# UNIX Process-Launching Mechanisms

Leon Towns-von Stauber, Occam's Razor
Seattle Area System Administrators Guild,
March 2007
http://www.occam.com/

# Contents

# Introduction

- For many years in UNIX-land, things were much the same
  - `init` was process #1, and it spawned all other processes, directly or indirectly, via `rc` scripts, `inittab` entries, etc.
  - `cron` and `at` started scheduled processes
  - `inetd` begat network daemons when called upon
- Things are changing...
  - In the last few years, several new alternatives to these traditional mechanisms have arisen
  - UNIX operating systems are becoming less consistent for sysadmins in this respect

# Introduction

- Scope of discussion: automated process-launching
  - Includes processes launched as part of boot process, at scheduled times, upon reception of network request, or automatically triggered by other events
  - Does not include processes launched as a direct result of user interaction
- Overview
  - Brief review of traditional UNIX mechanisms
  - New approaches for commercial UNIX variants: AIX, Solaris, Mac OS X
  - Linux-based and cross-platform projects

# Legal Notices

- This presentation Copyright © 2007 Leon Towns-von Stauber. All rights reserved.
- Trademark notices
  - UNIX® is a trademark of The Open Group. See http://www.unix.org/trademark.html.
  - IBM®, AIX®, DYNIX/ptx®, and other terms are trademarks of IBM. See http://www.ibm.com/legal/copytrade.shtml.
  - Sun™, Solaris™, and other terms are trademarks of Sun Microsystems. See http://www.sun.com/suntrademarks/.
  - Apple®, Mac OS X®, and other terms are trademarks of Apple. See http://www.apple.com/legal/appletmlist.html.

# Legal Notices

- Trademark notices (cont'd.)
  - IRIX® is a trademark of Silicon Graphics. See http://www.sgi.com/company_info/trademarks/sgi.html.
  - Linux® is a trademark of Linux Torvalds. See http://linuxmark.org/.
  - Debian™ is a trademark of the Debian Project.
  - Ubuntu® is a trademark of Canonical Ltd. See http://www.ubuntu.com/ubuntu/TrademarkPolicy/.
  - Red Hat® and Fedora™ are trademarks of Red Hat. See http://www.redhat.com/about/companyprofile/trademark/.
  - Other trademarks are the property of their respective owners.

# Traditional Mechanisms

# Traditional - `init`

- Traditionally, `init` is the first process started on a UNIX system after the kernel has been loaded into memory and initialized
  - Not always the very first, but always assigned PID `1`
- `init` is responsible for starting everything else
- Two main code paths have been around since the 1980s: BSD & System V
- BSD `init`
  - Processes `/etc/ttys` or `/etc/ttytab`
  - Runs `/etc/rc`, or sometimes `/etc/rc.boot`
    - Calls other run control scripts (`/etc/rc.*`)

# Traditional - SysV `init`

- AT&T System V UNIX introduced more sophisticated system initialization
- Run levels
  - BSD `init` generally recognizes a small number of operating system states: halted, single-user, multi-user
  - Run levels offer more finely graded, numerical rankings representing desired states of OS
  - 7-10 levels defined, but usually only a handful are actually used, and there's some discrepancy between UNIX flavors as to what each level represents

# Traditional - SysV `init`

- Processes `/etc/inittab`
  - Set up TTY devices, start certain processes, run `rc` scripts, each according to desired run level
  - Runs `rc` script to call startup scripts associated with run level

# Traditional - SysV `init`

- Startup scripts
  - Each service has its own self-contained startup script
    - Makes it easier to add or remove services
  - For run level *N*, located in `/etc/rc`*N*`.d/` or `/sbin/rc`*N*`.d/` or `/etc/rc.d/rc`*N*`.d/`
  - Links (preferably), hard or soft, to shell scripts in `../init.d/`
  - Each starts with `S` or `K` (passing arguments of `start` or `stop` to script), and a 2- or 3-digit sequence number
  - Executed in sequence, one at a time, when that run level is entered

# Traditional - SysV `init`

```
% uname -sr
SunOS 5.9
% ls -l /etc/rc2.d
total 65
lrwxrwxrwx   1 root          18 Dec  3  2004 K03opensshd -> ../init.d/opensshd
-rwxr--r--   6 root         391 Nov 10 05:56 K05volmgt
-rwxr--r--   3 root        1745 Apr  6  2002 S01MOUNTFSYS
-rwxr--r--   2 root        2010 Apr  6  2002 S05RMTMPFILES
-rwxr--r--   2 root        1781 Apr  6  2002 S20sysetup
-rwxr--r--   2 root         963 Dec  3  2004 S21perf
-rwxr--r--   3 root         499 Oct  8  2001 S25lom
-rwxr-xr-x   2 root        2012 Apr 14  2002 S30sysid.net
-rwxr--r--   5 root       12655 Jul  5  2006 S69inet
-rwxr--r--   5 root         420 Apr  6  2002 S71ldap.client
lrwxrwxrwx   1 root          13 Dec  3  2004 S71rpc -> ../init.d/rpc
-rwxr-xr-x   2 root        1498 Apr 14  2002 S71sysid.sys
-rwxr--r--   5 root        7197 Nov 12 12:42 S72inetsvc
-rwxr--r--   5 root         947 Dec  3  2004 S74syslog
-rwxr--r--   5 root         951 Apr  6  2002 S74xntpd
-rwxr--r--   5 root         504 Apr  6  2002 S75cron
-rwxr--r--   2 root        2519 Apr  6  2002 S75savecore
-rwxr--r--   5 root         556 Apr  6  2002 S76nscd
-rwxr--r--   5 root         597 Apr  6  2002 S88utmpd
-rwxr--r--   2 root         256 Apr  6  2002 S89PRESERVE
-rwxr--r--   5 root         416 Apr  6  2002 S99audit
-rwxr--r--   3 root         836 Apr  6  2002 s73nfs.client
-rwxr--r--   5 root         364 Apr  6  2002 s74autofs
-rwxr--r--   5 root        3202 Apr  6  2002 s88sendmail
```

