

PRIMER PARCIAL SERVICIOS TELEMÁTICOS

DANIELA ZUÑIGA PINO - 2200625

SEGUNDA PARTE

Se requiere configurar la funcionalidad de compresión, a través de la activación del módulo mod_deflate de apache para comprimir archivos usando codificación gzip.

comprobar el funcionamiento correcto de la compresión puede usar la herramienta de inspección de Google Chrome o de otros navegadores. Como se muestra en la figura.

Asegúrese de implementar/probar lo siguiente

- Servidor apache con recursos de suficiente tamaño para verificar la funcionalidad de compresión
- El servidor DNS se debe configurar apropiadamente con una zona llamada [su-nombre.com](#) y la máquina servidor se debe llamar [parcial](#). Por lo tanto, debe resolver para parcial.su-nombre.com
- Probar desde la terminal de la máquina virtual cliente.
- Probar desde un browser (Google)
- Verificar desde la herramienta de inspección de su browser
- Visualizar en un sniffer (Wireshark) la respuesta donde se muestre la compresión realizada.

SOLUCIÓN//

CONFIGURACIONES

DNS

1. Se crea la zona con el nombre respectivo

```
zone "daniela.com" IN {  
    type master;  
    file "daniela.com.fwd";  
};
```

2. Se crean los archivos de configuración

```
root@servidor:/var/named  
$ cat daniela.com.zone  
$ORIGIN daniela.com.  
$TTL 3H  
@ IN SOA parcial.daniela.com. root@daniela.com. (  
    0 ; serial  
    1D ; refresh  
    1H ; retry  
    1W ; expire  
    3H ) ; minimum  
@ IN NS parcial.daniela.com.  
host en la zona  
parcial IN A 192.168.50.3  
cliente IN A 192.168.50.2  
www IN CNAME parcial  
user IN CNAME cliente
```

3. Restart named.service

HTTP

1. Se crea la carpeta parcial, su archivo html y se asocia en el <VirtualHost>

```
<VirtualHost *:80>
    ServerName parcial.daniela.com
    ServerAlias www.daniela.com
    DocumentRoot /var/www/parcial
    CustomLog /var/www/parcial/request.log combined
    DirectoryIndex index.html
</VirtualHost>

<VirtualHost *:80>
    ServerName www.miotrositio.com
    ServerAlias miotrositio.com
    DocumentRoot /var/www/miotrositio
    CustomLog /var/www/miotrositio/request.log combined
    DirectoryIndex index.html
</VirtualHost>

<VirtualHost *:80>
    ServerName server.zuniga.com
    ServerAlias www.zuniga.com
    DocumentRoot /var/www/html
    CustomLog /var/www/html/request.log combined
    DirectoryIndex index.html
</VirtualHost>

IncludeOptional conf.d/*.conf

[root@servidor conf]# cd /var/www/
[root@servidor www]# ls
cgi-bin html miotrositio
[root@servidor www]# mkdir parcial
[root@servidor www]# cd /parcial
-bash: cd: /parcial: No such file or directory
[root@servidor www]# ls
cgi-bin html miotrositio parcial
[root@servidor www]# cd parcial
[root@servidor parcial]# nano index.html
[root@servidor parcial]# vim index.html

[8]+ Stopped vim index.html
[root@servidor parcial]# chmod 755 index.html
[root@servidor parcial]# ls -l
total 4
-rwxr-xr-x 1 root root 13 Feb 26 03:05 index.html
[root@servidor parcial]# cd ..
[root@servidor www]# ls
cgi-bin html miotrositio parcial
[root@servidor www]# chmod 755 parcial
[root@servidor www]# chown -R apache:apache /var/www/parcial
[root@servidor www]# service httpd restart
Redirecting to /bin/systemctl restart httpd.service
```

