Acesta este capitolul 10 — *Internetul* — al ediţiei electronică a cărţii *Reţele de calculatoare*, publicată la Casa Cărţii de Ştiinţă, în 2008, ISBN: 978-973-133-377-9.

Drepturile de autor aparţin subsemnatului, Radu-Lucian Lupşa.

Subsemnatul, Radu-Lucian Lupşa, acord oricui doreşte dreptul de a copia conţinutul acestei cărţi, integral sau parţial, cu condiţia atribuirii corecte autorului şi a păstrării acestei notiţe.

Cartea, integrală, poate fi descărcată gratuit de la adresa
`http://www.cs.ubbcluj.ro/~rlupsa/works/retele.pdf`

© 2008, Radu-Lucian Lupşa

© 2008, Radu-Lucian Lupşa

Capitolul 10

Internetul

Denumirea *Internet* desemnează două lucruri: pe de o parte un protocol de nivel reţea (*Internet Protocol*, *IP*, *protocolul Internet*), iar pe de altă parte reţeaua Internet, care este o reţea la scară mondială bazată pe protocolul Internet.

Capitolul de faţă prezintă:

- protocolul Internet (IP), împreună cu celelalte protocoale de bază ale reţelelor de tip Internet (TCP, DNS, ARP, etc.);
- câteva aspecte administrative legate de reţeaua mondială Internet.

10.1. Arhitectura reţelei

Facem în continuare o scurtă trecere în revistă a conceptelor de bază ale unei reţele bazate pe protocolul Internet. Aceste concepte vor fi detaliate în paragrafele care urmează.

Serviciul de comunicaţie oferit de o reţea Internet este de tip datagrame; în terminologia Internet acestea se numesc *pachete*.

Ca orice reţea (vezi capitolul 5), o reţea Internet este alcătuită din *noduri*, interconectate între ele. Într-o reţea Internet, toate nodurile pot acţiona ca noduri finale (adică să fie sursă sau destinaţie pentru comunicaţie). Sunt numite *staţii* (engl. *hosts*) nodurile ce nu pot acţiona ca noduri intermediare şi *rutere* nodurile ce pot acţiona ca noduri intermediare.

Staţiile sunt în mod uzual calculatoare (PC-uri, mainframe-uri), dispozitive mobile (PDA-uri), imprimantele de reţea sau alte dispozitive. Remarcăm că switch-urile Ethernet sunt noduri IP numai dacă sunt configurabile. În acest caz, ele au doar rol de staţie şi doar în scopul de-a putea fi contactate în vederea configurării.

© 2008, Radu-Lucian Lupşa

Nodurile intermediare sunt fie PC-uri, fie dispozitive dedicate (*rutere* dedicate).

Legăturile directe pot fi realizate prin linii seriale, linii telefonice cu modemuri, reţele locale IEEE 802, cablu TV, etc. Modul de utilizare a fiecărui tip de legătură directă de către o reţea Internet este standardizat prin standarde auxiliare (§ 10.5). Există chiar un standard [RFC 1149, 1990] de utilizare ca legături directe a porumbeilor călători; deşi standardul a fost publicat ca o glumă de 1 aprilie, el ilustrează foarte bine independenţa între nivele într-o reţea.

Din punctul de vedere al unei reţele Internet, o legătură directă este orice fel de canal de comunicaţie pe care reţeaua de tip Internet o poate folosi.

Fiecare nod este identificat prin una sau mai multe *adrese IP*. Cu excepţia unor adrese cu rol special, o adresă IP identifică unic un nod. Unele noduri, în special cele intermediare, au mai multe adrese IP asociate.

Adresele IP sunt arareori folosite direct de utilizatorii umani. În locul lor se utilizează *numele de domeniu*. Corespondenţa între un nume de domeniu şi adresa IP se realizează cu ajutorul sistemului DNS (*Domain Name Service*), descris în § 10.4.

Protocolul Internet a fost proiectat pentru a asigura o toleranţă deosebit de mare la pene. După căderea unor noduri sau a unor legături, dacă mai există totuşi un drum între două noduri el va fi găsit şi utilizat în cele din urmă. Această toleranţă la pene vine cu un preţ: nu există garanţii cu privire la întârzierea maximă în livrarea unui pachet sau debit minim garantat; ba chiar este posibil ca un pachet să fie pierdut complet (acest lucru se poate întâmpla cu pachetele surprinse pe drum de o pană, precum şi în caz de încărcare mare a reţelei), să ajungă în dublu exemplar sau două pachete să ajungă la destinaţie în ordine inversă a trimiterii.

Este sarcina nivelelor superioare să se descurce în aceste condiţii. În acest scop, între aplicaţie şi nivelul reţea este plasat un nivel intermediar, *nivelul transport*, cu rolul de-a furniza aplicaţiei un serviciu mai potrivit.

10.2. Protocolul IP

Protocolul Internet (engl. *Internet Protocol — IP*) descrie formatul pachetelor şi câteva aspecte privind activitatea nodurilor reţelei.

Protocolul IP are două versiuni aflate curent în uz: versiunea 4 (cea mai utilizată în prezent, numită prescurtat *IPv4*) standardizată prin [RFC 791, 1981] şi versiunea 6 (care se răspândeşte relativ încet, numită prescurtat *IPv6*) standardizată prin [RFC 2460, 1998].

© 2008, Radu-Lucian Lupşa

10.2.1. Structura pachetului IP

Un pachet IP este alcătuit dintr-un antet fix, un număr variabil de opţiuni şi, în final, datele utile.

Antetul fix conţine datele necesare pentru dirijarea pachetului. Conţinutul antetului fix este dat în tabelele 10.1 (pentru versiunea 4) şi 10.2 (pentru versiunea 6). Semnificaţia diferitelor câmpuri va fi descrisă în paragrafele care urmează.

Nume câmp	Lungime (biţi)	Rol
Versiune	4	Versiunea protocolului; valoarea este fixă: 4.
IHL	4	Lungimea antetului, inclusiv opţiunile, în grupuri de 32 biţi (valoarea minimă este 5, adică 160 biţi).
TOS	8	Tip serviciu (vezi § 10.2.6.2).
Lungime totala	16	Lungimea totală, antet plus date utile, în octeţi.
Identificare	16	Identificator pentru reasamblarea fragmentelor (vezi § 10.2.6.1).
Rezervat	1	Rezervat pentru extinderi ulterioare; are valoarea 0.
Nu fragmenta	1	vezi § 10.2.6.1.
Ultimul fragment	1	Marchează ultimul fragment sau un pachet nefragmentat (vezi § 10.2.6.1).
Deplasament	13	Deplasament pentru reasamblarea fragmentelor.
Timp de viaţă	8	Timpul rămas până la distrugerea pachetului (vezi § 10.2.5.3).
Protocol	8	Identificarea protocolului de nivel superior căruia îi aparţin datele utile.
Suma de control	16	Suma de control a antetului.
Adresă sursă	32	Adresa nodului ce a creat pachetul.
Adresă destinaţie	32	Adresa destinatarului final al pachetului.

Tabelul 10.1: Antetul IP versiunea 4

Opţiunile sunt informaţii pentru dirijarea pachetului pentru cazuri mai speciale; deoarece aceste informaţii nu sunt necesare decât pentru anumite tipuri de pachete, ele sunt prezente doar în pachetele în care este nevoie de

© 2008, Radu-Lucian Lupşa

Nume câmp	Lungime (biţi)	Rol
Versiune	4	Versiunea protocolului IP. Valoarea este fixă: 6.
Clasă trafic	8	tip serviciu (vezi § 10.2.6.2).
Etichetă flux	20	vezi § 10.3.1.8.
Lungime rest	16	Lungimea pachetului minus antetul fix, în octeţi.
Tip antet următor	8	Dacă există opţiuni, tipul primului antet opţional; altfel, protocolul căruia îi aparţin datele utile.
Limită salturi	8	Numărul maxim de salturi până la distrugerea pachetului (vezi § 10.2.5.3).
Adresa sursă	128	Adresa nodului ce a emis pachetul.
Adresa destinaţie	128	Adresa destinatarului final al pachetului.

Tabelul 10.2: Antetul IP versiunea 6

ele.

Datele utile sunt un şir de octeţi asupra căruia protocolul IP nu impune nici o restricţie, cu excepţia lungimii. Lungimea maximă admisă de protocol este de 65515 octeţi (65535 octeţi pachetul întreg) pentru IPv4 şi 65535 octeţi, inclusiv antetele opţionale, pentru IPv6. Este permis ca unele noduri să nu poată procesa pachete în care datele utile sunt mai lungi de 556 octeţi (576 octeţi tot pachetul) pentru IPv4 şi 1240 octeţi (1280 octeţi tot pachetul) pentru IPv6 (a se vedea şi § 10.2.6.1).

10.2.2. Bazele dirijării pachetelor IP

10.2.2.1. Subreţele şi interfeţe

O *subreţea* este o mulţime de noduri legate direct fiecare cu fiecare. De exemplu, o reţea Ethernet construită cu cabluri magistrală este o subreţea IP. O reţea Ethernet cu hub-uri sau switch-uri este de asemenea o subreţea IP întrucât, din punctul de vedere al calculatorului la care este ataşată o placă de reţea, o reţea Ethernet construită cu cablu magistrală se comportă identic cu o reţea construită cu hub-uri sau switch-uri. Ca alt exemplu, o linie serială construieşte o subreţea cu două calculatoare.

Interfaţa de reţea este un concept abstract care desemnează legătura dintre un nod şi o subreţea. În cazul în care legătura directă este realizată de

o reţea IEEE 802, interfaţa de reţea este placa de reţea împreună cu driver-ul ei.

Fiecare interfaţă de reţea are propria adresă IP. Ca urmare, un nod ce are n plăci de reţea va avea n adrese IP distincte.

Are sens să vorbim despre *interfeţele membre ale unei subreţele*, ca fiind interfeţele prin care nodurile din subreţea sunt conectate la acea subreţea. Adresele IP dintr-o subreţea sunt adresele IP ale interfeţelor din acea subreţea.

10.2.2.2. Prefixul de reţea

Fiecare subreţea are asociat un *prefix de reţea*, adică un anumit şir de biţi de lungime mai mică decât lungimea unei adrese IP. Toate adresele IP ale interfeţelor din acea subreţea trebuie să înceapă cu acel prefix de reţea. Prefixul unei subreţele nu este permis să fie prefix al unei adrese IP din afara acelei subreţele. Ca urmare, un prefix identifică unic o subreţea.

Sufixul unei adrese, adică şirul de biţi din adresă care nu fac parte din prefixul subreţelei, îl vom numi *adresa în cadrul subreţelei*. Numărul de biţi ai sufixului determină numărul de noduri ce pot fi membre ale subreţelei.

Adresele în care sufixul este format numai din biţi 0 sau numai din biţi 1 (aşadar două adrese pentru fiecare subreţea) sunt rezervate şi nu pot fi asignate nodurilor reţelei (a se vedea şi [RFC 1700, 1994]).

Exemplul 10.1: Pentru simplificarea exemplului vom presupune că adresele IP sunt doar de 8 biţi. Presupunem că o subreţea ar avea prefixul de reţea 10110. Adresa 10110010, dacă există, trebuie să desemneze o interfaţă din acea reţea.

Adresa 10111010 nu poate face parte din acea subreţea, deoarece nu începe cu prefixul subreţelei. Notăm că un nod care are o interfaţă în subreţeaua 10110 şi o interfaţă în altă reţea ar putea avea adresa 10111010 pe cea de-a doua interfaţă.

Din subreţeaua considerată, cu prefixul 10110, pot face parte adresele, în număr de $2^3 = 8$, din intervalul 10110000–10110111. Adresele 10110000 şi 10110111 sunt rezervate; rămân deci 6 adrese ce pot fi asignate nodurilor subreţelei.

