HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÀI GIẢNG MÔN

Lập trình mạng

Giảng viên: TS. Nguyễn Trọng Khánh

Điện thoại/E-mail: khanhnt82@gmail.com

Bộ môn: CNPM- Khoa CNTT1

Học kỳ/Năm biên soạn: 2021

HTTP – SMTP – FTP

Nội dung

- Giao thức HTTP
- Giao thức SMTP
- Giao thức FTP

Giao thức HTTP

HTTP Server - Java Socket

Step 1: Mở ServerSocket tại cổng 8080
Step 2: Chấp nhận kết nối, đọc yêu cầu
Step 3: Ghi ra luồng theo giao thức HTTP
Step 4: Đóng luồng
Step 5: Đóng socket

HTTP Client - Java Socket

Step 1: Mở Socket
Step 2: Mở luồng vào ra cho socket
Step 3: Đọc và ghi vào luồng theo giao thức của server
Step 4: Đóng luồng
Step 5: Đóng socket

Tại bước 3: ghi các yêu cầu HTTP ra luồng

HTTP – URL

- Tạo đối tượng URL, truyền địa chỉ url.
- Sử dụng đối tượng URL để tạo đối tượng URLConnection.
- Đọc từ URLConnection sử dụng lớp InputStreamReader và BufferedReader.
- Sử dụng phương thức readLine của BufferedReader.

HTTP: đọc nội dung web

```
URL page = new URL(address);
StringBuffer text = new StringBuffer();
HttpURLConnection conn = page.openConnection();
conn.connect();

InputStreamReader in = new
    InputStreamReader((InputStream) conn.getContent());

BufferedReader buff = new BufferedReader(in);
String line = buff.readLine();
while (line != null) {
    text.append(line + "\n");
    line = buff.readLine();
}
System.out.println(text.toString());
```

Giao thức SMTP

Một số thành phần trong mail

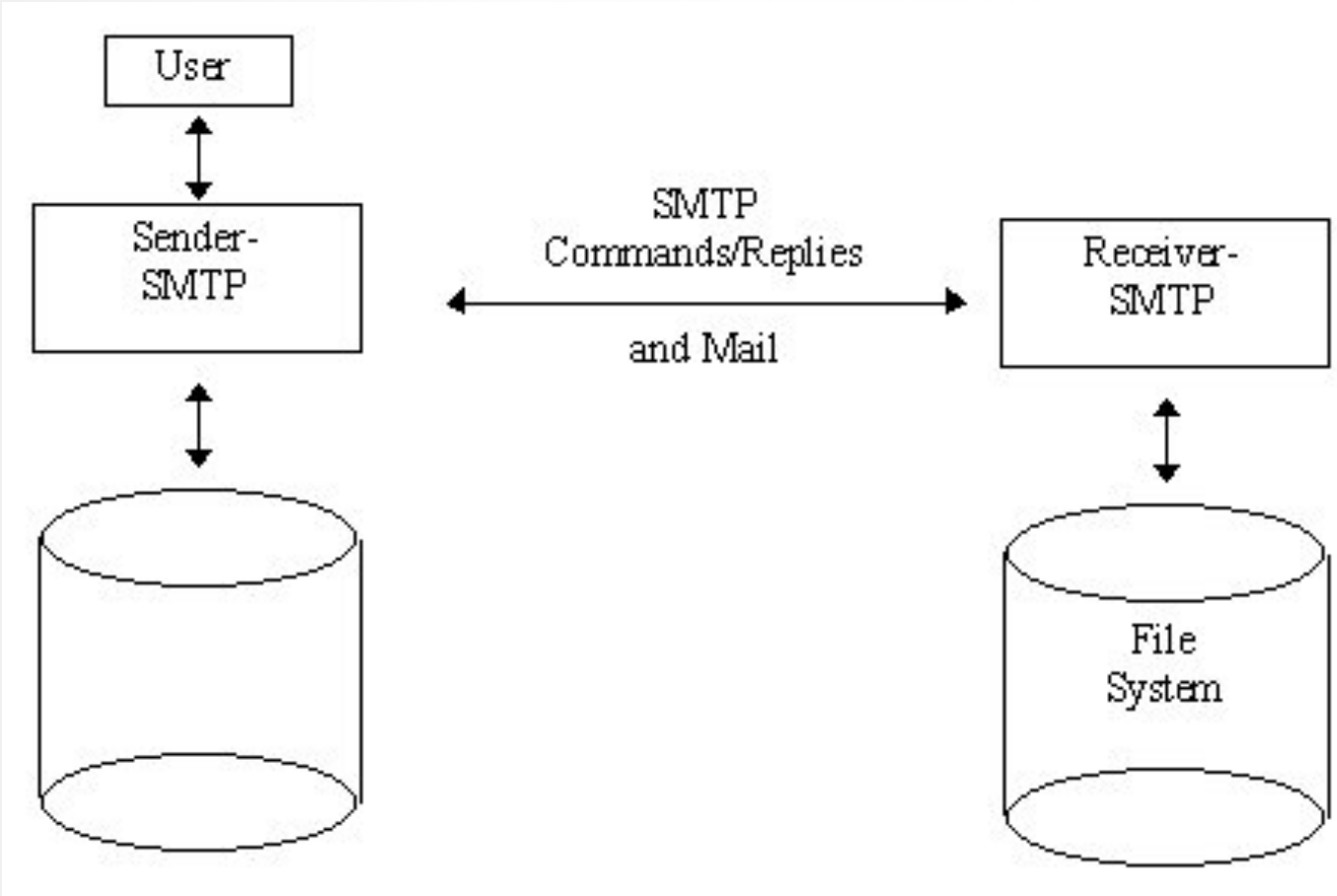
- Headers
  - Mô tả thông điệp
  - Mã ASCII
  - From, Reply-To, Date, To, CC, Subject
- Body
  - Các đoạn text mã NVT ASCII
- MIME (Multipurpose Internet Mail Extension): biểu diễn dữ liệu nhị phân (đính kèm): Text, Multipart, Message, Application, Image, Audio, Video
- Address: user@host.network

