "EXEC", "FUNCTION",
13           "NAME", "COUNT");
14     printa("%-16s %-16s %-16s %-.@8d\n", @);
15     trunc(@);
16 }
```

Tracking VM Events - Solaris

```
solaris# ./vmtop10.d
```

EXEC	FUNCTION	NAME	COUNT
tnslsnr	as_fault	prot_fault	687
arch	as_fault	as_fault	1014
oracle	page_reclaim	pgfrec	1534
oracle	page_reclaim	pgrec	1534
oracle	anon_private	cow_fault	1555
tnslsnr	as_fault	as_fault	1598
java	anon_zero	zfod	6354
oracle	anon_zero	zfod	6382
java	as_fault	as_fault	6489
m2loader	as_fault	as_fault	7332
oracle	as_fault	as_fault	16443

EXEC	FUNCTION	NAME	COUNT
java	page_reclaim	pgrec	49
run_m3loader	as_fault	as_fault	58
arch	anon_private	cow_fault	60
uname	as_fault	as_fault	120
tds_job_status	as_fault	as_fault	134
arch	as_fault	as_fault	229
java	anon_zero	zfod	2778
java	as_fault	as_fault	2907
oracle	anon_zero	zfod	3904
oracle	as_fault	as_fault	4512

Tracking Kernel Memory Allocations

```
solaris# dtrace -n 'fbt::kmem_cache_alloc:entry
{ @[args[0]->cache_name] = sum(args[0]->cache_bufsize); }'
dtrace: description 'fbt::kmem_cache_alloc:entry' matched 1 probe
^C
[...]
vn_cache                                1469760
streams_dblk_80                          1803648
streams_dblk_208                         1859520
HatHash                                   3276800
zio_buf_131072                           3407872
anon_cache                               3625920
kmem_alloc_1152                           3867264
kmem_alloc_192                           4345344
hment_t                                    5134272
streams_dblk_20304                         5267328
kmem_alloc_32                            5487296
kmem_alloc_64                            9930112
zio_cache                                 100685360
kmem_alloc_4096                           269684736
```

Tracking Kernel Memory Allocations

```
solaris# dtrace -n 'fbt::kmem_cache_alloc:entry
/args[0]->cache_name == "kmem_alloc_4096"/ { @[stack()] = count(); }'
dtrace: description 'fbt::kmem_cache_alloc:entry' matched 1 probe
^C
[...]
genunix`kmem_alloc+0x70
genunix`exec_args+0xe5
elfexec`elf32exec+0x3db
genunix`gexec+0x218
genunix`exec_common+0x917
genunix`exece+0xb
unix`sys_syscall32+0x101
144

genunix`kmem_zalloc+0x3b
genunix`anon_set_ptr+0xc9
genunix`anon_dup+0x83
genunix`segvn_dup+0x51c
genunix`as_dup+0xf8
genunix`cfork+0x661
genunix`fork1+0x10
unix`sys_syscall+0x17b
257

genunix`kmem_alloc+0x70
genunix`segvn_fault_anonpages+0x177
genunix`segvn_fault+0x23a
genunix`as_fault+0x205
unix`pagefault+0x8b
unix`trap+0x3d7
unix`_cmntrap+0x140
59978
```

Where are the Pagefaults Coming From?

```
solaris# dtrace -n 'fbt:unix:pagefault:entry { @[execname] = count(); }'  
dtrace: description 'fbt:unix:pagefault:entry' matched 1 probe  
^C  
[...]  
      tds_job_status                               2380  
      arch                                         3778  
      tnslsnr                                      5909  
      m2loader                                     8581  
      oracle                                       35603  
      java                                         53396
```

Back To Kernel Memory...

```
solaris# dtrace -n 'fbt:::vmem_alloc:entry
{ @[args[0]->vm_name] = sum(arg1); }'
dtrace: description 'fbt:::vmem_alloc:entry' matched 1 probe
^C
```

crypto	2
tl_minor_space	2
contracts	3
ip_minor_arena_la	47
ip_minor_arena_sa	79
bp_map	1040384
kmem_oversize	1314320
kmem_firewall_va	1343488
segkp	1400832
kmem_io_4G	20303872
heap	26734592

Back To Kernel Memory...

