What’s New in Solaris 10?

Leon Towns-von Stauber, Occam’s Razor
Seattle SAGE Group, March 2006
http://www.occam.com/

Contents

Introduction

- Solaris 10 has now been released for about a year
- Lots of changes; this presentation only covers some of the highlights
  - You can find a more comprehensive list at:
    - http://www.sun.com/software/solaris/whats_new.jsp

Legal Notices

- This presentation Copyright © 2006 Leon Towns-von Stauber. All rights reserved.
- Trademark notices
    - Sun™, Solaris™, OpenSolaris™, and other terms are trademarks of Sun Microsystems. See http://www.sun.com/suntrademarks/.
    - Other trademarks are the property of their respective owners.

OpenSolaris

OpenSolaris - Intro

- Solaris 10 binaries free for download from Sun
- OpenSolaris (dev branch of Solaris 10) source code free for download from http://www.opensolaris.org/os/
- Governed by Common Development and Distribution License (CDDL), based on Mozilla Public License
  - Recently Jonathan Schwartz mused on the possibility of adding GPL
- OpenSolaris is based on the released version of Solaris 10, and is the basis for future versions of Solaris
  - Actively developed by Sun engineers as well as external volunteers

OpenSolaris - Distributions

- Some OpenSolaris-based products
  - Nexenta (http://www.gnusolaris.org/gswiki)
    - Includes many GNU and other open source packages
    - Uses Debian package manager
  - BeleniX (http://belenix.sarovar.org/belenix_home.html)
    - Developed in Bangalore
  - Schillix (http://schillix.berlios.de/)
    - First OpenSolaris distro
  - Genesi (http://www.genesippc.com/)
    - OpenSolaris for PowerPC

Service Management Facility

SMF - Intro

- Solaris Service Manager part of Predictive Self Healing
- Replacement for `inittab`, `rc` scripts, and `inetd`
  - `inittab` much simpler in Solaris 10 (only 4 lines)
- Features
  - Automatic process restart
  - Dependency management
  - Parallel startup
  - Built-in TCP Wrapper support (including `rpcbind`)
  - And more!
- http://www.sun.com/bigadmin/content/selfheal/smf-quickstart.html

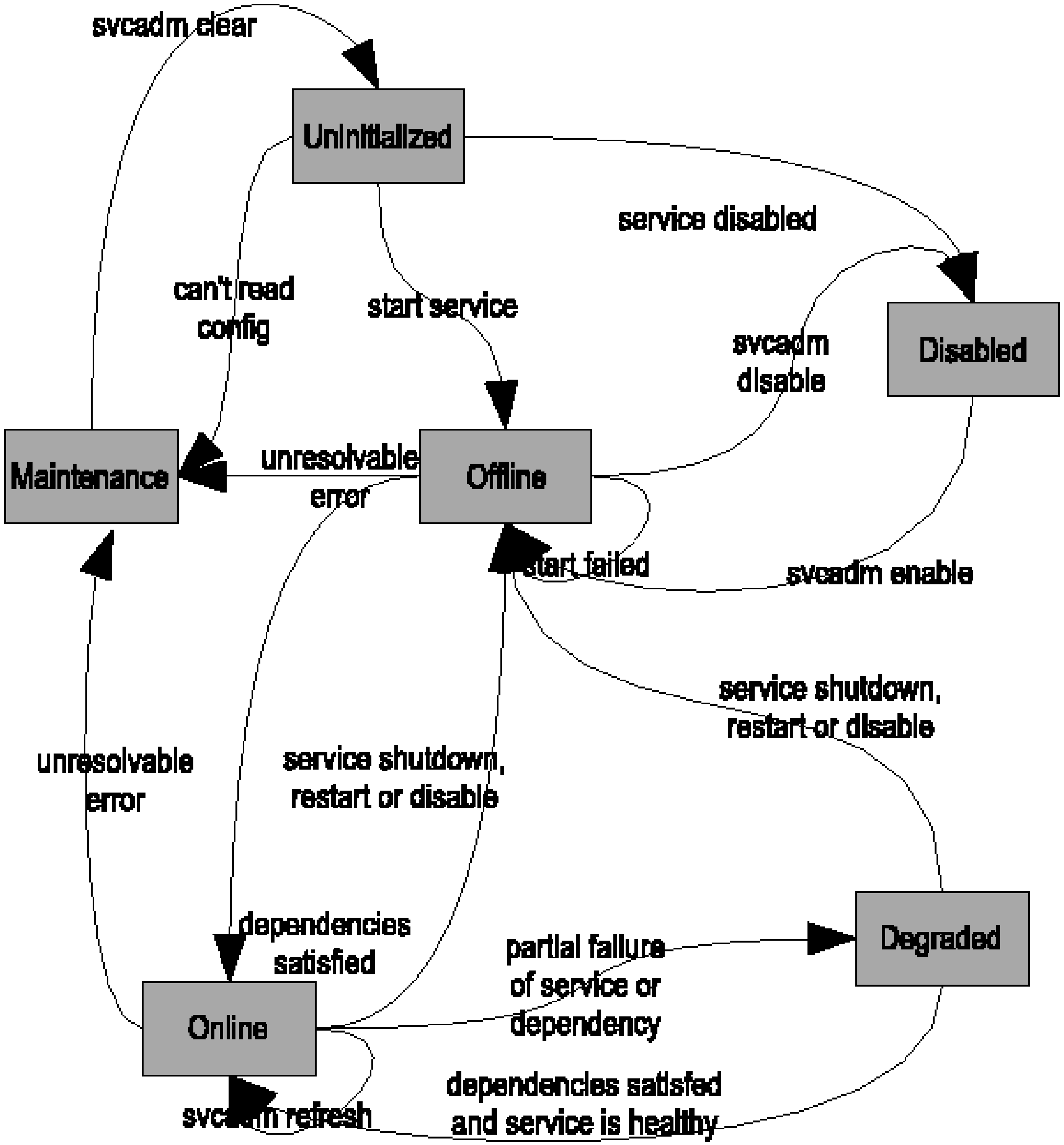
SMF - Daemons

- `init` starts `svc.startd` (and restarts if necessary)
- `svc.startd` starts `svc.configd`, `inetd`, and most services
- `inetd` is now a backward-compatible near-peer of `svc.startd`
    - Starts and restarts traditional `inetd`-based services, while `svc.startd` handles everything else

