Networked Programs

Getting Material from the Web!

Building a Web Browser!
• So far we’ve dealt with files. We’ve read data from them, but it’s possible to write data to them too.

• Servers across the internet are similar to files, except that in addition to getting data from them you also send them requests — which may result in data.

• You do this with sockets.

• If you write something to a socket, it is sent to the application at the other end of the socket. If you read from the socket, you are given the data that the other application has sent.
• The programs on both ends of this socket need to communicate in a way both understand, so one program isn’t waiting in vain for the other to send something.

• The *protocol* that the web uses for this is called HTTP: Hyper-Text Transport Protocol.

• There are many elements to this protocol. We’ll be interested in only a few.
Network Architecture....
Transport Control Protocol (TCP)

- Built on top of IP (Internet Protocol)
- Assumes IP might lose some data - stores and retransmits data if it seems to be lost
- Handles “flow control” using a transmit window
- Provides a nice reliable pipe

TCP Connections / Sockets

“In computer networking, an Internet socket or network socket is an endpoint of a bidirectional inter-process communication flow across an Internet Protocol-based computer network, such as the Internet.”

http://en.wikipedia.org/wiki/Internet_socket
TCP Port Numbers

• A port is an application-specific or process-specific software communications endpoint

• It allows multiple networked applications to coexist on the same server.

• There is a list of well-known TCP port numbers

http://en.wikipedia.org/wiki/TCP_and_UDP_port
Please connect me to the web server (port 80) on http://www.dr-chuck.com
Common TCP Ports

- Telnet (23) - Login
- SSH (22) - Secure Login
- HTTP (80)
- HTTPS (443) - Secure
- SMTP (25) (Mail)
- IMAP (143/220/993) - Mail Retrieval
- POP (109/110) - Mail Retrieval
- DNS (53) - Domain Name
- FTP (21) - File Transfer

Sometimes we see the port number in the URL if the web server is running on a “non-standard” port.
Sockets in Python

- Python has built-in support for TCP Sockets

```python
import socket
mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect( ('www.py4inf.com', 80) )
```

http://docs.python.org/library/socket.html
Application Protocol

• Since TCP (and Python) gives us a reliable socket, what do we want to do with the socket? What problem do we want to solve?

• Application Protocols
  • Mail
  • World Wide Web

HTTP - Hypertext Transport Protocol

- The dominant Application Layer Protocol on the Internet
- Invented for the Web - to Retrieve HTML, Images, Documents, etc.
- Extended to be data in addition to documents - RSS, Web Services, etc.
- Basic Concept - Make a Connection - Request a document - Retrieve the Document - Close the Connection

http://en.wikipedia.org/wiki/Http
HTTP

The HyperText Transport Protocol is the set of rules to allow browsers to retrieve web documents from servers over the Internet.
What is a Protocol?

- A set of rules that all parties follow so we can predict each other’s behavior
- And not bump into each other
  - On two-way roads in USA, drive on the right-hand side of the road
  - On two-way roads in the UK, drive on the left-hand side of the road
http://www.dr-chuck.com/page1.htm

http://www.youtube.com/watch?v=x2GylLq59rI

1:17 - 2:19
Getting Data From The Server

- Each time the user clicks on an anchor tag with an href= value to switch to a new page, the browser makes a connection to the web server and issues a “GET” request - to GET the content of the page at the specified URL.

- The server returns the HTML document to the browser, which formats and displays the document to the user.
Making an HTTP request

- Connect to the server like www.dr-chuck.com
  - a “hand shake”
- Request a document (or the default document)
  - GET http://www.dr-chuck.com/page1.htm
  - GET http://www.mlive.com/ann-arbor/
  - GET http://www.facebook.com
The First Page

If you like, you can switch to the Second Page.
The First Page

If you like, you can switch to the Second Page.
The First Page

If you like, you can switch to the Second Page.
GET http://www.dr-chuck.com/page2.htm
GET http://www.dr-chuck.com/page2.htm

<h1>The Second Page</h1>
<p>If you like, you can switch back to the <a href="page1.htm">First Page</a>.</p>
GET http://www.dr-chuck.com/page2.htm

<h1>The Second Page</h1>
<p>If you like, you can switch back to the <a href="page1.htm">First Page</a>.</p>

The First Page
If you like, you can switch to the Second Page.