```
solaris# dtrace -n 'fbt:::vmem_alloc:entry /args[0]->vm_name == "heap"/ { @[stack()] = count(); }'
dtrace: description 'fbt:::vmem_alloc:entry' matched 1 probe
^C
[...]
    unix`segkmem_xalloc+0x144
    unix`segkmem_alloc_io_4G+0x26
    genunix`vmem_xalloc+0x315
    genunix`vmem_alloc+0x155
    unix`kalloc+0x160
    unix`i_ddi_mem_alloc+0xd6
    rootnex`rootnex_setup_copybuf+0xe4
    rootnex`rootnex_bind_slowpath+0x2dd
    rootnex`rootnex_coredma_bindhdl+0x16c
    rootnex`rootnex_dma_bindhdl+0x1a
    genunix`ddi_dma_buf_bind_handle+0xb0
    sata`sata_dma_buf_setup+0x4b9
    sata`sata_scsi_init_pkt+0x1f5
    scsi`scsi_init_pkt+0x44
    sd`sd_setup_rw_pkt+0xe5
    sd`sd_initpkt_for_buf+0xa3
    sd`sd_start_cmds+0xa5
    sd`sd_core_iostart+0x87
    sd`sd_mapblockaddr_iostart+0x11a
    sd`sd_xbuf_strategy+0x46
    259

    unix`ppmapin+0x2f
    zfs`mappedread+0x84
    zfs`zfs_read+0x10e
    zfs`zfs_shim_read+0xc
    genunix`fop_read+0x31
    genunix`read+0x188
    genunix`read32+0xe
    unix`sys_syscall32+0x101
271
```

I/O

- Disk and network I/O are typically the largest latency bubbles in an application/workflow
- DTrace makes it possible to
 - Determine which processes/threads are generating IO load
 - The nature of the load
 - The latency of the IOs

I/O One-liners

Which processes are executing common I/O system calls:

```
dtrace -n 'syscall::*read:entry,  
syscall::*write:entry  
{ @rw[execname,probefunc] = count(); }'
```

Which file system types are targeted for reads and writes:

```
dtrace -n 'syscall::*read:entry,  
syscall::*write:entry  
{ @fs[execname, probefunc, fds[arg0].fi_fs] = count(); }
```

Which files are being read, and by which processes:

```
dtrace -n 'syscall::*read:entry  
{ @f[execname, fds[arg0].fi_pathname] = count(); }'
```

Which files are being written, and by which processes:

```
dtrace -n 'syscall::*write:entry  
{ @f[execname, fds[arg0].fi_pathname] = count(); }'
```

I/O One-liners

Which processes are generating network I/O (Solaris):

```
dtrace -n 'fbt:sockfs::entry
{ @[execname, probefunc] = count(); }'
```

Which processes are generating file system I/O (Solaris):

```
dtrace -n 'fsinfo:::
{ @fs[execname, probefunc] = count(); }'
```

What is the rate of disk I/O being issued:

```
dtrace -n 'io:::start
{ @io = count(); }
tick-1sec {
printa("Disk I/Os per second: %d\n", @io);
trunc(@io); }'
```

I/O by system call

```
1  #!/usr/sbin/dtrace -s
2
3  #pragma D option quiet
4
5  syscall:::entry
6  {
7      @ [execname, probefunc] = count();
8  }
9  END
10 {
11     trunc(@, 10);
12     printf("%-16s %-16s %-8s\n",
13            "EXEC", "SYSCALL", "COUNT");
14     printa("%-16s %-16s %-@8d\n", @);
```

System Calls to Determine I/O

```
solaris# ./sctop10.d
```

```
^C
```

	SYSCALL	COUNT
EXEC	stat	1127
java	getuid	1168
sge_shepherd	sigaction	1272
arch	ioctl	1599
dtrace	brk	1606
m2loader	read	1808
arch	close	2152
sge_shepherd	close	2197
java	lseek	6183
java	read	6388

System Calls to Determine I/O