SMF - States

- Each service is in one of seven states
    - Uninitialized - prior to processing
    - Offline - enabled, but not running
    - Online - enabled and running
    - Degraded - enabled and running, but with degraded functionality for some reason
    - Maintenance - enabled, but not running due to fault that cannot be repaired automatically
    - Disabled - administratively disabled
    - Legacy-Run - still managed by `init` scripts; SMF lists these, but can give no further state information

SMF - States



svcadm clear

Uninitialized

service disabled

can't read config

start service

svcadm disable

Disabled

Maintenance

unresolvable error

Offline

start failed

svcadm enable

service shutdown, restart or disable

unresolvable error

service shutdown, restart or disable

dependencies satisfied

partial failure of service or dependency

Degraded

Online

dependencies satisfied and service is healthy

svcadm refresh

Service States and Transitions

SMF - FMRIs

- Each service is identified with a unique Fault Management Resource Identifier (FMRI), which includes a category, the service provided, and the name of the service instance
- Examples
  - `svc:/system/system-log:default`
  - `svc:/system/filesystem/local:default`
  - `svc:/milestone/single-user:default`
  - `svc:/network/smtp:sendmail`
  - `lrc:/etc/rc3_d/S81volmgmt`
- Fortunately, unique abbreviations work when specifying FMRIs, such as `smtp` or `sendmail`

SMF - Files

- Config files
  - `/var/svc/manifest/`*`category`*`/`*`service`*`.xml`
    - Usually managed indirectly by calling `svccfg`
  - `/lib/svc/method/`*`script`*
    - Startup script
- Log files
  - `/var/svc/log/`*`fmri`*`.log`
  - `/etc/svc/volatile/`*`fmri`*`.log`
    - Startup log (not very interesting)

SMF - Commands

- `svcs` - List services, with state and time of last state change
- `svcs -a` - List all services, including disabled
- `svcs -l` *`FMRI...`* - List information about service
- `svcs -d` *`FMRI...`* - List services on which service depends
- `svcs -D` *`FMRI...`* - List services which depend on service
- `svcs -p` *`FMRI...`* - List processes associated with service
- `svcs -x [`*`FMRI...`*`]` - Display explanation for service state (usually to explain reason for degraded or maintenance state)

SMF - Commands

- `svcadm enable` *`FMRI`*`...`
- `svcadm disable` *`FMRI`*`...`
- `svcadm restart` *`FMRI`*`...`
- `svcadm clear` *`FMRI`*`...` - Clear degraded or maintenance state, attempt to start normally
- `inetadm` - List `inetd` services
- `inetadm -e` *`FMRI`*`...` - Enable service
- `inetadm -d` *`FMRI`*`...` - Disable service
- `inetadm -l` *`FMRI`*`...` - List service properties
- `inetadm -m` *`FMRI`*`...` *`name=value`*`...` - Modify service properties

SMF - Commands

- `svccfg` - Interactive mode
- `svccfg archive` - Dump the full configuration of all managed services to standard output
- `svccfg import` *`filename`* - Bring service described by specified XML manifest under SMF management
- `svccfg delete` *`FMRI`* - Delete service configuration
- `svccfg -s` *`FMRI`* `listprop` - List service properties
- `svccfg -s` *`FMRI`* `setprop` *`name=value`* - Modify service property

SMF - Examples

```
% svcs
STATE          STIME    FMRI
legacy_run     Dec_11   lrc:/etc/rc2_d/S10lu
legacy_run     Dec_11   lrc:/etc/rc2_d/S20sysetup
legacy_run     Dec_11   lrc:/etc/rc2_d/S72autoinstall
legacy_run     Dec_11   lrc:/etc/rc2_d/S73cachefs_daemon
legacy_run     Dec_11   lrc:/etc/rc2_d/S89PRESERVE
legacy_run     Dec_11   lrc:/etc/rc2_d/S95networker
legacy_run     Dec_11   lrc:/etc/rc2_d/S98deallocate
legacy_run     Dec_11   lrc:/etc/rc2_d/S99audit
legacy_run     Dec_11   lrc:/etc/rc3_d/S81volmgt
online         Dec_11   svc:/system/svc/restarter:default
online         Dec_11   svc:/network/pfil:default
online         Dec_11   svc:/network/loopback:default
online         Dec_11   svc:/network/physical:default
online         Dec_11   svc:/milestone/network:default
online         Dec_11   svc:/system/identity:node
online         Dec_11   svc:/system/metainit:default
online         Dec_11   svc:/system/filesystem/root:default
online         Dec_11   svc:/system/filesystem/usr:default
[...]
```

SMF - Examples

```
% svcs -l syslog-ng
fmri         svc:/system/syslog-ng:default
name         syslog-ng server
enabled      true
state        online
next_state   none
state_time   Fri Feb 24 00:01:14 2006
logfile      /var/svc/log/system-syslog-ng:default.log
restarter    svc:/system/svc/restarter:default
contract_id  16547
dependency   require_all/none svc:/milestone/sysconfig (online)
dependency   require_all/none svc:/system/filesystem/local (online)
dependency   optional_all/none svc:/system/filesystem/autofs
(disabled)
dependency   require_all/none svc:/milestone/name-services (online)
dependency   require_all/restart file://localhost/opt/local/etc/
syslog-ng.conf (online)
```

SMF - Examples

```
% svcs -p syslog-ng
STATE          STIME    FMRI
online          0:01:14 svc:/system/syslog-ng:default
                0:01:14    26747 syslog-ng
                0:01:14    26748 sh
                0:01:14    26749 sh
                0:01:14    26751 sh
                0:01:14    26753 sh
                0:01:14    26754 sh
                0:01:14    26755 sh
                0:01:14    26762 sec
                0:01:14    26765 sec
                0:01:14    26767 sec
                0:01:14    26768 sec
                0:01:14    26769 sec
                0:01:14    26771 sec
```

SMF - Examples

```
% inetadm
ENABLED   STATE          FMRI
disabled  disabled       svc:/network/rpc/gss:default
disabled  disabled       svc:/network/rpc/mdcomm:default
disabled  disabled       svc:/network/rpc/meta:default
disabled  disabled       svc:/network/rpc/metamed:default
disabled  disabled       svc:/network/rpc/metamh:default
disabled  disabled       svc:/network/rpc/rex:default
[...]
disabled  disabled       svc:/network/login:eklogin
disabled  disabled       svc:/network/login:klogin
disabled  disabled       svc:/network/login:rlogin
disabled  disabled       svc:/network/rexec:default
disabled  disabled       svc:/network/shell:default
disabled  disabled       svc:/network/shell:kshell
disabled  disabled       svc:/network/talk:default
enabled   online         svc:/network/rpc/smserver:default
disabled  disabled       svc:/application/print/rfc1179:default
disabled  disabled       svc:/network/rpc-100235_1/
rpc_ticotsord:default
```

