

Traducere de Raluca Lacatusu – raluca@infoiasi.ro  
Network Working Group  
Request For Comments: 1350  
STD: 33  
Obsoletes: RFC 783

K. Sollins  
MIT  
Iulie 1992

## PROTOCOLUL TFTP

### Statut

Acest RFC specifică standardele IAB protocol pentru Internet si solicită discuții și sugestii pentru îmbunătățire. Vă puteți referi la ediția curentă Standarde Oficiale pentru Protocol IAB pentru statutul acestui protocol. Distribuția este nelimitată.

### Sumar

TFTP este un protocol de transfer fișiere foarte simplu, de aici și numele său – Protocol pentru Transferul Trivial al Fișierelor – Trivial File Transfer Protocol, pe scurt TFTP. Fiecare pachet nonterminal este confirmat separat. Acest document descrie protocolul TFTP și tipurile sale de pachete. De asemenea sunt luate în discuție motivele care au stat la baza deciziilor de proiectare.

### Recunoștințe

Protocolul a fost inițial proiectat de Noel Chiappa, refăcut după aceea împreună cu Bob Baldwin și Dave Clark, cu comentarii de Steve Szymanski. Ultima revizie aduce modificări și contribuții pe marginea discuțiilor și sugestiilor date de Larry Allen, Noel Chiappa, Dave Clark, Geoff Cooper, Mike Greenwald, Liza Martin, David Reed, Craig Milo Rogers (of USC-ISI), Kathy Yellick și ale autorului. Modurile de confirmare și schema de transmisie au fost inspirate din modelul TCP(Transmission Control Protocol) și mecanismul de eroare a fost sugerat de modelul EFTP al lui PARC.

Revizia din mai 1992 pentru fixarea bug-ului "Sorcerer's Apprentice" din protocol și alte probleme minore a fost făcută de Noel Chiappa.

Cercetarea a fost sprijinită de Agenția Proiecte de Cercetare Avansată a Departamentului de Aparare și a fost monitorizată de Oficiul de Cercetare Navală cu număr de contract N00014-75-C-0661.

### 1. Scop

TFTP este un protocol simplu de transfer al fișierelor și de aceea a fost denumit Protocolul pentru Transferul Trivial al Fișierelor.

TFTP a fost implementat utilizand protocolul de transport UDP Internet User Datagram Protocol (UDP sau Datagram) pentru a putea fi folosit la transferul fișierelor între mașini aflate pe rețele diferite ce implementează UDP. (Aceasta nu exclude posibilitatea ca TFTP să fie bazat și pe alte tipuri de protocoale datagramă.) Este proiectat pentru a putea fi ușor de implementat. Tocmai de aceea îi vor lipsi majoritatea caracteristicilor unui FTP. Singurul lucru pe care îl face este să scrie și să citească fișiere (sau mesaje) de la/spre un server de la distanță. Nu poate lista directoare, și momentan nu are facilități pentru autentificarea unui utilizator. În comun cu celelalte protocoale Internet este transmisia a 8 biți de date.

Momentan sunt suportate 3 moduri de transfer: **netascii**(ASCII așa cum e definit în "USA Standard Code for Information Interchange" cu modificările din "Telnet Protocol Specification".) Specificăm că e vorba de un cod pe 8 biți. Termenul "netascii" va fi folosit pe tot parcursul acestui document înțelegând astfel o formă particulară de cod ascii; **octet** (acest mod de transfer înlocuiește modul "binar" al versiunilor anterioare ale acestui document) șir de 8 biți; **mail**, caractere netascii trimise unui utilizator (modul mail este învechit și nu ar trebui să fie implementat sau folosit). Alte moduri adiționale pot fi definite de perechi de host-uri care comunica între ele.

Secțiunea 4.2 ar trebui consultată pentru mai multe sugestii de spre TFTP.

