

Time For Changes



Thomas Nau

Thomas.Nau@uni-ulm.de

kiz

University of Ulm



Infrastructure Department

- 25 people in 2 teams installing, operating, ...
 - cluster based university wide backup-, mail-, directory-, portal-, fileservices, databases, administrative computing and more
 - compute services for Ulm and the much larger university in Stuttgart; several dozen scientific apps
 - LAN, MAN, WLAN based computer networks
 - phone system including VoIP installations
 - HelpDesk



2+ Years Solaris 10 in Production



4 HPC server
E6800, 24x U-III+
480GB RAM
2.5TB FC-AL disks



SysAdmins
Blade 1000



Mail/IMAP
LDAP
WWW
Oracle DB
NFS/Samba server
several V240/440



several V240
SunRay servers



SOHO router
EPIA VIA board



web cache V100
news server V100



firewalls V65x



Quick “Show Hands” On Solaris 10 Usage



Glowing Wires



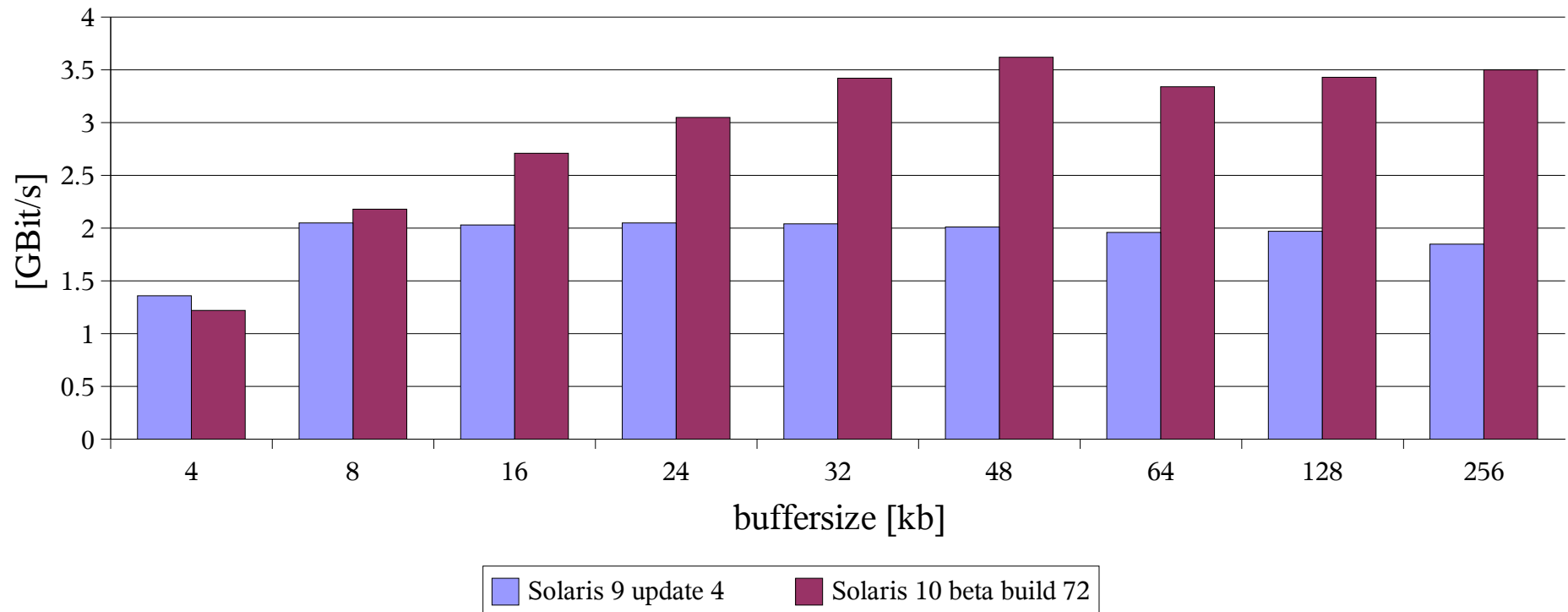
Performance

- increased network performance due to
 - redesigned network stack; better rates for connection establishment and close down
 - support for hardware checksums
 - zero-copy for *ce* and *bge* drivers; supported by *sendfile(3ext)*
 - changed defaults: multidata-transmit “on” by default



Performance

TCP loopback single-thread bulk transfer performance (V240)





