Universidad Carlos III de Madrid

GRADO EN INGENIERÍA EN
TECNOLOGÍAS DE TELECOMUNICACIÓN
Aplicaciones Telemáticas
Evaluación continua (E2)
(90 minutes) 22-12-2021 - Page 1

Name/Surname:_____

Instrucciones

- You can use any reasonable assumption (used in class) but you must explicitly state that you do it and what that assumption is, by briefly explaining it.
- Explain everything you do since the results provided without an explanation may not count for the grade

Problem 1 - e-mail

A student with the email account `alumno1@gmail.com` uses his home PC with domain name `clientpc.origin.com`. On such a PC, which is not an MTA (simply a mail client), there is a mail User Agent that the student uses to send an email like the following:

```
From: alumno1@gmail.com
To: juan@uc3m.es
To: sat@lenovo.com
CC: alumno2@gmail.com
Subject: merry christmas
Content-Type: multipart/mixed; boundary="MixedBoundaryString"

--MixedBoundaryString

Hi there!

Let me wish you a Merry Christmas and A Happy New Year!

See you soon

--MixedBoundaryString
Content-Type: image/jpg;name="greetingscard.jpg"
Content-Transfer-Encoding: base64
```

```
ARBITRARY_INFORMATION_ENCODED_IN_BASE64

--MixedBoundaryString--
```

All the headers of the mail, boundaries of the multipart, headers of the parts of the multipart as well as the body of the mail, except the content in base64, occupy 467 octets. To clarify, in other words, the whole mail except the content in base64 (which would go right where it is written "ARBITRARY_INFORMATION_ENCODED_IN_BASE64") occupies 467 octets.

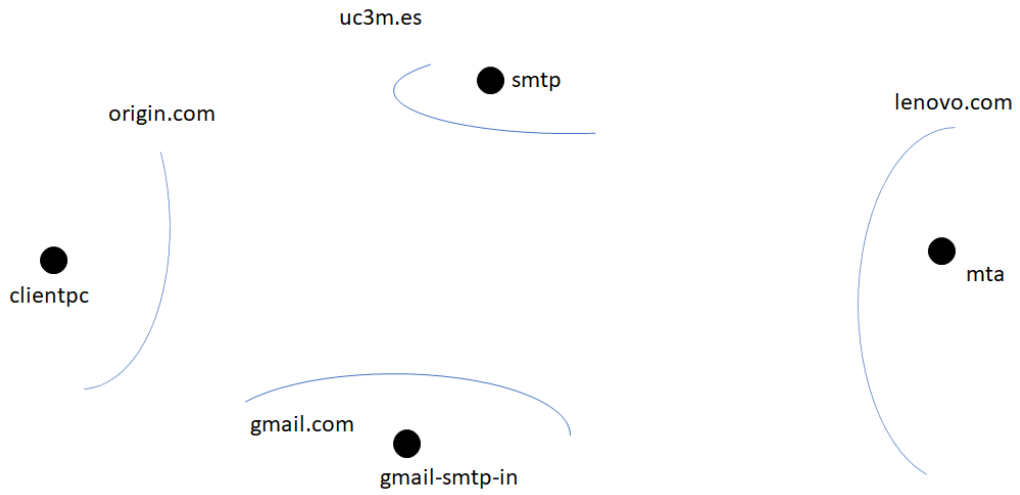

Figura 1: MTAs interconnection

The MTA servers described in Figure 1 can communicate freely with each other. The MTA of `gmail.com` is `gmail-smtp-in.gmail.com`, the input MTA of the domain `uc3m.es` is `smtp.uc3m.es` and the input MTA of the domain `lenovo.com` is `mta.lenovo.com`.

1. (0,5 pts) Indicate which header(s) are missing from the RFC822 format email described in the exam and **justify your reply**

   Solution:
   The email is a MIME email, as indicated by the header: `Content-Type: multipart/mixed; boundary="MixedBoundaryString"`.

   It is essential for such MIME emails to also include a header specifying that they are MIME, which is: `MIME-Version: 1.0`.

In the first part of the multipart message, there are no headers like `Content-Type` or `Content-Transfer-Encoding`, nor any other MIME-related headers, as seen in the second part. These headers are not mandatory when the text is standard ASCII, which it is in this case.