Un exemplu de asignare a adreselor este prezentat în figura 10.1. Pătrăţelele numerotate reprezintă calculatoarele, iar liniile reprezintă legăturile directe, figurate aici ca şi când ar fi realizate prin cabluri magistrală. De remarcat că nodul cu numărul 3 are două adrese, 10110001 şi 10111010, câte una pentru fiecare interfaţă.

© 2008, Radu-Lucian Lupşa

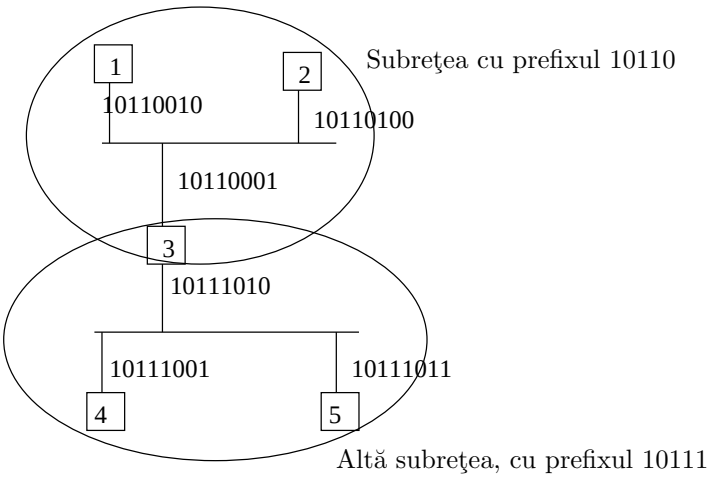

Figura 10.1: O reţea formată din două subreţele. Vezi exemplul 10.1

10.2.2.3. Tabela de dirijare

La primirea unui pachet IP, un nod execută următorul algoritm:

1. dacă adresa destinaţie este una din adresele nodului curent, pachetul este livrat local (nivelului superior);
2. altfel, dacă adresa destinaţie este adresa unui vecin direct, pachetul este livrat direct acelui vecin;
3. altfel, se determină vecinul direct cel mai apropiat de destinatarul pachetului şi i se dă pachetul, urmând ca acesta să-l trimită mai departe.

Pentru pasul 2, este necesar ca nodul să determine dacă adresa destinaţie corespunde unui vecin direct şi care este interfaţa prin care se realizează legătura. Livrarea efectivă este realizată de interfaţa de reţea; acesteia i se dă pachetul şi adresa IP a vecinului.

Pentru pasul 3, trebuie determinat în primul rând vecinul direct căruia i se va trimite pachetul. Dacă acesta are mai multe interfeţe, trebuie utilizată interfaţa prin intermediul căruia el este vecin nodului curent. O dată determinată adresa interfeţei, trimiterea pachetului se face ca la pasul 2.

Deciziile de la paşii 2 şi 3 se iau pe baza *tabelei de dirijare* a nodului curent. O tabelă de dirijare constă dintr-o mulţime de *reguli de dirijare*. Fiecare regulă asociază o *ţintă* unui *grup de adrese* destinaţie.

Grupul de adrese este specificat printr-un prefix de reţea. Pentru un pachet dat se aplică acea regulă de dirijare în care prefixul ce specifică grupul este prefix al adresei destinaţie a pachetului. Dacă există mai multe astfel de

reguli de dirijare, se aplică regula cu prefixul cel mai lung — adică cea mai specifică dintre regulile de dirijare aplicabile.

Ţinta poate fi fie o interfaţă, fie o adresă IP.

Dacă ţinta este o interfaţă, destinaţia trebuie să fie un vecin direct, accesibil prin acea interfaţă; în acest caz pachetul de dirijat este livrat direct destinatarului prin interfaţa dată în regulă, conform pasului 2.

Dacă ţinta este o adresă, aceasta trebuie să fie adresa unei interfeţe vecine. În acest caz pachetul de dirijat este trimis nodului vecin a cărui adresă este specificată în tabela de dirijare. Nodul vecin respectiv poartă denumirea de *gateway* şi trebuie să fie configurat să acţioneze ca nod intermediar.

De notat că adresa sursă şi adresa destinaţie din antetul IP nu se modifică în cursul acestei proceduri. Sursa rămâne nodul care a emis pachetul, iar destinaţia rămâne nodul căruia trebuie să-i fie livrat în cele din urmă pachetul. Atunci când modulul de reţea pasează unei interfeţe de reţea un pachet în vederea transmiterii pachetului către un nod vecin, modulul de reţea va comunica interfeţei două lucruri: pachetul, în care adresa destinaţie reprezintă destinatarul final, şi adresa vecinului direct căruia interfaţa îi va livra pachetul. Acesta din urmă poate fi diferit faţă de destinatarul final dacă este doar un intermediar pe drumul către destinatarul final.

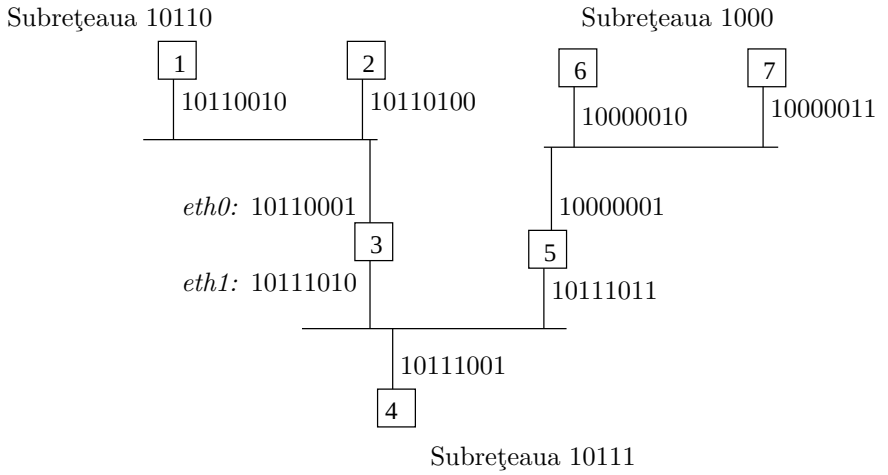

Figura 10.2: O reţea pentru exemplul 10.2

Exemplul 10.2: Fie reţeaua din figura 10.2, formată din trei subreţele. Pentru nodul nr. 3, au fost figurate şi numele interfeţelor de reţea: *eth0* către subreţeaua de sus şi *eth1* către subreţeaua de jos. Tabela de dirijare a nodului 3 este cea ilustrată în tabelul 10.3.

© 2008, Radu-Lucian Lupşa

Nr. crt.	Grup de adrese (prefix)	Ţinta
1.	10110	interfaţa *eth0*
2.	10111	interfaţa *eth1*
3.	1000	nodul 10111011

Tabelul 10.3: Tabela de dirijare pentru nodul 3 din figura 10.2 (exemplul 10.2).

Considerăm că nodul 3 primeşte un pachet cu destinaţia 10110010 (nodul 1). Singura regulă aplicabilă este regula 1, deoarece 10110 este prefix pentru 10110010. Conform acestei reguli, pachetul poate fi livrat direct prin interfaţa *eth0*.

Fie acum un pachet cu destinaţia 10000010 (nodul 6). Regula aplicabilă este regula 3. Conform acesteia, pachetul trebuie trimis nodului cu adresa 1011101, urmând ca acesta să-l trimită mai departe. Modulul IP caută în continuare regula aplicabilă pentru destinaţia 10111011 şi găseşte regula 2, conform căreia pachetul se trimite prin interfaţa *eth1*. Prin urmare, pachetul destinat lui 10000010 va fi trimis lui 10111011 prin interfaţă *eth1* (urmând ca nodul 5 să-l livreze mai departe nodului 6).

De remarcat că nodurile ce apar ca ţintă în regulile tabelelor de dirijare trebuie specificate prin adresele interfeţelor accesibile direct din nodul curent. În exemplul 10.2, este esenţial ca, în ultima regulă a tabelei de dirijare a nodului 3, nodul 5 să fie specificat prin adresa 10111011 şi nu prin adresa 10000001. Nodul cu adresa 10111011 este accesibil prin interfaţa *eth1*, conform regulii 2; dacă ar fi fost specificat prin adresa 10000001, nodul 3 nu ar fi putut determina cum să-i trimită pachetul.

În majoritatea cazurilor, tabela de dirijare are o regulă numită *implicită*, corespunzătoare prefixului vid şi, ca urmare, aplicată pentru pachetele pentru care nu este aplicabilă nici o altă regulă. Această regulă, dacă există, are totdeauna ca ţintă o adresă IP a unui nod vecin al nodului curent. Acest nod (de fapt, această adresă IP) poartă denumirea de *default gateway*.

10.2.3. Scrierea ca text a adreselor şi prefixelor

10.2.3.1. Scrierea adreselor IP

Atunci când o adresă IP este scrisă pe hârtie sau într-un fişier text, se afişează pe ecran sau se citeşte de la tastatură, adresa este scrisă într-un format standard descris în continuare.

© 2008, Radu-Lucian Lupşa

Adresele IP versiunea 4, care sunt şiruri de 32 de biţi, se scriu ca şir de 4 numere, scrise în baza 10, separate prin puncte. Fiecare număr este de fapt valoarea câte unui grup de 8 biţi, văzut ca număr. Această scriere se numeşte *notaţie zecimală cu punct*.

Pe lângă notaţia zecimală cu punct, adresele IP versiunea 4 pot fi scrise în *notaţia zecimală simplă*: se scrie direct valoarea adresei ca număr, scris în baza 10.

Exemplul 10.3: Fie adresa 1100-0000-1010-1000-0000-0000-0010-0010 (liniuţele au fost scrise numai pentru uşurarea citirii). Notaţia zecimală cu punct este 192.168.0.34. Notaţia zecimală simplă este 3232235554.

Adresele IP versiunea 6 sunt şiruri de 128 de biţi. Scrierea lor obişnuită se face ca un şir de 32 cifre hexa, fiecare reprezentând câte 4 biţi din adresă. Cifrele hexa sunt grupate câte 4, iar grupurile succesive sunt separate prin câte un caracter *două puncte*. Pentru a scurta scrierea, se permit următoarele optimizări:

- zerourile de la începutul unui grup pot să nu fie scrise;
- un grup cu valoarea 0 sau mai multe astfel de grupuri consecutive se pot elimina, împreună cu separatorii *două puncte* dintre ei, rămânând doar două caractere *două puncte* succesive. Acest lucru se poate face într-un singur loc al adresei, altfel s-ar crea evident o ambiguitate.

Exemplul 10.4: O posibilă adresă IPv6 este

```
fe80:0000:0000:0000:0213:8fff:fe4e:fbf4
```

Posibile scrieri prescurtate sunt

```
fe80:0:0:0:213:8fff:fe4e:fbf4
```

sau

```
fe80::213:8fff:fe4e:fbf4
```

Adresa `0:0:0:0:0:0:0:1` se scrie compact `::1`.

Pentru adrese IPv6 alocate în vederea compatibilităţii cu IPv4, este acceptată scrierea în care primii 96 biţi sunt scrişi în format IPv6, iar ultimii 32 de biţi sunt scrişi în format IPv4, separaţi de primii printr-un caracter *două puncte*.

© 2008, Radu-Lucian Lupşa

Exemplul 10.5: Adresa `0:0:0:0:0:0:c100:e122` se poate scrie şi

`0:0:0:0:0:0:193.0.225.34`

sau, mai compact,

`::193.0.225.34`

10.2.3.2. Scrierea prefixelor de reţea

Prefixele de reţea fiind de lungime variabilă, trebuie precizată atât valoarea efectivă a prefixului cât şi lungimea acestuia. Există două notaţii: notaţia cu adresa subreţelei şi lungimea prefixului şi notaţia cu adresa subreţelei şi masca de reţea.

