Building Enduro/X On IBM AIX 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* on IBM AIX Platform. Document is based on AIX 7.1. Compilers supported are GNU GCC and IBM xlC.

Chapter 2

Overview

This manual includes basic installation of Enduro/X which does not include building of documentation, does not include platform script (as there is issues with c++ headers), and does not use GPG-ME encryption for bridges.

For AIX there is not management tools for Posix queues (like */dev/mqueue* file system for Linux). Thus Enduro/X have implemented wrappers for *mq_open()* call that makes empty files on file system in *NDRX_QPATH* environment folder. This is kind of registry of queues opened by Enduro/X. Thus for performance reasons, it is recommended to mount a memory based partition where middle-ware can store these files.

Chapter 3

Installation process

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

1. git
2. cmake
3. flex
4. bison
5. libxml2
6. gcc (if try GNU Compiler Chain)

Packages to be installed

The following operations will be done from root user. This will download the all open source packages to local machine to /root/rpms folder.

```
# mkdir /root/rpms
# cd /root/rpms
# ftp -i www.oss4aix.org
# ftp -i www.oss4aix.org
Connected to www.oss4aix.org.
220 FTP Server ready.
Name (www.oss4aix.org:fora): anonymous
331 Anonymous login ok, send your complete email address as your password
Password:
230-
                *** Welcome to this anonymous ftp server! ***

     You are user 2 out of a maximum of 20 authorized anonymous logins.
     The current time here is Fri Jun 17 14:52:10 2016.
     If you experience any problems here, contact : michael@perzl.org


230 Anonymous login ok, restrictions apply.
ftp> bin
200 Type set to I
ftp> passive
```

```
Passive mode on.
ftp>
ftp>
ftp> cd /compatible/aix71
250 CWD command successful
ftp> mget *
227 Entering Passive Mode (178,254,6,100,136,212).
...
```

Once all packages are downloaded. We can start to install required ones. This will include also installation of gcc. However you may skip that if you use xlC. This last step in the package install block.

```
# rpm -i cmake-3.5.1-1.aix5.1.ppc.rpm
# rpm  -i openssl-1.0.1s-1.aix5.1.ppc.rpm
# rpm  -i --nodeps git-2.2.2-2.aix5.1.ppc.rpm
# rpm -i libiconv-1.14-3.aix5.1.ppc.rpm
# rpm -i --nodeps curl-7.47.1-1.aix5.1.ppc.rpm
# rpm -i openldap-2.4.23-0.4.aix5.1.ppc.rpm
# rpm -i libssh2-1.4.3-3.aix5.1.ppc.rpm zlib-1.2.4-2.aix5.1.ppc.rpm
# rpm -i flex-2.5.37-1.aix5.1.ppc.rpm
# rpm -i bison-3.0.4-1.aix5.1.ppc.rpm  m4-1.4.17-1.aix5.1.ppc.rpm libsigsegv-2.10-1.aix5.2. ↩
    ppc.rpm
# rpm -i --nodeps libxml2-2.9.3-1.aix5.1.ppc.rpm libxml2-devel-2.9.3-1.aix5.1.ppc.rpm
# rpm -i mktemp-1.7-1.aix5.1.ppc.rpm
# rpm -U --nodeps xz-*
# rpm -i gcc-4.8.3-1.aix7.1.ppc.rpm gcc-c++-4.8.3-1.aix7.1.ppc.rpm gcc-cpp-4.8.3-1.aix7.1. ↩
    ppc.rpm  libgcc-4.8.3-1.aix7.1.ppc.rpm libstdc++-devel-4.8.3-1.aix7.1.ppc.rpm  gmp ↩
    -6.1.0-1.aix5.1.ppc.rpm  mpfr-3.1.4-1.aix5.1.ppc.rpm libmpc-1.0.3-1.aix5.1.ppc.rpm  ↩
    libstdc++-4.8.3-1.aix7.1.ppc.rpm
```

NOTE the versions of packages for you might change. For some packages we do not need such large dependency list, thus we add *--nodeps* for some.

Getting the source code

For test purposes we will prepare new user for which Enduro/X will built (this adds the in the path the /opt/freeware/bin and xlC version 13 compiler. You may modify that of your needs.

```
# useradd -m user1
# su - user1

$ bash
$ cat << EOF >> .profile
PATH=/usr/bin:/etc:/usr/sbin:/usr/ucb:$HOME/bin:/usr/bin/X11:/sbin:.:/opt/freeware/bin:/opt ↩
    /IBM/xlC/13.1.3/bin
EOF
$ chmod +x .profile
$ source .profile
$ cd /home/user1
$ GIT_SSL_NO_VERIFY=true git clone https://github.com/endurox-dev/endurox
$ cd endurox
$ git config http.sslVerify "false"
```

