OpenSolaris Overview lecture

Rami Rosen

ramirose@gmail.com

February 2007

Table Of Contents

- OpenSolaris : Pros and Cons
- Versions
- Solaris – Linux interoperability
- Open Source model
- Solaris Doors and Solaris Zones
- DTrace and MDB
- Solaris kernel build and install
- Networking and STREAMS
- Solaris Annoyances

Disclaimer

Everything in this lecture shall not, under any circumstances, hold any legal liability whatsoever.

Any usage of the data and information in this document shall be solely on the responsibility of the user.

This lecture is not given on behalf of any company or organization.

I am not trying to persuade anybody to migrate to OpenSolaris; I am not convinced at all regarding a need to do so; however it is worth knowing OpenSolaris features.

What is OpenSolaris ?

- Solaris is a proprietary operating system developed by Sun MicroSystems ; it started as a BSD port in the 90'.
- The first step for releasing it as an open source was

  releasing the code of DTrace in January 2005 under CDDL.
- Solaris itself was released as open source in June 2005. (OS/networking consolidation)
- OpenSolaris is the first proprietary operating system to be released as open source.
- Runs on SPARC (64 bit) and i386/x86_64 processors.

Pros

- Multicore market is growing rapidly; Sun, primarily a hardware company, was the first in the multicore market.
- Intel released quad cores only recently, in November 2006. Later, also AMD did so. Recently, Intel revealed a prototype of a single CPU with 80 cores; it is said to be implemented within 5 years.
- Sun's new T1 UltraSPARC machine ("Niagra") has up to 8 cores on one CPU (32 threads) and up to 64 GB of RAM.
- based on CoolThreads technology (aimed at lower TCO).
  - Next generation (Niagra 2) will have up to 64 threads (still with 8 cores).

Pros - contd

- The improvement in performance is not linear; it is also dependent upon how applications are written (more parallel applications will naturally gain more performance) .
- We should mention that building multiprocessor machines is not a new idea ; there was a research on this subject in MIT by Agrawal et al. ("The MIT Alewife Machine", 1995).
  - Alewife: Up to 512 nodes; DSM and parallel computation.
- The new UltraSPARC T1 **hardware** source code is available: http://www.opensparc.net. (verilog design and specs, etc.)

Pros - contd

- We must mention that Intel multicore technology is different.
- DTrace is an advanced debugging mechanism, both for kernel and user space applications. Linux debugging tools like systemtap and kprobes are less sophisticated.
- Cost: Solaris support is usually cheaper than the other commercial distros (like RedHat enterprise).
- License: CDDL.
  - There was in the past an initiative in SUN to add GPLv3 licensing for OpenSolaris, but currently GPLv3 License for OpenSolaris is not adopted by SUN.

Pros - contd

- CDDL stands for “Common Development and Distribution License”.
  - It is based on the Mozilla Public License, version 1.1 ("MPL").
- Almost all cutting edge, new features of Linux are there: iSCSI, LVM (Logical Volume Manager), Xen, RAID, USB webcam support, SATA drivers, InfiniBand, and a lot more. Also for networking – polling (integrated in the kernel by default), bonding, wireless drivers, sctp protocol, IP Filters (correspond to IP tables), IPSec, IPMP, IPv6, and more.
- You can choose between Gnome/CDE desktops managers.

Pros - contd

- Solaris has features which are not (yet?) in Linux. For example, DTrace, ZFS, SIP protocol implementation, SMF,and more.
- SMF stands for: Service Management Facility.
  - comfortable when tracing problems; manages services dependency.
  - Usage example – restarting ssh : svcadm restart svc:/network/ssh
- Installing packages is quite easy; the yum/YaST parallel in Solaris is called “pkg-get”.
  - Simple to install : "pkg-get install pkgName".