NFSv4

- predecessors made use of 4+ ports
 - nfsd, nfs_acl: 2049
 - lockd: 4045
 - mountd: ≥ 1 port handled by *rpcbind(1m)*
 - statd: ≥ 1 port handled by *rpcbind(1m)*
- v4 uses 2049 only
 - firewall friendly
 - allows for *ssh(1)* tunneling



NFSv4 *ssh(1)* Tunnel Setup

```
# log into Solaris 10 server
#
root@alderaan:~> ssh -L 2049:borg:2049 root@borg
root@borg's password:
Last login: Sun Mar 20 19:02:49 2005 from alderaan.solaris-fans.net

root@borg# share
-          /export      root=borg      "NFSv4 Test"

# FreeBSD 5.3 client
# start mapping daemon and mount filesystem
#
root@alderaan:~> idmapd -d rz.uni-ulm.de

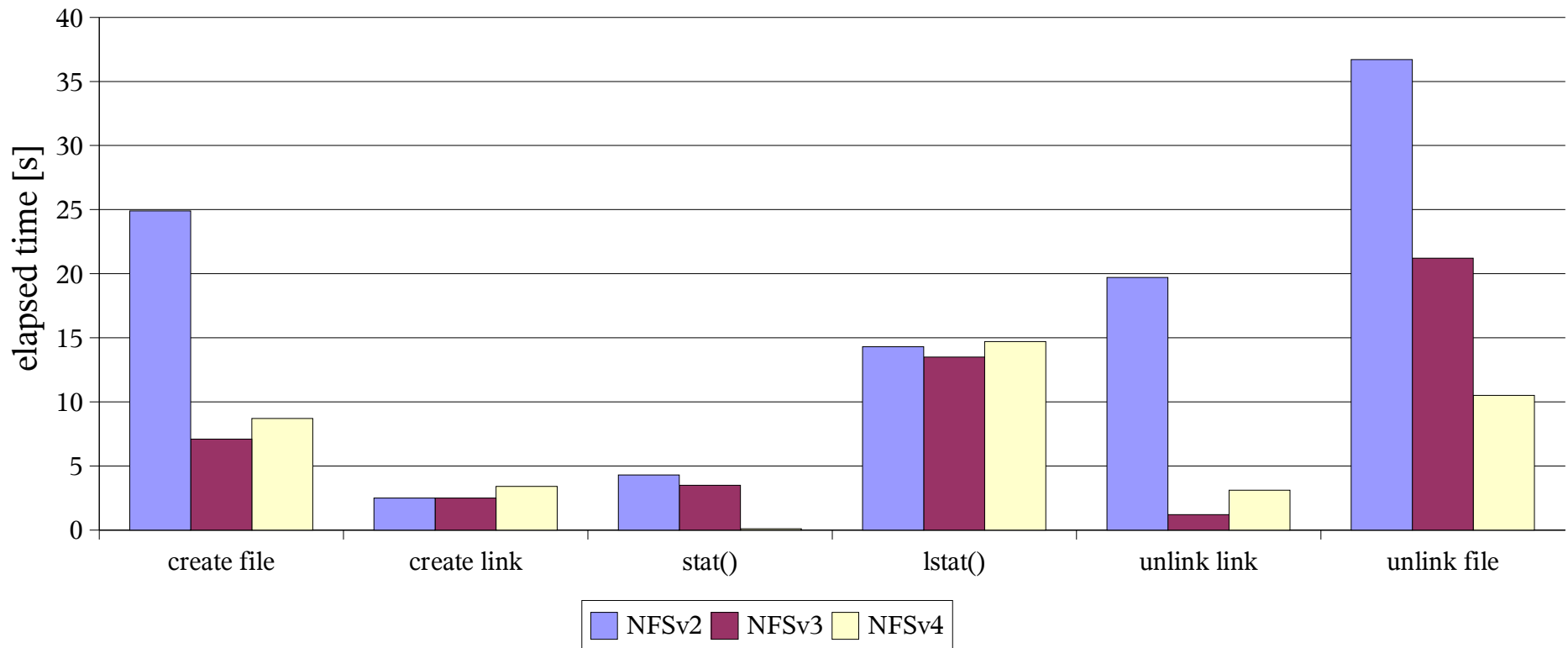
root@alderaan:~> mount_nfs4 localhost.rz.uni-ulm.de:/tmp /mnt

root@alderaan:~> ls -l /mnt
-rw-----  1 nau   jedis   80243 Mar 19 09:27 battleplan.pdf
-rw-----  1 leia  senat  670800 Mar 16 12:03 deathstar_scheme.pdf
drwx-----  2 nau   jedis    512 Mar 19 10:52 droids
-r--r----- 1 darth evils  480 Mar 11 10:26 empire_rules
```



