

Oregon State University

School of Electrical Engineering and Computer Science

# CS 261 – Recitation 1

## Compiling C on UNIX

**Winter 2015**



# Outline

- **Secure Shell**
- Basic UNIX commands
- The GNU Compiler Collection (gcc)
- Setting up your IDE

Downloads: <http://dropline.net/cs261>

# Secure Shell Tools

- Mac OS X / Linux
  - Open a terminal
  - Type “ssh [username]@[server]”
    - *ssh kuleszto@flip.engr.oregonstate.edu*
  - scp [file] [username]@[server]
    - Transmit, Cyberduck, Filezilla, etc. (if you want a GUI)
- Windows
  - Putty ([http://engineering.oregonstate.edu/computing/fileaccess/putty\\_ssh/](http://engineering.oregonstate.edu/computing/fileaccess/putty_ssh/))
  - WinSCP (<http://winscp.net/>)

# Putty Configuration

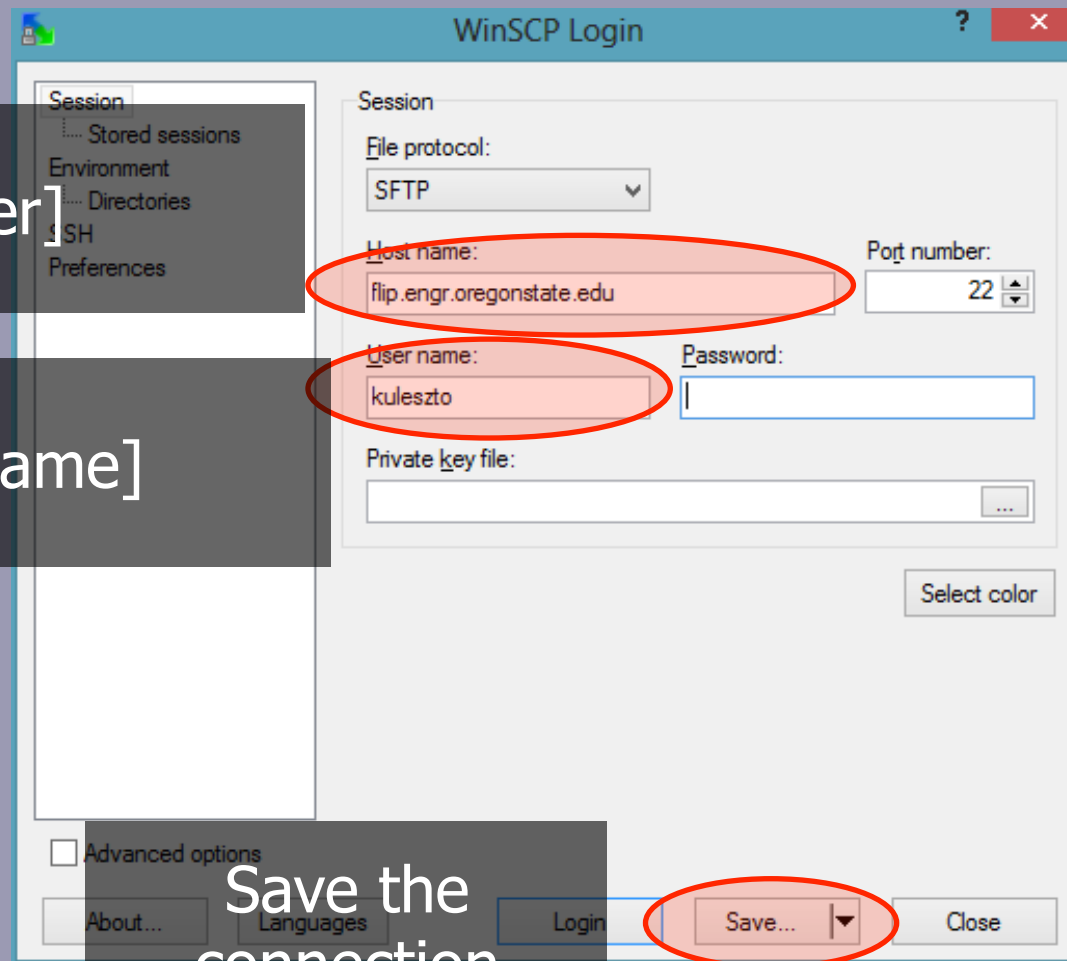
The screenshot shows the PuTTY Configuration window with several elements highlighted by red circles and annotated with text boxes:

- [username]@[server]**: A text box pointing to the Host Name field, which contains `kuleszto@flip.engr.oregonstate.edu`.
- Server name**: A text box pointing to the Saved Sessions list, which contains `flip`.
- Save the connection**: A text box pointing to the Save button.
- Open a saved connection**: A text box pointing to the Open button.

