



SENG 462 Tutorial #2

Chris Pearson

pearson@csc.uvic.ca



University
of Victoria

Project Deadlines

- ▶ Due tonight:
 - First log book entries
 - Validations are due Sunday night
- ▶ Due in one week (Jan 28):
 - Documentation:
 - Project Plan
 - Initial Requirements
 - Architecture
 - Demonstrated end-to-end capability
 - Demos in the tutorial Jan 28
- ▶ Due in two weeks (Feb 4):
 - Group project web site



Quick Stuff

- ▶ ~~Dedicated lab~~ Shared lab
 - You have priority on all the Linux workstations
 - Inform me if there are ANY problems with priority
- ▶ You do NOT have to use multiple servers until later
- ▶ You WILL lose 1% of your project mark for each missed logbook week



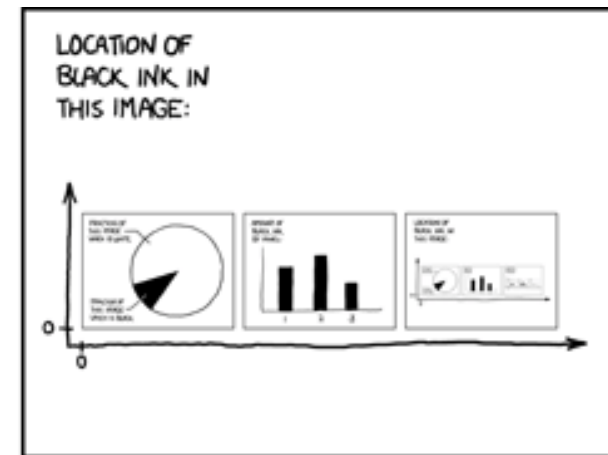
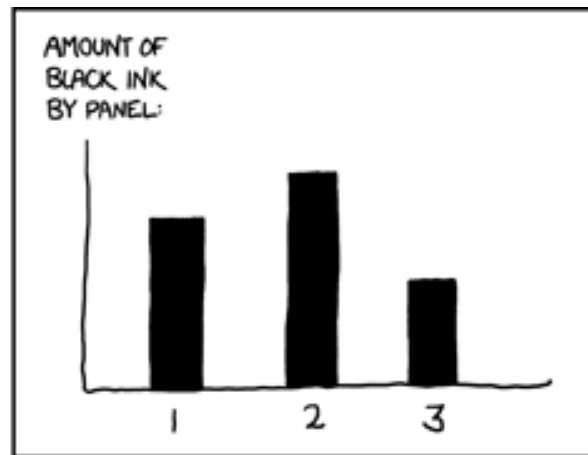
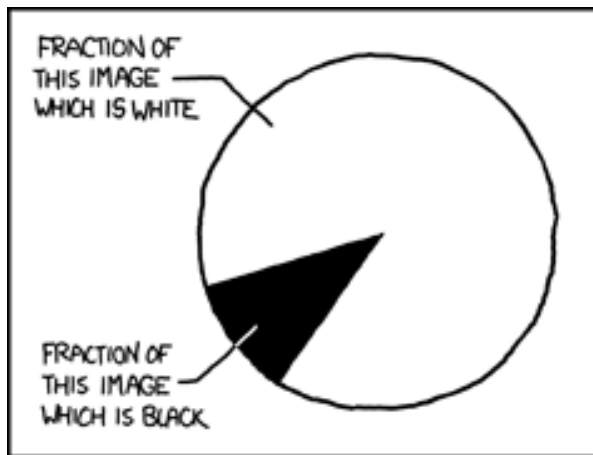
Quick Stuff (2)

- ▶ You are responsible for reading your UVic email
 - UVic mail forwarding: <https://netlink.uvic.ca/>
- ▶ RSS feed
 - The link may have not worked for some
 - <http://www.ece.uvic.ca/~seng462/ProjectWebSite/seng462tutorialfeed.xml>



Quick Stuff (3)