FUNCIONALIDAD DE COMPRESIÓN - GZIP

1. Se entra a `cd /etc/httpd/conf` y se añade

LoadModule deflate_module modules/mod_deflate.so

AddOutputFilterByType DEFLATE text/plain

AddOutputFilterByType DEFLATE text/html

AddOutputFilterByType DEFLATE text/xml

AddOutputFilterByType DEFLATE text/css

AddOutputFilterByType DEFLATE application/xml

AddOutputFilterByType DEFLATE application/xhtml+xml

AddOutputFilterByType DEFLATE application/rss+xml

AddOutputFilterByType DEFLATE application/javascript

AddOutputFilterByType DEFLATE application/x-javascript

AddOutputFilterByType DEFLATE application/octet-stream

AddOutputFilterByType DEFLATE image/jpeg

AddOutputFilterByType DEFLATE image/png

AddOutputFilterByType DEFLATE image/gif

AddOutputFilterByType DEFLATE video/mp4

AddOutputFilterByType DEFLATE video/webm

AddOutputFilterByType DEFLATE video/x-msvideo

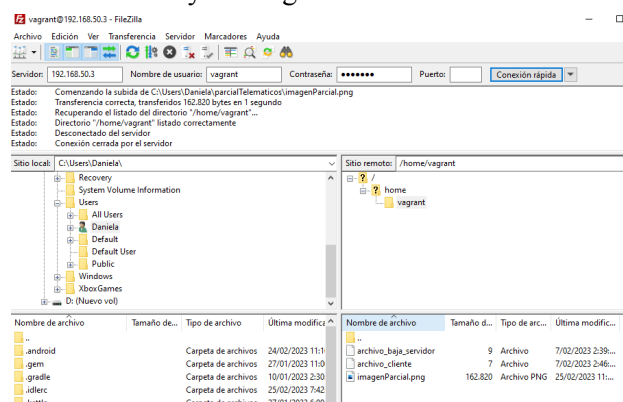
AddType application/octet-stream .pdf .zip .txt .mp4 .avi .jpg .png

```
LoadModule deflate_module modules/mod_deflate.so

# compress text, html, javascript, css, xml:
AddOutputFilterByType DEFLATE text/plain
AddOutputFilterByType DEFLATE text/html
AddOutputFilterByType DEFLATE text/xml
AddOutputFilterByType DEFLATE text/css
AddOutputFilterByType DEFLATE application/xml
AddOutputFilterByType DEFLATE application/xhtml+xml
AddOutputFilterByType DEFLATE application/rss+xml
AddOutputFilterByType DEFLATE application/javascript
AddOutputFilterByType DEFLATE application/x-javascript
AddOutputFilterByType DEFLATE application/octet-stream
AddOutputFilterByType DEFLATE image/jpeg
AddOutputFilterByType DEFLATE image/png
AddOutputFilterByType DEFLATE image/gif
AddOutputFilterByType DEFLATE video/mp4
AddOutputFilterByType DEFLATE video/webm
AddOutputFilterByType DEFLATE video/x-msvideo
AddType application/octet-stream .pdf .zip .txt .mp4 .avi .jpg .png

IncludeOptional conf.d/*.conf
```

2. Se sube la foto y la imagen a FTP desde filezilla



3. Se copia a la carpeta parcial los dos archivos subidos

```
C:\Users\Daniela\parcialTelematicos>vagrant ssh servidor
Last login: Sun Feb 26 02:14:28 2023 from 10.0.2.2
Last login: Sun Feb 26 02:14:28 2023 from 10.0.2.2
[vagrant@servidor ~]$ sudo -i
[root@servidor ~]# service vsftpd start
Redirecting to /bin/systemctl start vsftpd.service
[root@servidor ~]# service named start
Redirecting to /bin/systemctl start named.service
[root@servidor ~]# service httpd start
Redirecting to /bin/systemctl start httpd.service
[root@servidor ~]# cd /home/vagrant
[root@servidor vagrant]# ls
archivo_baja_servidor  archivo_cliente  imagenParcial.PNG  videoparcial.mp4
[root@servidor vagrant]# mv imagenParcial.PNG /var/www/parcial
[root@servidor vagrant]# ls
archivo_baja_servidor  archivo_cliente  videoparcial.mp4
[root@servidor vagrant]# mv videoparcial.mp4 /var/www/parcial
[root@servidor vagrant]# ls
archivo_baja_servidor  archivo_cliente
[root@servidor vagrant]# cd /var/www/parcial
[root@servidor parcial]# ls
imagenParcial.PNG  index.html  request.log  videoparcial.mp4
```