## 2. Privire de ansamblu asupra Protocolului

Orice transfer începe cu o cerere de scriere sau citire a unui fișier, operație care solicită o conexiune. Dacă serverul primește cererea, se stabilește conexiunea și fișierul este transmis în blocuri de lungime fixă de 512 octeți. Fiecare pachet de date conține un bloc de date și trebuie să fie recunoscut de un pachet de confirmare înainte ca următorul să poată fi transmis. Un pachet de date cu mai puțin de 512 de octeți semnalizează sfârșitul transferului. Dacă un pachet se pierde în rețea, receptorul va termina și va putea retransmite ultimul său pachet primit (care ar putea fi pachet de date sau de confirmare), determinându-l pe transmițătorul pachetului pierdut să îl retransmită. Ambele mașini implicate astfel în transfer sunt considerate ca receptori și transmițători. Una transmite date și primește confirmari, cealaltă transmite confirmari și primește date.

În mare parte erorile vor cauza întreruperea conexiunii. O eroare se semnalează prin trimiterea unui pachet eroare. Acest pachet nu va fi nici confirmat, nici retransmis (de exemplu un server TFTP sau utilizator pot termina după primirea unui mesaj eroare) așa încât la celălalt capăt al conexiunii nu mai ajunge. Astfel un time-out poate fi folosit pentru a detecta o terminare când pachetul eroare a fost pierdut.

Erorile sunt cauzate de 3 tipuri de evenimente: nu a fost satisfăcută cererea (de exemplu fișierul nu a fost găsit, neexistența utilizatorului sau violarea accesului),

primirea unui pachet care nu poate fi argumentat printr-o întârziere sau duplicare în rețea (de exemplu un pachet în formă incorectă), și pierderea accesului la o resursă necesară (de exemplu disk full sau acces nepermis în timpul unui transfer).

TFTP recunoaște doar o singură condiție de eroare care să nu cauzeze terminarea transferului, când portul sursă al unui pachet trimis este incorect. În acest caz, pachetul eroare este transmis la host-ul care l-a generat.

Acest protocol este foarte restrictiv pentru a simplifica implementarea. De exemplu lungimea fixă a blocurilor fac alocarea într-un mod simplu iar pasul de confirmare furnizează control continuu și elimină nevoia reordonării pachetelor de date care vin.

### 3. Relația cu alte Protocoale

Cum am menționat mai sus, TFTP este proiectat pentru a fi implementat pe modelul UDP. Și pentru că protocolul Datagram este implementat pe Internet Protocol, pachetele vor avea un header Internet, un header Datagramă, și un header TFTP. În plus, pachetele pot avea un alt header (LNI, header ARPA, etc.) pentru a permite accesul prin transportul medium local.

Cum arată și figura 3.1, ordinea conținuturilor pachetului va fi: header mediu local, dacă e folosit, header Internet, header Datagramă, header TFTP, urmat de restul rămas din pachetul TFTP (acesta poate reprezenta sau nu date dependente de tipul pachetului așa cum e specificat în headerul TFTP). TFTP nu specifică nici un fel de valori în headerul Internet. Pe de altă parte, câmpurile porturilor sursă și destinație ale header-ului Datagramă (formatul său este dat în appendix) sunt folosite de TFTP și lungimea câmpului reflectă mărimea unui pachet TFTP. Identificatorii Transferului (TID) folosiți de TFTP sunt preluați de layer-ul Datagramă pentru a fi folosiți ca porturi; tocmai de aceea ar trebui să se încadreze între 0 și 65,535. Inițializarea TID-urilor este discutată în secțiunea următoare.

Header-ul TFTP conține un câmp cu 2 octeți cod (opcodes) care indică tipul pachetului (de exemplu DATA, ERROR, etc.). Acești cod operatori și formatul diferitelor tipuri de pachete sunt discutate mai încolo în secțiunea pachetelor TFTP.

