Building an OpenSolaris Build Server

James MacFarlane
Staff Engineer
Solaris Revenue Product Engineering

Steps To A Build System

- A SPARC or x86 / x64 system
  - > Running the Solaris Express release
- The compilers
  - > Studio 11
- ON Tools
- Closed source binaries
- The code
- The environment settings

System OS Install

- Install the system with the latest Solaris Express version
  - http://www.opensolaris.org/os/downloads/
  - Leave space in /opt – 1.3 Gb minimum
  - Leave space for workspaces
    - Unbuilt code is 540 Mb
    - One fully built workspace is 4.8 Gb
  - Good to have a slice for Live Upgrade

`

The Compilers

- Studio 11 is a free download for OpenSolaris
  - > http://www.opensolaris.org/os/community/tools/sun_studio_tools/sun_studio_11_tools/
- Needs to be installed under /opt/SUNWspro/
- GNU gcc can also be used
  - > gcc included under /opt/sfw
  - > Latest info on building with gcc is here :
  - > http://opensolaris.org/os/community/tools/gcc/

`

Installing the ON Tools

- A set of tools used to build and install the resulting binaries
- Delivered as a Solaris package - SUNWonbld
- Installs under /opt/onbld
- Need to watch the “flag day” notices for updates :
  - > http://opensolaris.org/os/community/on/flag-days/
- Download from :
  - > http://dlc.sun.com/osol/on/downloads/current/

Closed Source Binaries

- Pre-compiled binaries for some components
- Required to make a complete install image
- Delivered as a tar file of binaries
  - DEBUG and RELEASE builds available
- Download from :
  - http://dlc.sun.com/osol/on/downloads/current/

`

Mercurial

- The source code manager for OpenSolaris
- Replaces the old Teamware system
- Included with snv_45 and later builds
- Downloads for earlier releases available
  - > http://opensolaris.org/os/community/tools/scm/

`

Mercurial Configuration

- Requires a working ssh connection to opensolaris.org
- Enable ssh compression
- Example ~/.ssh/config file :

```
Compression yes
# A SOCKS proxy may be needed to get through a firewall
host *.opensolaris.org
	ProxyCommand /usr/lib/ssh/ssh-socks5-proxy-connect -h {Socks Proxy} %h
%p
```

`

Getting The Code

- Use hg(1) to get the initial copy of the source
- Format :
  - > “hg clone {source} {destination}”
- Example :

```
hg clone ssh://anon@hg.opensolaris.org/hg/onnv/onnv-gate
requesting all changes
adding changesets
adding manifests
adding file changes
added 3685 changesets with 69850 changes to 43667 files
39929 files updated, 0 files merged, 0 files removed, 0 files unresolved
```

Cloning A Source Copy

- Use hg(1) to make a working copy
- Format :
  > “hg clone {source} {destination}”
- Example :

```
hg clone /code/onnv-gate /code/my_project
requesting all changes
adding changesets
adding manifests
adding file changes
added 3685 changesets with 69850 changes to 43667 files
39929 files updated, 0 files merged, 0 files removed, 0 files unresolved
```

Configuring The Environment

- Need to have correct $PATH defined :
  - Include /opt/SUNWspro/bin
  - Include /opt/onbld/bin
  - /usr/ccs/bin – only below snv_68

`

Configuring The Environment

- Customise the opensolaris.sh file
  - usr/src/tools/env/opensolaris.sh
- Need to set the following :
  - GATE – This workspace
  - CODEMGR_WS – Where this workspace is
  - ON_CLOSED_BINS – Where the closed source binaries are
- Example settings in the reference slides

`

Do A Build

- Best method is to use the nightly script
  - /opt/onbld/bin/nightly {env file}
- Will take a long time ...
  - It's not called nightly for nothing
- Will build the code, make the bfu archives, run lint, build the packages
- Example: (Sat in the top of the workspace)

```
/opt/onbld/bin/nightly usr/src/tools/env/opensolaris.sh
```

Building Individual Files

- Need to run a “nightly” build first to build all the required libraries and install the headers
- Need to set the same environment as “nightly” script requires. Use “bldenv”
  - > /opt/onbld/bin/bldenv usr/src/tools/env/opensolaris.sh
- Just use “dmake all” in the relevant source dir
  - > For the kernel, make everything under usr/src/uts

`

Install The Binaries

- Simple changes can be copied into place
- Complex changes and new builds use “bfu”
  - /opt/onbld/bin/bfu
  - Requires archives from a “nightly” build
  - Share the onbld tools to each target system
- Example : (as root on the target system)

```
bfu /net/{buildsvr}/{workspace}/archives/sparc/nightly
```