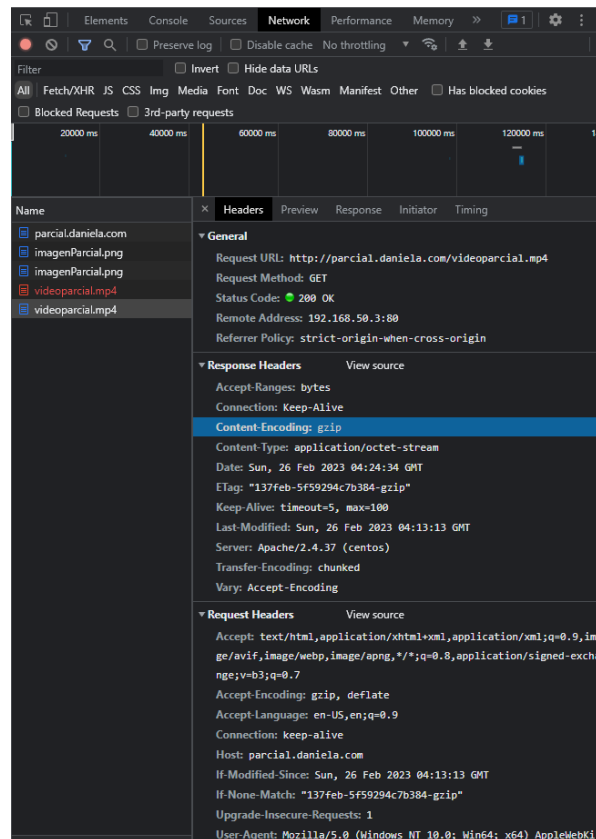
PRUEBAS DE FUNCIONAMIENTO

```
service vsftpd start
service named start
service httpd start
```

GOOGLE

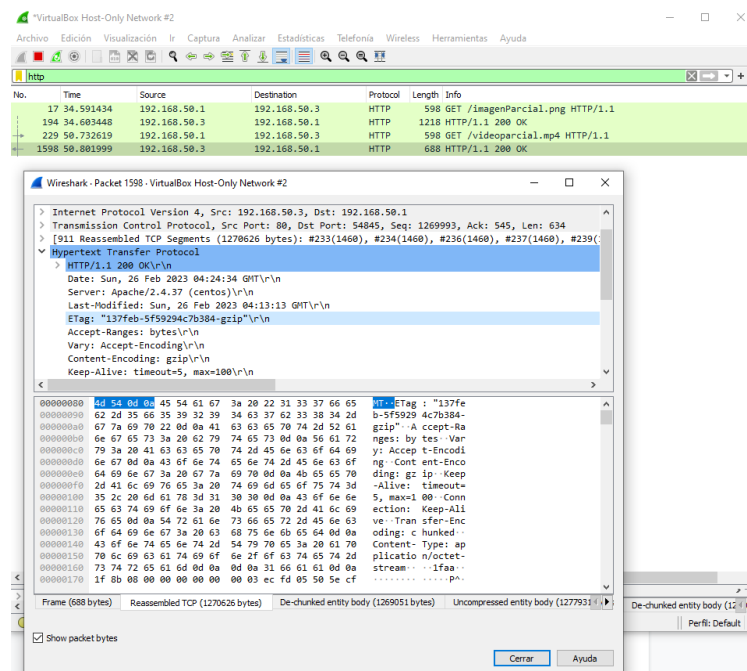
- Se entra a las páginas
parcial.daniela.com
 Link img: parcial.daniela.com/imagenParcial.png;
 Link video: parcial.daniela.com/videoparcial.mp4

- Se inspecciona en network y se selecciona img y video para ver como g-zip si está funcionando como queremos en el response



WIRESHARK

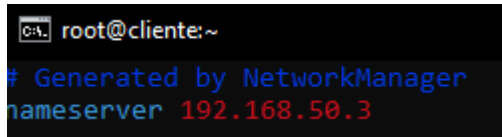
- Se entra a virtualbox #2 y se filtra por http
- Se hace la petición en google
- Se entra a la respuesta OK
- Buscamos HyperText y en Etag y Content podemos ver que se usa g-zip



