Ministère de l'éducation, de la culture et de la recherche
de la République de Moldova
Université technique de Moldavie
Faculté d'Ordinateur, Informatique et Microélectronique
Filière francophone "Informatique"

Compte Rendu

Programmation réseau

Travail pratique nr.3

Thème: HTTP Client

Effectué par l'étudiant(e) de gr FI-181 : Bonta Alexandr

Vérifié par le professeur : Buldumac Oleg

Chișinau 2021

Proiectul pe github: https://github.com/sasa-bonta/PR

Scopul:

Să se creeze o aplicație client http

Pentru nota 9 si 10:

- Cererile HTTP să fie făcute prin intermediul unui proxy
- Să se utilizeze expresiile regulate

Pentru nota 9 si 10:

- Să se utilizeze firele de execuții și tehnici de sincronizare
- Clientul trebuie să se poată autentifica pe resursă utilizînd cookies

Atenție:

- Pentru acest laborator utilizați librării HTTP deja existente, nu este necesar de a utiliza Socket API. Cine dorește poate să facă prin socket ca și la primul laborator.
- Clientul trebuie să facă cereri GET, POST, HEAD și OPTIONS
- Aplicația poate fi consolă sau GUI
- Nu sunteți limitați la funcțional, resura(pagina web) la care clientul o să facă cereri HTTP este la alegere.
- Vă recomand să folosiți proxy private și nu free: https://proxy-seller.com
- Aplicația elaborată trebuie să posede o logică bine definită

Întrebări la apărarea laboratorului:

• Cum este formatat corpul unei cereri HTTP pentru o cerere HTTP de tip POST ?

```
POST /test/demo_form.php HTTP/1.1
          Host: w3schools.com
          name1=value1&name2=value2
```

• De unde știe un client HTTP ce tip de conținut trimite serverul HTTP ?

In raspunsul serverului gasim Content-Type=Tipul_Continutului

• Cum decide un client dacă ar trebui să aibă încredere în certificatul unui server ?

Prin verificarea certificatului SSL

• Care este problema principală cu certificatele autosemnate ?

Browserele vor avertiza utilizatorii cu privire la autenticitatea serverului pe care incearca sa se conecteze, iar pentru volum mare de date si date sensibile (datele cardului bancar) este recomandat de procurat un certificat autorizat.

• Conexiunea persistentă HTTP – care sunt principalele beneficii ?

1. Procesorul este mai putin incarcat, de asemenea si consumul de memorie.

2. Se poate de utilizat HTTP pipelining.
3. Micsoreaza posibilitatea de supraincarcare a retelei.
4. Nu trebuie de creat o conexiune noua.
5. Erorile HTTP se itroc fara icheirea conexiunii.

• Ce este negocierea conținutului în HTTP și cum are loc ?

Dacă un client nu are cunoștințe despre suportul unui server pentru HTTP / 2.0, acesta începe cu HTTP / 1.1 și încearcă o actualizare la HTTP / 2.0 după cum urmează:

```
GET /default.htm HTTP/1.1

Host: server.example.com

Connection: Upgrade

Upgrade: HTTP/2.0
```

Dacă serverul nu acceptă noul protocol, acesta va răspunde pur și simplu clientului folosind HTTP / 1.1:

```
HTTP/1.1 200 OK

Content-length: 243

Content-type: text/html

...
```

Dacă serverul trece la noul protocol, acesta va semnaliza printr-un răspuns 101. Serverul trece la HTTP / 2.0 imediat după linia goală care încheie răspunsul 101 [I-D.ietf-httpbis-p2-semantics].

```
HTTP/1.1 101 Switching Protocols

Connection: Upgrade

Upgrade: HTTP/2.0


[ HTTP/2.0 frame ]
```

• Care sunt tipurile de negociere a conținutului HTTP ?

```
Abbreviated handshake

A full handshake
```

• Ce este un ETag în HTTP și cum funcționează ?

Antetul de răspuns HTTP ETag este un identificator pentru o versiune specifică a unei resurse. Acesta permite la cache să fie mai eficient și să economisească lățimea de bandă, deoarece un server web nu trebuie să retrimită un răspuns complet dacă conținutul nu s-a schimbat. În plus, etagurile ajută la prevenirea actualizărilor simultane ale unei resurse și evita suprascrierea reciproca ("mid-air collisions").

• Diferența dintre protocoalele fără stare și cele cu stare. Cărui tip îi aparține HTTP ?

HTTP este un protocol cu stare.

Protocoalele cu stare se mai numesc si link-state routing algorithm, sunt protocoale cu preferarea drumului minim (SPF shortest path first),menţin o bază de date complexă a topologiei reţelei. Spre deosebire deprotocoalele fara stare, ele folosind starea legăturilor dezvoltă şi întreţin o cunoaştere completă a routerelor de reţea, ca şi a felului cum sunt interconectate acestea.