# Traditional - SysV `init`

- Starting and stopping services
  - `init.d/`*`service`* `{ start | stop }`
  - May also have `restart`, `reload`, `force-reload`, `force-stop`
- Enabling and disabling services
  - Determined by presence or absence of appropriate links in the `rc` directories
  - `chkconfig`, `update-rc.d`, `rcstart` (more later)
- Obtaining service information
  - `init.d/`*`service`* `status` (not guaranteed available)

# Traditional - SysV `init`

- Features
  - Automatic process restart
    - For commands listed with `respawn` in `inittab`
  - Self-contained startup scripts per service
  - Sequential startup
    - No dependency management
- Basis for comparison with later mechanisms

# Traditional - SysV `init`

- Additional management tools
  - `chkconfig`
    - Red Hat, adopted from IRIX
    - Enabling and disabling services
      - `chkconfig` *`service`* `{ on | off }`
      - Can also specify run levels
- Listing services, with enabled status
  - `chkconfig --list [`*`service`*`]`

# Traditional - SysV `init`

- Additional management tools (cont'd.)
  - `update-rc.d`
    - Debian
    - Enabling and disabling services
      - `update-rc.d` *`service`* `start` *`N runlevel runlevel...`* `stop` *`N runlevel runlevel...`*
        - *`N`* are sequence numbers
      - `update-rc.d` *`service`* `defaults`
        - Enable for run levels 2-5
      - `update-rc.d` *`service`* `remove`
- See also `invoke-rc.d`

# Traditional - SysV `init`

- Additional management tools (cont'd.)
  - `rcstart`
    - http://www.occam.com/tools/
    - Bourne shell script, written in 2000, updated and released as open source in 2004
    - Cross-platform: Tested on HP-UX, Solaris, Red Hat Linux; IRIX and DYNIX/ptx supported but untested
    - Enabling and disabling services
      - `rcstart { -y | -n }` *`service service...`*
        - Changes `S*` links/scripts in `rc` directories to `s*` (thus disabling them), or reverse

# Traditional - Scheduled Processes

- Processes launched according to time
  - Also called "batch" jobs (mainframe term)
- `at` for one-time actions
  - Command to run, and time to run it, provided when executing `at` command
- `cron` for repeated actions
  - Commands and schedules configured in central and/or per-user `crontab` files
    - Only minor format variations across UNIX flavors
    - Typically use `crontab` command to make changes, and notify `cron` daemon

# Traditional - `inetd`

- `inetd` designed as a network "superserver"
  - Doesn't provide any network services directly
  - Listens for requests on behalf of network services, launches daemons or other processes on demand
  - Created to minimize system resource usage
- Configured in `/etc/inetd.conf`
- Starting and stopping services
  - Automatic
- Enabling and disabling services
  - Determined by entries in `inetd.conf`; comment out to disable

# Traditional - `inetd`

- `ctlinetd`
  - http://www.occam.com/tools/
  - Perl script, written in 1999, released as open source 2004
  - Cross-platform: Tested on DYNIX/ptx, HP-UX, Mac OS X, Solaris, various Linux distros
  - Enabling and disabling services
    - `ctlinetd { enable | disable }` *`service`*
      - Comments/uncomments entries in `inetd.conf`, reloads configuration
    - `ctlinetd commentall`
      - Comments out all entries, doesn't reload config

# Traditional - `inetd`

- `ctlinetd` (cont'd.)
  - Obtaining service information
    - `ctlinetd status [`*`service`*`...]`
      - Reports whether `inetd` is running, with PID
      - Reports whether specified services are enabled, and if TCP Wrappers are in place
    - `ctlinetd listenabled`
      - Lists all enabled services
  - Control of `inetd`
    - `ctlinetd { reload | restart | start | stop }`

# Traditional - inetd

```
% ctlinetd listenabled
printer is enabled, using tcp_wrappers
omni is enabled, using tcp_wrappers
inetd process running (pid 295)
% ctlinetd status ftp printer
ftp is multiply defined and disabled
printer is enabled, using tcp_wrappers
inetd process running (pid 295)
```