The Second Page
If you like, you can switch back to the First Page.
Internet Standards

- The standards for all of the Internet protocols (inner workings) are developed by an organization
- Internet Engineering Task Force (IETF)
- www.ietf.org
- Standards are called “RFCs” - “Request for Comments”

Hypertext Transfer Protocol -- HTTP/1.1

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information
5 Request

A request message from a client to a server includes, within the first line of that message, the method to be applied to the resource, the identifier of the resource, and the protocol version in use.

Request = Request-Line ; Section 5.1
* ( ( general-header ; Section 4.5
| request-header ; Section 5.3
| entity-header ) CRLF) ; Section 7.1
CRLF
[ message-body ] ; Section 4.3

5.1 Request-Line

The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by SP characters. No CR or LF is allowed except in the final CRLF sequence.

Request-Line = Method SP Request-URI SP HTTP-Version CRLF
“Hacking” HTTP

$ telnet www.dr-chuck.com 80
Trying 74.208.28.177...
Escape character is '^]'.
GET http://www.dr-chuck.com/page1.htm HTTP/1.0

<h1>The First Page</h1>
<p>If you like, you can switch to the Second Page</p>

Port 80 is the non-encrypted HTTP port
$ telnet www.dr-chuck.com 80
Trying 74.208.28.177...
Connected to www.dr-chuck.com. Escape character is '^['].
GET http://www.dr-chuck.com/page1.htm HTTP/1.0

<h1>The First Page</h1>
<p>If you like, you can switch to the
Connection closed by foreign host.
% telnet www.dr-chuck.com 80
New: Tsugi: A PHP framework for IMS LTI Tools
New: MOOCs: Charles Severance at TEDxKalamazoo

Free Courses / Educational Material:
- Coursera: Programming for Everybody (Python)
- Coursera: Internet History, Technology and Security
- SI 502 - Networked Computing

Books
- Sakai: Building an Open Source Community (2011, 2014)
- Using Google App Engine (O'Reilly 2009)

Web/Multimedia sites
- IEEE Computer - Computing Conversations Interviews (2011-present)
- Dr. Chuck sings the blues (2008)
- Dr. Chuck goes motocross racing (2007)
- A Film About Brent and His ATV (2005)
- Audition Tape (2003) for TechTV which was rejected :(
- Dr. Chuck goes stock car racing (2002)
- The Community Information Toolkit - A project to provide public libraries and other organizations a start on using Internet in Community Networking. (1999)

Software
- The Sakai Collaboration and Learning Environment
- Tsugi: A PHP framework for IMS LTI Tools
- IMS Learning Tools Interoperability
- My Reference Implementation of Mozilla Open Badges in PHP
A browser debugger reveals detail...

- Most browsers have a developer mode so you can watch it in action.
- It can help explore the HTTP request-response cycle.
- Some simple-looking pages involve lots of requests:
  - HTML page(s)
  - Image files
  - CSS Style Sheets
  - JavaScript files
Let’s Write a Web Browser!
An HTTP Request in Python

```python
import socket
mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('www.py4inf.com', 80))

mysock.send('GET http://www.py4inf.com/code/romeo.txt HTTP/1.0\n\n')

while True:
    data = mysock.recv(512)
    if (len(data) < 1):
        break
    print data
mysock.close()
```
HTTP/1.1 200 OK
Date: Sun, 14 Mar 2010 23:52:41 GMT
Server: Apache
Last-Modified: Tue, 29 Dec 2009 01:31:22 GMT
ETag: "143c1b33-a7-4b395bea"
Accept-Ranges: bytes
Content-Length: 167
Connection: close
Content-Type: text/plain

But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief

while True:
    data = mysock.recv(512)
    if ( len(data) < 1 ) : break
    print data