  mm6@andrew.cmu.edu

Các giao thức

- Giao thức chứa thông điệp: Đọc thông điệp từ Server
  - Internet Message Access Protocol (IMAP)
  - Post Office Protocol (POP)
- Giao thức vận chuyển thông điệp : Gửi thông điệp tới server (i.e. Simple Mail Transfer Protocol (SMTP))

SMTP



raddist.rad.com/networks/1998/smtp/smtp.htm

SMTP

- Chuyển mail từ host-tới-host qua TCP, cổng 25
- Gửi lệnh theo mã ASCII, kết thúc bằng dòng mới
- Vận chuyển không đồng bộ yêu cầu và trả lời giữa Sender-SMTP và Recipient-SMTP
- Bên nhận có thể máy đích hoặc trung gian, giựa trên máy chủ SMTP

SMTP

- Sender-SMTP thiết lập kênh chuyển với receiver-SMTP
- Sender-SMTP chuyển một lệnh MAIL với định danh của người gửi sender
- Receiver-SMTP trả lời
- Nếu ok, Sender-SMTP chuyển một lệnh RCPT nhận biết một hoặc nhiều người nhận
- Receiver-SMTP trả lời cho mỗi người nhận
- Nếu ok, Sender-SMTP gửi dữ liệu, kết thúc bởi một ký tự đặc biệt
- SMTP-receiver trả lời

Ví dụ

S: MAIL FROM:Smith@Alpha.ARPA
R: 250 OK

Code 250 means everything is OK

S: RCPT TO:Jones @Beta.ARPA
R: 250 DOESN'T NEED TO SAY OK SINCE 3 DIGIT CODE IS KEY

S: RCPT TO:Green@Beta.ARPA
R: 550 No such user here

Couldn't send message to Green@Beta.ARPA

S: DATA
R: 354 Start mail input; end with <CRLF>.<CRLF>
S: Random content…
S: More randome content
S: <CRLF>.<CRLF>
R: 250 OK

www.freesoft.org/CIE/RFC/821/4.htm

Lớp java.util.Properties

- Thừa kế HashMap: cặp khóa và giá trị
- Chứa các tập thuộc tính lâu dài; sử dụng luồng
- Tất cả khóa và giá trị đều kiểu String
- Phương thức để thiết lập và lấy giá trị:
  - public Object setProperty(String key, String value)
  - public String getProperty(String key)

Các lớp để gửi mail

- **Address**
- **Message**: Biểu diễn thông điệp
- **Transport**: Lớp biểu diễn giao thức giao vận, cho phép gửi các thông điệp email
- **Session**: Cung cấp giao tiếp giữa client email và mạng; hỗ trợ tạo và truy cập các đối tượng Lưu và Vận chuyển

Gửi mail

- Step 1 – Xác định giao thức

  props.setProperty(”xxxx", value)

- Step2 – Tạo session để gửi mail

  Session session = Session.getInstance(props, new javax.mail.
  Authenticator()...);

- Step 3 – Tạo mail qua lớp Message

  Message mess = **new MimeMessage(sess);**

  mess.setFrom(**new InternetAddress("yourmail@gmail.com"));**

  ...

