



Solaris 10

The Best Just Got A Whole Lot Better

Jim Mauro

Senior Staff Engineer

Performance & Availability Engineering

Sun Microsystems Inc.



Agenda

- Solaris design principles
- Solaris past
 - Cool stuff you have have missed
- Solaris 10 headline grabbers
 - Solaris Containers (Zones)
 - Solaris Dynamic Tracing (dtrace)
 - Predictive Self Healing
 - Process Rights Management
 - Zetabyte File System (ZFS)
- Mixed slides & demo throughout

Why Solaris?

Just what is it that these Sun engineers are so excited about?

- What are the Solaris design principles?
- Why do they matter?
- Where has Solaris been?
- Where is Solaris today?
- Where is Solaris going?

Solaris Design Principles

Just what is it that makes Solaris so special?

- Reliability
- Performance
- Availability
- Manageability
- Serviceability
- Security
- Platform neutrality

Reliability

A Fundamental Solaris Design Principle

- Must be reliable *above all else*
- Must have the processes to develop and deliver *bug-free* software
- Must have the infrastructure to always debug *root-cause* from first failure

Performance

A Fundamental Solaris Design Principle

- Must *scale* with available hardware resources
- Must *not degrade* under increased load
- Must allow for *deterministic latency*
- Must be *as fast as, or faster than* the competition

Availability

A Fundamental Solaris Design Principle

- Must have the ability to *recover* from non-fatal hardware failure
- Must have the ability to *restart* from application failure

Manageability

A Fundamental Solaris Design Principle

- Must have powerful abstractions that *simplify* system management
- Must allow for the management of *individual* hardware resources

Serviceability

A Fundamental Solaris Design Principle

- Must be able to diagnose arbitrary problems in a production environment
- Must be able to diagnose problems in hardware, system software, and apps
- Must be able to diagnose both fatal and transient problems
- Must automate diagnosis whenever possible (or document methodologies)

Security

A Fundamental Solaris Design Principle

- Must be *designed-in* (not just an afterthought)
- Must be secure *out-of-the-box*
- Must be designed to *minimize exposure* to security breach

Platform Neutrality

A Fundamental Solaris Design Principle

- Must continue to be platform-neutral
- Must continue to develop low-level abstractions with multiple platforms in mind

Solaris Design Upshot

- The Solaris design principles are unified by a single goal: to *reduce the costs* in information technology
- Solaris *costs less*:
 - Solaris costs less to develop on
 - Solaris costs less to deploy
 - Solaris costs less to manage
- Solaris lowers costs through *innovation*

Solaris Past

- Solaris design principles are evident in features introduced in past versions
- While thoroughly documented, new Solaris features aren't always well communicated
- Worth examining a few long-available Solaris features
- Not meant as a tutorial; see the referenced manual pages

Observability – Integrated Tool Set

- Solaris has a rich tool-set for understanding process behavior
- Tools are implemented in terms of operations on files in “/proc” pseudofilesystem (see **proc(4)**)
- Tools adhere to the Unix philosophy: small, simple, numerous

Solaris Past: Serviceability

Integrated Toolset: **truss(1)**

- The Grande Dame of /proc-based tools
- Traces system calls, signals, faults
- *Many* options; see **truss(1)**
- Can trace machine faults (e.g. page faults) with “-m”
- Can trace function calls with “-u”
- Can be used to *stop* processes on events of interest; see “-T”, “-S”, “-M”, “-U”, **prun(1)**

Solaris Past: Serviceability

Integrated Toolset: P-tools

- **pstop(1)**: stop a process
- **prun(1)**: continue a stopped process
- **pflags(1)**: display process info
 - If stopped, displays reason for stopping
 - Displays each LWP's signal mask
 - Displays process data model

Solaris Past: Serviceability

Integrated Toolset: P-tools, cont.

- **pmap(1)**: displays process memory map
 - With “-s” prints pagesizes
 - With “-x” prints resident memory
- **pfiles(1)**: displays open file info
 - For files, gives fstat(2) info
 - For sockets, gives socket name and peer name
 - For doors, gives door server as process name and PID

Solaris Past: Serviceability

Integrated Toolset: P-tools, cont.

- **pstack(1)**: gives stack trace for each thread in a process *or* from a core file
- **pargs(1)**: prints arguments to a process
 - Can print environment vars (“-e”)
 - Can print auxiliary vector (“-x”)
- **psig(1)**: prints signal disposition

Solaris Past: Serviceability

Integrated Toolset: P-tools, cont.

- **nohup(1)** has “-p” option to have a running process (or group) ignore HUP
- **pgrep(1)** finds a process based on name, prints matching PIDs
 - Can also find processes based on a host of matching criteria
 - Can print long output (“-l”)
- **pkill(1)** = pgrep(1) + kill(1)

Solaris Past: Serviceability

Integrated Toolset: P-tools, cont.

- **ptree(1)** prints tree of processes, showing parent/child relationships
- **preap(1)** forces parent to wait(2), reaping zombie processes
- **prstat(1)** provides ongoing display of top CPU-consuming processes
 - Superset of “top” utility (thread, CPU aware) and *much* faster
 - *Many* options; see `prstat(1)`

Solaris Past: Serviceability

Integrated Toolset: P-tools, cont.

- **gcore(1)** grabs a core file from a running process
- **pldd(1)** displays a process's dynamic shared objects
- **pwdx(1)** prints a process's current working directory
- **pcred(1)** prints a process's credentials
- **pwait(1)** waits for a process to exit

Solaris Past: Manageability

Resource Management

- A *project* is an accounting entity with users, properties (see **project(4)**)
- A *task* is a group of processes in a project (see **newtask(1)**)
- *Fair-share scheduler* allows CPU shares to be assigned to projects; see **priocntl(1M)**)
- Resources can be grouped into *pools*; see **pooladm(1M)**)

Solaris Past: Manageability

Resource Management, cont.

- Projects can be assigned a pool
- Tasks can be dynamically bound to a pool; see **poolbind(1M)**
- Use of software resources can be limited on a per process, per task or per project basis; see **rctladm(1M)**
- Use of memory can be limited using **rcapd(1M)**

Solaris Past: Performance

libumem

- Solaris kernel memory allocator has long been industry-leading allocator:
 - Fastest, most scalable allocator
 - Run-time debugging options
- **libumem(3LIB)** is a port of the kernel memory allocator to user-level
- Library exports both `malloc(3C)` and `umem_cache_alloc(3MALLOC)` interfaces

Solaris Past: Performance

libumem, cont.

- LD_PRELOAD can be used to interpose libumem(3LIB) on default malloc(3C)
- Complete **mdb(1)** debugging support
- Optional debugging features detect duplicate frees, buffer corruption, overruns; see **umem_debug(3MALLOC)**
- Optional debugging feature enables memory leak detection from core file

Solaris Past: Performance

Real-time

- Solaris *is* a real-time operating system!
- Fixed priority, preemptive scheduling; see “Real-Time Class” in **priocntl(1M)**
- *Fully* preemptive kernel
- Fine-grained processor control
- High-resolution, per-CPU interval timers; see “CLOCK_HIGHRES” in **timer_create(3RT)**

Solaris Past: Platform neutrality

Solaris x86

- Solaris has supported x86 since Solaris 2.1 – that's over ten years!
- The Solaris source code is platform-neutral; very little code in Solaris is platform-specific
- Within the bounds of possibility on the platform, *all* Solaris features are implemented on both SPARC and x86

Solaris Present

- After Solaris 9, architectural trajectories from Solaris 2.0 were largely concluded
- Several different engineering teams began to pursue new, radical ideas
- These ideas have taken 2-3 years to productize; they were made available in Solaris Express
- Most are available in Solaris 10
 - ZFS is coming soon....

Solaris 10

The Headline Grabbers

- Solaris Containers (Zones)
 - Dynamic Resource Pools
 - Solaris Dynamic Tracing
 - Predictive Self Healing
 - Fault Management Architecture (FMA)
 - Service Management Facility (SMF)
 - Process Rights Management
 - Zettabyte Filesystem (ZFS)
- And much, much more...**

Resource Management

Resource Management

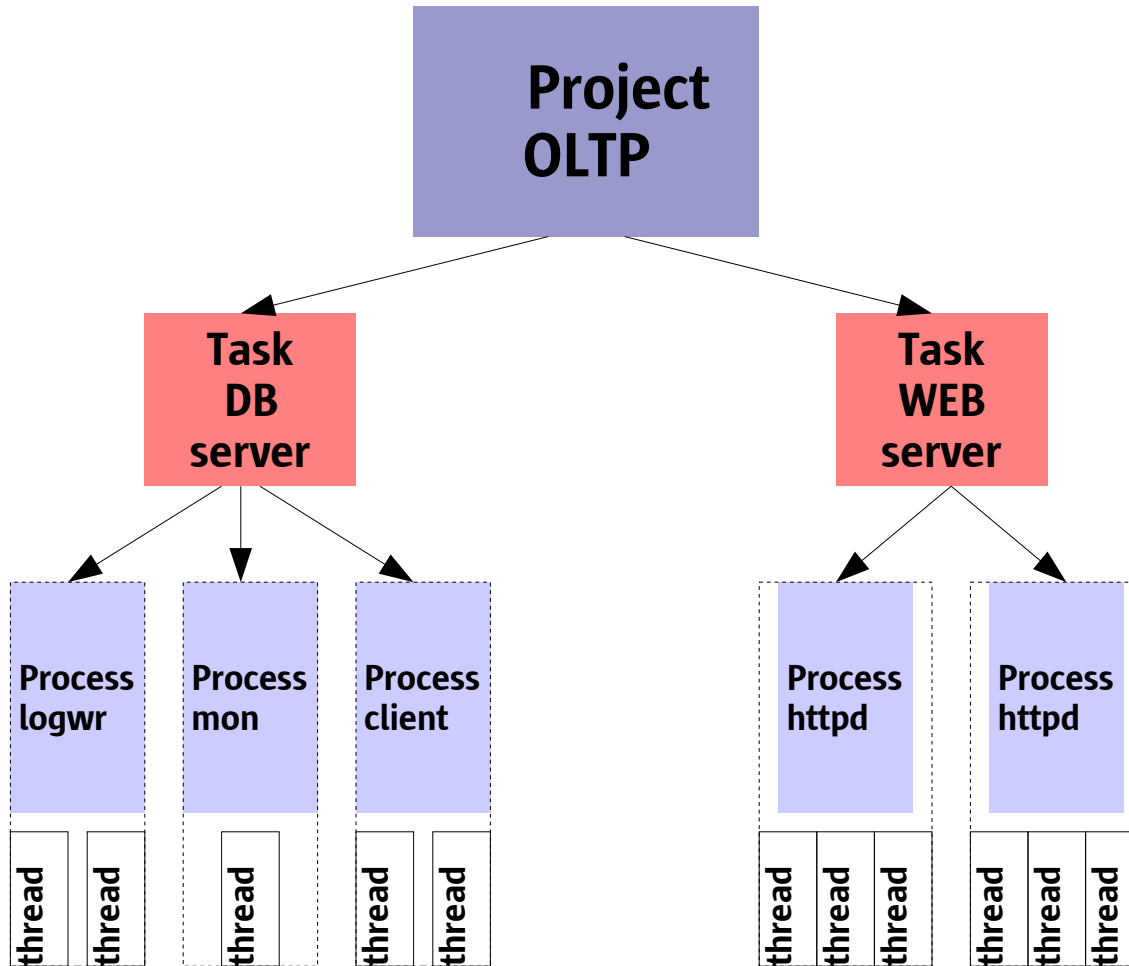
- Effectively allocating/partitioning hardware resources to applications/users.
 - Processors, memory, network IO, disk IO
 - Performance (SLAs), Security
- Solaris 8
 - Processor Sets
 - Grouping CPUs, binding processes/threads
 - Dynamic and easy