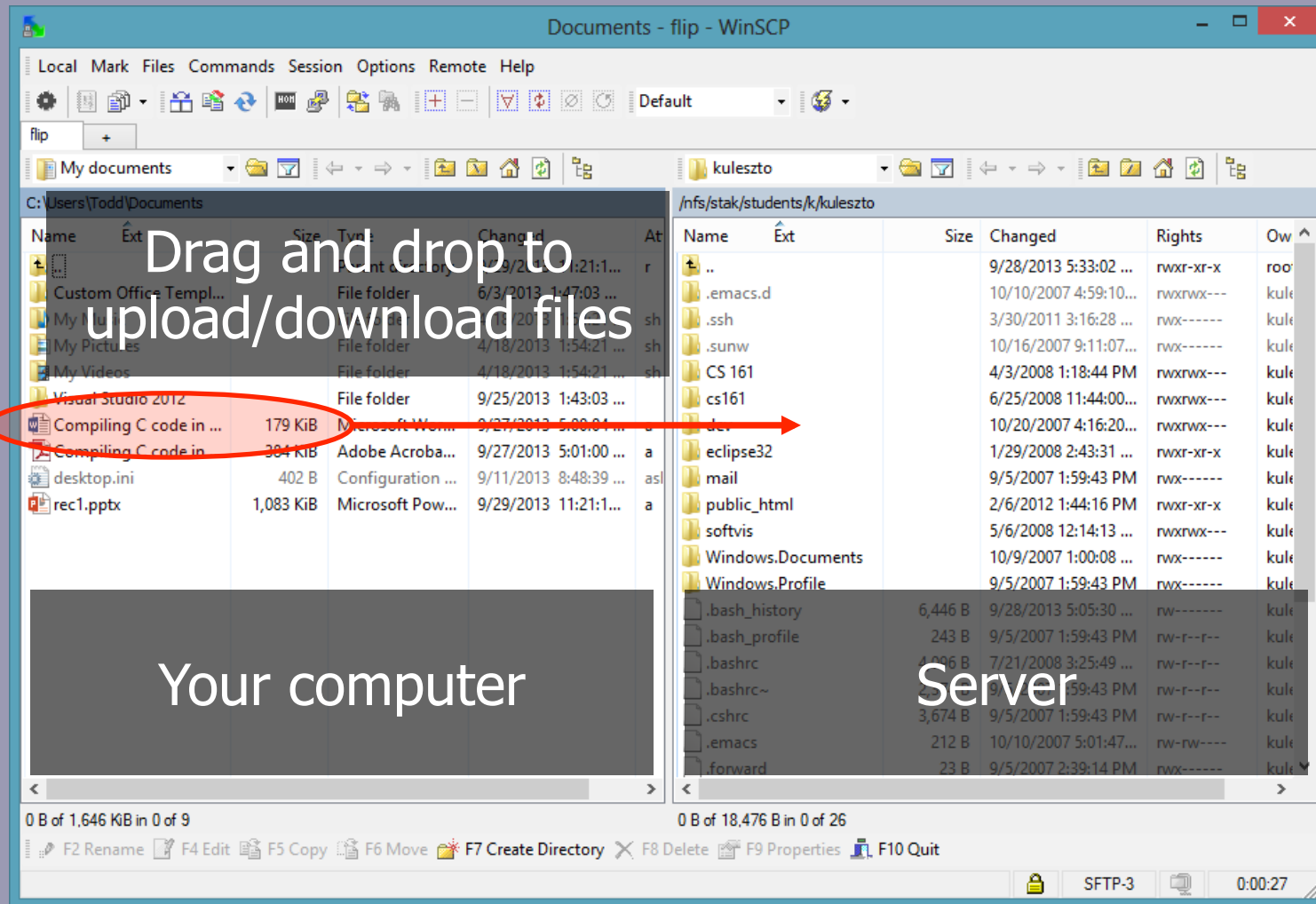
The PuTTY Configuration window itself has the following visible elements:

- Category:** A tree view on the left with categories: Session, Logging, Terminal, Window, Appearance, Behaviour, Translation, Selection, Colours, Connection, Data, Proxy, Telnet, Rlogin, SSH (selected), and Serial.
- Basic options for your PuTTY session**: The main configuration area on the right.
- Specify the destination you want to connect to**: A section containing the Host Name (or IP address) and Port fields.
- Connection type:** Radio buttons for Raw, Telnet, Rlogin, SSH (selected), and Serial.
- Load, save or delete a stored session**: A section containing the Saved Sessions list and buttons for Load, Save, and Delete.
- Close window on exit:** Radio buttons for Always, Never, and Only on clean exit (selected).
- Buttons:** About, Open, and Cancel buttons at the bottom.