- Step 4 – Gửi mail

  Message mess = send.createMessage(sess);

  Transport.*send(mess);*

Đọc mail

- Step 1 – Xác định giao thức

  props.setProperty("mail.store.protocol", protocol)

  protocol: imaps hoặc pop

- Step2 – Tạo session để đọc mail

  Session session = Session.getInstance(props, null);

- Step 3 –Truy cập mail qua lớp Store

  Store store = session.getStore(); store.connect("imap.gmail.com", "yourEmailId@gmail.com", "password");

  Folder inbox = store.getFolder("INBOX");

  inbox.open(Folder.READ_ONLY);

- Step 4 – Đọc inbox

  Message msg = inbox.getMessage(1);

  Message msg = inbox.getMessage(inbox.getMessageCount());

MessageSend.java

```
public Session createSession() {

    final String username = "yourmail@gmail.com";
    final String password = "yourpass";

    Properties props = new Properties();
    props.put("mail.smtp.host", "smtp.gmail.com");
    props.put("mail.smtp.socketFactory.port", "465");
    props.put("mail.smtp.socketFactory.class",
            "javax.net.ssl.SSLSocketFactory");
    props.put("mail.smtp.auth", "true");
    props.put("mail.smtp.port", "465");

    Session session = Session.getDefaultInstance(props,
        new javax.mail.Authenticator() {
            protected PasswordAuthentication getPasswordAuthentication() {
                return new PasswordAuthentication(username,password);
            }
        });

    return session;

}
```

MessageSend.java

```
public Message createMessage(Session sess) throws MessagingException {
    Message mess = new MimeMessage(sess);
    mess.setFrom(new InternetAddress("yourmail@gmail.com"));
    mess.setRecipients(Message.RecipientType.TO,
            InternetAddress.parse("friendmail@gmail.com", false));
    mess.setSubject("Test");
    mess.setText("This is a test of JavaMail's functionality.");
    mess.setSentDate(new Date());
    return mess;
}
```

MessageSend.java

```
public static void main(String[] args) {
    MessageSend send = new MessageSend();
    Session sess = send.createSession();
    try {
        Message mess = send.createMessage(sess);
        Transport.send(mess);
    } catch(MessagingException e) {
        System.out.println("Messaging Exception:  "+e);
    }
}
```

ReadingMail.java

```
Properties props = new Properties();
props.setProperty("mail.store.protocol", "imaps");
try {
    Session session = Session.getInstance(props, null);
    Store store = session.getStore();
    store.connect("imap.gmail.com", "yourEmailId@gmail.com", "password");
    Folder inbox = store.getFolder("INBOX");
    inbox.open(Folder.READ_ONLY);
    Message msg = inbox.getMessage(inbox.getMessageCount());
    Address[] in = msg.getFrom();
    for (Address address : in) {
        System.out.println("FROM:" + address.toString());
    }
    Multipart mp = (Multipart) msg.getContent();
    BodyPart bp = mp.getBodyPart(0);
    System.out.println("SENT DATE:" + msg.getSentDate());
    System.out.println("SUBJECT:" + msg.getSubject());
    System.out.println("CONTENT:" + bp.getContent());
} catch (Exception mex) {
    mex.printStackTrace();
}
```

Giao thức FTP

Thư viện Apache Commons Net API

- Thư viện hỗ trợ xây dựng ứng dụng upload/download file
- Lớp FTPClient: storeXXX chuyển file từ local → server
  - boolean storeFile(String remote, InputStream local)
  - OutputStream storeFileStream(String remote)
  - boolean storeUniqueFile(InputStream local)
  - boolean storeUniqueFile(String remote, InputStream local)
  - OutputStream storeUniqueFileStream()
  - OutputStream storeUniqueFileStream(String remote)
- *Local active mode:* Mở cổng trên client để server kết nối → thường bị chặn bởi tường lửa
- *Local passive mode:* Mở công trên server để client kết nối → không bị chặn

Upload file

- Kết nối và login tới server.
- Thiết lập kết nối kiểu *local passive*.
- Thiết lập kiểu dữ liệu truyền là nhị phân.
- Tạo InputStream cho file local.
- Xây dựng đường dẫn file từ xa cho server.
- Gọi các phương thức storeXXX() để truyền, có 2 cách:
  - Dựa trên InputStream
  - Dựa trên OutputStream
- Đóng luồng.
- Gọi phương thức completePendingCommand() để hoàn thành công việc.
- Thoát và đóng kết nối.

