LUCRAREA NR. 2
PROTOCOALE DE NIVEL APLICAȚIE UTILIZATE ÎN INTERNET

1. Obiective

Obiectivele acestui labroator sînt: familiarizarea cu protocoalele cele mai utilizate de nivel aplicație: FTP, TELNET, Finger, înțelegerea mecanismelor specifice de funcționare a acestor protocoale, familiarizarea cu diferitele implementări ale acestor protocoale.

2. Considerații teoretice

Cele mai cunoscute şi utilizate protocoale de nivel aplicatie sunt Telnet – oferă acces interactiv la un terminal aflat la distanță, respectiv FTP - pentru transferuri de fişiere.
Caracteristicile protocoalelor de nivel aplicație:

- sunt proiectate bazat pe protocoalele de nivel transport TCP şi UDP, multe protocoale de nivel aplicație folosesc TCP dar există aplicații proiectate folosind protocolul UDP datorită performanței ridicate oferite prin reducerea încărcării datorată stabilirii prealabile a conexiunii;
    - modelul de proiectare este modelul client-server.

2.1. Protocoalele nivelului transport

Protocolul TCP este un protocol orientat pe conexiune, proiectat explicit pentru a asigura un flux sigur de octeți de la un capăt la celălalt al conexiunii într-o inter-rețea nesigură. O inter-rețea diferă de o rețea propriu-zisă prin faptul că anumite părți ale sale pot diferi substanțial în *topologie, lățime de bandă, întârzieri* sau *dimensiunea pachetelor*. Implementările existente sunt bazate pe mecanismele de tip fereastră glisantă.

Datagrama IP

Antet IP	Antet TCP	Date TCP

Figura 2.1 *Încapsularea TCP într-o datagramă IP*

Structura antetului TCP este descrisă în continuare. Câmpurile **Port sursă** şi **Port destinație** identifică punctele finale ale conexiunii. Câmpurile **Număr de secvență** şi **Număr de confirmare** au semnificația funcțiilor lor uzuale. Trebuie notat că cel din urmă indică octetul următor aşteptat şi nu ultimul octet recepționat în mod corect. **Lungimea antetului TCP** indică numărul de cuvinte de 32 de biți care sunt conținute în antetul TCP. Această opțiune este importantă deoarece câmpul **Opțiuni** este de lungime variabilă.

Urmează 6 indicatori de câte un bit astfel: URG reprezintă urgent, şi este setat dacă **Indicatorul Urgent** este valid. Bitul ACK este 1 dacă câmpul de confirmare este valid. Bitul PSH îi spune receptorului să livreze aplicației informația, fără să o mai memoreze dacă este 1. Bitul RST este folosit pentru a desființa o conexiune care a devenit inutilizabilă. Bitul SYN este folosit pentru stabilirea unei conexiuni. El poate indica o cerere de conexiune dacă bitul ACK este 0 şi o conexiune acceptată dacă bitul ACK este 1. Bitul FIN este folosit pentru a termina o conexiune. Tehnica de achitare folosită este **piggybacking** pentru transmisia din direcția opusă, reprezentând numărul de secvență al primului octet pe care emițătorul se aşteaptă să-l primească.

32 de biți

<table>
<tr><td colspan="8">Port sursă</td><td>Port destinație</td></tr>
<tr><td colspan="9">Număr de secvență</td></tr>
<tr><td colspan="9">Număr de confirmare</td></tr>
<tr><td>Lung. antetului TCP</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="2">Dimensiunea ferestrei</td></tr>
<tr><td colspan="8">Sumă de control</td><td>Indicator urgent</td></tr>
<tr><td colspan="9">Opțiuni (unul sau mai multe cuvinte pe 32 de biți)</td></tr>
<tr><td colspan="9">Date (opțional)</td></tr>
</table>

Figura 2.2 *Structura unui pachet TCP*

Câmpul **Dimensiune fereastră** indică numărul de octeți care pot fi trimişi, începînd cu octetul confirmat. Dimensiunea maximă a ferestrei de recepție depinde de implementare, iar dimensiunea optimă în cursul transmisiei este subiectul unor algoritmi. Urmează un **Câmp de control** aplicabil antetului.