Resource Management

Solaris 9: Projects and Tasks

- A *project* is a resource management and control binding abstraction
 - defines users and attributes (see **project(4)**)
 - A tag bind and classify a workload
 - A user, group of users, or application
- A *task* is a group of processes acting together within a project
 - see **newtask(1)**

Project, Task, Process Hierarchy



Resource Management

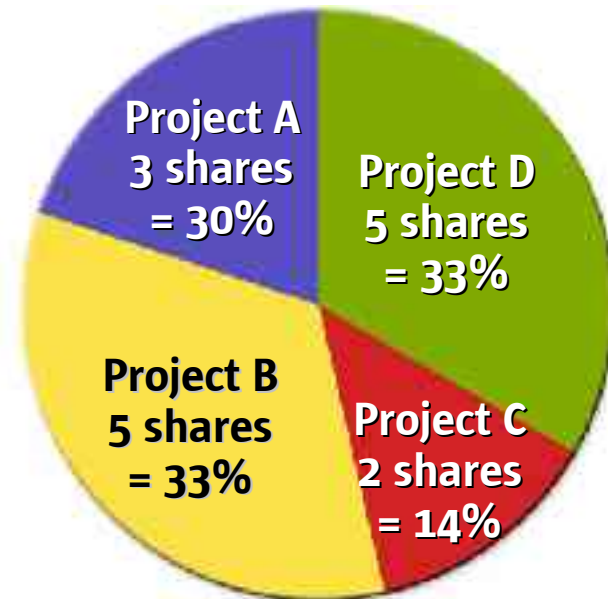
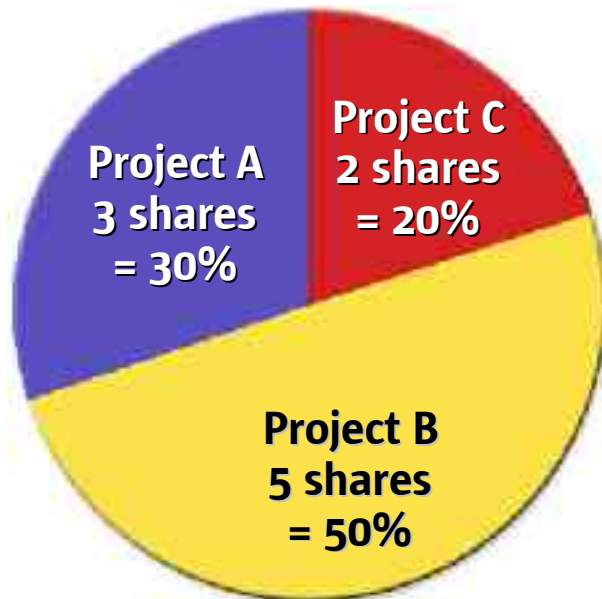
Solaris 9: Resource Pools

- Resources can be grouped into *pools*
 - initially CPUs (persistent processor sets)
 - memory later (use **racpd**(1M) for now)
- Projects can be assigned a pool
- Processes, tasks and projects can be dynamically bound to a pool
 - see **poolbind**(1M)

Resource Management

Solaris 9: Fair Share Scheduler

- Fair Share Scheduler (FSS)
 - allows CPU shares to be assigned to projects
 - otherwise behaves like TS (see `priocntl(1M)`)



Resource Management

Solaris 9 : Resource Controls

- See **rctladm(1M)**

```
project.cpu-shares  
task.max-lwps  
task.max-cpu-time  
process.max-cpu-time  
process.max-file-size  
process.max-data-size  
process.max-stack-size  
process.max-core-size  
process.max-file-descriptor  
process.max-address-space
```

Solaris 10 Resource Controls

- Additional resource controls

`process.max-sem-nsems`

`process.max-sem-ops`

`process.max-msg-qbytes`

`process.max-msg-messages`

`project.max-shm-ids`

`project.max-sem-ids`

`project.max-msg-ids`

`project.max-shm-memory`

`project.max-device-locked-memory`

Solaris 10 - Dynamic Resource Pools

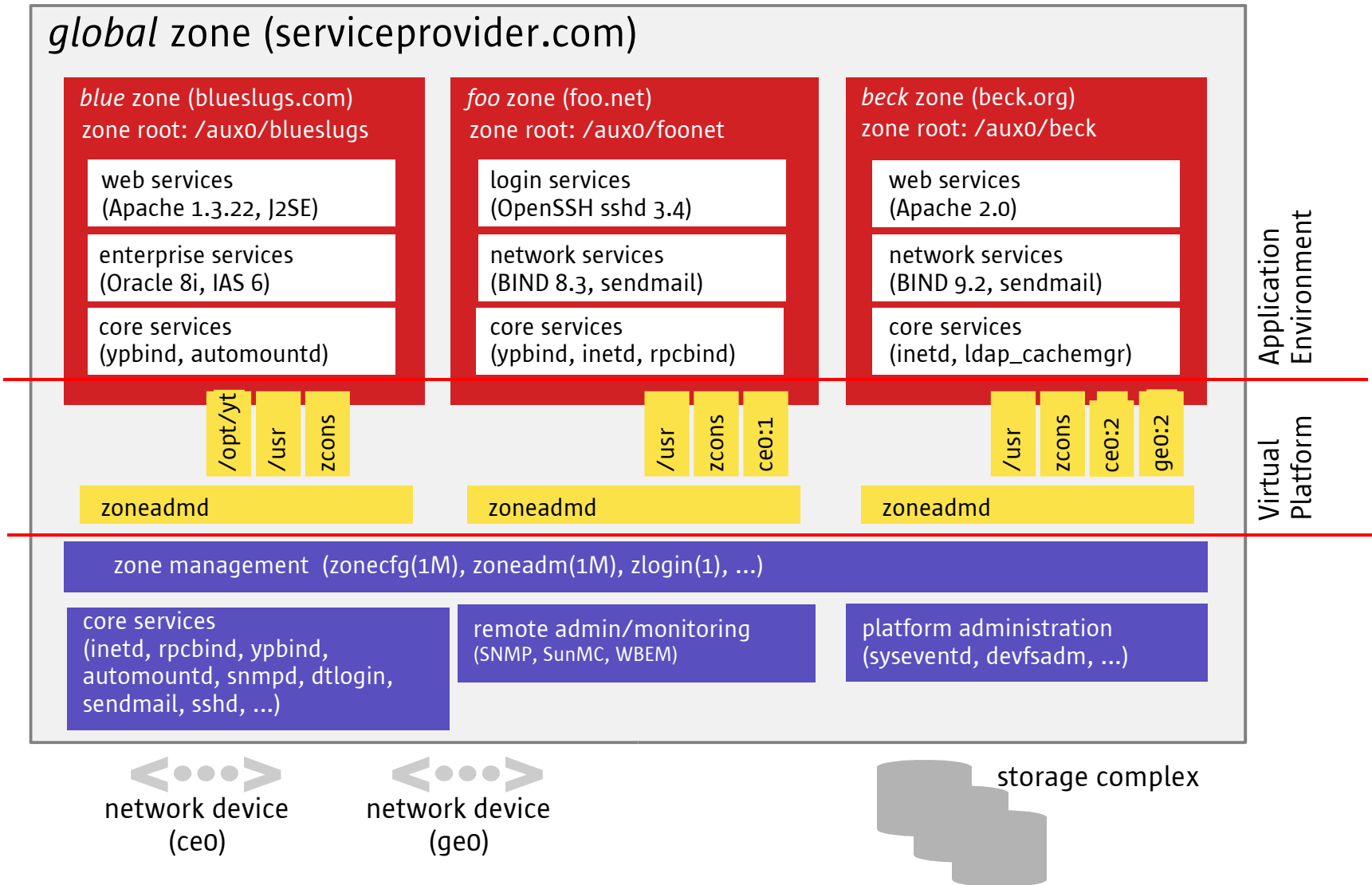
- Extend resource pools with added attribute to define performance thresholds
- New daemon, poold, monitors utilization and thresholds
- Dynamically re-allocate resources as needed