# AIX System Resource Controller

# AIX SRC

- Unlike the other non-traditional mechanisms discussed in this talk, SRC has actually been around for a long time, since at least 1990 (AIX 3.1)
  - But then AIX was always considered an oddball variant
- "Services" are called *subsystems*, processes are *subservers*
  - Subsystems contained in groups (e.g., `tcpip`, `nfs`)
- SRC daemon (`/usr/sbin/srcmstr`) started by `init` from `inittab`
- Replaces startup scripts (at least those provided by OS)
- `inetd` is an SRC subsystem
  - `inetd`-based services are SRC subservers

# AIX SRC

- Enabling and disabling services
  - `startsrc` commands in `/etc/inittab` or `rc` scripts
  - `mkssys`, `chssys`, `rmssys`
    - Creates, changes, or removes subserver configuration in ODM
- Starting and stopping services
  - `startsrc -s` *`subsystem`*, `stopsrc -s` *`subsystem`*
  - `startsrc -t` *`subserver`*, `stopsrc -t` *`subserver`*
    - Starts parent subsystem if not already running
  - `refresh -s` *`subsystem`*

# AIX SRC

- Obtaining service information
  - `lssrc -a`
    - Lists all subsystems
  - `lssrc -s` *`subsystem`*, `lssrc -t` *`subserver`*
    - Gets status of subsystem or subserver
  - `odmget -q subsysname=`*`subsystem`* `SRCsubsys`
    - Displays subsystem properties in ODM

# AIX SRC

```
% sudo lssrc -a
Subsystem         Group            PID          Status
 prngd            prng             331848       active
 syslogd          ras              389152       active
 portmap          portmap          393458       active
 inetd            tcpip            397544       active
 xntpd            tcpip            409854       active
 nimsh            nimclient        377066       active
 biod             nfs              401428       active
 rpc.statd        nfs              430094       active
 rpc.lockd        nfs              418044       active
 sddsrv                            499800       active
 sshd             ssh              487674       active
 ctrmc            rsct             479478       active
 IBM.ERRM         rsct_rm          454880       active
 IBM.HostRM       rsct_rm          442588       active
 IBM.ServiceRM    rsct_rm          438488       active
 IBM.DRM          rsct_rm          450562       active
 IBM.AuditRM      rsct_rm          467188       active
 IBM.CSMAgentRM   rsct_rm          434392       active
 IBM.LPRM         rsct_rm          475140       active
 qdaemon          spooler                       inoperative
 writesrv         spooler                       inoperative
 lpd              spooler                       inoperative
 rwhod            tcpip                         inoperative
 sendmail         mail                          inoperative
 snmpd            tcpip                         inoperative
[...]
```

# AIX SRC

```
% odmget -q subsysname=xntpd SRCsubsys

SRCsubsys:
        subsysname = "xntpd"
        synonym = ""
        cmdargs = ""
        path = "/usr/sbin/xntpd"
        uid = 0
        auditid = 0
        standin = "/dev/console"
        standout = "/dev/console"
        standerr = "/dev/console"
        action = 2
        multi = 0
        contact = 3
        svrkey = 0
        svrmtype = 0
        priority = 20
        signorm = 0
        sigforce = 0
        display = 1
        waittime = 20
        grpname = "tcpip"
```

# Solaris

# Service Management Facility

# Solaris SMF - Intro

- Service Management Facility introduced with Solaris 10 in 2005
- Open source, but only widely used in Solaris
  - License incompatibility with Linux (CDDL vs. GPL)
- Replacement for `inittab`, `rc` scripts, and `inetd`
  - `inittab` much simpler in Solaris 10 (only 4 lines)
- http://www.sun.com/bigadmin/content/selfheal/smf-quickstart.html

# Solaris SMF - Intro

- Features
  - Automatic process restart
  - Dependency management
  - Parallel startup
  - Built-in TCP Wrapper support (including `rpcbind`)

