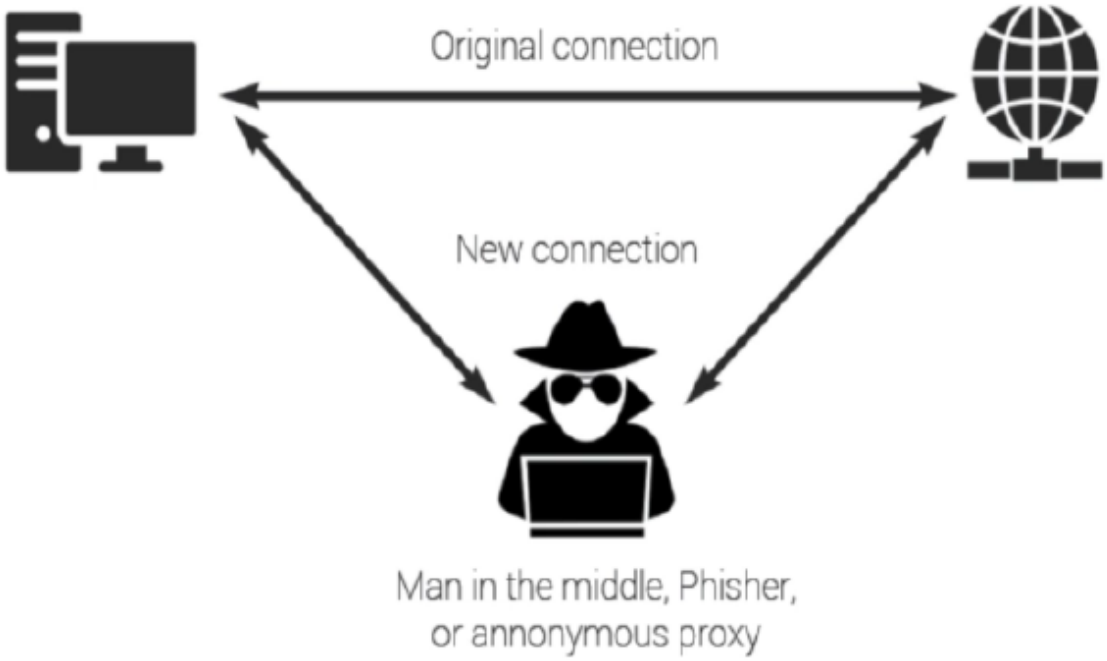
Урок 3. HTTP

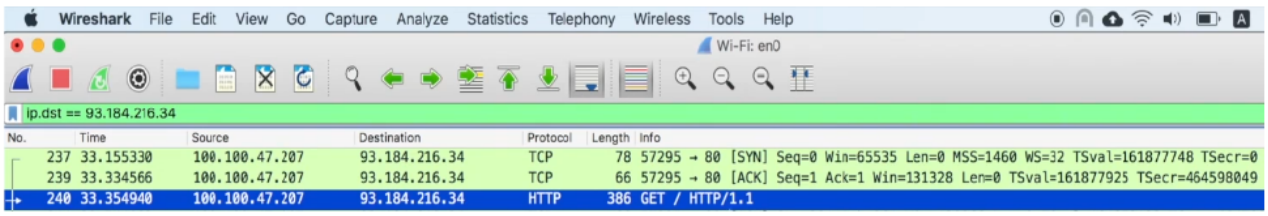
Заметки

```
Referer:
        rel = noreferrer
        referrerpolicy = no-referer
```