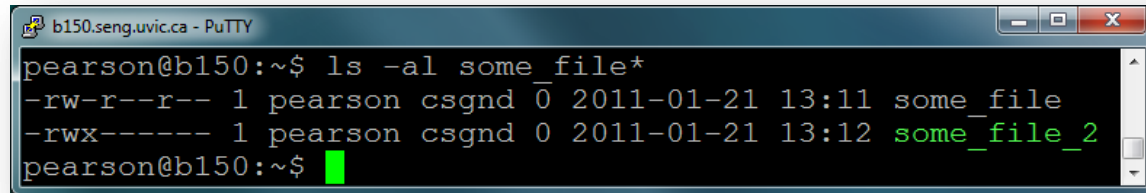
- ▶ Suggested reading: *Programmers Need To Learn Statistics Or I Will Kill Them All*
 - http://zedshaw.com/essays/programmer_stats.html
 - Do not read it if naughty language offends you



Cartoon source: xkcd.com

umask

- ▶ **umask:**
 - Sets the default permissions when editing files.
 - You want your group to be able to edit your files.
 - Group files must be accessible to all for SVN
 - See: http://www.linuxforums.org/articles/file-permissions_94.html

A terminal window titled 'b150.seng.uvic.ca - PuTTY' showing the output of the command 'ls -al some_file*'. The output lists two files: 'some_file' with permissions '-rw-r--r--' and 'some_file_2' with permissions '-rwx-----'. The user is 'pearson' and the group is 'csgnd'.

```
pearson@b150:~$ ls -al some_file*  
-rw-r--r-- 1 pearson csgnd 0 2011-01-21 13:11 some_file  
-rwx----- 1 pearson csgnd 0 2011-01-21 13:12 some_file_2  
pearson@b150:~$
```

- ▶ **How-to:**
 - Edit your `.bashrc` file
 - `le: pico /home/pearson/.bashrc`
 - Add the **umask** command: `umask 770`
 - User: full rights (7)
 - Group: full rights (7)
 - Everyone else: no rights (0)
 - **Make sure you do this for your group account**

