

Lab 4:

Electronic Mail

1 Previous Work:

It is essential for the correct development of this lab to study SMTP and POP3 protocols. Thus, you will be able to correctly answer the questions and better understand the captured traffic.

You can read about SMTP and POP3 in:

- Chapter 2.4 of Kurose's book.
- SMTP:
<http://www.rfc-editor.org/info/rfc5321>
- POP3:
<http://www.rfc-editor.org/info/rfc1939>
- Wireshark User Guide:
<http://www.wireshark.org/docs/>
- Java API Specification:
<http://docs.oracle.com/javase/8/docs/api/index.html>
-

2 The lab-4 aims.

In Lab-4, we will review the SMTP and POP3 protocols by capturing with Wireshark, SMTP and POP3 sessions, and programming in Java simple TCP clients that connects to the SMTP and POP3 servers running on the server `serveis-rdc.redes.upv.es`.

At the end of the Lab-4, you should be able to:

- List the sequence of commands used by the SMTP and POP3 clients for sending (or receiving) an e-mail message and also describe the meaning of those commands.
- Explain the format of e-mail messages (including head-to-body separation).
- Write a basic TCP client program in Java that is able to perform the dialog with the SMTP server (or POP3) using the necessary commands (not including the analysis of possible server error codes).

3 First part. Capture of mail protocol sessions.

The first part of the Lab-4 will consist of capturing and analysing SMTP and POP3 protocol sessions using Wireshark. To do this, we will configure the Thunderbird User Agent (UA) to access the e-mail accounts defined on the server `serveis-rdc.redes.upv.es`.

First, we will run Thunderbird. If it is the first time we run it, the following window will appear. Select "Skip this ... "



Note: If it is not the first time we run Thunderbird, we can get the next window through the Tools menu > Account Settings > Account Operations > Add Email Account.



In the window of the figure above, we will enter the user email address, which will be `redesXX@redes.upv.es` (replacing XX with number of your workstation),

Electronic Mail

and the password that will be provided by the teacher. When it is done, a new window appears. We must configure in it: the options of the protocol for mailbox management (incoming mail) and the protocol for sending e-mail (outgoing mail), as it is shown in the following figure.

Configuración de cuenta de correo

Su nombre: Alumno de Redes Su nombre, tal y como se muestra a los demás

Dirección de correo: redes01@redes.upv.es

Contraseña: *****

☒ Recordar contraseña

Se ha encontrado la siguiente configuración sondeando el servidor suministrado

	Nombre del servidor	Puerto	SSL	Identificación
Entrante: POP3	serveis-rdc.redes.upv.es	110	Ninguno	Contraseña normal
Saliente: SMTP	serveis-rdc.redes.upv.es	25	Ninguno	Contraseña normal
Nombre de usuario: Entrante:	redes01			
		Saliente:		redes01

Obtener una nueva cuenta Config. avanzada Cancelar **Volver a probar** Hecho

By pressing the Re-test (“*Volver a probar*”) button we can verify that the configuration of the ports is correct.

Finally, a window will warn us about the problem of using unencrypted passwords in unsafe protocols.

Configuración de cuenta de correo

¡Advertencia!

Configuración de entrada: serveis-rdc.redes.upv.es no usa cifrado.
► Detalles técnicos

Configuración de salida: serveis-rdc.redes.upv.es no usa cifrado.
► Detalles técnicos

Thunderbird le puede permitir recuperar su correo usando la configuración proporcionada. Sin embargo, debería contactar con su administrador o proveedor de correo electrónico en relación a estas conexiones inadecuadas. Vea la FAQ de Thunderbird para más información.

☒ Entiendo los riesgos.

Cambiar configuración **Hecho**

To verify that the configuration is correct, we will capture an SMTP session using Wireshark.