Headers like `Content-Length` cannot be included.

2. (2 pts) Write down the full SMTP trace from `clientPC`, where the UA is located, to the first MTA it connects to (`gmail-smtp-in.gmail.com`).

   Solution:

   The trace corresponds to a common scenario involving three recipients. It includes the greetings from both the server and the client, the `MAIL FROM` command, and three `RCPT TO` commands. The RFC822 email content (to replace `<email>`) must be terminated with a `.`.

```
S: 220 gmail-smtp-in.gmail.com ready ...
C: HELO clientpc.origin.com
S: 250 gmail-smtp-in.gmail.com
C: MAIL FROM: alumno1@gmail.com
S: 250 OK
C: RCPT TO: sat@lenovo.com
S: 250 OK
C: RCPT TO: juan@uc3m.es
S: 250 OK
C: RCPT TO: alumno2@gmail.com
S: 250 OK
C: DATA
S: 354
<email>
.
S: 250 OK
C: QUIT
S: 221
```

3. (2 pts) Indicate differences in connections, SMTP commands, as well as possible changes to the mail in RFC822 format transported by SMTP when the email leaves `gmail-smtp-in.gmail.com`

to the final destinations. Justify your answers by stating in detail where each change occurs and why..

Solution:

When the email reaches the Gmail SMTP relay (`gmail-smtp-in`), the delivery for recipients in the `gmail.com` domain is satisfied. However, there are still two recipients in different domains who need to receive the email.

For this reason, there will be two connections: - One between `gmail-smtp-in.gmail.com` and `smtp.uc3m.es` (a). - Another between `gmail-smtp-in.gmail.com` and `mta.lenovo.com` (b).

Changes in a)

In the greeting:

```
S: 220 smtp.uc3m.es
C: HELO gmail-smtp-in.gmail.com
```

For the recipient (SMTP), as there will be a single recipient:

```
C: RCPT TO: juan@uc3m.es
S: 250 OK
```

The email in RFC822 format will remain unchanged except for the trace headers that will be added:

```
Received: from gmail-smtp-in.gmail.com ([a.b.c.d]:PPPP)
   by smtp.uc3m.es ([X.y.z.q]:PPPP) with SMTP id ... ;
   DOW, DD MM YYYY HH:MM:SS -$$$$ (ZT)
```

Changes in b)

In the greeting:

```
S: 220 mta.lenovo.com
C: HELO gmail-smtp-in.gmail.com
```

For the recipient (SMTP), as there will be a single recipient:

```
C: RCPT TO: sat@lenovo.com
S: 250 OK
```

The email in RFC822 format will remain unchanged except for the trace headers that will be added:

```
Received: from gmail-smtp-in.gmail.com ([a.b.c.d]:PPPP)
   by mta.lenovo.com ([X.y.z.q]:PPPP) with SMTP id ... ;
   DOW, DD MM YYYY HH:MM:SS -$$$$ (ZT)
```

4. (2 pts) Calculate the total size of the mail with the dimensions specified at the beginning of the problem and considering that the file `greetingscard.jpg`, that is sent in the email, has a huge size of 68153 bytes[1]

Solution:

The problem specifies that the size of the email should include the given 467 bytes for the corresponding part of the email, along with the size of the file encoded in Base64.

Base64 encoding converts arbitrary binary data into text using a reduced set of characters constructed from 6 bits. It encodes any binary information by processing groups of 3 bytes (24 bits) and generating 4 characters from the Base64 code table, each corresponding to 6 bits. These characters are finally represented as 8-bit characters (32 bits). This results in a ratio of $32/24 = 4/3$.

Since 68153 is not divisible by 3, we need to add one padding byte, transforming the total calculation to $(68153 + 1) \times 4/3 = 90872$. Considering that the added padding must be announced by appending a "`=`" character at the end of the encoding, and adding the rest of the email size, the total becomes:

$$467 + 90872 + 1 = 91340 \text{ Bytes.}$$

Solution:

The problem specifies that the size of the email should include the given 467 bytes for the corresponding part of the email, along with the size of the file encoded in Base64.