În momentul calculului, **Suma de control** este poziționată pe zero, iar câmpul de date este completat cu un octet suplimentar nul dacă lungimea lui este un număr impar. Aceasta este calculată după algoritmul de la protocolul IP şi se aplică tuturor câmpurilor unui pachet TCP (antet şi date), inclusiv asupra unor câmpuri speciale ce alcătuiesc un pseudo-antet (*TCP pseudo header*). Aceste câmpuri livrate de protocolul IP către TCP alături de segmentul de date, au ca rol să prevină o incorectă dirijare IP.

Câmpul **Opțiuni** a fost proiectat pentru a include o serie de facilități suplimentare neacoperite de antetul obişnuit. După aceea urmează **Datele**, care sunt opționale.

Câmpurile **Port sursa** şi **Port destinatie**, fiecare de câte 16 biți, reprezintă numerele de porturi TCP pentru sursa şi respectiv destinația transmisiei. Ele se asociază punctelor de capăt ale comunicației (*communication end points*). Dintre aceste numere de port, unele sunt asociate (prealocate) unor aplicații comune, precum **telnet**, **ftp.**

Nume serviciu	**Număr port**
Echo	7 (tcp, udp)
Discard	9
Daytime	13 (udp)
ftp	21 (tcp)
telnet	23 (tcp)
Smtp	25 (tcp)
Timed	37 (tcp, udp)
http	80 (tcp)
nntp	119 (tcp)
Login	513
Shell	514

Protocolul TCP, având o complexitate ridicată, are nevoie pentru implementarea operațiilor de control, de diverşi algoritmi specifici şi ceasuri de control (*timers*). Algoritmii utilizați în diferite implementări pentru mărirea eficienței, cum ar fi: prevenirea transmiterii de pachete TCP de mici dimensiuni, eliberarea 'grăbită' a ferestrei de recepție, start întârziat pentru evitarea congestiei.

Câteva specificații de timeout folosite sunt:

- perioada retransmisiei (*retransmission timeout*), perioadă de timp între emiterea unui pachet TCP şi receptarea achitării sale

- interval pentru asigurarea persistenței conexiunii TCP (*persistence timer*).

Protocolul UDP este un protocol fără conexiune, proiectat pentru a oferi aplicațiilor o modalitate de a trimite datagrame IP neprelucrate, încapsulate şi pe care le transmite fără a stabili in prealabil o conexiune. Multe aplicații *client-server* în care o parte emite cereri, iar cealaltă emite răspunsuri preferă să trimită datagrame UDP decât să stabilească şi apoi să elibereze o conexiune. Antetul UDP este prezentat în figura următoare:

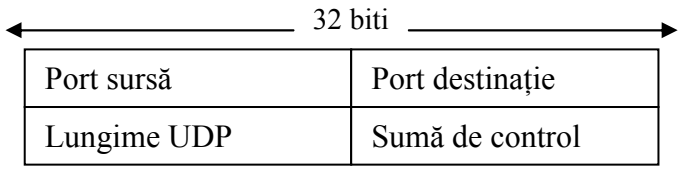

Figura 2.3 *Structura unui pachet UDP*

Un segment UDP constă dintr-un antet de 8 octeți urmat de date, porturile au aceeaşi semnificație ca şi în cazul protocolului TCP, indicînd punctele finale ale conexiunii. Câmpul **Lungime UDP** include cei 8 octeți ai antetului plus datele, iar câmpul **Sumă control UDP** reprezintă suma de control aplicată întregului segment cu condiția că se aplică unui număr par de octeți.

