

INTERNET PROTOCOL - SPECIFICAȚII
Programul DARPA INTERNET

Septembrie 1981

Întocmit pentru

Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia 22209

De către

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, California 90291

CUPRINS

PREFAȚĂ	3
1. INTRODUCERE	4
1.1 Motivație	4
1.2 Scop	4
1.3 Interfețe	4
1.4 Operare	5
2. PRIVIRE DE ANSAMBLU	6
2.1 Legătura cu alte protocoale	6
2.2 Modul de operare	6
2.3 Descrierea funcționalității	7
2.4 Porțile (Gateways)	9
3. SPECIFICAREA PROTOCOLULUI	10
3.1 Formatul antetului internet	10
3.2 Discuție	19
3.3 Interfețe	26
ANEXA A: Exemple și scenarii	28
ANEXA B: Ordinea de transmitere a datelor	32
GLOSAR	33
REFERINȚE	36

PREFAȚĂ

Acest document prezintă standardul DoD Internet Protocol și se bazează pe șase ediții anterioare ale specificațiilor ARPA Internet Protocol, preluând de acolo o mare parte a informațiilor. Există multe persoane care au contribuit la acest document, atât la prezentarea conceptelor cât și la elaborarea textului.

Această ediție revizuieste aspecte referitoare la adresare, tratarea erorilor, coduri de opțiuni și securitate, precedență, comportamente și restricții în cadrul protocolului IP.

Jon Postel
Editor

RFC: 791

Înlocuiește: RFC 760

IEN 128, 123, 111,

80, 54, 44, 41, 28, 26

PROTOCOLUL INTERNET PROGRAMUL DARPA INTERNET SPECIFICAREA PROTOCOLULUI

1. INTRODUCERE

1.1. Motivație

Protocolul IP este destinat folosirii în sisteme interconectate de rețele de calculatoare ce comunică prin schimburi de pachete. Un astfel de sistem este numit "catenet"[1]. Protocolul se referă la transmiterea unor blocuri de date numite datagrame de la surse la destinații, unde sursele și destinațiile sunt gazde (*host-uri*) identificate prin adrese de lungime fixă. Protocolul IP se referă deasemenea la fragmentarea și reasamblarea datagramelor lungi, atunci când este nevoie, pentru a putea fi transmise prin rețele "small packet".

1.2. Scop

Protocolul IP se limitează la a oferi funcțiile necesare transiterii unui pachet de biți (o datagramă internet) de la o sursă la o destinație printr-un sistem de rețele interconectate. Nu există mecanisme care să asigure consistența datelor, controlul fluxului, secvențierea sau alte servicii întâlnite de obicei în protocoalele de tip "host-to-host". Protocolul IP poate profita de serviciile rețelelor care îl folosesc pentru a oferi diferite tipuri de servicii și cu diferite caracteristici.

1.3. Interfețe

Acest protocol este apelat de protocoalele "host-to-host" într-un mediu internet. Protocolul apelează la rândul său la protocoalele de rețea locale pentru a transporta datagramele până la următoarea gateway (poartă) sau până la destinație.

De exemplu, un modul TCP va apela la modulul internet pentru a considera un segment TCP (inclusiv antetul TCP și datele propriu-zise) drept partea de date a unei datagrame internet. Modulul TCP va oferi adresele și alți parametri din antetul internet modulului internet ca argumente ale apelului. Modulul internet va crea apoi o datagramă și va apela la interfața de rețea locală pentru a transmite datagrama internet.

În cazul ARPANET, de exemplu, modulul internet va apela la un modul al rețelei locale care va adăuga așa-numitul "1822 leader"[2] la datagrama internet creând astfel un mesaj ARPANET ce va fi transmis către IMP. Adresa ARPANET va fi dedusă din adresa internet de către interfața de rețea locală și va fi adresa unei gazde ARPANET, gazdă ce poate fi o poartă de legătură cu alte rețele.

1.4. Operare

Protocolul IP implementează două funcții de bază: adresarea și fragmentarea.

Modulele internet folosesc adresa din antetul internet pentru a transmite datagramele către destinație. Selectarea unei căi pentru transmitere se numește rutare.

Modulele internet folosesc câmpuri din antetul internet pentru a fragmenta și reasambla datagramele când acest lucru este necesar, la transmiterea prin rețele "*small packet*".

Modelul de operare: există un modul internet implementat de fiecare gazdă ce participă la comunicare și de fiecare poartă ce interconectează rețele. Aceste module au reguli comune pentru interpretarea câmpurilor de adrese și pentru fragmentarea și asamblarea datagramelor. În plus, aceste module (mai ales în cazul porților), conțin proceduri ce permit luarea unor decizii de rutare și alte facilități.

Protocolul IP tratează fiecare datagramă ca o entitate independentă, fără legături cu alte datagrame. Nu există legături sau circuite logice (virutale sau de altă natură).

Protocolul IP folosește patru mecanisme cheie: **Type of Service** (tipul serviciului), **Time to Live** (timpul de viață), **Options** (opțiuni) și **Header Checksum** (verificarea antetului).

- **Type of Service** este folosit pentru a indica ce calitate are serviciul dorit și reprezintă de fapt o mulțime abstractă sau generalizată de parametri ce caracterizează opțiunile oferite de rețelele care formează internetul. Acest mod de a indica tipul serviciului trebuie folosit de porți pentru a selecta parametrii actuali ai transmisiei pentru o rețea anume, rețeaua folosită pentru următorul hop sau următoarea gateway, atunci când are loc rutarea unei datagrame.
- **Time to Live** reprezintă o limită superioară pentru timpul de viață al unei datagrame. Este setat de gazda care trimite datagrama și decrementat în fiecare punct al rutei. Dacă ajunge la valoarea zero înainte ca datagrama să ajungă la destinație, datagrama este distrusă. Timpul de viață poate fi considerat un timp limită de autodistrugere.
- **Options** oferă funcții de control necesare sau folosite în unele situații, dar care nu sunt folosite la comunicațiile uzuale. Aceste opțiuni includ informații despre timp, securitate și rutare specială.
- **Header Checksum** oferă facilitatea de a verifica dacă informațiile folosite la procesarea unei datagrame au fost transmise corect. Datele pot conține erori. Dacă verificarea eșuează, datagrama este ignorată de entitatea care detectează eroarea.

Protocolul IP nu oferă o facilitate de comunicare sigură. Nu există confirmare *end-to-end* sau *hop-by-hop*. Nu se face controlul erorilor pentru date, ci doar pentru antet. Nu există retransmiteri. Nu se face controlul fluxului.

Erorile detectate pot fi anunțate prin protocolul ICMP (Internet Control Message Protocols[3]) care este implementat în modulul IP.

2. PRIVIRE DE ANSAMBLU

2.1. Legătura cu alte protocoale

Diagrama de mai jos indică locul protocolului IP în ierarhia de protocoale:

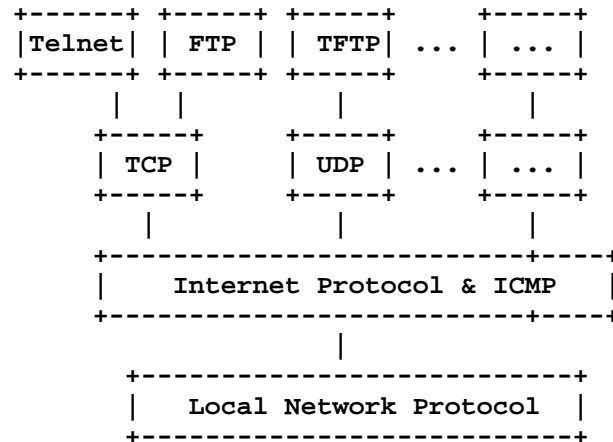


Figura 1. Relațiile dintre protocoale

Pe nivelele superioare protocolului IP se află protocoalele *host-to-host* de nivel înalt iar pe nivelul inferior se află protocolul rețelei locale. În acest context o rețea locală poate fi o rețea mică dintr-o clădire sau o rețea extinsă ca ARPANET.

2.2. Modul de operare

Modul de operare pentru transmiterea unei datagrame de la o aplicație la alta este ilustrat de scenariul următor:

Presupunem că această transmisie va implica o poartă intermediară.

Aplicația emițătoare pregătește datele și apelează la modulul internet local pentru a trimite datele ca o datagramă, adresa destinație și alți parametri fiind dați ca argumente ale apelului.

Modulul internet pregătește un antet de datagramă, la care atașează datele. Modulul determină adresa unei rețele locale ce corespunde adresei internet, în acest caz fiind adresa unei porți.

Trimite datagrama și adresa de rețea către interfața rețelei locale.

Interfața rețelei locale crează un antet specific rețelei, la care atașează datagrama, apoi trimite rezultatul prin rețeaua locală.

Datagrama ajunge la poarta specificată în antetul rețelei locale, rețeaua locală elimină acest antet din datagramă și o trimite modulului internet. Modulul internet determină din adresa internet faptul că datagrama trebuie trimisă mai departe la o altă gazdă dintr-o a doua rețea. Modulul internet determină o adresă de rețea pentru gazda destinație. Apoi apelează la interfața de rețea a acelei rețele pentru a trimite datagrama.

Această interfață de rețea creează un antet de rețea căruia îi atașează datagrama și trimite rezultatul către gazda destinație.

La destinație interfața de rețea locală elimină antetul de rețea și trimite datagrama modului internet.

Modulul internet determină faptul că datagrama este destinată unei aplicații ce rulează pe această gazdă. Trimite aplicației datagrama ca răspuns la un apel de sistem, transmitând adresa sursă și alți parametri ca rezultate ale apelului.

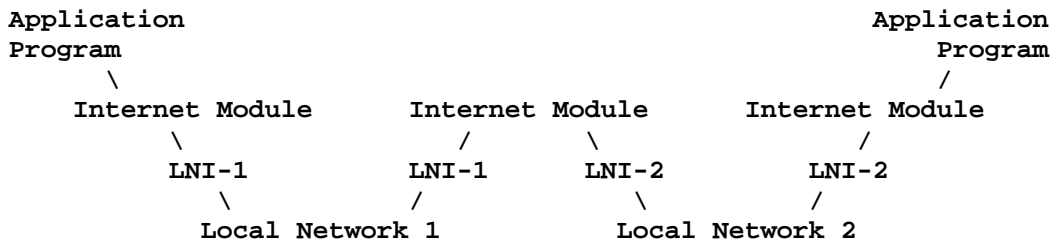


Figura 2. Calea de transmitere

2.3. Descrierea funcționalității

Scopul protocolului IP este de a muta datagrame printr-o mulțime de rețele interconectate. Acest lucru se realizează pasând datagramele de la un modul internet la altul până ajung la destinație.