NFSv4

NFS fileoperation benchmark (code courtesy Jarod Jenson)





More Enhancements

- UFS logging “on” by default for all filesystems (yes, this means “/” too)
- logging UFS improves or exceeds non-logging UFS performance
 - test run internally by Sun
 - may improve NFS server performance too
 - range from a few percent to a magnitude



Wrap-Up

- no performance penalty for new features
- our most noticeable gains in networking stack are
 - for busy web or web-caching servers
 - and firewalls
- network performance of low-end machines not limited by processor anymore but by link speed



Wrap-Up

- NFSv4 delivers equal to NFSv3 performance even so protocol is more complex
- speed-up expected when v4 specific functions such as “compound operations” kick in
- allows for good MS Windows® support due to ID mapping now based on strings
- single config file with appropriate defaults
- will “rock” even more in combination with ZFS



Crime Scene Investigations



The “Old Way” Of Debugging

- *truss(1)*, *mdb(1)* or *pstack(1)*
- not dynamic; problems with timing or transient errors
- probing affects the target
- different tools for different problems



Improvements But No Real Solution

- *pfiles(1)* lists filenames
- *cputrack(1m)* and *cpustat(1m)* can be made available to all users by granting *cpc_cpu* privilege
- thread level support added for
 - *truss(1)*
 - *pflags(1)*
 - *pstack(1)*
- *pstack(1)* supports bundled *Java* release



The New Era: DTrace

- 40.000+ probes distributed over kernel and modules
 - can be enabled independently with almost no overhead
- provides scripting, interpreted in kernel context
- tools already built on top:
 - *plockstat(1m)*, *er_kernel(1)* (part of Sun Studio 10)
 - could easily replace others such as *truss(1)*
- access restricted using *Least Privileges*
 - you may grant user *dtrace_{kernel, proc, user}* privileges



DTrace Example: IO Statistics

```
#!/usr/sbin/dtrace -s

#pragma D option quiet

io:::start
{
    @[args[1]->dev_statname, args[2]->fi_name, execname] =
        sum(args[0]->b_bcount);
}

END
{
    printf("%10s %20s %20s %15s\n", "DEVICE", "FILE", "APP", "BYTES");
    printa("%10s %20s %20s %15@d\n", @);
}
```



DTrace Example: IO Statistics

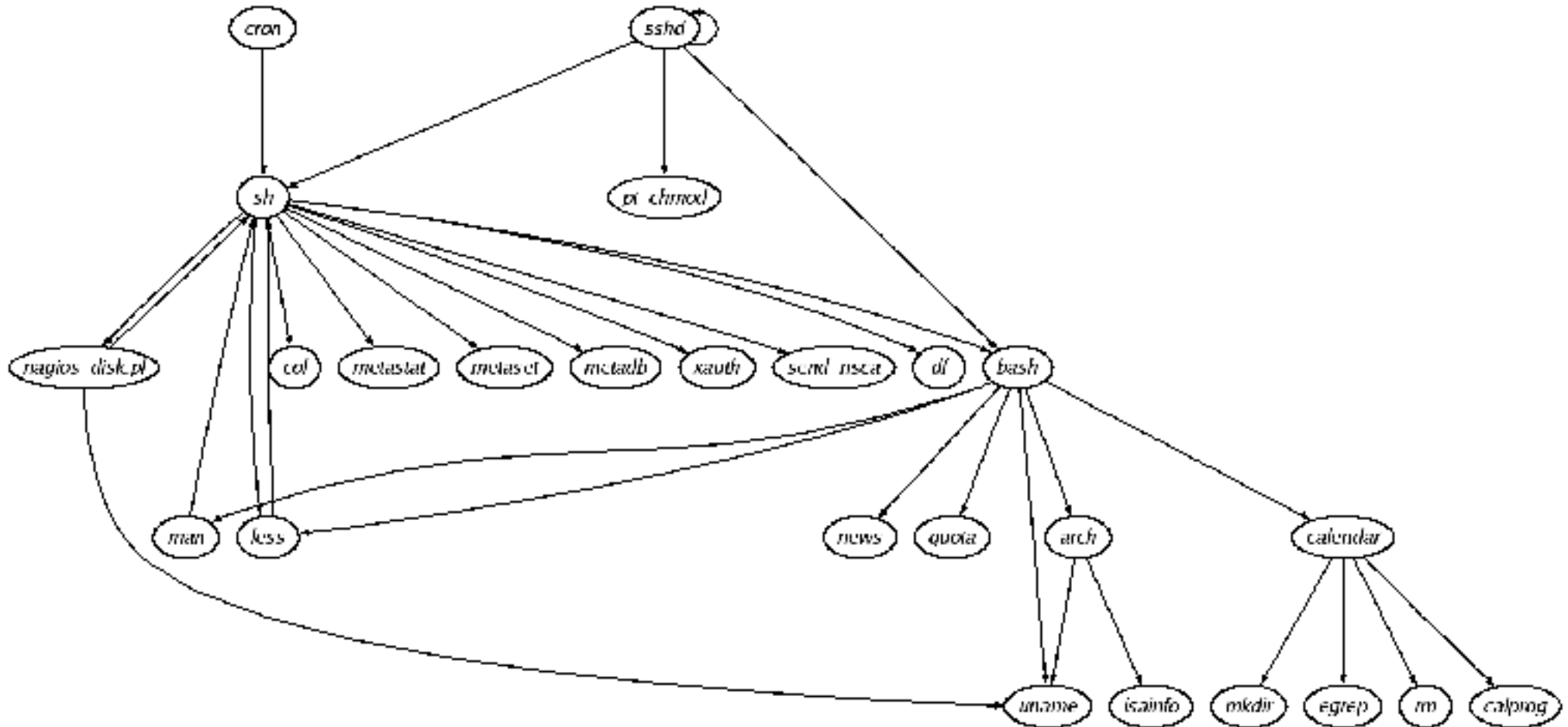
```
root@borg# ./fileio.d
```

```
^C
```

