# WEB SISTEMAK
# 2020/2021 IKASTURTEA

## HTTP - HyperText Transfer Protocol

### Content Length vs Transfer Encoding
### Erantzunaren kodifikazioa

2021-02-12 M (2. zatia)

# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH ETA TRANSFER ENCODING

- Web zerbitzari batek itzulitako erantzun bat irakurri behar duen web bezero batek, mezuaren gorputza non amaitzen den jakin behar du.
  - HTTP protokoloak mezuaren gorputzaren luzeera adierazteko duen modu bat “Content-Length” goiburua da.
    - » OHARRA: HTTP erantzuna bidali aurretik, “Content-Length” goiburuak erantzunean sartuko den edukiaren tamaina ezagutzea eskatzen du.

- Erantzunean datu kopuru handi bat itzuli behar bada, ez dago datu horien tamaina kalkulatzerik eskaera guztiz prozesatzen denerarte; adibidez: demagun datu base bateko eskaera baten emaitzak web orri batean bistaratu nahi direla.
  - “Content-Length” goiburua erabiltzekotan, mezuaren gorputzean doazen datuen tamaina erantzuna bidali aurretik ezagutu beharra dago. Honek, alde batetik, datu baseko eskaeraren prozesamendua bukatzeari itxoitea suposatzen du. Bestetik, datuak gordetzeko buffer handi bat behar da.
  - Erantzuna bidali aurretik hau prozesatzeko beharrezkoa den denborak eta erantzunak saretik bidaiatzen ematen duen denborak sortutako atzerapenak erabiltzailearen esperientzian eragin negatiboa dauka.

- HTTP protokoloak erantzunaren edukia zatika bidaltzen joateko modua eskeintzen du: “Transfer-Encoding” goiburua “chunked” balioarekin.

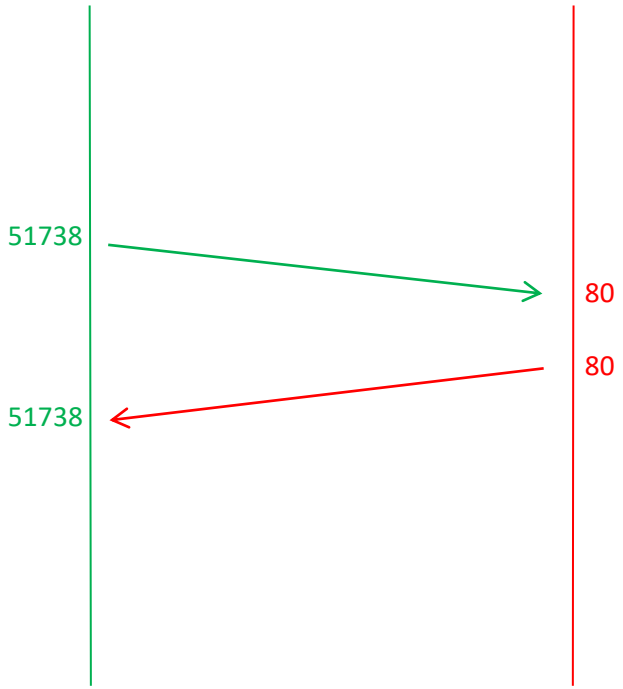
# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH ETA TRANSFER ENCODING

- “Content-Length” adibidea: zerbitzariak “Hello World!!” testua bidali behar duela suposatuko dugu.

**“Content-Length”-dun HTTP erantzuna**

```
HTTP/1.1 200 OK
Date: Thu, 20 Nov 2015 20:25:52 GMT
Last-Modified: Tue, 17 Sep 2015 13:00:02 GMT
ETag: "1a968-3ec-4e693e61bb8b6"
Content-Length: 13
Content-Type: text/plain

Hello World!!
```



# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH ETA TRANSFER ENCODING

- OHARRA: erantzunaren edukia TCP mailako MTU-a baino handiagoa bada, HTTP mezua hainbat TCP segmentutan bidaliko da.
- Adibidea: TCP mailaren MTU-a 187 zortzikote direla suposatuko dugu.