Gazdele și porțile (gateways) din sistemul internet dețin module internet. Datagramele sunt rutate de la un modul internet la altul prin rețele individuale pe baza interpretării unei adrese internet. Deci, un mecanism important al protocolului IP este adresa internet.

La rutarea mesajelor de la un modul internet la altul, datagramele pot traversa o rețea în care dimensiunea maximă a unui pachet este mai mică decât dimensiunea datagramii. Pentru a rezolva această problemă, protocolul IP pune la dispoziție un mecanism de fragmentare.

Adresarea

Se face distincție între nume, adrese și rute[4]. Un nume arată ce căutăm. O adresă ne spune unde se găsește. O rută ne arată cum putem ajunge acolo. Protocolul IP se ocupă în primul rând de adrese. Protocelele de nivel mai înalt (*host-to-host* sau *aplicații*) trebuie să facă maparea de la nume la adrese. Modulul internet mapează adrese internet la adrese de rețea locale. Procedurile de nivel scăzut (rețea locală sau gateway) realizează maparea de la adrese de rețea la rute.

Adresele au o lungime fixă de patru octeți (32 de biți). O adresă începe cu numărul unei rețele, urmat de adresa locală (numită câmpul "rest"). Există trei clase de adrese internet: în clasa A, bitul cel mai semnificativ este 0, următorii 7 biți identifică rețeaua, iar ultimii 24 de biți reprezintă adresa locală; în clasa B, cei mai semnificativi 2 biți sunt 1, respectiv 0, următorii 14 biți identifică rețeaua, iar ultimii 16 biți reprezintă adresa locală; în clasa C, cei mai semnificativi 3 biți

sunt 1, 1 și 0, următorii 21 de biți identifică rețeaua, iar ultimii 8 biți reprezintă adresa locală.

Maparea adreselor internet la adrese de rețea trebuie făcută cu grijă; o singură gazdă trebuie să se poată comporta ca și cum ar fi mai multe gazde distincte, folosind în acest scop adrese internet distincte. Unele gazde vor avea deasemenea mai multe interfețe fizice (*multi-homing*).

Mai exact, trebuie ca orice gazdă să poată avea mai multe interfețe fizice cu rețeaua, fiecare având mai multe adrese internet logice.

Exemple de mapare a adreselor pot fi găsite în "Address Mappings"[5].

Fragmentarea

Fragmentarea unei datagrame internet este necesară când datagrama pleacă dintr-o rețea ce permite o dimensiune mare a pachetelor și, pentru a ajunge la destinație, trebuie să treacă printr-o rețea ce limitează pachetele la o dimensiune mai mică.

O datagramă internet poate fi marcată "nu fragmentați". Orice datagramă astfel marcată nu trebuie fragmentată în nici un caz. Dacă o astfel de datagramă nu poate ajunge la destinație fără a fi fragmentată, ea trebuie eliminată.

Fragmentarea, transmiterea și reasamblarea de-a lungul unei rețele locale care este invizibilă pentru modulul internet se numește fragmentare intranet și poate fi folosită[6].

Procedura care se ocupă de fragmentare internet și reasamblare trebuie să fie capabilă să împartă datagrama într-un număr oarecare de fragmente ce pot fi reasamblate mai târziu. La primirea fragmentelor, se folosește câmpul de identificare pentru ca fragmente din diferite datagrame să nu fie amestecate. Câmpul ce conține offset-ul fragmentului ne indică poziția acestuia în datagrama originală. Offset-ul și lungimea fragmentului determină exact porțiunea din datagrama originală acoperită de respectivul fragment. Flag-ul "*more-fragments*" (mai multe fragmente), dacă este resetat, indică ultimul fragment. Aceste câmpuri oferă suficiente informații pentru a reasambla datagramele.

Câmpul de identificare este folosit pentru a diferenția fragmentele unei datagrame de fragmentele altor datagrame. Modulul internet care inițiază transmiterea unei datagrame setează câmpul de identificare la o valoare care trebuie să fie unică pentru perechea sursă-destinație și protocolul respectiv pe toată durata cât datagrama va fi activă în sistemul internet. Modulul internet care inițiază transmiterea unei datagrame complete setează flag-ul "*more-fragments*" pe zero și offset-ul fragmentului pe zero.

Pentru a fragmenta o datagramă lungă, un modul internet (de exemplu, într-o poartă) creează două datagrame noi, în antetele cărora copie toate câmpurile din antetul internet al datagramei inițiale. Datele din datagrama lungă sunt împărțite în două părți, folosind blocuri de câte 8 octeți (64 biți) (prima parte este obligatoriu un multiplu întreg de 8 octeți, a doua nu). Notăm numărul de blocuri de câte 8 octeți din prima parte *NFB* (*Number of Fragment Blocks*). Datele din prima parte sunt incluse în prima datagramă nouă și câmpul ce reține lungimea totală este setat la lungimea primei datagrame. Flag-ul "*more-fragments*" este setat pe unu. Datele din a doua parte sunt incluse în a doua datagramă nouă și câmpul ce reține lungimea totală este setat la lungimea celei de a doua datagrame.

Flag-ul "*more-fragments*" conține aceeași valoare ca datagrama lungă. Offset-ul fragmentului pentru a doua datagramă nouă este setat la valoarea acestui câmp pentru datagrama inițială la care se adună NFB.

Acest procedeu poate fi generalizat pentru o împărțite în n fragmente.

Pentru a asambla fragmentele unei datagrame (de exemplu, la destinație), un modul internet combină datagramele ale căror valori coincid pe următoarele patru câmpuri: identificare, sursă, destinație și protocol. Asamblarea se face plasând porțiunile de date ale fiecărui fragment în poziția lor relativă indicată de offset-ul din antetul fragmentului. Primul fragment va avea offset-ul zero, iar ultimul fragment va avea flag-ul "*more-fragments*" setat pe zero.

2.4. Porțile (Gateways)

Porțile implementează protocolul IP pentru a putea primi și trimite mai departe datagrame în cadrul rețelelor. Porțile implementează deasemenea protocolul GGP (*Gateway to Gateway Protocol* [7]) pentru a coordona rutarea și diferite alte informații legate de controlul internetului.

Într-o poartă, protocoalele de nivel înalt nu e obligatoriu să fie implementate, iar funcțiile GGP sunt adăugate la modulul IP.

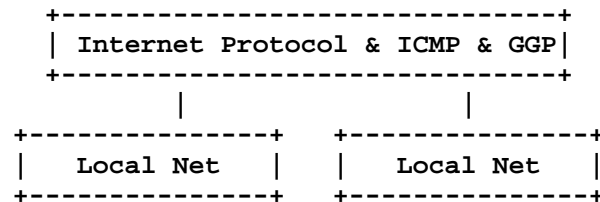


Figura 3. Protocoalele folosite de porți

3. SPECIFICAREA PROTOCOLULUI

3.1. Formatul antetului internet

Un antet internet are următorul format:

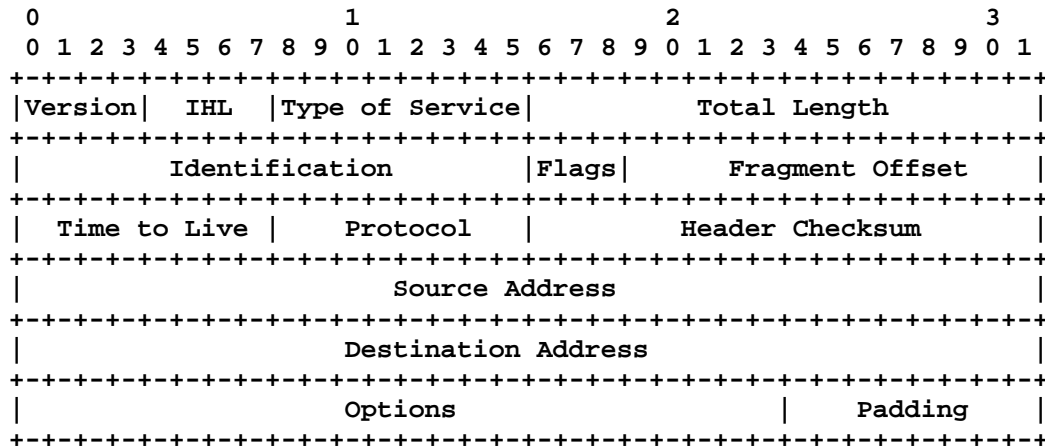


Figura 4. Exemplu de antet internet

Se observă că fiecare poziție marcată reprezintă un bit.

Version: 4 biți

Acest câmp indică formatul antetului internet. Acest document descrie versiunea 4.

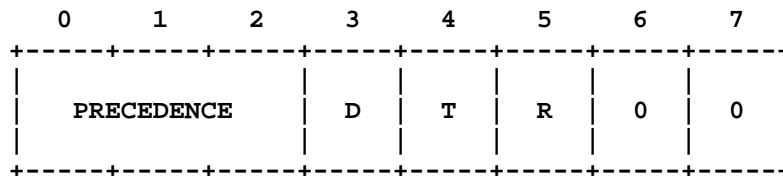
IHL: 4 biți

Internet Header Length reține lungimea antetului internet în cuvinte de 32 de biți, indicând astfel locul de unde încep datele. Valoarea minimă corectă pentru acest câmp este 5.

Type of Service: 8 biți

Câmpul *Type of Service* oferă informații referitoare la parametrii abstracți ai calității serviciului solicitat. Acești parametri trebuie folosiți la alegerea parametrilor efectivi ai serviciului atunci când se transmite o datagramă printr-o rețea anume. Multe rețele oferă precedența serviciilor, care tratează oarecum traficul de înaltă precedență ca fiind mai important decât alt trafic (în general impunând un nivel minim al traficului atunci când rețeaua este încărcată). Alegerea majoră este un compromis între întârziere mică (*low-delay*), siguranță mare (*high-reliability*) și trafic mare (*high-throughput*).

- Biții 0-2: Precedență (*Precedence*).
- Bitul 3: Întârziere (*Delay*)
 - 0 = Întârziere normală, 1 = Întârziere mică.
- Bitul 4: Trafic (*Throughput*)
 - 0 = Trafic normal, 1 = Trafic mare.
- Bitul 5: Siguranță (*Reliability*)
 - 0 = Siguranță normală, 1 = Siguranță mare.
- Biții 6-7: Rezervați pentru folosire ulterioară.