Solaris 10 - Containers

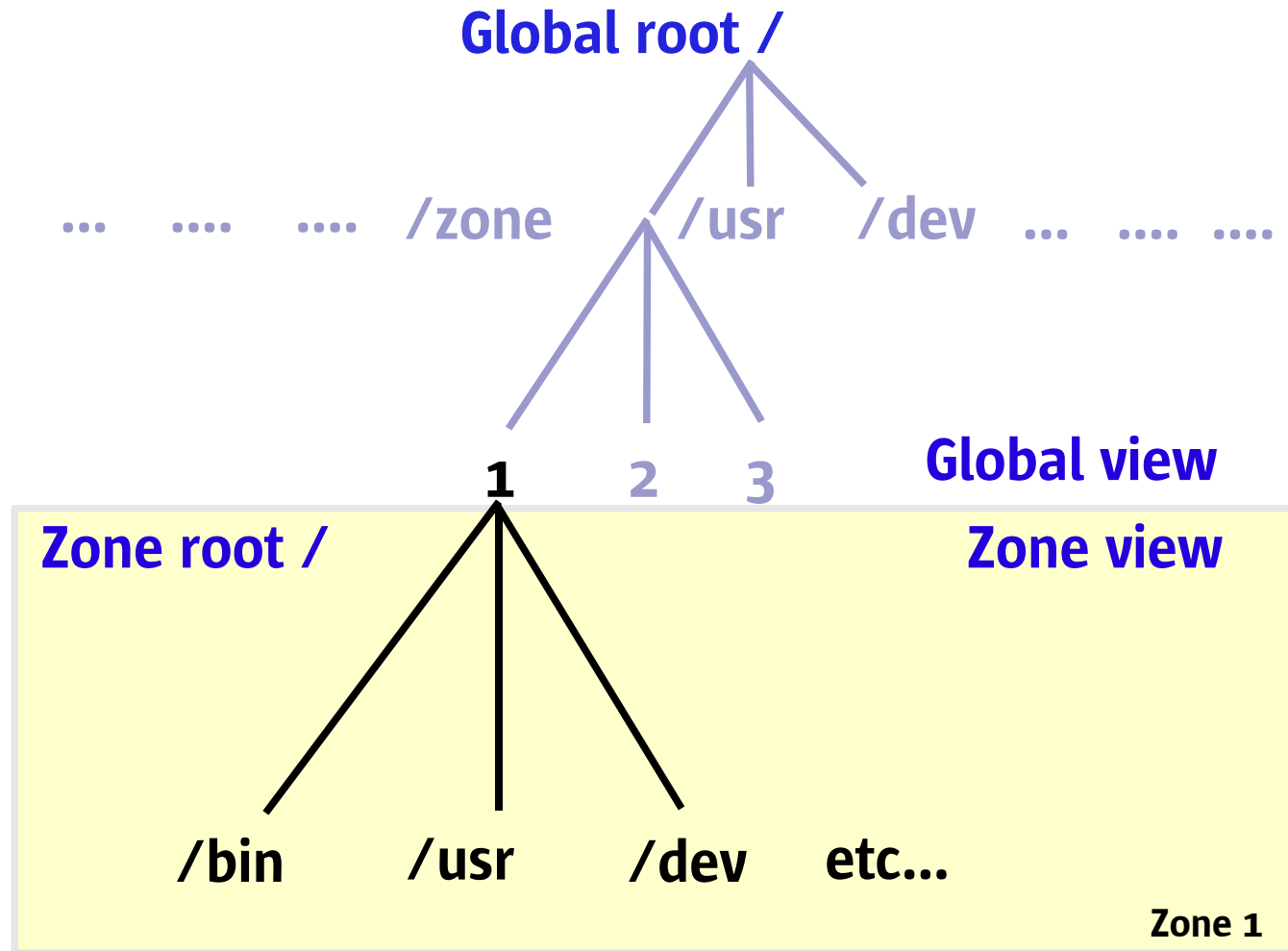
Taking Resource Management to the Next Level

- Zones
 - Virtualized operating system environments
 - Appear as booted instances of Solaris
 - Share a single instance of the Solaris kernel
 - Global zone sees all processes, but a subset runs in isolation within each additional zone
 - Fully intergrated with existing resource management framework

Zones Diagram



Zones – File Systems



Zones: Security

- Each zone has a security boundary around it
- Zones run with reduced privilege
- Important namespaces are isolated
- A compromised zone is not able to escalate its privileges; thus it cannot compromise the whole system or another zone
- Processes running in non-global zones (even as root) are not able to affect activity in other zones

Zones: Processes

- Certain system calls are not permitted or have restricted scope inside a zone
- From the global zone, all processes can be seen but control is privileged
- From within a zone, only processes in the same zone can be seen or affected
- `proc(4)` has been virtualized to only show processes in the same zone

Zones: File Systems

- Each zone is allocated its own root file system and cannot see that of others
- Processes cannot escape out of a zone, unlike `chroot` (2)
- File systems like `/usr` can be inherited in a read-only manner
- File systems such as `autofs` (4) have been virtualized per zone

Zones: Networking

- Single TCP/IP stack for the system as a whole so zones are shielded from most configuration like routing and devices
- Each zone has its own IP port space and is assigned IPv4/IPv6 addresses
- Applications can bind to `INADDR_ANY` and will only get traffic for that zone
- Zones cannot see the traffic of others

Zones: Identity

- Each zone controls its node name, RPC domain name, time zone and locale
- Each zone can use a different naming service such as DNS, LDAP and NIS
- Separate `/etc/passwd` files means that root can be delegated to the zone
- User ids may map to different names when domains differ (as with NFS now)

Zones: Interprocess Communications

- Expected IPC mechanisms such as System V IPC, STREAMS, sockets, `libdoor(3LIB)` and loopback transports are available inside a zone
- Key name spaces virtualized per zone
- Inter-zone communication is available using the network (software loopback)
- Global zone can setup rendezvous too

Zones: Devices

- Zones see an subset of “safe” pseudo devices in their `/dev` directory
- Devices like `/dev/random` are safe but others like `/dev/kmem` are not
- Zones can modify the permissions of their devices but cannot `mknod(2)`
- Physical device files like those for raw disks can be put in a zone with caution

Basic Commands

- `zonecfg(1M)` – configure a zone
- `zoneadm(1M)` – basic zone administration
 - Install or remove a zone
 - View status of a zone
 - Boot a zone
- `zlogin(1)` – enter a zone
 - Interactive
 - Non-interactive
 - Console
- `Zoneadmd`
 - Manages the virtual platform
 - One per local zone

zonecfg(1M) -z <zone> <cmd>

- **create** – create a zone configuration
- **delete** – delete a zone configuration
- **cancel** – cancel current operation
- **commit** – save configuration on stable storage
- **verify** – validate the zone configuration
- **add** <resource>
- **remove** <resource>
- **info**
- **help**

zonecfg(1M) Global Properties

- **zonepath**: location of the local zone root (in the global zone)
- **autoboot**: specifies whether the local zone is booted when the global zone boots
- **pool**: which resource pool zone should be bound to

zonecfg(1m) Resources

- `fs`: file system private to the local zone
- `inherit-pkg-dir`: directory which should have its associated packages “inherited” from the global zone
- `net`: network interface
- `device`: device
- `rctl`: resource control
- `attr`: generic attribute

Inherited Package Directories

- Four default `inherit-pkg-dir` resources provided
 - `/lib`, `/platform`, `/sbin`, `/usr`
 - Implemented via a read-only loopback file system mount which provides security as well as storage and virtual memory efficiencies
- `/opt` is good to add to this list, unless it will be configured differently than in the global zone

Example: Configuring a zone

```
# zonecfg -z zone1
zone1: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zone1> create
zonecfg:zone1> set zonepath=/export/home/zones/zone1
zonecfg:zone1> set autoboot=false
zonecfg:zone1> add net
zonecfg:zone1:net> set physical=e1000g0
zonecfg:zone1:net> set address=192.168.100.11/24
zonecfg:zone1:net> end
zonecfg:zone1> add inherit-pkg-dir
zonecfg:zone1:inherit-pkg-dir> set dir=/opt
zonecfg:zone1:inherit-pkg-dir> end
zonecfg:zone1> verify
zonecfg:zone1> commit
zonecfg:zone1> ^D
```

Configuration files

- /etc/zones
 - SUNWdefault.xml – defaults for zonecfg
 - <zone>.xml – zone configuration file
 - index – state information for all zones
- Zone DTD at
/usr/share/lib/xml/dtd/zonecfg.dtd.1

/etc/zones/zone1.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE zone PUBLIC "-//Sun Microsystems Inc//DTD Zones//EN"
    "file:///usr/share/lib/xml/dtd/zonecfg.dtd.1">
<!--
    DO NOT EDIT THIS FILE. Use zonecfg(1M) instead.
-->
<zone name="zone1" zonepath="/export/zones/zone1" autoboot="false">
  <inherited-pkg-dir directory="/lib"/>
  <inherited-pkg-dir directory="/platform"/>
  <inherited-pkg-dir directory="/sbin"/>
  <inherited-pkg-dir directory="/usr"/>
  <network address="192.168.100.11/24" physical="e1000g0"/>
  <inherited-pkg-dir directory="/opt"/>
</zone>
```

-

zoneadm(1M) -z <zone> <cmd>

- zoneadm(1M) is used by the global zone administrator to
 - **install** a new root file system for a configured zone
 - **list** zones and optionally their state
 - **verify** whether the configuration of an installed zone is semantically complete and ready to be installed
 - **boot** or **ready** an installed zone
 - **halt** or **reboot** a running zone
 - **uninstall** the root file system of an installed zone

Example: Installing a zone

```
global# zoneadm list -cv
```

ID	NAME	STATUS	PATH
0	global	running	/
-	zone1	configured	/
			export/home/zones/zone1