SMF - Examples

```
% inetadm -l shell:default
SCOPE    NAME=VALUE
         name="shell"
         endpoint_type="stream"
         proto="tcp6only,tcp"
         isrpc=FALSE
         wait=FALSE
         exec="/usr/sbin/in.rshd"
         user="root"
default  bind_addr=""
default  bind_fail_max=-1
default  bind_fail_interval=-1
default  max_con_rate=-1
default  max_copies=-1
default  con_rate_offline=-1
default  failrate_cnt=40
default  failrate_interval=60
default  inherit_env=TRUE
default  tcp_trace=TRUE
default  tcp_wrappers=TRUE
```

```
% inetadm -p
NAME=VALUE
bind_addr=""
bind_fail_max=-1
bind_fail_interval=-1
max_con_rate=-1
max_copies=-1
con_rate_offline=-1
failrate_cnt=40
failrate_interval=60
inherit_env=TRUE
tcp_trace=TRUE
tcp_wrappers=TRUE
```

SMF - Examples

```
% svccfg -s syslog-ng listprop
milestone                         dependency
milestone/entities                fmri     svc:/milestone/sysconfig
milestone/grouping                astring  require_all
milestone/restart_on              astring  none
milestone/type                    astring  service
filesystem                        dependency
filesystem/entities               fmri     svc:/system/filesystem/local
filesystem/grouping               astring  require_all
filesystem/restart_on             astring  none
filesystem/type                   astring  service
[...]
start                             method
start/exec                        astring  /lib/svc/method/syslog-ng
start/timeout_seconds             count    600
start/type                        astring  method
[...]
refresh                           method
refresh/exec                      astring  ":kill -HUP"
refresh/timeout_seconds           count    60
refresh/type                      astring  method
tm_common_name                    template
tm_common_name/C                  ustring  "syslog-ng server"
tm_man_syslog-ng                  template
tm_man_syslog-ng/manpath          astring  /opt/local/man
tm_man_syslog-ng/section          astring  8
tm_man_syslog-ng/title            astring  syslog-ng
```

SMF - Examples

```
% cat /var/svc/manifest/system/syslog-ng.xml
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">

<service_bundle type='manifest' name='PMSslog:syslog'>

<service
        name='system/syslog-ng'
        type='service'
        version='1'>

        <create_default_instance enabled='true' />

        <single_instance/>

        <dependency
                name='milestone'
                grouping='require_all'
                restart_on='none'
                type='service'>
                <service_fmri value='svc:/milestone/sysconfig' />
        </dependency>
[...]
```

SMF - Examples

```
% cat /var/svc/manifest/system/syslog-ng.xml (cont’d.)
[...]
        <!--
          syslogd(1M) can log to non-root local directories.
        -->
        <dependency
                name='filesystem'
                grouping='require_all'
                restart_on='none'
                type='service'>
                <service_fmri value='svc:/system/filesystem/local' />
        </dependency>
[...]
        <!--
                The system-log start method includes a "savecore -m".
                Use an appropriately long timeout value.
        -->
        <exec_method
                type='method'
                name='start'
                exec='/lib/svc/method/syslog-ng'
                timeout_seconds='600' />
[...]
        <exec_method
                type='method'
                name='refresh'
                exec=':kill -HUP'
                timeout_seconds='60' />
[...]
```

SMF - Examples

```
% cat /var/svc/manifest/system/syslog-ng.xml (cont’d.)
[...]
        <property_group name='general' type='framework'>
                <!-- to start stop syslog daemon -->
                <propval name='action_authorization' type='astring'
                        value='solaris.smf.manage.syslog-ng' />
        </property_group>

        <stability value='Unstable' />

        <template>
                <common_name>
                        <loctext xml:lang='C'>
                        syslog-ng server
                        </loctext>
                </common_name>
                <documentation>
                        <manpage title='syslog-ng' section='8'
                                manpath='/opt/local/man' />
                </documentation>
        </template>
</service>

</service_bundle>
```

SMF - Examples

```
% cat /lib/svc/method/syslog-ng
#!/sbin/sh

DAEMON=/opt/local/sbin/syslog-ng
USER=syslog
CONFFILE=/opt/local/etc/syslog-ng.conf
PIDFILE=/var/run/syslog-ng.pid

echo 'syslog-ng service starting.'

# Before syslogd starts, save any messages from previous crash dumps so that
# messages appear in chronological order.
/usr/bin/savecore -m
if [ -r /etc/dumpadm.conf ]; then
        . /etc/dumpadm.conf
        [ -n "$DUMPADM_DEVICE" -a "x$DUMPADM_DEVICE" != xswap ] && \
                /usr/bin/savecore -m -f $DUMPADM_DEVICE
fi

$DAEMON -u $USER -f $CONFFILE -p $PIDFILE
```

SMF - Procedures

- Some suggested system setup procedures
- Enable DNS (not always enabled by default)
  - `svcadm enable dns/client`
- Enable NTP
  - `svccfg import /var/svc/manifest/network/ntp.xml`
  - `svcadm enable ntp`
- Disable unnecessary network services
  - `svcadm disable nisplus autofs nfs ...`

SMF - Procedures

- Set default parameters for `inetd` services
  - `inetadm -M tcp_trace=TRUE`
  - `inetadm -M tcp_wrappers=TRUE`
- Enable accounting
  - `svcadm enable sar`
  - `crontab -e sys`, uncomment `sar` jobs

Basic Audit Reporting Tool

BART - Intro

- BART is a file integrity checker
  - Like Tripwire
  - For each file, stores size, permissions, ownership, mod time, and an MD5 hash of the contents
- Right after OS load, get an initial snapshot
  - `bart create > bart_manifest.initial`
  - Keep it somewhere safe from modification
- Compare manifests to look for unplanned discrepancies (possibly the result of intruder actions)
  - `bart create > bart_manifest.2006-02-28`
  - `bart compare bart_manifest.initial bart_manifest.2006-02-28`

