Internet Networking

Chate Patanothai

Objective

- To understand the concept of sockets
- To learn how to send and receive data through sockets
- To implement network clients and servers
- To communicate with web servers and server-side applications through Hypertext Transfer Protocol (HTTP)

Two Computers Communicating Across the Internet

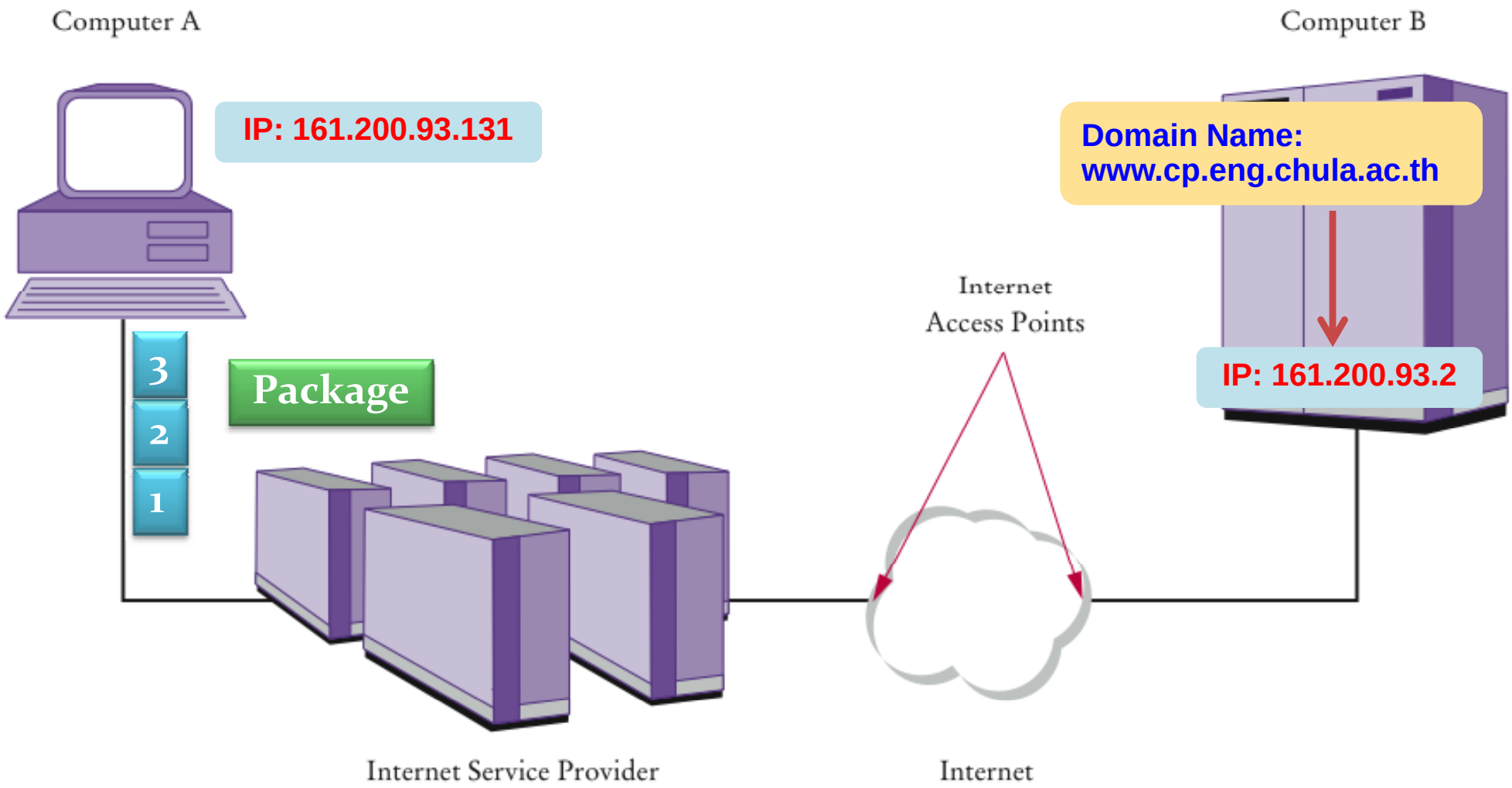

Figure 1 Two Computers Communicating Across the Internet

Transmission Control Protocol

- Internet Protocol (IP) does not notify the sender if data is lost or garbled
- This is the job of a higher level protocol *Transmission Control Protocol* (TCP)
- The most commonly used Internet services use TCP with IP (TCP/IP)

