
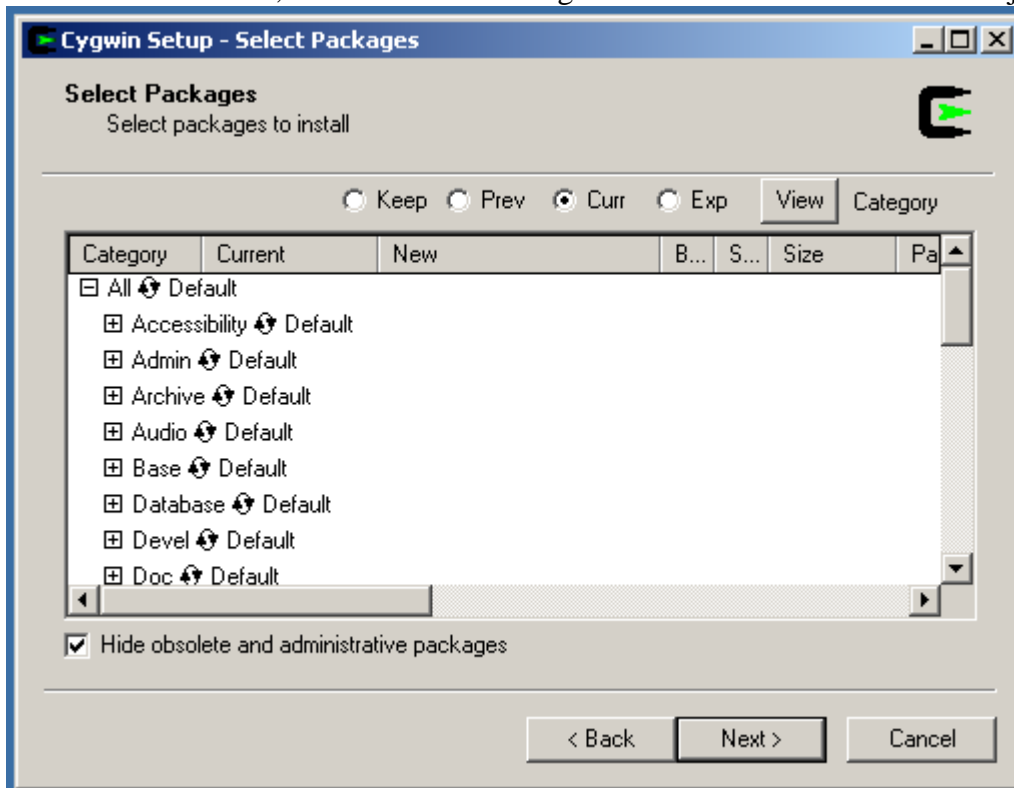


Accessing lnxsrv0x.seas.ucla.edu (where “x” is 1, 2, or 3) using cygwin (for Windows only) (**optional**)

- Download from <http://www.cygwin.com/>
- Warning: Takes up to several hours to install
- Easiest way:
 - Next to “All”, click icon  to change from default->install. This will just install everything.



- Using cygwin
 - Have to start X server. Icon name varies depending on version
 - Terminal should start up.
 - At Terminal type:
 - `ssh -2 -Y lnxsrv0x.seas.ucla.edu`
 - where “x” is 1, 2, or 3
 - If you need to specify your SEAS login name, then do this instead
 - `ssh -2 -Y -l <login name> lnxsrv0x.seas.ucla.edu`
 - where “x” is 1, 2, or 3
 - <login name> is your login name
 - From there, you can use your favorite editor to edit files
 - emacs
 - vim
 - pico
- Uploading/downloading files
 - `sftp <username>@lnxsrv0x.seas.ucla.edu`
 - where “x” is 1, 2, or 3
 - commands
 - `put <filename>`
 - uploads files
 - `get <filename>`

- downloads files

Using gdb (optional)

- GDB is used to help debug programs
- Changes needed to use GDB
 - Edit Makefile to use the `-g` flag
 - For Lab 1, data lab
 - In the file Makefile, change:
 - `CFLAGS = -O -Wall`
 - To:
 - `CFLAGS = -O -Wall -g`
 - If you're program is crashing, you want to dump out the core, so change the limit of the `coredumpsize` to unlimited. It is by default set to 0. Type the following at the command prompt before running your program
 - `limit coredumpsize unlimited`
 - Now whenever your program crashed, it will dump out the core to a file `core.XXXXX`, where `XXXXX` is some number.
- Debugging segfaults
 - Run:
 - `gdb <program> <core file>`
 - For lab 1 it would be: `gdb btest core.XXXXX`
 - Useful commands.
 - `backtrace`
 - lists the program call stack
 - `up`
 - goes up a level in the program call stack
 - `down`
 - goes down a level in the program call stack
 - `list`
 - lists the code where the segfault occurred
 - `print <variable>`
 - prints out the current contents of `<variable>`
- Running through code
 - Starting
 - `gdb <program>`
 - For Lab 1
 - `gdb btest`
 - `run`
 - starts program execution
 - Useful commands
 - `break <filename>:<function>`
 - point at which program breaks from execution.
 - `break <filename>:<line number>`
 - `delete <breakpoint>`
 - deletes the breakpoint
 - `step`
 - goes to next line of code. Steps into functions.
 - `continue`
 - continues executing code.
 - `next`
 - goes to next line of code. Does not step into functions.