• Avantajele cheie ale HTTP/2 în comparație cu HTTP/1.1

HTTP/2 crește considerabil viteza de încărcare a site-urilor pe desktop și pe telefoanele mobile, reduce consumul de trafic internet, oferă securitate sporită și alte facilități.

• Ce este un tip MIME, din ce constă și pentru ce se folosește ?

MIME este un stardat care indica natura si formatul documentului, fisierului sau colectiei de biti. Este definit si standardizat in IETF's RFC 6838.

• Care este diferența dintre GET și POST ?

1. **Folosind GET, informatia transmisa de la utilizator este incarcata in url**, pe **cand cea de la POST nu este in url**, ea este transmisa prin conexiunea curenta http. Astfel, un url la care a fost transmisa o cerere POST nu se poate salva pentru o accesare ulterioara.

2. Folosind GET, **lungimea datelor primite de la utilizator este limitata** la aproximativ 255 de caractere, lungimea maxima a unui URL. Astfel, formulare care accepta mesaje, sau incarcare de fisiere, sau informatie mai lunga, NU se pot transmite prin GET.

3. Datorita faptului ca informatiile de la utilizator se transmit prin url in cazul metodei GET, **nu putem transmite prin GET informatiile dintr-un formular care are parola**, pentru ca, parola va fi vizibila in URL. Decat daca codam parola respectiva.

• Care este diferența dintre PUT și POST ?

1. Metoda PUT este apelată atunci când trebuie să modificați o singură resursă, în timp ce metoda POST este apelată atunci când trebuie să adăugați o resursă copil.
2. Răspunsul metodei PUT poate fi memorat în cache, dar nu puteți memora în cache răspunsurile metodei PUT.
3. Puteți utiliza interogarea UPDATE în PUT, în timp ce puteți utiliza crearea interogării în POST.
4. În metoda PUT, clientul decide ce resursă URI trebuie să aibă, iar în metoda POST, serverul decide ce resursă URI trebuie să aibă.
5. PUT funcționează specific, în timp ce POST funcționează abstract.
6. Dacă trimiteți aceeași solicitare PUT de mai multe ori, rezultatul va rămâne același, dar dacă trimiteți aceeași cerere POST de mai multe ori, veți primi rezultate diferite.
7. Metoda PUT este idempotentă (neschimbata), în timp ce metoda POST nu este idempotentă (neschimbate).

• Care sunt metodele idempotente în HTTP și care sunt scopul lor.

Metodele idempotente în HTTP sunt: OPTIONS, GET, HEAD, PUT, DELETE.

Mai multe cereri identice vor conduce la returnarea aceluiași răspuns (aceeași reprezentare)

• Cum sunt identificate resursele în protocolul HTTP ?

<protocol>://<nume_DNS>/<nume_local>

- protocol - este protocolul folosit (de cele mai multe ori http),
- nume_DNS - este numele domeniului pe care se află resursa,
- nume_local - este format din calea și numele resursei de pe discul local.

• Care sunt metodele sigure și nesigure în HTTP ?

Sigure	Nesigure
Get	**Post**
Head	**Put**
	Delete

• Pentru ce este nevoie de cURL ?

Pentru a efectua HTTP request-uri din consola, este de asemenea utilizat in ecipamente pentru automobile, rutere, televizoare …

• Pentru ce este nevoie de HTTP Proxy?

Pentru a crea o punte intre client si server, astfel daca vom face apel catre un site, apelul v-a fi facut din numele serverului Proxy.

• Diferența dintre autentificare și autorizare ?

Principala diferență între autentificare și autorizare este că autentificarea este procesul de verificare a detaliilor unui utilizator pentru identificarea acestuia și acordarea accesului la sistem, în timp ce autorizarea este procesul de verificare a privilegiilor sau permisiunilor utilizatorului autentificat pentru a accesa resursele sistemului.

• Care sunt metodele de autentificare HTTP ?

Autentificare simpla(Basic authentification), HTTP SSL Certificate Mapping, autentificare prin proxy, HTTP NTLMSSP Authentification, HTTP GSS-API/SSPI authentification using SPENGO and Kerberos

• Modalități de identificare a utilizatorilor în HTTP

HTTP request headers

IP address

Long URLs

Cookies

Login information (authentication)

• HTTP cookies – pentru ce se folosește ?

Cu ajutorul cookies serverul reține faptul că utilizatorul s-a autentificat, și îi va permite acțiuni specifice celor autentificați, de asemena cookies ajuta la setarea preferintelor utilizatorului si pastrarea acestora dupa finisarea sesiunii

Proiectul pe github: https://github.com/sasa-bonta/PR