Pros - contd

- A lot of common applications are available.
- UFS has logging since Solaris 7; starting with Solaris 9, UFS logging is default.
- ZFS advanced file system – 128 bit ; block level checksum.
- You don't need to run fsck on ZFS.
- Sun was the first to deliver NFS and VFS, which are widely used.
- Will ZFS also gain success outside Solaris ?
- Caveat:currently zfs cannot be a root filesystem. (under work)

Cons

- On the whole, currently less hardware devices are supported in Solaris than in other OSs.
- Hardware support: Most common hardware is supported. However, most vendors, which develop Solaris drivers for their devices, don't release the source code.

Cons - contd

- On the other hand, they do usually release the Linux drivers for the same devices. This can sometimes be a restriction.
- Sometimes device drivers are left in alpha version for quite a time. For example, the ZyDas USB wireless nic driver in sourceforge.net.
- see: http://zd1211sol.sourceforge.net.

Cons - contd

- Porting drivers from Linux to Solaris is not straight forward.
- The learning curve for DDI API, which is needed for porting drivers, is time consuming. Probably nic drivers are more difficult because they are STREAMS character drivers, and should use GLD and DLPI (Still , some of the API is like Linux, for example , IOCTLS).
- A good starting point for this task can be Murayama free nic drivers page.
- Some advanced Linux features are missing: just for example, kexec , RCU and more (though there is a readers/writers lock implementation , written in “C”).

Cons - contd

- Currently Solaris is for i386 and x86_64 and SPARC processors; there is a project for Solaris on PPC (named “Polaris”). BSD supports 40 platforms, and Linux more than that.
- For the embedded market- no Solaris for processors without MMU. (like ARM-7).
- System Administration: there is some resemblance between Linux and Solaris; but some tasks are handled differently (for example, services). Learning Solaris System Administration when migrating from Linux can be time consuming, depends on what you are up to.

Solaris Versions

Solaris Express.

- The newest version of Solaris; you can pay less than 100$ for a basic yearly support.
- OpenSolaris:
- You can download the binaries or code from SUN repositories.
- Basically corresponds to Solaris Express, but has more code which did not yet pass full QA.

Solaris Versions - Contd

- If you want to build the kernel from source, you must do it under a Solaris Express machine.
- Solaris 10 (The stable version)
- Solaris Express (particular builds of

  the OS codenamed Nevada).
- Every several months there is update

  to Solaris 10; the last one is update3 (s10u3)

Solaris - Linux Interoperability

File Systems Interoperability

- Default File System in Solaris is UFS. (“Unix File System”). You cannot choose other type of filesystem while installing (whereas in Linux you can).
- The more advanced filesystem of Solaris is the ZFS.
- The name origin is "Zettabyte File System".
- ZettaByte is 2^70 bytes.
- ZFS is a 128-bit file system.

File Systems Interoperability - contd

- There is some attempt to write ZFS userspace file system in Linux.
- You can mount Solaris UFS partitions as read-only from Linux x86_64/i386.
- You can also build the UFS module on Linux with support for read-write, but this is considered risky.

File Systems Interoperability - contd

- You cannot mount ZFS Solaris partitions from Linux.
- You can mount ext3 partitions from Solaris after installing two packages.
- You cannot mount Linux reiserfs partitions.
- There is VFAT filesystem (“PCFS”) in solaris.

Execution environment Interoperability

- You can build and run Linux applications after installing a special zone called lx brandZ; more on zones and brandZ later.
  - Restriction: This Linux zone can be created only in x86 platforms, not in SPARC.
  - Restriction: There are some limitations on what you can run; you cannot load linux kernel modules, for example, in lx_zone.

Solaris - Linux Interoperability (Misc).

- Solaris iscsi initiator can detect a Linux iscsi target, and also vice versa.
- You can install a dual-boot Solaris - Linux machine. (Grub was patched in Solaris to support its boot process).
- There are already Linux distros on Niagra
- The first one was Ubuntu with help from Dave Miller. (After he tried to boot Fedora Core on Niagra and it crashed).