2.2 Exemple de protocoale de nivel aplicație

2.2.1 Transferul de fişiere

Aplicația de transfer de fişiere este una dintre cele mai comune (alături de transmisia mesajelor prin poşta electronică) oferind serviciile cele mai solicitate de marea masă a utilizatorilor de rețele. Principala implementare existentă, **FTP** (File Transfer Protocol, descris de RFC 959), este una dintre cele mai vechi, dar răspunde încă tuturor cerințelor ivite pentru transferul fişierelor între calculatoare. Complexitatea protocolului **FTP** constă în modalitățile multiple oferite pentru manipularea fişierelor şi a structurilor de fişiere, lucrul cu fişiere cu structuri diferite (ASCII, binare, comprimate, necomprimate) şi a conversiei necesare. Protocolul oferă şi elemente de securitate, prin necesitatea cunoaşterii de către utilizator a unui cont şi parole valide pe calculatorul 'depărtat'.

O variantă simplificată, folosită de utilizatori mai ales pentru descărcarea aplicațiilor software (*downline loading*), este **TFTP** (*Trivial FTP,* RFC 783). El este mai rapid, folosind ca mecanism suport de transport datagrame UDP şi implementează doar un subset dintre serviciile FTP (nu este posibilă de exemplu parcurgerea ierarhiei de directoare a calculatorului la distanță).

2.2.1.1 FTP

FTP- protocolul utilizat pentru transfer de fişiere (upload-download) în Internet, se bazează pe modelul de comunicație client-server şi foloseşte portul 21. O sesiune FTP este stabilită şi este menținută până când clientul încheie transmisia sau până la momentul la care apar erori de comunicație. Stabilirea conexiunii cu serverul (un demon FTP care rulează în server) presupune transmiterea unui login şi a unei parole. Tipul de conexiune cunoscută sub numele de *FTP anonymous*, presupune transmiterea ca login ID a numelui anonymous iar ca şi parolă a propriei adrese de e-mail.

Acest serviciu permite creare şi modificare de directoare, ştergere şi redenumire de fişiere şi alte funcții legate de managementul de fişiere. Transferul de fişiere presupune stabilirea unei conexiuni secundare, sub forma unei legături de date între cele două maşini, iar după terminarea transferului conexiunea se va încheia automat.

`Sintaxa: FTP nume_host`

După afişarea prompterului FTP de conectare pe maşina la distanță (după specificarea numelui şi a parolei de acces) se aşteaptă una din comenzile din listă:

- conectarea la un host remote (`open`- selectează hostul remote şi inițiază sesiunea de login);
- selectarea unui director (`cd` pentru alegerea directorului curent, doar pentru acele directoare pentru care userul are acces);
- listarea fişierelor disponibile transferului `Dir;`
- definirea modului de transfer (mode de transfer bloc, stream de date şi a tipului ASCII, EBCDIC, imagine);
- copiere de fişiere de la şi către hostul *`remote (Get - mget, Put- mput);`*
- terminarea tramsferului şi deconectare (`Quit, Close`*).*

Utilizarea *modului de transfer proxy* – permite clienților care au o conexiune slabă să folosească un alt server (proxy) pentru a efectua transferul.

Comunicația se desfăşoară folosind specificațiile modelului client server, astfel clientul lansează cererea iar serverul răspunde. Lista codurilor de răspuns, în care biții 2,3 permit implementarea detalierii răspunsurilor este următoarea.

Tabelul 2.1 *Coduri de răspuns FTP*

1xx	Raspuns pozitiv preliminar
2xx	Raspuns pozitiv completare
3xx	Raspuns pozitiv intermediar
4xx	Completare de raspuns negativ tranzitoriu
5xx	Completare raspuns negativ permanent

2.2.1.2 TFTP

Trivial File Transfer Protocol (RFC 1350, RFC 2349) este un protocol standard ce permite transferul de fişiere de la un calculator pe care rulează un server TFTP. Protocolul este implementat peste protocolul UDP. Cererea inițială a clientului este lansată pe portul 69 apoi se va determina portul care va fi folosit pentru comunicația efectivă. Protocolul nu oferă suport pentru autentificare utilizator fiind astfel un protcol nesigur.

```
Sintaxa: TFTP – i nume_host GET|PUT sursa destinatia
```