**"Content-Length"-dun HTTP erantzuna**

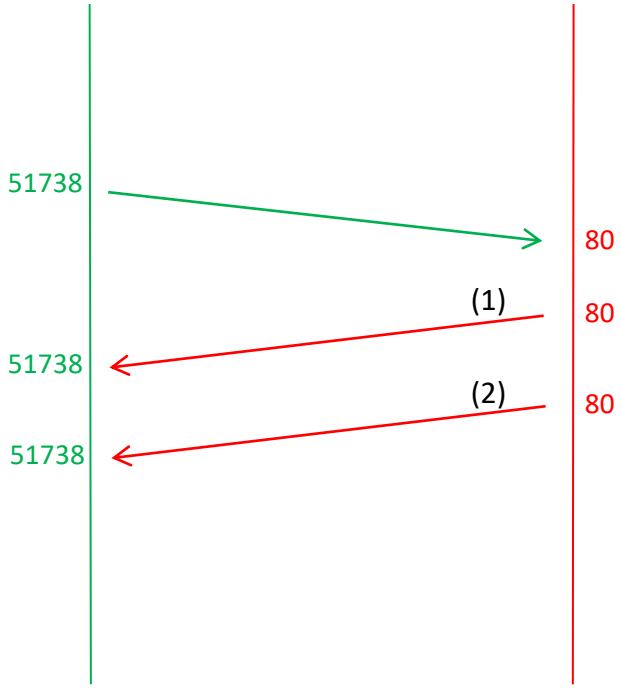
(1)

```
HTTP/1.1 200 OK
Date: Thu, 20 Nov 2015 20:25:52 GMT
Last-Modified: Tue, 17 Sep 2015 13:00:02 GMT
ETag: "1a968-3ec-4e693e61bb8b6"
Content-Length: 13
Content-Type: text/plain

Hello
```

(2)

```
World!!
```



# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH ETA TRANSFER ENCODING

- “Transfer-Encoding: chunked” adibidea: zerbitzariak “Hello World!!” testua bidali behar duela suposatuko dugu.

**“Transfer-Encoding”-dun HTTP erantzuna**

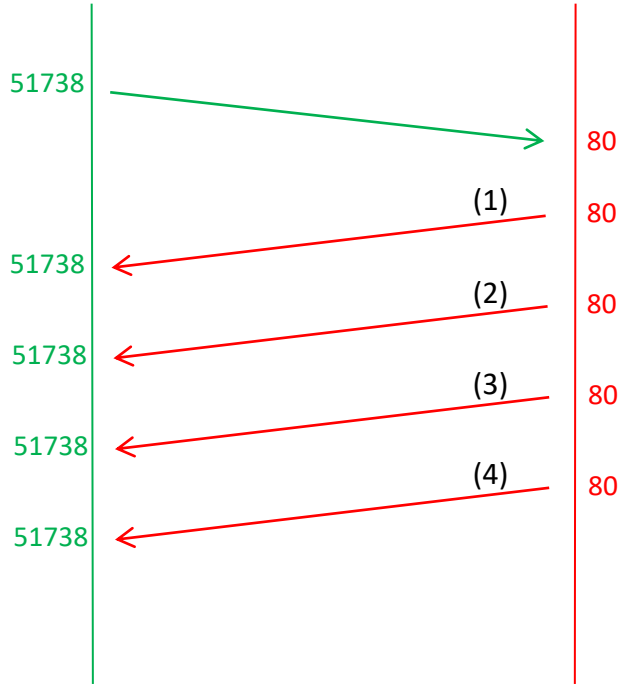
(1)
```
HTTP/1.1 200 OK
Date: Thu, 20 Nov 2015 20:25:52 GMT
Last-Modified: Tue, 17 Sep 2015 13:00:02 GMT
ETag: "1a968-3ec-4e693e61bb8b6“
Content-Type: text/plain
Transfer-Encoding: chunked

6\r\nHello \r\n
```

(2) `6\r\nWorld!\r\n`

(3) `1\r\n!\r\n`

(4) `0\r\n\r\n`




Zati bakoitzaren edukiaren luzeera notazio hamaseitarrean adierazten da.