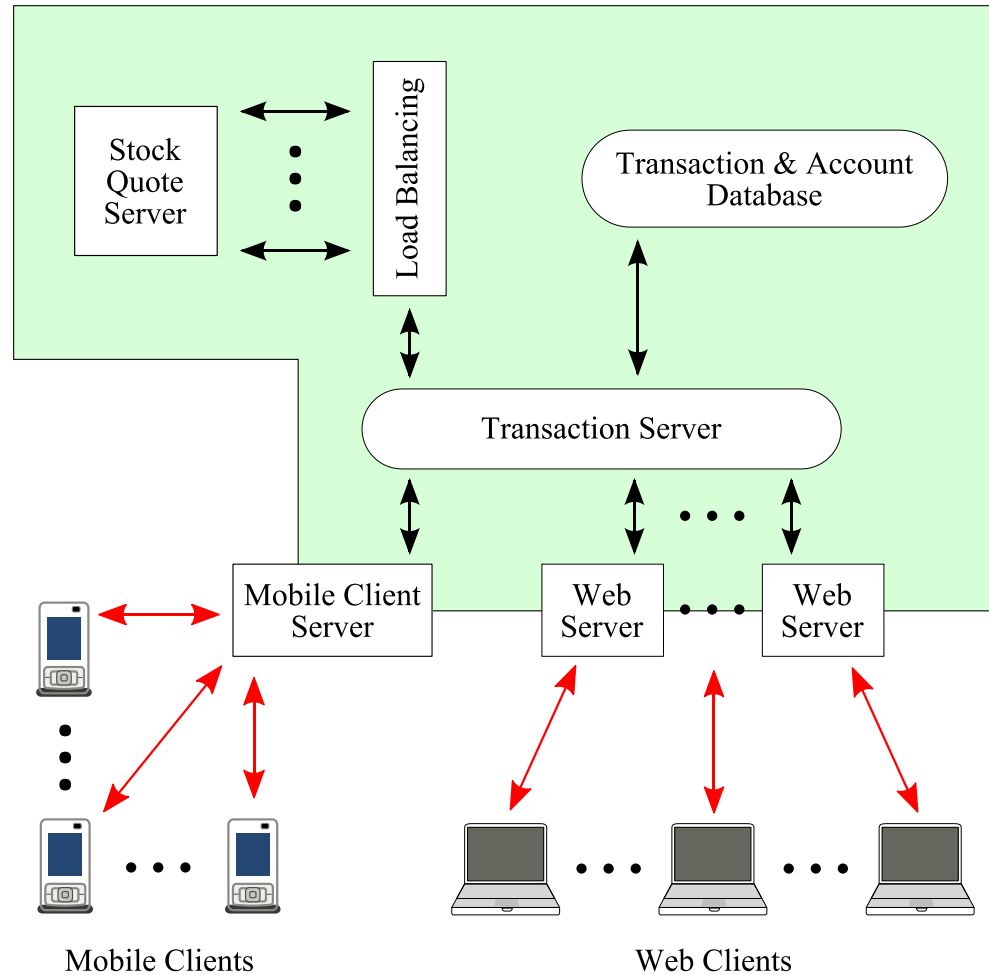


Marks, Marks, Marks

- ▶ Why to I recommend Python?
 - The marks are in the documentation
 - Python is easy to work with and easy to debug
- ▶ “Transaction and cost performance relative to the other groups”
 - Why this shouldn’t worry you...
(No, seriously, this shouldn’t worry you!)



The System



Sockets

```
import socket
# Create the socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# Connect the socket
s.connect(('www.uvic.ca',80))
# Do a simple HTTP request (use httplib in actual code.)
s.send("GET / HTTP/1.0\nHost: www.uvic.ca\n\n")
# Read and print up to 1k of data.
data = s.recv(1024)
print data
# close the connection, and the socket
s.close()
```



Sockets are Nothing Special....