FTPUploadDemo

```
FTPClient ftpClient = new FTPClient();
try {

    ftpClient.connect(server, port);
    ftpClient.login(user, pass);
    ftpClient.enterLocalPassiveMode();

    ftpClient.setFileType(FTP.BINARY_FILE_TYPE);

    // APPROACH #1: uploads first file using an InputStream
    File firstLocalFile = new File("D:/Test/Projects.zip");

    String firstRemoteFile = "Projects.zip";
    InputStream inputStream = new FileInputStream(firstLocalFile);

    System.out.println("Start uploading first file");
    boolean done = ftpClient.storeFile(firstRemoteFile, inputStream);
    inputStream.close();
    if (done) {
        System.out.println("The first file is uploaded successfully.");
    }
```

FTPUploadDemo

```
// APPROACH #2: uploads second file using an OutputStream
File secondLocalFile = new File("E:/Test/Report.doc");
String secondRemoteFile = "test/Report.doc";
inputStream = new FileInputStream(secondLocalFile);

System.out.println("Start uploading second file");
OutputStream outputStream = ftpClient.storeFileStream(secondRemoteFile);
byte[] bytesIn = new byte[4096];
int read = 0;

while ((read = inputStream.read(bytesIn)) != -1) {
    outputStream.write(bytesIn, 0, read);
}
```

FTPUploadDemo

```
        inputStream.close();
        outputStream.close();

        boolean completed = ftpClient.completePendingCommand();
        if (completed) {
            System.out.println("The second file is uploaded successfully.");
        }

    } catch (IOException ex) {
        System.out.println("Error: " + ex.getMessage());
        ex.printStackTrace();
    } finally {
        try {
            if (ftpClient.isConnected()) {
                ftpClient.logout();
                ftpClient.disconnect();
            }
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
```

Download file

- Kết nối và login tới server.
- Thiết lập kết nối kiểu *local passive*.
- Thiết lập kiểu dữ liệu truyền là nhị phân.
- Thiết lập đường dẫn file từ xa
- Tạo OutputStream để ghi file local.
- Nhận file
  - retrieveFile() :
  - retrieveFileStream()
- Đóng luồng.
- Thoát và đóng kết nối.

FTPDownloadDemo

```
FTPClient ftpClient = new FTPClient();
try {

    ftpClient.connect(server, port);
    ftpClient.login(user, pass);
    ftpClient.enterLocalPassiveMode();
    ftpClient.setFileType(FTP.BINARY_FILE_TYPE);

    // APPROACH #1: using retrieveFile(String, OutputStream)
    String remoteFile1 = "/test/video.mp4";
    File downloadFile1 = new File("D:/Downloads/video.mp4");
    OutputStream outputStream1 = new BufferedOutputStream(new FileOutputStream(downloadFile1));
    boolean success = ftpClient.retrieveFile(remoteFile1, outputStream1);
    outputStream1.close();

    if (success) {
        System.out.println("File #1 has been downloaded successfully.");
    }
```

FTPDownloadDemo

```
// APPROACH #2: using InputStream retrieveFileStream(String)
String remoteFile2 = "/test/song.mp3";
File downloadFile2 = new File("D:/Downloads/song.mp3");
OutputStream outputStream2 = new BufferedOutputStream(new FileOutputStream(downloadFile2));
InputStream inputStream = ftpClient.retrieveFileStream(remoteFile2);
byte[] bytesArray = new byte[4096];
int bytesRead = -1;
while ((bytesRead = inputStream.read(bytesArray)) != -1) {
    outputStream2.write(bytesArray, 0, bytesRead);
}

success = ftpClient.completePendingCommand();
if (success) {
    System.out.println("File #2 has been downloaded successfully.");
}
```

FTPDownloadDemo

```
        outputStream2.close();
        inputStream.close();

    } catch (IOException ex) {
        System.out.println("Error: " + ex.getMessage());
        ex.printStackTrace();
    } finally {
        try {
            if (ftpClient.isConnected()) {
                ftpClient.logout();
                ftpClient.disconnect();
            }
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
```

Bài tập

Cài đặt theo giao thức HTTP để đọc nội dung trang web yahoo.com

Bài tập

Cài đặt đầy đủ theo mô hình giao thức SMTP cho ví dụ trong bài

Bài tập

Cài đặt theo mô hình giao thức FTP cho bài toán:

- Download file từ server
- Code theo mô hình MVC
- Client nhập tên file download và tên thư mục cần lưu file từ người dùng

Questions?