În notaţia cu adresă şi lungime, prefixul se completează cu zerouri la lungimea unei adrese IP (adică la 32 de biţi pentru versiunea 4 şi la 128 de biţi pentru versiunea 6); rezultatul se numeşte *adresa de reţea*. Adresa de reţea se scrie ca şi când ar fi o adresă IP normală, după care se scrie (fără spaţiu) un caracter *slash* (`/`) urmat de lungimea prefixului scrisă ca număr în baza 10.

Exemplul 10.6: Prefixul IPv4 1100-0000-1010-1000-110 se scrie, în notaţia cu adresă de reţea şi lungime (notaţie cu slash) 192.168.192.0/19. Prefixul 1100-0000-1010-1000-1100-0000 se scrie 192.168.192.0/24. De remarcat importanţa specificării lungimii.

În notaţia cu adresă de reţea şi mască de reţea, se scrie mai întâi adresa de reţea, ca şi în cazul scrierii cu adresă şi lungime, după care se scrie (cu un *slash* între ele sau în rubrici separate) aşa-numita *mască de reţea*. Masca de reţea constă dintr-un şir de biţi 1 de lungimea prefixului de reţea urmat de un şir de biţi 0, având în total lungimea unei adrese IP. Mască de reţea, se scrie ca şi când ar fi o adresă IP.

Notaţia cu adresă şi mască se utilizează numai pentru IP versiunea 4.

Exemplul 10.7: Prefixul 1100-0000-1010-1000-110 se scrie, în notaţia cu adresă de reţea şi masca, 192.168.192.0/255.255.224.0. Prefixul 1100-0000-1010-1000-1100-0000 se scrie 192.168.192.0/255.255.255.0.

10.2.4. Alocarea adreselor IP şi prefixelor de reţea

Alocarea adreselor IP pentru reţeaua mondială Internet se realizează de către *Internet Assigned Numbers Authority* (*IANA*). Mai multe despre alocare se găseşte la [IANA,]. Deşi nu este actualizat, este instructiv de citit şi [RFC 1700, 1994].

10.2.4.1. Alocarea pe utilizări

Adresele IPv4 sunt împărţite după cum urmează:

- Adresele cu prefixele 0.0.0.0/8 şi şi 127.0.0.0/8 sunt rezervate. Adresa 127.0.0.1, pentru fiecare nod, desemnează acel nod, cu alte cuvinte un pachet destinat adresei 127.0.0.1 este totdeauna livrat nodului curent. Adresa 0.0.0.0 înseamnă adresă necunoscută; poate fi folosită doar ca adresă sursă în pachete emise de un nod care încearcă să îşi afle propria adresă.
- Adresele cu prefixul 224.0.0.0/4 sunt utilizate ca adrese de multicast (aşa-numita *clasă D*).
- Adresele cu prefixul 240.0.0.0/4 sunt rezervate (aşa-numita *clasă E*).
- Adresele cu prefixele 10.0.0.0/8, 172.16.0.0/12 şi 192.168.0.0/16 sunt numite *adrese private* [RFC 1918, 1996]. Aceste adrese pot fi utilizate intern de oricine, fără să fie necesar alocarea la IANA, însă cu condiţia ca pachetele purtând astfel de adrese ca sursă sau destinaţie să nu ajungă în afara nodurilor gestionate de acea persoană sau instituţie. Aceste adrese se utilizează pentru acele noduri, din reţeaua proprie a unei instituţii, care nu au nevoie de acces direct la Internet. Mai multe detalii despre utilizarea acestor adrese vor fi date în § 10.7.2
- Restul adreselor se alocă normal nodurilor din Internet.

10.2.4.2. Alocarea adreselor şi dirijarea ierarhică

În lipsa oricăror grupări ale adreselor, majoritatea nodurilor din Internet ar trebui să aibă în tabela de dirijare câte o regulă pentru fiecare nod. O asemenea soluţie nu este realizabilă practic la scară mondială. Din această cauză, adresele se alocă instituţiilor doritoare în blocuri de adrese, fiecare bloc având un prefix unic, întocmai ca în cazul subreţelelor.

Un bloc de adrese se alocă unei subreţele sau grup de subreţele interconectate care apar, din restul Internetului, ca o singură subreţea. Din afara subreţelei corespunzătoare unui bloc, toate pachetele destinate adreselor din bloc sunt dirijate identic, conform unei reguli care are ca prefix prefixul blocului. În tabela de dirijare a unui nod oarecare din Internet va fi necesar astfel câte o regulă pentru fiecare bloc, şi nu câte o regulă pentru fiecare nod.

Pentru stabilirea dimensiunilor blocurilor, iniţial adresele IP versiunea 4 au fost împărţite în *clase*:

A: Adresele cu prefixul 0.0.0.0/1 au fost împărţite în 128 blocuri alocabile, fiecare bloc având câte 2^{24} adrese. Lungimea prefixului unui bloc este de 8 biţi.

© 2008, Radu-Lucian Lupşa

B: Adresele cu prefixul 128.0.0.0/2 au fost împărţite în 16384 blocuri de câte 65536 adrese. Prefixul unui bloc este de 16 biţi.

C: Adresele cu prefixul 192.0.0.0/3 au fost împărţite în 2^{21} blocuri de câte 256 adrese. Lungimea prefixului unui bloc este de 24 de biţi.

Ideea împărţirii între clasele A, B şi C era aceea ca, dându-se o adresă IP, să se poată determina lungimea prefixului blocului din care face parte. Acest lucru simplifică mult calcularea tabelelor de dirijare şi chiar căutarea în tabela de dirijare a regulii aplicabile.

Împărţirea în clase s-a dovedit prea inflexibilă. Pe de o parte, împărţirea este ineficientă, ducând la alocarea câte unui bloc de clasă B (adică 65536 adrese) pentru instituţii care nu aveau nevoie de mai mult de câteva sute de adrese. Pe de altă parte, nu există nici o corelaţie între blocurile de adrese alocate unor instituţii diferite dar din aceeaşi zonă geografică; în consecinţă, pentru majoritatea ruterelor din Internet este nevoie de câte o regulă de dirijare pentru fiecare instituţie căreia i s-a alocat un bloc de adrese.

Ca urmare s-a decis o nouă schemă de alocare a blocurilor de adrese. Noua schemă se numeşte CIDR (engl. *Classless InterDomain Routing*) şi este descrisă în [RFC 1518, 1993].

În schema CIDR, un prefix de bloc poate avea orice lungime. O instituţie ce doreşte acces Internet poate solicita alocarea unui bloc de adrese, cu un număr de adrese egal cu o putere a lui 2.

O instituţie care furnizează acces Internet altor instituţii este încurajată să aloce mai departe, din blocul alocat ei, sub-blocuri pentru instituţiile cărora le oferă acces Internet. Astfel, din afara reţelei furnizorului de acces Internet, reţeaua furnizorului împreună cu toţi clienţii lui se vede ca o singură subreţea în care toate adresele au acelaşi prefix.

CIDR mai prevede o grupare a blocurilor pe continente, astfel încât pentru un nod aflat pe un continent toate (sau majoritatea) adreselor de pe un alt continent să se dirijeze conform unei singure reguli. Această grupare este aplicabilă doar adreselor care nu erau deja alocate la momentul introducerii CIDR; CIDR nu şi-a pus problema realocării adreselor deja alocate.

Pentru adresele IP versiunea 6 se foloseşte numai schema CIDR.

10.2.5. Erori la dirijare şi protocolul ICMP

Protocolul ICMP (*Internet Control Message Protocol*) are scop diagnosticarea diverselor probleme legate de dirijarea pachetelor IP. Fiind strâns legat de protocolul IP, ICMP are două versiuni, ICMP pentru IPv4, descris în [RFC 792, 1981] şi numit uneori ICMPv4, şi ICMP pentru IPv6, descris în [RFC 2463, 1998] şi numit şi ICMPv6.

Protocolul constă în transmiterea, în anumite situaţii, a unor *pachete ICMP*. Un pachet ICMP este un pachet IP în care câmpul *protocol* are valoarea 1 pentru ICMPv4, respectiv 58 pentru ICMPv6, iar zona de date utile este structurată conform standardului ICMP.

Pachetele ICMP sunt de obicei generate de modulul de reţea al unui nod, ca urmare a unei erori apărute în livrarea unui pachet IP. Pachete ICMP mai pot fi generate şi de programe utilizator, prin intermediul *socket*-urilor de tip `SOCK_RAW`. Astfel de aplicaţii servesc la testarea funcţionării reţelei.

O dată generat, un pachet ICMP este transmis prin reţea ca orice alt pachet IP. Ajuns la destinaţie, modulul de reţea (IP) al nodului destinaţie examinează câmpul *protocol* şi, constatând că este vorba de un pachet ICMP, îl livrează modulului ICMP al nodului destinaţie. Modulul ICMP trebuie să fie prezent şi funcţional în orice nod IP; în implementările obişnuite este parte a nucleului sistemului de operare al calculatorului ce constituie nodul.

Datele utile sunt formatate conform standardului ICMP şi încep cu doi întregi pe câte 8 biţi reprezentând tipul şi subtipul mesajului ICMP (vezi tabelul 10.4). Formatul restului pachetului depinde de tipul mesajului ICMP; în majoritatea cazurilor, este prezentă o copie a primilor câteva zeci de octeţi din pachetului IP care a dus la generarea pachetului ICMP.

Situaţiile ce duc la generarea pachetelor ICMP, precum şi acţiunile întreprinse de un nod la primirea unui pachet ICMP, sunt descrise în paragrafele următoare.

10.2.5.1. Pachete nelivrabile

Un nod declară un pachet nelivrabil dacă:

- nici o regulă din tabela de dirijare a nodului nu este aplicabilă destinaţiei pachetului; sau
- interfaţa de reţea prin care trebuie trimis pachetul nu este funcţională sau nu poate livra pachetul destinatarului (destinatarul nu răspunde).

În aceste cazuri, nodul curent trimite un pachet ICMP, având:

- *adresa sursa:* adresa nodului curent,
- *adresa destinaţie:* adresa sursă a pachetului nelivrabil,
- *tip: destination unreachable.*

Pachetul ICMP mai cuprinde antetul pachetului ce nu a putut fi livrat. Destinatarul pachetului ICMP, care este de fapt sursa pachetului nelivrabil, trebuie să analizeze antetul pachetului returnat şi să informeze nivelul superior (probabil TCP sau UDP) asupra problemei.

© 2008, Radu-Lucian Lupşa

Tip	Subtip	Ce semnalizează
3 — Destination unreachable	0 — network unreachable 1 — host unreachable 3 — protocol unreachable	pachet nelivrabil, conform § 10.2.5.1
	4 — fragmentation needed	pachet prea mare şi flagul *nu fragmenta* setat; vezi § 10.2.6.1
	5 — source route failed	pachetul a avut opţiunea *dirijare de către sursă* şi ruta specificată este invalidă.
11 — time exceeded	0 — TTL exceeded	pachetul se află de prea mult timp în reţea (probabil ciclează), § 10.2.5.3
	1 — fragment reassembly time exceeded	probabil fragment pierdut, § 10.2.6.1
12 — parameter problem		pachet neconform cu standardul
4 — source quench		cerere încetinire sursă, § 10.2.5.4
5 — redirect	0 — network 1 — host 2 — TOS & network 3 — TOS & host	redirecţionare, § 10.2.5.5
8 — echo request		cerere ecou, § 10.2.5.2
9 — echo reply		răspuns ecou, § 10.2.5.2

Tabelul 10.4: Tipuri şi subtipuri mai importante de pachete ICMPv4

10.2.5.2. Diagnosticarea funcţionării rutelor

Testarea funcţionării comunicaţiei la nivel reţea este un test simplu şi extrem de util în găsirea penelor dintr-o reţea.

În acest scop, pe majoritatea sistemelor există o comandă utilizator, numită `ping`, care testează legătura dintre nodul curent şi nodul specificat.