```
global# zoneadm -z zone1 install
```

```
Preparing to install zone <zone1>.
```

```
Creating list of files to copy from the global zone.
```

```
Copying <2144> files to the zone.
```

```
Initializing zone product registry.
```

```
Determining zone package initialization order.
```

```
Preparing to initialize <804> packages on the zone.
```

```
Initialized <804> packages on zone.
```

```
Zone <zone1> is initialized.
```

Understanding Complex Systems

Understanding Complex Systems

The Traditional Way - Many Tools But ...

- Difficult to *see* what is happening inside the kernel (and user-level applications)
- No way to *observe* behaviour across the entire software stack, inter-processes
- System/Process duality obscures reality
 - System-centric "xyzstat" utilities
 - Process-centric "/proc" tools
- Too much data to filter and correlate

Understanding Complex Systems

The Traditional Way - It Gets Worse ...

- Experiments are *hard* to conduct
 - Few iterations are possible
 - Theories are many, but often untestable
 - Premature conclusions become inevitable
- Methods are dictated by available data
 - *Pattern matching* instead of *problem solving*
 - *Rules-of-thumb* instead of *hard science*
- Established methods determine future tools and the data they provide

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

Slashdot post, Nov 2003

DTrace

The Solaris Dynamic Tracing Observability Revolution

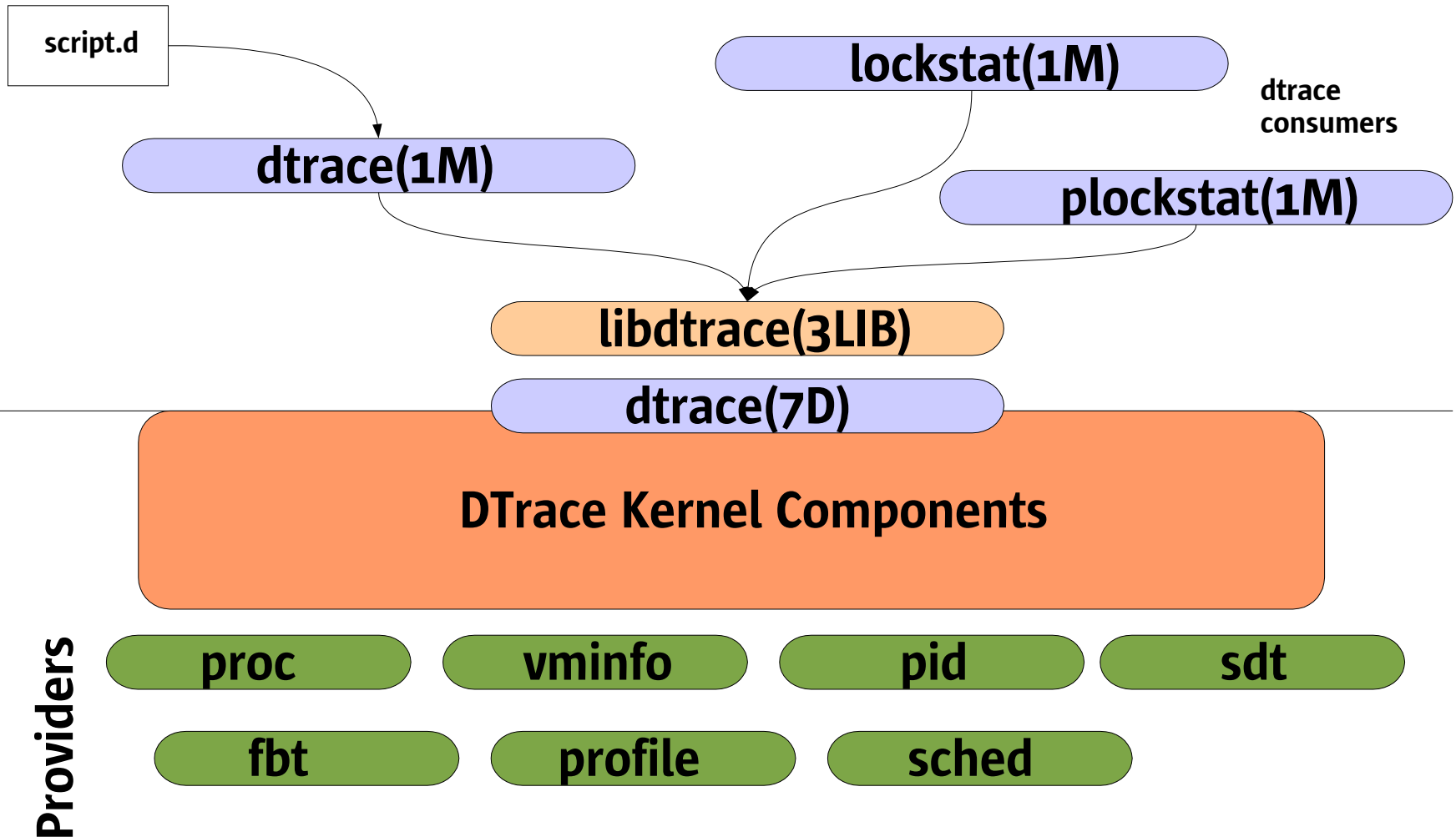
- Seamless, *global* view of the system from user-level thread to kernel
- Not reliant on pre-determined trace points, but *dynamic instrumentation*
- Data *aggregation* at source minimizes postprocessing requirements
- Built for live use on *production* systems

DTrace

The Solaris Dynamic Tracing Observability Revolution

- Ease-of-use and *instant gratification* engenders serious *hypothesis testing*
- Instrumentation directed by high-level control language (not unlike AWK or C) for easy scripting and command line use
- Comprehensive probe coverage and powerful data management allow for *concise* answers to *arbitrary* questions

DTrace – The Big Picture



DTrace

- Probes
 - A point of instrumentation
 - Each probe has a unique name and ID
- Providers
 - A abstraction of a kernel subsystem to dtrace
 - designed to make life easier, and improve level and type of information
- Consumers
 - Users of dtrace

DTrace (cont)

- Command line
 - `dtrace(1M)`, `lockstat(1M)`, `plockstat(1M)`
 - Scripts
 - The 'D' Language
 - Facilitates more sophisticated probing
 - probes, clauses (actions!) and predicates
 - 'C' language operators
 - Variables & Aggregations
- ```
#!/usr/sbin/dtrace -s
```

# DTrace

## The Solaris Dynamic Tracing Observability Revolution

- Not just for diagnosing problems
- Not just for kernel engineers
- Not just for service personnel
- Not just for application developers
- Not just for system administrators
- Serious fun
- Not to be missed!

# Predictive Self Healing

# Predictive Self Healing

- Combines Solaris 10's Fault Management Architecture (FMA) and Service Management Facility (SMF)
- Both of these software subsystems are new to Solaris 10
- Designed with a common goal – Improve system availability, reliability and serviceability through intelligent monitoring, detection, recovery, logging and messaging
- Ease-of-use also critical

# Fault Management Architecture

## Goals

- Improve Service Availability
- Create a single administrator & service model:
  - Associate every fault with a useful impact**
  - Associate every fault with a corrective action**
  - Automate self-healing and recovery**



# Fault Management Architecture

## Core Concepts

**Detection** reliably observe errors

**Data Capture** collect information

**Naming** encode using FMRI scheme

**Event Protocol** new protocol/facilities



# Fault Management Architecture

Core Concepts *Continued*

**Diagnosis** determine cause of errors

**Dependency** semantic relationships

**Action** report/respond, prevent faults

**History** persistently record events

# Traditional Error Handling

Complex, Time-Consuming, and Error-Prone

- Each component operates independently to handle and communicate errors
- Humans try to diagnose *fault*, *impact*, and appropriate *corrective action*



**WARNING:** /io-unit@fe0200000/sbi@0,0/dma@0,81000/esp@0,8000000 (esp0):  
Connected command timeout for Target 0.0

**NOTICE:** correctable error detected by pci0 (upa mid 1f) during DVMA  
read transaction AFSR=40f40000.1f800000 AFAR=00000000.a25b4000 ...

**[AFT0]** errID 0x0000004d.23105c04 Corrected Memory Error on U1004 is  
Intermittent  
**[AFT0]** errID 0x0000004d.23105c04 ECC Data Bit 14 was in error and  
corrected

# Fault Management Architecture

## User-friendly Messages

- Include impact and action statements in a consistent format
- All events are managed and coordinated through a single management service

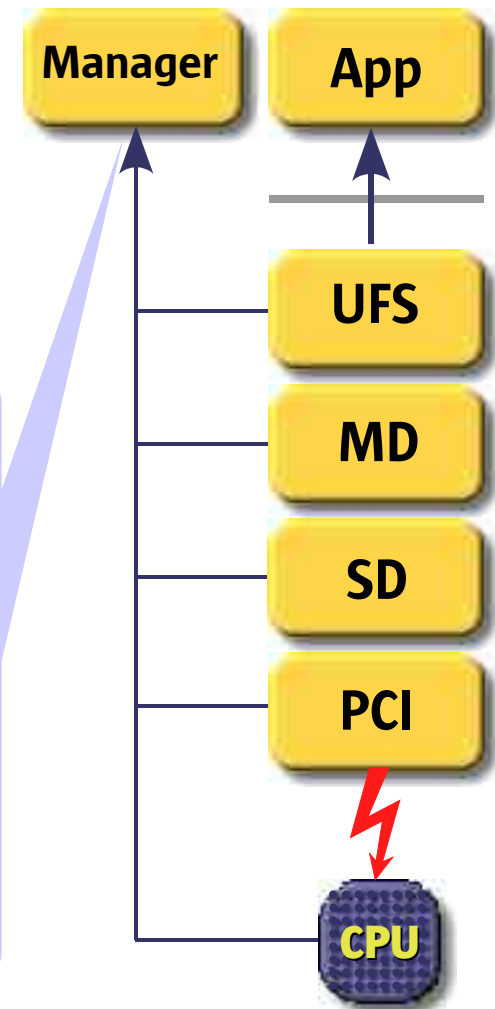
SUNW-MSG-ID: SF20000-W84N-KP3A-TF; TYPE: Fault, VER: 1, SEVERITY: Minor

AUTO-RESPONSE: Removal of the faulty memory resources has been initiated

IMPACT: Reduction in available memory resources

REQ-ACTION: A service call should be scheduled to inspect/replace the suspect components

DESC: A correctable memory data error occurred which has been diagnosed to be caused by a fault in a memory hardware component.



# Fault Management Architecture

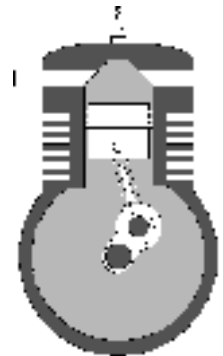
## Eversholt Diagnostic Engine

### Eversholt Fault Diagnosis Technology

- Algorithm for diagnosing faults
- Driven by a Fault Tree
- Specified in a language called Eversholt

### Other Components

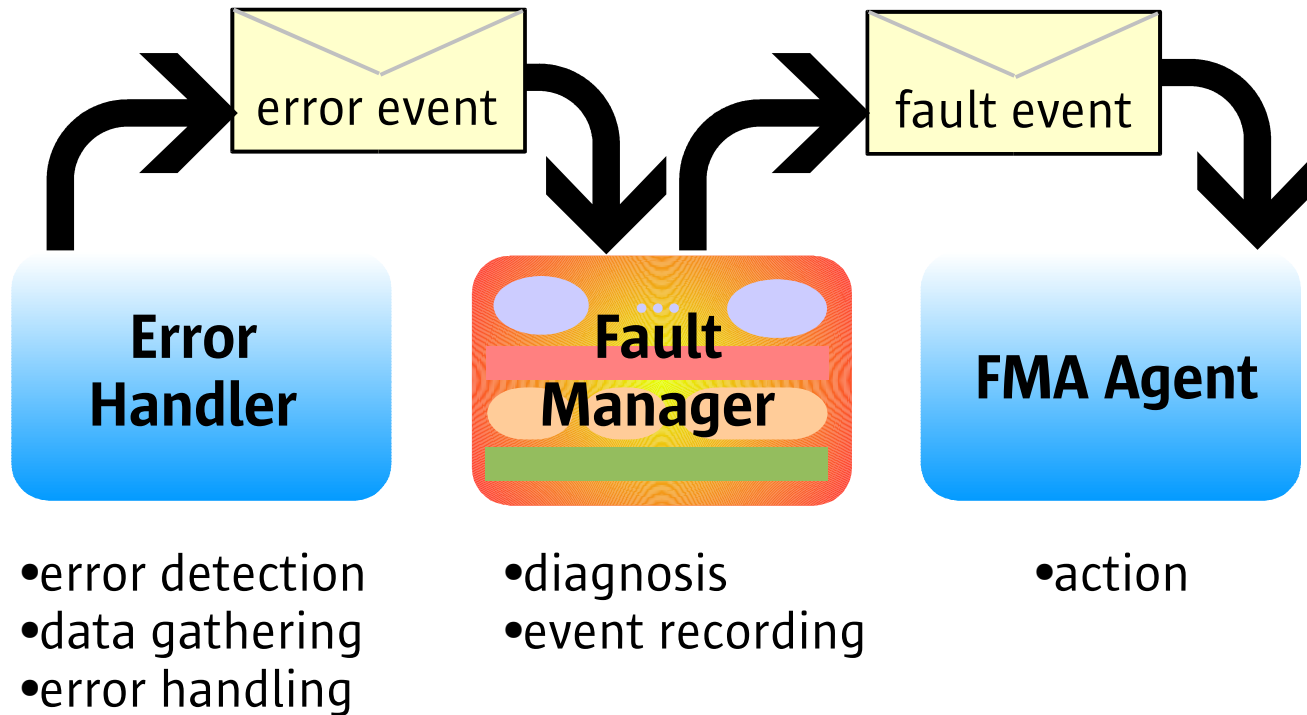
- Language compiler
- Prototype Diagnosis Engine
- Simulation environment



**Trivia:** Eversholt is a village in Bedfordshire, UK where the authors Emrys Williams and Andy Rudoff worked.

# Fault Management Architecture

## Fault Management Flow



# Fault Management Architecture

## Fault Management Tools

**fmadm(1M)** - view/manage activities

- View, load, unload, and reset modules
- View list of faulty resources (and more)

**fmdump(1M)** - view log files

- View time, UUID, and Message ID for fault log
- View time and Private event detail for error log
- Match events by UUID, class, time range

**fmstat(1M)** - view statistics

- View time and event statistics for all modules
- View Private bean counters for a given module



# Fault Management Architecture

Coming Soon...

**sun.com/msg/** (*message code*)

- Customer web-site will provide latest repair procedures for each diagnosis
- Links to information on latest FMA capabilities, updates, and plans
- No passwords – totally free access

sun.com/msg/[SF20000-W84N-KP3A-TF](http://sun.com/msg/SF20000-W84N-KP3A-TF)

SUNW-MSG-ID: [SF20000-W84N-KP3A-TF](http://sun.com/msg/SF20000-W84N-KP3A-TF); TYPE: Fault,  
VER: 1, SEVERITY: Minor

AUTO-RESPONSE: Removal of the faulty memory resources has been initiated



# Fault Management Example