BART - Manifest

```
% cat bart_manifest.initial
! Version 1.0
! Wednesday, July 27, 2005 (14:36:30)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/-i F 0 100644 user::rw-,group::r--,mask:r--,other:r-- 42e7fd5e 0 0
d41d8cd98f00b204e9800998ecf8427e
/.rhosts L 9 120777 - 42e7fd4f 0 0 /dev/null
/.shosts L 9 120777 - 42e7fd52 0 0 /dev/null
/.ssh/authorized_keys F 818 100600 user::rw-,group::---,mask:---,other:--- 42af545c 0
40 6a8955607dee81922482664241b16d55
/.ssh/prng_seed F 1024 100600 user::rw-,group::---,mask:---,other:--- 42e7f317 0 0
1e04d1b9eff896531c3c52e630b47587
/.sunw/pkcs11_softtoken/objstore_info F 103 100600
user::rw-,group::---,mask:---,other:--- 42dd93b3 0 0 46f1a97d295cd9e3342518b2416dd2a0
/bin L 9 120777 - 42dd8ed5 0 0 ./usr/bin
/cdrom/cdrom0 L 20 120777 - 42dec372 0 60001 ./sol_10_305_sparc_3
/core F 8401969 100600 user::rw-,group::---,mask:---,other:--- 42deb733 0 0
c3408654d417bb230a306614ae281057
/dev/.devfsadm_daemon.lock F 0 100644 user::rw-,group::r--,mask:r--,other:r-- 42dec14e
0 0 d41d8cd98f00b204e9800998ecf8427e
[...]
```

BART - Comparison

```
% bart compare bart_manifest.initial bart_manifest.2006-02-28
/.ssh/prng_seed:
  mtime  control:42e7f317  test:439c8a55
  contents  control:1e04d1b9eff896531c3c52e630b47587
test:a92c9005c1019117cf3c41964c5723b0
/core:
  delete
/dev/.devfsadm_dev.lock:
  mtime  control:42decab6  test:439c8a38
  contents  control:ad09238337ad5f5aa1d2aae04af6d849  test:
919f0e2671e55c474253ef9546f4df23
/dev/.devlink_db:
  mtime  control:42e7f0d1  test:439c8a3c
  contents  control:5cacb03566d008110ecf2b204fb25b4b
test:f584d78592590fc8e11f4de3692a3dbd
/etc/.pwd.lock:
  add
/etc/coreadm.conf:
  mtime  control:42e7f0ca  test:439c8a36
[...]
```

Password Management

Passwords - Hashing Algorithms

- Alternate hashing algorithms introduced in Solaris 9
  - Linux & *BSD-compatible MD5 and Blowfish, in addition to standard UNIX `crypt` (DES)
- Listed in `/etc/security/crypt.conf`
- Configured in `/etc/security/policy.conf`
  - For example, change `CRYPT_DEFAULT` from `__unix__` (`crypt`) to `1` (Linux/BSD MD5)
    - Change your password, and your hash goes from this:
      - `e1KPbn9iJYCPA`
    - to this:
      - `$1$9VJEDOoi$djFLClN9L3adytQklAn3f.`

Passwords - Checks

- Configurable checks on new passwords
- Before, could only specify minimum length
- Now have:
  - More sophisticated complexity checking (length, character types, etc.)
  - Checks against password history (previously used password
  - Checks against password dictionary
    - Use `mkpwdict` to build up a password dictionary
- Configured in `/etc/default/passwd`

Passwords - Disabling

- Distinction between locked and no-login accounts
  - Locked: Password hash is `*LK*`, all access denied (include key-based SSH authentication, cron, etc.)
    - `passwd -l` *`username`*
  - No-login: Password hash is `NP`, password-based logins denied but other account uses are available
    - `passwd -N` *`username`*

ZFS

ZFS - Intro

- Brand new filesystem from Sun, not an iteration of UFS
- Currently released with OpenSolaris, not part of Solaris 10 until later this year
    - Even then, no support for booting from ZFS until later
- End the Suffering -- Data management should be:
    - Simple
    - Powerful
    - Safe
    - Fast

ZFS - Intro

- Design objectives
    - Simple administration
    - End-to-end data integrity
    - High performance
    - High capacity
- Major design elements
    - Pooled storage
    - Advanced checksumming
    - Transactional operation
    - Copy-on-write

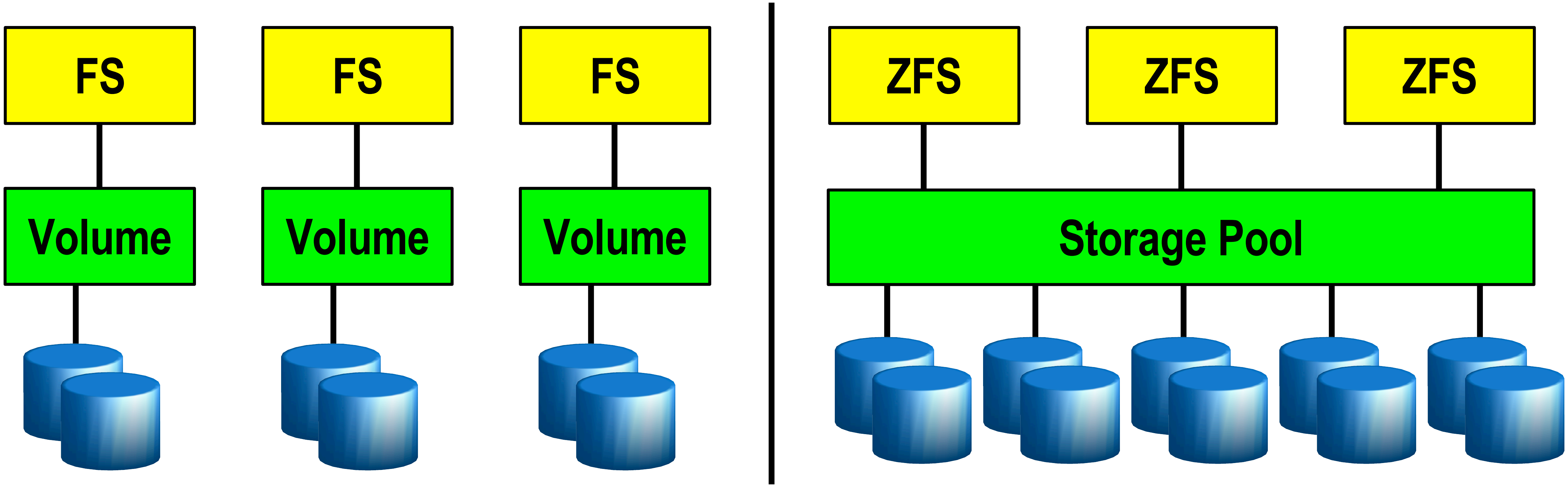
ZFS - Pooled Storage

- Modern volume management grew up as a stepwise extension of simple disk management
  - In the beginning, you had a filesystem on a disk
  - Need more space, more bandwidth, more reliability
  - Simplest next step: Keep filesystem management the same, combine multiple physical disks into logical volumes, try to hide complexity of underlying physical implementation from the filesystem
    - Filesystems are harder to write than volume managers