Comanda `ping` funcţionează prin trimiterea unui pachet ICMP cu tipul *echo request* (rom. cerere ecou) către nodul specificat. Nodul destinaţie al unui pachet *echo request* răspunde prin trimiterea înapoi (către sursa pachetului *echo request*) a unui pachet ICMP cu tipul *echo reply* (rom. *răspuns ecou*). Pachetul *echo reply* este livrat comenzii `ping`.

Pachetele *echo request* şi *echo reply* se mai numesc uneori *ping* şi *pong*. Pachetele cu tipurile *ping* şi *pong* au prevăzute, conform standardului, două câmpuri, *identificare* şi *nr. secvenţă*, pe baza cărora nucleul sistemului şi comanda `ping` identifică corespondenţele între pachetele *ping* trimise şi pachetele *pong* recepţionate. Pachetele *ping* şi *pong* au prevăzut şi un câmp, de dimensiune arbitrară, de date utile; scopul acestui câmp este testarea transmiterii pachetelor mari.

Pe lângă comanda `ping` care testează funcţionarea unei legături, există o comandă, `traceroute` (pe sisteme de tip Unix) sau `tracert` (pe Windows), care afişează adresele ruterelor prin care trece un pachet pentru o anumită destinaţie.

Există mai multe metode pentru a afla drumul spre un anumit nod. Metoda utilizată de comanda `traceroute` se bazează pe trimiterea, spre acel nod, a unor pachete *ping* cu valori mici pentru timpul de viaţă (vezi § 10.2.5.3). Un astfel de pachet parcurge începutul drumului spre nodul destinaţie, însă, după parcurgerea unui număr de noduri intermediare egal cu valoarea iniţială a timpului de viaţă, provoacă trimiterea înapoi a unui pachet ICMP de tip *TTL exceeded*. Trimiţând pachete *ping* cu diferite valori pentru timpul de viaţă, se primesc pachete *TTL exceeded* de la diferitele noduri de pe traseul spre destinaţie.

O altă posibilitate de-a afla ruta spre un anumit nod este furnizată de un antet opţional, standardizat şi în IPv4 şi în IPv6, care cere ruterelor să-şi scrie fiecare adresa în acest antet opţional.

10.2.5.3. Ciclarea pachetelor IP

Este posibil să existe (temporar) inconsistenţe în tabelele de dirijare. De exemplu, se poate ca tabela de dirijare a nodului A să indice nodul B ca nod următor pe ruta către C, iar tabela nodului B să indice ca nod următor pe ruta către C nodul A. În acest caz, dacă A primeşte un pachet destinat lui

C i-l va trimite lui B, B va pasa pachetul înapoi lui A, ş. a. m. d.

Pentru a preveni ciclarea nelimitată a pachetelor în astfel de cazuri, în antetul IP este prevăzut un câmp numit *timp de viaţă*. Valoarea acestui câmp este iniţializată de către nodul sursă al pachetului (valoarea iniţială este de ordinul zecilor) şi este scăzută cel puţin cu 1 de către fiecare nod prin care trece pachetul. Dacă valoarea ajunge la 0, nodul nu mai trimite mai departe pachetul ci îl ignoră sau trimite înapoi un pachet ICMP cu tipul *time exceeded*, subtipul *time to live (TTL) exceeded* (rom. *depăşire timp de viaţă*) pentru a semnala situaţia.

10.2.5.4. Congestia

În general, prin *congestie* se înţelege situaţia în care într-un nod intră pachete într-un ritm mai rapid decât poate nodul să retrimită pachetele, rezultând de aici o funcţionare proastă a reţelei (vezi § 5.3).

În cazul congestiei, nodul congestionat poate cere sursei să reducă traficul prin trimiterea către aceasta a unui pachet ICMP cu tipul *source quench*.

10.2.5.5. Redirecţionarea

Un nod, care primeşte un pachet şi constată că trebuie trimis mai departe în aceeaşi subreţea din care a sosit pachetul, poate informa sursa pachetului cu privire la faptul că pachetul a mers pe o rută neoptimă. Informarea se face printr-un pachet ICMP cu tipul *redirect*.

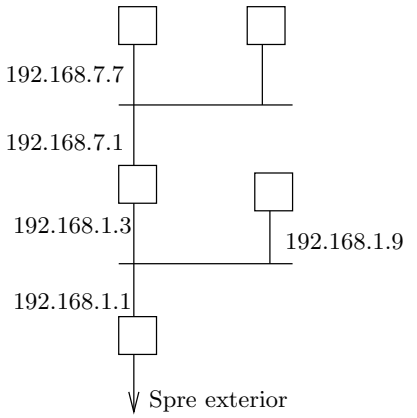

Figura 10.3: O reţea pentru ilustrarea redirecţionării pachetelor (vezi exemplul 10.8)

Exemplul 10.8: Considerăm reţeaua din figura 10.3. Pentru nodurile din

subreţeaua 192.168.1.0/24, ar trebui să existe în tabela de dirijare:

- o regulă care să asocieze prefixului 192.168.7.0/24 gateway-ul 192.168.1.3;
- o regulă indicând ca *default gateway* adresa 192.168.1.1.

În practică, pentru simplificarea administrării, se omite configurarea primei reguli pe toate nodurile cu excepţia nodului 192.168.1.1. În consecinţă, o staţie din subreţeaua 192.168.1.0/24, de exemplu 192.168.1.9, care are de trimis un pachet către un nod din subreţeaua 192.168.7.0/24, de exemplu către 192.168.7.7, va trimite pachetul lui 192.168.1.1 în loc de 192.168.1.3. Nodul 192.168.1.1 trimite mai departe pachetul către 192.168.1.3 şi, totodată, trimite un pachet *ICMP redirect* către 192.168.1.9; aceasta din urmă îşi poate actualiza tabela de dirijare pentru a trimite direct la 192.168.1.3 următoarele pachete destinate nodurilor din subreţeaua 192.168.7.0/24.

10.2.6. Alte chestiuni privind dirijarea pachetelor

10.2.6.1. Dimensiunea maximă a pachetelor şi fragmentarea

Dimensiunea maximă a unui pachet IP este de 64 KiB.

Pe de altă parte, legătura directă între două noduri, dacă are noţiunea de pachet, are o dimensiune maximă a pachetului, care poate fi mai mică decât dimensiunea maximă a pachetului IP: de exemplu, un pachet Ethernet are o dimensiune maximă de 1500 octeţi.

Dacă un pachet IP de transmis este mai mare decât dimensiunea maximă a pachetelor admise de legătura directă între două noduri de pe traseu, există următoarele acţiuni posibile:

- se face o fragmentare şi reasamblare la nivelul legăturii directe, în mod invizibil faţă de nivelul reţea;
- se face o fragmentare şi reasamblare la nivelul reţea (IP);
- se refuză livrarea pachetelor IP şi se lasă în sarcina nivelului superior să se descurce, eventual furnizându-i acestuia dimensiunea maximă acceptabilă a pachetului.

Trebuie remarcat că, în 1981, când s-a standardizat protocolul Internet, era mult prea mult să se ceară fiecărui nod să dispună de câte 64 KiB de memorie pentru memorarea fiecărui pachet.

Fragmentarea la nivelul legăturii directe nu priveşte protocolul IP. Protocolul IP versiunea 6 cere ca nivelul legăturii directe să permită transmiterea pachetelor IP de până la 1280 B, recomandabil până la 1500 B.

© 2008, Radu-Lucian Lupşa

Pentru a permite producerea, de către nivelul superior, a unor pachete IP suficient de mici, există un protocol pentru aflarea dimensiunii maxime a pachetelor ce pot trece prin legăturile directe. Protocolul este descris în [RFC 1981, 1996].

Protocolul Internet permite şi fragmentarea la nivel reţea a pachetelor.

Pentru IP versiunea 4, câmpurile necesare pentru fragmentarea şi reasamblarea pachetelor sunt prevăzute în antetul standard. De asemenea, există un flag, *nu fragmenta*, care cere ruterelor de pe traseu să nu încerce fragmentarea ci în schimb să abandoneze transmiterea pachetului.

Pentru IP versiunea 6, câmpurile privind fragmentarea au fost mutate într-un antet opţional, deoarece este probabil să nu fie utilizate frecvent. Fragmentarea poate fi făcută doar de emiţătorul pachetului, ruterele de pe traseu fiind obligate să abandoneze transmiterea în cazul în care pachetul este prea mare.

Fragmentele sunt pachete IP obişnuite, care se transmit independent unul de altul din punctul în care s-a efectuat fragmentarea.

Nodul destinaţie efectuează reasamblarea pachetelor. În acest scop se utilizează câmpurile *identificare* şi *deplasament* şi flagul *mai urmează fragmente*. Astfel, un pachet se va asambla din fragmente având toate aceeaşi valoare în câmpurile *identificare*, *adresă sursă*, *adresă destinaţie* şi *protocol*. Pachetul asamblat va avea antetul identic cu al fragmentelor (mai puţin câmpurile ce controlează fragmentarea). Datele utile vor fi reconstituite din datele utile ale fragmentelor. Câmpul *deplasament* al unui fragment arată locul datelor utile din fragment în cadrul pachetului (reamintim că fragmentele, ca orice pachete IP, se pot pierde, pot fi duplicate şi ordinea lor de sosire poate fi inversată). Lungimea pachetului este determinată din faptul că, în ultimul fragment, flagul *mai urmează fragmente* are valoarea 0.

Destinaţia încearcă reasamblarea unui pachet din momentul în care a primit primul fragment al pachetului. Dacă celelalte fragmente nu sosesc într-un interval de timp suficent de scurt, nodul abandonează reasamblarea şi trimite înapoi un pachet ICMP cu tipul *time exceeded*, subtipul *fragment reassembly time exceeded*.

10.2.6.2. Calitatea serviciului

Dacă un nod este relativ aglomerat, acesta trebuie să ia decizii privind prioritatea pachetelor:

- dacă unele pachete trebuie trimise cât mai repede, faţă de altele care pot fi ţinute mai mult în coada de aşteptare;
- la umplerea memoriei ruterului, care pachete pot fi aruncate (distruse).

Câmpul *tip serviciu* din antetul IP conţine informaţii despre nivelul de calitate a serviciului cerut de emiţătorul pachetului; în funcţie de acesta, modulul de reţea ia deciziile privind ordinea de prioritate a pachetelor.

10.2.7. Configurarea şi testarea unei reţele IP locale

10.2.7.1. Alegerea parametrilor

În majoritatea cazurilor, într-o reţea locală, subreţelele IP, adică legăturile directe între nodurile IP, se realizează prin intermediul unor reţele din familia IEEE 802 (Ethernet sau 802.11). Primul lucru ce trebuie stabilit este alcătuirea subreţelelor.

În continuare se stabileşte, pentru fiecare subreţea IP, prefixul de reţea corespunzător. Prefixul fiecărei subreţele trebuie, pe de o parte, să permită alocarea unui număr suficient de adrese nodurilor din subreţea şi, pe de altă parte, să ducă la alocarea de adrese dintre adresele alocate de furnizorul de acces Internet sau dintre adresele rezervate pentru reţele private (vezi şi § 10.7.2 pentru alte considerente privind utilizarea adreselor private).

Pentru fiecare subreţea IP, nodurile componente trebuie să facă parte din acelaşi VLAN 802.1Q (dacă se definesc VLAN-uri) şi ca urmare trebuie să facă parte din aceeaşi reţea 802 fizică. Această cerinţă este determinată de faptul că, în cadrul unei subreţele IP, fiecare nod trebuie să poată comunica cu orice alt nod al subreţelei fără a implica dirijare la nivel IP; comunicarea trebuie să fie realizată de nivelul inferior, adică de reţeaua 802.