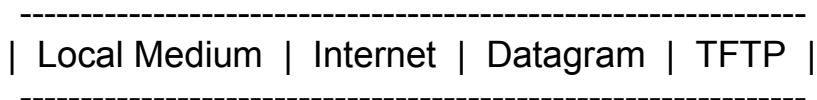


Figura 3-1 : Ordinea Header-elor

### 4. Protocolul Initial de Conectare

Stabilirea unui transfer ce face prin trimiterea unei cereri ( WRQ pentru a scrie într-un sistem de fișiere străin, sau RRQ pentru a citi din el) și primirea unui răspuns pozitiv - un pachet de confirmare pentru scriere, sau primul pachet

de date pentru citire. În general, un pachet de confirmare va conține numărul de bloc al pachetului care a fost recunoscut. Fiecare pachet de date are asociat un număr de bloc; numerele blocurilor sunt consecutive și încep cu 1. Dacă răspunsul pozitiv la o cerere de scriere este un pachet de confirmare, în acest caz special, blocul va fi numerotat cu 0. (În mod normal, dacă un pachet de confirmare recunoaște un pachet de date, atunci va conține și numărul de bloc al acestuia.) Dacă răspunsul este un pachet eroare, atunci cererea a fost respinsă.

Pentru a crea conexiunea, fiecare capăt al conexiunii alege un TID care va fi folosit pe toata durata ei. TID-urile ar trebui alese aleatoriu, pentru ca probabilitatea ca un număr să fie ales de două ori succesiv să fie foarte mică. Fiecare pachet are asociat două TID-uri la capetele conexiunii, TID-ul sursă și TID-ul destinație. Aceste TID-uri sunt transmise către UDP (sau alt protocol datagramă) ca porturi sursă, respectiv destinație. Un host client își alege TID-ul sursă cum am descris mai sus, și transmite cererea inițială TID-ului 69 zecimal (105 octal) de la host-ul server. Răspunsul la cerere, dacă are loc ca o operație obișnuită, folosește un TID ales de server ca TID-ul sursă și TID-ul ales pentru mesajul anterior de către client ca TID-ul destinație. Cele două TID-uri sunt folosite apoi pentru restul transferului.

Următorul exemplu ne arată pașii care trebuie parcurși pentru stabilirea unei conexiuni de scriere a unui fișier. Rețineți că WRQ, ACK și DATA sunt numele cererii de scriere, recunoașterii și respectiv a tipului de dată a pachetelor. Appendix-ul conține un exemplu similar de citire a unui fișier.

1. Host-ul A trimite un WRQ host-ului B cu sursa=TID A, destinația =69.
2. Host-ul B trimite un ACK (cu numărul de bloc 0) host-ului A cu sursa= TID B, destinația= TID A.

Din acest moment conexiunea s-a stabilit și primul pachet de date poate fi trimis de Host A cu o secvență cu numărul 1. În următoarea etapă și în toate celelalte etape, host-urile trebuie să se asigure că sursa TID se potrivește cu valoarea acceptată în etapa 1 și 2. Dacă o sursă TID nu se potrivește, pachetul ar trebui respins, considerat ca trimis eronat din altă parte. Un pachet eroare ar trebui trimis la sursa pachetului incorect, fără a stânjeni transferul. Aceasta se poate realiza numai dacă TFTP-ul primește de fapt un pachet cu un TID incorect. Dacă protocoalele ajutoare nu îl acceptă, această eroare particulară nu se va arăta.

Următorul exemplu ilustrează o operație corectă a protocolului în care poate surveni situația de mai sus. Host-ul A trimite un pachet Host-ului B. Undeva în rețea, pachetul trimis este duplicat, și ca rezultat două confirmări sunt returnate la Host-ul A, cu TID-uri diferite alese de la host-ul B fiind răspunsul la cele două cereri. Când primul răspuns sosește, host-ul A menține conexiunea. Când al doilea răspuns al cererii ajunge, ar trebui respins, dar nu există nici un

motiv pentru terminarea primei conexiuni. De aceea, dacă pentru cele două conexiuni sunt alese TID-uri diferite la Host-ul B iar Host-ul A verifică sursa TID a mesajului primit, prima conexiune va putea fi menținută în timp ce a doua este respinsă la returnarea unor pachete eroare.

## 5. Pachete TFTP

TFTP suportă 5 tipuri de pachete, fiecare din ele fiind menționat mai sus.

cod	operație
1	Read Request(RRQ) - Cererea de citire
2	Write Request(WRQ) - Cererea de scriere
3	Data(Data)
4	Acknowledgment (ACK) - Confirmare
5	Error(ERROR) - Eroare

Header - ul TFTP a unui pachet conține codul asociat acestuia.