ZFS - Pooled Storage

- Volume management possesses inherent problems
  - Storage mainly allocated by hand, using complex toolsets
  - Storage fragmented into volume groups, logical volumes
  - Filesystem bandwidth limited by particulars of underlying configuration
- In a ZFS storage pool:
  - No partitions to allocate, grow, or shrink
  - All storage is shared, used as needed
  - All disk bandwidth available all the time
  - Easily managed ways to impose limits if needed

ZFS - Pooled Storage



Volume Management vs. Pooled Storage

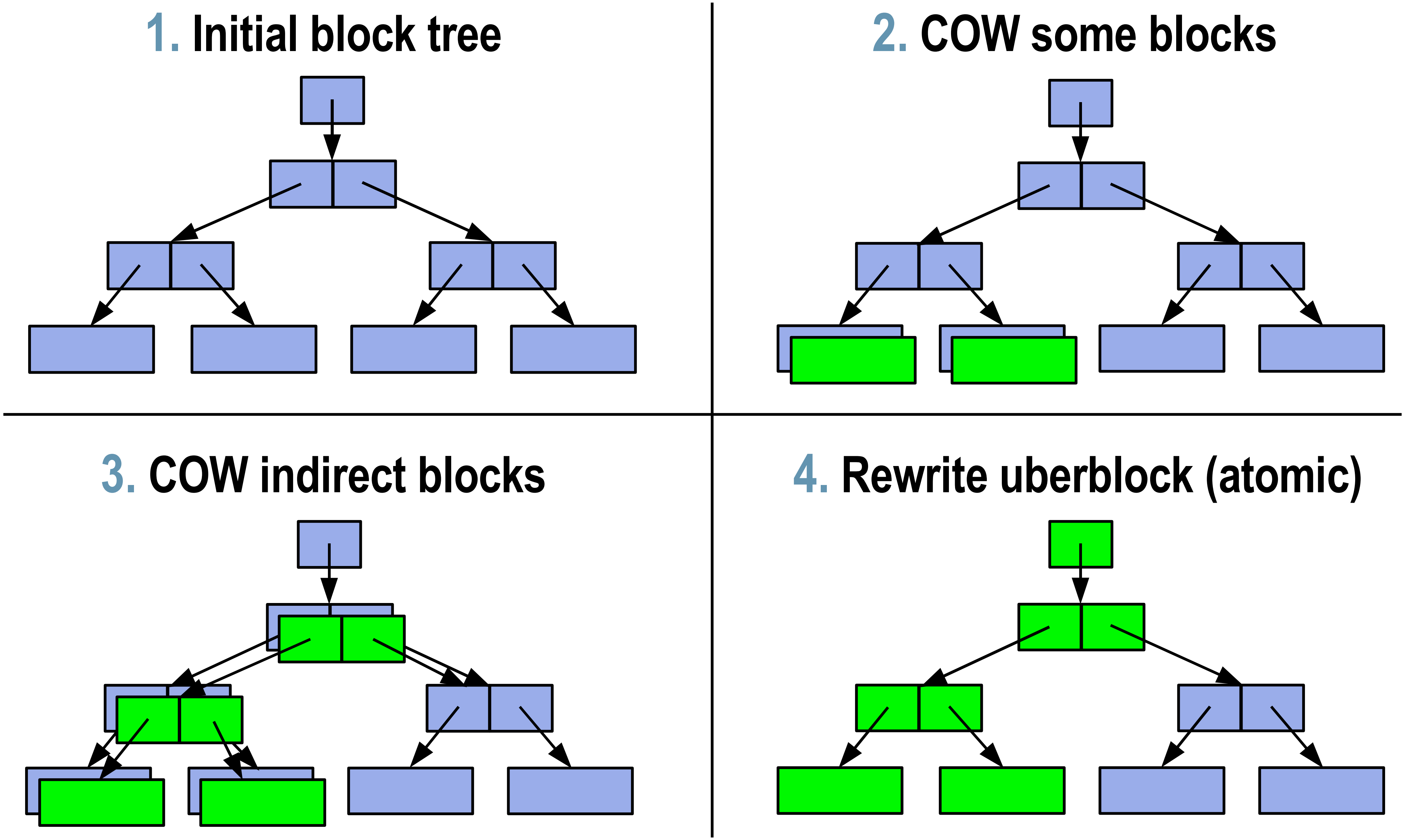
ZFS - Transactions

- Traditional filesystems write data block-by-block
  - Power loss during write leads to loss of consistency
    - Journaling can work around some of this, but adds complexity and performance hit to filesystem
- ZFS writes complete transactions
  - Writes are all-or-nothing
  - Filesystem always in a consistent state, no need for journaling
  - Writes are aggregated into single transactions for improved performance

ZFS - Copy-on-write

- Live data is never overwritten
  - New data written to unused spot on disk
  - New parent indirect blocks written, pointing to new data
  - Finally, uberblock pointers switched over (atomic change)
- On-disk state always valid
  - Changes don’t take effect until transaction is complete and pointers switched
- Snapshots are easy
  - Keep old blocks around, with old & new uberblock
  - Taking snapshots actually easier than not (no need to free old blocks)

ZFS - Copy-on-write



1. Initial block tree

2. COW some blocks

3. COW indirect blocks

4. Rewrite uberblock (atomic)

Copy-on-write Procedure

ZFS - Data Integrity

- Traditionally, each data block has a checksum
  - Notices unexpected change after write (bit rot)
  - However, there are many ways to write data with valid block checksums, but that completely mess up the filesystem
    - Example: Accidental overwrite of existing data has valid block checksum, but creates filesystem inconsistency
- ZFS checksum is in parent indirect block, not with data
  - Indirect blocks have checksums in their parent blocks, all the way up to the uberblock
  - Checksums in the uberblock validate the entire tree (thus, the entire storage pool)

ZFS - Data Integrity

- Self-validating nature of ZFS checksumming lets you tell which disk in a mirrored pair has the good data, in case one has suffered corruption
  - You can then copy good blocks from one disk to the other (self-healing data)
  - Disk mirrors in ZFS can heal data in background with periodic scrubbing, before inconsistencies are encountered during regular operation
  - Same functionality used to resilver a mirrored pair

ZFS - Scalability

- ZFS is a 128-bit filesystem
  - Capacity of 256 septillion terabytes
    - Exceeds the quantum information storage capacity of all atoms on Earth
- No limits on numbers of files, directories, etc.
  - No inodes

ZFS - Performance

- Copy-on-write results in all writes being sequential
- Dynamic striping over all disks in storage pool maximizes use of bandwidth
- Supports multiple block sizes, automatically chosen by workload
- Pipelined I/O, intelligent prefetch, parallel operations, etc.