Precedență

- 111 - Network Control
- 110 - Internetwork Control
- 101 - CRITIC/ECP
- 100 - Flash Override
- 011 - Flash
- 010 - Immediate
- 001 - Priority
- 000 - Routine

Folosirea indicatorilor **Delay, Throughput, și Reliability** poate crește costul serviciului. În multe rețele îmbunătățirea unuia dintre parametri duce automat la scăderea performanței altui parametru.

Cu excepția unor cazuri foarte neobișnuite, cel mult doi dintre acești trei indicatori ar trebui setați.

Tipul serviciului este folosit pentru a specifica felul în care va fi tratată datagrama în timpul transmiterii sale prin sistemul internet. Exemple de mapare a tipurilor de servicii la servicii efective oferite de rețele ca AUTODIN II, ARPANET, SATNET și PRNET se găsesc în "Service Mappings"[8].

Precedența de tip **Network Control** este destinată doar folosirii în interiorul unei rețele. Folosirea și controlul efectiv depind de fiecare rețea în parte. Precedența de tip **Internetwork Control** este destinată doar porților. Dacă o rețea anume e interesată de folosirea efectivă a acestor indicatori de precedență, atunci este responsabilitatea rețelei respective să controleze accesul la și folosirea indicatorilor.

Total Length: 16 biți

Reprezintă lungimea datagramii, măsurată în octeți, incluzând antetul internet și datele. Acest câmp permite ca lungimea unei datagramme să fie de maxim 65.535 de octeți. Datagrammele atât de lungi sunt considerate nepractice pentru majoritatea gazdelor și rețelelor. Toate gazdele trebuie să poată primi datagramme de până la 576 de octeți (fie că ele ajung întregi sau fragmentate). Este recomandat ca o gazdă să nu trimită datagramme mai mari de 576 de octeți decât există siguranța că gazda destinație poate să accepte aceste datagramme.

Numărul 576 a fost ales pentru a permite ca blocuri de dimensiune rezonabilă să fie transmise împreună cu antetul necesar. De exemplu, această dimensiune permite ca un bloc de 512 octeți împreună cu un antet de 64 să încapă într-o datagramă. Lungimea maximă a unui antet internet este de 60 de octeți și un antet tipic are 20 de octeți, permițând o margine pentru antetele protoalelor de nivel mai înalt.

Identification: 16 biți

O valoare de identificare este asignată de gazda care trimite datagrama pentru a permite asamblarea fragmentelor datagrammei.

Flags: 3 biți

Diferite flag-uri de control.

Bitul 0: rezervat, trebuie să fie zero
 Bitul 1: (DF) 0 = Poți fragmenta, 1 = Nu fragmenta (*Don't Fragment*).
 Bitul 2: (MF) 0 = Ultimul fragment, 1 = Mai există fragmente (*More Fragments*).

0	1	2	
	D	M	
0	F	F	

Fragment Offset: 13 biți

Acest câmp indică locul fragmentului în datagramă.

Offset-ul fragmentului este măsurat în unități de câte 8 octeți (64 de biți). Primul fragment are offset-ul zero.

Time to Live: 8 biți

Indică timpul maxim pe care o datagramă îl poate petrece în sistemul internet. Dacă valoarea câmpului devine zero, atunci datagrama trebuie distrusă. Acest câmp este modificat la procesarea antetului internet. Timpul este măsurat în secunde, dar deoarece fiecare modul care procesează datagrama trebuie să decrementeze TTL-ul cu măcar o unitate chiar dacă procesarea durează mai puțin de o secundă, TTL-ul trebuie văzut doar ca o limită superioară a timpului cât o datagramă poate exista. Intenția este de a cauza distrugerea datagramelor ce nu pot ajunge la destinație și de a restrânge durata maximă de viață a unei datagrame.

Protocol: 8 biți

Acest câmp indică protocolul de nivel superior folosit în secțiunea de date a datagramei. Valorile pentru diferite protocoale sunt prezentate în "Assigned Numbers"[9].

Header Checksum: 16 biți

Permite verificarea doar pentru antet. Deoarece unele câmpuri din antet se modifică (de exemplu, *TTL*), acest câmp este recalculat și verificat de fiecare dată când antetul este procesat.

Algoritmul de calculare este:

Câmpul *Header Checksum* este setat pe zero. Se calculează suma în complement față de unu a tuturor cuvintelor de 16 biți din antet. Complementul pe 16 biți față de 1 al acestei valori reprezintă valoarea câmpului *Header Checksum*.

Aceasta este o valoare ușor de calculat și experimentele au arătat că este adecvată, dar e temporară și poate fi înlocuită de o procedură CRC, în funcție de experimentele ulterioare.

Source Address: 32 biți

Adresa sursă. Vezi secțiunea 3.2.

Destination Address: 32 biți

Adresa destinație. Vezi secțiunea 3.2.

Options: variabilă

Opțiunile pot să apară sau nu în datagrame. Ele trebuie să fie implementate de toate modulele IP (gazde și porți). Opțională este doar transmiterea lor într-o anumită datagramă, nu și implementarea.

În unele medii opțiunile de securitate pot fi cerute pentru toate datagramele.

Câmpul de opțiuni poate varia în lungime. Pot exista zero sau mai multe opțiuni. Există două cazuri pentru formatul unei opțiuni:

Cazul 1: Un singur octet cu tipul opțiunii.

Cazul 2: Un octet ce indică tipul, un octet ce indică lungimea opțiunii și octeții cu datele efective.

Octetul ce reține lungimea opțiunii numără octetul de tip și octetul de lungime pe lângă octeții de date.

Octetul ce reține tipul opțiunii (*option-type octet*) are 3 câmpuri:

1 bit flag copiat
2 biți clasa opțiunii
5 biți numărul opțiunii

Flag-ul copiat indică faptul că această opțiune este copiată în toate fragmentele datagramei la fragmentare.

0 = nu e copiat
1 = e copiat

Clasele de opțiuni sunt:

0 = control
1 = rezervat pentru o folosire ulterioară
2 = debugging și măsurare
3 = rezervat pentru o folosire ulterioară

Următoarele opțiuni internet sunt definite:

CLASĂ	NUMĂR	LUNGIME	DESCRIERE
0	0	-	<i>Sfârșitul listei de opțiuni.</i> Această opțiune ocupă doar un octet; nu are octet de lungime.
0	1	-	<i>Nici o operație.</i> Această opțiune ocupă doar un octet; nu are octet de lungime.
0	2	11	<i>Securitate.</i> Folosită pentru a reține coduri legate de securitate, compartimentare, grupul utilizator (TCC) și tratarea restricțiilor, compatibile cu cerințele DOD.
0	3	var.	<i>Rutare "sursă liberă".</i> Folosită pentru a ruta datagrama pe baza informațiilor furnizate de sursă.
0	9	var.	<i>Rutare "sursă exactă".</i> Folosită pentru a ruta datagrama pe baza informațiilor furnizate de sursă.

0	7	var.	<i>Inregistrează ruta. Folosită pentru a detecta pe ce rută este transportată datagrama.</i>
0	8	4	<i>Identificator de flux. Folosită pentru a stoca identificatorul fluxului.</i>
2	4	var.	<i>Internet Timestamp</i>

Definiții de opțiuni tipice:

Sfârșitul listei de opțiuni (End of Option List)

```
+-----+
|00000000|
+-----+
  type=0
```

Opțiunea indică sfârșitul listei de opțiuni. Acesta poate să nu coincidă cu sfârșitul antetului, conform lungimii antetului. Opțiunea nu este folosită la sfârșitul fiecărei opțiuni. Ea trebuie folosită doar dacă sfârșitul listei de opțiuni nu coincide cu sfârșitul antetului.

Poate fi copiată, introdusă sau ștearsă la fragmentare sau din orice alt motiv.

Nici o operație (No operation)

```
+-----+
|00000001|
+-----+
  Type=1
```

Această opțiune poate fi folosită între opțiuni, de exemplu, pentru a alinia începutul următoarei opțiuni la o limită de 32 de biți.

Poate fi copiată, introdusă sau ștearsă la fragmentare sau din orice alt motiv.

Securitate (Security)

Această opțiune oferă o modalitate pentru gazde de a trimite parametri legați de securitate, compartimentare, tratarea restricțiilor și TCC (grupul utilizator). Formatul opțiunii este următorul:

```
+-----+-----+---//---+---//---+---//---+---//---+
|10000010|00001011|SSS SSS|CCC CCC|HHH HHH| TCC  |
+-----+-----+---//---+---//---+---+---//---+---+
  Type=130 Length=11
```

Securitate (câmpul S): 16 biți

Specifică unul din cele 16 nivele de securitate (opt dintre ele sunt rezervate pentru o folosire ulterioară).

```
00000000 00000000 - Unclassified
11110001 00110101 - Confidential
01111000 10011010 - EFTO
10111100 01001101 - MMMM
01011110 00100110 - PROG
```

```

10101111 00010011 - Restricted
11010111 10001000 - Secret
01101011 11000101 - Top Secret
00110101 11100010 - nefolosit
10011010 11110001 - nefolosit
01001101 01111000 - nefolosit
00100100 10111101 - nefolosit
00010011 01011110 - nefolosit
10001001 10101111 - nefolosit
11000100 11010110 - nefolosit
11100010 01101011 - nefolosit

```

Compartimente (câmpul C): 16 biți

Când informația transmisă nu e compartimentată, toți biții sunt setați pe zero. Alte valori pentru acest câmp pot fi obținute de la *Defense Intelligence Agency*.

Tratarea restricțiilor (câmpul H): 16 biți

Valorile pentru marcaje de control și de permisiune sunt digrafuri alfanumerice și sunt definite în *Defense Intelligence Agency Manual DIAM 65-19, "Standard Security Markings"*.

Codul pentru controlul transmisiei (TCC field): 24 biți

Oferă un mod de a descongiona traficul și de a defini comunități de interes controlate. Valorile acestui câmp sunt trigrafuri și sunt disponibile în *HQ DCA Code 530*.

Trebuie copiată la fragmentare. Această opțiune (Securitate) apare cel mult o dată într-o datagramă.