Solaris-Linux InterOperability Misc-Contd

- In the Linux kernel there is a sparc64 architecture subtree maintained by Dave Miller.

Solaris Open source Model

- There is no central mailing list like Linux Kernel Mailing List (LKML) which is open and everybody can send patches.
- You can provide patches, but right now you have to work with a sponsor from Sun to get them back into the code.
- In the future we are expected to see more and more contributors from outside SUN.

Solaris Open source Model - Contd

- Everybody can send a BUG to a BUG database site: http://bugs.opensolaris.org.
- Everybody can send RFE (Request for Feature Enhancement). Some are implemented.
- Code reviews and PSARC docs.
- PSARC stands for “Platform Software Architecture Review Committee”.

Solaris Open source Model - Contd

- OpenSolaris “flag days” site announces new features which future builds will have, new PSARC docs, etc.

All that hype:

- There are many mailing list.
- There are several distributions for Solaris.
- A live cd (belenix)

Solaris Open source Model - Contd.

- There is also a (1GB) USB disk on key version for this belenix.
- There are user clubs around the globe.
  - Many SUN kernel developers give lectures there.
- Many developer maintain blogs under SUN OpenSolaris site, including SUN new CEO (Jonathan Schwartz).
- A central blog site : http://planetsolaris.org.
- OpenSolaris days around the globe.

Solaris Open source Model - Contd.

- The German Unix User Group

  has organized the first OpenSolaris Developer

  Conference (27.2.07).
- OpenSolaris Weekly News web site.

Solaris Doors

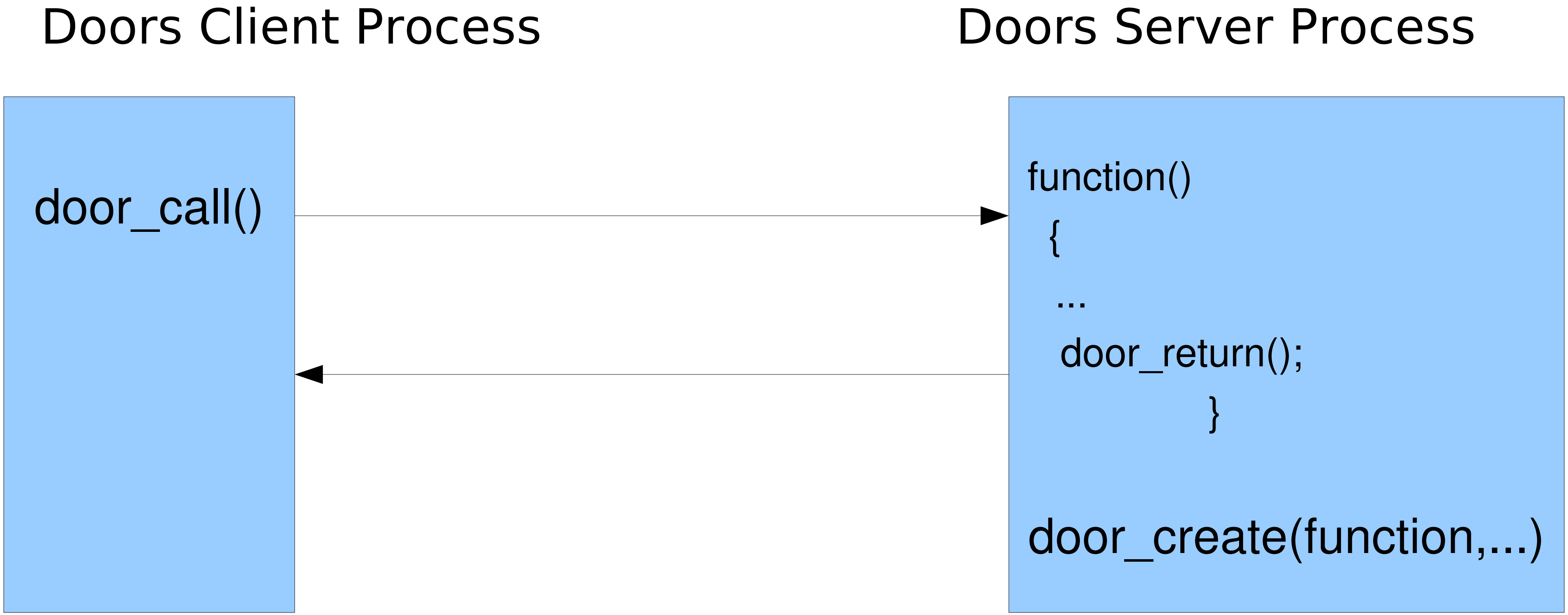
- A lightweight and fast IPC mechanism.
- Used in many places: zones management, iscsi-target admin ,etc.
- Resembles Unix local sockets.
- Communication via a local file system file descriptor.
- There is a project for Linux doors in sf.net: http://ldoor.sourceforge.net