HTTP Header

HTTP Body
Making HTTP Easier With urllib
Using `urllib` in Python

Since HTTP is so common, we have a library that does all the socket work for us and makes web pages look like a file

```python
import urllib
fhand = urllib.urlopen('http://www.py4inf.com/code/romeo.txt')

for line in fhand:
    print line.strip()
```

[http://docs.python.org/library/urllib.html](http://docs.python.org/library/urllib.html)
import urllib
fhand = urllib.urlopen('http://www.py4inf.com/code/romeo.txt')
for line in fhand:
    print line.strip()

But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief

http://docs.python.org/library/urllib.html
import urllib
fhand = urllib.urlopen('http://www.py4inf.com/code/romeo.txt')

counts = dict()
for line in fhand:
    words = line.split()
    for word in words:
        counts[word] = counts.get(word, 0) + 1
print counts
import urllib
fhand = urllib.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print line.strip()

<h1>The First Page</h1>
<p>
</p>
import urllib
fhand = urllib.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print line.strip()

<h1>The First Page</h1>
<p>If you like, you can switch to the <a href="http://www.dr-chuck.com/page2.htm">Second Page</a>.</p>
Parsing HTML
(a.k.a. Web Scraping)
What is Web Scraping?

• When a program or script pretends to be a browser and retrieves web pages, looks at those web pages, extracts information, and then looks at more web pages.

• Search engines scrape web pages - we call this “spidering the web” or “web crawling”

http://en.wikipedia.org/wiki/Web_scraping
http://en.wikipedia.org/wiki/Web_crawler
import urllib
fhand = urllib.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print line.strip()
Why Scrape?

• Pull data - particularly social data - who links to who?

• Get your own data back out of some system that has no “export capability”

• Monitor a site for new information

• Spider the web to make a database for a search engine
Scraping Web Pages

• There is some controversy about web page scraping and some sites are a bit snippy about it.

• Google: facebook scraping block

• Republishing copyrighted information is not allowed

• Violating terms of service is not allowed
User Conduct

You understand that except for advertising programs offered by us on the Site (e.g., Facebook Flyers, Facebook Marketplace), the Service and the Site are available for your personal, non-commercial use only. You represent, warrant and agree that no materials of any kind submitted through your account or otherwise posted, transmitted, or shared by you on or through the Service will violate or infringe upon the rights of any third party, including copyright, trademark, privacy, publicity or other personal or proprietary rights; or contain libelous, defamatory or otherwise unlawful material.

In addition, you agree not to use the Service or the Site to:

- harvest or collect email addresses or other contact information of other users from the Service or the Site by electronic or other means for the purposes of sending unsolicited emails or other unsolicited communications;
- use the Service or the Site in any unlawful manner or in any other manner that could damage, disable, overburden or impair the Site;
- use automated scripts to collect information from or otherwise interact with the Service or the Site;
The Easy Way - Beautiful Soup

• You could do string searches the hard way

• Or use the free software called BeautifulSoup from www.crummy.com

http://www.crummy.com/software/BeautifulSoup/
http://www.pythonlearn.com/code/BeautifulSoup.py

Place the BeautifulSoup.py file in the same folder as your Python code...

https://www.crummy.com/software/BeautifulSoup/
http://www.pythonlearn.com/code/BeautifulSoup.py
import urllib
from BeautifulSoup import *

url = raw_input('Enter - ')

html = urllib.urlopen(url).read()
soup = BeautifulSoup(html)

# Retrieve a list of the anchor tags
# Each tag is like a dictionary of HTML attributes

tags = soup('a')

for tag in tags:
    print tag.get('href', None)
html = urllib.urlopen(url).read()
soup = BeautifulSoup(html)

tags = soup('a')
for tag in tags:
    print tag.get('href', None)

gifts = [i for i in gifts if i not in used]

python urllinks.py
Enter - http://www.dr-chuck.com/page1.htm
http://www.dr-chuck.com/page2.htm
Summary

- The TCP/IP gives us pipes / sockets between applications
- We designed application protocols to make use of these pipes
- HyperText Transport Protocol (HTTP) is a simple yet powerful protocol
- Python has good support for sockets, HTTP, and HTML parsing