Sursă liberă (Loose Source) și înregistrează ruta (Record route)

```

+-----+-----+-----+-----//-----+
|10000011| length | pointer|      route data      |
+-----+-----+-----+-----//-----+
Type=131

```

Opțiunea *loose source* și *record route* (LSRR) reprezintă un mijloc prin care sursa datagramii poate oferi informații de rutare ce vor fi utilizate de către porți pentru a direcționa datagrama spre destinație și totodată un mod de a înregistra informații despre rută.

Opțiunea începe cu codul de specificare a tipului. Al doilea octet este lungimea opțiunii, care include octetul pentru tip, octetul pentru lungime, pointerul și octeții cu datele de rutare. Al treilea octet este pointerul la datele din câmpul "*route data*", ce indică octetul cu care începe următoarea adresă sursă ce trebuie procesată. Pointerul este relativ la această opțiune, iar cea mai mică valoare validă este 4.

Câmpul "*route data*" este compus dintr-o serie de adrese internet, fiecare având 32 de biți (4 octeți). Dacă pointerul este mai mare decât lungimea, atunci ruta sursă este goală (și ruta

înregistrată este plină) iar în continuare rutarea se va face doar pe baza câmpului ce specifică adresa destinație.

Dacă datagrama a ajuns la adresa destinație și pointerul nu este mai mare decât lungimea, atunci următoarea adresă din ruta sursă înlocuiește adresa din câmpul destinație, ruta înregistrată înlocuiește adresa sursă tocmai folosită și pointerul este incrementat cu 4.

Adresa rutei înregistrate este chiar adresa internet a modulului internet, așa cum este ea cunoscută în mediul în care datagrama este trimisă.

Această procedură de a înlocui ruta sursă cu ruta înregistrată (cu toate că este în ordinea inversă celei necesare pentru a fi folosită ca o rută sursă), face ca partea de opțiuni și antetul IP ca un întreg să aibă lungime constantă pe măsură ce datagrama este transportată prin internet.

Această opțiune reprezintă o rută sursă liberă deoarece orice poartă sau gazdă IP are dreptul de a folosi orice rută a oricărei porți intermediare pentru ca datagrama să ajungă la următoarea adresă din rută.

Opțiunea trebuie copiată la fragmentare. Apare cel mult o dată într-o datagramă.

Sursă strictă (Strict Source) și înregistrează ruta (Record route)

```
+-----+-----+-----+-----//-----+
|10001001| length | pointer|      route data      |
+-----+-----+-----+-----//-----+
Type=137
```

Opțiunea *strict source* și *record route* (SSRR) reprezintă un mijloc prin care sursa datagramii poate oferi informații de rutare ce vor fi utilizate de către porți pentru a direcționa datagrama spre destinație și totodată un mod de a înregistra informații despre rută.

Opțiunea începe cu codul de specificare a tipului. Al doilea octet este lungimea opțiunii, care include octetul pentru tip, octetul pentru lungime, pointerul și octeții cu datele de rutare. Al treilea octet este pointerul la datele din câmpul "route data", ce indică octetul cu care începe următoarea adresă sursă ce trebuie procesată. Pointerul este relativ la această opțiune, iar cea mai mică valoare validă este 4.

Câmpul "route data" este compus dintr-o serie de adrese internet, fiecare având 32 de biți (4 octeți). Dacă pointerul este mai mare decât lungimea, atunci ruta sursă este goală (și ruta înregistrată este plină) iar în continuare rutarea se va face doar pe baza câmpului ce specifică adresa destinație.

Dacă datagrama a ajuns la adresa destinație și pointerul nu este mai mare decât lungimea, atunci următoarea adresă din ruta sursă înlocuiește adresa din câmpul destinație, ruta înregistrată înlocuiește adresa sursă tocmai folosită și pointerul este incrementat cu 4.

Adresa rutei înregistrate este chiar adresa internet a modulului internet, așa cum este ea cunoscută în mediul în care datagrama este trimisă.

Această procedură de a înlocui ruta sursă cu ruta înregistrată (cu toate că este în ordinea inversă celei necesare pentru a fi folosită ca o rută sursă), face ca partea de opțiuni și antetul IP ca un întreg să aibă lungime constantă pe măsură ce datagrama este transportată prin internet.

Această opțiune reprezintă o rută sursă strictă deoarece orice poartă sau gazdă IP trebuie să trimită datagrama direct la următoarea adresă din ruta sursă numai prin rețeaua conectată direct, indicată de următoarea adresă ce trebuie atinsă pentru a ajunge la următoarea poartă sau gazdă specificată în rută.

Opțiunea trebuie copiată la fragmentare. Apare cel mult o dată într-o datagramă.

Înregistrează ruta (Record route)

```
+-----+-----+-----+-----//-----+
|00000111| length | pointer|   route data   |
+-----+-----+-----+-----//-----+
Type=7
```

Opțiunea *record route* oferă un mod de a înregistra ruta unei datagramme internet.

Opțiunea începe cu codul de specificare a tipului. Al doilea octet este lungimea opțiunii, care include octetul pentru tip, octetul pentru lungime, pointerul și octeții cu datele de rutare. Al treilea octet este pointerul la datele din câmpul "*route data*", ce indică octetul cu care începe următoarea zonă de stocare a unei adrese. Pointerul este relativ la această opțiune, iar cea mai mică valoare validă este 4.

O rută înregistrată este compusă dintr-o serie de adrese internet, fiecare având 32 de biți (4 octeți). Dacă pointerul este mai mare decât lungimea, atunci zona care reține ruta este plină. Gazda inițială trebuie să aloce pentru această opțiune o zonă de date destul de mare pentru a putea reține toată adresa așteptată. Dimensiunea opțiunii nu se schimbă prin adăugarea de adrese. Conținutul inițial al zonei de date trebuie să fie zero.

Când un modul internet rutează o datagramă, el verifică dacă opțiunea *record route* este prezentă. În caz afirmativ, inserează propria adresă internet, așa cum este cunoscută în mediul în care datagrama va fi trimisă, în cadrul rutei înregistrate, începând cu octetul indicat de pointer, și incrementează pointerul cu 4.

Dacă zona de memorare a rutei este deja plină (pointerul depășește lungimea), atunci datagrama este trimisă fără a insera adresa în zona respectivă. Dacă există loc, dar nu suficient pentru o adresă, datagrama originală este considerată eronată și este distrusă. În ambele cazuri un mesaj ICMP ce specifică eroarea ar putea fi trimis gazdei sursă [3].

Câmpul *Timestamp* este aliniat la dreapta și reține pe 32 de biți un timestamp în milisecunde, de la ora 0:00 UT. Dacă timpul nu este disponibil în milisecunde sau nu poate fi specificat relativ la ora 0:00 UT, atunci orice timp poate fi inserat ca timestamp cu condiția ca cel mai semnificativ bit al câmpului să fie setat pe unu pentru a indica folosirea unei valori nestandard.

Gazda inițială trebuie să aloce acestei opțiuni o zonă de date destul de mare pentru a reține toate informațiile așteptate. Dimensiunea opțiunii nu se modifică prin adăugarea de timestamps. Conținutul inițial al zonei de date trebuie să fie zero sau să conțină perechi adresă internet/zero.

Dacă zona de date pentru timestamps este deja ocupată (pointerul depășește lungimea), atunci datagrama este trimisă mai departe fără a insera nici un timestamp, iar câmpul *overflow* este incrementat cu unu.

Dacă mai există loc, dar nu suficient pentru un întreg timestamp, sau dacă valoarea câmpului *overflow* nu mai poate fi incrementată, datagrama originală este considerată eronată și este distrusă. În ambele cazuri un mesaj ICMP ce specifică eroarea ar putea fi trimis gazdei sursă [3].

Opțiunea *timestamp* nu este copiată la fragmentare. Este reținută în primul fragment. Apare cel mult o dată într-o datagramă.

Padding: variabilă

Padding-ul unui antet internet este folosit pentru a asigura faptul că antetul are o lungime multiplu de 32 de biți. Valoarea de padding este zero.

3.2. Discuție

Implementarea unui protocol trebuie să fie robustă. Fiecare implementare trebuie să fie pregătită pentru a interopera cu implementări create de alte persoane. Scopul acestui document este de a fi exact în prezentarea protocolului. Dar sunt posibile interpretări diferite. În general, o implementare trebuie să fie conservatoare în privința comportamentului de trimitere a datagramelor și liberală în comportamentul de primire. Mai exact, trebuie să aibă grijă să trimită datagrame bine formate, dar trebuie să accepte orice datagramă pe care o poate interpreta (de exemplu, să nu obiecteze referitor la erorile tehnice cât timp înțelesul este clar).

Principalul serviciu internet este bazat pe datagrame și oferă suport pentru fragmentarea datagramelor de către porți, reasamblarea lor făcându-se de către modulul internet din cadrul gazdei destinație. Bineînțeles, fragmentarea și reasamblarea datagramelor în interiorul unei rețele sau printr-o înțelegere privată între porțile rețelei sunt de asemenea permise, ele fiind transparente pentru protocoalele internet și pentru protocoalele de nivel mai înalt. Acest tip de fragmentare și reasamblare transparentă se mai numește și fragmentare dependentă de rețea ("*network-dependent fragmentation*") sau fragmentare intranet și nu este discutată în acest document.

Adresele internet fac distincție între surse și destinații la nivel de gazdă și oferă deasemenea un câmp de protocol. Se presupune că fiecare protocol va oferi orice multiplexare necesară în cadrul unei gazde.

Adresarea

Pentru a oferi flexibilitate în asignarea adreselor rețelelor și pentru a permite numărul mare de rețele de dimensiuni mici sau mijlocii, câmpul adresă este codificat pentru a specifica un număr mic de rețele cu multe gazde, un număr moderat de rețele cu număr mediu de gazde și un număr mare de rețele cu puține gazde. În plus există un cod ce poate fi folosit pentru un mod extins de adresare.

Formatul unei adrese:

Biții semnificativi	Format	Clasă
0	7 biți - rețea, 24 biți - gazdă	a
10	14 biți - rețea, 16 biți - gazdă	b
110	21 biți - rețea, 8 biți - gazdă	c
111	mod extins de adresare	

O valoare zero în câmpul rețea identifică rețeaua locală. Acest mod de adresare e folosit doar în anumite mesaje ICMP. Modul extins de adresare este nedefinit. Aceste caracteristici sunt rezervate pentru folosirea ulterioară.

Valorile efective asignate adreselor rețelelor se găsesc în "Assigned numbers" [9].

