LLFI Installation Guide

1 LLVM Fault Injector - LLFI

Description :
An LLVM Tool for fault injection, easily map between fault at IR and source level, configurable and extensible.

2 Pre-requisites

1. CMake installed
2. LLVM version 2.9, built with CMake
3. Python
4. Python YAML library installed (PyYAML)
5. llvm-gcc 4.2.1 (frontend for llvm 2.9)
6. Machines with 64/32 bit Linux/OS X.
7. Java 7

3 Installation

A. Steps to install Cmake

1. You need to have a C compiler already installed
2. Go to http://www.cmake.org/cmake/resources/software.html to download CMake based on your OSI.
3. In the terminal change the current working directory to the directory where the Cmake was downloaded.
4. $ ./bootstrap
5. $ make
6. $ sudo make install

B. Steps to install llvm-2.9 and llvm-gcc 4.2.1

1. Go to http://llvm.org/releases/download.html#2.9 to download LLVM source code and LLVM-GCC 4.2 Front End Binaries for your system.
2. For LLVM-GCC 4.2 Front End Binaries

- cd where-you-want-the-front-end-to-live
- gunzip –stdout llvm-gcc-4.2-version-platform.tar.gz | tar -xvf -

3. Build llvm-2.9 *** WITH CMAKE ***.
   (Note: If Mac system is used please make the following changes before building LLVM with Cmake :
   a Open the file llvm-2.9/include/llvm/ADT/IntervalMap.h
   b Find and replace the line" Node[NewNode] = this->map->newNode<NodeT>();" with "Node[NewNode] = this->map->template newNode<NodeT>();"
   c Open the file llvm-2.9/include/llvm/ADT/PointerUnion.h
   d Find and replace the line "return Ty(Val).is<T>();" with return "Ty(Val).template is<T>();"
   e Find and replace the line "return Ty(Val).get<T>();" with "return Ty(Val).template get<T>();"
)
   - $ mkdir mybuilddir
   - $ cd mybuilddir
   - $ mkdir llvm_build
   - $ cd llvm_build
   - Execute this command on the shell replacing path/to/llvm/source/root with the path to the root of your LLVM source tree:
     $ cmake path/to/llvm/source/root
   - $ make

C. Steps to install Python

1. Go to http://www.python.org/getit/ to download Python.
2. In the terminal change the current working directory to the directory where the Python was downloaded.
3. $ ./configure
4. $ make
5. $ make test
6. $ sudo make install

D. Steps to install PyYAML

1. Go to http://pyyaml.org/wiki/PyYAML to download PyYAML library.
2. In the terminal change the current working directory to the directory where the PyYAML was downloaded.
3. $ python setup.py install.

E Steps to install Java 7

1. Go to http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html to download Java 7
2. Unzip the file in your system Java source directory

F. Steps to build LLFI

1. Extract the code from LLFI archive (/LLFI-Cisco-master)

2. Go to /LLFI-Cisco-master directory and run './setup –help' to see how to build LLFI to a different directory
3. eg : $ ./setup -LLVM_DST_ROOT <LLVM CMake build root dir> -LLVM_SRC_ROOT <LLVM source root dir> -LLVM_GXX_BIN_DIR <llvm-gcc/g++'s parent dir> -LLFI_BUILD_ROOT <path where you want to build LLFI>

   (Note : <LLVM CMake build root dir>: Make sure you build LLVM with CMake and pass build root directory here
   <llvm-gcc/g++'s parent dir > (optional): You don't need to set it if it is in system path)

F. Set Environment Variables using tcsh shell

1. Set the 'PYTHONPATH' environment variable with the path of the installed Python yaml file directory .
   - **$ open .tcshrc**
   - **setenv PYTHONPATH {Path of Python yaml file directory}**
     eg: usr/Python 2.7/site-packages/
   - Create an environment variable "llfibuild" with the path of the llfi build directory.
     $ open .tcshrc
     setenv llfibuild {Path of llfi build directory}

G. Launch LLFI

1. Go to /LLFI-Cisco-master/LLFI-GUI directory and run 'java -jar llfi_gui.jar'.
2. The directory /LLFI-Cisco-master/LLFI-GUI will be the project directory.

4 Running LLFI on your target applications

For more details, you can follow the instructions on https://github.com/karthikp-ubc/LLFI-Cisco/wiki.