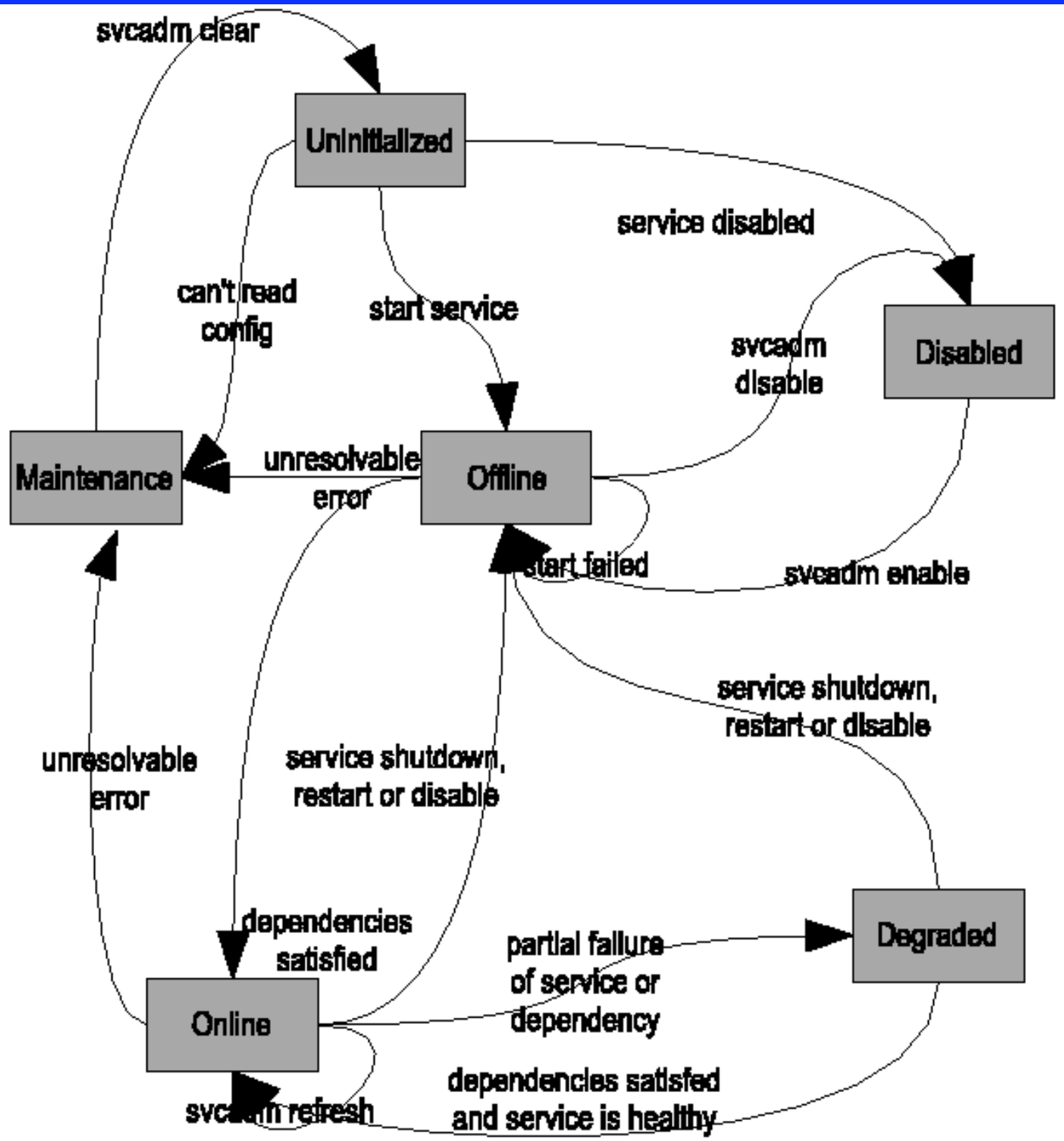
# Solaris SMF - Daemons

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

# Solaris 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 SysV startup scripts; SMF lists these, but can give no further state information

# Solaris SMF - States




Service States and Transitions

# Solaris 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`

# Solaris 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 Repository - working copy of service definitions
  - `/etc/svc/repository.db`

# Solaris SMF - Commands

- Enabling and disabling services
  - `svcadm { enable | disable | restart }` *`FMRI...`*
  - `inetadm { -e | -d }` *`FMRI...`*
    - Enable/disable `inetd` services
  - `ctlinetd` also supports SMF
- Obtaining service information
  - `svcs` - Lists services, with state and time of last state change
  - `svcs -a` - Lists all services, including disabled
  - `inetadm` - Lists `inetd` services

# Solaris SMF - Commands

- Obtaining service information (cont'd.)
  - `svcs -d` *`FMRI...`* - Services on which service depends
  - `svcs -D` *`FMRI...`* - Services which depend on service
  - `svcs -l` *`FMRI...`* - Information about service
  - `svcs -p` *`FMRI...`* - Processes associated with service
  - `svcs -x [`*`FMRI...`*`]` - Explanation for service state
  - `svccfg -s` *`FMRI`* `listprop` - Lists service properties
  - `inetadm -l` *`FMRI...`* - Lists `inetd` service properties

# Solaris SMF - Commands

- Configuring services
  - `svccfg` - Interactive mode
  - `svccfg import` *`filename`* - Brings service described by specified XML manifest under SMF management, importing it into SMF repository
  - `svccfg delete` *`FMRI`* - Deletes service configuration
  - `svccfg -s` *`FMRI`* `setprop` *`name=value`* - Modifies service property
  - `inetadm -m` *`FMRI... name=value...`* - Modifies `inetd` service properties

# Solaris 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
[...]
```

# Solaris 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)
```

# Solaris 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
```

# Solaris 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
```

# Solaris 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
```

# Solaris 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
```

# Solaris 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>
[...]
```

# Solaris 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' />
[...]
```

# Solaris 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>
```

# Solaris 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
```

# Mac OS X

# Mac OS X - Intro

- Mac OS X has had a variety of unorthodox process-launching mechanisms since version 10.0 (in 2001)
  - Apple willing to experiment with UNIX traditions
  - NEXTSTEP used traditional BSD `rc` scripts
- Mechanisms under discussion
  - Startup Items
  - Login Hooks
  - `watchdog`
  - Mach Bootstrap Daemons
  - `launchd`