De notat însă că în cadrul aceleiaşi reţele IEEE 802, şi chiar în cadrul aceluiaşi VLAN 802.1Q, se pot defini mai multe subreţele IP. Astfel de subreţele lucrează independent una de cealaltă şi necesită rutere pentru a fi legate logic între ele. Pentru ca un nod să fie membru în subreţelele IP stabilite în aceeaşi reţea fizică este suficient să definească mai multe adrese IP pe aceeaşi placă de reţea (vezi § 10.5, în special § 10.5.1 pentru detalii).

Notă: independenţa subreţelelor IP de pe acelaşi VLAN este limitată de faptul că subreţelele partajează debitul maxim de transmisie şi că un intrus care ar sparge un calculator ar putea avea acces la toate subreţelelel IP stabilite pe VLAN-ul sau reţeaua fizică din care face parte calculatorul spart.

Configurarea tabelelor de dirijare trebuie să fie făcută astfel încât, pentru orice nod sursă şi pentru orice nod destinaţie, fiecare nod de pe traseul unui pachet să găsească corect următorul nod. În reţelele cu structură arborescentă (în care între oricare două subreţele există un singur drum posibil), acest lucru se realizează de regulă astfel:

- Pentru fiecare subreţea se alege, dintre nodurile ce acţionează ca rutere către alte subreţele, un *default gateway*. Acesta se alege de regulă ca

fiind nodul din subreţea cel mai apropiat de ieşirea spre restul Internetului. Se obişnuieşte ca nodul ales ca *default gateway* să primească adresa IP cea mai mică din subreţea (adică adresa în care sufixul are valoarea 1).

- Pe toate nodurile subreţelei se configurează ca *default gateway* nodul ales ca *default gateway* al subreţelei. Pentru nodurile care fac parte din mai multe subreţele, se ia *default gateway*-ul din subreţeaua cea mai apropiată de exterior (astfel un nod nu va avea ca *default gateway* pe el însuşi).
- Pe nodul ales ca *default gateway* pentru o subreţea se vor configura rutele către subreţelele „din subordine“ — subreţelele mai depărtate de legătura spre exterior decât subreţeaua considerată.

Mai notăm că într-o tabelă de dirijare statică nu se pot configura, pentru toleranţă la pene, mai multe căi spre o aceeaşi destinaţie. Dacă se doreşte aşa ceva este necesară instalarea unui program de calcul automat al tabelei de dirijare.

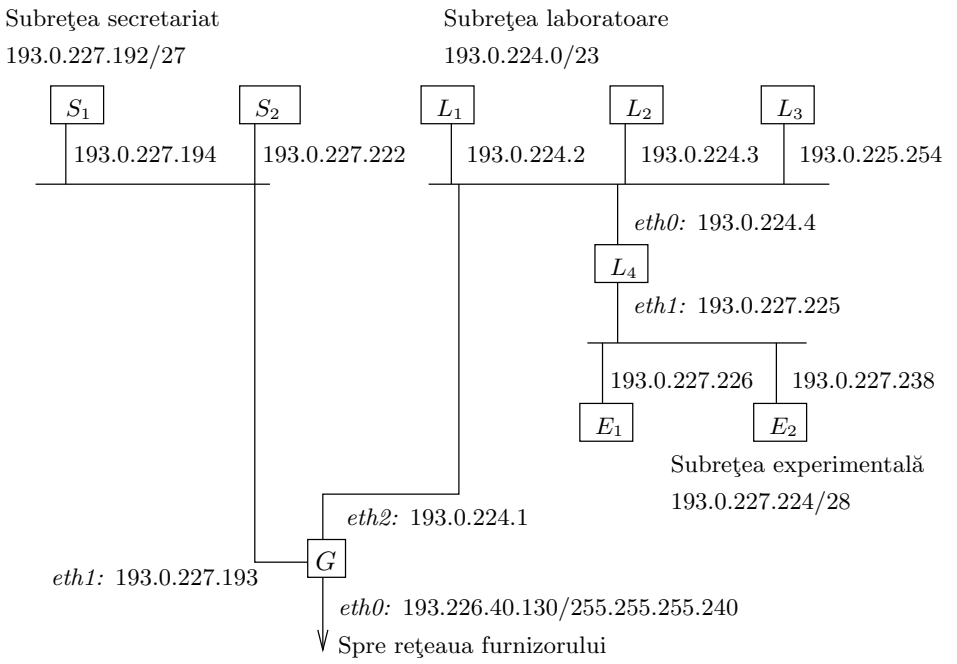

Figura 10.4: Reţea pentru exemplul 10.9

Exemplul 10.9: Să considerăm că avem de construit o reţea într-o şcoală. Presupunem că am obţinut alocarea blocului de adrese 193.0.224.0/22 pentru utilizare în reţeaua proprie şi că ruterul ce asigură legătura cu reţeaua furnizorului de acces Internet a primit adresa (în reţeaua furnizorului) 193.226.40.130 cu masca 255.255.255.240 (prefix de 28 de biţi).

Să presupunem, de asemenea, că s-a decis împărţirea reţelei interne în trei subreţele (fig. 10.4), respectiv pentru secretariat, laboratorul de informatică şi o reţea specială pentru experimente. Împărţirea luată ca exemplu este o împărţire tipică din considerente de trafic şi de securitate: reţeaua experimentală trebuie să poată fi izolată uşor de restul reţelei, iar secretariatul este separat faţă de traficul şi eventual atacurile dinspre laboratorarele de informatică.

Fiecare subreţea este construită dintr-un număr de switch-uri Ethernet, *access point*-uri 802.11 şi calculatoarele respective. Switch-urile şi *access point*-urile nu au fost figurate explicit deoarece ele nu sunt vizibile din punctul de vedere al nivelului IP.

Pentru conectarea celor trei subreţele împreună şi la Internet configurăm două rutere: G, dotat cu trei plăci de reţea, care leagă reţeaua secretariatului, reţeaua din laboratoare şi reţeaua furnizorului de acces Internet şi L_4 (probabil amplasat în laborator, pentru a fi la îndemână în timpul experimentelor), dotat cu două plăci de reţea, care leagă reţeaua experimentală de reţeaua din laborator.

Odată stabilite subreţelele, să alocăm adresele. Blocul de adrese disponibile este 193.0.224.0/22, conţinând 1024 de adrese. Putem crea blocuri având ca dimensiuni puteri ale lui 2: 512, 256, 128, 64, 32, 16, 8 sau 4 adrese. Începem prin a aloca laboratoarelor un bloc cât mai mare, de 512 adrese (510 utilizabile efectiv), anume 193.0.224.0/23. Din blocul de 512 adrese rămas (193.0.226.0/23), să alocăm 32 adrese secretariatului şi 16 adrese reţelei experimentale. Este bine să le alocăm cât mai compact, pentru ca dintre adresele nealocate să păstrăm posibilitatea de-a aloca blocuri cât mai mari. Vom aloca cele două blocuri de 32 şi 16 adrese din ultimul bloc de 64 de adrese din cele 512 libere: 193.0.227.192/27 pentru secretariat şi 193.0.227.224/28 pentru reţeaua experimentală.

Pentru fiecare din cele trei subreţele, există o alegere naturală pentru *default gateway:* G pentru reţeaua secretariatului şi pentru reţeaua din laboratoare şi, respectiv, L_4 pentru reţeaua experimentală. În fiecare caz, *default gateway*-ul este nodul cel mai apropiat de exterior. În fiecare subreţea, adresa dată ruterului cu rol de *default gateway* este cea mai mică adresă din acea subreţea.

© 2008, Radu-Lucian Lupşa

Să vedem acum cum trebuie configurate tabelele de dirijare. Pentru staţii, tabelele sunt formate din câte două reguli: o regulă pentru livrarea directă, care asociază prefixului subreţelei unica interfaţă de reţea, şi o regulă implicită, care asociază prefixului vid adresa *default gateway*-ului.

Pentru nodul L_4, tabela de dirijare are trei reguli, două fiind pentru livrarea directă prin cele două interfeţe, iar a treia este regula implicită:

- 193.0.224.0/23 → *eth0*;
- 193.0.227.224/28 → *eth1*;
- 0.0.0.0/0 → 193.0.224.1 (prin *eth0*).

Pentru nodul G avem 5 reguli: trei reguli de livrare directă prin cele trei interfeţe, o regulă implicită indicând *default gateway*-ul reţelei furnizorului de acces Internet şi o regulă pentru dirijarea spre subreţeaua „subordonată“ 193.0.227.224/28:

- 193.226.40.128/28 → *eth0*;
- 193.0.227.192/27 → *eth1*;
- 193.0.224.0/23 → *eth2*;
- 0.0.0.0/0 → 193.226.40.129 (prin *eth0*).
- 193.0.227.224/28 → 193.0.224.1 (prin *eth1*).

10.2.7.2. Configurarea parametrilor de reţea pe diverse sisteme de operare

Pe sistemele Windows, există două posibilităţi de configurare: comanda `ipconfig` (în mod text) şi seria de casete de dialog din Start/ Control panel/ Network/ *interfaţă*. Prin ambele interfeţe se realizează atât modificarea parametrilor din modulul IP din nucleul sistemului de operare cât şi scrierea lor în *Windows registry* pentru reîncărcarea lor la repornirea sistemului. Comportamentul de ruter, dacă este dorit, trebuie activat explicit.

Pe sistemele de tip Linux configurarea este puţin mai complicată, dar şi mult mai flexibilă. Comanda `ifconfig` setează parametrii legaţi de interfeţele de reţea, anume adresa IP şi masca de reţea. Tot comanda `ifconfig` introduce în tabela de dirijare regulile de livrare directă (de fapt acesta este motivul pentru care are nevoie de masca de reţea). Tabela de dirijare se afişează şi se modifică cu comenzile `route` sau `ip` (prima este mai simplă şi se găseşte pe toate sistemele, a doua este mai complexă şi serveşte şi altor scopuri). De remarcat că oprirea unei interfeţe de reţea cu `ifconfig` duce automat la eliminarea din tabela de dirijare a tuturor regulilor ce au ca ţintă

© 2008, Radu-Lucian Lupşa

adrese accesibile prin acea interfaţă. Activarea comportamentului de ruter se face cu o comandă `sysctl`.

Comenzile `ifconfig`, `route`, `ip` şi `sysctl` au efect imediat asupra modulului IP din nucleu, dar configurările respective se pierd la repornirea sistemului. Parametrii de reţea utilizaţi la pornirea sistemului sunt luaţi de script-urile de iniţializare din nişte fişiere text; din păcate, fiecare distribuţie de Linux are propriile fişiere de configurare. De exemplu, Fedora plasează datele în fişiere în directorul `/etc/sysconfig/network-scripts`.

10.2.7.3. Testarea şi depanarea reţelelor

Cel mai util instrument de depanare pentru problemele de conectivitate este comanda `ping`. Pentru buna funcţionare a acesteia şi a comenzii `traceroute`, este necesar ca, dacă este instalat un filtru de pachete (firewall), acesta să permită trecerea pachetelor ICMP cu tipurile *echo-request*, *echo-reply*, *destination unreachable* şi *time exceeded*.

Primul lucru ce trebuie testat, în cazul unei probleme de conectivitate sau după o lucrare mai amplă la reţea, este dacă legăturile directe funcţionează. Acest lucru se face dând comanda `ping` pentru câte un vecin din fiecare subreţea din care face parte calculatorul de pe care se efectuează testul. Dacă `ping`-ul merge, înseamnă că legătura funcţionează (plăcile de reţea, cablurile, switch-urile şi *access point*-urile de pe traseu) şi că adresele IP şi măştile de reţea sunt „suficient de bune“ pentru ca nodurile să fie văzute ca făcând parte din aceeaşi subreţea. Dacă `ping`-ul nu merge şi pachetele *ping* şi *pong* nu sunt filtrate, pana trebuie căutată printre lucrurile enumerate până aici.