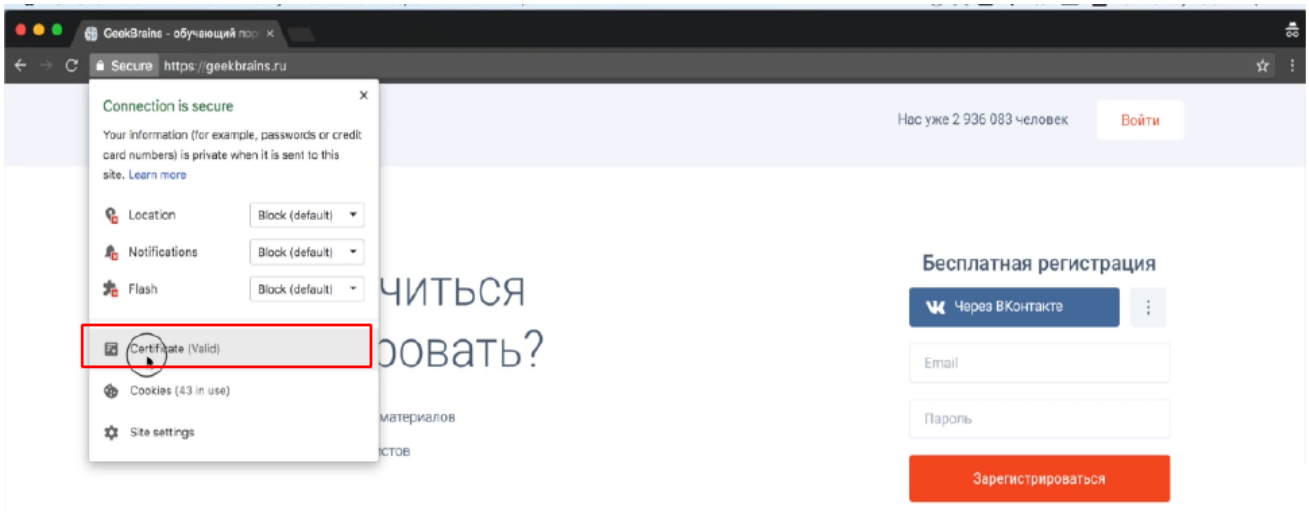
1. Example.com
2. <a href="http://example.com"
   referrerpolicy="no-referrer">ReferrerPolicy Attribute



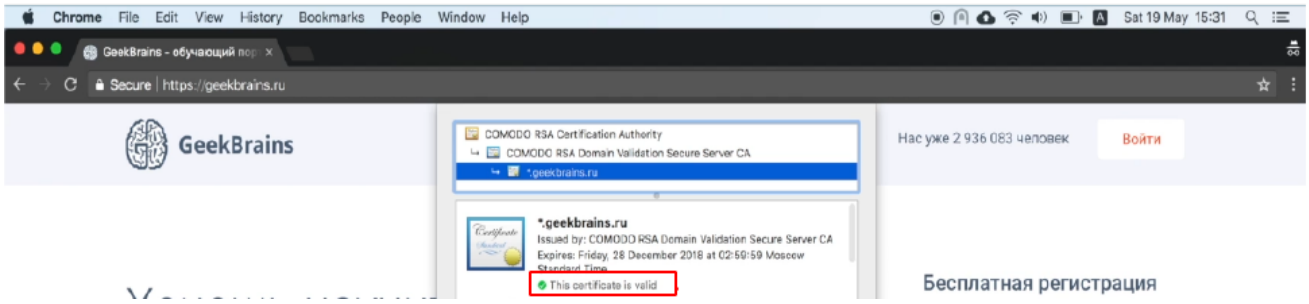
столбце Source=100.100.47.207) на IP-адрес сайта example.com Destination=93.184.216.34:



Wireshark расшифровывает, что это GET-запрос за главной страницей.