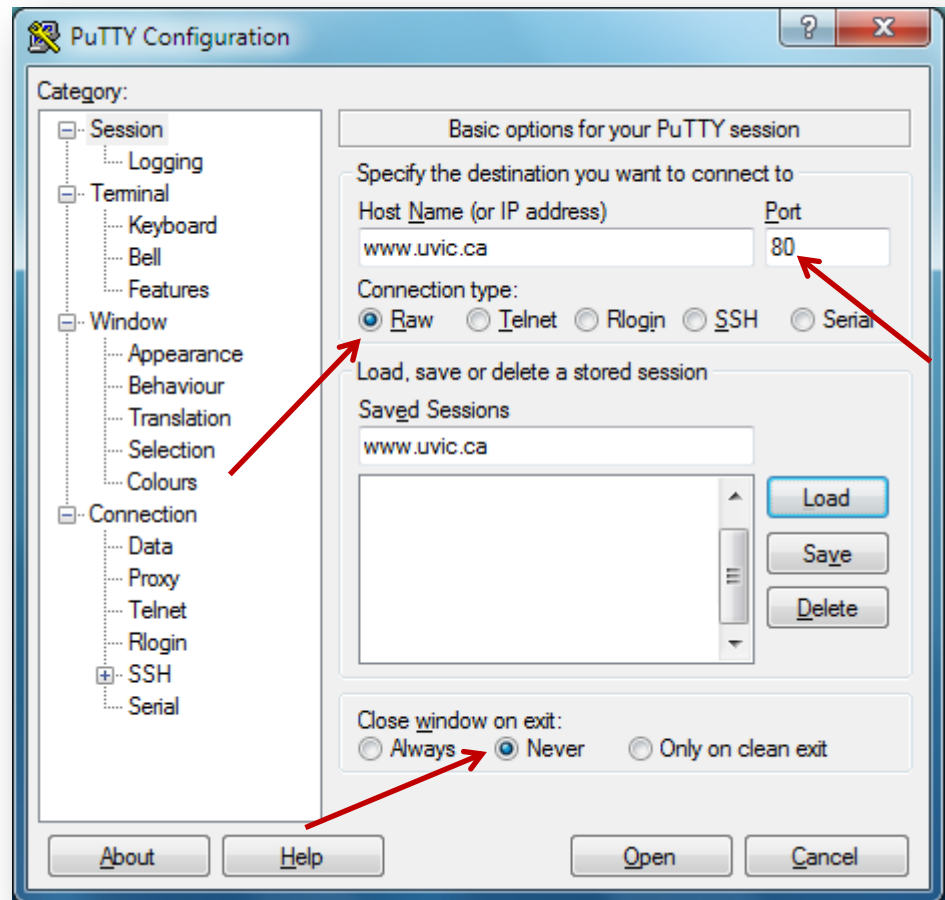
- ▶ You can connect to any unencrypted socket using PuTTY
 - Raw connection
 - Keep window open

```
GET / HTTP/1.0
Host: www.uvic.ca

HTTP/1.1 200 OK
Date: Fri, 15 Jan 2010 00:35:51 GMT
Server: Apache/2.2.13 (Unix) mod_ssl/2.2.13
Connection: close
Content-Type: text/html
Set-Cookie: BIGipServerPOOL_www.uvic.ca_prod

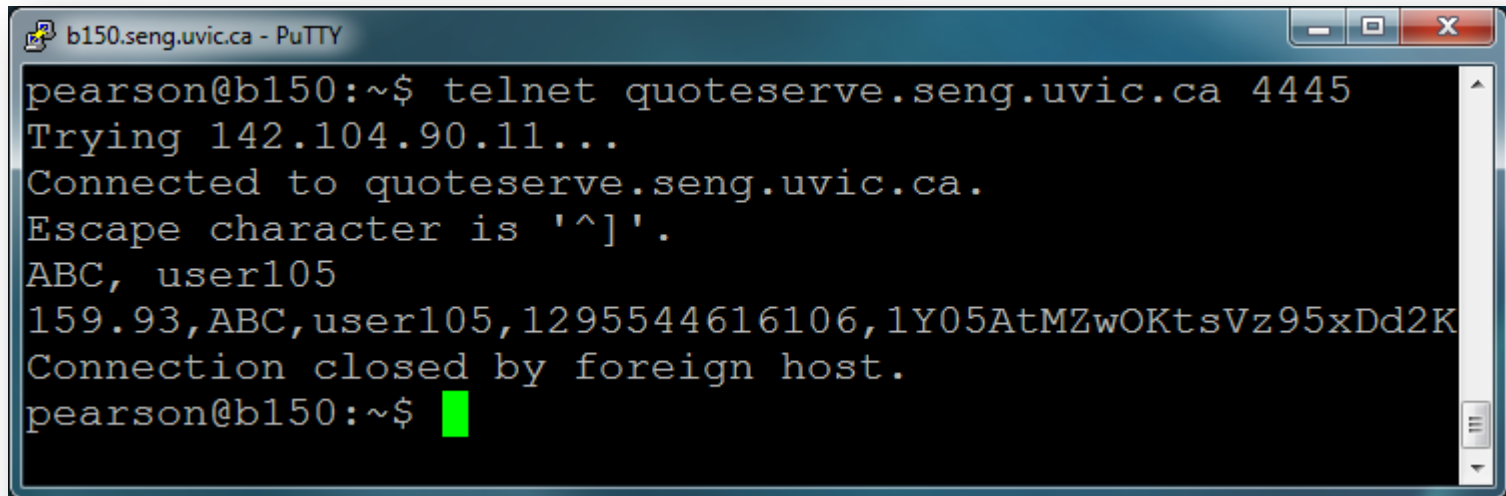
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
g/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" la
<head>
<meta http-equiv="X-UA-Compatible" content="I
<meta http-equiv="Content-Type" content="text
<title>
University of Victoria
_
```



Sockets are Nothing Special....

- ▶ So what about the quote server...?