Solaris Doors - contd

- Within the kernel, doors are implemented as a pseudo file system, which is loaded during boot.
- You can invoke a procedure in a different process by calling door_call().
- You can pass argument to this call.
- User space applications use doors and also the kernel (upcalls).

Solaris Doors - contd

- A Door Client and a Door Server Interaction:



Doors Client Process

Doors Server Process

```
door_call()
```

```
function()
 {
  ...
   door_return();
                }

door_create(function,...)
```

Solaris Doors - contd

- After you call door_create() in the server, you assign it to a path in the filesystem for door by calling fattach(door_id,path).
- What if this path is already held by other door server?
- For this you call fdetach() before

Solaris Doors - contd

- A process can call multiple times to door_create().
- Each such call adds a door_node in the process descriptor (proc_t) door list.
- You can pass an argument to a door_call().
- You can also get a result back in the argument which you pass to door_call().

Creating a door server - example:

Here is a somewhat simplified example:

```
int newfd;
int did = door_create(server,0,0);
newfd = creat("/temp/door",0444);
(void) fdetach("/temp/door");
fattach(did,"/temp/door");
pause();
```

Creating a door server - contd.

```
...
void server(void *cookie, char *argp, size_t
   arg_size, door_desc_t *dp, uint_t n_desc)
{ ...
...
}
```

Creating a door client - example:

```
typedef struct myArg {

int id;

} myArg_t;

...

door_arg_t darg;

int doorfd = open("/temp/door",O_RDONLY);

myArg_t* arg = (myArg_t*)malloc(sizeof(myArg_t));
```

Creating a door client - example:

```
if (!arg)
  exit(0);
arg->id = 12;
darg.data_ptr  = (char*)arg;
darg.data_size = sizeof(*arg);
darg.desc_ptr = NULL;
darg.desc_num = 0;
door_call(doorfd,&darg);
```

Zones

- Zones is a virtualization solution.
- zonecfg and zoneadm are the user space tools for zones configuration, creation and management.
- These user space tools communicate with the kernel via doors IPC mechanism.
- Each zone has a numeric ID and a unique name.

Zones - Contd.

- The global zone has ID 0 and is always running (cannot be halted).
- Booting a zone is in fact forking a process (called “newproc” method)
- A kernel thread named zsched with id 0 is created in that zone and it starts the init process in that zone (id 1).
- A zone runs with subset of privileges(5).

Zones - Contd.

- Some operations are not allowed in a zone.
- For example : mknod from inside a zone will return “mknod: Not owner”.
- Also raw sockets are prohibited.
  - socket(AF_INET,SOCK_RAW,IPPROTO_UDP);

    will generate EPROTONOSUPPORT error. (“Protocol not supported” error.)
  - What about : socket(AF_INET,SOCK_RAW,IPPROTO_ICMP); ?
- Currently IPFilter cannot be used to filter traffic passing between two zones on the same system. (There is RFE...)

Zones - Contd.