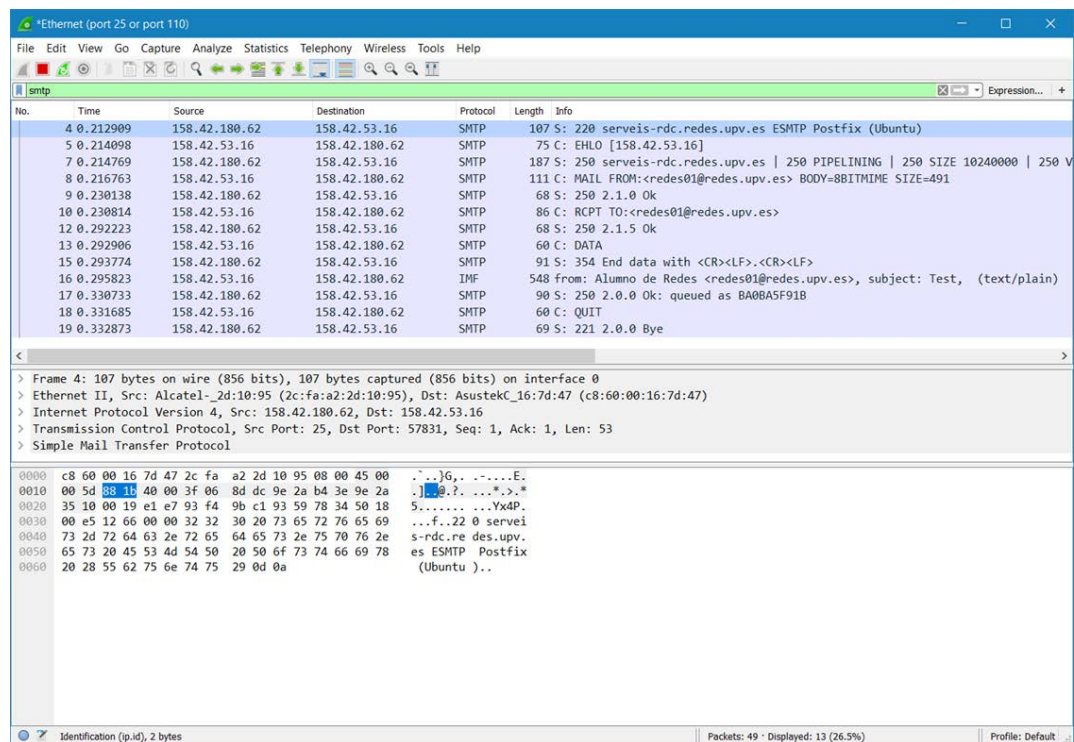
Exercise 1: SMTP Capture

Capture an SMTP session. You must launch the Wireshark program and set it to capture SMTP traffic (port 25 filter).

Using Thunderbird, compose an email destined to `redesXX@redes.upv.es` (replacing XX with the number of your workstation) and send it.

You can also filter information about the SMTP protocol using a display filter once the capture is done.

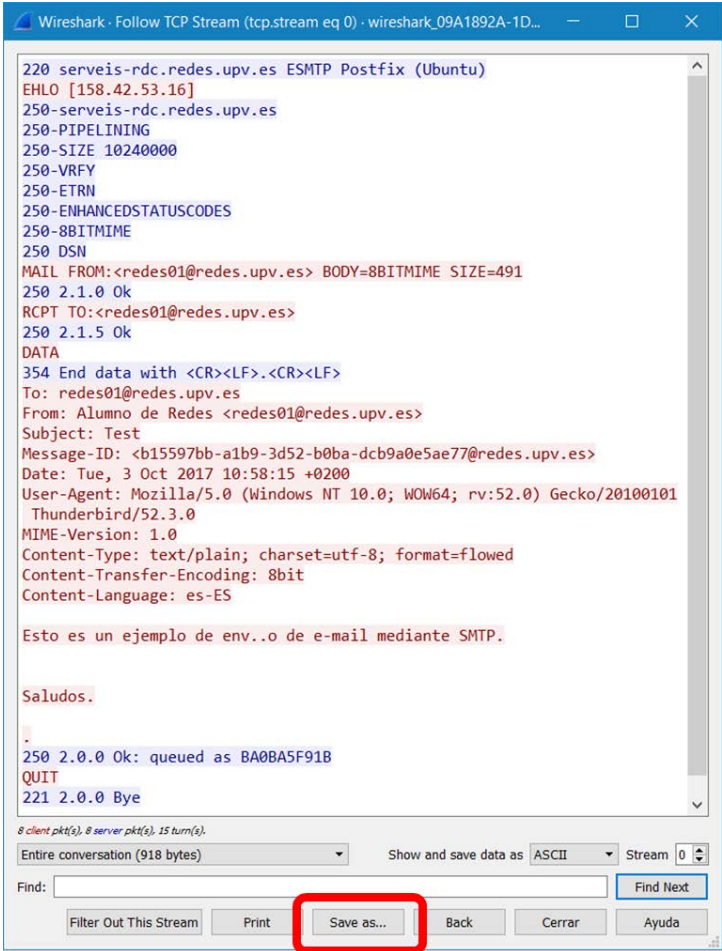
You should get something similar to the figure below:



Select the first SMTP message and apply the "Follow TCP Stream" analysis. Then save the dialog in a text file for later analysis.

The following image shows that dialog.