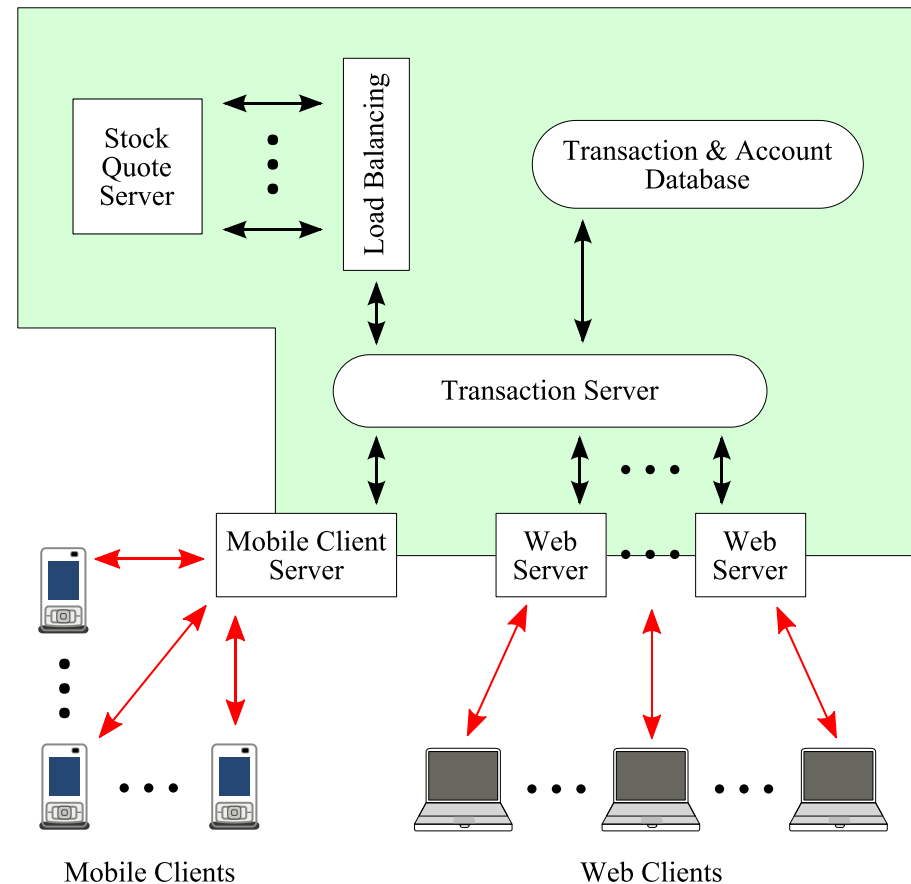
```
b150.seng.uvic.ca - PuTTY
pearson@b150:~$ telnet quoteserve.seng.uvic.ca 4445
Trying 142.104.90.11...
Connected to quoteserve.seng.uvic.ca.
Escape character is '^]'.
ABC, user105
159.93,ABC,user105,1295544616106,1Y05AtMZwOKtsVz95xDd2K
Connection closed by foreign host.
pearson@b150:~$
```

- ▶ Yep, it's exactly the same!
 - Note that the quote server will only reply to requests from within the SENG network



Secure Socket Layer (SSL)

- ▶ Some network traffic must be secured:
 - Between web browsers and web servers
 - Between N95 and its server



Apache and SSL

▶ See other instructions at:

- `http://www.devside.net/guides/linux/apache-ssl-deflate`

▶ First download OpenSSL

- `wget http://www.openssl.org/source/openssl-1.0.0c.tar.gz`

▶ Unpack

- `tar -xvzf openssl-1.0.0c.tar.gz`



Apache and SSL

- ▶ Not root access
 - Without root access, you must tell it where to install using “--prefix”
 - This demo will use `/home/pearson/local`
- ▶ Make a “local” directory
 - `mkdir /home/pearson/local`
- ▶ Build OpenSSL
 - `cd ~`
 - `mkdir local`
 - `cd openssl-1.0.0c`
 - `./config --prefix=/home/pearson/local/openssl`
 - `make -j 2`
 - `make install`



Apache

► Download Apache

- `wget http://apache.mirror.rafael.ca/httpd/httpd-2.2.17.tar.gz`

► Unpack

- `tar -xvzf httpd-2.2.17.tar.gz`



Apache

► Build Apache

- `cd httpd-2.2.17`
- `./configure --prefix=/home/pearson/local/apache --enable-ssl --with-ssl=/home/pearson/local/openssl`
- `make -j 2`
- `make install`



modwsgi (for Python)

▶ Download modwsgi

- `wget`
`http://modwsgi.googlecode.com/files/mod_wsgi-3.3.tar.gz`

▶ Unpack

- `tar -xvzf mod_wsgi-3.3.tar.gz`



modwsgi (for Python)

