Building Enduro/X On GNU/Linux Platform

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
1.0	2015-11	Initial draft	MV

Contents

Chapter 1

About manual

This manual describes how to build *Enduro/X* on fresh installed Ubuntu 14.04 and Centos 6.x. Process includes description of kernel configuration, required package installation and finally finishing all with unit test completion.

This installation manual assumes that OS user for installation is *user1*, located at */home/user1*. The Enduro/X system will build at path */home/user1/endurox*.

Chapter 2

Operating System configuration

To get Enduro/X unit tested and running, environmental configuration is required see ex_adminman(guides)(Enduro/X Administration Manual, Setup System) section.

Chapter 3

Getting the source & building

This chapter describes how to install dependencies on different GNU/Linux flavors and later does get the source code and builds it.

3.1 OS dependency installation

This chapter describes package installation commands on different Linux distributions.

3.1.1 System level dependencies - for Gentoo Linux

```
$ su - root
# emerge sync
# emerge -av indent asciidoc dblatex libxml2 cmake dia \
        flex bison zlib openssl app-crypt/gpgme dev-vcs/git
```

3.1.2 System level dependencies - for Debian 8.2

The same goes on Raspbian (Raspberry Pi OS)

```
$ sudo apt-get update
$ sudo apt-get install indent asciidoc dblatex libxml2-dev \
        cmake dia flex bison gcc g++ zlib1g-dev \
        libssl-dev libcrypto++9 libcrypto++-dev \
        libgpgme11-dev libxml2-dev git
```

3.1.3 System level dependencies - for Ubuntu 14.04

```
$ sudo apt-get update
$ sudo apt-get install indent asciidoc dblatex libxml2-dev \
        cmake dia flex bison gcc g++ zlib1g-dev \
        libssl-dev libcrypto++9 libcrypto++-dev \
        libgpgme11-dev libxml2-dev git
```

3.1.4 System level dependencies - for Ubuntu 16.04 (Mint 18), 18.04

```
$ sudo apt-get update
$ sudo apt-get install indent asciidoc dblatex libxml2-dev \
        cmake dia flex bison gcc g++ zlib1g-dev \
        libssl-dev libcrypto++-dev \
        libgpgme11-dev libxml2-dev libxml2-utils git
```

For Linux Mint Mate 19+ additional packages are required (or you get following error during build):

```
a2x: ERROR: missing configuration file: /etc/asciidoc/dblatex/asciidoc-dblatex.xsl
```

Install:

```
$ sudo apt-get update asciidoc-dblatex
```

3.1.5 System level dependencies - for Centos 6.x

As some packages like 'dia' does not exists in base Centos install, then few of them we will borrow from Fedora project via epel-release. Which should be installed in following way: Download the latest epel-release rpm from (e.g. epel-release-6-8.noarch.rpm or later) http://dl.fedoraproject.org/pub/epel/6/x86_64/ Install epel-release rpm:

```
$ su - root
# wget http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
# rpm -Uvh epel-release*rpm
```

Now install system depenencies

```
# yum update
# yum groupinstall 'Development Tools'
# yum install git cmake asciidoc \
        openssl openssl-devel \
        gpgme-devel dia libxml2-devel
# exit
$
```

3.1.6 System level dependencies - for Centos/RHEL/Oracle Linux 7.x

Now install system dependencies

But if you run build on **Oracle Linux**, you need to enable optional repo for asciidoc/gpgme-devel/dblatex

```
# yum install yum-utils
# yum-config-manager --enable ol7_optional_latest
```

```
# yum groupinstall 'Development Tools'
# yum install git cmake asciidoc openssl openssl-devel \
        gpgme-devel redhat-lsb dblatex libxml2-devel
```

Centos 7 does not ship with 'dia' package. Thus we will install Fodra Core package: dia-0.97.2-5.fc19.x86_64.rpm. Also we need to install additional deps to run dia.

```
# yum install -y cairo-gobject-devel gtk2 gtk2-devel gdk-pixbuf2-devel \
                libglade2-devel libgnomeui.x86_64 wget libgnomeui
# wget http://ftp.scientificlinux.org/linux/fedora/releases/19/Fodora/x86_64/os/Packages/d/ ↩
    dia-0.97.2-5.fc19.x86_64.rpm
# rpm -i --nodeps dia-0.97.2-5.fc19.x86_64.rpm
```