Electronic Mail



Wireshark - Follow TCP Stream (tcp.stream eq 0) - wireshark_09A1892A-1D...

```
220 serveis-rdc.redes.upv.es ESMTP Postfix (Ubuntu)
EHLO [158.42.53.16]
250-serveis-rdc.redes.upv.es
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
MAIL FROM:<redes01@redes.upv.es> BODY=8BITMIME SIZE=491
250 2.1.0 Ok
RCPT TO:<redes01@redes.upv.es>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
To: redes01@redes.upv.es
From: Alumno de Redes <redes01@redes.upv.es>
Subject: Test
Message-ID: <b15597bb-a1b9-3d52-b0ba-dcb9a0e5ae77@redes.upv.es>
Date: Tue, 3 Oct 2017 10:58:15 +0200
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:52.0) Gecko/20100101
Thunderbird/52.3.0
MIME-Version: 1.0
Content-Type: text/plain; charset=utf-8; format=flowed
Content-Transfer-Encoding: 8bit
Content-Language: es-ES

Esto es un ejemplo de env..o de e-mail mediante SMTP.

Saludos.

.
250 2.0.0 Ok: queued as BA0BA5F91B
QUIT
221 2.0.0 Bye
```

8 client pkt(s), 8 server pkt(s), 15 turn(s).

Entire conversation (918 bytes) Show and save data as ASCII Stream 0

Find: Find Next

Filter Out This Stream Print Save as... Back Cerrar Ayuda

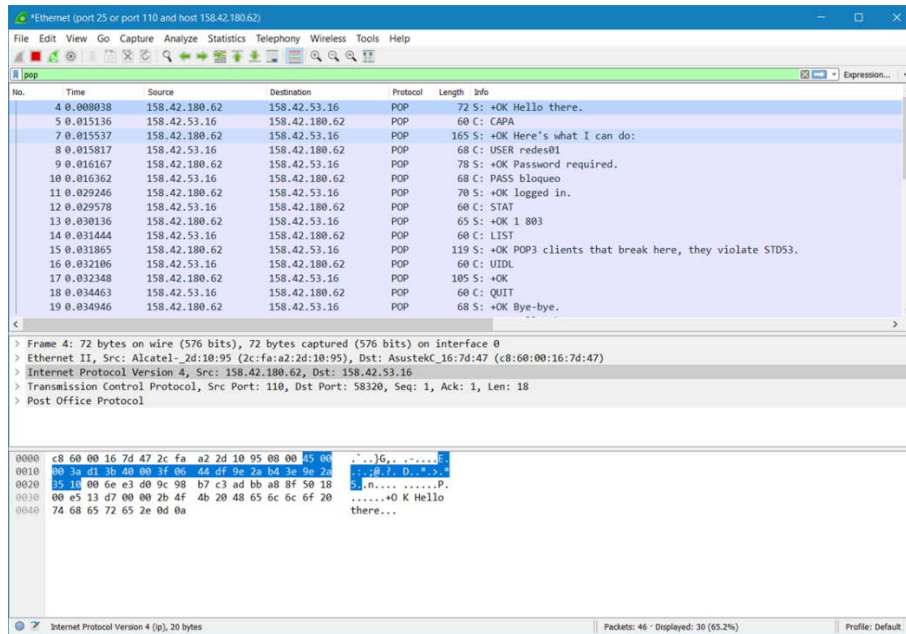
The dialog that you have saved will help you to develop a SMTP client later in Lab-4.

Exercise 2: POP3 Capture

Capture a POP3 session. You must launch the Wireshark and capture POP3 traffic (port 110 filter). In Thunderbird, compose an email destined to `redesXX@redes.upv.es` (replacing XX with the number of your workstation) and send it (up to this point, only SMTP traffic was generated). Then press the "Receive Messages" button.

You can also filter information about the POP3 protocol using a display filter once the capture is done.

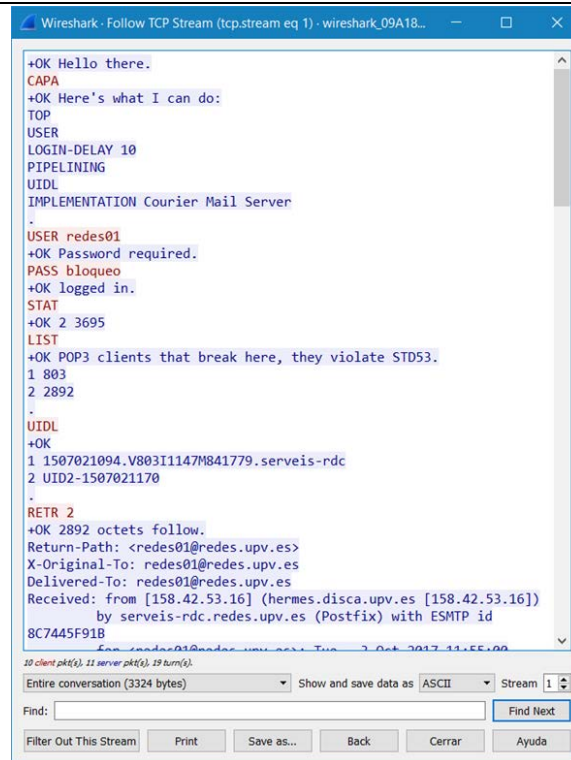
You should see something similar to the figure below:



As in the previous exercise, select the first POP3 message and apply the "Follow TCP Stream" analysis.