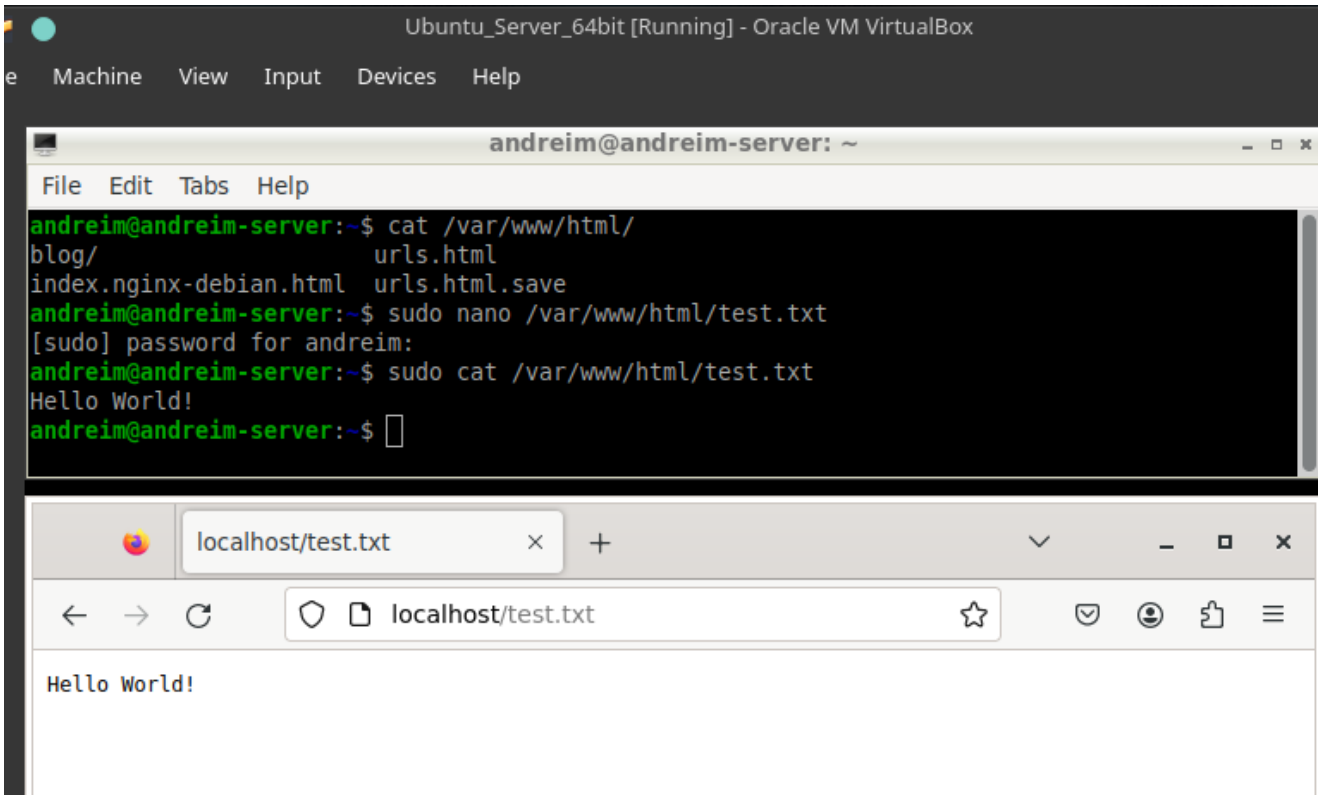
Мы получаем информацию о сертификате, можно посмотреть больше информации:



Задание

1. Создать файл test.txt в корневом каталоге сервера. Получить этот файл через браузер.