ZFS - Administration

- ZFS filesystems are not space allocations, but control points
  - Can set quotas or reservations to control space usage
- Make as many as you want!
  - E.g., one per user: different quotas, different privileges, and a lot faster to run `df` than `du`
- Creating filesystems under another filesystem inherits properties of parent as defaults; can manage large numbers of filesystems via parent-child relationships
- Mounting and NFS-sharing filesystems done within ZFS; no need for entries in `vfstab` or `dfstab`
- Everything is done online

ZFS - Administration

- Example: Create mirrored pool, create and mount home filesystem, change mount point, create user home directory, set a quota, export home directories, add space to pool
  - `zpool create poolA mirror c0t0d0 c1t0d0`
  - `zfs create poolA/home`
  - `zfs set mountpoint=/export/home poolA/home`
  - `zfs create poolA/home/user1`
  - `zfs set quota=10g poolA/home/user1`
  - `zfs set sharenfs=rw poolA/home`
  - `zpool add poolA mirror c2t0d0 c3t0d0`

ZFS - Potpourri

- Supports NTFS-style ACLs
- ZFS has undergone frequent, brutal test procedures at Sun
  - Over a million forced, violent crashes without loss of data integrity
- Interesting statistics: number of lines of code in Solaris implementations of UFS and ZFS
  - UFS: 86,953 lines
    - With volume manager: 324,854 lines
  - ZFS: 71,312 lines

ZFS - Resources

- Much more to ZFS
- http://www.opensolaris.org/os/community/zfs/
- *SysAdmin* 2/06 and 3/06
    - "The Best File System in the World?", Peter Baer Galvin
    - Also online at samag.com

Containers

Containers - Intro

- Solaris Containers is a term for the combination of Solaris Zones and Resource Management
- Solaris Zones is a method of system virtualization
  - Zones have distinct user process spaces and system configurations (networking, user accounts, etc.)
  - However, all zones share the same OS kernel
  - Like FreeBSD Jails or Linux VServer
- Resource Management
  - Allocate CPU and memory to zones
- http://www.sun.com/bigadmin/features/articles/solaris_zones.html
- http://www.sun.com/bigadmin/content/zones/

Containers - Zones

- One global zone, multiple non-global zones
- Commands
  - `zonecfg` - Create, delete, and configure zones
  - `zoneadm` - Initialize, boot, and halt zones
  - `zlogin` - Log into a non-global zone from the global zone without going through the network
  - `zonename` - Display name of current zone
  - Many others have been made zone-aware (from the global zone), such as `ps`, `ipcs`, `pgrep`, `pkill`, `ptree`, `prstat`, `df`, and `ifconfig`

Containers - Zones

- Example: Create zone, set root filesystem (need ~100 MB for zone), set to boot on system startup, initialize, boot, and login to zone console

```
zonecfg -z zone1 create
zonecfg -z zone1 set zonepath=/zones/zone1
zonecfg -z zone1 set autoboot=true
zoneadm -z zone1 install
zoneadm -z zone1 boot
zlogin -C zone1
```

Containers - Resource Management

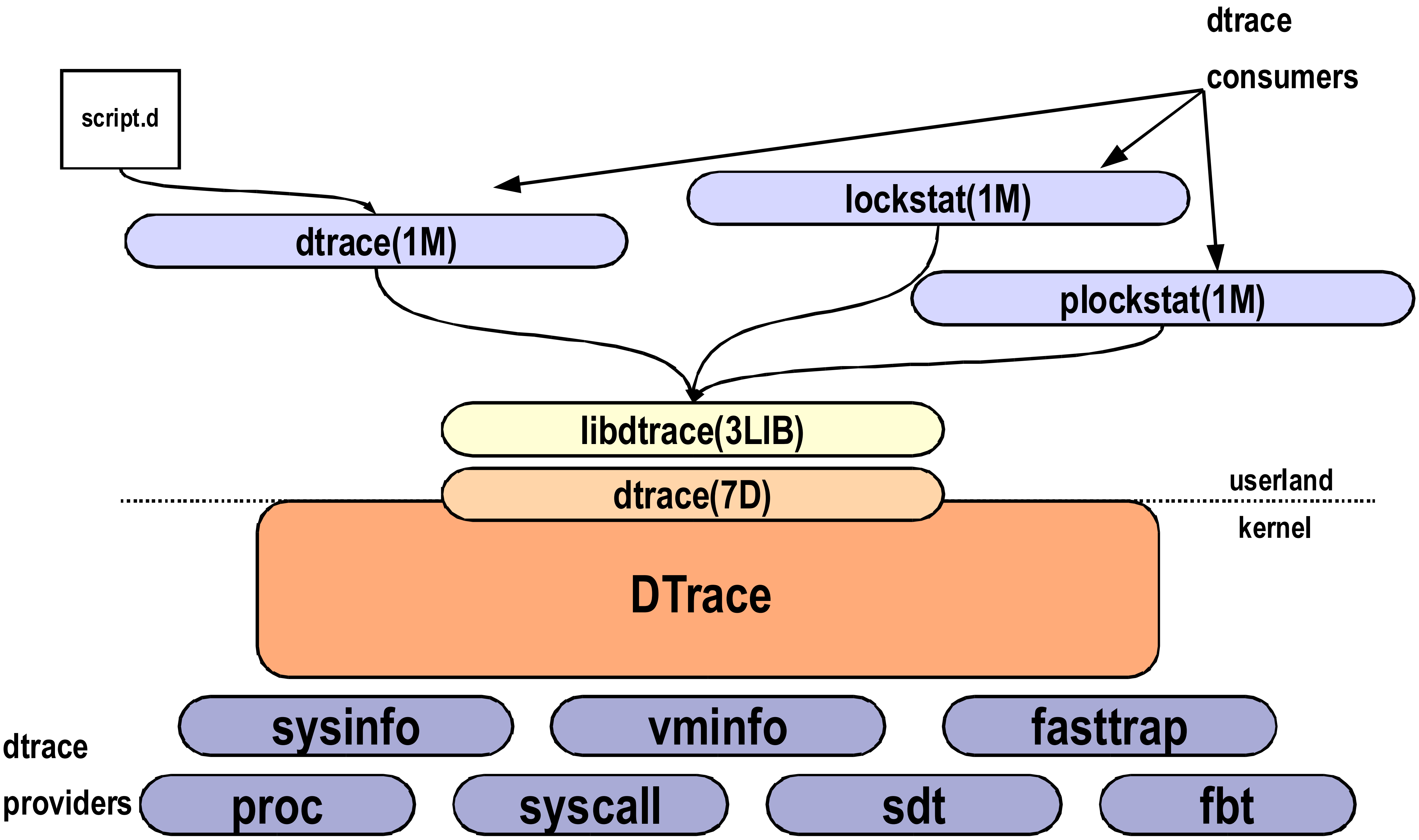
- `pooladm` - Enable, activate, and list resource pools
- `poolcfg` - Configure resource pools