```
1 #!/usr/sbin/dtrace -s
2
3 #pragma D option quiet
4
5 syscall::*:read*:entry,
6 syscall::*:write*:entry
7 {
8     @ [execname, probefunc, fds[arg0].fi_fs]
9         = count();
10 }
11 END
12 {
13     trunc(@, 10);
14     printf("%-16s %-16s %-8s %-8s\n",
15             "EXEC", "SYSCALL", "FS", "COUNT");
16     printa("%-16s %-16s %-8s %d\n", @);
17 }
```

System Calls to Determine I/O

```
solaris# ./scrwtop10.d
```

```
^C
```

	SYSCALL	FS	COUNT
EXEC			
Xvnc	read	sockfs	671
oracle	pwrite	zfs	1080
arch	read	lofs	1188
mysqld	read	sockfs	1385
oracle	write	sockfs	2295
oracle	read	sockfs	2322
java	write	nfs4	4538
java	read	sockfs	5630
java	read	zfs	15703
java	read	lofs	29359

I/O by FS Operation

```
solaris# ./fstop10.d
```

```
^C
```

EXEC	FS FUNC	COUNT
oracle	fop_write	14494
java	fop_readlink	16410
java	fop_seek	28161
oracle	fop_rwlock	28551
oracle	fop_rwunlock	28613
java	fop_access	32845
java	fop_read	59105
java	fop_rwlock	60449
java	fop_rwunlock	60449
java	fop_lookup	110724

I/O by FS Operation

```
1 #!/usr/sbin/dtrace -qs
2
3 fsinfo:::
4 {
5     @ [execname, probename,
6      args[0]->fi_fs, args[0]->fi.pathname] = count();
7 }
8 END
9 {
10    trunc(@,10);
11    printf("%-16s %-8s %-8s %-32s %-8s\n",
12          "EXEC", "FS FUNC", "FS TYPE", "PATH", "COUNT");
13    printa("%-16s %-8s %-8s %-32s %-.8d\n", @);
```

I/O by FS Operation

```
solaris# ./fstop10_enhanced.d
^C
EXEC          FS FUNC  FS TYPE   PATH                                COUNT
java          lookup  ufs      /var                               39
java          lookup  ufs      /var/webconsole                   39
java          lookup  ufs      /var/webconsole/domains           39
java          lookup  ufs      /var/webconsole/domains/console    39
java          lookup  ufs      /usr/share/webconsole/webapps     70
java          lookup  ufs      /usr                           88
java          lookup  ufs      /usr/share                      88
java          lookup  ufs      /usr/share/webconsole                 88
fsflush       inactive tmpfs   /tmp/out1                         1706
fsflush       putpage  tmpfs   /tmp/out1                         1706
```

Looking at Disk I/O

```
1 #!/usr/sbin/dtrace -Cs
2
3 #pragma D option quiet
4
5 #define PRINT_HDR printf("%-8s %-16s %-8s %-16s\n", "RPS", "RD BYTES", "WPS", "WR BYTES");
6
7 dtrace:::BEGIN
8 {
9     PRINT_HDR
10 }
11
12 io:::start
13 /execname == $$1 && args[0]->b_flags & B_READ/
14 {
15     @rps = count();
16     @rbytes = sum(args[0]->b_bcount);
17 }
18
19 io:::start
20 /execname == $$1 && args[0]->b_flags & B_WRITE/
21 {
22     @wps = count();
23     @wbytes = sum(args[0]->b_bcount);
24 }
25 tick-1sec
26 {
27     printa("%-@8d %-@16d %-@8d %-@16d\n", @rps, @rbytes, @wps, @wbytes);
28     trunc(@rps); trunc(@rbytes); trunc(@wps); trunc(@wbytes);
29 }
30 tick-1sec
31 /x++ == 20/
32 {
33     PRINT_HDR
34     x = 0;
35 }
```

Looking at Disk I/O

