SENG 462 Tutorial #3

Chris Pearson
pearson@csc.uvic.ca

University of Victoria

Today

- Review of networking, ports, and sockets
- More detail about web requests
  - What the protocol is
  - Why plain text protocols can be so nice
  - The standard methods of passing data via web requests
- Load Generators & Workload Files
- Log files

University of Victoria

Project Deadlines

- Due now:
  - Demonstrated end-to-end capability
- Due tonight:
  - Documentation:
    - Project Plan
    - Initial Requirements
    - Architecture
  - Printed copy AND emailed PDF
- Due in one week (Feb 4):
  - Group project web site
- Due in two weeks (Feb 11):
  - Verified execution of 1 user workload file

Notes

- Remember:
  - You are not expected to already know everything required to do this project.
- ASK QUESTIONS.
  - You don’t get to 4th year by being stupid
  - If you have a question, others in the room have the same question

Networking and Ports

- The part you know:
  - You connect to another computer via its name
    - ie: www.uvic.ca
- The part you probably know:
  - Your computer actually connects by IP address
    - ie:142.104.193.247
  - Your computer contacts a DNS to get the IP address
- The part you might not know:
  - You give the other computer a port number
    - ie: 22 = telnet, 80 = HTTP, 443 = HTTPS
  - This tells the remote computer what service to you want to communicate with

University of Victoria

Sockets

- A metaphor for a connection
- The server on the listening / hosting side opens a socket at a port number
  - ie: A web server listens at port 80
- A computer that wants to use that service makes a connection to that socket
  - They "plug into the socket", so to speak

Web Services

- Last week, we connected to a web server with PuTTY and sent a command like:
  - `GET /path/to/file/index.html HTTP/1.0`
- The protocol is simple:
  - `<initial line>` ⇦ Different for requesting vs. replying

```
<initial line>
Header1: value1
Header2: value2
Header3: value3

<optional message body goes here, such as file
contents or query data; it can be many lines
long, and can contain binary data>
```

Web Services (2)

- Request example:

  -
    ```
    GET /path/file.html HTTP/1.0
    From: someuser@example.com
    User-Agent: HTTPTool/1.0
    ```
- Reply example:

  -
    ```
    HTTP/1.0 200 OK
    Date: Fri, 22 Jan 2010 13:30:00 PST
    Content-Type: text/html
    Content-Length: 1354

    <html>
    <body>
    Some sort of example web page text
    </body>
    </html>
    ```

Transmitting Data to a Server

- First method: POST
  - This is how a typical web form works
  - Variables are set in the web page
  - Data is set when “submit” is pressed

- Example:

  -
    ```
    POST /path/to/script.cgi HTTP/1.0
    From: somebody@example.com
    User-Agent: HTTPTool/1.0
    Content-Type: application/x-www-form-urlencoded
    Content-Length: 33

    var1=something&var2=another_thing
    ```

Transmitting Data to a Server (2)

- Second method: GET
  - This is really a modification to the "normal" method of requesting web data
  - Data can be encoded into the URL
    - `ie: www.uvic.ca/index.html?var1=a&var2=b`
- Example:

```
GET /path/to/script.cgi?var1=a&var2=b HTTP/1.0
From: somebody@example.com
User-Agent: HTTPTool/1.0
```

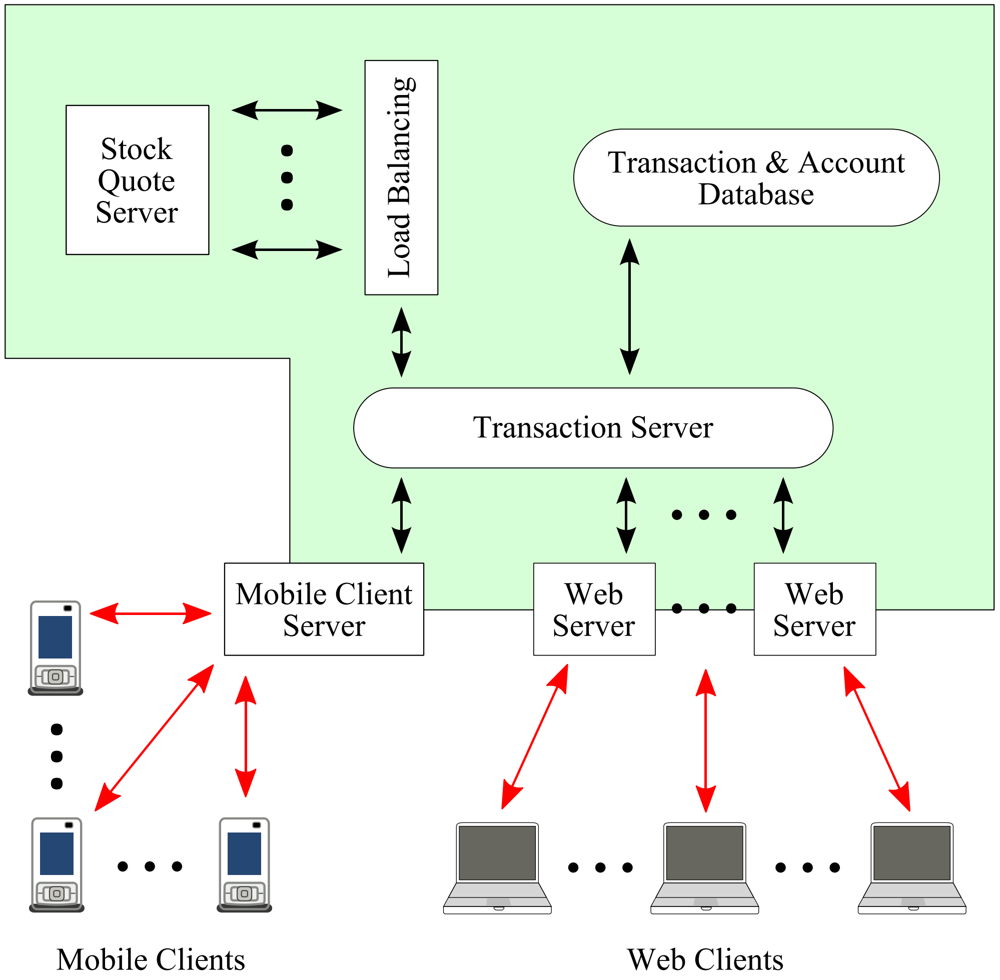
University of Victoria

Listening to Sockets

```
import socket
# Create the socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(('', 80))          # Bind the socket to a port
s.listen(1)               # Allow 1 waiting connection
conn, addr = s.accept() # Wait for a connection
# Could spawn a thread for the connection here,
# then return to listening for new connections
while 1:                  # Echo the data back to the sender
    data = conn.recv(1024)
    if not data: break
    conn.send(data)
conn.close()              # Close the connection
```

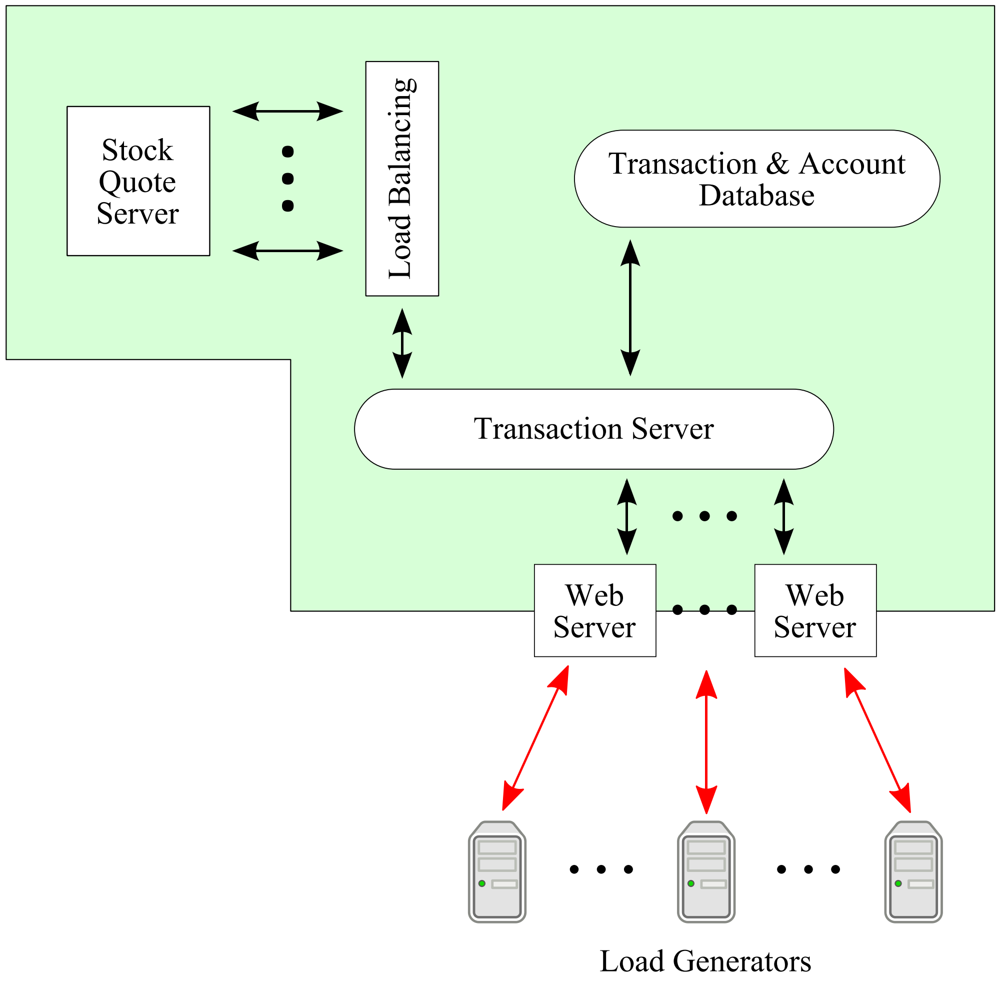
University of Victoria

The System



Stock Quote Server
Load Balancing
Transaction & Account Database
Transaction Server
Mobile Client Server
Web Server
Web Server
Mobile Clients
Web Clients

Load Generators



Workload Generator(s)

- Simulate web interface use
  - The web server "shouldn't know the difference" between a real user and the workload generator
- Input a workload data file