- TCP's job
  - Attempt to deliver the data
  - Try again if there are failures
  - Notify the sender whether or not the attempt was successful

Port Numbers

- When data are sent to a computer, they need to indicate which program is to receive the data
- IP uses *port numbers* for multiple services
  - *A port number is an integer between* **`0`** *and* **`65,535`**
  - *The sending program must know the port number of the receiving program*
  - *This number is included in the transmitted data*
- *Default port number*
  - *FTP – 20*
  - *Telnet – 23*
  - *SMTP – 25*
  - *HTTP – 80*
  - *HTTPS – 443*

Application Level Protocols

- TCP/IP mechanism establishes an Internet connection between two ports on two computers
- Each Internet application has its own *application protocol*
- This application protocol describes how data for that application are transmitted

Hypertext Transfer Protocol (HTTP)

- Application protocol used by the World Wide Web
- A web address is called a Uniform Resource Locator (URL)
- You type a URL into the address window of your browser
  - *For example,* `http://horstmann.com/index.html`

HTTP Commands

Command	Meaning
GET	Return the requested item
HEAD	Request only the header information of an item
OPTIONS	Request communications options of an item
POST	Supply input to a server-side command and return the result
PUT	Store an Item on the server
DELETE	Delete an item on the server
TRACE	Trace server communication

Client and Server Sockets

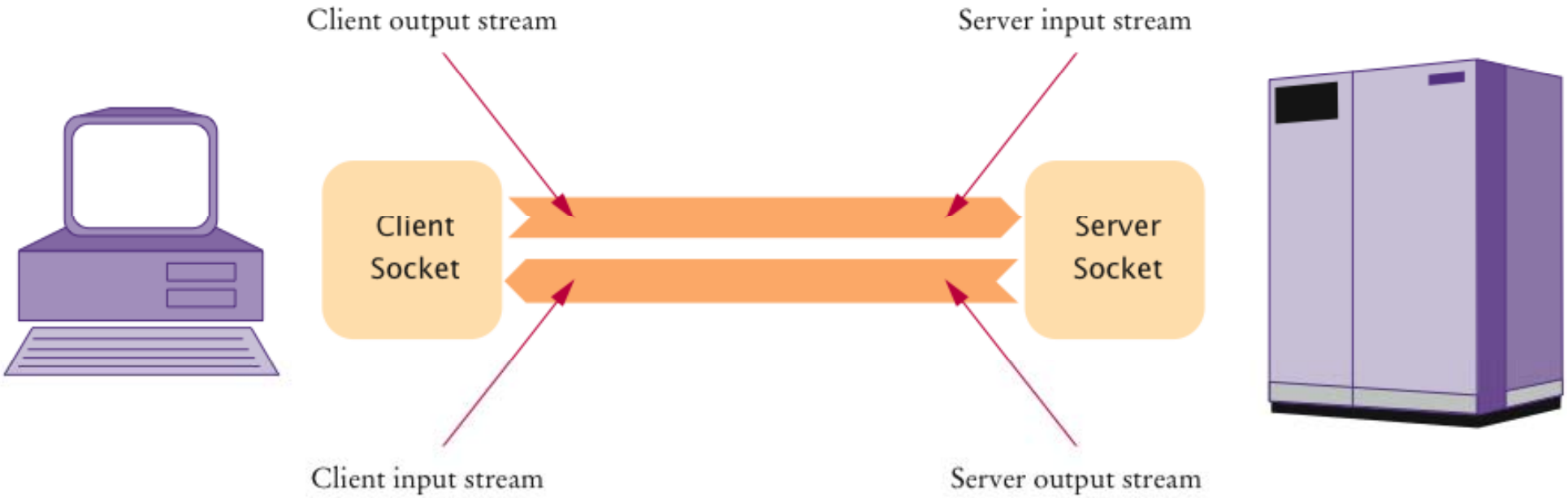

Figure 4 Client and Server Sockets

A Client Program – Sockets

- A <u>socket</u> is an object that encapsulates a TCP/IP connection
- There is a *socket* on both ends of a connection
- Create a socket in a Java program:

```
Socket s = new Socket(hostname, portnumber);
```