Save the dialog in a text file for further analysis.

Electronic Mail



The image shows a Wireshark window titled "Wireshark · Follow TCP Stream (tcp.stream eq 1) - wireshark_09A18...". The main pane displays a POP3 conversation between a client and a server. The text is as follows:

```
+OK Hello there.  
CAPA  
+OK Here's what I can do:  
TOP  
USER  
LOGIN-DELAY 10  
PIPELINING  
UIDL  
IMPLEMENTATION Courier Mail Server  
.  
USER redes01  
+OK Password required.  
PASS bloqueo  
+OK logged in.  
STAT  
+OK 2 3695  
LIST  
+OK POP3 clients that break here, they violate STD53.  
1 803  
2 2892  
.  
UIDL  
+OK  
1 1507021094.V803I1147M841779.serveis-rcd  
2 UID2-1507021170  
.  
RETR 2  
+OK 2892 octets follow.  
Return-Path: <redes01@redes.upv.es>  
X-Original-To: redes01@redes.upv.es  
Delivered-To: redes01@redes.upv.es  
Received: from [158.42.53.16] (hermes.disca.upv.es [158.42.53.16])  
by serveis-rcd.redes.upv.es (Postfix) with ESMTP id  
8C7445F91B  
from <redes01@redes.upv.es> Tue, 2 Oct 2017 11:55:00  
10 client pkt(s), 11 server pkt(s), 19 turn(s).  
Entire conversation (3324 bytes) Show and save data as ASCII Stream 1  
Find: Find Next  
Filter Out This Stream Print Save as... Back Cerrar Ayuda
```

The saved dialog will help you to develop a POP3 client later in Lab-4.

4 Second part. SMTP Client Program

Let's start creating a simple Java program that connects to the SMTP server running on `serveis-rcd.redes.upv.es`. It will send an email to the account `redesXX@redes.upv.es` (replacing XX with the number of your workstation). The e-mail you will send must contain at least the "From:", "To:" and "Subject:" headers. In the middle of message body, you have to include a line beginning with the dotted character (.) as text. In order to simplify the program, the e-mail addresses of the sender and receiver of the message will be fixed. Also, the message sent will be fixed. In a more elaborate version of the program these information (addresses and message text) can be read from the keyboard.

To facilitate the development of program, a first incomplete version of the SMTP client, called "ClientSMTP_incompleto.java", is provided. You can download it from *PoliformaT -> Recursos -> Practicas -> Cuatrimestre A -> Practica 4*.

Notice that the file name does not match the name of the class that it defines, so if you use the **javac** compiler you will need to modify one of the two names, to avoid compiler fails.

Exercise 3: SMTP Client Program.

Complete "ClientSMTP_incompleto.java" program to send an email to the address redeXX@redes.upv.es. As sender address you can use one of your email addresses.

The email must contain at least the "From:", "To:" and "Subject:" headers. In the middle of message body, you have to include a line beginning with the dotted character (.) as text. Check the proper functioning of the program by running. Modify the program and run it several times with different text in the "Subject:" header. You can also send an email to your own email address and verify that you receive it.

5 Third part. POP3 Client Program

Next, we will complete the code of a POP3 client. It will connect to the mail server and will perform various operations with the mailbox.

An incomplete first version of the POP3 client called "ClientePOP3_incompleto.java" can be downloaded from *PoliformaT* - >Recursos->Practicas->Cuatrimestre A->Practica 4. **Notice that the file name does not match the name of the class that it defines**, so if you use the **javac** compiler you will need to modify one of the two names, to avoid compiler fails.

Exercise4: POP3 Client Program

Complete "ClientePOP3_incompleto.java" program to read and delete the messages stored in the mailbox associated with the address redeXX@redes.upv.es that is located at serveis-rdc.redes.upv.es. Check the proper functioning of the program by running it. Read and delete the various messages that you sent in the previous exercise.

Question 1.1. What happened to the line of the message beginning with a dot?

Question 1.2. What do you think would have happened if you had sent a line containing a single dot in your data?

As you can notice, in the line beginning with a dot, the dot has disappeared. This is because SMTP client duplicate the dot when it find a line that begins with a dot, i.e. it adds an extra dot. It allows the user to send any type of text, including a line that only contains a dot. However, our SMTP client program did not implement this condition of the standard.

Electronic Mail

On the other hand, when the server finds a line that begins with a dot, it eliminates it, if the line includes more characters besides the end of line, it treats it like a normal line. Conversely, if the line only contained the dot and the <CRLF> characters, it detects the end of message.