```
/*
 * List of fault management administrative tasks provided by fmadm
 */

fmadm config

MODULE VERSION DESCRIPTION
cpumem-diagnosis 1.0 UltraSPARC-III CPU/Memory Diagnosis
cpumem-retire 1.0 CPU/Memory Retire Agent
fmd-self-diagnosis 1.0 Fault Manager Self-Diagnosis
syslog-msgs 1.0 Syslog Messaging Agent
```

# Fault Management Example

```
/*
 * Inject a kernel instruction uncorrectable ecache write-back
 error
 */
mtst kiwdu
/*
 * New fault diagnosis message produced by fmd(1M) upon problem
 diagnosis
 * by the cpumem_diagnosis module
 */
SUNW-MSG-ID: SUN4U-8000-AC, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Tue Apr 13 13:51:07 PDT 2004
PLATFORM: SUNW,Sun-Fire-V210, CSN: -, HOSTNAME: barback
SOURCE: cpumem-diagnosis, REV: 1.0
EVENT-ID: dd022489-9053-6aa5-a562-adf1959a7b0a
DESC: The number of errors associated with this CPU has exceeded
 acceptable levels. Refer to http://sun.com/msg/SUN4U-8000-AC
 for more information.
AUTO-RESPONSE: An attempt will be made to remove the affected CPU
 from service.
IMPACT: Performance of this system may be affected.
REC-ACTION: Schedule a repair procedure to replace the affected
 CPU. Use fmdump -v -u <EVENT_ID> to identify the CPU.
```

# Example: SUN4U-8000-AC

Article for Message ID: SUN4U-8000-AC

CPU errors exceeded acceptable levels

Type

Fault

Severity

Major

Description

The number of errors associated with this CPU has exceeded acceptable levels.

Automated Response

The fault manager will attempt to remove the affected CPU from service.

Impact

System performance may be affected.

Suggested Action for System Administrator

Schedule a repair procedure to replace the affected CPU, the identity of which can be determined using `fmddump -v -u <EVENT_ID>`.

Details

The Message ID: SUN4U-8000-AC indicates diagnosis has determined that a CPU is faulty. The Solaris fault manager arranged an automated attempt to disable this CPU. The recommended action for the system administrator is to contact Sun support so a Sun service technician can replace the affected component.

# Fault Management Example

```
/*
 * Action taken as prescribed. We see that CPU 0 has been diagnosed as
 * as "faulty".
 */

fmdump -v -u dd022489-9053-6aa5-a562-adf1959a7b0a
TIME UUID SUNW-MSG-ID
Apr 13 13:51:07.2242 dd022489-9053-6aa5-a562-adf1959a7b0a SUN4U-8000-AC
100% fault.cpu.ultraSPARC-III.l2cachedata
FRU: legacy-hc:///component=MB
rsrc: cpu:///cpuid=0/serial=130C3562148

fmadm faulty
STATE RESOURCE / UUID

faulted cpu:///cpuid=0/serial=130C3562148
dd022489-9053-6aa5-a562-adf1959a7b0a

```

# Fault Management Example

```
/*
 * In response to the diagnosis, our cpumem-retire agent has proactively
 * taken the faulty CPU out of the system until a repair action has
 * taken place.
 * System is up and running while a repair is scheduled for
 * a convenient time. For DR-capable systems, repair activities can
 * occur without service outage.
 */
psrinfo
0 faulted since 04/13/2004 13:54:38
1 on-line since 04/13/2004 13:54:02
/*
 * CPU has been repaired and is now ready to be re-integrated into the
 * the system
 */
fmadm repair cpu:///cpuid=0/serial=130C3562148
fmadm: recorded repair to cpu:///cpuid=0/serial=130C3562148
fmadm faulty
STATE RESOURCE / UUID

```

# Fault Management Example

```
/*
 * Additional error information related to the fault can be obtained from
 * the ereport log.
 */
fmdump -V
TIME UUID SUNW-MSG-ID
Apr 13 13:45:59.8543 3a073a20-1e97-4f7e-dfd8-9546f4889ff8 SUN4U-8000-AC

TIME CLASS ENA
Apr 13 13:45:58.7736 ereport.cpu.ultraSPARC-IIIi.wdu 0x2fb5f956a0400001

nvlist version: 0
 version = 0x0
 class = list.suspect
 uuid = 3a073a20-1e97-4f7e-dfd8-9546f4889ff8
 code = SUN4U-8000-AC
 diag-time = 1081889159 852427
 de = (embedded nvlist)
...
```

# Fault Management Example