- http://developer.apple.com/technotes/tn2005/tn2083.html

# Mac OS X - Startup Items

- `/etc/rc` calls `SystemStarter`
- `SystemStarter` processes services in `/System/Library/StartupItems/` (vendor-provided) and `/Library/StartupItems/` (custom)
- Each item is a directory containing:
  - Executable (usually a shell script) named the same as the directory, which is called with a `start` argument
  - `StartupParameters.plist` - XML config file, containing:
    - Description
    - Services provided, required, and used
    - Preference: `First`, `Early`, `Late`, `Last`, `None`

# Mac OS X - Startup Items

```
% sw_vers
ProductName:    Mac OS X
ProductVersion: 10.4.8
BuildVersion:   8L127
% uname -sr
Darwin 8.8.0
% ls -l /System/Library/StartupItems/
total 0
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 Apache
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 AppServices
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 AppleShare
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 AuthServer
drwxr-xr-x   5 root  wheel  170 Jan 31  2006 CrashReporter
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 Disks
drwxr-xr-x   4 root  wheel  136 Jan 30  2006 FibreChannel
drwxr-xr-x   4 root  wheel  136 Feb 25  2006 IFCStart
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 IPServices
drwxr-xr-x   4 root  wheel  136 Feb  8  2006 Metadata
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 NFS
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 NIS
drwxr-xr-x   4 root  wheel  136 Feb 19  2006 NetworkTime
drwxr-xr-x   5 root  wheel  170 Mar 22  2006 PrintingServices
drwxr-xr-x   5 root  wheel  170 Dec 18 10:46 RemoteDesktopAgent
drwxr-xr-x   5 root  wheel  170 Jan 30  2006 SNMP
```

# Mac OS X - Startup Items

```
% cat /System/Library/StartupItems/ConfigServer/
StartupParameters.plist
{
  Description     = "network configuration";
  Provides        = ("Network Configuration");
  Uses            = ("System Tuning");
  OrderPreference = "First";
}
```

# Mac OS X - Startup Items

- Enabling and disabling services
  - Usually done by setting parameter in `/etc/hostconfig`
- Features
  - Parallel startup
    - Sequence is non-deterministic, defined by dependencies and preferences
  - Dependency management

# Mac OS X - Login Hooks

- Program to run upon every login (or logout)
  - Runs as root, receives username as argument
  - Can be used to set up user environment, enforce policy, etc.
- In `/etc/ttys`, add `-LoginHook` *`/path/to/executable`* and/or `-LogoutHook` *`/path/to/executable`* as argument to loginwindow command for `console` device
- Starting with Mac OS X 10.3, can use `defaults` command
  - `defaults write com.apple.loginwindow LoginHook` *`/path/to/executable`*
  - `defaults delete com.apple.loginwindow LoginHook`

# Mac OS X - Login Hooks

- Example login hook: `updateByHostPrefs`
  - http://www.occam.com/tools/
- Enabling and disabling services
  - Determined by existence of loginwindow argument in `/etc/ttys`, or loginwindow default in defaults database
- Starting and stopping services
  - Upon login or logout
- Deprecated in Tiger in favor of `launchd`

# Mac OS X - `watchdog`

- Included in Mac OS X Server (not in regular client version)
- Started by `Watchdog` startup item
- Starts and restarts (as necessary) several server daemons
  - Reads `/etc/watchdog.conf`, similar to SysV `inittab`
- Also resets automatic reboot timer in system's power management unit (PMU)
  - If timer ever expires, machines suffers a hard reboot
  - Meant to automatically recover a hung system
- Deprecated by `launchd` in Tiger Server
  - Now `watchdogtimerd` left as remnant, handling reboot timer

# Mac OS X - watchdog

```
% tail -8 /etc/watchdog.conf

pwd:respawn:/usr/sbin/PasswordService -n
PSM:respawn:/usr/sbin/PrintServiceMonitor -x    # Server Printing service
mm:off:/usr/sbin/MacintoshManagementServer -x   # Macintosh Manager service
postfix:respawn:/usr/libexec/postfix/master     # Mail services - SMTP
cyrus:off:/usr/bin/cyrus/bin/master             # Mail services - IMAP & POP:SA2
kadmind:respawn:/usr/sbin/kadmind -passwordserver
kdc:respawn:/usr/sbin/krb5kdc
```

# Mac OS X - `watchdog`

- Enabling and disabling services
  - Set by service's action field in `/etc/watchdog.conf`
  - Possible actions: `off`, `respawn`, `boot`, `bootonce`, `now`
- Starting and stopping services
  - Automatic

# Mac OS X - Mach Bootstrap Daemons

- Introduced in Panther (Mac OS X 10.3), 2003
  - Replaces certain startup items, and login hooks
- `/etc/rc` calls `register_mach_bootstrap_servers` to process XML property lists in `/etc/mach_init.d/`
  - For configured services, registers Mach ports with the kernel's bootstrap task, via `mach_init`
    - A Mach *task* is analogous to a process running within the kernel; a Mach *port* is used to send messages to a task
- Daemon is launched on demand, when another process opens communication with its Mach port, thus conserving system resources

