Building Enduro/X On Oracle Solaris Platform

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
1.0	2016-06	Initial draft	MV

Contents

Chapter 1

About manual

This manual describes how to build *Enduro/X* Oracle Solaris platform. Document is based on Solaris 11 on x86 machine. Compiler used to Enduro/X is GCC.

Chapter 2

Overview

This manual includes basic installation of Enduro/X which does not include building of documentation and does not use GPG-ME encryption for bridges.

For Solaris Posix queues actually are virtual files in /tmp directory with *.MQD* prefix. Enduro/X for queue management uses these files.

For production releases emulated message queue is selected (-DDEFINE_FORCEEMQ=ON). This is due to fact that Posix queues build in Solaris are unstable on provides weird behaviour (like getting queue attributes causes program to freeze (wait some lock on q?), etc.).

Chapter 3

Installation process

The installation process will install required pen source packages from *http://www.opencsw.org*. you may install packages with different approach. This is just a sample process for getting build system working on under Solaris. For getting Enduro/X to work basically we need following packages:

1. git
2. cmake
3. flex
4. bison
5. libxml2
6. gcc/g++

3.1 Packages to be installed

The following operations will be done from root user. This will download and install open source packages to local machine:

```
# pkgadd -d http://get.opencsw.org/now
# /opt/csw/bin/pkgutil -U
# /opt/csw/bin/pkgutil -y -i git libxml2_dev flex bison cmake gmake
```

3.2 If installing GCC...

```
# /opt/csw/bin/pkgutil -y -i gcc4core gcc4g++
```

3.3 If installing Sun Studio (on Solaris 11)...

- According to: https://pkg-register.oracle.com/register/product_info/6/
- Register and request access, download the and pkg.oracle.com.key.pem pkg.oracle.com.certificate.pem from Oracle to server.

```
# mkdir -m 0775 -p /var/pkg/ssl
# cp -i download-directory/pkg.oracle.com.key.pem /var/pkg/ssl
# cp -i download-directory/pkg.oracle.com.certificate.pem /var/pkg/ssl

# pkg set-publisher \
-k /var/pkg/ssl/pkg.oracle.com.key.pem \
-c /var/pkg/ssl/pkg.oracle.com.certificate.pem \
-G '*' -g https://pkg.oracle.com/solarisstudio/release solarisstudio

# pkg list -af 'pkg://solarisstudio/developer/solarisstudio-124/*'

# pkg install -nv solarisstudio-124

# pkg install solarisstudio-124
```

3.4 If installing Sun Studio (on Solaris 10)...

Firstly download the "tarfile" version from Oracle, visit http://www.oracle.com/technetwork/server-storage/solarisstudio/downloads/-tarfiles-studio-12-4-3048109.html

Next once the tar file is on your server, extract it and start the install:

```
# bzip2 -d SolarisStudio12.4-solaris-x86-bin.tar.bz2
# tar -xf SolarisStudio12.4-solaris-x86-bin.tar
# cd SolarisStudio12.4-solaris-x86-bin
# ./install_patches.sh
# mv solarisstudio12.4 /opt
```

For Solaris 10 also we need gmake to work as "make", thus

```
# ln -s /opt/csw/bin/gmake /usr/bin/make
```

This will put the compiler in "/opt/solarisstudio12.4" the path later used in this tutorial.

Also for solaris 10 we need "ar" tool which we will use from GNU package:

```
# /opt/csw/bin/pkgutil -y -i gcc4core gcc4g++
```

Chapter 4

Getting the source code

Firstly we need to add "user1" under which we will perform build actions. For test purposes we will parepare new user for which Enduro/X will built (this adds the in the path the /opt/csw/bin. You may modify that of your needs. (add /opt/solarisstudio12.4/bin if Sun studio is installed)

4.1 Adding user on Solaris 11

We add the user "user1" and also set the open file limit to 4096, by default it is 256 which is too low for unit testing.

```
# useradd -m user1
# passwd user1
# projadd -K "process.max-file-descriptor=(basic,10000,deny)" proj.files
# usermod -K "project=proj.files" user1
# su - user1
```

4.2 Adding user on Solaris 10

```
# useradd -d /export/home/user1 -m -s /usr/bin/bash -c "User1 User1" user1
# mkdir /export/home/user1
# chown user1:staff /export/home/user1
```

4.3 Preparing the user environment

```
$ bash
$ cat << EOF >> .profile
export PATH=$PATH:/opt/csw/bin:/opt/solarisstudio12.4/bin
EOF
$ chmod +x .profile
$ source .profile
$ cd /export/home/user1
$ GIT_SSL_NO_VERIFY=true git clone https://github.com/endurox-dev/endurox
$ cd endurox
$ git config http.sslVerify "false"
```

4.4 Enduro/X basic Environment configuration for HOME directory

This code bellow creates *ndrx_home* executable file which loads basic environment, so that you can use sample configuration provided by Enduro/X in *sampleconfig* directory. This also assumes that you are going to install to *$HOME/endurox/dist* folder. The file bellow will override the sample configuration.

```
$ cat << EOF > $HOME/ndrx_home
#!/bin/bash

echo "Loading ndrx_home..."
# Where app domain lives
export NDRX_APPHOME=$HOME/endurox
# Where NDRX runtime lives
export NDRX_HOME=$HOME/endurox/dist/bin
# Debug config too
export NDRX_DEBUG_CONF=$HOME/endurox/sampleconfig/debug.conf
# NDRX config too.
export NDRX_CONFIG=$HOME/endurox/sampleconfig/ndrxconfig.xml

export PATH=$PATH:$HOME/projects/endurox/dist/bin

export FLDTBLDIR=$HOME/endurox/ubftest/ubftab

export PATH=$PATH:/opt/csw/bin:$HOME/endurox/dist/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HOME/endurox/dist/lib64:/opt/csw/lib
# Solaris message queues live in tmp:
export NDRX_QPATH=/tmp

EOF

$ chmod +x $HOME/ndrx_home
```