```
fmdump -eV
TIME CLASS
Apr 13 13:45:58.773699750 ereport.cpu.ultraSPARC-IIIi.wdu
nvlist version: 0
 class = ereport.cpu.ultraSPARC-IIIi.wdu
 ena = 0x2fb5f956a0400001
 detector = (embedded nvlist)
 nvlist version: 0
 version = 0x0
 scheme = cpu
 cpuid = 0x0
 cpumask = 0x24
 serial = 0x130c3562148
 (end detector)

 afsr = 0x200000000c
 afar-status = 0x1
 afar = 0x102ab89000
 pc = 0x12877c4
 tl = 0x0
 tt = 0x63
 privileged = 1
 multiple = 0
 syndrome-status = 0x1
 syndrome = 0xc
 l2-cache-ways = 0x4
 syndrome = 0xc
 l2-cache-ways = 0x4
...
```

# Service Management Facility

- Problem:
  - Ad-hoc mechanisms for managing services:
    - /etc/\* files
    - /etc/rc\*.d/\* scripts
    - multi-service daemons (e.g. init, inetd)
  - No operating system support for service-based management
  - Little or no dependency checking
  - Simplistic or no service resilience
  - “Thousands of text files” not a design standard

# Service Management Facility

- Solution:
  - Framework for service management
    - Repository for configuration data
    - Define dependencies and relationships between apps
    - Administrative enable/disable controls
    - Fine-grained access control
  - Integrated with fault management architecture
    - Automated single-node restart
  - Easy service/application installation
    - Including support for legacy applications

# smf ( 5 ) Service

- What's a service?
  - Abstract description of a long-lived software object
  - Each instance of a service has a well-defined state and a well-defined error boundary [process contract]
  - Each service defines “methods” and “dependencies”
    - Start, stop, refresh, etc.; interservice relationships
- A consistent specification
  - Can state dependency characteristics (grouping, restart)
  - Generic restart facility provided by default; customized restart capabilities available to vendor
- Admins can get a meaningful *system* view

# SMF Identifier

- FMRI – Fault Management Resource Identifier
  - `svc://localhost/network/login:rlogin`
    - Absolute path
  - `svc://network/login:rlogin`
    - Path relative to local machine
  - `network/login:rlogin`
  - `rlogin`
  - `svc://localhost/system/cryptosvc:default`
  - `svc://system/cryptosvc:default`
  - `lrc:/etc/rc3_d/S90samba`

# Service States

- **online**
  - instance enabled and started
- **offline**
  - instance enabled but not yet running
    - transitory state
- **disabled**
  - instance not enabled and not running
- **maintenance**
  - instance encountered an error, manual intervention required

# Service States

- **legacy\_run**
  - used by legacy services
  - service can be observed by not managed by SMF
- **degraded**
  - instance enabled but running at limited capacity
- **uninitialized**
  - initial state for all services before their configuration has been read

## svcs ( 1 ) in action

- Pre-SMF “service” characteristics
  - what does sendmail depend on?
  - what depends on sendmail?
  - which processes constitute the mail service?
  - .....

```
$ pgrep -lf sendmail
708 /usr/lib/sendmail -bd -q15m
685 /usr/lib/sendmail -Ac -q15m

$ find /etc -name *sendmail* -print
/etc/rc0.d/K36sendmail
/etc/rc1.d/K36sendmail
/etc/rc2.d/S88sendmail
/etc/rcS.d/K36sendmail

$ less /etc/rc2.d/S88sendmail
....
```

# svcs ( 1 ) in action

- List active instances, sorted by state, time
- Show dependencies (-d) and dependents (-D)
- Show member processes (-p), additional details (-v)

```

$ svcs
STATE STIME FMRI
....
online 18:18:30 svc:/network/http:apache
online 18:18:29 svc:/network/smtp:sendmail
....
$ svcs -p network/smtp:sendmail
STATE STIME FMRI
online 18:18:29 svc:/network/smtp:sendmail
 18:18:29 100180 sendmail
 18:18:29 100181 sendmail

$ svcs -v network/smtp:sendmail
STATE NSTATE STIME CTID FMRI
online - 18:18:29 21 svc:/network/smtp:sendmail

$ svcs -d network/smtp:sendmail
STATE STIME FMRI
online 18:17:44 svc:/system/identity:domain
online 18:17:52 svc:/network/service:default
....

```

# svcs ( 1 ) in action

- List active instances, sorted by state, time
- Show dependencies (-d) and dependents (-D)
- Show member processes (-p), additional details (-v)

```
$ svcs -D network/physical
STATE STIME FMRI
disabled Nov_24 svc:/network/dns/client:default
disabled Nov_24 svc:/network/dns/server:default
disabled Nov_24 svc:/network/rarp:default
disabled Nov_24 svc:/network/rpc/bootparams:default
disabled Nov_24 svc:/network/slp:default
disabled Nov_24 svc:/network/shell:kshell
online Nov_24 svc:/application/print/cleanup:default
online Nov_24 svc:/system/identity:node
online Nov_24 svc:/system/identity:domain
online Nov_24 svc:/network/initial:default
online Nov_24 svc:/milestone/single-user:default
online Nov_24 svc:/network/inetd:default
online Nov_24 svc:/network/nfs/client:default
online Nov_24 svc:/network/shell:tcp
online Nov_24 svc:/network/shell:tcp6only
online Nov_24 svc:/network/nfs/server:default
```

```
$
```

## svcs ( 1 ) in action

- Diagnose instances in unusual state (-x)

```
$ svcs -x
svc:/application/print/server:default (LP Print Service)
 State: disabled since Thu 30 Sep 2004 01:14:16 PM PDT
 Reason: Disabled by an administrator.
 See: http://sun.com/msg/SMF-8000-05
 See: lpsched(1M)
 Impact: 1 service is not running.

svc:/system/metainit:default (Solaris Volume Manager(SVM) Init service.)
 State: maintenance since Thu 30 Sep 2004 01:14:16 PM PDT
 Reason: Completes a dependency cycle.
 See: http://sun.com/msg/SMF-8000-HP
 See: metainit(1M)
 Impact: 0 services are not running.
```

# svcs ( 1 ) in action

- List service details (-l)

```
$ svcs -l network/smtp:sendmail
fmri svc:/network/smtp:sendmail
enabled true
state online
next_state none
restarter svc:/system/svc/restarter:default
contract_id 46
dependency require_all/refresh file://localhost/etc/nsswitch.conf (-)
dependency require_all/refresh file://localhost/etc/mail/sendmail.cf (-)
dependency optional_all/none svc:/system/system-log (online)
dependency require_all/refresh svc:/system/identity:domain (online)
dependency require_all/refresh svc:/milestone/name-services (online)
dependency require_all/none svc:/network/service (online)
dependency require_all/none svc:/system/filesystem/local (online)

$ svcs -l system/metainit
fmri svc:/system/metainit:default
name Solaris Volume Manager(SVM) Init service.
enabled false
state maintenance
next_state none
restarter svc:/system/svc/restarter:default
dependency require_all/none svc:/system/filesystem/minimal (online)
dependency require_all/none svc:/system/identity:node (online)
```

# svcadm ( 1M ) in action

- pre-SMF disabling of a “service”

```
$ pgrep -lf sendmail
708 /usr/lib/sendmail -bd -q15m
685 /usr/lib/sendmail -Ac -q15m

$ pkill sendmail

.....reboot.....

$ pgrep -lf sendmail
355 /usr/lib/sendmail -bd -q15m
276 /usr/lib/sendmail -Ac -q15m
```

- No log of why sendmail went down
- Reboot will restart sendmail, is this desired?

```
$ mv /etc/rc2.d/S88sendmail /etc/rc2.d/no_S88sendmail

.....apply patch/upgrade.....

$ ls /etc/rc2.d/S88sendmail
/etc/rc2.d/S88sendmail
```

# svcadm ( 1M ) in action

- Enable, disable, refresh, restart service instances
- Mark in special states (maintenance, degraded)

```
$ grep ambreesh /etc/user_attr
ambreesh:::auths=solaris.smf.modify,solaris.smf.manage
```

```
$ svcs -a network/http:apache
STATE STIME FMRI
uninitialized 19:17:33 svc:/network/http:apache
```

```
$ svcadm enable network/http:apache
STATE STIME FMRI
online 19:19:01 svc:/network/http:apache
```

```
$ vi /etc/apache/httpd.conf
```

```
$ svcadm refresh network/http:apache
```

```
$ svcs -a network/http:apache
STATE STIME FMRI
online 19:19:33 svc:/network/http:apache
```

```
$ svcadm disable network/http:apache
```

```
$ svcs -a network/http:apache
STATE STIME FMRI
disabled 19:20:07 svc:/network/http:apache
```

# Service Manifest

- Each package delivering services does so via a “service manifest”
- xml file containing description of service
  - `/var/svc/manifest`
  - Dependencies on other services and methods for service instance start/stop/refresh
  - Default properties and “service template”, which provides support for administrative apps via
    - Localized property descriptions
    - Links to documentation
    - Soon: meaningful property values (valid ranges, definitions, etc.)

# coreadm ( 1M ) service description

```
<service name='system/coreadm' type='service' version='1'>
 <create_default_instance enabled='false' />
 <single_instance />
 <dependency name='usr' grouping='require_all'
 restart_on='none' type='service'>
 <service_fmri value='svc:/system/filesystem/minimal' />
 </dependency>
 <exec_method type='method' name='start'
 exec='/usr/bin/coreadm -u' timeout_seconds='3' />
 <exec_method type='method' name='stop'
 exec=':true' timeout_seconds='0' />
 <property_group name='startd' type='framework'>
 <propval name='duration' type='astring' value='transient' />
 </property_group>

 <stability value='Unstable' />
 <template>
 <common_name><loctext xml:lang='C'>
 System-wide core file configuration service.
 </loctext></common_name>

 <documentation>
 <manpage title='coreadm' section='1M'
 manpath='/usr/share/man' />
 </documentation>
 </template>
</service>
```

# SMF repository

- All manifests stored in persistent, transaction-based repository
  - Transactions/snapshots allow “undo”, rollback to safe configuration
  - Repository can be local, in directory [later], or mixed [later]
- **NOT** a giant registry: mainly svc mgmt properties
- Data imported either at boot time from manifests or via `svccfg import`
- `svccfg(1M)` used to manipulate repository data
- `svccprop(1M)` used to view repository data
- `svc.configd(1M)` is the repository deamon

# svccfg ( 1M ) in action

- View properties of a service

```
$ svccfg
svc:> list
....
network/rpc/nisplus
....
platform/i86pc/kdmconfig
network/fs/tcp6
svc:> select network/ssh
svc:/network/ssh> select default
svc:/network/ssh:default> listprop
general framework
general/package astring SUNWsshdr
general/enabled boolean true
restarter framework NONPERSISTENT
restarter/contract count 280
restarter/start_pid count 7756
restarter/auxiliary_state astring none
restarter/next_state astring none
restarter/state astring online
restarter/state_timestamp time 1101530796.234165000
```

# svccfg(1M) in action

- Add a new property “myprop”