Dacă `ping`-ul merge pe legăturile directe, se trece la verificarea legăturilor între subreţele diferite. Dacă o legătură indirectă nu funcţionează deşi toate legăturile directe ce ar trebui să fie utilizate funcţionează, problema este de la regulile de dirijare (sau de la un filtru de pachete; de aceea este bine ca pachetele *ping* şi *pong* să nu fie filtrate). Există două lucruri uşor de scăpat din vedere aici: faptul că pentru ca `ping`-ul să meargă este necesar ca dirijarea să funcţioneze corect în ambele sensuri şi faptul că între regulile de dirijare intră inclusiv regulile de *default gateway* de pe staţii.

De exemplu, o staţie care nu are configurat *default gateway* va putea comunica cu vecinii direcţi, dar nu şi cu alte calculatoare — nici măcar cu *default gateway*-ul, dacă esre specificat prin altă adresă decât cea din aceeaşi subreţea cu staţia configurată. Alt exemplu: la reţeaua din exemplul 10.9, dacă pe nodul G nu se pune regula care asociază prefixului 193.0.227.224/28 *gateway*-ul 193.0.224.4, atunci pachetele dinspre subreţeaua 193.0.227.224/28

© 2008, Radu-Lucian Lupşa

pot să iasă spre Internet, însă pachetele dinspre Internet nu trec de nodul G. Un `ping` executat de pe E_1 către L_4 merge, însă către L_2 nu merge. În acest din urmă caz, pachetele *ping* ajung la L_2, dar pachetele *pong* sunt trimise de L_2 către G (conform regulii implicite). G, neavând altă regulă, aplică şi el regula implicită şi le trimite către 193.226.40.129 (*default gateway*-ul din reţeaua furnizorului, nefigurat pe desen). De aici pachetele se întorc înapoi spre G, deoarece furnizorul ştie că toată reţeaua 193.0.224.0/22 este în spatele lui G. Astfel, pachetele *pong* ciclează între G şi 193.226.40.129.

10.3. Nivelul transport

Aplicaţiile nu folosesc direct protocolul IP din mai multe motive:

- dacă două aplicaţii se execută pe acelaşi calculator, este necesar ca nucleul sistemului de operare să determine cărei aplicaţii îi este destinat fiecare pachet sosit;
- serviciul oferit direct de nivelul reţea (pachete ce se pot pierde, pot sosi în altă ordine şi, în anumite cazuri, pot fi duplicate) este de obicei inadecvat.

Adaptarea între nevoile aplicaţiilor şi serviciile oferite de nivelul reţea IP cade în sarcina *nivelului transport*. Nivelul transport constă dintr-o componentă a nucleului sistemului de operare, la care se adaugă uneori nişte funcţii de bibliotecă.

Componenta nivelului transport situată în nucleul din sistemului de operare furnizează aplicaţiei funcţiile sistem din familia `socket()`. Serviciile oferite aplicaţiei prin *socket*-uri de tip *stream* sunt implementate utilizând protocolul TCP. Serviciile oferite prin *socket*-uri de tip *dgram* sunt implementate prin protocolul UDP. Modulele reţelei IP şi locul modulelor TCP şi UDP sunt arătate în figura 10.5.

10.3.1. Conexiuni cu livrare garantată: protocolul TCP

Scopul protocolului TCP (*Transmission Control Protocol*) este acela de a realiza o conexiune de tip flux de octeţi, bidirecţională, cu livrare garantată. Protocolul este descris în [RFC 793, 1981].

Mai în detaliu, TCP oferă:

- serviciu de tip *conexiune*, cu cele trei faze, de deschidere conexiune, comunicaţie şi închidere conexiune;
- trasnportă *flux de octeţi*, adică emiţătorul trimite un şir de octeţi, negrupaţi în mesaje, de lungime arbitrară;

© 2008, Radu-Lucian Lupşa

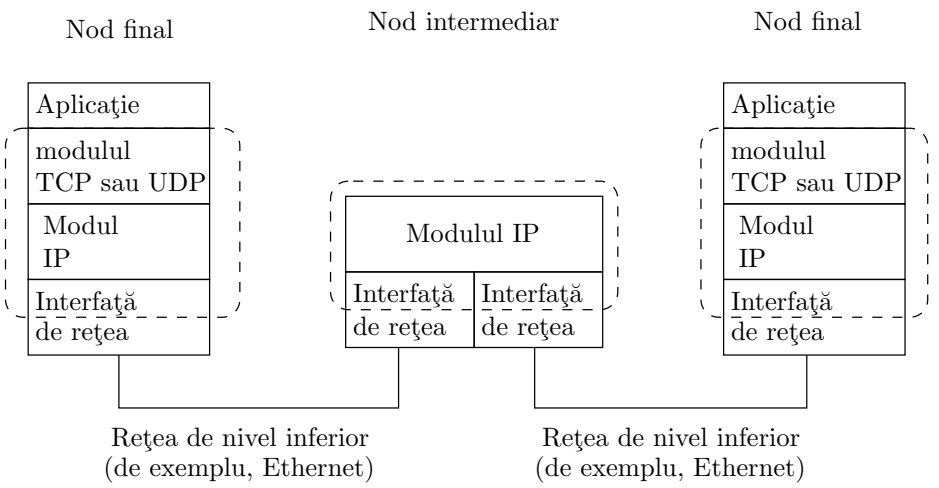

Figura 10.5: Modulele unei reţele IP. Partea inclusă în sistemul de operare este delimitată cu linie punctată.

- legătură *bidirecţională*, adică fiecare din cei doi parteneri de comunicaţie poate trimite date celuilalt;
- *livrare sigură*, adică octeţii trimişi de emiţător ajung la receptor sigur, fără duplicate şi în aceeaşi ordine în care au fost trimişi.

Modulul TCP are la dispoziţie, pentru realizarea conexiunilor TCP, facilităţile oferite de reţeaua IP.

10.3.1.1. Principiul conexiunii TCP

Livrarea sigură este obţinută pe baza unui mecanism de numerotare şi confirmare a pachetelor, aşa cum am văzut în § 4.3. Mecanismul este implementat după cum urmează. Pentru început considerăm transmisia unidirecţională şi conexiunea deja deschisă.

Zona de date utile a unui pachet IP ce transportă date pentru protocolul TCP conţine un *antet TCP* şi *datele utile TCP*. Antetul TCP este descris în tabelul 10.5.

Fiecărui octet al fluxului de date utile (de transportat de către TCP) i se asociază un *număr de secvenţă*; octeţi consecutivi au asociate numere de secvenţă consecutive.

Fiecare pachet TCP conţine, în zona de date utile, un şir de octeţi utili consecutivi. Câmpul *număr de secvenţă* din antetul TCP conţine numărul de secvenţă al primului octet din datele utile.

© 2008, Radu-Lucian Lupşa

Nume câmp	Dim. (biţi)	Observaţii
Port sursă	16	
Port destinaţie	16	
Nr. secvenţă	32	
Nr. confirmare	32	
Deplasament date	4	poziţia datelor utile în pachet, dependent de dimensiunea opţiunilor
Rezervat	6	valoarea 0
Urgent	1	vezi § 10.3.1.11
Acknowledge	1	indică faptul că numărul de confirmare este valid
Push	1	arată că pachetul trebuie trimis urgent, fără a fi ţinut în diverse zone tampon
Reset	1	cere închiderea forţată a conexiunii
Synchronize	1	cere deschiderea conexiunii
Finalize	1	cere închiderea conexiunii
Dim. fereastră	16	vezi § 10.3.1.8
Suma de control	16	suma de control a antetului
Poziţie date urgente	16	vezi § 10.3.1.11
Opţiuni	variabil	

Tabelul 10.5: Antetul TCP

Modulul TCP receptor ţine evidenţa numărului de secvenţă al ultimului octet primit. La primirea unui pachet TCP, modulul receptor determină dacă datele utile sunt octeţi deja primiţi (duplicate), octeţi ce vin imediat în continuarea celor primiţi până în acel moment sau octeţi până la care există octeţi lipsă (nerecepţionaţi încă din cauza unui pachet pierdut sau întârziat). Octeţii primiţi în continuarea celor deja primiţi sunt puşi într-o coadă pentru a fi livraţi aplicaţiei la cererea acesteia.

La primirea unui pachet TCP, receptorul trimite înapoi un pachet TCP de confirmare. Un pachet TCP de confirmare are în antetul TCP flagul *acknowledge* setat şi în câmpul *număr de confirmare* numărul de secvenţă al următorului octet aşteptat de la emiţător. Un pachet cu număr de confirmare n informează emiţătorul că toţi octeţii cu numere de secvenţă mai mici sau egale cu $n-1$ au fost primiţi de receptor şi nu mai trebuie retransmişi.

Emiţătorul retransmite octeţii neconfirmaţi. Datele utile, furnizate de aplicaţie emiţătorului, sunt păstrate într-o zonă tampon şi ţinute acolo până la confirmarea primirii lor de către receptor. Datele trimise şi neconfirmate într-un anumit interval de timp se retransmit. Datele noi intrate în zona tampon sunt trimise cu un nou pachet. Dacă un pachet nu este confirmat şi între prima trimitere şi momentul retrimiterii au mai sosit date de la aplicaţie, emiţătoul poate la retrimitere să concateneze datele vechi neconfirmate cu cele noi.

Exemplul 10.10: În figura 10.6 este prezentată (simplificat) o parte dintr-un schimb de pachete corespunzător unei conexiuni TCP. Presupunem că aplicaţia sursă are de trimis şirul `abcdefghi`, şi că acesta este pasat modulului TCP emiţător în etape, întâi şirul `abcd`, apoi `efg`, `h` şi în final `i`. Mai presupunem conexiunea TCP deja deschisă şi numărul curent de secvenţă 10. Să analizăm puţin schimbul de pachete:

- Emiţătorul trimite un prim pachet, cu număr de secvenţă 10 şi date utile şirul de 4 octeţi `abcd`. Aceşti 4 octeţi au numere de secvenţă respectiv 10, 11, 12 şi 13; primul dintre acestea este scris în câmpul *număr de secvenţă* al antetului TCP.
- La primirea acestui pachet, receptorul răspunde cu un pachet de confirmare, cu numărul de confirmare 14 (acesta este următorul număr de secvenţă aşteptat).
- Emiţătorul trimite acum următorul pachet de date, cu numărul de secvenţă 14 şi date utile `efg`. Presupunem că acest pachet se pierde.
- Ca urmare a primirii de la aplicaţia sursă a următorului octet, `h`, emiţătorul TCP trimite imediat următorul pachet, cu numărul de secvenţă

© 2008, Radu-Lucian Lupşa

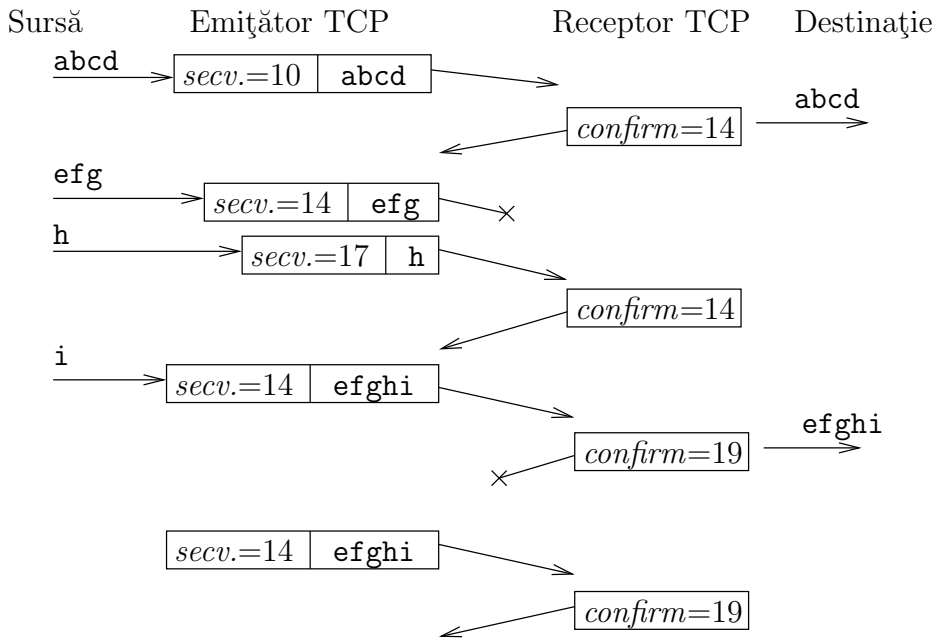

Figura 10.6: O secvenţă de pachete TCP schimbate între emiţător (stânga) şi receptor (dreapta); vezi exemplul 10.10.