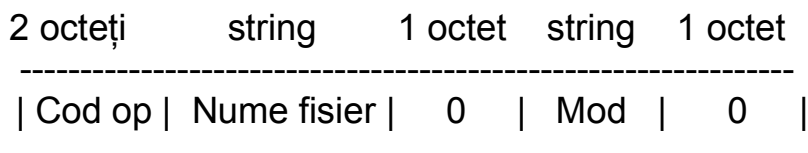


fig 5.1 : Pachetul RRQ/WRQ

Pachetele RRQ si WRQ (cu codul op 1 respectiv 2) au formatul ca în fig 5.1. Numele fișierului este o secvență de octeți în netascii care se termină într-un octet 0. Câmpul mod conține șirul "netascii", "octet", sau "mail" (sau orice altă combinație de minuscule și majuscule, precum "NETASCII", NetAscii",etc.) indicând cele trei moduri definite in protocol. Un host care primește modul de data netascii trebuie să traducă datele în formatul său propriu. Modul octet este folosit pentru transferul unui fișier care are formatul 8-bit al mașinii care îl transferă. Se consideră că fiecare tip de mașină are un singur format 8-bit care este mai comun, și că acest format va fi ales. De exemplu, pe o mașină DEC-20, pe 36 biți, acesta înseamnă patru 8 biți de date într-un cuvânt

Dacă un host primește un fișier octet și apoi îl returnează, fișierul returnat trebuie să fie identic cu originalul. Modul mail folosește numele recipientului mail în locul unui fișier și trebuie să înceapă cu o cerere de scriere WRQ. Altfel ar fi identic cu modul netascii. Șirul unui recipient mail ar trebui să fie de forma "nume\_utilizator" sau "nume\_utilizator@nume\_host". Dacă se utilizează a doua formă, permite opțiunea mail ca forward de la un computer de retransmisie.

Situația de mai sus presupune că și receptorul și transmițătorul operează în același mod, dar nu e motiv pentru a crede că este cazul. De exemplu s-ar putea construi un server stoc. Nu e nici un motiv să credem că o asemenea mașină necesită interpretarea netascii în propriul său format. Mai degrabă,

furnizorul ar putea transmite fişiere în netascii, dar serverul stoc ar putea mai bine să le stocheze fără interpretare în formatul 8-bit. O altă situaţie similară este o problemă care există momentan în sistemele DEC-20. Nici modul netascii, nici cel octet nu accesează toţi biţii într-un cuvânt. S-ar putea crea un mod special pentru o astfel de maşină care citeşte toţi biţii într-un cuvânt, dar în care receptorul a stocat informaţia în formatul 8-bit. Când un astfel de fişier este refăcut din stocare, trebuie să fie adus la forma iniţială pentru a putea fi folosit, deci modul reversibil trebuie să fie de asemenea implementat.

Utilizator va trebui să-şi amintească ceva informaţii pentru a realiza aceasta. În ambele exemple, pachetele la cerere ar specifica modul octet host-ului străin , dar cel local ar fi în alt mod. Nici o astfel de maşină sau aplicaţie de specificare a modului nu au fost specificate în TFTP, dar un ar fi compatibilă cu această specificare.

Este posibil să definim alte moduri pentru cooperarea între perechi de host-uri, deşi ar trebui făcute cu grijă. Nu e nici o necesitate pentru celelalte host-uri să implementeze acestea. Nu e nici o autoritate centrală care să definească aceste moduri sau să le atribuie nume.

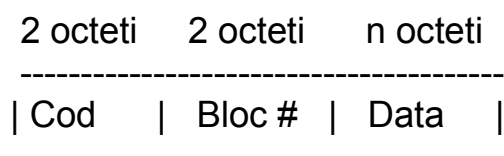


Figura 5.2: pachet DATA

Datele sunt de fapt transferate în pachete de date infatisate deja în figura 5.2. Pachetele de date (cu codul 3) au un număr de bloc și un câmp data. Numerele blocurilor din pachetele de date încep cu 1 și cresc cu 1 pentru fiecare bloc de date nou. Această restricție permite programului să folosească un singur număr pentru a face diferența între pachete noi și cele duplicate.