# Mac OS X - Mach Bootstrap Daemons

- The same mechanism can also be used to launch per-user processes, or *agents*
  - Upon login, the loginwindow application calls `register_mach_bootstrap_servers` to process XML property lists in `/etc/mach_init_per_user.d/`
  - Ports are registered in the user's bootstrap namespace, instead of the global namespace; only available in that login session
- Enabling and disabling services
  - Determined by presence of property list
- Starting and stopping services
  - Automatic

# Mac OS X - Mach Bootstrap Daemons

```
% sw_vers
ProductName:    Mac OS X
ProductVersion: 10.4.8
BuildVersion:   8L127
% uname -sr
Darwin 8.8.0
% ls -l /etc/mach_init.d
total 160
-rw-r--r--   1 root  wheel  405 Mar  7  2006 ATSServer.plist
-rw-r--r--   1 root  wheel  360 Mar 11  2006 DirectoryService.plist
-rw-r--r--   1 root  wheel  447 Aug 30  2006 IIDCAssistant.plist
-rw-r--r--   1 root  wheel  363 Jan 30  2006 KerberosAutoConfig.plist
-r--r--r--   1 root  wheel  421 Aug 22  2006 WindowServer.plist
-rw-r--r--   1 root  wheel  392 Jan 30  2006 configd.plist
-rw-r--r--   1 root  wheel  356 Jan 30  2006 coreaudiod.plist
-rw-r--r--   1 root  wheel  385 Jan 30  2006 coreservicesd.plist
-rw-r--r--   1 root  wheel  413 Sep 11 17:12 dashboardadvisoryd.plist
-rw-r--r--   1 root  wheel  420 Jan 30  2006 diskarbitrationd.plist
-rw-r--r--   1 root  wheel  372 Jan 30  2006 distnoted.plist
-rw-r--r--   1 root  wheel  502 Jan 31  2006 hdiejectd.plist
-rw-r--r--   1 root  wheel  402 Jan 30  2006 kuncd.plist
-r--r--r--   1 root  wheel  340 Feb  9  2006 lookupd.plist
-rw-r--r--   1 root  wheel  436 Feb  8  2006 mds.plist
-rw-r--r--   1 root  wheel  347 Jan 30  2006 memberd.plist
-r--r--r--   1 root  wheel  360 Feb  9  2006 notifyd.plist
-rw-r--r--   1 root  wheel  339 Aug 11  2006 ocspd.plist
-rw-r--r--   1 root  wheel  392 Jan 30  2006 scsid.plist
-rw-r--r--   1 root  wheel  390 Feb  8  2006 securityd.plist
```

# Mac OS X - Mach Bootstrap Daemons

```
% cat /etc/mach_init.d/configd.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>ServiceName</key>
        <string>com.apple.SystemConfiguration.configd</string>
        <key>Command</key>
        <string>/usr/sbin/configd</string>
        <key>OnDemand</key>
        <false/>
</dict>
</plist>
```

# Mac OS X - Mach Bootstrap Daemons

```
% sw_vers
ProductName:    Mac OS X
ProductVersion: 10.4.8
BuildVersion:   8L127
% uname -sr
Darwin 8.8.0
% ls -l /etc/mach_init_per_user.d
total 96
-rw-r--r--   1 root  wheel  463 Jan 30  2006 AddressBookSharing.plist
-rw-r--r--   1 root  wheel  453 Feb 12  2006 CCacheServer.plist
-rw-r--r--   1 root  wheel  433 Aug 22  2005 CoreMIDIServer.plist
-rw-r--r--   1 root  wheel  410 Jan 31  2006 Crash Reporter.plist
-rw-r--r--   1 root  wheel  461 Mar  7  2006 FontValidator.plist
-rw-r--r--   1 root  wheel  456 Feb 12  2006 KerberosAgent.plist
-rw-r--r--   1 root  wheel  401 Mar  7  2006 MirrorAgent.plist
-rw-r--r--   1 root  wheel  430 Jan 30  2006 Network Diagnostics.plist
-rw-r--r--   1 root  wheel  423 Jan 30  2006 airport_hookupd.plist
-rw-r--r--   1 root  wheel  407 Jan 30  2006 airportd.plist
-rw-r--r--   1 root  wheel  467 Feb  8  2006 dmnotify.plist
-rw-r--r--   1 root  wheel  404 Feb 25  2006 syncuid.plist
```

# Mac OS X - Mach Bootstrap Daemons

- Features
  - Parallel startup
  - On-demand execution
    - Triggers: interprocess communication, network contact, device attachment, etc.
    - No explicit dependency management
  - Supports per-user processes
- Deprecated in Tiger in favor of `launchd`

# Mac OS X - `launchd`

- `launchd` is a replacement for `init`, `rc` scripts, `SystemStarter`, login hooks, `watchdog`, Mach bootstrap daemons, `cron`, `inetd`, ...
  - Not quite there yet; other methods are deprecated, but still functional in Tiger
  - `launchd` is now PID `1`
- Introduced in Tiger (Mac OS X 10.4), 2005
- Similar design goals to Solaris 10 SMF
- Open source, but only widely used on Mac OS X
  - License incompatibility with Linux (APSL vs. GPL)
    - Re-released under Apache license August 2006