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

# Where app domain lives
export NDRX_APPHOME=/home/user1/endurox
# Where NDRX runtime lives
export NDRX_HOME=/home/user1/endurox/dist/bin
# Debug config too
export NDRX_DEBUG_CONF=/home/user1/endurox/sampleconfig/debug.conf

# NDRX config too.
export NDRX_CONFIG=/home/user1/endurox/sampleconfig/ndrxconfig.xml

# Access for binaries
export PATH=$PATH:/$HOME/endurox/dist/bin

# LIBPATH for .so
export LD_LIBRARY_PATH=:$HOME/endurox/dist/lib64

# UBF/FML field tables
export FLDTBLDIR=$HOME/endurox/ubftest/ubftab

################################################################
# For AIX we do not have a tools for Posix queue listings, thus
# we will use temporary files (pipes) in special folder
rm -rf /tmp/mq
mkdir /tmp/mq
export NDRX_QPATH=/tmp/mq
################################################################

EOF

$ chmod +x $HOME/ndrx_home
```

Building the code with xlC

It is assumed that xlC is default compiler on the system, thus following shall make the building ok:

```
$ export OBJECT_MODE=64
$ cd /home/user1/endurox
$ cmake -DDEFINE_DISABLEPSCRIPT=ON -DDEFINE_DISABLEDOC=ON -DDEFINE_DISABLEGPGME=ON - ↩
    DCMAKE_INSTALL_PREFIX:PATH=`pwd`/dist .
$ make
$ make install
```

Building the code with GCC

If you previously have installed gcc (C/C++) compiler open source package. Then you can build Enduro/X with GCC compiler. To prepare for GCC build, do following steps:

```
$ cd /home/user1/endurox
$ export OBJECT_MODE=64
$ export CC=gcc
$ export CXX=g++
$ export CFLAGS=-maix64
$ export CXXFLAGS=-maix64
```

```
$ cmake -DDEFINE_DISABLEPSCRIPT=ON -DDEFINE_DISABLEDOC=ON -DDEFINE_DISABLEGPGME=ON - ↩
    DCMAKE_INSTALL_PREFIX:PATH=`pwd`/dist .
$ make
$ make install
```

Chapter 4

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.

UBF/FML Unit testing

```
$ cd /home/user1/endurox/sampleconfig
$ source setndrx
$ cd /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.
```

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. Also for AIX there are small default limits of max file size. It is recommended to increase it to some 10 GB or so. To run the ATMI tests do following:

```
$ cd /home/user1/endurox/atmitest
$ nohup ./run.sh &
$ tail -f /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 5

Trouble shooting

Problems with libxml2

You may experience issues with libxml2 version between free-ware and AIX system provided. The error looks like:

```
$ ./cpmsrv
exec(): 0509-036 Cannot load program ./cpmsrv because of the following errors:
        0509-150   Dependent module /opt/freeware/lib/libxml2.a(libxml2.shr_64.o) could not ↩
            be loaded.
        0509-152   Member libxml2.shr_64.o is not found in archive
```

It seems that linker is using /ccs/lib/libxml2.a but at runtime picks up /opt/freeware/lib/libxml2.a. One way to solve this is to replace freeware version with system provided file. That could be done in following way:

```
# cd /opt/freeware/lib
# mv libxml2.a backup.libxml2.a
# ln -s /usr/ccs/lib/libxml2.a .
```

Rebuilding with other compiler

To switch the compilers, it is recommended to clean up CMake cached files before doing configuration for other compiler, for example (switching from xlC to GCC):

```
$ rm -rf CMakeCache.txt Makefile CMakeFiles/
$ export OBJECT_MODE=64
$ export CC=gcc
$ gcc
gcc: fatal error: no input files
compilation terminated.
$ export CXX=g++
$ export CFLAGS=-maix64
$ export CXXFLAGS=-maix64
$ cmake -DDEFINE_DISABLEPSCRIPT=ON -DDEFINE_DISABLEDOC=ON -DDEFINE_DISABLEGPGME=ON - ↩
    DCMAKE_INSTALL_PREFIX:PATH=`pwd`/dist .
-- The C compiler identification is GNU 4.8.3
-- The CXX compiler identification is GNU 4.8.3
-- Check for working C compiler: /usr/bin/gcc
-- Check for working C compiler: /usr/bin/gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
```

```
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/g++
-- Check for working CXX compiler: /usr/bin/g++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
...
```

Thread local storage issues

On AIX 6.1 there with gcc version 4.8.3 works on with *thread flag. However, it looks like On AIX 7.1 with the same gcc version* thread local storage is not working. The symptoms are that varous test cases fail, for example test028 (tmqueue). While this happens it is recommended to use xlC compiler.

Chapter 6

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 7

Additional documentation

Resources

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