- Under /proc of this file system, you will see only processes that this zone created.
- Each Zone has a subset of the global zone services.
- Each Zone has svc.configd and svc.startd daemons of its own for service management.

Example: zone creation (my-zone)

```
zonecfg -z my-zone -f myzone.txt
```

```
create -b
set zonepath=/export/home/myzone
set autoboot=false
set pool=pool_default
set limitpriv=default,sys_time
add inherit-pkg-dir
set dir=/lib
end
add inherit-pkg-dir
set dir=/platform
```

Example: zone creation (my-zone) - Contd.

```
end

add inherit-pkg-dir

set dir=/usr

end

add fs

set dir=/usr/local

set special=/opt/local

set type=lofs

end

add net
```

Example: zone creation (my-zone)

```
set physical=e100g0

set address=192.168.0.30

end
```

Zone States

- Zone states: There is a finite state machine of Zone states: a Zone can be in one of six states:
- CONFIGURED, INSTALLED , READY , RUNNING, SHUTTING_DOWN and DOWN.
- A Zone which is INSTALLED does not have yet a “virtual platform”.

Zone States

- A zone is in a “INSTALLED” state after setting a

  virtual platform. A process named zsched was created (id 0), but no other processes are running.

- A zone is in a “RUNNING” state after init and other processes started.

- A zone is in “SHUTTING_DOWN” or “DOWN” while the zone is being halted.

Zone States - Contd.

You can view all zones and their states by: zoneadm list -vc:

```
ID NAME   STATUS    PATH
0 global   running    /
- my-zone  Installed   /export/home/my-zone
```

- Installed zones do not have an id yet.

Zone States - Contd.

- After you created a zone, you install it by

zoneadm -z my-zone install

- boot the zone: zoneadm -z myzone boot
- Halt a zone: zoneadm -z myzone halt
- Login into a zone: zlogin myzone.

Zone States - Contd.

- Each zone has a zoneadmd daemon process of its own; state transitions are performed by calls from zoneadm to this daemon.
- All zoneadmd daemons run in the global zone.

Zones process model

- Processes within one zone (other than the global zone) must **not** be able to affect activity of other processes.
- Processes which run on a different zone (other than the global zone) should **not** be able to even see other processes which run on a different zone.

Zones process model - Contd

- So running “pkill processName” from the global zone will kill **all** processes with that name in **all** zones.
- On the other hand, each zone has a /proc

of it's own an cannot see under /proc other

pids of processes in other zones.

Zones - misc

- With IP instances, each zone has an instance of the TCP/IP stack.
- IP instances are enabled with **set ip-type=exclusive** when creating a zone with zonecfg.
- IP instances is part of CrossBow project.
- Solaris Containers are zones coupled with resource management.

Branded Zones

- Branded Zones enable us to create non-global zones which contain non native operating environments.
- The lx brand provides a Linux environment

  under Solaris.
  - We create an lx brand also by zonecfg.
  - We use set brand=lx when configuring with zonecfg

Branded Zones - Contd

- The lx brand enables us to install CentOS 3.x

or RedHat Enterprise 3.x inside a zone.

- The lx zone supports only user level applications. You cannot use device

  drivers or kernel modules in an lx zone.

- Interpostion points for system calls.

Networking and STREAMS.

- FireEngine – the new network architecture in OpenSolaris 10.
- FireEngine introudced “squeue” , a serialized queue.
- STREAMS defines standard interfaces for character I/O.
- Data on a stream is passed as messages.
- The Message Block is the basic message

  which is used in STREAMS.