MÁQUINA VIRTUAL CLIENTE

CURL

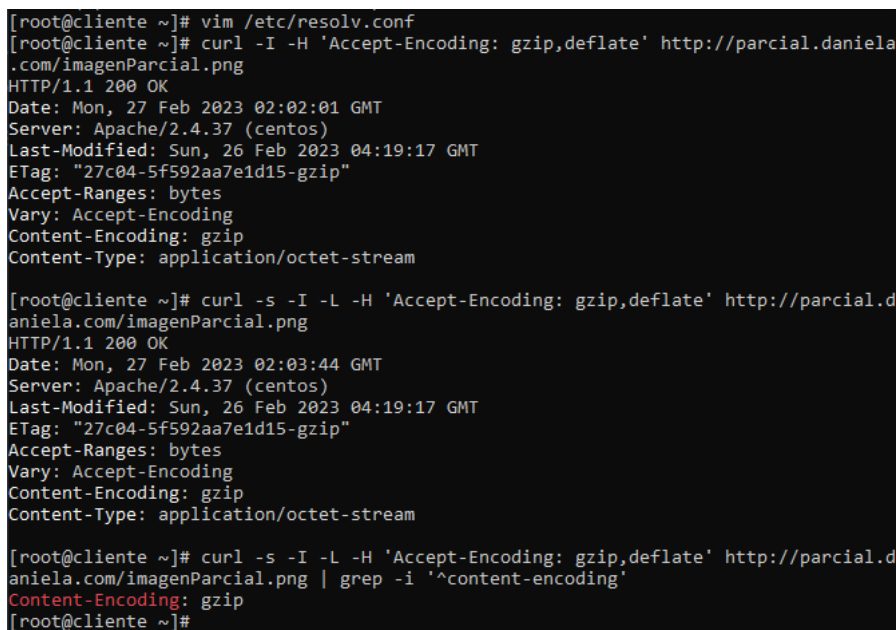
vim /etc/resolv.conf



```
root@cliente:~  
# Generated by NetworkManager  
nameserver 192.168.50.3
```

```
curl -I -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/imagenParcial.png  
curl -s -I -L -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/imagenParcial.png  
curl -s -I -L -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/imagenParcial.png  
| grep -i '^content-encoding'
```

```
curl -I -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/videoparcial.mp4  
curl -s -I -L -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/videoparcial.mp4  
curl -s -I -L -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/videoparcial.mp4 |  
grep -i '^content-encoding'
```



```
[root@cliente ~]# vim /etc/resolv.conf  
[root@cliente ~]# curl -I -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/imagenParcial.png  
HTTP/1.1 200 OK  
Date: Mon, 27 Feb 2023 02:02:01 GMT  
Server: Apache/2.4.37 (centos)  
Last-Modified: Sun, 26 Feb 2023 04:19:17 GMT  
ETag: "27c04-5f592aa7e1d15-gzip"  
Accept-Ranges: bytes  
Vary: Accept-Encoding  
Content-Encoding: gzip  
Content-Type: application/octet-stream  
  
[root@cliente ~]# curl -s -I -L -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/imagenParcial.png  
HTTP/1.1 200 OK  
Date: Mon, 27 Feb 2023 02:03:44 GMT  
Server: Apache/2.4.37 (centos)  
Last-Modified: Sun, 26 Feb 2023 04:19:17 GMT  
ETag: "27c04-5f592aa7e1d15-gzip"  
Accept-Ranges: bytes  
Vary: Accept-Encoding  
Content-Encoding: gzip  
Content-Type: application/octet-stream  
  
[root@cliente ~]# curl -s -I -L -H 'Accept-Encoding: gzip,deflate' http://parcial.daniela.com/imagenParcial.png | grep -i '^content-encoding'  
Content-Encoding: gzip  
[root@cliente ~]#
```

TERCERA PARTE




TÚNEL

Realice un túnel hacia el servidor web implementado en clase, de manera que los recursos de su servidor web puedan ser visualizados desde cualquier lugar por fuera de su red local. Para efectos de prueba, agregar una página personalizada a su sitio web.

Se sugiere usar:

- Port forwarding en Vagrant
- Vagrant share
- ngrok (agregarlo al path --variables de entorno-- una vez instalado)

1. Se inicia sesión <https://dashboard.ngrok.com/get-started/setup> y se instala ngrok en la carpeta donde esta vagrant

Disco local (C:) > Usuarios > Daniela > parcialTelematicos	
Nombre	Fecha de modificación
 .vagrant	21/02/2023 11:09 p. m.
 ngrok.exe	13/01/2023 4:20 p. m.
 Vagrantfile	6/02/2023 10:15 a. m.

2. Se inicia ngrok y se pega este comando:

```
ngrok http 192.168.50.3:80
```

```
ngrok config add-authtoken 2MFNqGlj9FJon4MpektwVVVe89a_5gHKRwqAmnNyxj5LiVz36
```

```
C:\Users\Daniela\parcialTelematicos>ngrok.exe - ngrok http 192.168.50.3:80
ngrok
We added a plan for ngrok hobbyists @ https://ngrok.com/personal
Session Status      online
Account             daniela.zuniga_pino@uao.edu.co (Plan: Free)
Version             3.1.1
Region              United States (us)
Latency             107ms
Web Interface       http://127.0.0.1:4040
Forwarding           https://b7cf-186-83-221-90.ngrok.io -> http://192.168.50.3:80
Connections
  ttl    opn    rt1    rt5    p50    p90
   1      0    0.01   0.00   5.66   5.66
HTTP Requests
-----
GET /favicon.ico    404 Not Found
GET /               200 OK
```