- Установить в терминале программу curl, получить тот же файл с помощью этой программы.
- Установить telnet или netcat, получить тот же файл с помощью одной из этих программ.




```
sudo apt install telnet curl netcat
curl -i https://geekbrains.ru | less

curl localhost/test.txt
telnet localhost 80   GET /test.txt HTTP/1.1   Host: localhost
netcat localhost 80   GET /test.txt   ...
```

```
andreim@andreim-server:~$ curl localhost/test.txt
Hello World!
andreim@andreim-server:~$ telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET /test.txt HTTP/1.1
Host: localhost
                        Enter
                        Enter
                        Enter

HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Mon, 12 Jun 2023 07:18:41 GMT
Content-Type: text/plain
Content-Length: 13
Last-Modified: Mon, 12 Jun 2023 07:04:57 GMT
Connection: keep-alive
ETag: "6486c399-d"
Accept-Ranges: bytes

Hello World!
```

```
andreim@andreim-server:~$ netcat localhost 80
GET /test.txt
Hello World!
^C
andreim@andreim-server:~$ netcat localhost 80
GET /test.txt HTTP/1.1
Host: localhost

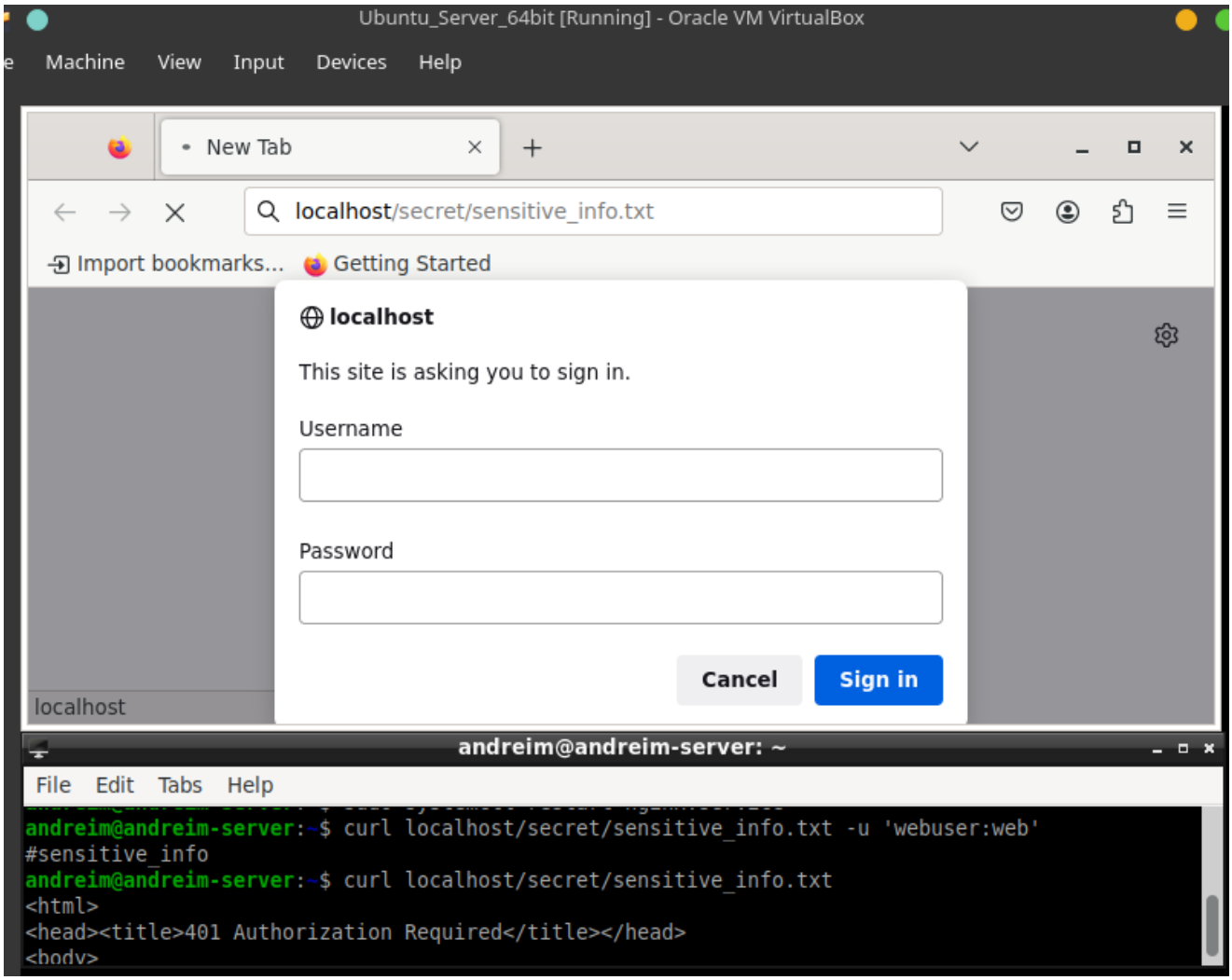
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Mon, 12 Jun 2023 07:27:10 GMT
Content-Type: text/plain
Content-Length: 13
Last-Modified: Mon, 12 Jun 2023 07:04:57 GMT
Connection: keep-alive
ETag: "6486c399-d"
Accept-Ranges: bytes

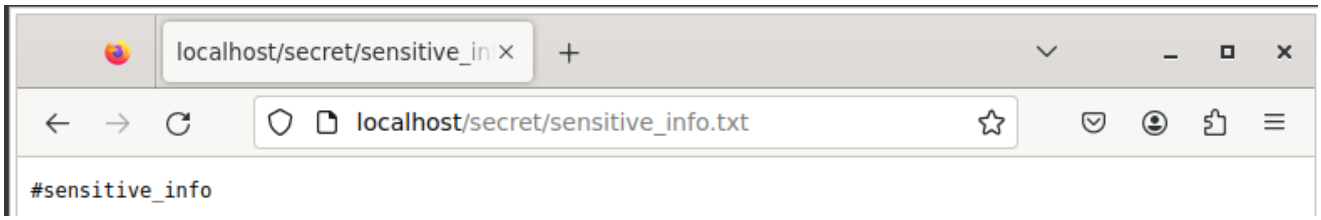
Hello World!
```