Base64 encoding converts arbitrary binary data into text using a reduced set of characters constructed from 6 bits. It encodes any binary information by processing groups of 3 bytes (24 bits) and generating 4 characters from the Base64 code table, each corresponding to 6 bits. These characters are finally represented as 8-bit characters (32 bits). This results in a ratio of $32/24 = 4/3$.

[1] you can consider the size specified for the corresponding part of the mail of 467 bytes without prejudice to (or considering) question number 1

Since 68153 is not divisible by 3, we need to add one padding byte, transforming the total calculation to $(68153 + 1) \times 4/3 = 90872$. Considering that the added padding must be announced by appending a "=" character at the end of the encoding, and adding the rest of the email size, the total becomes:

$$467 + 90872 + 1 = 91340 \text{ Bytes.}$$

5. (1,5 pts) Consider that the finally encoded email is sent directly over TCP (without considering the commands of the SMTP protocol) by a connection with an MTU of 1500 and an announced window of 11680, in which there are no restrictions by speed, nor congestion. How many RTTs will take to send the mail from the beginning to the end of the connection?

   Solution:
   With an MSS of 1460 (MTU of 1500) and assuming no optional headers in IP or TCP, we calculate $91340/1460 = 62{,}56 \approx 63$ segments to send all the data.

   Since TCP uses a window mechanism with a value of $V_{ef} = \text{mín}(cwnd, WIN)$ and considering that $WIN = 11680/1460 = 8$ segments, we have $V_{ef} \leq 8$.

   However, the value $V_{ef} = 8$ is not reached immediately, as the window must grow to that point via the increase in *cwnd*. This growth stabilizes at or above WIN, assuming no congestion, after 3 RTTs from the establishment of the connection. Specifically, the window grows from 1 to 2, then from 2 to 4, and finally from 4 to 8 over the first three RTTs after the connection is established. During this time, 7 segments will have been sent, leaving $63 - 7 = 56$ segments still to be transmitted.

   These 56 segments will be sent at a rate of 8 per RTT, regardless of *cwnd* increasing further, because V_{ef} will be limited by WIN. The transmission of these segments will take $56/8 = 7$ RTTs.

   The total time required will be: $1{,}5 + 3 + 7 + 1{,}5 = 13$ RTTs (considering establishment and closure as 1.5 RTTs each).

6. (2 pts) Let suppose that when the student writes the email in his UA, the recipient `alumno2@gmail.com` is placed in BCC (or CCO in Spanish) and then delivers it. Indicate and **adequately justify** what **differences** would exist with:

   - The email RFC822 format written in the exam

- The trace at the out of the UA (compared to your response in question 2)
- The differences stated in question 3

Solution:
The email in RCF822 format will not include any reference to `alumno2@gmail.com` anywhere. There is no equivalent header to use, such as `CCO` or `BCC`, in this case.

The trace will formally remain the same as in section 2. The justification is that, despite `alumno2@gmail.com` not being present in the email, SMTP must still deliver it. The only difference is that the transported email will be different since it does not include `alumno2@gmail.com`, as explained at the beginning.

In the third case, there is no variation. There will again be two connections, and the number of `RCPT TO` commands will remain the same for both connections. The justification is that it is not necessary to send the email to `alumno@gmail.com`, as it remains within Gmail's infrastructure. The trace will formally remain the same, except that the transported email will be different since it does not include `alumno2@gmail.com`, as explained at the beginning.

Problem 2 - HTTP

The host named `server.webops.com` has an apache2 server installed to serve `solar.energia.com` and `eolica.energia.com`. Both domains are independent zones belonging to two different companies inside the big corporation `energia.com`.
The server IP address is `65.65.65.80`.

1. (2pts) What is the minimal change in the DNS of `eolica.energia.com` so that requests to `http://www.eolica.energia.com` are responded by the server? Write down the DNS requests and replies for a user to resolve the domain. Let all the caches be empty and indicate which replies are authoritative.

   Solution:
   Minimal changes: An alias needs to be configured from www.eolica.energia.com to server.webops.com. Attention!! This should point to the server's name, not its IP address. It could be done by pointing the name www.eolica.energia.com to the IP of server.webops.com, but this approach is less scalable since any change to the IP of server.webops.com would

require updating everything...