- Connect to the HTTP port of server, `horstmann.com`:

```
final int HTTP_PORT = 80;
Socket s = new Socket("horstmann.com", HTTP_PORT);
```

- If it can't find the host, the `Socket` constructor throws an `UnknownHostException`

A Client Program – Input

- Obtain the input streams and scanner:

```
InputStream instream = s.getInputStream();
Scanner in = new Scanner(instream)
```

- The socket catches the server's response and you can read it through `instream`

```
String input = in.nextLine()
```

- When finish communicating with the server, close the socket:

```
s.close();
```

A Client Program – Output

- Obtain the output streams and PrintWriter:

```
OutputStream outstream = s.getOutputStream();
PrintWriter out = new PrintWriter(outStream);
```

- When you send data to **`outstream`**, the socket forwards them to the server
- A **`PrintWriter`** buffers the characters and only sends when the buffer is full
- When sending a command:

```
out.print(command);
out.flush();
```

WebGet.java

```
1   import java.io.InputStream;
2   import java.io.IOException;
3   import java.io.OutputStream;
4   import java.io.PrintWriter;
5   import java.net.Socket;
6   import java.util.Scanner;
7
8   /**
9       This program demonstrates how to use a socket to communicate
10      with a web server. Supply the name of the host and the
11      resource on the command-line, for example
12      java WebGet horstmann.com index.html
13  */
14  public class WebGet
15  {
16      public static void main(String[] args) throws IOException
17      {
18          // Get command-line arguments
19
20          String host;
21          String resource;
22
```

Continued

WebGet.java (cont.)

```
23          if (args.length == 2)
24          {
25             host = args[0];
26             resource = args[1];
27          }
28          else
29          {
30             System.out.println("Getting / from horstmann.com");
31             host = "horstmann.com";
32             resource = "/";
33          }
34
35          // Open socket
36
37          final int HTTP_PORT = 80;
38          Socket s = new Socket(host, HTTP_PORT);
39
40          // Get streams
41
42          InputStream instream = s.getInputStream();
43          OutputStream outstream = s.getOutputStream();
44
```

Continued

WebGet.java (cont.)

```
        // Turn streams into scanners and writers

        Scanner in = new Scanner(instream);
        PrintWriter out = new PrintWriter(outstream);

        // Send command

        String command = "GET " + resource + " HTTP/1.1\n"
           + "Host: " + host + "\n\n";
        out.print(command);
        out.flush();

        // Read server response

        while (in.hasNextLine())
        {
           String input = in.nextLine();
           System.out.println(input);
        }

        // Always close the socket at the end

        s.close();
     }
  }
```

Continued

WebGet.java (cont.)

Program Run:

```
Getting / from horstmann.com
HTTP/1.1 200 OK
Date: Thu, 17 Sep 2009 14:15:04 GMT
Server: Apache/1.3.41 (Unix) Sun-ONE-ASP/4.0.2
. . .
Content-Length: 6654
Content-Type: text/html
<html>
<head><title>Cay Horstmann's Home Page</title></head>
<body>
<h1>Welcome to Cay Horstmann's Home Page</h1>
. . .
</body>
</html>
```

A Server Program

- Sample server program: enables clients to manage bank accounts in a bank
- When you develop a server application, you need some application-level protocol
- The client can use this protocol to interact with the server
- A simple bank access protocol is described on the next slide

Using the Telnet Program to Connect to the BankServer

```
Terminal
File  Edit  View  Terminal  Go  Help
~$ telnet localhost 8888
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
DEPOSIT 3 1000
3 1000.0
WITHDRAW 3 500
3 500.0
QUIT
Connection closed by foreign host.
~$
```