2. Создать на сервере файл sensitive_info.txt. Добавить базовую HTTP авторизацию для этого файла.

- Получить этот файл через браузер.
- Получить тот же файл с помощью curl и telnet или netcat.

```
sudo apt install apache2-utils
sudo htpasswd -c /var/www/pass webuser
sudo nano /etc/nginx/sites-available/default
sudo systemctl restart nginx.service
```






```
echo -n 'webuser:web' | openssl base64 -base64
```

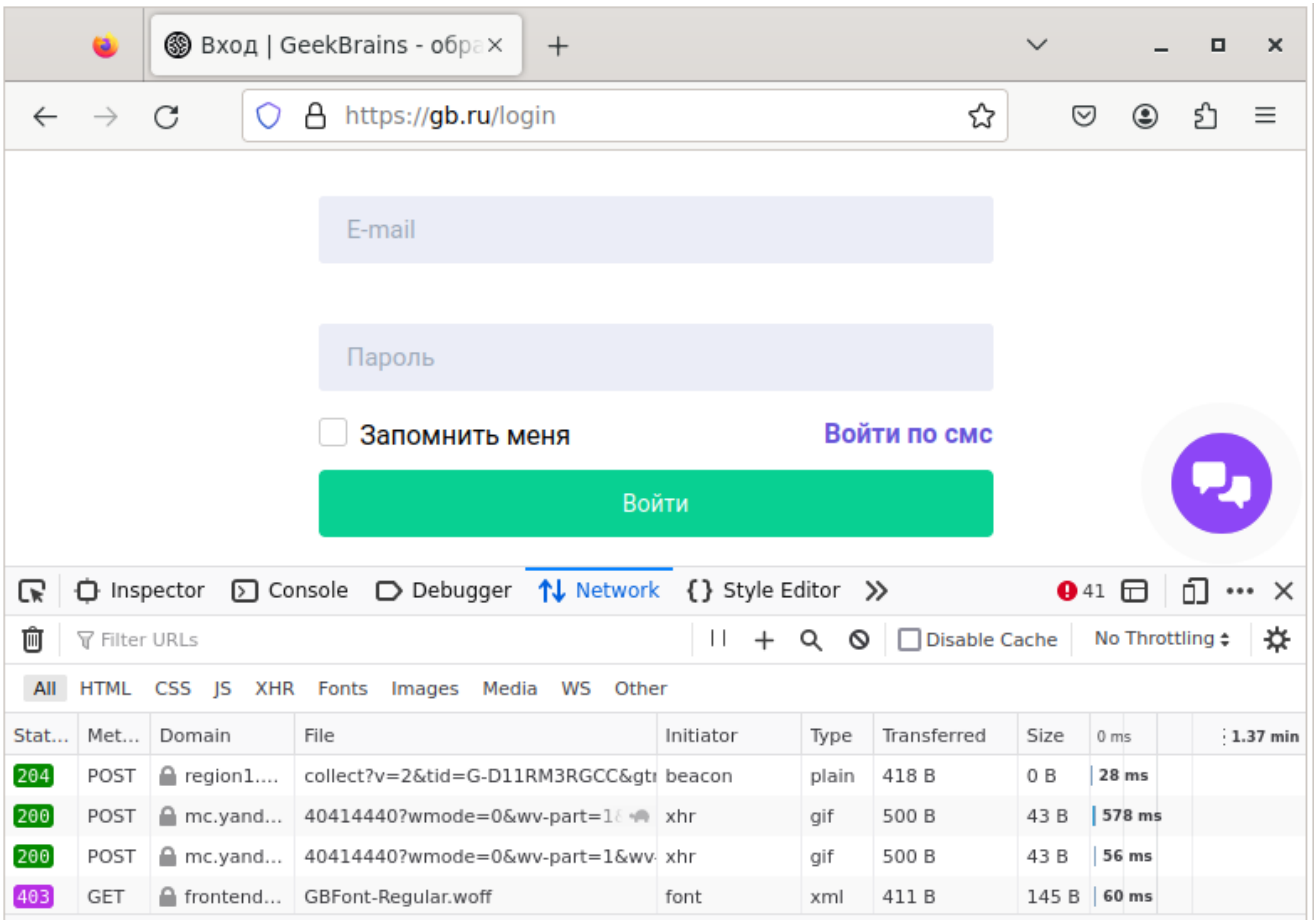
```