3. Se abre cmd, se inicia vagrant, se inician los servicios

```
C:\Users\Daniela\parcialTelematicos>vagrant ssh servidor
Last login: Wed Feb 22 01:28:02 2023 from 10.0.2.2
Last login: Wed Feb 22 01:28:02 2023 from 10.0.2.2
[vagrant@servidor ~]$ sudo -i
[root@servidor ~]# service vsftpd start
Redirecting to /bin/systemctl start vsftpd.service
[root@servidor ~]# service named start
Redirecting to /bin/systemctl start named.service
[root@servidor ~]# service httpd start
Redirecting to /bin/systemctl start httpd.service
[root@servidor ~]#
```

4. **Sustentación:** se copia el link de forwarding y se ingresa



CUARTA PARTE

Usando su lenguaje preferido, implemente y pruebe una red peer-to-peer basada en blockchain

Tutorial:  Build Your Own Blockchain Using Python

```
import hashlib
import datetime

# Creamos la clase Block, que representa cada bloque en nuestra cadena de bloques
class Block:
    def __init__(self, data, timestamp, previous_hash):
        self.data = data # datos almacenados en el bloque
        self.timestamp = timestamp # marca de tiempo del bloque
        self.previous_hash = previous_hash # hash del bloque anterior
        self.hash = self.calculate_hash() # calculamos el hash del bloque actual

    def calculate_hash(self):
        # función para calcular el hash del bloque actual
        hash_data = str(self.data) + str(self.timestamp) + str(self.previous_hash)
        return hashlib.sha256(hash_data.encode()).hexdigest()

# Creamos la clase Blockchain, que es la cadena de bloques
class Blockchain:
    def __init__(self):
        self.chain = [self.create_danielapino_block()] # inicializamos la cadena de bloques con el bloque

    def create_danielapino_block(self):
        # función para crear el bloque daniela pino
        return Block("Daniela Pino Block", datetime.datetime.now(), "0")

    def add_block(self, new_block):
        # función para añadir un nuevo bloque a la cadena de bloques
        new_block.previous_hash = self.chain[-1].hash # definimos el hash del bloque anterior
        new_block.hash = new_block.calculate_hash() # calculamos el hash del nuevo bloque
        self.chain.append(new_block) # añadimos el nuevo bloque a la cadena de bloques

    def print_blocks(self):
        # función para imprimir los bloques en la cadena de bloques
        for block in self.chain:
            print("Data:", block.data) # mostramos los datos del bloque
            print("Timestamp:", block.timestamp) # mostramos la marca de tiempo del bloque
            print("Previous hash:", block.previous_hash) # mostramos el hash del bloque anterior
            print("Hash:", block.hash) # mostramos el hash del bloque actual
            print("\n") # agregamos una línea en blanco entre bloques

# Creamos una instancia de la clase Blockchain y añadimos algunos bloques de ejemplo
blockchain = Blockchain()
blockchain.add_block(Block("First block data", datetime.datetime.now(), "danielapino_block_hash"))
blockchain.add_block(Block("Second block data", datetime.datetime.now(), ""))
blockchain.add_block(Block("Third block data", datetime.datetime.now(), ""))

# Imprimimos los bloques en la cadena de bloques
blockchain.print_blocks()
```

Sustentación:

```
=== RESTART: C:/Users/Daniela/parcialTelematicos/blockchainDanielaParcial.py ===  
Data: Daniela Pino Block  
Timestamp: 2023-02-25 20:19:37.247743  
Previous hash: 0  
Hash: 58d5b878bb04c527a4d9b9e5c77962157c2f95d5cbfdbff9bcf2c8d9e62ee7fd
```

```
Data: First block data  
Timestamp: 2023-02-25 20:19:37.247743  
Previous hash: 58d5b878bb04c527a4d9b9e5c77962157c2f95d5cbfdbff9bcf2c8d9e62ee7fd  
Hash: 6a61ebf1327391fa9535a9b01fa378cf7c9979e0a4639d174e6a41a8858c1fa0
```

```
Data: Second block data  
Timestamp: 2023-02-25 20:19:37.247743  
Previous hash: 6a61ebf1327391fa9535a9b01fa378cf7c9979e0a4639d174e6a41a8858c1fa0  
Hash: 38c0819c66adc4b3e522f482195328a473982f0deb2203a57794d92432be540c
```

```
Data: Third block data  
Timestamp: 2023-02-25 20:19:37.247743  
Previous hash: 38c0819c66adc4b3e522f482195328a473982f0deb2203a57794d92432be540c  
Hash: 5ea45790d48e6d57e745d33fa10d471a3238d172cd1d2e9a762024c58a598769
```