Seems that RHEL/Centos/Oracle Linux 7 ship with old CMake package which generates defective RPMs. Thus it The installation might give you following errors

```
$ sudo rpm -i *.rpm
        file /usr/share/man from install of endurox-3.5.1-1.x86_64 conflicts with file from ↩
             package filesystem-3.2-21.el7.x86_64
        file /usr/share/man/man3 from install of endurox-3.5.1-1.x86_64 conflicts with file ↩
             from package filesystem-3.2-21.el7.x86_64
        file /usr/share/man/man5 from install of endurox-3.5.1-1.x86_64 conflicts with file ↩
             from package filesystem-3.2-21.el7.x86_64
        file /usr/share/man/man8 from install of endurox-3.5.1-1.x86_64 conflicts with file ↩
             from package filesystem-3.2-21.el7.x86_64

$ cmake --version
cmake version 2.8.12.2
```

Install new CMake from sources:

```
$ su - root
# yum remove cmake
# exit
$ cd
$ wget https://cmake.org/files/v3.7/cmake-3.7.2.tar.gz
$ tar -xzf cmake-3.7.2.tar.gz
$ cd cmake-3.7.2
$ ./configure
$ make
$ su - root
# make install
# cmake --version
cmake version 3.7.2

CMake suite maintained and supported by Kitware (kitware.com/cmake).
```

3.1.7 System level dependencies - for Suse Linux Enterprise Server 12.3

To install all required dependencies, you need following sets of DVDs (or other sources), or later

- SLE SERVER, DVD1 (e.g. SLE-12-SP3-Server-DVD-x86_64-GM-DVD1.iso)
- SLE SERVER, DVD2 (e.g. SLE-12-SP3-Server-DVD-x86_64-GM-DVD2.iso)
- SLE SDK, DVD1 (e.g. SLE-12-SP2-SDK-DVD-x86_64-GM-DVD1.iso)
- SLE SDK, DVD2 (e.g. SLE-12-SP2-SDK-DVD-x86_64-GM-DVD2.iso)

Add these in the "Configured Software Repositories dialog" in YaST tool. Also ensure that RPM database is updated of available packages. One way to do this is Open the "Software Management" in the YaST, it will re-scan the available software sources.

installation of packages:

```
# zypper install git-core cmake flex bison gcc libxml2 libgpgme11 gcc-c++ \
libxml2-devel libgpgme-devel asciidoc cmake dia rpm-build
```

3.1.8 AsciiDoc Integration with Dia

Also Enduro/X includes documentation in sources, thus additional config is needed so that `Dia` package can build illustrations needed for manuals.

```
$ sudo mkdir /etc/asciidoc/filters/dia
$ sudo -s
# cat << EOF > /etc/asciidoc/filters/dia/dia-filter.conf
#
# AsciiDoc Dia filter configuration file.
#
# Version: 0.1

[blockdef-listing]
dia-style=template="dia-block",subs=(),posattrs=("style","file","target","size"),filter=' ↩
    dia -t png -e "{outdir={indir}}/{imagesdir={imagesdir?/}{target}" "{outdir}/{file}" { ↩
    size?-s {size}} > /dev/null'

[dia-block]
template::[image-blockmacro]
EOF
```

3.2 Getting the Source code

```
# useradd -m user1
# su - user1
$ cd /home/user1
$ git clone https://github.com/endurox-dev/endurox endurox
```

3.3 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.

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

# 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

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

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

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

#  To complete unit tests:
export NDRX_MSGSIZEMAX=1049600

# Increase stack size
ulimit -s 30751
```

```
EOF

$ chmod +x $HOME/ndrx_home
```

Note
If you develop in Gnome (e.g. Mate) session, then do 'export DESKTOP_SESSION=gnome' before run IDE (e.g. NetBeans).

3.4 Building the code

```
$ cd /home/user1/endurox
# If you want to have install folder to /home/user1/endurox/dist
# if you want system level install then run just $ cmake -DCMAKE_INSTALL_PREFIX:PATH=/usr .
$ cmake -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.

4.1 UBF/FML Unit testing

```
$ cd /home/user1/endurox/ubftest
$ ./ubfunit1 2>/dev/null
Running "main"...
Completed "main": 5751 passes, 0 failures, 0 exceptions.
```

4.2 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. To run the ATMI tests do following:

```
$ cd /home/user1/endurox/atmitest
$ nohup ./run.sh &
$ tail -f /home/user1/endurox/atmitest/test.out
...
Setting domain 2
Server executable = tpbridge    Id = 101 :      Shutdown succeeded.
Server executable = convsv21    Id = 50 :       Shutdown succeeded.
Server executable = atmi.sv21   Id = 30 :       Shutdown succeeded.
Server executable = tmsrv       Id = 10 :       Shutdown succeeded.
Shutdown finished. 4 processes stopped.
atmiclt21: no process found
************ FINISHED TEST: [test021_xafull/run.sh] with 0 ************
Running "main"...
Running "main"...
Completed "main": 21 passes, 0 failures, 0 exceptions.
```

Chapter 5

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 machine and run it there with out need of building. This process is described in [BINARY_INSTALL] guide.

Chapter 6

Additional documentation

6.1 Resources

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