```
svc:/network/ssh:default> setprop general/myprop=astring:"demoval"
svc:/network/ssh:default> listprop
general framework
general/package astring SUNWsshdr
general/enabled boolean true
general/myprop astring demoval
restarter framework NONPERSISTENT
restarter/contract count 280
restarter/start_pid count 7756
restarter/auxiliary_state astring none
restarter/next_state astring none
restarter/state astring online
restarter/state_timestamp time 1101530796.234165000
svc:/network/ssh:default> delprop general/myprop
svc:/network/ssh:default> listprop
general framework
general/package astring SUNWsshdr
general/enabled boolean true
restarter framework NONPERSISTENT
restarter/contract count 280
restarter/start_pid count 7756
restarter/auxiliary_state astring none
restarter/next_state astring none
restarter/state astring online
restarter/state_timestamp time 1101530796.234165000
```

# SMF Components

- Master Restarter

- `svc.startd(1M)`

- responsible for starting and restarting services
    - starts services when their dependencies are met
    - restarts failed services
    - shuts down services when dependencies no longer met

```
$ pgrep -lf ssh
7759 /usr/lib/ssh/sshd
```

```
$ pkill ssh
```

```
$ pgrep -lf ssh
8635 /usr/lib/ssh/sshd
```

# SMF Components

- Delegated Restarter
  - restarter for a related set of services

# SMF Components

- Delegated Restarter
  - `inetd(1M)` the only one so far
    - responsible for internet services
    - `/etc/inet/inetd.conf` now deprecated
    - `inetadm(1M)` manages internet services
    - `inetconv(1M)` converts `inetd.conf` into `inetd`

```
$ svcs -R network/inetd:default
STATE STIME FMRI
disabled Sep_30 svc:/network/rpc/meta:tcp
....
online Oct_01 svc:/network/rpc/ttdbserver:tcp6
online Oct_03 svc:/network/shell:tcp
online 16:58:06 svc:/network/finger:default
online 19:11:06 svc:/network/rpc/rstat:udp
online 23:08:21 svc:/network/rpc/rusers:udp
offline Sep_30 svc:/application/print/rfc1179:default
$ svcs -R network/inetd:default | wc -l
66
```

# inetadm(1M) in action

- List internet services and view/modify their properties

```

$ inetadm
ENABLED STATE FMRI
enabled online svc:/network/rpc/gss:ticotsord
disabled disabled svc:/network/tname:default
enabled online svc:/network/security/ktkt_warn:ticotsord
enabled online svc:/network/telnet:default
.....
disabled disabled svc:/network/apocd/udp:default
disabled disabled svc:/network/uucp:default
disabled disabled svc:/network/security/krb5_prop:tcp
enabled online svc:/network/rpc-100235_1/rpc_ticotsord:ticotsord
enabled online svc:/network/rpc-100424_1/rpc_ticotsord:ticotsord
enabled online svc:/network/rpc-100083_1/rpc_tcp:tcp
enabled online svc:/network/rpc-100083_1/rpc_tcp:tcp6
enabled online svc:/network/rpc-100068_2-5/rpc_udp:udp
enabled online svc:/network/rpc-100068_2-5/rpc_udp:udp6
enabled online svc:/network/fs/tcp6:default

$inetadm -e network/tname

$inetadm | grep tname
enabled online svc:/network/tname:default

```

# inetadm(1M) in action

- List internet services and view/modify their properties

```
$ inetadm -l network/telnet
SCOPE NAME=VALUE
 name="telnet"
 endpoint_type="stream"
 proto="tcp6"
 isrpc=FALSE
 wait=FALSE
 exec="/usr/sbin/in.telnetd"
 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=FALSE
default tcp_wrappers=FALSE
```

# inetadm(1M) in action

- List internet services and view/modify their properties

```
$ inetadm -m network/telnet tcp_trace=TRUE
```

```
$ inetadm -l network/telnet
```

```
SCOPE NAME=VALUE
 name="telnet"
 endpoint_type="stream"
 proto="tcp6"
 isrpc=FALSE
 wait=FALSE
 exec="/usr/sbin/in.telnetd"
 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
 tcp_trace=TRUE
default tcp_wrappers=FALSE
```

# Boot process

- milestones analogous to run-levels
  - S milestone/single-user:default
  - milestone/name-service
  - 2 milestone/multi-user:default
  - 3 milestone/multi-user-server:default
- Boot without any services enabled, then enable all services