Chapter 5

Building the code with GCC

It is assumed that gcc is default compiler on the system (i.e. Oracle Studio not installed), thus following cmake will pick up gcc by default:

5.1 Solaris 11

```
$ cd /export/home/user1/endurox
$ cmake -DCMAKE_CXX_COMPILER=g++ -DCMAKE_C_COMPILER=gcc\
-DDEFINE_DISABLEDOC=ON -DDEFINE_DISABLEGPGME=ON -DCMAKE_INSTALL_PREFIX:PATH=`pwd`/dist - ↩
    DDEFINE_FORCEEMQ=ON .
$ make
$ make install
```

5.2 Solaris 10

```
$ cd /export/home/user1/endurox
$ cmake -D CMAKE_AR=/opt/csw/gnu/ar -DCMAKE_CXX_COMPILER=g++ -DCMAKE_C_COMPILER=gcc \
-DDEFINE_DISABLEDOC=ON -DDEFINE_DISABLEGPGME=ON -DCMAKE_INSTALL_PREFIX:PATH=`pwd`/dist - ↩
    DDEFINE_FORCEEMQ=ON .
$ make
$ make install
```

Chapter 6

Building the code with Solaris Studio

The compilation will be done in 64bit mode

6.1 Solaris 11

```
$ cd /export/home/user1/endurox
$ cmake -DCMAKE_INSTALL_PREFIX:PATH=`pwd`/dist -DDEFINE_DISABLEGPGME=ON -DDEFINE_DISABLEDOC ↩
    =ON  -DDEFINE_FORCEEMQ=ON .
```

6.2 Solaris 10

This assumes that GCC is installed, and "ar" from gcc will be used.

```
$ cd /export/home/user1/endurox
$ cmake -D CMAKE_AR=/opt/csw/gnu/ar \
-DCMAKE_INSTALL_PREFIX:PATH=`pwd`/dist -DDEFINE_DISABLEGPGME=ON -DDEFINE_DISABLEDOC=ON  - ↩
    DDEFINE_FORCEEMQ=ON .
```

Chapter 7

Building the code

```
$ cd /export/home/user1/endurox
$ make
$ make install
```

This will produce binaries in */export/home/user1/endurox/dist* folder.

Chapter 8

Unit Testing

Enduro/X basically consists of two parts: . XATMI runtime; . UBF/FML buffer processing. Each of these two sub-systems have own units tests.

8.1 UBF/FML Unit testing

```
$ cd /export/home/user1/endurox/sampleconfig
$ source setndrx
$ cd /export/home/user1/endurox/ubftest
$ ./ubfunit1 2>/dev/null
Running "main" (76 tests)...
Completed "ubf_basic_tests": 198 passes, 0 failures, 0 exceptions.
Completed "ubf_Badd_tests": 225 passes, 0 failures, 0 exceptions.
Completed "ubf_genbuf_tests": 334 passes, 0 failures, 0 exceptions.
Completed "ubf_cfchg_tests": 2058 passes, 0 failures, 0 exceptions.
Completed "ubf_cfget_tests": 2232 passes, 0 failures, 0 exceptions.
Completed "ubf_fdel_tests": 2303 passes, 0 failures, 0 exceptions.
Completed "ubf_expr_tests": 3106 passes, 0 failures, 0 exceptions.
Completed "ubf_fnext_tests": 3184 passes, 0 failures, 0 exceptions.
Completed "ubf_fproj_tests": 3548 passes, 0 failures, 0 exceptions.
Completed "ubf_mem_tests": 4438 passes, 0 failures, 0 exceptions.
Completed "ubf_fupdate_tests": 4613 passes, 0 failures, 0 exceptions.
Completed "ubf_fconcat_tests": 4768 passes, 0 failures, 0 exceptions.
Completed "ubf_find_tests": 5020 passes, 0 failures, 0 exceptions.
Completed "ubf_get_tests": 5247 passes, 0 failures, 0 exceptions.
Completed "ubf_print_tests": 5655 passes, 0 failures, 0 exceptions.
Completed "ubf_macro_tests": 5666 passes, 0 failures, 0 exceptions.
Completed "ubf_readwrite_tests": 5764 passes, 0 failures, 0 exceptions.
Completed "ubf_mkfldhdr_tests": 5770 passes, 0 failures, 0 exceptions.
Completed "main": 5770 passes, 0 failures, 0 exceptions.
```

8.2 XATMI Unit testing

ATMI testing might take some time. Also ensure that you have few Gigabytes of free disk space, as logging requires some space (about ~10 GB).

```
$ cd /export/home/user1/endurox/atmitest
$ nohup ./run.sh &
$ tail -f /export/home/user1/endurox/atmitest/test.out
...
```

```
************ FINISHED TEST: [test028_tmq/run.sh] with 0 ************
Completed "atmi_test_all": 28 passes, 0 failures, 0 exceptions.
Completed "main": 28 passes, 0 failures, 0 exceptions.
```

Chapter 9

Conclusions

At finish you have a configured system which is read to process the transactions by Enduro/X runtime. It is possible to copy the binary version (*dist*) folder to other same architecture machines and run it there with out need of building.

Chapter 10

Additional documentation

10.1 Resources

[1] [BINARY_INSTALL] See Enduro/X binary_install manual.