Installing XV6 Operating System on Ubuntu

Dr. Mayank Pandey
Motilal Nehru National Institute of Technology Allahabad, Prayagraj

Xv6 - Installation

- Use ubuntu virtual machine to separate dubug and make xv-6
- First get the system up to date :
  - *sudo apt-get update*
  - This would update the list of apt packages that can be upgraded.
  - **Does not** actually upgrade the packages, just checks how many updates have been released.

- Build-essentials : a package containing important applications for developers.
  - *sudo apt-get install build-essential*
  - Contains important compilers as well as other packages needed to build an application.

XV6 Installation (continued)

- Gcc-multilib : required for cross compilation.
  - *sudo apt-get install gcc-multilib*
  - Cross compilation : you are working on a 64 bit machine, and want to compile a program for a 32 bit architecture.

- Qemu : pc emulator to run xv6 on.
  - *sudo apt-get install qemu*

Xv6 -Installation (continued)

- Git : needed if we're installing from a Git repository
  - *sudo apt-get install git-core*
  - It is a well known version control system for projects

- Now, we need to configure git to use our proxy :
  - *git config --global http.proxy http://username:password@proxiURL:proxiPort*
  - *git config --global https.proxy http://username:password@proxiURL:proxiPort*
  - *git config --global http.sslVerify false*
  - *Replace with edcguest:edcguest@172.31.100.27:3128*
  - *If you don't do this, an HTTP 407 error is thrown by git.*

Xv6 -Installation (continued)

- Now that we have the toolchain. Get the xv6 source code :
  - *git clone -b lab1 http://compas.cs.stonybrook.edu/~nhonarmand/courses/fa17/cse306/xv6-labs.git xv6*
  - *This places the xv6 code into an xv6 folder in your home directory.*
- Now, to compile xv6 :
  - *Navigate to xv6 folder in terminal*
  - *make*
  - After that, to run xv6 on top of qemu : *make qemu*

GDB – GNU Symbolic Debugger

- It is a utility to look into the working of another program as it executes.
- Breakpoint, catchpoint, or watchpoint
  - It tells GDB to stop program execution at a certain point, so that you can inspect what is going on in the process
  - It can be a function call, or line number etc.
  - Syntax : breakpoint(or just “b”) <function, line etc.>
- Continue :
  - After you have inspected this breakpoint, it resumes execution till next breakpoint occurs.
  - Syntax : continue(or just “c”)
- Since, qemu is a virtual processor, GDB allows you to inspect what a real computer looks like it is doing when you use GDB while xv6 is running on it.
- A good introduction to GDB's working can be found here :

  https://zoo.cs.yale.edu/classes/cs422/2010/lec/l2-hw

QEMU

- It is a free and open source emulator.
- QEMU can emulate many architectures x86, ARM, MIPS etc.
- It can also emulate peripherals, network interface cards.
- This allows us to run code designed for one machine on an architecturally different physical machine.
- For example, xv6 is a 32 bit operating system, but can be run on a 64 bit system using QEMU.

References

- https://compas.cs.stonybrook.edu/~nhonarmand/courses/fa17/cse306/lab1.html
- http://sourceware.org/gdb/current/onlinedocs/gdb
- https://stackoverflow.com/questions/24907140/git-returns-http-error-407-from-proxy-after-connect
- http://janfan.cn/english-blog/english/2014/06/17/how-i-learn-os.html