► Build modwsgi

- `cd mod_wsgi-3.3`
- `./configure --with-apxs=/home/pearson/local/apache/bin/apxs`
- `make -j 2`
 - Ignore these warnings:
 - warning: `"_POSIX_C_SOURCE"` redefined
 - warning: `"_XOPEN_SOURCE"` redefined
- `make install`



modwsgi (for Python)

- ▶ Create a modwsgi test file:

- `/home/pearson/local/apache/htdocs/test.wsgi`

- ▶ It should contain the following Python code:

```
def application(environ, start_response):  
    status = '200 OK'  
    output = 'Hello World!'  
  
    response_headers = [('Content-type', 'text/plain'),  
                        ('Content-Length', str(len(output)))]  
    start_response(status, response_headers)  
  
    return [output]
```



Configure Apache

► Make encryption keys

- `cd ~/local/apache/conf`
- `~/local/openssl/bin/openssl req -new -out server.csr`
 - Any temp passphrase will do
 - Do not make “A challenge password” at the end of Distinguished Name
- `~/local/openssl/bin/openssl rsa -in privkey.pem -out server.key`
 - This removes the passphrase from the private key
- `~/local/openssl/bin/openssl x509 -in server.csr -out server.crt -req -signkey server.key -days 365`
 - Generates the certificate / public key



Configure Apache

- ▶ `cd ~/local/apache/conf`
- ▶ `vi httpd.conf`
- ▶ **Remove comment from SSL include line:**
 - `# Secure (SSL/TLS) connections`
 - `Include conf/extra/httpd-ssl.conf`
- ▶ **Comment out the existing listen line:**
 - `#Listen 80`
- ▶ **Add two lines for modwsgi:**
 - `LoadModule wsgi_module modules/mod_wsgi.so`
 - `WSGIScriptAlias /test
/home/pearson/local/apache/htdocs/test.wsgi`



Configure Apache

- ▶ `vi extra/httpd-ssl.conf`
 - Configure this to be the port you wish to use
 - `ie: Listen 44443`
 - Edit the Virtual Host
 - `<VirtualHost _default_:44443>`
 - `ServerName b150.seng.uvic.ca:44443`



Start Apache

- ▶ The standard start command:

- `~/local/apache/bin/apachectl start`

- ▶ Other commands:

- `~/local/apache/bin/apachectl restart`

- Tells Apache to restart, but not to interrupt any pages being sent.