# Mac OS X - `launchd`

- Features
  - Parallel startup
  - On-demand execution
    - Triggers: interprocess communication, network contact, device attachment, etc.
    - No explicit dependency management
  - Automatic process restart
  - Supports per-user processes
  - Combines multiple mechanisms, simplifying configuration

# Mac OS X - `launchd`

- Features (cont'd.)
  - Handles daemonization for processes
  - Opens privileged ports on behalf of daemons
  - Built-in `chroot` functionality

# Mac OS X - `launchd`

- System daemons or per-user agents are configured by XML property lists
  - See `launchd.plist` man page for syntax
  - Daemon config files are in either `/System/Library/LaunchDaemons/` or `/Library/LaunchDaemons/`
  - Agent config files are in `/System/Library/LaunchAgents/`, `/Library/LaunchAgents/`, or `~/Library/LaunchAgents/`

# Mac OS X - launchd

```
% ls -l /System/Library/LaunchDaemons/
total 256
-rw-r--r--   1 root  wheel   678 Jan 30  2006 bootps.plist
-rw-r--r--   1 root  wheel   495 Jan 30  2006 com.apple.KernelEventAgent.plist
-rw-r--r--   1 root  wheel   445 Feb 21  2006 com.apple.atrun.plist
-rw-r--r--   1 root  wheel   458 Sep 11 17:12 com.apple.dashboard.advisory.fetch.plist
-rw-r--r--   1 root  wheel   447 Jan 30  2006 com.apple.mDNSResponder.plist
-rw-r--r--   1 root  wheel   437 Feb  9  2006 com.apple.nibindd.plist
-rw-r--r--   1 root  wheel   579 Jan 30  2006 com.apple.periodic-daily.plist
-rw-r--r--   1 root  wheel   623 Jan 30  2006 com.apple.periodic-monthly.plist
-rw-r--r--   1 root  wheel   625 Jan 30  2006 com.apple.periodic-weekly.plist
-rw-r--r--   1 root  wheel   524 Jan 30  2006 com.apple.portmap.plist
-rw-r--r--   1 root  wheel   516 Jan 30  2006 com.apple.syslogd.plist
-rw-r--r--   1 root  wheel   574 Jan 30  2006 com.apple.xgridagentd.plist
-rw-r--r--   1 root  wheel   594 Jan 30  2006 com.apple.xgridcontrollerd.plist
-rw-r--r--   1 root  wheel   516 Jan 30  2006 com.vix.cron.plist
-rw-r--r--   1 root  wheel   628 Jan 30  2006 comsat.plist
[...]
-rw-r--r--   1 root  wheel   446 Jan 30  2006 org.isc.named.plist
-rw-r--r--   1 root  wheel   531 Aug 22  2005 org.postfix.master.plist
-rw-r--r--   1 root  wheel   566 Jan 30  2006 org.xinetd.xinetd.plist
-rw-r--r--   1 root  wheel   729 Jan 30  2006 printer.plist
-rw-r--r--   1 root  wheel   575 Feb 23  2006 shell.plist
-rw-r--r--   1 root  wheel   800 Dec 18 09:04 smbd.plist
-rw-r--r--   1 root  wheel   828 Dec 18 09:55 ssh.plist
-rw-r--r--   1 root  wheel   692 Mar  2  2006 swat.plist
-rw-r--r--   1 root  wheel   615 Feb 23  2006 telnet.plist
-rw-r--r--   1 root  wheel   715 Dec 18 09:04 tftp.plist
```

# Mac OS X - launchd

```
% cat /System/Library/LaunchDaemons/com.apple.syslogd.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Label</key>
    <string>com.apple.syslogd</string>
    <key>ServiceDescription</key>
    <string>Apple System Log Daemon</string>
    <key>OnDemand</key>
    <false/>
    <key>ProgramArguments</key>
    <array>
            <string>/usr/sbin/syslogd</string>
    </array>
    <key>ServiceIPC</key>
    <false/>
</dict>
</plist>
```

# Mac OS X - `launchd`

- Can run scheduled processes using `StartCalendarInterval` key, replacing `cron`
  - If system asleep at scheduled time, runs job upon waking (similar to `anacron`)
- Handles `inetd` services with `inetdCompatibility` key
- Enabling and disabling services
  - `launchctl { load | unload }` *`/path/to/config...`*
- Starting and stopping services
  - `launchctl { start | stop }` *`service service...`*
- Obtaining service information
  - `launchctl list`

# Mac OS X - launchd

```
% sudo launchctl list
com.apple.dashboard.advisory.fetch
com.apple.KernelEventAgent
com.apple.mDNSResponder
com.apple.nibindd
com.apple.periodic-daily
com.apple.periodic-monthly
com.apple.periodic-weekly
com.apple.portmap
com.apple.syslogd
com.vix.cron
org.postfix.master
org.xinetd.xinetd
com.openssh.sshd
```