Better explained: The DNS requests and responses to resolve the IP would be:

1. REQ A www.eolica.energia.com @rootserver
   REPLY answer=0 AUTHORITY: ns of .com ADDITIONAL: the IPs of
   the ns of .com
2. REQ A www.eolica.energia.com @.com
   REPLY answer=0 AUTHORITY: ns of .energia.com ADDITIONAL: the IPs of
   the ns of energia.com which we call IP-ns-eolica
3. REQ A www.eolica.energia.com @IP-ns-eolica
   REPLY CNAME server.webops.com [authorized response - aa]

-- we obtained a CNAME, but the client wants an IP. The request must
be repeated now considering server.webops.com ---

4. REQ A server.webops.com @rootserver
   REPLY answer=0 AUTHORITY: ns of .com ADDITIONAL: the IPs of
   the ns of .com
5. REQ A server.webops.com @.com
   AUTHORITY: ns of .webops.com ADDITIONAL: the IPs of
   the ns of webops.com which we call IP-ns-webops
6. REQ A server.webops.com @IP-ns-webops
   REPLY server.webops.com IN A TTL 65.65.65.80 [authorized response - aa]

2. (2pts) Write down the HTTP GET requests using protocol version 1.1 to get the pages `http://www.solar.energia.com` and `www.eolica.energia.com` include all the compulsory request headers.

Solution:

```
GET / HTTP/1.1
Host: www.solar.energia.com

GET / HTTP/1.1
```

```
Host: www.eolica.energia.com
```

3. (2pts) A user browsing `http://www.solar.energia.com` notes the load of the images `http://energia.com/solar-diag.jpg` and `http://energia.com/alt-diag.jpg`. Knowing that the DNS server of `energia.com` is doing load balancing of that address using two different IPs, and knowing that the TTL of the A record of `energia.com` is 10 seconds. Calculate if each image will be loaded from a different IP or both will load from the same. Suppose the size of each image is 250 KBytes, MSS=1460, the link is 2Mbps and 30 ms propagation delay. Explain the calculation procedure followed.

   The question only makes sense if we consider HTTP/1.0 clients, because HTTP/1.1 or later clients reuse the same connection to download all objects until the connection's keepalive expires. Let's perform the calculation for 1.0 clients:

   Corrected calculation:

   ```
   Number of segments = 250000/1460 = 171
   tx = 1500*8/2*10^6 = 6*10^-3 = 6ms;
   Continuous sending window:
   Vec = (tx + 2tp)/tx = (6ms + 60ms)/6ms = 66/6 = 11 segments;
   Total time = RTT + RTT + RTT + RTT + RTT + RTT + ...
   cwnd =  1   2   4   8   11  11 ...
   ```

   When it reaches cwnd=11, it has transmitted 8+4+2+1=15 segments. It has taken 4 RTTs (from 1 to 2, from 2 to 4, from 4 to 8, and from 8 to 11). There are 171-15=156 segments left, which in RTTs is: 156/11=14.18 -¿15 RTTs. Thus, considering the connection setup, it needs 15+4+1.5=20.5 RTTs = 20.5*0.066=1.353 s.

   Therefore, there would not be enough time for the cache entry to expire, and it would resolve to the same address, as the DNS would not have enough time to return the second address for load balancing.

4. (2pts) Imagine now that the server is configured with one of the two following configuration options: a) a 15 seconds KeepAliveTimeout; or b) activating HTTP Pipelining. Recalculate the timing of question 3 if needed and justify if any of the configuration has some effect in the load of the images from the same or different IPs.

Solution:

If the server is configured to keep the connection open for more than 15 seconds, the connection will not be closed.

HTTP Pipelining in HTTP/1.1 refers to the process of making multiple requests over a single connection before receiving the response to the first request. This mechanism allows for efficient use of the connection by reducing the overhead of establishing multiple connections and potentially improving performance by sending requests back-to-back without waiting for each response.

5. (2pts) Explain if the server can do some optimization to lower the images transmission time. Indicate the HTTP headers and values that make that optimization possible.

Solución: Transfer-Encoding: compress Transfer-Encoding: deflate Transfer-Encoding: gzip