```
solaris# ./disk_io.d java
RPS          RD BYTES          WPS          WR BYTES
6112        50069504        9363        76701696
5873        48111616        9482        77676544
5920        48496640        9303        76210176
5943        48685056        9345        76554240
5939        48652288        9210        75448320
5885        48209920        9264        75890688
6045        49520640        9192        75300864
5975        48947200        9415        77127680
5973        48930816        9305        76226560
^C
4808        39387136        7583        62119936
```

Looking at Disk I/O

```
1  #!/usr/sbin/dtrace -s
2
3  #pragma D option quiet
4
5  dtrace:::BEGIN { trace("Tracing...Output after 10 seconds, or Ctrl-C\n"); }
6
7  io:::start
8  {
9      start[args[0]->b_edev, args[0]->b_blkno] = timestamp;
10 }
11
12 io:::done
13 /start[args[0]->b_edev, args[0]->b_blkno]/
14 {
15     this->elapsed =
16     (timestamp - start[args[0]->b_edev, args[0]->b_blkno]) / 1000000;
17     @iot[args[1]->dev_statname,
18     args[0]->b_flags & B_READ ? "READS(ms)" : "WRITES(ms)"] =
19     quantize(this->elapsed);
20     start[args[0]->b_edev, args[0]->b_blkno] = 0;
21 }
22 tick-10sec
23 {
24     printa(@iot);
25     exit(0);
26 }
```

Looking at Disk I/O

```
solaris# ./iotimeq.d
```

ssd153

A histogram titled "Distribution" showing the count of reads for different time intervals. The x-axis is labeled "READS (ms)" and the y-axis is labeled "value". The distribution is right-skewed, with most reads falling between 8 and 16 ms.

value	Distribution	count
0		0
1		4
2		10
4	@@	146
8	@@@@@@@	889
16	@@@@@@@	1946
32	@@@	338
64		0

ssd148

		WRITES (ms)
value	Distribution	count
1		0
2		5
4	@	75
8	@@@@	368
16	@@@@@@@@	1336
32	@@@@@@@@	1424
64	@	97
128		0

Looking at Network

```
1  #!/usr/sbin/dtrace -s
2
3  #pragma D option quiet
4
5  fbt:sockfs::entry
6  {
7      @ [execname, probefunc] = count();
8  }
9  END
10 {
11     printf("%-16s %-24s %-8s\n", "EXEC", "SOCKFS FUNC",
12           "COUNT");
13     printa("%-16s %-24s %-*d\n", @);
14 } 0
```

Looking at Network

```
solaris# ./sock.d
^C
EXEC          SOCKFS FUNC           COUNT
gnome-panel   socktpi_ioctl        1
m2loader      getsonode          1
. . .
oracle        so_update_attrs     4855
java          so_lock_read_intr   4984
java          so_unlock_read     4984
java          socktpi_read       4984
java          sotpi_recvmsg      4984
java          so_update_attrs     7525
```

Looking at Network

```
solaris# dtrace -n 'fbt:sockfs:so_update_attrs:entry /execname == "java"/
{ @[stack()] = count(); }'
dtrace: description 'fbt:sockfs:so_update_attrs:entry' matched 1 probe
^C
```

```
sockfs`socktpi_read+0x32
genunix`fop_read+0x31
genunix`read+0x188
genunix`read32+0xe
unix`sys_syscall32+0x101
    3
```

```
sockfs`socktpi_write+0x161
genunix`fop_write+0x31
genunix`write+0x287
unix`sys_syscall+0x17b
    195
```

```
sockfs`sendit+0x17d
sockfs`send+0x6a
unix`sys_syscall+0x17b
    1126
```

```
sockfs`socktpi_read+0x32
genunix`fop_read+0x31
genunix`read+0x188
unix`sys_syscall+0x17b
    2481
```

Summary

- DTrace empowers users to observe their systems and workloads in unprecedented ways
- Problems that used to take days or weeks to root-cause can be diagnosed in hours
- DTrace facilitates fast theory testing and drill-down analysis
- As with any other tool, growth and knowledge comes from experience and use
- Controlled, known experiments are an excellent method for learning DTrace
- Quoting Bryan Cantrill, *DTrace is a workhorse, not a showhorse* – put it to work!