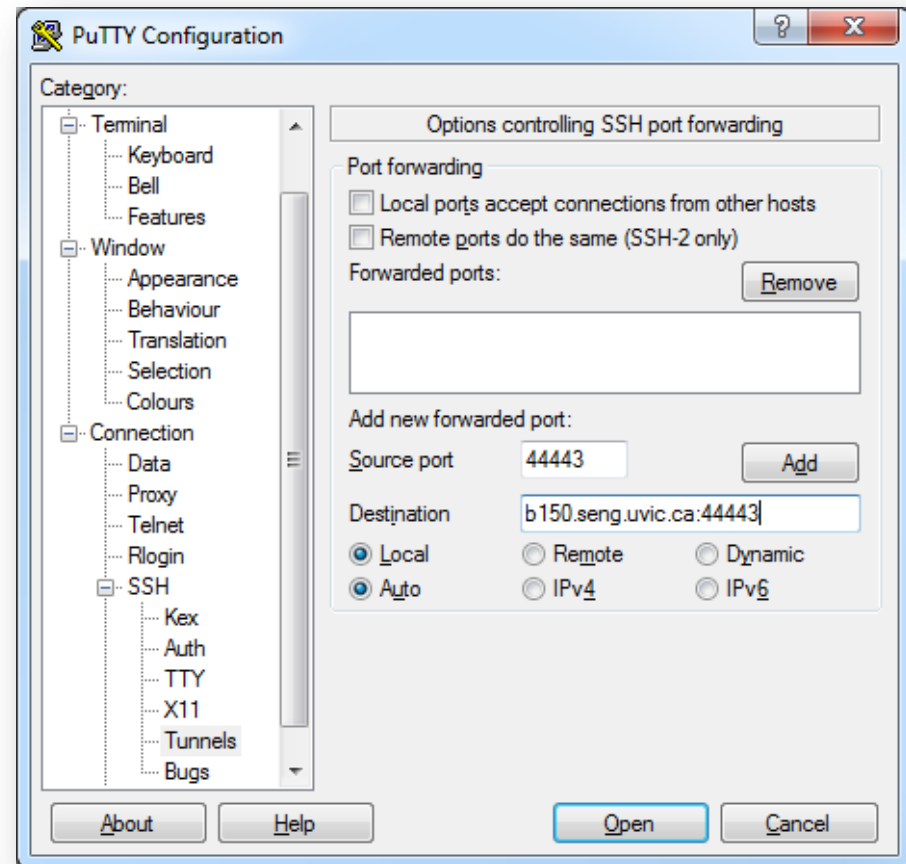
- `~/local/apache/bin/apachectl stop`

- Forces Apache to immediately stop



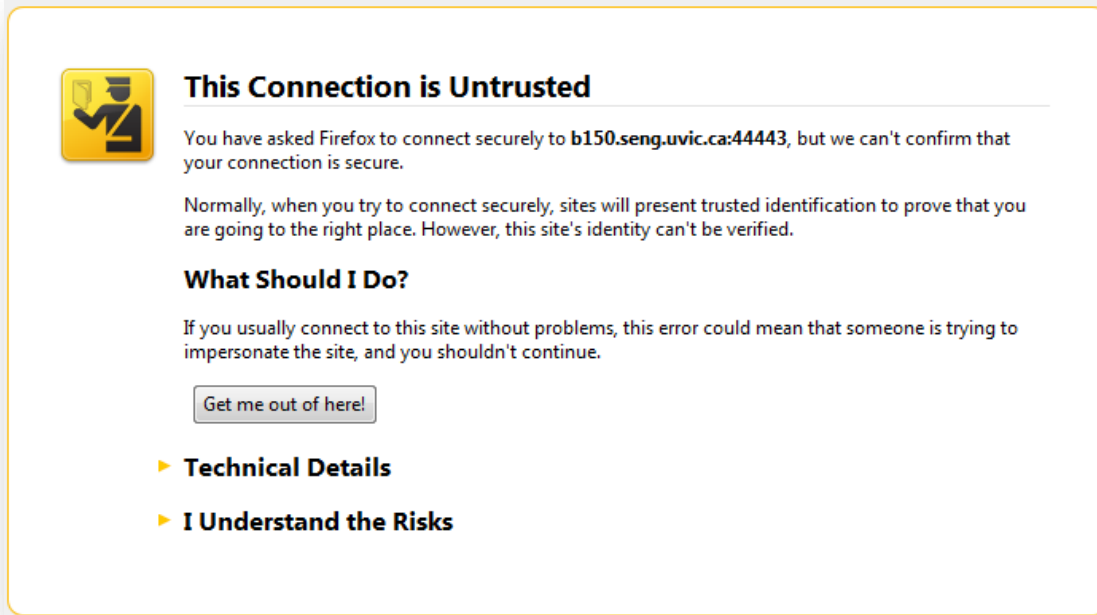
Test Apache

- ▶ From the Engineering network:
 - `https://b150.seng.uvic.ca:44443`
- ▶ From a non-Engineering network, use tunnels
 - `http://putty.org/`
 - Set up port forwarding in Putty
 - `https://127.0.0.1:44443`



Test Apache

- ▶ Add an exception for this warning:



This Connection is Untrusted

You have asked Firefox to connect securely to **b150.seng.uvic.ca:44443**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

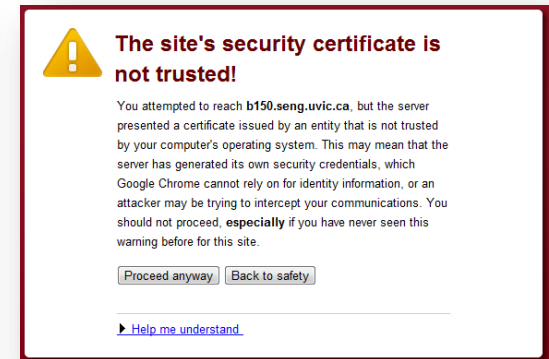
What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