`i` indică tip de transfer
`nume_host` reprezintă numele maşinii remote
`GET- PUT` fişier remote-local (determină direcția transferului)
`Sursa-Destinația` – reprezintă numele complet al fişierului de transferat, respectiv destinațiai indicates the type of the transfer;

Protocolul TFTP poate implementa şi o opțiune multicast care permite mai multor clienți să preia simultan fişiere de la un server folosind pachete tip multicast. Această opțiune este folosită de maşini identice care doresc să descarce simultan aceeaşi configurație de pe un server TFTP. Deoarece protocolul nu posedă un mecanism de autentificare, serverul trebuie să

restricționeze accesul în scriere pentru directoarele publice, în acest sens există implementări care oferă liste de hosturi care au acces la un anumit server TFTP.

2.2.2 Accesul la distanță. Protocoalele TELNET şi Rlogin

Accesul la distanță la un calculator legat la rețea (localizat oriunde), se realizează sub forma unei sesiuni de la terminal (pentru această acțiune se mai întâlnesc terminologiile remote login şi terminal virtual), protocolul cel mai cunoscut pentru această operație fiind Telnet (RFC 854).

Cele două aplicații utilizate pentru conectarea la distanță folosind protocoalele TCP-IP sunt: *Telnet (Terminal emulation)* protocolul care a fost proiectat pentru a permite conectivitate între maşini care utilizează sisteme de operare diferite (sisteme heterogene utilizînd orice tip de terminal) şi *Rlogin* protocolul dezvoltat pentru a oferi funcționalitate de conectare la distanță doar maşinilor Unix.

2.2.2.1 Telnet

Telnet este una din aplicațiile de conectare la distanță cele mai populare, numită aplicatia de login la distanță a Internetului bazată pe paradigma client-server. Funcționalitatea minimală a acestui protocol implementează noțiunea de terminal de rețea virtual, oferind suport pentru negocierea prealabilă a conexiunii. Aplicația telnet permite clientului să ruleze programe pe calculatorul la distanță, astfel ea transmite caracterele preluate de la tastatură şi întoarce rezultatele rulării programului pe maşina la distanță.

O sesiune telnet implică deschiderea unei conexiuni între terminal (sau maşina locală) şi calculatorul depărtat, legat la rețea; de aceea numele maşinii la distanță trebuie să fie cunoscut. Conexiunea se desface de către utilizator şi se revine în contextul maşinii locale. Orice implementare telnet trebuie să cuprindă emulatoare pentru terminalele cele mai diverse (IBM 3270, Digital VT100, etc.).

`Sintaxa: Telnet nume_host`

`Nume_host` – Nume_host- reprezintă numele hostului la distanță pe care rulează serverul (demonul) telnet, iar tipul de terminal reprezintă tipul terminalului care va fi emulat pe maşina la distanță.
Elementele pe care se bazează stabilirea unei conexiuni folosind protocolul Telnet sunt: conceptul de terminal virtual – dispozitiv generic având structura de bază corepunzătoare diverselor implementări de terminale reale, o viziune simetrică a terminalelor şi proceselor, negocierea opțiunilor terminal.
Terminalul virtual conține un printer (display) şi o tastatură pentru a permite transferul şi recepționarea de date în format ASCII, operînd în mod semiduplex la nivel de linie, bazat pe o funcție de echo local.(oferă încărcare mai redusă dacă există şi o funcție de echo la distanță)

Comenzile interne neaccesibile utilizatorilor au următoarea structură.

