Solaris 10

Randy Fishel
Sun Microsystems, Inc.
What's UP?

- DTrace – Dynamic Tracing
- N1 Grid Containers (a.k.a Zones)
- Greenline/Service Management Facility
- Fault Management Architecture
- AMD64
- Solaris 10 Networking
- Network Performance Futures
- Java Desktop System - JDS
- Zetabyte File System – ZFS
- Project Janus
- Open Source
- Useful Information
Solaris 10 Design Principles

- Performance
- Security
- Reliability
- Availability
- Manageability
- Serviceability
- Platform Neutrality
Dynamic Tracing – Performance Bottleneck Buster

- Breakthrough approach for tuning
  - Power tool for real-time analysis, diagnosis
- Safe and comprehensive
  - Non invasive, little overhead, easy to use
  - One view into both system and application level
  - Over 30,000 data monitoring points
- Designed for live use on production systems
  - No need to force failure, then do postmortem debug
  - No need to re-create the problem on test systems
  - No need to run different, slow, instrumented OS in production
- Reduced costs
  - Solutions found in minutes or hours, not days or weeks
  - Optimized apps: cases of 3-30x speedups already seen
Why Dynamic Tracing?

- Well-defined techniques for debugging *fatal, non-reproducible* failure:
  - Obtain core file or crash dump
  - Debug problem *postmortem* using mdb(1), dbx(1)

- Techniques for debugging *transient* failures are much more ad hoc
  - Typical techniques push traditional tools (e.g. truss(1), mdb(1)) beyond their design centers
  - Many transient problems cannot be debugged at all using extant techniques
Exploring DTrace

- DTrace is available to the public starting in Solaris Express 11/03:
  http://www.sun.com/software/solaris/solaris-express

- BigAdmin has a page and discussion forum dedicated to DTrace:
  http://www.sun.com/bigadmin/content/dtrace

- The DTrace AnswerBook is available for public download there as well
N1 Grid Containers (Zones)

- Basic concept: isolated execution environment within a Solaris instance
- Includes resource, security, failure isolation
- Lightweight, flexible, efficient
- One OS to manage
- Components:
  - Resource management (CPU, memory, ...)
  - Security/namespace isolation (zones)
N1 Grid Containers (Zones)

- Provides virtualized OS services that look like different Solaris instances
- Isolates applications from each other
- Hides the details of the underlying OS
- Provides almost arbitrary granularity in isolation and/or sharing resources
- Application environment is compatible for existing programs.
When to deploy Zones

• **Hostile and untrustworthy applications**
  - Example: Two webservers each binding to port 80
  - Untrusted software that should be isolated

• **Data center consolidation**
  - Multiple databases with different administrators

• **Hosting**
  - Consolidate many small customers onto a server giving some or all of them the root password

• **Software development**
  - A cheap way to simulate a set of production systems, test software installation, etc.
Grid Containers Block Diagram

**global zone** (serviceprovider.com)

- **blue zone** (blueslugs.com)
  - zone root: /aux0/blueslugs
  - web services (Apache 1.3.22, J2SE)
  - enterprise services (Oracle 8i, IAS 6)
  - core services (ypbind, automountd)

- **foo zone** (foo.net)
  - zone root: /aux0/foonet
  - login services (OpenSSH ssdh 3.4)
  - network services (BIND 8.3, sendmail)
  - core services (ypbind, inetd, rpcbind)

- **beck zone** (beck.org)
  - zone root: /aux0/beck
  - web services (Apache 2.0)
  - network services (BIND 9.2, sendmail)
  - core services (inetd, ldap_cachemgr)

**zone management** (zonecfg(1M), zoneadm(1M), zlogin(1), ...)

- core services (inetd, rpcbind, ypbind, automountd, snmpd, dtlogin, sendmail, sshd, ...)
- remote admin/monitoring (SNMP, SunMC, WBEM)
- platform administration (syseventd, devfsadm, ...)

**network device** (ce0)

**network device** (ge0)

**storage complex**
Greenline

• Problem:
  – Ad hoc mechanisms for managing services:
    • /etc files
    • Rc scripts

• Solution:
  – Framework for service management
    • Repository for configuration data
    • Administrative enable/disable controls
    • Fine-grained access control
  – Link between applications and FMA
    Automated single-node restart
What’s a Greenline service?

- A persistently-running application
- A named instance of the service entity schema:
  - Start, stop, restart, health/status service methods
  - Properties (bundles)
  - Restart relationship(s)
- Example: Internet restarter service
  - `init.d` code → method
  - `inetd.conf` → properties
  - `rc.d` order → milestone dependency
Start-up and configuration

Today

- /dev/*
  - network interfaces
- /etc/inittab
- /etc/init.d/*
- /etc/rc?.d/*
- /etc/inet/inetd.conf

Tomorrow

- low-level devices
- invocation, termination
- properties
- service properties
- file system permissions
- security
- repository
- DSS datastores
- NSS backends
- delegated roles
- local files
- security profiles
- entity authorizations
- local cache
Architecture schematic

management agent
observability agent
inet-service
repository API
svc.configd(1M)
svc.startd(1M)
init(1M)
KERNEL
service
process contract
repository client
Predictive Self Healing — Online Recovery with Automated Services