Resolving Conflicts

- Sometimes bfu archives conflict with changes on the target system
- Files can be resolved by hand
- Better to use “acr”
  - /opt/onbld/bin/acr

Building an OpenSolaris Build Server

Nightly Script Usage

```
USAGE='Usage: nightly [-in] [-V VERS ] [ -S E|D|H|O ] <env_file>
Where:
    -i      Fast incremental options (no clobber, lint, check)
    -n       Do not do a bringover
    -V VERS set the build version string to VERS
    -S       Build a variant of the source product
            E - build exportable source
            D - build domestic source (exportable + crypt)
            H - build hybrid source (binaries + deleted source)
            O - build (only) open source
    <env_file>  file in Bourne shell syntax that sets and exports
    variables that configure the operation of this script and many of
    the scripts this one calls. If <env_file> does not exist,
    it will be looked for in $OPTHOME/onbld/env.

non-DEBUG is the default build type. Build options can be set in the
NIGHTLY_OPTIONS variable in the <env_file>
```

NIGHTLY_OPTIONS

-A	check for ABI differences in .so files
-C	check for cstyle/hdrchk errors
-D	do a build with DEBUG on
-F	do _not_ do a non-DEBUG build
-G	gate keeper default group of options (-au)
-I	integration engineer default group of options (-ampu)
-M	do not run pmodes (safe file permission checker)
-N	do not run protocmp
-R	default group of options for building a release (-mp)
-U	update proto area in the parent
-V VERS	set the build version string to VERS
-X	copy x86 IHV proto area

NIGHTLY_OPTIONS (cont.)

-a	create cpio archives
-f	find unreferenced files
-i	do an incremental build (no "make clobber")
-l	do "make lint" in $LINTDIRS (default: $SRC y)
-m	send mail to $MAILTO at end of build
-n	do not do a bringover
-o	build using root privileges to set OWNER/GROUP (old style)
-p	create packages
-r	check ELF runtime attributes in the proto area
-t	build and use the tools in $SRC/tools
-u	update proto_list_$MACH and friends in the parent workspace; when used with -f, also build an unrefmaster.out in the parent

NIGHTLY_OPTIONS (cont.)

-w report on differences between previous and current proto areas

-z compress cpio archives with gzip

-W Do not report warnings (freeware gate ONLY)

-S Build a variant of the source product

E - build exportable source

D - build domestic source (exportable + crypt)

H - build hybrid source (binaries + deleted source)

O - build (only) open source

`

Example opensolaris.sh Settings

For a system with the source under /code/ws/onnv-gate and the closed source binaries under /code/binaries.

```
# This is a variable for the rest of the script - GATE doesn't matter to
# nightly itself
GATE=onnv-gate;                    export GATE

# CODEMGR_WS - where is your workspace at (or what should nightly name
it)
CODEMGR_WS="/code/ws/$GATE";                    export CODEMGR_WS

# Location of encumbered binaries.
ON_CLOSED_BINS="/code/binaries/closed";        export ON_CLOSED_BINS
```

Updating A Source Copy

- Use hg(1) to refresh a workspace
- Format :
  - “hg pull {source}”
  - “hg update {source}”

`

Updating A Source Copy

- Example :

```
5 > hg pull
pulling from ssh://anon@hg.opensolaris.org/hg/onnv/onnv-gate
searching for changes
adding changesets
adding manifests
adding file changes
added 112 changesets with 928 changes to 831 files
(run 'hg update' to get a working copy)
6 > hg update
824 files updated, 0 files merged, 177 files removed, 0 files
unresolved
```

Making Changes

- After editing the file check your changes
  - > use "hg diff {filename}"
- Example :

```
hg diff metastat.c
diff -r 48f0fd311ddb usr/src/cmd/lvm/util/metastat.c
--- a/usr/src/cmd/lvm/util/metastat.c   Tue Feb 20 05:32:53 2007 -0800
+++ b/usr/src/cmd/lvm/util/metastat.c   Wed Feb 21 16:12:55 2007
+0000
@@ -85,6 +85,9 @@ static int    sp_match(md_sp_t *part, struc
 static void    sp_free_list(struct sp_base_list *lp);

+/*
+ * This is an important comment !
+ */
```

Making Changes

- Once happy, commit the changes
  - use “hg commit {filename}”
- Example :

```
hg commit metastat.c

Comments are good !
HG: changed usr/src/cmd/lvm/util/metastat.c
```

`

Putting Changes Back

- Push the changes back to the parent gate
  - > use “hg push {parent}”
- Example :

```
hg push /code/ws/jmf/nv_project
pushing to /code/ws/jmf/nv_project
searching for changes
adding changesets
adding manifests
adding file changes
added 1 changesets with 1 changes to 1 files
```