# WinSCP Configuration



# WinSCP Usage



# UNIX Servers @ OSU

- ENGR
  - flip.engr.oregonstate.edu (RHEL 6.5)
  - flop.engr.oregonstate.edu (RHEL 6.5)
- ONID
  - shell.onid.oregonstate.edu (Debian 6.0)

# Outline

- Secure Shell
- **Basic UNIX commands**
- The GNU Compiler Collection (gcc)
- Setting up your IDE



# Basic UNIX Commands

Command	Description
<code>ls</code>	Lists files and folders (use " <code>ls -l</code> " for a "long" listing)
<code>cd [dir name]</code>	Change directory (" <code>cd ..</code> " will go up a directory)
<code>pwd</code>	Print name of current directory
<code>mkdir [dir name]</code>	Make a directory
<code>rmdir [dir name]</code>	Remove a directory
<code>cp [file] [new file]</code>	Copy a file (use " <code>cp -r</code> " to copy a directory)
<code>mv [file] [new file]</code>	Move (or rename) a file or directory
<code>rm [file]</code>	Remove a file
<code>cat [file]</code>	Show file contents
<code>exit</code>	Close connection to server
<code>ctrl-c</code>	Kill the current process

# Basic UNIX Commands

- For more info on a command, use the manual page:  
\$ man ls
- Tutorial: <http://www.ee.surrey.ac.uk/Teaching/Unix/>

# Outline

- Secure Shell
- Basic UNIX commands
- **The GNU Compiler Collection (gcc)**
- Setting up your IDE

# GCC

- GCC is the standard UNIX/Linux/Mac OS X compiler for about a dozen languages (e.g., C, C++, Objective C, Fortran, Java)
- Compiling with GCC:

*gcc <list of options> sourcefile.c*

e.g.: *gcc -Wall -std=c99 -o test test.c*

- Compiling multiple files:

*gcc <list of options> <list of source files>*

e.g.: *gcc -Wall -std=c99 -o test test1.c test2.c test3.c*

# Makefile Example

A makefile is like a script for the compiler. We'll provide makefiles for many of your assignments so you'll be able to compile by typing 'make'.

Contents of a makefile:

```
default:main
```

Executed when you  
type "make"

```
main: main.c
```

```
gcc -Wall -std=c99 main.c -o main
```

```
clean:
```

```
rm main main.o
```

Executed when you  
type "make clean"

# Make Tutorial

<http://mrbook.org/tutorials/make/>

# Practice

Follow the instructions under  
“Practice working from the command line”  
from  
[http://classes.engr.oregonstate.edu/eecs/  
winter2015/cs261-001/lab1.php](http://classes.engr.oregonstate.edu/eecs/winter2015/cs261-001/lab1.php)

# Outline

- Secure Shell
- Basic UNIX commands
- The GNU Compiler Collection (gcc)
- **Setting up your IDE**



# C Programming IDEs

- Mac only: **Xcode**
- Windows only: **Visual Studio**

# Always Test on UNIX!

You can use any IDE to develop and test your C application before submitting. However, UNIX is the environment in which the program will be graded.

Make sure your program will **compile and run without errors or warnings** using GCC on ***'flip.engr.oregonstate.edu'***.

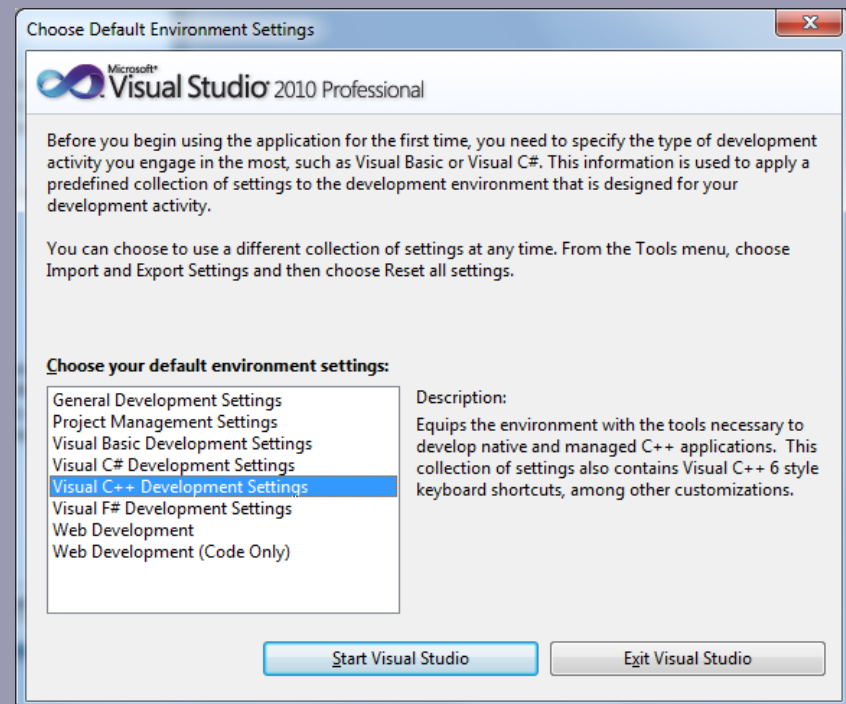
# Microsoft Visual Studio 2013

- Download it from [https://secure.engr.oregonstate.edu:8000/teach.php?type=want\\_auth](https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth)
  - Click on “Microsoft Dreamspark Login”
  - Visual Studio is under “Development Tools”

# Setting up Visual Studio

- First time you start MSVS

- The first time you start visual studio it will ask what environment settings to use.
- Select “Visual C++”
- On EECS lab machines you may wait a loooong time.



# Setting up your IDE

See our handouts on Visual Studio and Xcode

That's all for today!