# Linux and Cross-Platform Projects

# Linux/Cross-Platform - `anacron`

- Adjunct to `cron`
- Runs processes as soon as possible after scheduled time
  - Useful for machines without constant uptime
- Can't schedule processes in increments less than a day
- Doesn't run continuously
  - Called from startup scripts, `cron`, etc.
- Widely ported

# Linux/Cross-Platform - `xinetd`

- Replacement for `inetd`
- Features
  - More extensive configurability
    - Typically configured by `/etc/xinetd.conf`, and by individual service files in `/etc/xinetd.d/`
  - Improved security
    - Built-in network access control
    - Rate-limited connections
  - More sophisticated logging

# Linux/Cross-Platform - `xinetd`

- Now included with many operating systems: Linux, Mac OS X (albeit deprecated in Tiger), etc.
- Supported by `ctlinetd`
- http://www.xinetd.org/

# Linux/Cross-Platform - daemontools

- Daniel J. Bernstein
  - Also author of `qmail`, `djbdns`, etc.
- Daemons
  - `svscanboot`
    - Sets environment at boot, starts `svscan`
  - `svscan`
    - Repeatedly scans `/service/` directory, starts `supervise` process for each service directory found
  - `supervise`
    - For each service, runs `/service/`*`service`*`/run` script
    - Restarts service if necessary

# Linux/Cross-Platform - daemontools

- Enabling and disabling services
  - Determined by presence of service subdirectory
- Starting and stopping services
  - `svc { -u | -d } service service...`
- Obtaining service information
  - `svstat service service...`
- http://code.dogmap.org/svscan-1/
  - Running `svscan` as PID `1`
- http://cr.yp.to/daemontools.html

# Linux/Cross-Platform - `runit`

- Replacement for `init`
- Based on daemontools
- Features
  - Parallel startup
  - Automatic process restart
  - Supports per-user processes
  - Handles daemonization for processes
  - Ported to Linux, *BSD, Mac OS X, Solaris

# Linux/Cross-Platform - `runit`

- Daemons
  - `runit`
    - PID `1`, starts `runsvdir`
  - `runsvdir`
    - Repeatedly scans `/var/service/`, starts `runsv` process for each service directory found
  - `runsv`
    - For each service, runs `/var/service/`*`service`*`/run` script
    - Restarts service if necessary

# Linux/Cross-Platform - `runit`

- Enabling and disabling services
  - Determined by presence of service subdirectory
- Starting and stopping services
  - `sv { up | down }` *`service service...`*
- Obtaining service information
  - `sv status` *`service service...`*
- http://smarden.org/runit/

# Linux/Cross-Platform - Upstart

- Replacement for `init`
- Developed for Ubuntu 6.10, with an eye on portability to Debian and other Linux distros
- Design goals similar to Mac OS X `launchd`
  - In particular, support for dynamic hardware and network connectivity on desktops and laptops
  - "Event-based" `init`
- Future possibilities
  - Per-user processes
  - Scheduled processes (replace `cron`, `anacron`, `at`)
  - Replace `inetd`

# Linux/Cross-Platform - Upstart

- Features
  - Parallel startup
  - On-demand execution
    - Triggers: interprocess communication, network contact, device attachment, etc.
  - Dependency management
  - Automatic process restart

# Linux/Cross-Platform - Upstart

- Each service is in one of six states
  - Stopped - waiting for triggering event
  - Waiting - waiting for dependency
  - Starting - startup script running
  - Running - startup script completed successfully
  - Restarting - restart script running
  - Stopping - stop script running

# Linux/Cross-Platform - Upstart

- Enabling and disabling services
  - Determined by presence of job file in `/etc/event.d/`
- Starting and stopping services
  - Automatic, or
  - `{ start | stop }` *`service`*
- Obtaining service information
  - `status` *`service`*
  - `initctl list`
    - Lists all jobs, with states

# Linux/Cross-Platform - Upstart

- http://upstart.ubuntu.com/
- http://www.linux.com/print.pl?sid=06/09/18/1623244
- https://wiki.ubuntu.com/ReplacementInit
- http://upstart.ubuntu.com/doc/getting-started.html

# Linux/Cross-Platform - `init` Replacements

- Other Linux `init` replacements
  - SuSE 10 `startpar`, `insserv`
    - Dependency management, parallel startup
  - "Improve the Debian Boot Process"
    - http://initscripts-ng.alioth.debian.org/soc2006-bootsystem/deliverable1.html
      - Includes a survey of `init` replacements on other platforms
  - "Fedora New Init System"
    - http://fedoraproject.org/wiki/FCNewInit
    - Includes a survey of `init` replacements on other platforms

# Conclusion

- Lots of inventive work being done in the area of process-launching mechanisms, particularly replacements for traditional UNIX processes: `init`, `cron`, `inetd`
- The most interesting look to be SMF, `launchd`, and Upstart
- Unfortunately, due to licensing woes and entrenchment, there appears to be very little chance of these projects meeting on common ground, which means we'll be left with yet another seemingly unnecessary variance between UNIX flavors

# UNIX Process-Launching Mechanisms

Leon Towns-von Stauber, Occam's Razor
Seattle Area System Administrators Guild,
March 2007
http://www.occam.com/