17 şi date utile `h` (presupunem că emiţătorul nu utilizează algoritmul Nagle, § 10.3.1.10).

- La primirea acestui pachet (cu număr de secvenţă 17), receptorul nu îl poate confirma deoarece nu a primit numerele de secvenţă 14–16; dacă în acest moment receptorul ar trimite un pachet cu număr de confirmare 18, emiţătorul ar crede că toate numerele de secvenţă până la 17 inclusiv au fost primite şi nu ar mai retrimite niciodată numerele de secvenţă 14–16. Ca urmare, receptorul are două posibilităţi: să ignore pachetul primit (adică să nu trimită nici un pachet înapoi) sau să retrimită numărul de confirmare 14; în exemplul de faţă am considerat a doua variantă.
- Presupunem acum că pe de o parte a expirat time-out-ul emiţătorului pentru numerele de secvenţă 14–17 şi pe de altă parte că a primit de la aplicaţia sursă următorul octet (`i`). Emiţătorul împachetează acum tot ce are de (re)trimis într-un singur pachet şi trimite numărul de secvenţa 14 şi datele utile `efghi`.
- Receptorul confirmă pachetul primit, trimiţând numărul de confirmare 19. Presupunem că pachetul de confirmare respectiv se pierde.
- Emiţătorul retrimite pachetul anterior, după expirarea time-out-ului de la trimiterea lui.
- Receptorul constată că a primit un duplicat al datelor precedente, însă retrimite confirmarea cu numărul 19. Netrimiterea în acest moment a confirmării ar duce la repetarea la infinit de către emiţător a pachetului precedent. De notat că, deşi receptorul primeşte un duplicat, emiţătorul încă nu ştie de primirea datelor de către receptor.

În continuarea schimbului de pachete ilustrat, nu se vor mai trimite pachete până ce aplicaţia sursă nu va da emiţătorului TCP noi date de transmis.

Am presupus pâna aici că numerele de secvenţă sunt numere naturale care pot creşte nedeterminat de mult. În realitate, numerele de secvenţă sunt reprezentate (vezi tabelul 10.5) pe 32 de biţi. Cum un număr de secvenţă este asociat unui octet transmis, rezultă că există numere de secvenţă distincte doar pentru 4 GiB de date; după aceea numerele de secvenţă încep să se repete. Aceasta nu era o problemă în anii ’80, deoarece la 10 Mbit/s repetarea unui număr de secvenţă apare cel mai repede după aproape o oră, timp în care este puţin probabil să mai existe în reţea un pachet vechi cu acelaşi număr de secvenţă. Într-o reţea cu debit de 1 Gbit/s, se pot transmite 4 GiB în 34 secunde, ceea ce face foarte posibilă o confuzie între un pachet care s-a „încurcat“ prin reţea timp de 34 s şi un pachet nou care poartă informaţie situată, în cadrul conexiunii, 4 GiB mai încolo, şi are acelaşi număr de secvenţă.

© 2008, Radu-Lucian Lupşa

Eliminarea riscului de confuzie între pachete la debite mari de transmisie a datelor se realizează, conform [RFC 1323, 1992], prin introducerea în antetul TCP a unui câmp opţional cuprinzând un marcaj de timp. Metoda se aplică doar la comunicaţia între module TCP care implementează RFC 1323. În cazul în care unul din modulele TCP nu implementează RFC 1323, modulele TCP vor avea grijă să nu repete un număr de secvenţă mai devreme de câteva minute, oprind efectiv transmisia la nevoie.

10.3.1.2. Comunicaţia bidirecţională

Cele două sensuri de comunicaţie ale unei conexiuni TCP funcţionează (aproape) complet independent. Numerele de secvenţă utilizate pe cele două sensuri evoluează independent.

Totuşi, datele utile pentru un sens sunt plasate în acelaşi pachet TCP cu confirmarea pentru celălalt sens. Astfel, fiecare pachet TCP conţine întotdeauna date pentru un sens şi confirmare pentru celălalt sens.

Dacă este necesară transmiterea unor date, dar nu şi a unei confirmări, în câmpul *număr de confirmare* se repetă ultimul număr de confirmare transmis.

Dacă este necesar să se transmită doar o confirmare, fără date utile pentru celălalt sens, atunci zona de date utile se face de dimensiune 0 iar în câmpul *număr de secvenţă* se pune numărul de secvenţă al următorului octet ce va fi transmis.

10.3.1.3. Deschiderea şi închiderea conexiunii

Pentru fiecare sens de comunicaţie se consideră câte doi octeţi fictivi: un *octet de pornire* aflat înaintea primului octet util al datelor transmise şi un *octet de încheiere* aflat după ultimul octet al datelor utile. Aceşti octeţii fictivi au asociate numere de secvenţă ca şi când ar fi octeţi obişnuiţi. Ei nu sunt transmişi în zona de date utile; prezenţa lor într-un pachet este semnalizată prin setarea flagurilor *synchronize* şi respectiv *finalize* din antetul TCP.

Astfel, dacă un pachet TCP are flagul *synchronize* setat, numărul de secvenţă n şi zona de date utile conţinând k octeţi, înseamnă că pachetul transmite $k+1$ octeţi dintre care primul, cu numărul de secvenţă n, este octetul fictiv de pornire, urmat de cei k octeţi din zona de date utile, cu numere de la $n+1$ la $n+k$.

Iniţiatorul unei comunicaţii TCP trimite un pachet TCP conţinând un octet fictiv de pornire, fără a seta flagul *acknowledge* (acesta este singurul pachet ce nu are flagul *acknowledge* setat) şi punând o valoare arbitrară (care va fi ignorată la destinaţie) în câmpul *număr de confirmare*.

Numărul de secvenţă al octetului fictiv de pornire este la alegerea iniţiatorului comunicaţiei.

Receptorul pachetului iniţial răspunde, dacă doreşte să accepte conexiunea, printr-un pachet TCP care, pe de o parte, confirmă pachetul de iniţiere primit, iar pe de altă parte conţine la rândul lui un octet fictiv de pornire.

Fiecare parte consideră conexiunea deschisă în momentul în care sunt satisfăcute simultan condiţiile:

- octetul fictiv de pornire trimis de ea a fost confirmat;
- a primit un octet fictiv de pornire de la partener.

Fiecare parte poate trimite date înainte de a dispune de o conexiune deschisă; datele primite înainte de momentul în care conexiunea este deschisă nu pot fi livrate aplicaţiei până la deschiderea completă a conexiunii.

Închiderea conexiunii se face separat pe fiecare sens de comunicaţie. Marcarea închiderii unui sens se face trimiţând un octet fictiv de încheiere. După trimitera octetului de încheiere este interzis să se mai trimită date noi. Ca urmare, orice pachete (de confirmare) trimise de partea care a închis conexiunea vor avea ca număr de secvenţă numărul imediat următor octetului fictiv de încheiere şi date utile vide. Octetul fictiv de încheiere se confirmă normal. O conexiune poate funcţiona oricât de mult timp cu un sens închis.

Nr. pachet	Sens	Nr. secv.	Nr. confirm.	Flaguri	Date utile
1	$A \to B$	123	0	syn	—
2	$A \leftarrow B$	25	124	syn,ack	—
3	$A \to B$	124	26	ack	`abc`
4	$A \leftarrow B$	26	127	ack	—
5	$A \to B$	127	26	ack,fin	`de`
6	$A \leftarrow B$	26	130	ack	—
7	$A \leftarrow B$	26	130	ack	`xyz`
8	$A \to B$	130	29	ack	—
9	$A \leftarrow B$	29	130	ack,fin	—
10	$A \to B$	130	30	ack	—

Tabelul 10.6: Un exemplu de schimb de pachete în cadrul unei conexiuni. Vezi exemplul 10.11.

Exemplul 10.11: Un exemplu de schimb de pachete în cadrul unei conexiuni este dat în tabelul 10.6.

© 2008, Radu-Lucian Lupşa

Pentru a urmări uşor schimbul de pachete, să remarcăm că A trimite lui B un număr de 7 octeţi din care 2 fictivi, „⟨syn⟩`abcde`⟨fin⟩“, cu numerele de secvenţă de la 123 la 129 inclusiv, iar B trimite lui A un număr de 5 octeţi din care 2 fictivi, „⟨syn⟩`xyz`⟨fin⟩“, cu numerele de secvenţă de la 25 la 29 inclusiv. Fiecare pachet care transportă octeţi numerotaţi, indiferent dacă aceştia sunt date utile sau octeţi fictivi ⟨syn⟩ sau ⟨fin⟩, trebuie confirmat. Pachetele ce conţin doar numărul de confirmare nu se confirmă.

Din punctul de vedere a lui A, conexiunea este complet deschisă la primirea pachetului nr. 2; din punctul de vedere al lui B conexiunea este complet deschisă la primirea pachetului 3. După trimiterea pachetului 5, A nu mai are voie să trimită date noi; poate doar să repete datele vechi (dacă nu ar fi primit pachetul 6 ar fi trebuit să retrimită pachetul 5) şi să trimită confirmări. B consideră conexiunea complet închisă după primirea pachetului 10 (a primit un ⟨fin⟩ de la A şi a trimis şi i s-a confirmat ⟨fin⟩-ul propriu). A poate considera de asemenea conexiunea complet închisă după trimiterea pachetului 10, însă mai trebuie să păstreze un timp datele despre conexiune pentru cazul în care pachetul 10 s-ar pierde şi B ar repeta pachetul 9.

O problemă specială legată de închiderea conexiunii este problema determinării duratei de la închiderea conexiunii până la momentul în care datele asociate conexiunii nu mai sunt necesare şi memoria asociată poate fi eliberată.

Din punctul de vedere al unui modul TCP, conexiunea este închisă în momentul în care sunt îndeplinite condiţiile:

- modulul TCP a trimis octetul special de încheiere,
- modulul TCP a primit confirmarea propriului octet de încheiere,
- modulul TCP a primit un octet special de încheiere de la partener.

După închiderea conexiunii din punctul de vedere al modulului TCP local, există încă posibilitatea ca modulul TCP partener să nu primească confirmarea modulului TCP local pentru octetul special de închidere trimis de modulul TCP partener). Urmarea este că modulul TCP partener nu consideră închisă conexiunea (conform regulilor de mai sus) şi retrimite octetul special de încheiere până la confirmarea acestuia. Modulul TCP local ar trebui să păstreze informaţiile despre conexiune până când modulul TCP partener primeşte confirmarea octetului său de încheiere. Din păcate, determinarea acestui moment este imposibilă, deoarece din acel moment modulul TCP partener nu va mai trimite nici un pachet (conexiunea fiind închisă). Ca soluţie de compromis, protocolul TCP prevede păstrarea datelor despre conexiune un anumit interval de timp (de ordinul câtorva minute) după închiderea

© 2008, Radu-Lucian Lupşa

conexiunii.

10.3.1.4. Alegerea numărului iniţial de secvenţă

Numărul de secvenţă al octetului fictiv de pornire, numit şi *număr iniţial de secvenţă* (engl. *initial sequence number*, *ISN*), trebuie ales în aşa fel încât să nu poată exista confuzie între numere de secvenţă dintr-o conexiune veche şi cele din conexiunea curentă între aceleaşi două părţi (aceleaşi adrese IP şi numere de port).