Cod interp.	Cod comandă	Opțiuni negociate

Cele mai importante comenzi sunt:

- `close` – terminare conexiune curentă;
- `open name` – permite stabilirea unei conexiuni cu maşina specificată;
- *`display`* – afişare parametri de operare;
- *`mode`* – stabilire mod de transfer (mod linie sau caracter);
- *`quit`* – ieşire mod telnet;
- *`send`* – transmite caractere;
- *set* – setare parametri de operare;
- *`status`* – afişare informații de stare.

Tabelul 2.2 *Tipuri de opțiuni Telnet*

Comanda	**Cod**	**Comentariu**
SE	240	Sfârsit de subnegociere
NOP	241	Nici o operație
SB	250	Indică subnegocierea unei opțiuni următoare
WILL	251	Indică confirmarea (cererea) de utilizare
Wont	252	Indică refuzul de a utiliza (sau de a continua)
DO	253	Cerere de utilizare (sau confirmare a utilizării)
DON'T	254	Cerere de oprire a utilizării către partea conexă

Comanda	Cod	Comentariu
Break	243	Caracterul break in NVT
IAC	255	Interpretată ca şi comandă

Operația de negociere a opțiunilor este simetrică, fiecare capăt de comunicație poate transmite una din următoarele cereri pentru o anumită opțiune:

- `will` – emițătorul doreşte validarea opțiunii;
- `do` – emițătorul doreşte ca receptorul să valideze opțiunea;
- `wont` – emițătorul doreşte invalidarea opțiunii;
- `don't` – emițătorul doreşte ca receptorul să invalideze opțiunea.

Scopul protocolului Telnet îl constituie crearea unei interfețe standard pentru hosturile care comunică în rețea. Un server de telnet există doar pentru maşini Unix, pe acestea pot rula programele şi comenzile Unix cum ar fi mc, pine, ping, Lynx. Procesarea în cazul aplicației Telnet se desfăşoară pe maşina la distanță. La nivel prezentare (ASCII), nivelul sesiune (pentru transmitere) apoi la nivel transport datele sunt segmentate şi se va verifica adresa de port şi corectitudinea lor. La nivel rețea se vor adăuga antetele cu adresele de IP sursă şi destinație, iar dacă sursa cererii nu posedă o adresă fizică va genera o cerere de rezolvare adresa de tip arp.

Tabelul 2.3 *Coduri de opțiuni Telnet*

1	Echo
3	Suppress go ahead
5	Status
6	Timing mark
24	Terminal type
31	Window size
31	Terminal speed
33	Remote flux control
34	Timing mark
36	Line mode
37	Environment variables

Maşina remote va executa comanda şi va folosi aceleaşi mecanisme pentru a transmite răspunsul, astfel intregul proces se va repeta până ce clientul încheie sesiunea la distanță. Procesoarele Berkeley UNIX încorporează comanda rlogin, cu o funcționalitate analogă.

2.2.2.2 Protocolul Finger

```
Sintaxa: FINGER user@numehost(sau finger numehost)
```

Comanda permite afişarea informaţiilor despre useri pentru orice sistem care rulează un serviciu standard de tip finger. Opțiunea *user* este facultativă şi returnează lista userilor sistemului identificat prin numehost. *Numehost* reprezintă numele maşinii gazdă pe care se doreşte căutarea, dacă această comandă este utilizată cu un host necooperativ (pe care nu rulează demonul fingerd) solicitarea va fi refuzată. Serviciul finger este inclus şi în clienții Windows NT, astfel se pot obține informații specifice oricărui sistem care rulează un serviciu finger standard. Serverul NT nu include un serviciu de tip finger (acesta este disponibil public la Centrul European Academic Windows NT la adresa www.emwac.ed.ac.uk)

3. Desfăşurarea lucrării

3.1. Să se testeze protocoalele de nivel aplicație descrise in lucrare folosind facilitățile de documentare on line.
3.2. Să se studieze diferențele între telnet şi rlogin. Testați o sesiune de conectare Telnet în modul de lucru caracter, respectiv linie.
3.3. Realizați transferuri diferite de la diferite adrese folosind diferiți clienti (Linux, Windows).

Notițe