Adresele locale, asignate de rețeaua locală, trebuie să permită ca o singură gazdă fizică să se comporte ca mai multe gazde internet distincte. Mai exact, trebuie să existe un mod de mapare între adresele gazdelor internet și interfețele rețelei/gazdei care să permită ca mai multe adrese internet să corespundă unei interfețe. Trebuie deasemenea să fie permis ca o gazdă să aibă mai multe interfețe fizice și să trateze datagramele de la aceste interfețe ca și cum toate ar fi destinate unei singure gazde.

Mapări între adrese internet și adrese pentru ARPANET, SATNET, PRNET și alte rețele sunt descrise în "Address Mappings" [5].

Fragmentarea și reasamblarea

Câmpul de identificare internet (ID) este folosit împreună cu adresele sursă și destinație și câmpurile de protocol pentru a identifica fragmentele de datagramă în vederea reasamblării.

Flag-ul *More Fragments (MF)* este setat dacă datagrama nu reprezintă ultimul fragment. Câmpul *Fragment Offset* identifică locația fragmentului, relativ la începutul datagramei nefragmentate. Fragmentele sunt numărate în unități de 8 octeți. Strategia de fragmentare este proiectată astfel încât o datagramă nefragmentată are toate informațiile despre fragmentare zero ($MF = 0$, $fragment\ offset = 0$). Dacă o datagramă internet este fragmentată, zona sa de date trebuie împărțită în zone de câte 8 octeți.

Acest format permite $2^{13} = 8192$ fragmente de câte 8 octeți fiecare pentru un total de 65.536 de octeți. Se poate observa consistența cu lungimea totală a unei datagrame (bineînțeles, antetul este numărat în lungimea totală și nu în fragmente).

Când are loc fragmentarea, unele opțiuni sunt copiate, altele rămân doar în primul fragment.

Fiecare modul internet trebuie să fie capabil să trimită o datagramă de 68 de octeți fără a o fragmenta. Aceasta deoarece un antet internet poate avea până la 60 de octeți, iar lungimea minimă a unui fragment este de 8 octeți.

Fiecare destinație internet trebuie să fie capabilă să primească o datagramă de 576 de octeți, fragmentată sau nu.

Câmpurile care pot fi afectate de fragmentare:

- (1) options
- (2) more fragments
- (3) fragment offset
- (4) internet header length
- (5) total length
- (6) header checksum

Dacă flag-ul *Don't Fragment (DF)* este setat, atunci NU este permisă fragmentarea datagramei respective, cu toate că ea ar putea fi distrusă. Acest lucru poate fi folosit pentru a interzice fragmentarea atunci când gazda sursă nu deține suficiente resurse pentru a reasambla fragmentele.

Un exemplu de folosire a caracteristicii *Don't Fragment* este în cazul gazdelor mici. O gazdă mică ar putea avea un program de bootare care acceptă o singură datagramă, o stochează în memorie și apoi o execută.

Procedurile de fragmentare și reasamblare se descriu cel mai ușor prin exemple. Procedurile de mai jos sunt exemple de implementări.

Notații generale în următoarele pseudo-programe: "*=<*" înseamnă "mai mic sau egal", "*#*" înseamnă "diferit", "*=*" înseamnă "egal", "*<-*" înseamnă "ia valoarea", "*x to y*" include *x* și exclude *y*; de exemplu, "*4 to 7*" va include 4, 5 și 6 (dar nu 7).

Un exemplu de procedură de fragmentare:

Datagrama de dimensiune maximă ce poate fi transmisă prin următoarea rețea este numită unitatea maximă de transmisie (*maximum transmission unit - MTU*).

Dacă lungimea totală este mai mică sau egală cu unitatea maximă de transmisie, atunci se trece la următorul pas din procesarea datagramei. Altfel, datagrama se împarte în două fragmente, primul fragment având dimensiunea maximă, iar al doilea conținând restul datagramei. Pentru primul fragment se trece la următorul pas din procesarea datagramei, în timp ce pentru al doilea fragment se apelează din nou această procedură (fragmentul ar putea fi în continuare prea mare).

Notații:

- FO - Fragment Offset (Offset-ul fragmentului)
- IHL - Internet Header Length (Lungimea antetului)
- DF - Don't Fragment flag (Flag-ul "Nu fragmenta")
- MF - More Fragments flag (Flag-ul "Mai multe fragmente")
- TL - Total Length (Lungimea totală)
- OFO - Old Fragment Offset (Offset-ul fragmentului vechi)

OIHL - Old Internet Header Length (Lungimea antetului vechi)
 OMF - Old More Fragments flag (Flag-ul "Mai multe fragmente" vechi)
 OTL - Old Total Length (Lungimea totală veche)
 NFB - Number of Fragment Blocks (Numărul de blocuri din fragment)
 MTU - Maximum Transmission Unit (Unitatea maximă de transmisie)

Procedura:

IF (TL <= MTU) THEN Se trece la următorul pas în procesarea datagramei
 ELSE IF (DF = 1) THEN Se distruge datagrama
 ELSE

Pentru a produce primul fragment:

- (1) Copie antetul internet original;
- (2) OIHL <- IHL; OTL <- TL; OFO <- FO; OMF <- MF;
- (3) NFB <- (MTU-IHL*4)/8;
- (4) Atașează primii NFB*8 octeți de date;
- (5) Corectează antetul:
 MF <- 1; TL <- (IHL*4)+(NFB*8);
 Recalculează câmpul Checksum;
- (6) Acest fragment trece la următorul pas din procesarea datagramei;

Pentru a produce cel de-al doilea fragment:

- (7) Copie selectiv antetul internet (unele opțiuni nu sunt copiate; vezi definițiile lor)
- (8) Adaugă datele rămase;
- (9) Corectează antetul:
 IHL <- (((OIHL*4)-(lungimea opțiunilor necopiate))+3)/4;
 TL <- OTL - NFB*8 - (OIHL-IHL)*4);
 FO <- OFO + NFB; MF <- OMF;
 Recalculează câmpul Checksum;
- (10) Pentru acest fragment se aplică testul; TERMINAT.

În procedura de mai sus fiecare fragment (cu excepția ultimului) are dimensiunea maximă posibilă. O alternativă ar fi să generăm datagrame de lungimi mai mici. De exemplu, se poate implementa o procedură de fragmentare care să divizeze în mod repetat datagrame mari în jumătate până când fragmentele rezultate au o lungime mai mică decât dimensiunea unității de transmisie maxime.

Un exemplu de procedură de reasamblare:

Pentru fiecare datagramă buffer-ul de identificare este calculat prin concatenarea sursei, destinației, protocolului și câmpurilor de identificare. Dacă datagrama este întreagă (câmpurile *Fragment Offset* și *More Fragments* sunt zero), atunci sunt eliberate orice resurse de reasamblare asociate cu acest buffer și datagrama este trimisă la următor pas din procesarea sa.

Dacă nu există alt fragment cu același buffer de identificare, atunci se alocă resurse pentru reasamblare: un buffer de date, un buffer de antet, un tablou de biți ce corespund blocurilor fragmentului, un câmp ce reține lungimea totală a datelor și un cronometru (*timer*). Datele din fragment sunt copiate în buffer-ul de date conform cu offset-ul și lungimea fragmentului, iar în tabloul de biți sunt setați biții corespunzători blocurilor primite.

Dacă acesta este primul fragment (offset-ul fragmentului este zero), atunci antetul său este plasat în buffer-ul de antet. Dacă este

ultimul fragment (câmpul *More Fragments* este zero), atunci este calculată lungimea totală a datelor. Dacă acest fragment face datagrama completă (fapt ce poate fi verificat prin consultarea tabelii de biți), atunci se trece la următorul pas din procesarea datagrammei; altfel, cronometrul este setat la maximum dintre valoarea curentă a cronometrului și valoarea câmpului *Time to Live* din acest fragment; și procedura de reasamblare se termină.

Dacă timpul indicat de cronometru expiră, atunci toate resursele de reasamblare alocate buffer-ului respectiv sunt eliberate. Inițial cronometrul este setat la o limită inferioară a timpului de așteptare pentru realocare. Aceasta deoarece timpul de așteptare va crește în cazul în care câmpul *Time to Live* al fragmentului primit este mai mare decât valoarea curentă a cronometrului, dar nu va crește dacă *Time to Live* are o valoare mai mică. Valoare maximă pe care o poate lua cronometrul este valoarea maximă a câmpului *Time to Live* (aproximativ 4.25 minute). Recomandarea actuală pentru inițializarea cronometrului este 15 secunde. Această valoare s-ar putea schimba pe măsură ce se acumulează experiență în folosirea acestui protocol. Se observă că alegerea valorii acestui parametru este legată de capacitatea disponibilă a buffer-ului și de rata de transmisie a datelor; mai exact, produsul dintre rata de transmisie a datelor și valoarea cronometrului = dimensiunea buffer-ului (de exemplu, 10Kb/s X 15s = 150Kb).

Notații:

FO - Fragment Offset (Offset-ul fragmentului)
 IHL - Internet Header Length (Lungimea antetului)
 MF - More Fragments flag (Flag-ul "Mai multe fragmente")
 TTL - Time To Live (Timpul de viață)
 NFB - Number of Fragment Blocks (Numărul de blocuri din fragment)
 TL - Total Length (Lungimea totală)
 TDL - Total Data Length (Lungimea totală a datelor)
 BUFID - Buffer Identifier (Identificatorul buffer-ului)
 RCVBT - Fragment Received Bit Table (Tabloul de biți al fragmentului)
 TLB - Timer Lower Bound (Limita inferioară pentru cronometru)

Procedura:

```
(1)  BUFID <- sursa|destinația|protocolul|identificarea;
(2)  IF (FO = 0 AND MF = 0)
(3)    THEN IF (buffer-ul cu BUFID este alocat)
(4)      THEN Eliberează resursele pentru acest BUFID;
(5)      Treci la următorul pas din procesare; TERMINAT.
(6)  ELSE IF (nici un buffer cu BUFID nu este alocat)
(7)    THEN Alocă resurse de reasamblare cu BUFID;
      TIMER <- TLB; TDL <- 0;
(8)  Copie datele din fragment în buffer-ul de date cu BUFID
      de la octetul FO*8 până la octetul (TL-(IHL*4))+FO*8;
(9)  Setează biții RCVBT de la FO
      până la FO+((TL-(IHL*4)+7)/8);
(10) IF (MF = 0) THEN TDL <- TL-(IHL*4)+(FO*8)
(11) IF (FO = 0) THEN Copie antetul în buffer-ul de antet
(12) IF (TDL # 0
(13)   AND toți biții RCVBT de la 0
      până la (TDL+7)/8 sunt setați)
(14)   THEN TL <- TDL+(IHL*4)
(15)   Treci la următorul pas din procesare;
(16)   Eliberează toate resursele de reasamblare
```

- pentru BUFID; TERMINAT.
- (17) TIMER <- MAX(TIMER,TTL);
- (18) Așteaptă până la următorul fragment sau până când
 expiră timpul;
- (19) Expiră timpul: eliberează toate resursele cu BUFID; TERMINAT.