2, 3 eta 4 zatiek ez daramate HTTP goibururik. Zuzenean TCP segmentuetan kapsulatuta doaz.

# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH ETA TRANSFER ENCODING

- Bi moduen arteko ezberdintasunak :
  - "Content-Length" goiburuarekin HTTP mezuak ez dira erantzunaren eduki osoa prozesatzen denerarte bidaltzen; aldiz, "Transfer-Encoding“ goiburuarekin HTTP mezuak eduki zatiak prozesatu ahala bidaltzen dira.
  - "Content-Length" goiburuarekin HTTP mezuen zatitzea TCP mailak egiten du; aldiz, "Transfer-Encoding" goiburuarekin HTTP mailak berak egiten du.

# ADIBIDEA:

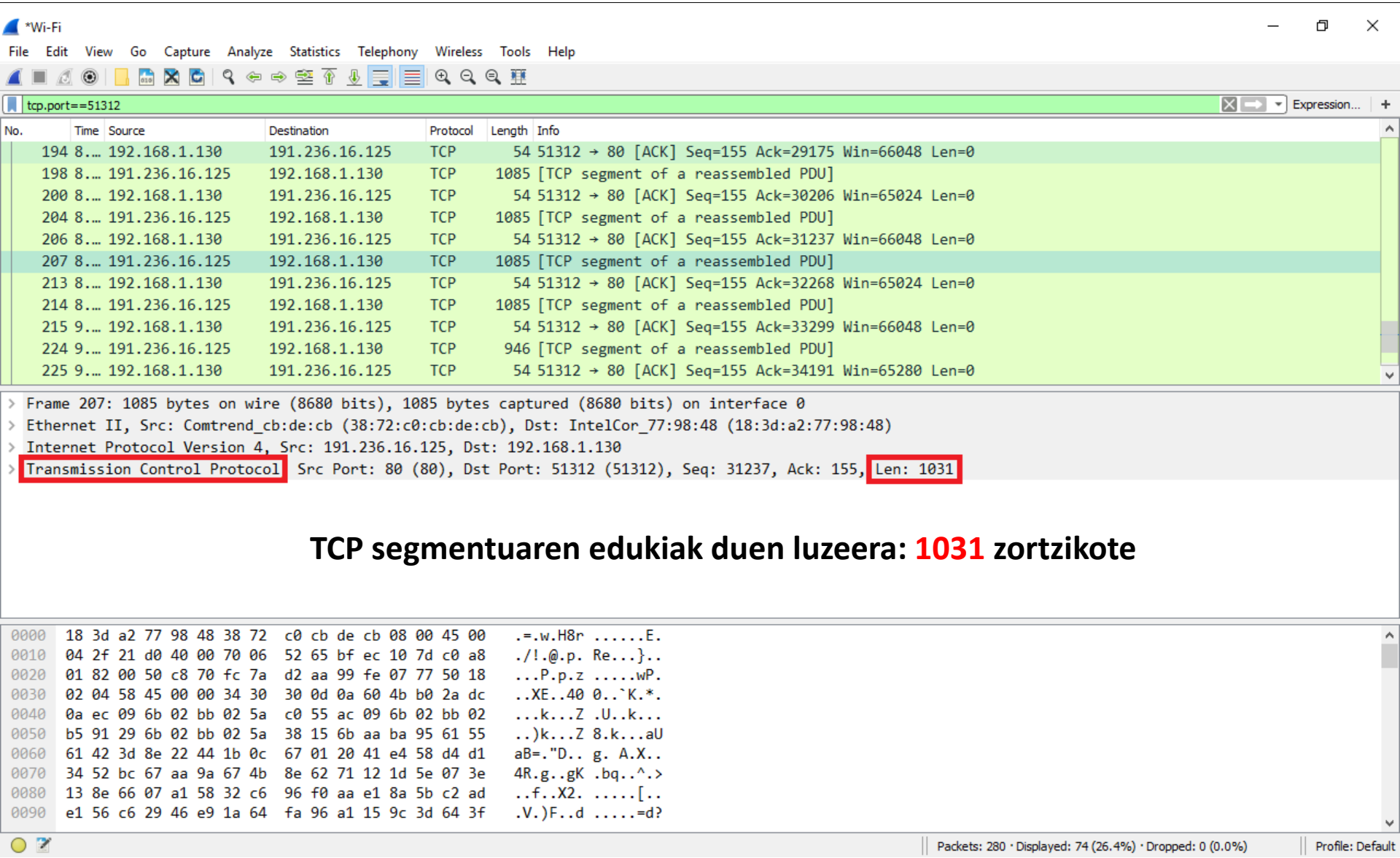
## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA

- Nabigatzaile batean ondorengo irudia deskargatu:
  http://www.httpwatch.com/httpgallery/chunked/chunkedimage.aspx

- Wireshark erabiliz, HTTP erantzunaren harrapaketa aztertu:
  - Zein da TCP segmentuaren datu eremuaren eta HTTP mezua zatiaren arteko tamainen erlazioa??
    - 3 + 2 + 1024 + 2 = 1031 → Aztertu zenbaki hauek nondik datozen

# ADIBIDEA:

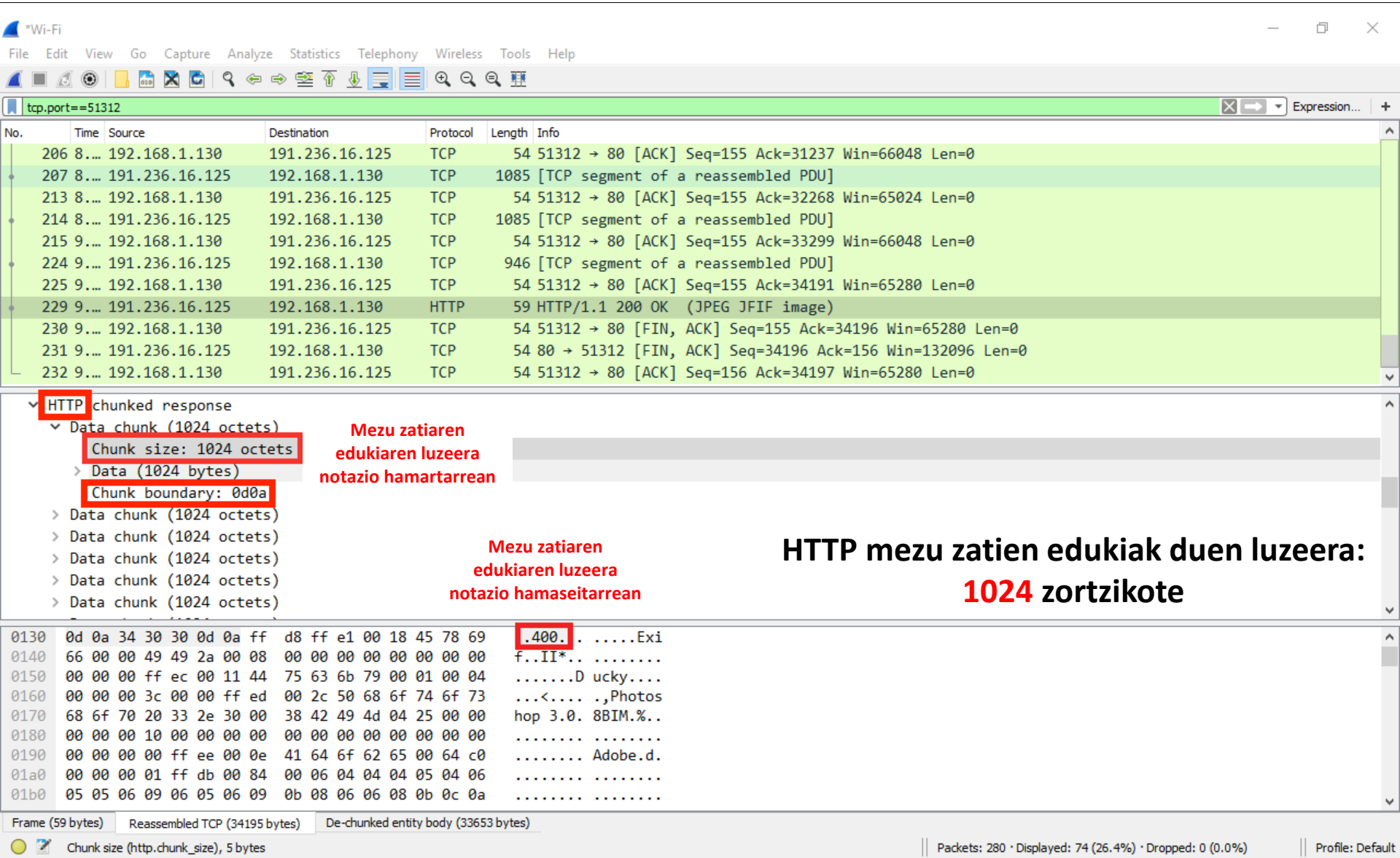
## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA




**TCP segmentuaren edukiak duen luzeera: 1031 zortzikote**

# ADIBIDEA:

## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA



# ADIBIDEA:

## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA

LABURPENA:

- Aurreko irudietan ikusten denez,
  - TCP segmentuaren edukiak duen luzeera: **1031** zortzikote
  - HTTP mezuaren eduki zatiak duten luzeera: **1024** zortzikote

- HTTP zati batek ondorengo eredua du:
  - ZATIAREN_EDUKIAREN_LUZEERA\r\nZATIAREN_EDUKIA\r\n

- HTTP zatia TCP segmentuaren eduki eremuan sartzen da, beraz:
  - len("1024") + len("\r\n") + 1024 + len("\r\n") = 4 + 2 + 1024 + 2 = **1032**

- 1031 ≠ 1032 → Zergaitik ez datoz bat?
  - Zatiaren luzeera notazio hamaseitarrean sartu behar da!!!
    - 1024 notazio hamartarrean = 400 notazio hamaseitarrean
    - len("400") + len("\r\n") + 1024 + len("\r\n") = 3 + 2 + 1024 + 2 = **1031**

# HTTP-REN FUNTZIONAMENDUA: MEZUAREN GORPUTZA

- Mezuaren gorputza zortzikotez (byte-z) osoturik dago:
  - Testua (testu sinplea, HTML, CSS, XML, JSON, CSS, …)
  - Eduki binarioa (PDF, bideoa, exekutagarriak, etc.)

- Eduki binarioak aplikazio edo plataforma jakin batzuentzako pentsaturik daude. Adibidez, PDF-ak PDF-irakurle batek soilik ulertu ditzake ondo. PDF bat testu editore arrunt batekin zabaldu edo irteera estandarrera bidaltzen bada, bere edukiaren adierazpen oker bat bistaratzen da, testu oinarria duena.

- Testua programa ugarirekin bistaratu daiteke, baina guztiek ez dute zortzikoteak dekodifikatzeko konfigurazio berdina erabiltzen.
  - Adibidez, jatorrian fitxategi bat UTF-8 erabiliz kodifikatu bazan, ”ñ” ikurra C3 B1 byte-ekin gordeko da.
  - Fitxategia ISO-8859-1-en konfiguratutako testu editore baten bitartez zabaltzen badugu, dekodifikazioaren eraginez ondorengo bi ikurrak bistaratuko dira: Ã±

# KODIFIKAZIOARI BURUZKO ARGIBIDE BATZUK...

| Kode taula | 4. byte | 3. byte | 2. byte | 1. byte |
|---|---|---|---|---|
| **US-ASCII (7 bit)** | | | | 0 |
| **ISO-8859-1 (1 zortzikote)** | | | | |
| **UTF-8 (1-4 zortzikote)** | | | | 0 |
| | | | 110 | 10 |
| | | 1110 | 10 | 10 |
| | 11110 | 10 | 10 | 10 |

Taula honetan kontutan hartu beharrekoa:

- latin-1 -ek eta UTF-8 -k US-ASCII kode taula barnean daramate.
- latin-1 -en, US-ASCII -rekiko gehitutako ikurrek “1” dute MSB-n.