Figure 5 Using the Telnet Program to Connect to the Bank Server

URL Connections

- `URLConnection` class
  - *Provides convenient support for HTTP*
  - *Can also handle FTP (file transfer protocol)*
  - *Takes care of socket connection for you*
  - *Makes it easy to communicate with a web server without giving HTTP commands*

URL Connections

- Construct a `URL` object from a URL starting with the http or ftp prefix:

```
URL u = new URL("http://horstmann.com/index.html");
```

- Use the `URL`'s `openConnection` method to get the `URLConnection`:

```
URLConnection connection = u.openConnection();
```

- Call the `getInputStream` method to obtain an input stream:

```
InputStream instream = connection.getInputStream();
```

- You can turn the stream into a scanner in the usual way

HTTP Commands

```
command
request properties
blank line
```

- *HTTP command*
  - *Such as* `GET item HTTP/1.0`
- *request properties*
  - *Such as* `If-Modified-Since: date`
- *blank line*
  - *Separates the command and its request properties from the input data*

URLConnection Class

- Has methods to set request properties:

```
connection.setIfModifiedSince(date);
```

- Set the request properties before calling `getInputStream`
- The `URLConnection` class sends all the request properties that are set to the web server

Server Response

- Server response:

```
status line containing response code
response parameters
blank line
```

- For example:

```
HTTP/1.1 200 OK
Date: Tue, 24 Aug 2010 00:15:48 GMT
Server: Apache/1.3.3 (Unix)
Last-Modified: Sat, 26 Jun 2010 20:53:38 GMT
Content-Length: 4813
Content-Type: text/html
blank line
requested data
```

Retrieving Response Code and Message

- Cast the `URLConnection` object to the `HttpURLConnection` subclass
- Get the response code with `getResponseCode`
- Get the response message with `getResponseMessage`:

```
HttpURLConnection httpConnection = (HttpURLConnection)
   connection;
int code = httpConnection.getResponseCode(); // e.g., 404
String message = httpConnection.getResponseMessage(); //
   e.g., "Not found"
```

Retrieve Other Response Information from `URLConnection`

- Content length:

```
int length = connection.getContentLength();
```

- Content type:

```
String type = connection.getContentType();
```

URLGet.java

```
1   import java.io.InputStream;
2   import java.io.IOException;
3   import java.io.OutputStream;
4   import java.io.PrintWriter;
5   import java.net.HttpURLConnection;
6   import java.net.URL;
7   import java.net.URLConnection;
8   import java.util.Scanner;
9
10  /**
11      This program demonstrates how to use an URL connection
12      to communicate with a web server. Supply the URL on the
13      command-line, for example
14        java URLGet http://horstmann.com/index.html
15  */
16  public class URLGet
17  {
18      public static void main(String[] args) throws IOException
19      {
```

Continued

URLGet.java (cont.)

```
20          // Get command-line arguments
21
22          String urlString;
23          if (args.length == 1)
24             urlString = args[0];
25          else
26          {
27             urlString = "http://horstmann.com/";
28             System.out.println("Using " + urlString);
29          }
30
31          // Open connection
32
33          URL u = new URL(urlString);
34          URLConnection connection = u.openConnection();
35
```

Continued

URLGet.java (cont.)

```
36          // Check if response code is HTTP_OK (200)
37
38          HttpURLConnection httpConnection
39                = (HttpURLConnection) connection;
40          int code = httpConnection.getResponseCode();
41          String message = httpConnection.getResponseMessage();
42          System.out.println(code + " " + message);
43          if (code != HttpURLConnection.HTTP_OK)
44             return;
45
46          // Read server response
47
48          InputStream instream = connection.getInputStream();
49          Scanner in = new Scanner(instream);
50
51          while (in.hasNextLine())
52          {
53             String input = in.nextLine();
54             System.out.println(input);
55          }
56       }
57    }
```

Continued

URLGet.java (cont.)

Program Run:

```
Using http://horstmann.com/
200 OK
<html>
<head><title>Cay Horstmann's Home Page</title></head>
<body>
<h1>Welcome to Cay Horstmann's Home Page</h1>
…
</body>
</html>
```