În cazul în care două sau mai multe fragmente conțin aceleași date (date identice sau care se suprapun parțial), această procedură va folosi în buffer-ul de date copia sosită cel mai recent.

Identificare

Alegerea valorii de identificare pentru o datagramă se bazează pe nevoia de a asigura o modalitate de identificare în mod unic a fragmentelor unei anumite datagrame. Modulul de protocol care se ocupă cu reasamblarea fragmentelor consideră că mai multe fragmente aparțin aceleiași datagrame dacă au aceeași sursă, aceeași destinație, același protocol și aceeași valoarea a câmpului *Identifior*. În acest fel, gazda sursă trebuie să dea câmpului *Identifior* o valoare unică pentru sursa, destinația și protocolul respectiv, atât timp cât datagrama sau fragmente ale sale ar putea fi active în sistemul internet.

Se pare că modulul ce trimite datagrame trebuie să păstreze un tablou al identificatorilor, având o intrare pentru fiecare destinație cu care a comunicat în ultimul timp (timpul maxim de viață al unui pachet pentru internet).

În orice caz, deoarece câmpul *Identifior* permite 65,536 de valori diferite, unele gazde ar putea folosi identificatori unici independent de destinație.

Unele protocoale de nivel mai înalt pot alege identificatorii. De exemplu, modulele protocolului TCP ar putea retransmite un segment TCP identic, și probabilitatea unei recepționări corecte va crește dacă retransmisia va avea același identificator ca transmisia originală, de vreme ce fragmente din ambele datagrame pot fi folosite pentru a construi corect segmentul TCP.

Type of Service

Câmpul *Type of Service (TOS)* este destinat selectării calității serviciilor internet. Tipul serviciului este specificat prin parametrii abstracți *precedence*, *delay*, *throughput* și *reliability*. Acești parametri abstracți vor fi mapați la parametrii efectivi ai rețelelor particulare prin care va trece datagrama.

Precedence. O măsură a importanței datagramei.

Delay. Transmiterea promptă este importantă pentru datagramele cu această indicație.

Throughput. O rata bună de transmitere este importantă pentru datagramele cu această indicație.

Reliability. Un efort substanțial pentru a asigura transmiterea este important pentru datagramele cu această indicație.

De exemplu, rețeaua ARPANET are un bit de prioritate și se poate alege între mesaje "standard" (tipul 0) și mesaje "necontrolate" (tipul 3), (alegerea între mesaje cu pachet unic și cu pachete multiple poate fi

considerată un parametru al serviciului). Mesajele necontrolate au tendința de a fi livrate cu mai puțină siguranță și suferă o întârziere mai mică. Să presupunem că o datagramă trebuie trimisă prin ARPANET. Fie tipul de serviciu dat prin:

```
Precedence: 5
Delay:      0
Throughput: 1
Reliability: 1
```

În acest exemplu, maparea acestor parametri la cei disponibili pentru ARPANET ar consta în setarea bitului de prioritate ARPANET (deoarece valoarea câmpului *Precedence* este în jumătatea superioară a domeniului său), selectarea mesajelor standard (deoarece cerințele *Throughput* și *Reliability* sunt indicate, iar *Delay* nu). Mai multe detalii despre maparea serviciilor se găsesc în "Service Mappings" [8].

Time to Live

Timpul de viață este setat de gazda care trimite datagrama la timpul maxim pe care o datagramă îl poate petrece în sistemul internet. Dacă o datagramă există în sistemul internet mai mult decât timpul său de viață, atunci ea trebuie distrusă.

Acest câmp trebuie decrementat în fiecare punct în care antetul internet este procesat, pentru a reflecta timpul petrecut procesând datagrama. Chiar dacă nu sunt disponibile informații locale despre timpul efectiv de procesare, câmpul trebuie decrementat cu 1. Timpul este măsurat în secunde (de exemplu, valoarea 1 înseamnă o secundă). Astfel, timpul maxim de viață este de 255 de secunde sau 4.25 minute. Deoarece fiecare modul care procesează datagrama trebuie să decrementeze TTL-ul cu măcar o unitate chiar dacă procesarea durează mai puțin de o secundă, TTL-ul trebuie văzut doar ca o limită superioară a timpului cât o datagramă poate exista. Intenția este de a cauza distrugerea datagramelor ce nu pot ajunge la destinație și de a restrânge durata maximă de viață a unei datagrame.

Unele protocoale sigure de nivel mai înalt se bazează pe presupunerea că datagramele duplicate nu vor sosi decât după trecerea unui anumit timp. Câmpul TTL reprezintă un mod prin care aceste protocoale se pot asigura că presupunerile lor sunt adevărate.

Options

Câmpul *Option* este opțional în fiecare datagramă, dar obligatoriu în implementare. Mai exact, prezența sau absența unei opțiuni este la alegerea celui care trimite, dar fiecare modul internet trebuie să poată parsa fiecare opțiune. Mai multe opțiuni pot fi prezente în acest câmp.

Este posibil ca opțiunile să nu aibă ca lungime un multiplu de 32 de biți. Antetul internet trebuie completat cu octeți zero. Primul dintre acești octeți va fi interpretat ca o opțiune *end-of-options*, iar restul vor fi considerați padding-ul antetului.

Fiecare modul internet trebuie să poată interpreta fiecare opțiune. Opțiunea *Security* este obligatorie pentru specificarea traficului ca fiind secret, restricționat sau compartimentat.

Checksum

Valoarea acestui câmp este recalculată dacă se modifică antetul. De exemplu, dacă se micșorează timpul de viață, dacă au loc adăugări sau modificări ale opțiunilor, sau dacă are loc fragmentarea. Acest câmp la nivelul internet are rolul de a proteja câmpurile antetului de erorile de transmisie.

Există aplicații care acceptă câțiva biți de date eronați, dar nu acceptă întârzieri de retransmisie. Dacă protocolul IP ar forța corecția datelor, aceste aplicații nu ar putea fi suportate.

Errors

Erorile apărute pot fi raportate cu ajutorul mesajelor ICMP [3].

3.3. Interfețe

Descrierea funcțională a interfeței IP cu utilizatorul este, în cel mai bun caz, fictivă, deoarece fiecare sistem de operare va oferi diferite facilități. În consecință, trebuie să avertizăm cititorii că implementări IP diferite ar putea avea diferite interfețe cu utilizatorul. Totuși, orice implementare IP trebuie să ofere un set minim de servicii pentru a se garanta faptul că toate implementările IP suportă aceeași ierarhie de protocoale. Această secțiune specifică interfețele funcționale cerute tuturor implementărilor IP.

Protocolul IP are interfață pe de o parte cu rețeaua locală și pe de altă parte cu un protocol de nivel mai înalt sau cu o aplicație. În cele ce urmează, protocolul de nivel mai înalt sau aplicația (sau chiar un program al unei porți(gateway)) vor fi numite "utilizator", întrucât folosesc modulul internet. Deoarece protocolul IP e orientat pe datagrame, există o memorie sau o stare minimă menținută între transmisiile succesive de datagrame, și fiecare apel al modulului IP de către utilizator trebuie să ofere toate informațiile necesare pentru ca IP-ul să execute serviciul cerut.

Un exemplu de interfață de nivel înalt

Cele două exemple de apeluri de mai jos satisfac cerințele de comunicare ale utilizatorului, adresate modulului IP ("=>" înseamnă întoarce):

```
SEND (src, dst, prot, TOS, TTL, BufPTR, len, Id, DF, opt => result)
```

unde:

```
src = adresa sursă
dst = adresa destinație
prot = protocolul
TOS = tipul serviciului (type of service)
TTL = timpul de viață (time to live)
BufPTR = pointerul la buffer
len = lungimea buffer-ului
Id = identificatorul
DF = flag-ul "Don't Fragment"
opt = opțiunile
result = răspunsul:
    OK = datagrama a fost trimisă
    Error = eroare la argumente sau la rețeaua locală
```

Se observă că precedența este inclusă în TOS și securitatea/compartimentarea sunt transmise ca opțiuni.

RECV (BufPTR, prot, => result, src, dst, TOS, len, opt)

unde:

BufPTR = pointerul la buffer
 prot = protocolul
 result = răspunsul:
 OK = datagrama a fost primită
 Error = eroare la argumente
 len = lungimea buffer-ului
 src = adresa sursă
 dst = adresa destinație
 TOS = tipul serviciului (*type of service*)
 opt = opțiunile

Când utilizatorul trimite o datagramă, apelează procedura SEND, oferind toate argumentele. Modulul IP, la primirea acestui apel, verifică argumentele și pregătește și trimite mesajul. Dacă argumentele sunt valide și datagrama este acceptată de rețeaua locală, apelul returnează cu succes. Dacă argumentele nu sunt valide sau datagrama nu este acceptată de rețeaua locală, apelul returnează fără succes. În acest caz, trebuie alcătuit un raport rezonabil privind cauza problemei, dar detaliile unui astfel de raport sunt lăsate pe seama implementărilor individuale.

Când o datagramă ajunge la modulul IP din rețeaua locală, poate să existe sau nu un apel RECV în așteptare lansat de utilizator. În primul caz, apelul este satisfăcut prin pasarea informațiilor din datagramă către utilizator. În al doilea caz, utilizatorul este anunțat că există o datagramă în așteptare. Dacă utilizatorul adresat nu există, un mesaj de eroare ICMP este trimis către sursa datagramei, iar datele sunt ignorate.

Înștiințarea utilizatorului se poate face printr-o pseudo-întrerupere sau un mecanism similar, în funcție de mediul sistemului de operare ce conține implementarea.

Un apel RECV al utilizatorului poate fi satisfăcut imediat de o datagramă aflată în așteptare, sau apelul poate fi trecut în așteptare până la sosirea unei datagrame.