Câmpul dată este de la 0 la 512 octeți. Dacă este 512 octeți, blocul nu este ultimul bloc de date; dacă este de la 0 la 511 octeți, semnalizează sfârșitul transferului. (a se vedea secțiunea Terminare Normală pentru detalii.)

Toate pachetele cu excepția celor ACK duplicate și cele folosite pentru terminare sunt confirmate înaintea timpului final. Trimiterea unui pachet DATA este o confirmare pentru primul ACK al pachetului DATA anterior. Pachetele WRQ și DATA sunt confirmate de către pachetele ACK și ERROR, în timp ce

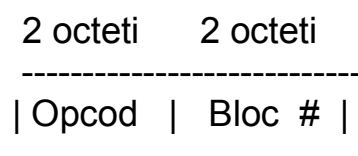


Figura 5-3: pachetul ACK

pachetele RRQ si ACK sunt confirmate de pachetele DATA si ERROR. Figura 5.3 infatiseaza un pachet ACK; codul op este 4. Numarul bloc intr-un pachet ACK repeta numarul bloc al pachetului DATA care a fost confirmat. O WRQ este confirmata cu un pachet ACK avand un numar bloc 0.

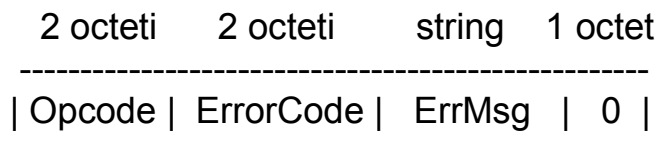


Figura 5.4: pachet EROARE

Un pachet EROARE( cu opcode 5) are forma infatisata in figura 5.4 si poate fi confirmarea oricarui tip de pachet. Codul eroare este un intreg indicand natura erorii. Un tabel de valori este dat si in apendix. (Retineti ca unele coduri erori au fost adaugate acestei versiuni a documentului.) Mesajul eroare este destinat consumului uman, si ar trebui sa fie in netascii. Ca toate celelalte string-uri, se termina intr-un octet 0.

## 5. Terminarea Normala

Sfarsitul unui transfer este marcat de un pachet DATA ce contine intre 0 si 511 octeti de date (de exemplu, lungimea datagrama < 516). Acest pachet este confirmat de catre un pachet ACK ca toate celelalte pachete DATA. Host-ul ce confirma ultimul pachet DATA poate termina partea sa de conexiune prin trimiterea ultimului ACK. Pe de alta parte, asteptarea este incurajata. Aceasta inseamna ca host-ul care trimite ultimul ACK va astepta pentru un timp inainte terminarii pentru a retransmite ultimul ACK daca a fost pierdut. Cel care confirma va sti ca ACK a fost pierdut daca primeste ultimul pachet DATA din nou. Host-ul care trimite ultimul DATA trebuie sa il retransmita pana cand pachetul este confirmat sau host-ul transmitator e la final. Daca raspunsul este ACK, transmiterea a fost completata cu succes. Daca transmitatorul de date este la final si nu e pregatit sa transmita mai mult, transferul inca ar fi putut fi completat cu succes, dupa care reseaua ar putea fi experimentat o problema. Este de asemena posibil in acest caz ca transferul sa fi fost fara succes. In ambele cazuri, conexiunea insa a fost inchisa.

## 6. Terminarea Prematura

Daca o cerere nu poate fi tratata, sau o eroare intervine in timpul transferului, atunci un pachet EROARE (cu cod 5) este trimis. Aceasta doar din "politete" pentru ca nu va fi retransmis sau confirmat, deci ar putea sa nu fie trimis niciodata. Timpurile finale ar putea fi folosite pentru a detecta erori.