DEVICE	FILE	APP	BYTES
...			
ssd6	<none>	imapd	1130496
md16384	<none>	imapd	1145856
ssd1	<none>	imapd	1146880
ssd6	wordlist.db	bogofilter	2195456
md16384	wordlist.db	bogofilter	4661248
ssd1	wordlist.db	bogofilter	4661248
ssd6	cyrus.cache.NEW	imapd	6136832
ssd1	cyrus.cache.NEW	imapd	6136832
md16384	cyrus.cache.NEW	imapd	6136832

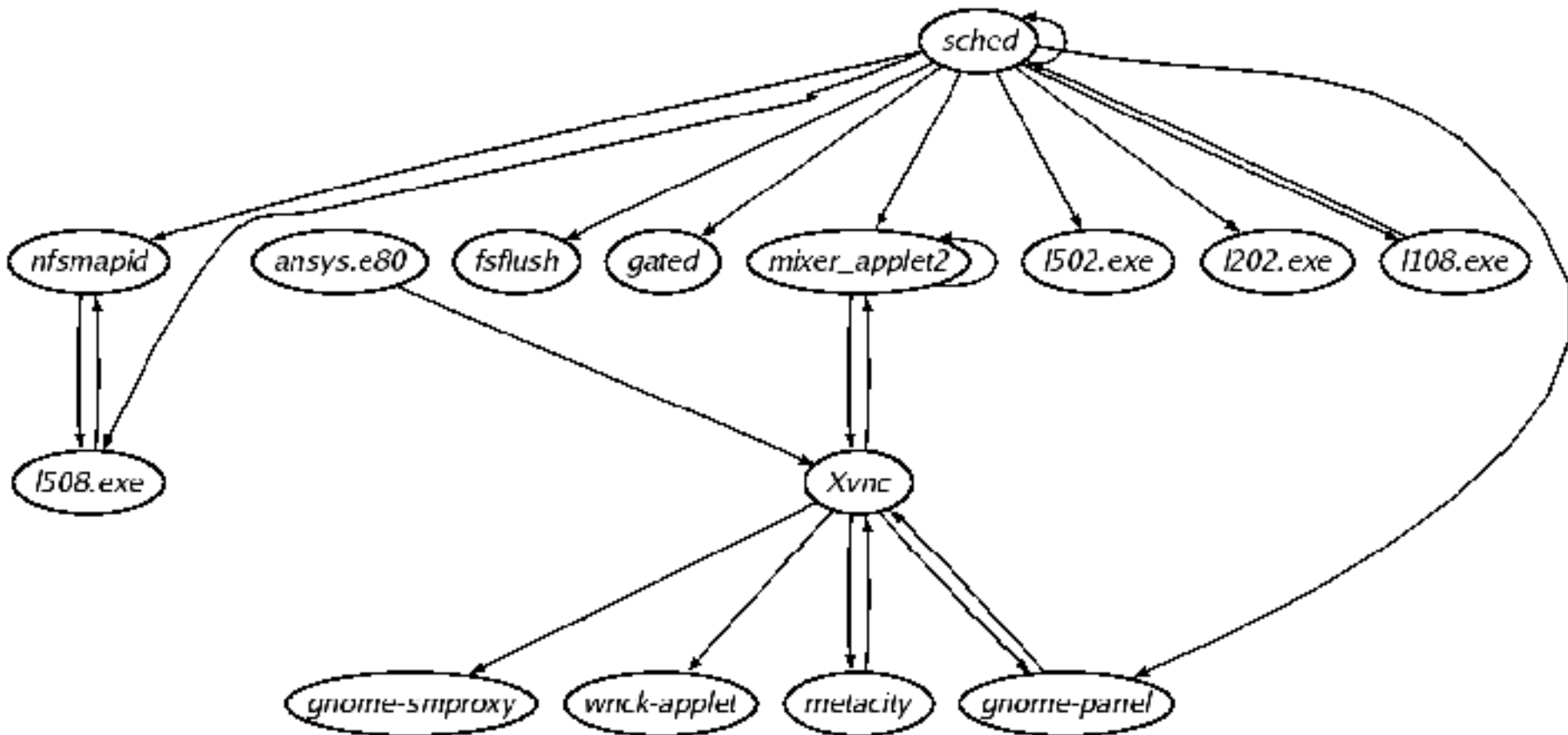


DTrace Example: Execution Paths





DTrace Example: Wakeup Graph



Time For Changes



File View Timeline Help

Find Text:

Functions Callers Callee Source Lines Disassembly

CPU Cycles (C)	KCPU Cycles (C)	KCPU Cycles (C)	Name
3,038	0,170	10,450	syscall_trap32
0,002	59,192	59,092	<Total>
0,001	0,104	0,214	syscall_trap

Data for Selected Object:

Name: read
 PC Address: 4: 0x01097178
 Size: 1.16
 Source File: (unknown)
 Object File: file io.er/archives/gemmix
 Load Object: <gemmix>
 Mangled Name:
 Aliases:

Process Times (sec.) / Counts

	Exclusive	Inclusive
KCPU Cycles:	0.056 (12705.683)	3.031 (512907.1)
"count":	4112675000000	2271600000000

0.055 0.155 9.031 read

2,808	0,115	2,818	ufs_read
0,057	0,103	0,010	socket_read
0,031	0,100	0,114	getc
0,033	0,121	0,056	releasef
0,010	0,101	0,111	io_exit
0,000	0,111	0,011	top_block
0,005	0,109	0,018	uto_block
0,004	0,104	0,014	top_read
0,004	1,192	1,032	mutex_enter
0,004	0,110	0,010	nbl_need_check
0,004	0,102	0,112	sv_inter
0,003	0,	0,013	fifo_read
0,003	0,110	0,010	top_block
0,003	0,103	0,013	io_read



Wrap-Up