The Message Block

Common types for b_datap can be DATA or PROTO or other.

```
struct msgb      *b_next;    /* next message on queue */
struct msgb      *b_prev;    /* previous message on queue */
struct msgb      *b_cont;    /* next message block */
unsigned char   *b_rptr;    /* 1st unread data byte of buffer */
unsigned char   *b_wptr;    /* 1st unwritten data byte of
    buffer */
struct datab    *b_datap;   /* pointer to data block */
unsigned char   b_band;     /* message priority  */
unsigned short  b_flag;     /* used by stream head  */
```

IP filters

- Started in about 1993.
- Runs on many flavours of Unix.
- Until b52 , IP filter used a STREAM module called “pfil”.
- This was replaced from b52 by function calls to improve performance.
- A layer 2 (MAC layer) filtering is planned.

• CrossBow

- CrossBow – virtualization networking project.
- You can control QoS parameters of

  virtual nics.
- Example for setting vnic bandwidth : dladm create-vnic -d bge0 -m 0:1:2:3:4:5 -b 10000
- Dladm = administer data links utility.

DTrace

- Advanced debugging and tracing tools.
- DTrace stands for Dynamic Tracing.
- Scripts should be written in a language called “D”. Many examples in /usr/demo/dtrace.
- DTrace User Guide: the ultimate manual for DTrace. (more than 400 pages).
- fbt is a Function Boundary Testing provider; shows entry and return of almost every function in every kernel module.

DTrace - contd.

fbt Example:

```
#!/usr/sbin/dtrace -Fs
#pragma D option flowindent
fbt:crash:main_func:entry
{
self->traceme = 1;
}
fbt:::
/self->traceme/
```

DTrace - contd.

```
{}
syscall::ioctl:return
/self->traceme/
{
self->traceme = 0;
exit(0);
}
```

DTrace - contd.

- ./crash.d output:

```
CPU FUNCTION
 0  -> main_func
 0  <- main_func
 0  -> cv_broadcast
 0  <- cv_broadcast
0  -> ire_cache_lookup
 0  <- ire_cache_lookup
 0  -> ip_tcp_input
 0    -> ip_cksum
```

MDB

- DTrace is not the only debugging tool; there is also MDB, the modular debugger.
- In case of panic, 2 files are created under

/var/crash/host: unix.pid and vmcore.pid.

- mdb -k unix.0 vmcore.0 enables getting information, by running commands like ::status and panic_thread::findstack -v.

MDB - contd

Example: derefencing null pointer in a module caused panic.

```
mdb -k unix.18 vmcore.18

Loading modules: [ unix krtld genunix specfs dtrace uppc
   pcplusmp scsi_vhci ufs ip hook neti sctp arp usba uhci nca
   lofs md cpc fcip fctl random crypto fcp zfs logindmux ptm
   sppp nfs ]

> ::status

debugging crash dump vmcore.18 (64-bit) from sbc23

operating system: 5.11 snv_54 (i86pc)
```

MDB - contd

panic message:

```
BAD TRAP: type=e (#pf Page fault) rp=fffffe80000b96c0 addr=0
    occurred in module
```

"crash" due to a NULL pointer dereference

dump content: kernel pages only

MDB - contd

```
> *panic_thread::findstack -v
stack pointer for thread fffffe80000b9c80: fffffe80000b9340
  fffffe80000b93b0 mutex_vector_enter+0x4be(ffffffff801c2da8)
  fffffe80000b9420 0xfffffe80000b93f8()
  fffffe80000b9510 panic+0x9c()
  fffffe80000b95a0 die+0xc8(e, fffffe80000b96c0, 0, 0)
  fffffe80000b96b0 trap+0x12ec(fffffe80000b96c0, 0, 0)
    ...
```

Xen

- There is a port of Xen to OpenSolaris.
- Last version was released in august 2006.
- Currently supports x86 and x86_64.
- SPARC port requires a substantial amount of work.
- Does not currently support HVM.

Solaris Projects

- http://www.opensolaris.org/os/projects
- QEMU.
- Solaris iSCSI Target (Should be integrated in S10U4).
- Virtual Console.
- Zone Manager.
- Crossbow: Network Virtualization and Resource Control.
- Fuse on Solaris.
- Object Storage Device (OSD) support for Solaris.

Solaris Projects-contd.