## Appendix

Ordinea headerelor:

```
-----
| Local Medium | Internet | Datagram | TFTP |
-----
```

Formatele TFTP

Tip	Op #	Format fără header			
	2 octeți	string	1 octet	string	1 octet
RRQ/WRQ	01-02	Nume fisier	0	Mod	0
	2 octeti	2 octeti	n octeti		
DATA	Cod	Bloc #	Data		
	2 octeti	2 octeti			
ACK	Opcod	Bloc #			
	2 octeti	2 octeti	string	1 octet	
EROARE	Opcode	ErrorCode	ErrMsg	0	

Protocolul Initial de Conectare pentru citirea din fisier

1. Host-ul A trimite un RRQ host-ului B cu sursa = TID A, destinația = 69.
2. Host-ul B trimite DATA (cu numărul de bloc = 1) host-ului A cu sursa= TID B, destinația = TID A.

Coduri Eroare

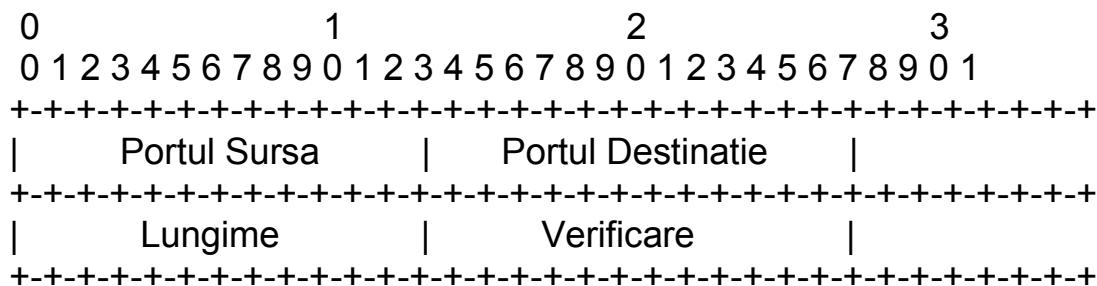
Valoare	Comentariu
0	Nu e definit, vezi mesajul de eroare (daca exista).
1	Fisierul nu a fost gasit.
2	Violarea accesului.

- 3 Disk plin sau alocare depasita.
- 4 Operatie TFTP ilegala.
- 5 ID transferului necunoscut.
- 6 Fisierul exista deja.
- 7 Nu exista acest utilizator.

### Header pentru Datagrama

(acesta a fost inclus din convenienta. TFTP-ul nu e nevoie sa fie implementat folosind protocolul UDP).

### Format



### Valoarea campurilor:

Portul Sursa	selectat de furnizorul pachetului.
Portul Destinatie	selectat de masina destinatie (69 pentru RRQ sau WRQ)
Lungime	Numarul de octeti intr-un pachet UDP, incluzand header UDP

Verificare                      Referinta 2 descrie regulile pentru verificarea calculului.  
(Cel care implementeaza ar trebui sa fie sigur ca algoritmul corect este utilizat aici.)

(Nota: TFTP paseaza identificatorii de transfer (TID-urile) protocolului UDP pentru a putea fi folositi ca port sursa si destinatie.)

### Referinte

- [1] USA Standard Code for Information Interchange, USASI X3.4-1968.
- [2] Postel, J., "User Datagram Protocol," RFC 768, USC/Information Sciences Institute, 28 August 1980.
- [3] Postel, J., "Telnet Protocol Specification," RFC 764, USC/Information Sciences Institute, June, 1980.

[4] Braden, R., Editor, "Requirements for Internet Hosts -- Application and Support", RFC 1123, USC/Information Sciences Institute, October 1989.

#### Consideratii de securitate

Pentru ca TFTP nu include login sau mecanisme de control a accesului, aceasta sarcina cade in drepturile furnizate de serverul TFTP, fara violarea securitatii sistemului de fisiere al serverului. TFTP este deseori instalat cu controale astfel incat doar fisierele care au acces de citire public sunt valabile iar fisierele de scriere nu sunt permise.

#### Adresa autorului:

Karen R. Sollins  
Massachusetts Institute of Technology  
Laboratory for Computer Science  
545 Technology Square  
Cambridge, MA 02139-1986

Phone: (617) 253-6006

EMail: SOLLINS@LCS.MIT.EDU