Dynamic Tracing

DTrace - Intro

- Instrumentation points (“probes”) built into the kernel
- Run queries against these probles on a live system, with negligible performance penalty
  - Avoids measurement effect
- D scripting language
  - Lots of examples in `/usr/demo/dtrace/`
- http://www.sun.com/bigadmin/content/dtrace/

DTrace - Intro



DTrace Providers & Consumers

DTrace - Examples

- View of files as they're being opened for reading

```
# dtrace -n 'ufs_read:entry { printf("%s", stringof(args[0]->v_path)); }'
dtrace: description 'ufs_read:entry ' matched 1 probe
CPU     ID                    FUNCTION:NAME
  0  16845                   ufs_read:entry /usr/ucb/../bin/more
  0  16845                   ufs_read:entry /usr/ucb/../bin/more
  0  16845                   ufs_read:entry /usr/ucb/../bin/more
  0  16845                   ufs_read:entry /lib/ld.so.1.32770
  0  16845                   ufs_read:entry /lib/ld.so.1.32770
  0  16845                   ufs_read:entry /usr/share/lib/terminfo//v/vt100
  0  16845                   ufs_read:entry /etc/nsswitch.conf
  0  16845                   ufs_read:entry /etc/nsswitch.conf
  0  16845                   ufs_read:entry /etc/nsswitch.conf
  0  16845                   ufs_read:entry /etc/stmp
  0  16845                   ufs_read:entry /var/adm/lastlog
  0  16845                   ufs_read:entry /etc/default/login
  0  16845                   ufs_read:entry /etc/default/login
  0  16845                   ufs_read:entry /etc/project
  0  16845                   ufs_read:entry /etc/project
  0  16845                   ufs_read:entry /etc/security/policy.conf
  0  16845                   ufs_read:entry /etc/security/policy.conf
  0  16845                   ufs_read:entry /etc/security/policy.conf
 ^C
```

DTrace - Examples

- Distribution of `write(2)` sizes per executable

```
# dtrace -n 'syscall::write:entry { @[execname] = quantize(arg2); }'
dtrace: description 'syscall::write:entry ' matched 1 probe
^C

  dtrace
           value  ------------- Distribution ------------- count
               0 |                                         0
               1 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ 1
               2 |                                         0

  sshd
           value  ------------- Distribution ------------- count
               0 |                                         0
               1 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ 1
               2 |                                         0

  syslog-ng
           value  ------------- Distribution ------------- count
              16 |                                         0
              32 |@@@@@@@@                                 24
              64 |@@@@@@@@@@@@@@@@@@@@@                    66
             128 |@@@@@@@@@                                27
             256 |@@                                       6
             512 |                                         0
```

DTrace - Examples

- Distribution of system calls, and processes responsible

```
# dtrace -n 'syscall::entry { @[probefunc] = count() }'
dtrace: description 'syscall::entry ' matched 227 probes
^C

  lwp_continue                                                     1
  lwp_create                                                       1
 [...]
  write                                                           84
  lwp_sigmask                                                    113
  ioctl                                                         1067
  fstat64                                                       1625
  pollsys                                                       3310
  gtime                                                         5139
# dtrace -n 'syscall::gtime:entry { @[execname,pid] = count() }'
^C

  nscd                                                   115               12
  sendmail                                               285               16
  syslog-ng                                            14644              184
  sec                                                  14662              238
  sec                                                  14668              238
  sec                                                  14666              238
  sec                                                  14663              238
  sec                                                  14654              238
  sec                                                  14667             2003
```

DTrace - Examples

- Functions calls by process with specified PID

```
# dtrace -n 'pid14644:::entry { @[probefunc] = count() }'
dtrace: description 'syscall::entry ' matched 227 probes
^C

  __open                                                           1
  _cerror                                                          1
  _close                                                           1
  _open                                                            1
  chmod                                                            1
  chown                                                            1
  close                                                            1
[...]
  memset                                                       22409
  memcpy                                                       25035
  do_prepare_write                                             25497
  realfree                                                     25894
  prepare                                                      26915
  do_filter_or                                                 27659
  __regexec_C                                                  35565
  regexec                                                      35565
  assert_no_libc_locks_held                                    47849
  lmutex_lock                                                  47849
  lmutex_unlock                                                47849
  mutex_unlock_queue                                           47849
  tolower                                                      63264
```

DTrace - Examples

- D script to see processes running `exec(2)`

```
# cat /usr/demo/dtrace/whoexec.d
[...]
proc:::exec
{
        self->parent = execname;
}

proc:::exec-success
/self->parent != NULL/
{
        @[self->parent, execname] = count();
        self->parent = NULL;
}

proc:::exec-failure
/self->parent != NULL/
{
        self->parent = NULL;
}

END
{
        printf("%-20s %-20s %s\n", "WHO", "WHAT", "COUNT");
        printa("%-20s %-20s %@d\n", @);
}
```

DTrace - Examples

- D script to see processes running exec(2)

```
# dtrace -s /usr/demo/dtrace/whoexec.d
^C
WHO                  WHAT                 COUNT
cron                 sh                   1
sh                   logger               1
tcsh                 more                 1
dtrace               dtrace               2
sudo                 dtrace               2
tcsh                 date                 2
tcsh                 sudo                 2
sh                   more                 3
sh                   col                  3
sh                   neqn                 3
sh                   mv                   3
sh                   tbl                  3
sh                   nroff                3
tcsh                 man                  3
man                  sh                   9
```

logadm

`logadm` - Intro

- New log rotation utility, replacing venerable `newsyslog`
  - Actually introduced in Solaris 9
  - Much more extensively and easily configurable
- Configuration in `/etc/logadm.conf`
  - Manually edited, or via `logadm`