- most powerful and outstanding **system analysis tool**; a real *Swiss Army Knife* useful to everyone
- start with *mpstat(1m)* or *vmstat(1m)* to identify the potential performance problem area, then drill down using *DTrace*
- don't forget about DTrace's speculation feature



Wrap-Up

- lots of examples in
 - */usr/demo/dtrace*
 - *<http://www.brendangregg.com/dtrace.html>*
 - *<http://www.solarisinternals.com/si/dtrace/>*



Health Insurance



Predictive Self-Healing

- *Solaris Fault Manager*
 - monitors devices gathering telemetry information
 - correlates events, evaluates possible problems based on fault trees and tags them with universal unique ID
 - possible actions include
 - logging and messaging
 - disable components such as off-lining CPUs or page retirement
 - UUID information available at <http://www.sun.com/msg>
 - see *fmadm(1m)*, *fmdump(1m)* and *fmstat(1m)*



Predictive Self-Healing

dumping the event log

```
root@borg# fmdump -av
TIME                UUID                SUNW-MSG-ID
May 31 12:54:40.7100 c5018a8c-76da-cf9f-dca6-a071d8216be3 SUN4U-8000-1A
  100%  fault.memory.page
        FRU: mem:///component=/N0/SB3/P2/B1/D2,J15501
        rsrc: mem:///pa=62321a4000/component=/N0/SB3/P2/B1/D2,J15501

# lookup faulty components
#
root@borg# fmadm faulty -a
STATE RESOURCE / UUID
-----
faulted mem:///pa=62321a4000/component=/N0/SB3/P2/B1/D2,J15501
        c5018a8c-76da-cf9f-dca6-a071d8216be3
-----
```