```
[1] ADD,oY01WVirLr,63511.53
[2] QUOTE,oY01WVirLr,S
[3] BUY,oY01WVirLr,S,276.83
[4] COMMIT_BUY,oY01WVirLr
[5] COMMIT_SELL,oY01WVirLr
[6] CANCEL_SELL,oY01WVirLr
[7] QUOTE,oY01WVirLr,S
```

Workload Files

- The list of commands is on the project website
- Do <u>not</u> expect the workload files to be perfect
  - At least one has an intentional error in it

- Final documentation hint: At what points in your system must there be error checking?

University of Victoria

Log Files

- Transactions on your system must be logged
  - These logs are submitted to the SENG 462 website to confirm completion of workload file milestones
- The contents of the files are fully described in the XML Schema file `logfile.xsd`, available on the project web site

Log File XSD

- The log entries are defined by `logfile.xsd`

```
<xsd:complexType name="LogType">
 <xsd:choice minOccurs="0" maxOccurs="unbounded">
  <xsd:element name="userCommand" type="UserCommandType
  <xsd:element name="quoteServer" type="QuoteServerType
  <xsd:element name="accountTransaction" type="AccountT
  <xsd:element name="systemEvent" type="SystemEventType
  <xsd:element name="errorEvent" type="ErrorEventType"/
  <xsd:element name="debugEvent" type="DebugType"/>
 </xsd:choice>
</xsd:complexType>
```

Log File Examples

- Transaction number follow an individual transaction through the system
- They do not have to match the numbers in the workload file

```
<userCommand>
        <timestamp>1167631200000</timestamp>
        <server>CLT1</server>
        <transactionNum>1</transactionNum>
        <command>ADD</command>
        <username>jiosesdo</username>
        <funds>100.00</funds>
</userCommand>
<accountTransaction>
        <timestamp>1167631200200</timestamp>
        <server>CLT2</server>
        <transactionNum>1</transactionNum>
        <action>add</action>
        <username>jiosesdo</username>
        <funds>100.00</funds>
</accountTransaction>
```

University of Victoria

Log File Examples (2)

- Quote server hits include a crypto key
- The crypto keys must match the logged quote server data
    - Yes, this is checked

```
<quoteServer>
        <timestamp>1167631203000</timestamp>
        <server>QSRV1</server>
        <transactionNum>2</transactionNum>
        <quoteServerTime>1167631203000</quoteSe
        <username>jiosesdo</username>
        <stockSymbol>ABC</stockSymbol>
        <price>10.00</price>
        <cryptokey>IRrR7UeTO35kSWUgG0QJKmB35sL2
</quoteServer>
```

University of Victoria

Log File Examples (3)

```
<userCommand>
        <timestamp>1167631205200</timestamp>
        <server>CLT2</server>
        <transactionNum>4</transactionNum>
        <command>SELL</command>
        <username>bob</username>
        <stockSymbol>GHI</stockSymbol>
        <funds>1000.00</funds>
</userCommand>
<errorEvent>
        <timestamp>1167631206000</timestamp>
        <server>CLT2</server>
        <transactionNum>4</transactionNum>
        <command>SELL</command>
        <username>bob</username>
        <stockSymbol>GHI</stockSymbol>
        <funds>1000.00</funds>
        <errorMessage>Account bob does not exist</errorMessage>
</errorEvent>
```

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 #3

Chris Pearson
pearson@csc.uvic.ca

University of Victoria