- ... and more (currently there are 73 projects)

Solaris kernel build and install

- There is a release of OpenSolaris build about every 2 weeks.
- There are also closed bins + build tools for each build.
- You must build OpenSolaris from source under Solaris express community release (SXCR).

Solaris kernel build and install-contd

- OpenSolaris is available in 4 forms: tar.bz2,SVN,bittorent and mercurial.
- There is a web interface with search capabilities, based on java OpenGrok engine: http://src.opensolaris.org/source
- The build process is quite simple as we will see.
- The build process is done by running a korn script named "nightly" thus:
- nightly opensolaris.sh & (also for SPARC).

Solaris kernel build and install-contd

- And then install by:

  Install -G kernelHook -k sun4v/i86pc.

- The build process takes about 3-4 hours on x86_64 machine with 3GB CPU and with 1GB of RAM.
- Also on Niagra with 8 cores and 16 GB of RAM, the build takes about 4 hours.
- The log file (named "nightly.log") can be 60MB - 70MB in size.

Solaris kernel build and install-contd

- The build creates 32 and 64 bit objects (no way to select only one architecture).
- As opposed to Linux, there is no way to configure (there is no parallel to Linux "make menuconfig").
- There is a "flag days" site, where there is info about future versions.
- You can choose between gcc and Sun Studio for compiling the kernel.

Solaris Annoyances

- This list is based on personal perspective.
- No virtual console upon installation (there is however a virtual console project).
- You cannot select which packages to install in the installation process.
- No Midnight Commander upon install.
- Many services under /etc/init.d/ do not have "status" or "restart" option.

Solaris Annoyances-contd

- Unloading modules must be done in two stages: first modinfo to get the module number, than “modunload -i numberOfModule.”
- You can write a script... or simply type each time: “modunload -i `modinfo | grep module | cut -c1-3`” :-)
- There is no one image file for a dvd ; you should download dvd in 5 zipped pieces , unzip, md5sum, and concatenate.
- Some of the user space tools are written using Lex/Yecc. (Guess why this can annoy anybody?).

Solaris Annoyances-contd

- Many flags to common commands (du, fdisk and more) are different in Solaris than Linux.
- Naming of disks is different from Linux ; for example: /dev/rdsk/c0t1d0p0.
- Rebooting of a NIAGRA machine takes almost ten minutes...
- /proc is a bit different ;example: no

  /proc/pid/cmdarg ;use pargs -pid

Summary

- OpenSolaris has some interesting features, like ZFS, DTrace, zones, doors IPC, and more.
- Due to the recent changes in SUN in the last two years regarding open source policy, we might see that Linux will adopt some of Solaris features.
- When thinking about Solaris we should verify hw compatibility as well as consider the time curve for learning sysadmin tasks and different kernel implementation and API.

Books

- **Solaris Internals**. **(Second Edition)** http://www.solarisinternals.com/si/index.php

By Richard McDougall, Jim Mauro

1072 pages.

- **Solaris(TM) Performance and Tools:** DTrace and MDB Techniques for Solaris 10 and OpenSolaris (Solaris Series)

By Richard McDougall, Jim Mauro

- **Solaris Systems Programming** , Rich Teer.

http://www.rite-group.com/rich/ssp/

Links

- Documentation:

  The ultimate source: http://docs.sun.com/app/docs

- OpenSolaris Portal:
- http://www.opensolaris.org/os/
- Online Source Code: (Web Interface)

  http://src.opensolaris.org/source/

- Murayama free nic drivers: http://homepage2.nifty.com/mrym3/taiyodo/eng)

Links - contd

- http://planetsolaris.org/
- Tips for tuning solaris:

  http://arthur.van-dam.net/twiki/bin/view/Arthur/TuningSolaris
- Linux doors:

  http://sourceforge.net/projects/ldoor/
- ZFS on FUSE

http://developer.berlios.de/projects/zfs-fuse/

Questions?

Thank You !