Predictive Self-Healing

- *Service Management Facility, smf(5)*
 - makes OS aware of a “service” instead of a process by grouping and knowing about dependencies
 - provides monitoring, restarting and centralized management of services
 - “killed” process will be restarted automatically
 - already replaced most scripts in `/etc/rc?.d/` but general functionality kept for compatibility
 - significantly changed *inetd(1m)* behavior
 - does not process `/etc/inet/inetd.conf` anymore



Predictive Self-Healing

debugging *syslogd(1m)* startup problem

```
root@borg# svcs -x system-log
svc:/system/system-log:default (system log)
  State: maintenance since Thu May 31 18:39:17 2005
  Reason: Start method exited with $SMF_EXIT_ERR_CONFIG.
    See: http://sun.com/msg/SMF-8000-KS
    See: syslogd(1M)
    See: /var/svc/log/system-system-log:default.log
  Impact: This service is not running.

# check log (does not provide more details)
#
root@borg# tail /var/svc/log/system-system-log:default.log
...
[ May 31 18:39:17 Executing start method ("/lib/svc/method/system-log") ]
[ May 31 18:39:17 Method "start" exited with status 96 ]
```



Wrap-Up

- improved fault handling and history of problems
- boot time to console prompt greatly reduced due to silent parallel launching of services
- nice centralized administrative approach and property handling
- **pitfalls** (caused us the only severe downtime)
 - SysAdmin must change way of working (e.g. “kill”)
 - changed *inetd(1m)* may break software installation

Time For Changes



No More
“All Or Nothing”



Least Privilege Model

- currently 48 privileges defined
- all granted by default to superuser for backwards compatibility but can be withdrawn if required
- 5 privileges, e.g. *proc_exec*, define basic group which is granted to everyone
- implemented as 4 sets and represented as strings
- every application gets the privileges it requires to get the job done **but no more**



Least Privilege Model

Example

```
# get shell PID and assigned privileges
#
nau@borg:~> ppriv $$
7847:    -bash
flags = <none>
        E: basic
        I: basic
        P: basic
        L: all

# effective set
# inheritable set
# permitted set
# limit set

# debugging
#
nau@borg:~> ppriv -e -D cat /etc/shadow
cat[12529]: missing privilege "file_dac_read" (euid = 3201,
        syscall = 225) needed at ufs_access+0x3c
cat: /etc/shadow: Permission denied
```



Least Privilege Model

Example continued

```
# grant shell additional privileges
# and try again
#
root@borg# ppriv -s EP+file_dac_read 7847

nau@borg:~> ppriv $$
7847:    -bash
flags = <none>
        E: basic,file_dac_read
        I: basic,file_dac_read
        P: basic,file_dac_read
        L: all

nau@borg:~> ppriv -e -D cat /etc/shadow
root:WONT_SHOW_YOU_THE_HASH:6445:::::
daemon:*LK*:12714:::::
bin:*LK*:12714:::::
...
```



Zones

- similar basic concept as found in FreeBSD jails
- single kernel instance only
 - very efficient
 - single NFS server, *IPfilter* or routing table only
 - cannot boot from NFS server (well, not officially)
- isolation of applications in non-global zones
 - restricted access limiting e.g. *snoop(1m)* or *dtrace(1m)*
 - applies to UID 0 in local zone too



Zones

- *Solaris Containers*: zones tightly integrated with resource management
 - use *Dynamic Resource Pools* and *FSS(7)*
 - two level approach possible
 - *DRP* for dynamic inter-zone CPU resource allocation
 - *FSS(7)* for projects within each individual zone
 - CPUs are virtualized if *DRP* are used e.g. *psrinfo(1m)*
- patch and package handling is aware of zones
- **painful update a migration**: re-install is required



Wrap-Up

- significantly improved out-of-the-box security
- improved job training quality for our trainees as each gets “full” access to own zone
- allow for quick-setup of efficient staging, testing and hosting environment
 - important **psychological** effect of granting “root” access
- zones can implement “DMZ in a box” for firewalls
- no *DTrace* or *privileges(5)* in local zones



Conclusion

Time For Changes



USE THE FORCE!



Update to Solaris 10 now