[Get me out of here!](#)

- ▶ **Technical Details**
- ▶ **I Understand the Risks**

Firefox



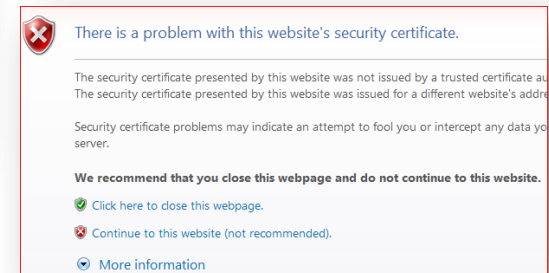
The site's security certificate is not trusted!

You attempted to reach **b150.seng.uvic.ca**, but the server presented a certificate issued by an entity that is not trusted by your computer's operating system. This may mean that the server has generated its own security credentials, which Google Chrome cannot rely on for identity information, or an attacker may be trying to intercept your communications. You should not proceed, **especially** if you have never seen this warning before for this site.

[Proceed anyway](#) [Back to safety](#)

[Help me understand](#)

Google Chrome



There is a problem with this website's security certificate.

The security certificate presented by this website was not issued by a trusted certificate authority. The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

[Click here to close this webpage.](#)

[Continue to this website \(not recommended\).](#)

[More information](#)

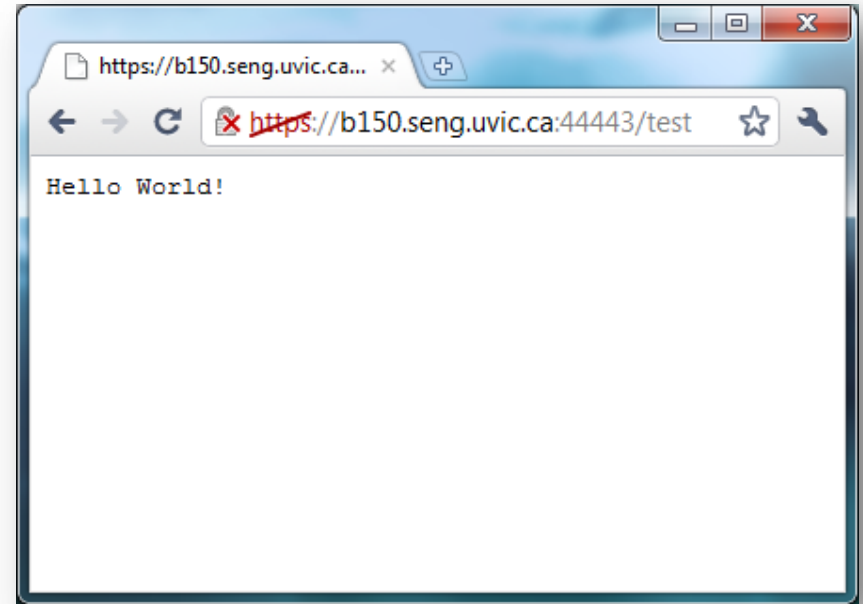
Internet Explorer



Test Apache



The Apache test page



The modwsgi quick start app



Future Tutorials

- ▶ Approximate schedule – this is flexible:
 1. Intro
 2. Discuss the system as a whole
 3. Building custom servers on the lab systems
 - Apache web server
 - OpenSSL for secure (https) web pages
 - Databases
 - Whatever else you need to custom build
 4. Sockets, moving data between systems securely, etc
 5. Using the command files with a workload generator
 6. Log files – what data you need to submit for the deadlines
 7. Testing, collecting data and statistics
 - Possibly talk about jUnit/PyUnit testing, if requested
 8. PyS60 – Python on the Nokia N95 phones
 9. Optimizations – databases and the quote server





SENG 462 Tutorial #2

Chris Pearson

pearson@csc.uvic.ca



University
of Victoria