andreim@andreim-server:~$ telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET /secret/sensitive_info.txt HTTP/1.1
Host: localhost
Authorization: Basic d2VidXNlcjp3ZWI=

HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Mon, 12 Jun 2023 09:49:53 GMT
Content-Type: text/plain
Content-Length: 16
Last-Modified: Mon, 12 Jun 2023 08:17:50 GMT
Connection: keep-alive
ETag: "6486d4ae-10"
Accept-Ranges: bytes

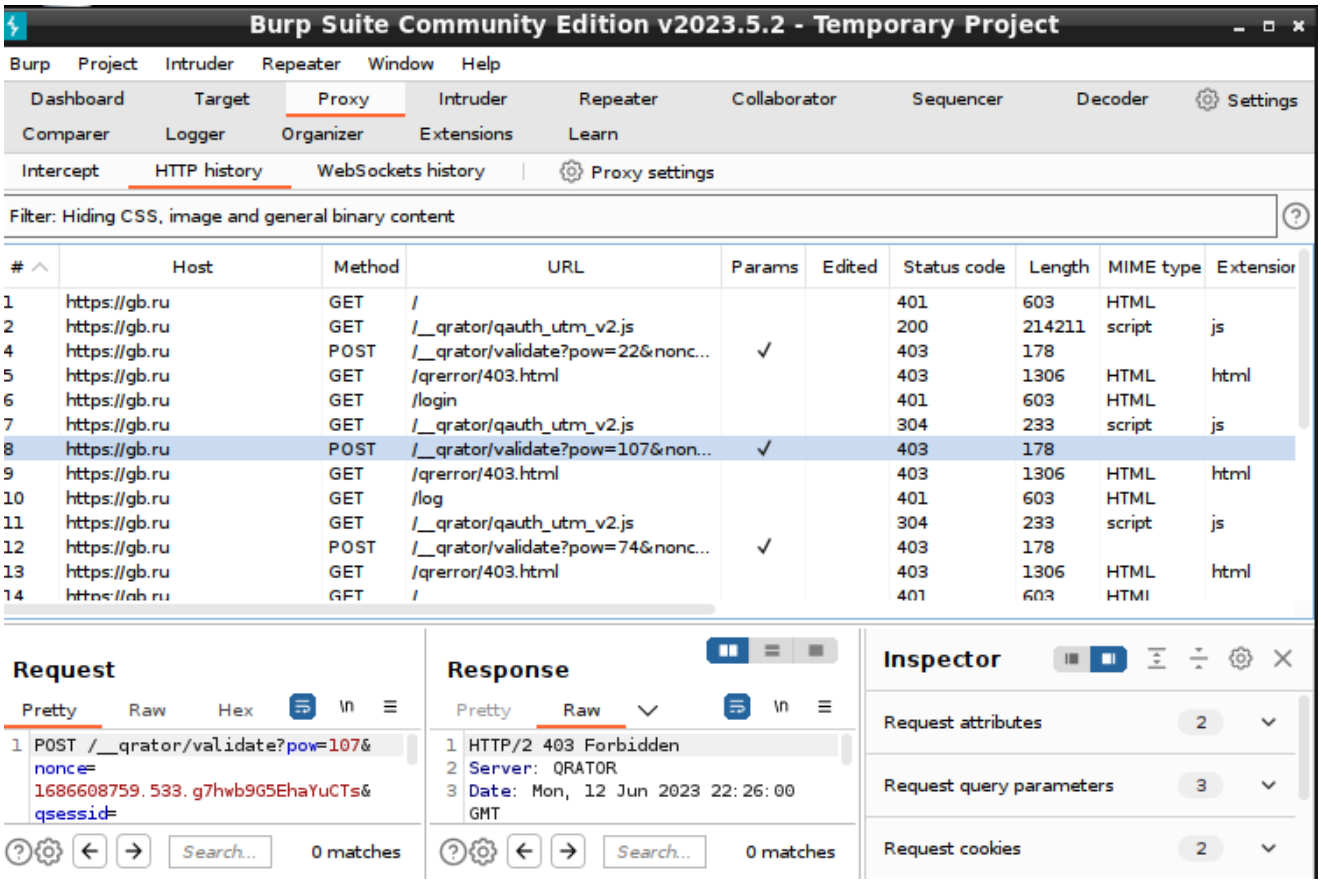
#sensitive_info
```