Adresa sursă este inclusă în apelul de trimitere, deoarece gazda sursă poate avea mai multe adrese (mai multe conexiuni fizice sau mai multe adrese logice). Modulul internet trebuie să verifice ca adresa sursă să fie una din adresele valide pentru această gazdă.

Deasemenea, o implementare poate permite sau poate impune un apel către modulul internet pentru a indica interesul în sau utilizarea rezervată a unei clase de datagrame (de exemplu, toate datagramele ce au o anumită valoare în câmpul *Protocol*).

Această secțiune caracterizează din punct de vedere funcțional o interfață între IP și utilizator. Notățiile folosite sunt similare majorității apelurilor de proceduri sau funcții folosite în limbajele de nivel înalt, dar această folosire nu are intenția de a exclude apeluri sistem de tipul trap (de exemplu: SVC, UUC, EMT), sau orice altă formă de comunicare între procese.

ANEXA A: Exemple și scenarii

Exemplul 1:

Acesta este un exemplu de datagramă ce conține un minim de informații:

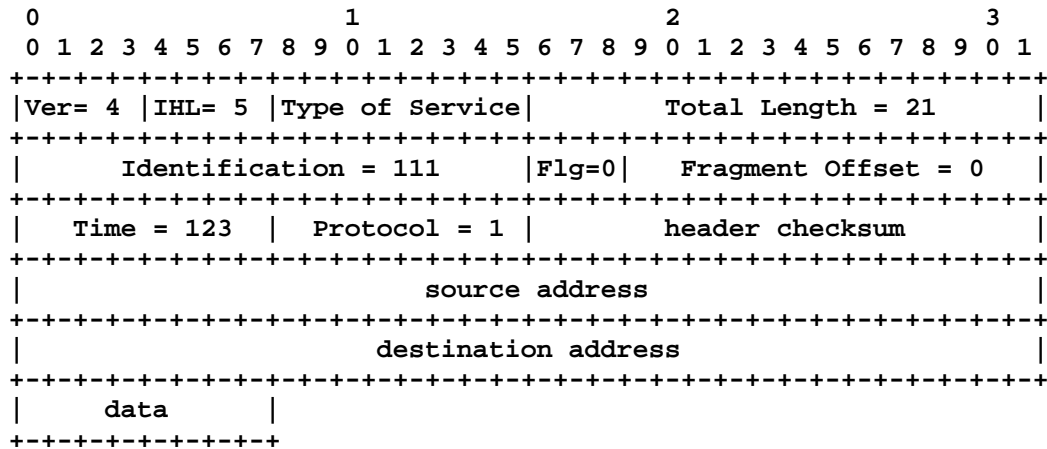


Figura 5. Exemplu de datagramă

Se observă că fiecare poziție marcată reprezintă un bit.

Aceasta este o datagramă în versiunea 4 a protocolului IP; antetul este format din cinci cuvinte pe 32 de biți, iar lungimea totală a datagramei este de 21 de octeți. Această datagramă este completă (nu este un fragment).

Exemplul 2:

În acest exemplu prezentăm mai întâi o datagramă de lungime moderată (452 de octeți de date), apoi două fragmente ce ar putea rezulta din fragmentarea datagramei, dacă dimensiunea maximă permisă pentru o transmisie ar fi de 280 de octeți.

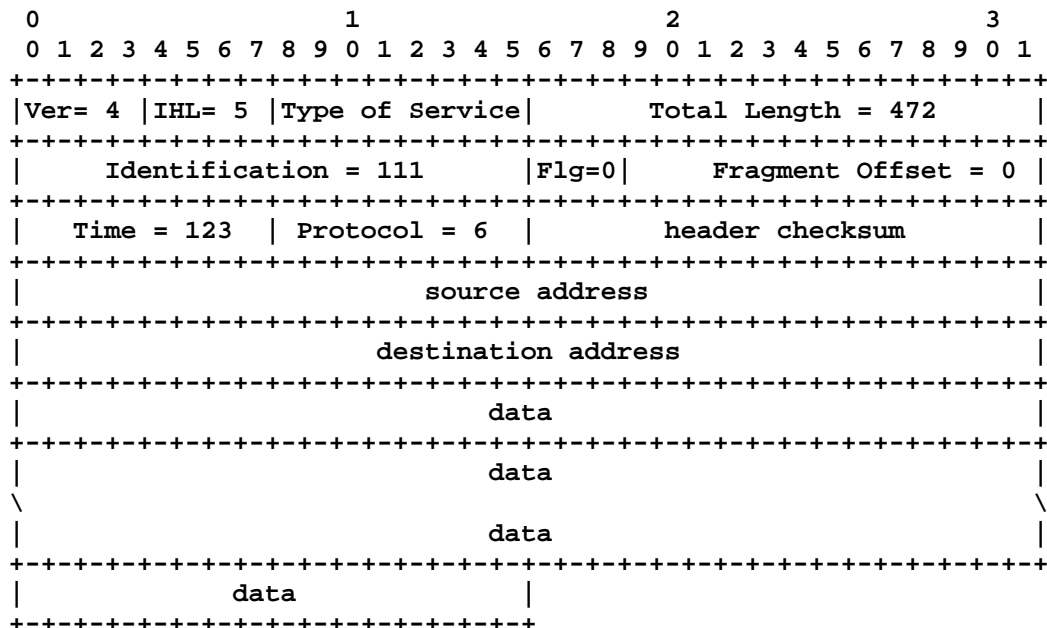


Figura 6. Exemplu de datagramă

Primul fragment care rezultă din tăierea datagramei după 256 de octeți de date:

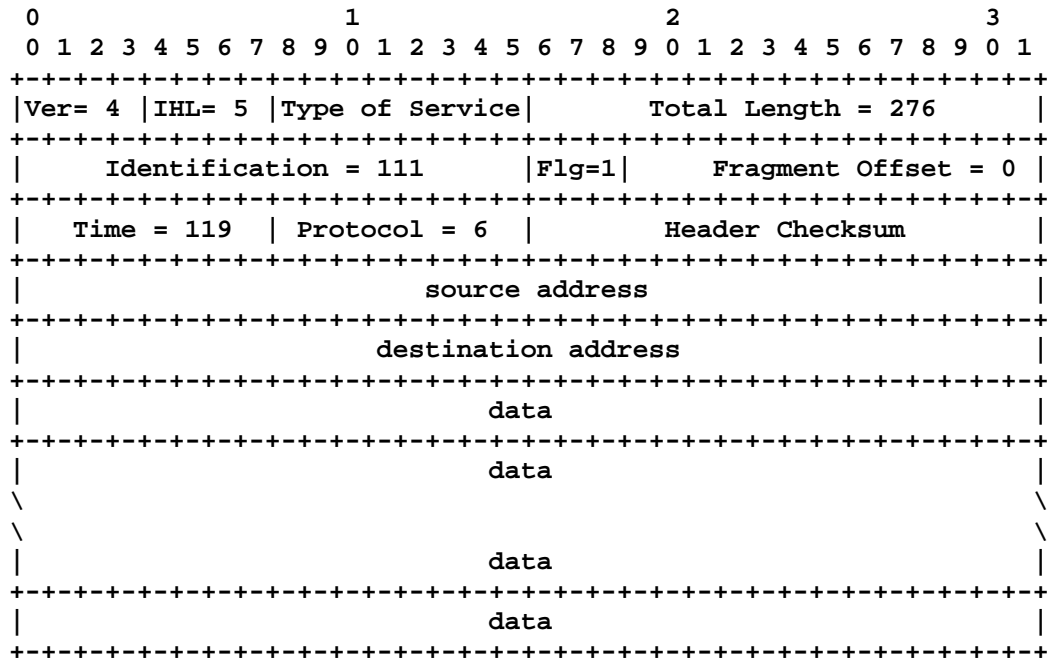


Figura 7. Exemplu de fragment internet

Al doilea fragment.

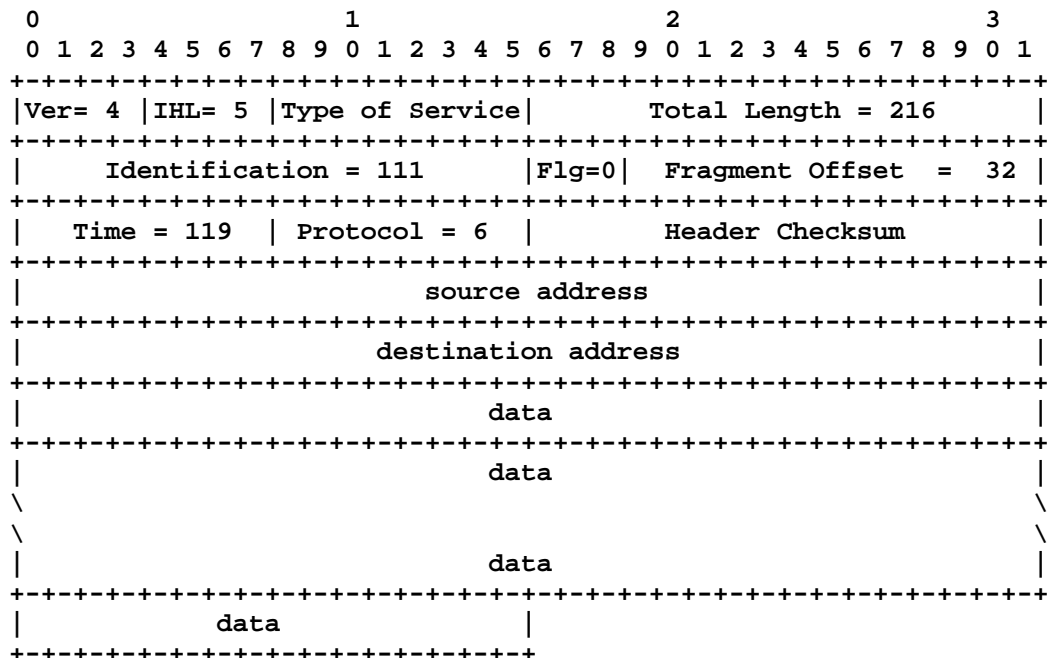


Figura 8. Exemplu de fragment internet

Exemplul 3:

Acest exemplu prezintă o datagramă care conține opțiuni:

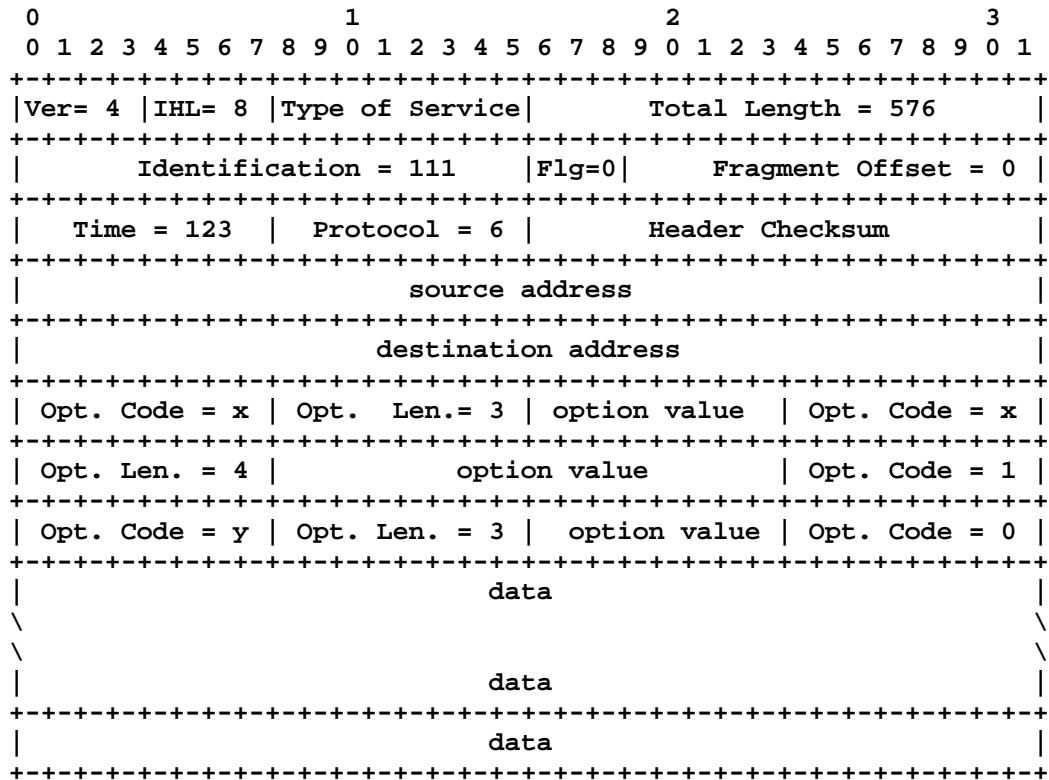


Figura 9. Exemplu de datagramă

ANEXA B: Ordinea de transmitere a datelor

Ordinea de transmitere a antetului și a datelor descrise în acest document este rezolvată la nivel de octet. Când o diagramă prezintă un grup de octeți, ordinea de transmitere a acestora este ordinea normală în care octeții sunt citați în limba engleză. De exemplu, în diagrama următoare octeții sunt transmiși în ordinea în care sunt numerotați.

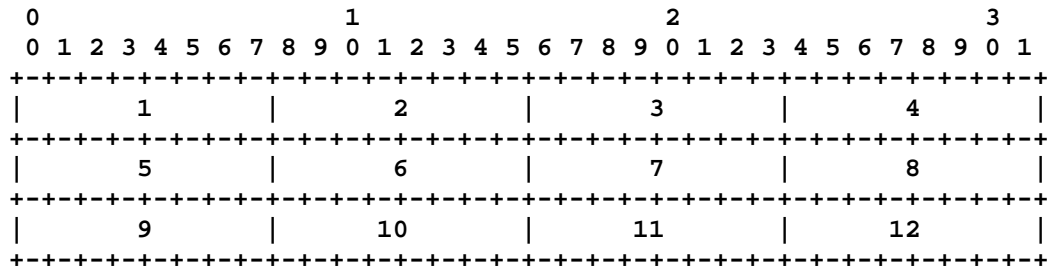


Figura 10. Ordinea de transmitere a octeților

Atunci când un octet reprezintă o cantitate numerică, cel mai din stânga bit al diagramei este cel mai semnificativ. Mai exact, bitul etichetat cu 0 este cel mai semnificativ. De exemplu, diagrama de mai jos reprezintă valoarea zecimală 170.

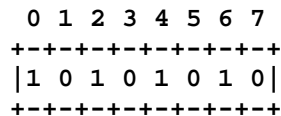


Figura 11. Semnificația biților

În mod similar, când un câmp format din mai mulți octeți reprezintă o cantitate numerică, bitul cel mai din stânga al întregului câmp este cel mai semnificativ. Când o cantitate reprezentată pe mai mulți octeți este transmisă, cel mai semnificativ octet este transmis primul.

GLOSAR

1822

BBN Report 1822, "The Specification of the Interconnection of a Host and an IMP". Specificarea interfeței dintre o gazdă și rețeaua ARPANET.

ARPANET leader

Informațiile de control pentru un mesaj ARPANET la interfața gazdă-IMP.

ARPANET message

Unitatea de transmisie între o gazdă și un IMP în rețeaua ARPANET. Dimensiunea maximă este de în jur de 1012 octeți (8096 biți).

ARPANET packet

O unitate de transmisie folosită pe plan intern în rețeaua ARPANET între IMP-uri. Dimensiunea maximă este de în jur de 126 octeți (1008 biți).

Destination

Adresa destinație, un câmp al antetului internet.

DF

Bitul *Don't Fragment* din câmpul de flag-uri.

Flags

Un câmp al antetului internet ce conține diferite flag-uri de control.

Fragment Offset

Acest câmp al antetului internet indică locul unui fragment într-o datagramă internet.

GGP

Protocolul *Gateway to Gateway*, folosit mai ales între porți pentru a controla rutarea și alte funcții ale porților.

Header

Informații de control aflate la începutul unui mesaj, segment, datagramă, pachet sau bloc de date.

ICMP

Internet Control Message Protocol, implementat în modulul internet, este folosit între porți (*gateways*) și gazde, dar și între diferite gazde pentru a raporta erorile și pentru a da sugestii de rutare.

Identification

Un câmp al antetului internet ce conține valoarea de identificare asignată de gazda care trimite datagrama pentru a ajuta la asamblarea fragmentelor datagramelor.

IHL

Câmpul *Internet Header Length* al antetului internet reprezintă lungimea antetului măsurată în cuvinte de câte 32 de biți.

IMP

Interface Message Processor, switch-ul de pachete al rețelei ARPANET.

Internet Address

O adresă sursă sau destinație de 4 octeți (32 de biți) alcătuită din două câmpuri: *Network* (Rețea) și *Local Address* (Adresa locală).

Internet datagram

Unitatea de date schimbată de o pereche de module internet (include antetul internet).

Internet Fragment

O porțiune de date dintr-o datagramă internet cu un antet internet.

Local Address

Adresa locală a unei gazde într-o rețea. Maparea efectivă a unei adrese internet locale la o adresă a unei gazde este destul de generală, permițând mapări de tipul *many to one*.

MF

Flag-ul *More-Fragments* din câmpul de flag-uri al antetului internet.

Module

O implementare, de obicei în software, a unui protocol sau a unei alte proceduri.

More-fragments Flag

Un flag ce indică dacă datagrama respectivă conține sfârșitul unei datagrame, flag din câmpul de flag-uri al antetului internet.

NFB

Numărul de blocuri (*The Number of Fragment Blocks*) dintr-o porțiune de date a unui fragment internet. Mai exact, lungimea unei porțiuni de date măsurată în unități de câte 8 octeți.

Octet

Un octet ce conține opt biți.

Options

Câmpul *Options* al antetului internet poate conține diverse opțiuni și fiecare opțiune poate avea o lungime de mai mulți octeți.

Padding

Câmpul *Padding* al antetului internet este folosit pentru a asigura faptul că datele încep de la un octet multiplu de 32 de biți. Valoarea de padding este zero.

Protocol

Un câmp al antetului internet; în acest document, identificatorul următorului protocol de nivel mai înalt.

Rest

Porțiunea ce conține adresa locală în cadrul unei adrese internet.

Source

Un câmp al antetului internet; adresa sursă.

TCP

Transmission Control Protocol: un protocol *host-to-host* pentru comunicarea sigură în medii internet.

TCP Segment

Unitatea de date schimbată între modulele TCP(include antetul TCP).

TFTP

Trivial File Transfer Protocol: Un protocol simplu de transmitere a fișierelor bazat pe UDP.

Time to Live

Un câmp al antetului internet ce indică o limită superioară a timpului în care o datagramă internet poate exista.

TOS

Tipul serviciului (*Type of Service*).

Total Length

Câmpul *Total Length* din antetul internet reprezintă lungimea datagrammei în octeți, incluzând antetul internet și datele.

TTL

Timpul de viață (*Time to Live*).

Type of Service

Un câmp al antetului internet ce indică tipul (sau calitatea) serviciului pentru o datagramă internet.

UDP

User Datagram Protocol: Un protocol la nivel de utilizator pentru aplicații orientate pe tranzacții.

User

Utilizatorul protocolului internet. Acesta poate fi un modul al unui protocol de nivel mai înalt, o aplicație sau un program al unei porți (*gateway*).

Version

Câmpul *Version* indică formatul antetului internet.

REFERINȚE

- [1] Cerf, V., "The Catenet Model for Internetworking", Information Processing Techniques Office, Defense Advanced Research Projects Agency, IEN 48, Iulie 1978.
- [2] Bolt Beranek și Newman, "Specification for the Interconnection of a Host and an IMP", BBN Technical Report 1822, Revizuită Mai 1978.
- [3] Postel, J., "Internet Control Message Protocol - DARPA Internet Program Protocol Specification", RFC 792, USC/Information Sciences Institute, Septembrie 1981.
- [4] Shoch, J., "Inter-Network Naming, Addressing, and Routing", COMPCON, IEEE Computer Society, Toamna 1978.
- [5] Postel, J., "Address Mappings," RFC 796, USC/Information Sciences Institute, Septembrie 1981.
- [6] Shoch, J., "Packet Fragmentation in Inter-Network Protocols", Computer Networks, v. 3, n. 1, Februarie 1979.
- [7] Strazisar, V., "How to Build a Gateway", IEN 109, Bolt Beranek și Newman, August 1979.
- [8] Postel, J., "Service Mappings," RFC 795, USC/Information Sciences Institute, Septembrie 1981.
- [9] Postel, J., "Assigned Numbers," RFC 790, USC/Information Sciences Institute, Septembrie 1981.