# KODIFIKAZIOARI BURUZKO ARGIBIDE BATZUK...

| Kode taula | Ikurra | Hex | | binarioa |
|---|---|---|---|---|
| US-ASCII (7 bit) | H | -- | 48 | 0 1 0 0 1 0 0 0 |
| | ñ | -- | -- | |
| | á | -- | -- | |
| ISO-8869-1 (1 zortzikote) | H | -- | 48 | 0 1 0 0 1 0 0 0 |
| | ñ | -- | F1 | 1 1 1 1 0 0 0 1 |
| | á | -- | E1 | 1 1 1 0 0 0 0 1 |
| UTF 8 (1 - 4 zortzikote) | H | -- | 48 | 0 1 0 0 1 0 0 0 |
| | ñ | C3 | B1 | 1 1 0 0 0 0 1 1 1 0 1 1 0 0 0 1 |
| | á | C3 | A1 | 1 1 0 0 0 0 1 1 1 0 1 0 0 0 0 1 |

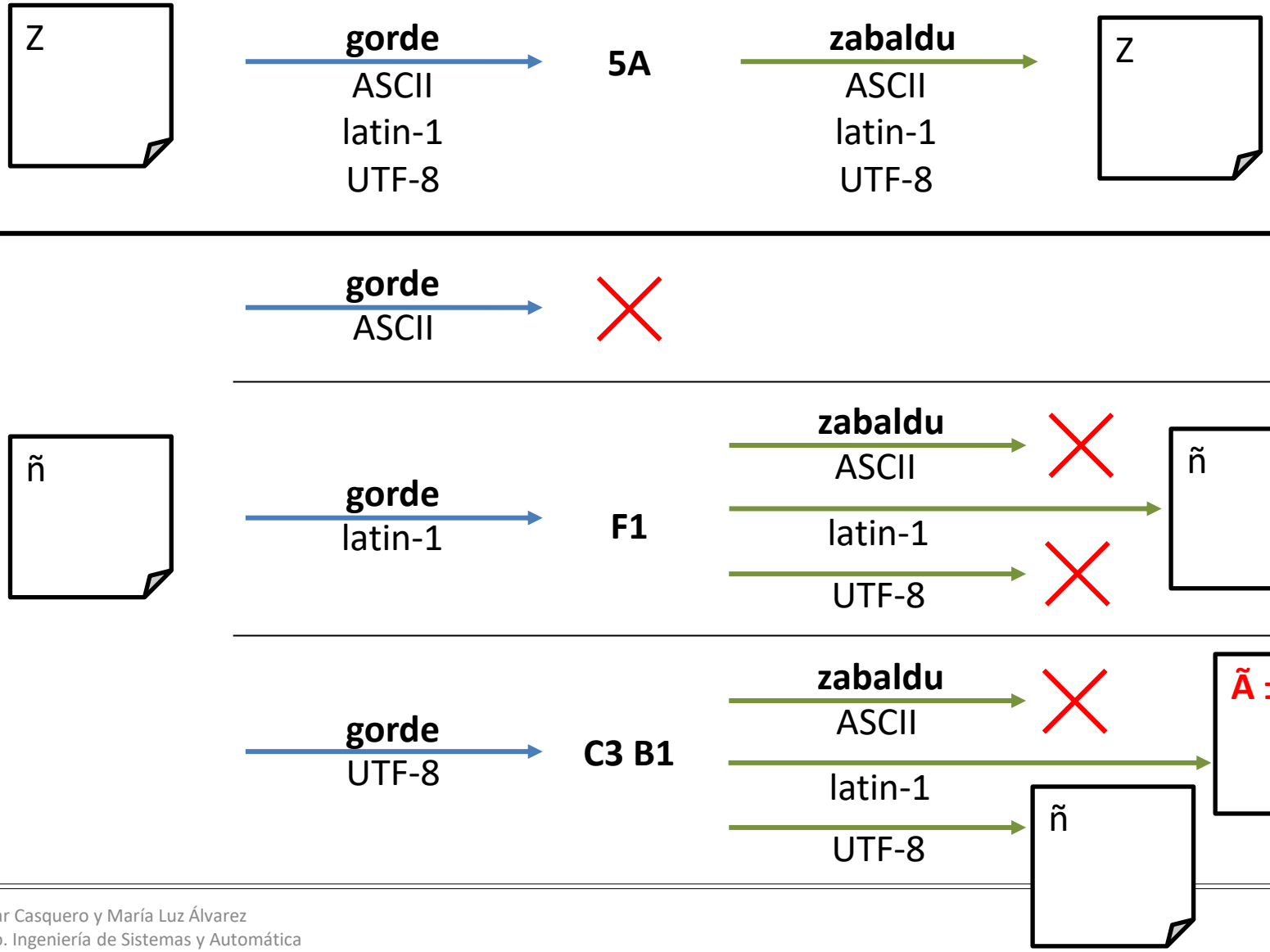
- Demagun ondorengo testua kodifikatu nahi dela: Hola, Iñaki Vázquez!