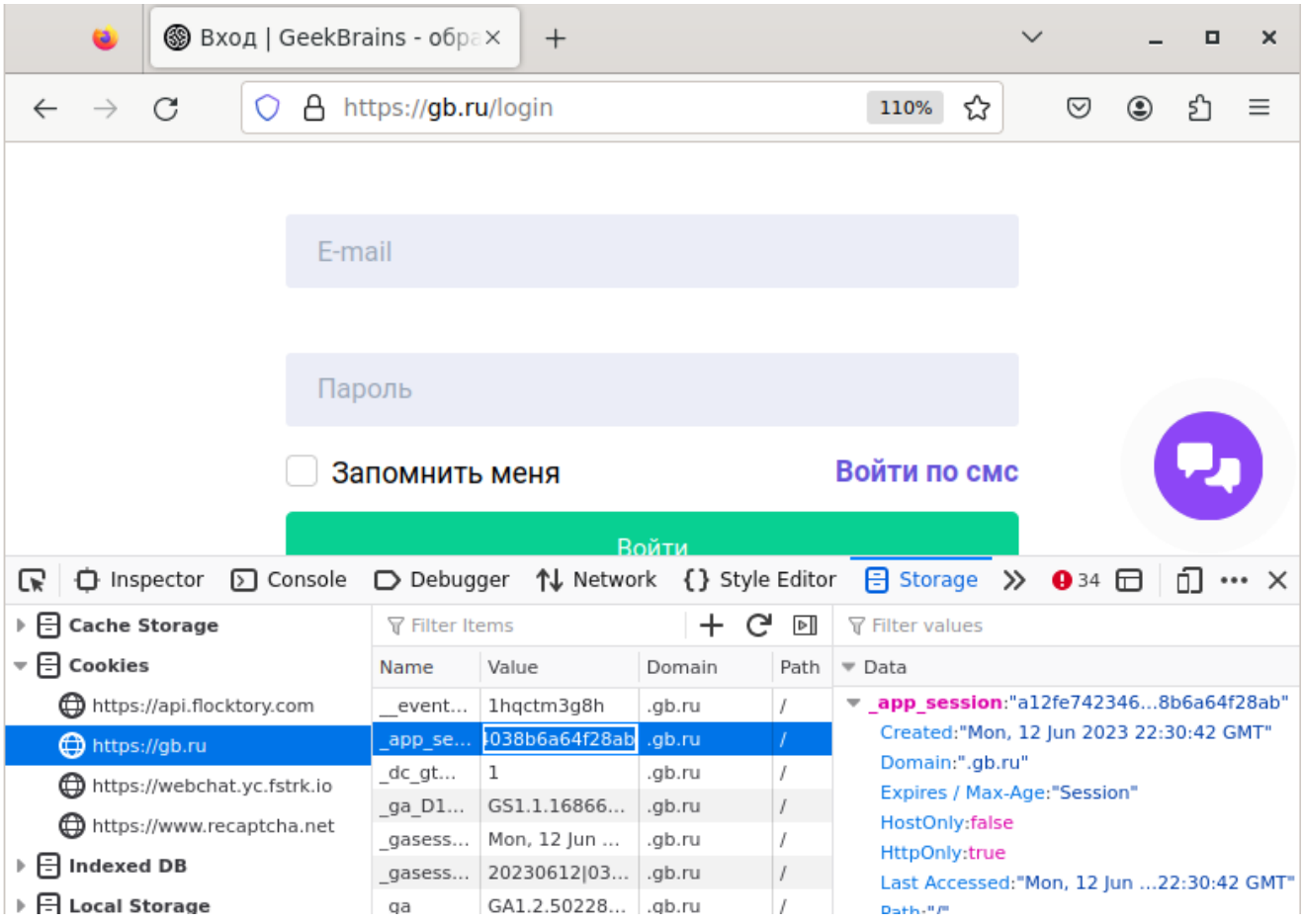
```
netcat localhost 80
...
```

3. Открыть инструменты разработчика, вкладку Сеть (Network). Зайти на сайт https://geekbrains.ru. Проанализировать куки каждого запроса за HTML и картинками. Какие запросы уходят с куками, а какие без кук? Почему в каждом из случаев происходит именно такое поведение?



```
Firefox -> gb.ru -> Inspect(Q) -> network / storage
Burp Suite -> Proxy -> HTTP history -> Browser: gb.ru
```





Куки в браузере выглядят как пары "ключ-значение" и похожи на GET-параметры, используются в качестве идентификатора сессии, сохраняются данные на стороне клиента и его браузера.

Например, после запроса на логин-пароль одна из Куки, которую выставил сервер - сессионная (app_session). Заменив логин и пароль, получем сессионную куки и доступ к аккаунту пользователя. Если знаем сессионную куки, то возможен доступ к аккаунту пользователя без знания логина и пароля ...

Куки имеют метки с жизенным циклом сессии:

```
        Expires - время жизни куки
        Domain - на какой домен проставляется куки
        Secure - отправлять куки через защищенное соединение
```

4. • Для выполнения этого задания вам потребуется:
   1)Настроить домены `attacker.com`, `sub.attacker.com`, `sub.sub.attacker.com`, `victim.com`. Каждый из этих доменов должен указывать на `127.0.0.1`
   2)Настроить установку кук для доменов. Добавьте следующий конфигурационный файл nginx (изменив root сервера на свой):

```
$ cat /etc/nginx/sites-available/cookie-research.conf