`logadm` - UTC

- `logadm` **always** works in UTC (unlike `cron`)
  - Ignores time zones
  - Timestamps generated for rotated files are in UTC
  - Example: Tried running `logadm` `cron` job at 23:58, to divide logs easily by whole days
    - However, rotated logs for 3/8/2006 would be named with a datestamp of 2006-03-09, since `logadm` thought it was 07:58 of the next day
      - Kind of confusing

`logadm` - Example

- Key to example `logadm.conf` lines
  - `-C` - Retain this many old copies (0 for unlimited)
  - `-N` - Don’t complain about missing log files
  - `-c` - Rotate by copying file then truncating to zero length
  - `-p` - Rotate this often
  - `-P` - Time of last rotation, in UTC (automatically updated)
  - `-t` - Name of rotated file (including macros)
  - `-z` - Compress rotated files with `gzip`, keeping this many uncompressed (doesn’t seem to work properly)
  - `-a` - Execute this command after rotation

logadm - Example

```
% cat /etc/logadm.conf
[...]
/var/log/sec/* -C 0 -N -p 1w -t '/var/log/archive/sec/$basename.%F' -z 0
/var/log/byapp/* -C 30 -N -c -p 1w -t '/var/log/archive/byapp/$basename.%F' -z 0
'/var/log/bysev/[0-9]*' -C 5 -N -c -p 1w -t '/var/log/archive/bysev/$basename.%F' -z 0
/var/log/all -C 0 -P 'Thu Mar  9 08:01:00 2006' -a '/usr/sbin/svcadm restart syslog-
ng' -p 1d -t /var/log/archive/all.%F -z 0
/var/log/sec/all_reduced -P 'Wed Mar  8 08:01:00 2006'
/var/log/sec/root_su -P 'Wed Mar  8 08:01:00 2006'
/var/log/byapp/memory -P 'Wed Mar  8 08:01:00 2006'
/var/log/byapp/netapp -P 'Wed Mar  8 08:01:00 2006'
/var/log/byapp/scsi -P 'Wed Mar  8 08:01:00 2006'
/var/log/byapp/su -P 'Wed Mar  8 08:01:00 2006'
/var/log/byfac/auth -P 'Wed Mar  8 08:01:00 2006'
/var/log/byfac/daemon -P 'Wed Mar  8 08:01:00 2006'
/var/log/byfac/kern -P 'Wed Mar  8 08:01:00 2006'
/var/log/byfac/local0 -P 'Wed Mar  8 08:01:00 2006'
/var/log/byfac/local2 -P 'Wed Mar  8 08:01:00 2006'
/var/log/byfac/local3 -P 'Wed Mar  8 08:01:00 2006'
[...]
```

More New Features

New Features - Security

- Process rights management
  - Grant users limited superuser privileges
  - See `privileges(5)` man page
  - Configured in `/etc/user_attr`
- IP Filter built-in
  - See `ipfilter(5)`, `ipf(1M)`, and `ipnat(1M)` man pages
- OpenSSL, SASL, TCP Wrappers included
  - OpenSSL doesn't include all algorithms (like higher-strength AES)

New Features - Network

- Improved TCP/IP performance (FireEngine)
  - http://www.sun.com/bigadmin/content/networkperf/
- High-speed connectivity: 10-Gb Ethernet, InfiniBand
- Storage networking protocols: NFSv4, iSCSI
- VoIP protocols: SIP, SCTP
- Routing protocols: OSPFv2, BGP-4
  - `routeadm` - New command to manage IP forwarding and routing

New Features - Network

```
% routeadm -p
ipv4-forwarding persistent=disabled default=disabled
current=disabled
ipv4-routing persistent=default default=disabled
current=disabled
ipv6-forwarding persistent=disabled default=disabled
current=disabled
ipv6-routing persistent=disabled default=disabled
current=disabled
ipv4-routing-daemon persistent="/usr/sbin/in.routed" default="/
usr/sbin/in.routed"
ipv4-routing-daemon-args persistent="" default=""
ipv4-routing-stop-cmd persistent="kill -TERM `cat /var/tmp/
in.routed.pid`" default="kill -TERM `cat /var/tmp/
in.routed.pid`"
ipv6-routing-daemon persistent="/usr/lib/inet/in.ripngd"
default="/usr/lib/inet/in.ripngd"
ipv6-routing-daemon-args persistent="-s" default="-s"
ipv6-routing-stop-cmd persistent="kill -TERM `cat /var/tmp/
in.ripngd.pid`" default="kill -TERM `cat /var/tmp/
in.ripngd.pid`"
```

New Features - Other

- Fault Manager
  - Like SMF, another component of Predictive Self Healing
  - See `fmd(1M)` man page

```
# fmadm config
MODULE                   VERSION STATUS   DESCRIPTION
USII-io-diagnosis        1.0     active   UltraSPARC-II I/O Diagnosis
cpumem-retire            1.0     active   CPU/Memory Retire Agent
eft                      1.13    active   eft diagnosis engine
fmd-self-diagnosis       1.0     active   Fault Manager Self-Diagnosis
io-retire                1.0     active   I/O Retire Agent
syslog-msgs              1.0     active   Syslog Messaging Agent
# fmstat
module             ev_recv ev_acpt wait  svc_t  %w  %b  open solve  memsz  bufsz
USII-io-diagnosis        0       0  0.0    0.2   0   0     0     0      0      0
cpumem-retire            0       0  0.0    0.2   0   0     0     0      0      0
eft                      0       0  0.0    0.2   0   0     0     0   704K      0
fmd-self-diagnosis       0       0  0.0    0.2   0   0     0     0      0      0
io-retire                0       0  0.0    0.2   0   0     0     0      0      0
syslog-msgs              0       0  0.0    0.3   0   0     0     0    32b      0
# fmdump
TIME                 UUID                                 SUNW-MSG-ID
fmdump: /var/fm/fmd/fltlog is empty
```

New Features - Other

- Names of open files kept in `/proc`
  - `pfiles` now prints pathnames of open files, in addition to inode numbers and other statistics
- SysV IPC dynamic tuning
- `gcc` included (Yay!)
- Webmin included
- Java Desktop (GNOME) included

What's New in Solaris 10?

Leon Towns-von Stauber, Occam's Razor
Seattle SAGE Group, March 2006
http://www.occam.com/