  US-ASCII: Ezin da kodifikatu, “ñ” y “á” ikurrak ez baitira US-ASCII hiztegian existitzen.

  Latin-1: 48 6F 6C 61 2C 20 49 F1 61 6B 69 20 56 E1 7A 71 76 65 7A 21

  UTF-8: 48 6F 6C 61 2C 20 49 C3 B1 61 6B 69 20 56 C3 A1 7A 71 76 65 7A 21

# KODIFIKAZIOARI BURUZKO ARGIBIDE BATZUK...




Oskar Casquero y María Luz Álvarez
Dpto. Ingeniería de Sistemas y Automática

# HTTP-REN FUNTZIONAMENDUA: MEZUAREN GORPUTZA

- “Content-Type” goiburaren bitartez, zerbitzariak, itzultzen duen eduki mota adierazi dezake: text/html, application/pdf, …

- Nabigatzaileak “Content-Type” goiburua irakurtzen duenean, bere balioa testu motakoa edo beste mota batekoa ote den egiaztatzen du.
  - Edukia beste mota batekoa bada, bera zabaltzeko plugin-ik konfiguratuta ote duen egiaztatzen du. Horrela ez bada, erabiltzaileari edukia gordetzeko edo dagokion aplikazioarekin zabaltzeko aukera ematen dio.
  - Edukia testu motakoa bada, nabigatzialeak berak interpretatu (bistaratu) egiten du. Horretarako zelan kodifikatuta dagoen jakin behar du.
    - HTTP-k edukiaren kodifikazioa “Content-Type” goiburuan adierazteko aukera du:

      Content-Type: text/html; charset=ISO-8859-1

# ADIBIDEA:

## KODIFIKAZIO ARAZO BATEN AZTERKETA

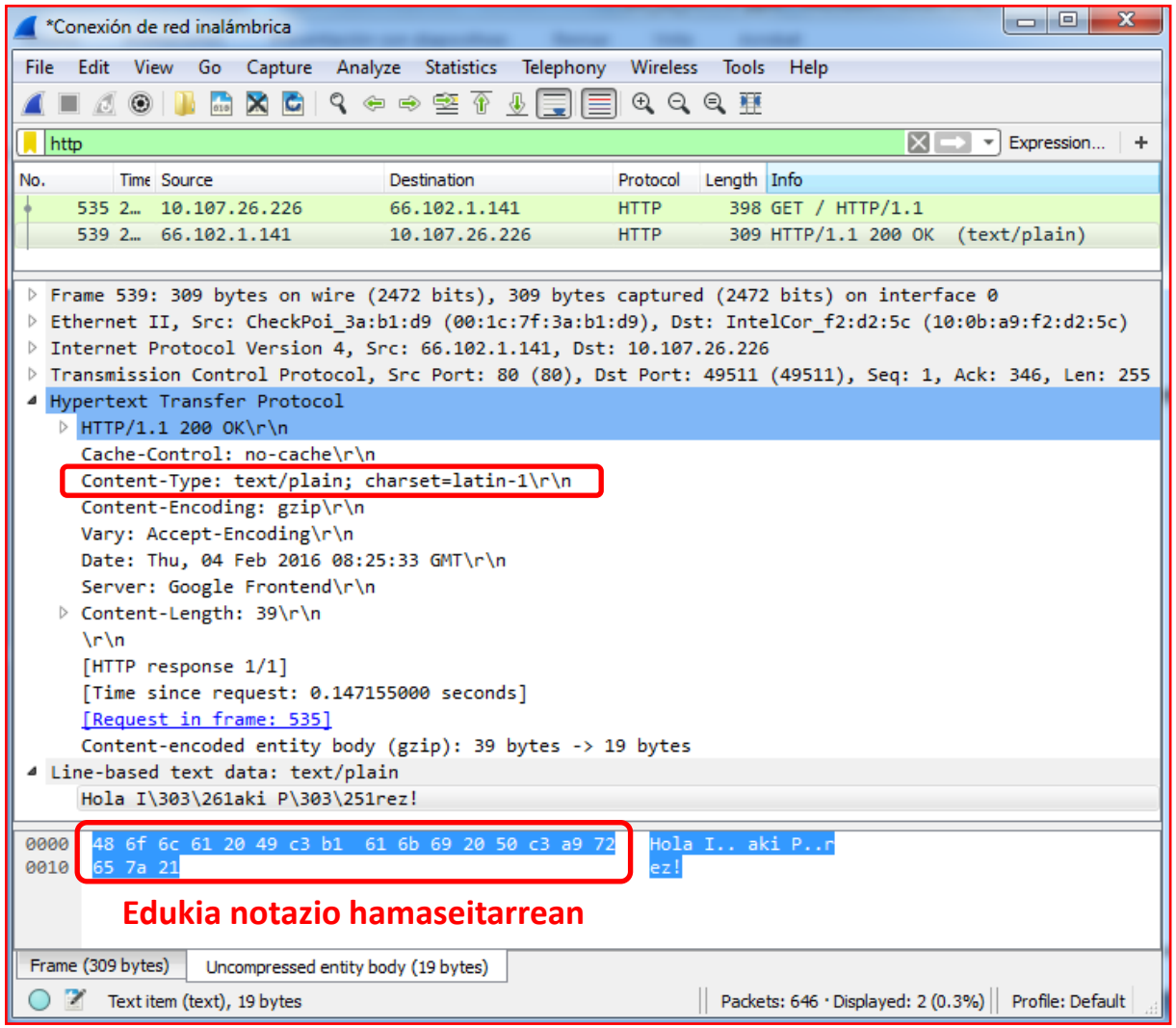
- Wireshark erabilita, nabigatzailean ondorengo URI-a zabaldu eta bere kodifikazio arazoaren arrazoia aurkitu

  http://ws-responsecontentcoding.appspot.com/

±

# ADIBIDEA:

## KODIFIKAZIO ARAZO BATEN AZTERKETA




- Nabigatzaileak edukia “text/plain” motakoa dela ikusten du, beraz, bera arduratzen edukia interpretatzeaz.
- Nola interpretatu behar du testua nabigatzaileak? Hau da, zein kodifikazio taula erabili behar du? “charset” parametroak adierazten duena; kasu honetan: latin-1.

```
48 → H
6F → o
6C → l
61 → a
20 →
49 → I
C3 → Ã
B1 → ±
```

# ADIBIDEA:
# KODIFIKAZIO ARAZO BATEN AZTERKETA

- Zergaitik gertatzen da hau?
  - Zerbitzaria programatu duenak edukia latin-1 -en kodifikatuta dagoela adierazten duen kode lerroa idatzi du:

```
self.response.charset = 'latin-1'
```

  - Baina ez du kontuan izan *write* metodoa erabitzen denean zerbitzariak edukia UTF-8 -n kodifikatzen duela:

```
self.response.write("Hola Iñaki Pérez!")
```

**Content-Type: text/plain; charset: latin-1**

```
import webapp2class MainHandler(webapp2.RequestHandler):
    def get(self):
        self.response.content_type = 'text/plain'
        self.response.charset = 'latin-1'
        self.response.write("Hola Iñaki Pérez!")

app = webapp2.WSGIApplication([('/', MainHandler)], debug=True)
```