server {
listen 80;
server_name attacker.com;
root /var/www/html;
```

```
location / {
 add_header "Set-Cookie" "test1=attacker-com_sub-attacker-com;
Domain=sub.attacker.com";
        add_header "Set-Cookie" "test2=attacker-com_victim-com;
Domain=victim.com";
        try_files $uri $uri/ =404;
}
```

```
}

server {
listen 80;
server_name sub.attacker.com;
```

```
root /var/www/html;
```

```
location / {
 add_header "Set-Cookie" "test3=sub-attacker-com_attacker-com;
Domain=attacker.com";
        try_files $uri $uri/ =404;
}
```

```
}

server {
listen 80;
server_name sub.sub.attacker.com;
root /var/www/html;
```

```
location / {
 add_header "Set-Cookie" "test4=sub-sub-attacker-com_attacker-com;
Domain=attacker.com";
        try_files $uri $uri/ =404;
}
```

```
}
```
``` Проведите исследование механизма проставления кук, для этого попробуйте установить следующие куки: 1. С домена`attacker.com`на домен`sub.attacker.com` 2. С домена`attacker.com`на домен`victim.com` 3. С домена`sub.attacker.com`на домен`attacker.com` 4. С домена`sub.sub.attacker.com`на домен`attacker.com`
По каждому пункту ответьте на вопросы:
1. Куда установились куки?
2. Если не установились, то почему?
Обобщите полученные знания и напишите вывод в формате: "Домен может проставлять куки для себя, для ... и ..., но не может проставлять куки для ..., ... и ...".
[](https://gb.ru/lessons/330026/homework)
5. (*) Сгенерировать самоподписанный сертификат и разместить его на своем сервере.
```

```
'''
```