- Breakthrough approach to service availability
  - Error detection & aggregation, auto recovery
- Reduced downtime
  - Components proactively offlined before failure
  - Automatic service restart
  - Diagnosis & mitigation in milliseconds, not hours
- Reduced complexity
  - Simplified error reporting
  - All system & service interdependencies recorded and correlated
- Reduced costs
  - Reduced system downtime, increased utilization
  - Higher server-to-administrator ratio
Predictive Self Healing – Phase 1

• Solaris FMA infrastructure and Fault Manager:
  – New tools for admins, ops, and service
  – New structured log files for telemetry data
  – Live diagnosis updates without reboots

• Standardized fault messaging agent

• Messages linked to new customer web site

• Automatic service restart (Service Manager)

• Diagnosis for UltraSPARC- III, IV CPU and Memory

• Automatic CPU and memory retire

• Improved software resilience to I/O failures
Solaris/ AMD64

- Native 64bit architecture for AMD64/ Opteron
- 64bit execution on some Intel processors
- Will also run 32bit binaries as Solaris Sparc
- Will run 32bit Linux binaries under Janus
Java Desktop System - JDS

- Integrates desktop and user productivity tools in one component:
  - Gnome 2.6
  - Mozilla 1.7
  - Evolution 1.4.6
  - StarOffice 7 PP3
  - APOC 1.0
  - Xorg xserver (X11r6.7)
Network Performance

• Turn performance into competitive advantage for SMI, partners and customers
  – Add performance to Solaris' reputation for quality and reliability
  – Strong horizontal and vertical scalability
  – Robust out of the box performance
• Improve network performance over 25%
• Optimize TCP/IP with respect to each other
• Optimized the IP Classifier
Performance Strategy

- Partner with ISVs, IHVs, and customers to understand and address performance issues
- Focus on improving whole stack and application performance
  - Utilize micro/macro benchmarking and customer workloads
- Exploit emerging HW technology while being platform agnostic
  - Multithreaded (e.g. CMP, SMT, CMT), SSE2, 64-bit AMD, etc.
  - 10 GbE, TOE, iSCSI, RDMA, Crypto Offload, etc.
- Leverage synergy between organizations within Sun to drive integrated performance
Solaris 10: Key Network Technologies

- **FireEngine**: Overall TCP/IP performance enhancement
- **MultiData**: Bulk data throughput
- **Better ZeroCopy support**: (ftp server faster by 30%)
- **SCTP**: Support for the protocol itself and the sockets API
- **Fully deployable IPv6**
- **Wanboot**: Remote boot machine using HTTP/HTTPS
  - Does not require DHCP
  - Works across firewall
- **Leadville**: Fibre channel on x86
FE - IP Classifier

• Use a connection classifier early in IP for incoming packet

• The connection structure ('connp') contains all the necessary information:
  - The CPU/queue on which the packet needs to be processed
  - The string of functions necessary to process the packet
Network Performance Futures

- **Nemo:**
  - Dynamic switching between interrupt and polling
  - 10Gbps NIC support
  - Vlan and Trunking support for the masses
  - Far easier to create drivers
- NCA merge to FireEngine (NL7C)
- UDP performance (yosemite)
- TOE and other offload support
- Asynchronous socket support
- SIP - proxy, redirect and registrar servers
- Diameter and RTP ....
Nemo: GLD v3

- Next version of GLD (for Sparc and x86)
- Trunking, vlan, dynamic polling, chaining support
- High performance framework
- Large segment offload support
- Will make writing device driver a breeze
Dynamic Filesystem (ZFS)

- **Pooled storage**
  - Completely eliminates the antique notion of volumes
  - Does for storage what VM did for memory

- **End-to-end data integrity**
  - Historically considered “too expensive”
  - Turns out, no it isn't
  - And the alternative is unacceptable

- **Everything is transactional**
  - Keeps things always consistent on disk
  - Removes almost all constraints on I/O order
  - Allows us to get huge performance wins
FS/ Volume vs. ZFS

**Traditional Volumes**
- Abstraction: virtual disk
- Partition/volume for each FS
- Grow/shrink by hand
- Each FS has limited bandwidth
- Storage is fragmented, stranded

**ZFS Pooled Storage**
- Abstraction: malloc/free
- No partitions to manage
- Grow/shrink automatically
- All bandwidth always available
- Pool allows space to be shared
Project Janus – Linux Binary Compatibility

• Run Linux applications natively on Solaris x86
• Stand-alone executables require no additional support
• Available in Solaris 10 update 1
Open Source

- More integration of Open Source software into Solaris
- More of Solaris introduced into the Open Source community
- Community developers soon can contribute to Solaris
Useful Information

- http://www.sun.com/bigadmin/xperts/
- http://www.sun.com/bigadmin/content/dtrace
Solaris 10
Soon, at a location near you
Sun Microsystems, Inc.
Questions?

Sun Microsystems, Inc.