```
ok boot -m milestone=none
```

```
login as root
```

# Boot process

- Verbose boot available at boot prompt
  - Persistently verbose? Set options/logging to

`verbose on system/svc/restarter:default`

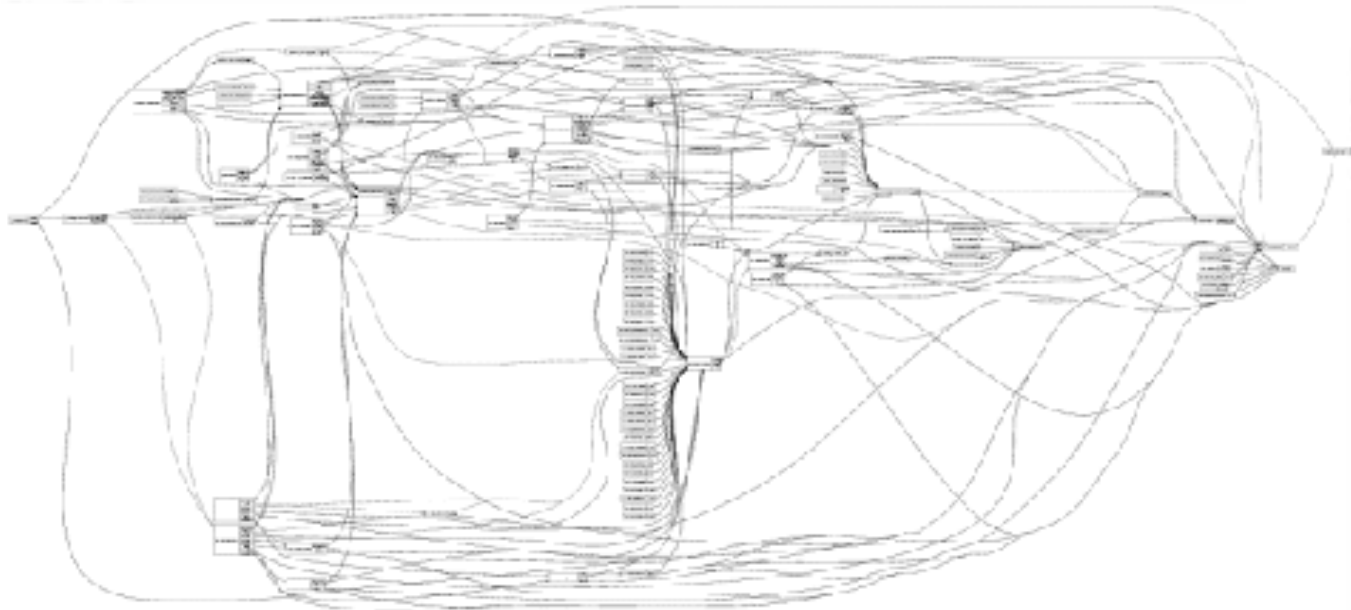
```
Select (b)oot or (i)nterpreter: b -mverbose
SunOS Release 5.10 Version smf-mdb-on10 32-bit
Copyright 1983-2004 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.
DEBUG enabled
WARNING: consconfig: could not find driver for screen device /isa/display@1,3b0
[network/loopback:default starting (Loopback network interface)]
[network/pfil:default starting (pfil)]
[system/filesystem/root:default starting (Root filesystem mount)]
[network/physical:default starting (Physical network interfaces)]
Oct 6 12:21:15/14: system start time was Wed Oct 6 12:21:10 2004
[system/filesystem/usr:default starting (/usr and / mounted read/write)]
[system/identity:node starting (system identity (nodename))]
Hostname: trolls-b10
[system/device/local:default starting (Standard Solaris device configuration.)]
[system/filesystem/minimal:default starting (Local filesystem mounts)]
[milestone/devices:default starting (Device configuration milestone.)]
[system/identity:domain starting (system identity (domainname))]
[system/cryptosvc:default starting (Cryptographic services)]
[system/manifest-import:default starting (Service manifest import)]
[system/sysevent:default starting (System event notification service.)]
NIS domain name is mpklab.sfbay.sun.com
[system/coreadm:default starting (System-wide core file configuration service)]
....
```

# Boot process

- `init` reads `/etc/inittab`

- `smf::sysinit:/lib/svc/bin/svc.startd >/dev/msglog 2<>/dev/msglog </dev/console`

- starts up master restarter



# Predictive Self Healing

- Sophisticated and robust software infrastructure for proactively dealing with fault and error events
- Fault management integrated with service management
- Faults, and the services they effect, can be correlated, allowing for restart/recovery of services impacted by fault events

# Process Rights Management

# Process Rights Management

## Going back to our Roots

- Problem:
  - Current “all or nothing” privilege model leads to security problems
  - Applications needing only a few privileges need to run as root (network daemons)
  - No way to limit root's privileges
  - No easy way for non-root users to perform privileged operations

# Process Rights Management

## Going back to our Roots

- Solution:
  - Fine-grained privileges allow apps and users to run with just the privileges they need
  - Currently, ~40 privileges (see **privileges(5)**)
  - By default, root has all privileges, but now even root can be restricted
  - Privileges for a running process can be viewed or changed with **ppriv(1)**
  - Privileges can be changed for a user by modifying **user\_attr(4)**

# Redefining Storage

# Redefining Storage

## Beyond Volume Management

- Problem:
  - Today's volume management solutions and filesystems are not designed to cope with the complexity or scale of current storage needs

# Redefining Storage

## Beyond Volume Management

- Solution:
  - New filesystem eliminates concept of volumes
    - Allocation from shared storage pools
  - Massive storage capacity (128 bit)
    - A "Zettabyte" filesystem
  - Checksums on all data and metadata
  - Always consistent (no fsck)
  - Performance rips!

# FireEngine – New TCP/IP Architecture

- Why?
  - Faster networks
  - Increased net services
  - Architecture getting a little long-in-the-tooth
- Analysis of volume workloads
  - Transactional
  - WEB
  - Tier 1

# FireEngine

- Network throughput
- Connection setup/teardown
- First byte latency
- Connection & CPU scalability
- Efficiency

# The Evolving Threads Model

- Solaris 2.x – Solaris 7
  - Two level model (n x m)
    - N user threads multiplexed onto M LWPs
- Solaris 8
  - Introduce new, 1 level model
  - Plug compatible with applications
- Solaris 9
  - New 1 level model becomes the default
- Solaris 10
  - Unified process model

# Solaris 10

There's so much more!

- Project "Atlas"
  - Small systems performance tuning
  - x86 focus, but benefits for all!
- Project "Janus"
  - Run Linux binaries on Solaris
    - Linux binary execution environment
  - X86 only

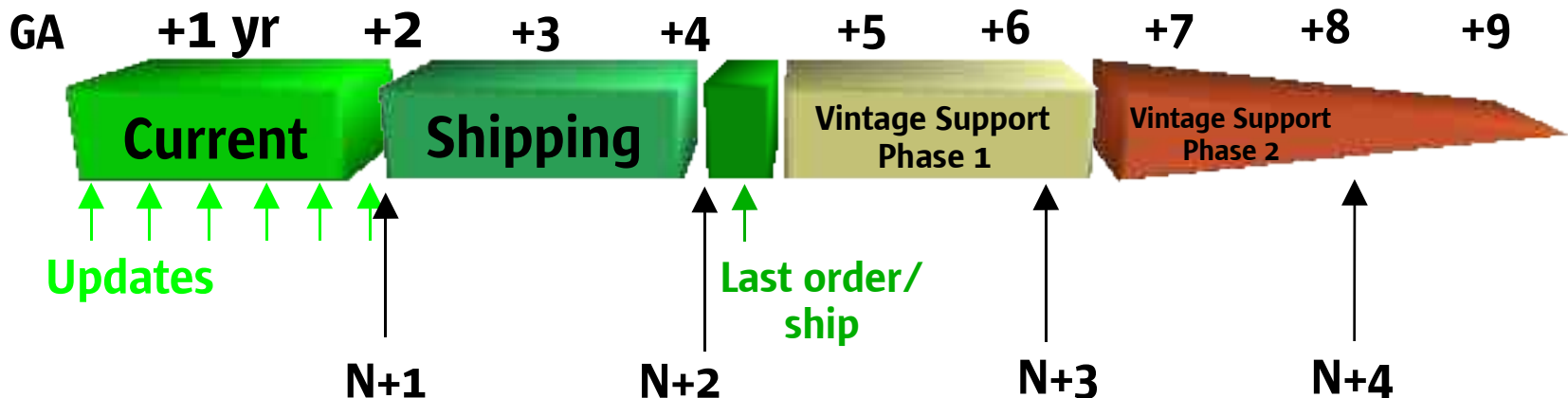
# Throughput Computing

- Processor Technology
  - CMP – Chip Multiprocessing
    - Multiple execution cores on a single “chip”
      - Multiple threads
      - UltraSPARC IV, Niagara
  - SMT – Simultaneous Multithreading
    - Multiple threads executing instructions on shared silicon
  - VT – Vertically Threaded
    - Multiple threads multiplexed on a single pipeline

# Solaris Release Roadmap

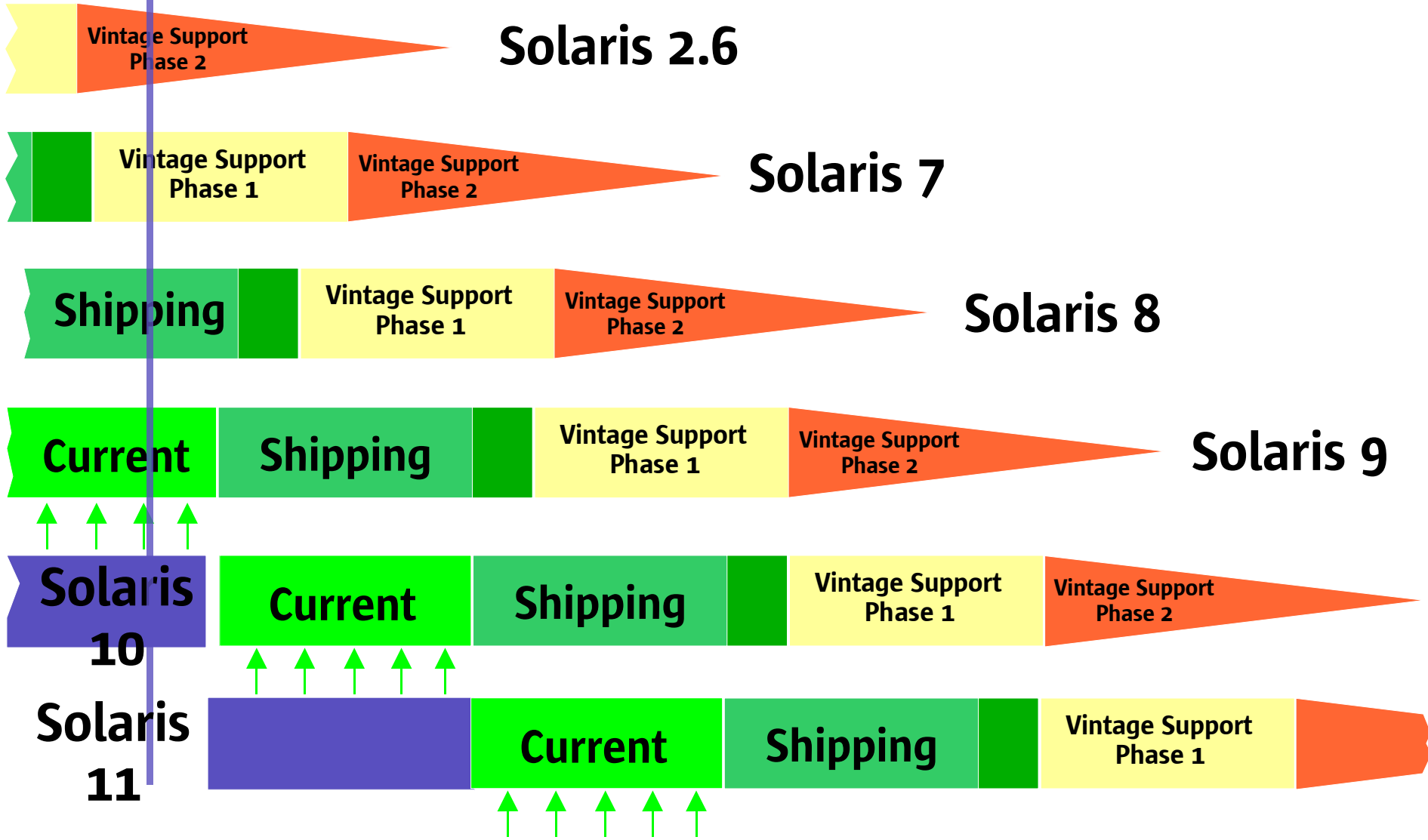
- Named release planned every 2-3 years
  - Application compatibility guaranteed
- Approximately four updates per year
- Shipping life: 4-6 years (or more)
- Support life: 10-11 years (or more)

Solaris N



# Solaris Release Roadmap

*April 2004*



# Solaris 10 Milestones

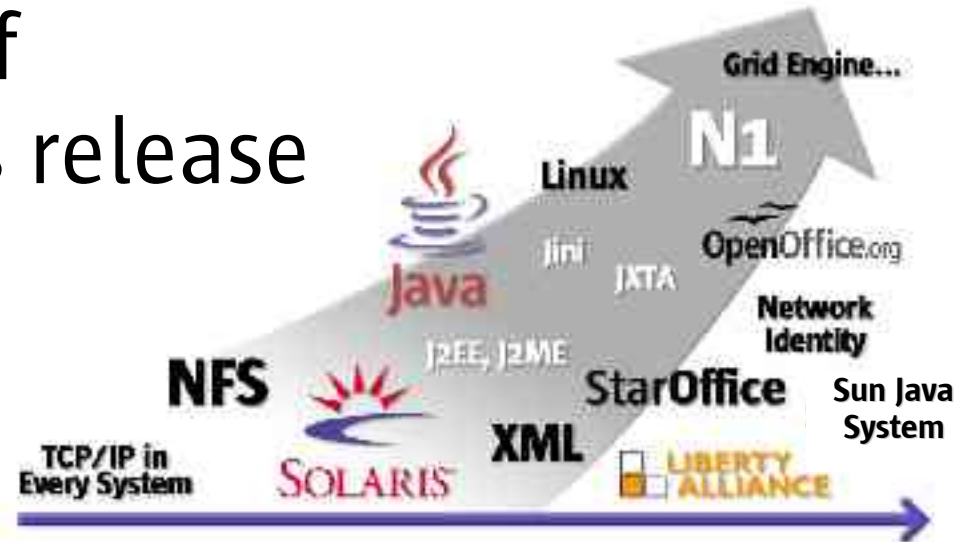
(Future dates subject to change)

- Build 1: January 2002
- Production servers within Sun:  
April 2002
- Software Express for Solaris™:  
September 2003 (ongoing)
- Beta program: March 2004
- Release: January, 2005

# Software Express

Timely Access to Leading-Edge Technology

- Monthly builds of upcoming Solaris release
  - Early access to new features
  - Free access for downloads
  - \$99/year subscription for support



*Delivering disruptive innovation to Sun's customers*

# Software Express for Solaris: 3/04

## Optimal Utilization

*N1 Grid Containers*

## Relentless Availability

*Predictive Self Hea  
Next-gen Filesyst*

## Extreme Performance

*Dynamic Tracing  
Network  
Entry Systems  
NFSv4*

## Unparalleled Security

*Process Rights Management  
Crypto Infrastructure  
IP Filter*

## Platform Choice

*New UltraSPARC IV,  
New AMD Opteron  
Linux Compatibility*





## Solaris 10

[james.mauro@sun.com](mailto:james.mauro@sun.com)