Ideal, modulul TCP ar trebui să păstreze datele despre o conexiune atât timp cât mai pot exista în reţea pachete aparţinând conexiunii. În aceste condiţii, la redeschiderea unei conexiuni între aceleaşi două părţi, fiecare parte poate atribui octetului fictiv de pornire numărul de secvenţă imediat următor numărului de secvenţă asociat octetului fictiv de încheiere al conexiunii precedente. În acest caz, putem privi conexiunile succesive ca fiind o singură conexiune în care transmisiile sunt delimitate prin secvenţe de octeţi fictivi ⟨fin⟩⟨syn⟩.

Dacă de la precedenta conexiune a trecut destul timp pentru ca pachetele corespunzătoare să nu mai existe în reţea (fie au ajuns la destinaţie, fie au fost distruse ca urmare a depăşirii timpului de viaţă), alegerea numărului iniţial de secvenţă poate fi făcută oricum. Există câteva considerente, enumerate mai jos, care duc la utilitatea unor alegeri particulare.

Un prim considerent este îngreunarea unor atacuri de tip *IP spoofing*.

Un atac *IP spoofing* (numiu uneori simplu *spoofing*) constă în a trimite pachete IP în care se falsifică valoarea câmpului *adresă sursă*. Scopul unui astfel de atac este de-a înşela un mecanism de autentificare bazat pe adresa IP a partenerului de comunicaţie sau de-a deturna o conexiune TCP deja autentificată. În acest din urmă caz, adversarul lasă un client legitim să se conecteze la server şi, după efectuarea autentificării, adversarul injectează un mesaj destinat serverului şi având ca adresă sursă adresa clientului autentificat. Mesajul este interpretat de server ca provenind de la clientul autentificat şi, dacă conţine o comandă autorizată pentru client, comanda este executată, deşi în realitate provine de la adversar.

De multe ori, într-un atac de tip *spoofing* adversarul nu are şi posibilitatea de-a determina ruterele de pe traseu să-i permită recuperarea pachetelor de răspuns. Un atac în astfel de condiţii se numeşte *blind spoofing*.

Un atac *blind spoofing* se contracarează foarte simplu generând aleator numărul iniţial de secvenţă, adică făcând ca valoarea lui să fie imprevizibilă pentru adversar. Cum adversarul trebuie să emită pachete cu numere de secvenţă şi de confirmare valide, în cazul acestor măsuri adversarul trebuie

© 2008, Radu-Lucian Lupşa

efectiv să ghicească numărul de secvenţă.

Un al doilea considerent este prevenirea atacului *syn flooding*. Într-un astfel de atac, adversarul trimite multe pachete TCP de deschidere de conexiune (cu flagul *synchronize* pe 1 şi *acknowledge* pe 0), cu diferite valori pentru câmpurile *port sursă* şi uneori *adresă sursă* (este posibil ca adversarul să execute şi *IP spoofing*). Maşina atacată trebuie să aloce o structură de date în care să memoreze datele despre conexiune până la terminarea conexiunii sau la expirarea unui time-out. Adversarul însă nu mai trimite nimic pe nici una dintre conexiuni, mulţumindu-se să ţină ocupată memorie pe maşina atacată; este vorba deci de un atac *denial of service*.

Contramăsura, numită *syn cookie*, se realizează astfel. O maşină care aşteaptă cereri de conectare generează aleator un şir de biţi, pe care îl ţine secret. La primirea unei cereri de conectare, maşina alege, ca număr de secvenţă iniţial (pentru sensul dinspre ea spre iniţiatorul conexiunii), valoarea unei funcţie de dispersie criptografică, aplicată asupra numărului de secvenţă al pachetului primit concatenat cu un şirul secret. Apoi, maşina ţintă a conexiunii trimite pachetul de răspuns, acceptând numărul de secvenţă propus de iniţiatorul conexiunii şi conţinând numărul de secvenţă astfel generat. Maşina ţintă nu înregistrează încă, în tabela de conexiuni, conexiunea astfel deschisă. În cazul unei conexiuni reale, la trimiterea următorului pachet de către iniţiatorul conexiunii, maşina ţintă verifică numărul de secvenţă folosind aceeaşi funcţie de dispersie, după care memorează conexiunea în tabelul de conexiuni. În acest fel, o conexiune creată dintr-un atac *syn flooding* nu ocupă memorie, în schimb o conexiune legitimă se poate deschide.

10.3.1.5. Închiderea forţată a conexiunii

Refuzul cererii de deschidere a unei conexiuni se face trimiţând iniţiatorului un pachet TCP cu flagul *reset* setat.

La primirea unui pachet care nu corespunde unei conexiuni deschise (adică pachetul este primit într-un moment în care conexiunea este închisă şi pachetul nu are flagul *synchronize* setat), modulul TCP trebuie să trimită înapoi un pachet cu flagul *reset* setat. Ideea este ca, dacă nodul curent a căzut şi a fost repornit, pierzând evidenţa conexiunilor deschise, să informeze toţi partenerii de comunicaţie care încearcă să continue comunicaţia cu el că datele despre comunicaţie au fost pierdute.

Un nod care a primit un pachet cu flagul *reset* ca răspuns la un pachet TCP pentru o conexiune trebuie să abandoneze forţat conexiunea. Aplicaţia ce utilizează acea conexiune este informată, printr-un cod de eroare, la următoarea operaţie privind conexiunea.

Alte situaţii care conduc la abandonarea unei conexiuni sunt:

- depăşirea unui număr de încercări de trimitere de date fără a primi confirmare;
- primirea unui pachet ICMP cu tipul *destination unreachable*.

Timpul cât emiţătorul încearcă retrimiterea pachetelor neconfirmate, până în momentul în care declară legătura căzută, este de ordinul zecilor de secunde până la 2–3 minute. Ca urmare, întreruperea pe termen scurt a unui cablu de reţea nu duce la întreruperea conexiunilor TCP ce utilizau acel cablu. De asemenea, dacă un nod sau o legătură a reţelei IP cade, dar rămâne un drum alternativ între capetele conexiunii TCP, iar nivelul reţea reface tabelele de dirijare pentru a folosi acel drum alternativ şi acest lucru se întâmplă suficient de repede pentru ca modulul TCP să nu declare conexiunea căzută, atunci conexiunea TCP continuă să funcţioneze normal, cu pachetele IP circulând pe noua rută.

Dacă, pe o conexiune TCP deschisă, toate datele vechi au fost trimise şi confirmate, atunci nu se transmit pachete IP câtă vreme nu sunt date utile noi de transmis. Ca urmare, dacă aplicaţiile nu comunică vreme îndelungată pe o conexiune TCP deschisă, căderea legăturii sau a unuia din capete nu este detectată de celălalt capăt decât în momentul în care acesta are de transmis date. Dacă acel capăt nu are de transmis date vreme îndelungată, de exemplu câteva ore sau chiar zile, conexiunea rămâne deschisă din punctul de vedere al modulului TCP local. Pentru a evita o astfel de situaţie, modulul TCP poate fi instruit — via un apel `fcntl()` asupra socket-ului corespunzător conexiunii — să trimită din când în când câte un pachet fără date utile, doar pentru a solicita o confirmare de la celălalt capăt. Opţiunea aceasta se numeşte *keep alive* (rom. *menţine în viaţă*), deşi mai degrabă ar trebui numită *testează că mai este în viaţă*. Utilizarea opţiunii *keep alive* are şi un dezavantaj: căderea temporară a reţelei are şanse mai mari să ducă la abandonarea conexiunii.

10.3.1.6. Identificarea aplicaţiei destinaţie

Pentru a identifica aplicaţia căreia îi sunt destinate datele primite, conexiunile TCP sunt identificate printr-un cvadruplet format din adresele IP ale celor două capete şi *porturile* celor două capete. Porturile sunt numere în intervalul 1–65535 utilizate pentru a deosebi conexiunile stabilite între aceleaşi adrese IP.

Sistemul de operare ţine evidenţa conexiunilor deschise. Pentru fiecare conexiune, sistemul reţine adresa IP locală, portul local, adresa IP a celuilalt capăt şi portul de la celălalt capăt. De asemenea, sistemul reţine aplicaţia care a deschis conexiunea.

© 2008, Radu-Lucian Lupşa

La primirea unui pachet TCP, sistemul ia adresele sursă şi destinaţie din antetul IP şi portul sursă şi destinaţie din antetul TCP. Pe baza acestora sistemul identifică conexiunea căreia îi aparţine pachetul. De aici sistemul regăseşte pe de o parte informaţiile necesare gestionării conexiunii (zone tampon, numere de secvenţă, etc) şi pe de altă parte aplicaţia căreia îi sunt destinate datele.

10.3.1.7. Corespondenţa între funcţiile `socket()` şi acţiunile modulului TCP

Funcţionalitatea modulului TCP este oferită programului utilizator prin intermediul funcţiilor sistem din familia `socket()`. Funcţiile `socket()` şi modul lor de utilizare într-o aplicaţie au fost studiate în § 8.1. Aici vom arăta legătura dintre apelarea de către o aplicaţie a funcţiilor *socket* şi acţiunile modulului TCP de expediere şi de recepţie a unor pachete.

Apelul `socket(PF_INET, SOCK_STREAM, 0)` are ca efect doar crearea unei structuri de date pentru apelurile ulterioare.

Pe partea de server, apelurile `bind()` şi `listen()` au, de asemenea, ca efect doar completarea unor informaţii în structurile de date. Dacă aplicaţia a apelat direct `listen()` fără a fi apelat înainte `bind()`, sistemul (modulul TCP) îi alocă un port liber.

La primirea unui pachet de deschidere conexiune (cu flagul *synchronize* setat) pentru o adresă şi un număr de port pentru care se aşteaptă deschiderea unei conexiuni (a fost deja apelat `listen()`), modulul TCP execută schimbul de pachete de deschidere a conexiunii. După aceea, modulul TCP aşteaptă ca aplicaţia să apeleze `accept()` pentru a-i putea semnaliza deschiderea unei noi conexiuni. În cazul în care nu se aşteaptă deschiderea unei conexiuni, modulul TCP trimite un pachet cu flagul *reset*.

Dacă, în adresa locală dată funcţiei `bind()`, s-a specificat constanta `INADDR_ANY`, este acceptat un pachet de deschidere de conexiune ce specifică ca adresă destinaţie oricare dintre adresele nodului curent. Dacă în apelul `bind()` s-a specificat o anumită adresă a nodului curent, este acceptată deschiderea conexiunii doar dacă adresa destinaţie a pachetului de deschidere este adresa specificată în apelul `bind()`.

Pe partea de client, apelul `connect()` este cel care determină trimiterea unui pachet cu flagul *synchronize*. Funcţia `connect()` aşteaptă fie deschiderea completă a conexiunii (confirmarea pachetului *syn* plus un pachet *syn* de la server), fie o semnalizare de eroare (un pachet TCP cu flagul *reset* sau un pachet ICMP). În caz favorabil, funcţia `connect()` returnează 0, în caz defavorabil semnalizează eroarea survenită.

10.3.1.8. Controlul fluxului

TCP are şi rol de control al fluxului, adică de-a încetini emiţătorul în cazul în care receptorul nu este capabil să proceseze datele suficient de repede.

O metodă extremă de control al fluxului este neconfirmarea de către receptor a octeţilor ce nu pot fi procesaţi. Metoda aceasta are însă dezavantajul că determină emiţătorul să retrimită octeţii respectivi, generând trafic inutil.

Metoda utilizată de TCP este ca receptorul să semnalizeze emiţătorului, prin valoarea câmpului *dimensiune fereastră* din antetul TCP, numărul de octeţi pe care receptorul este capabil să-i recepţioneze în acel moment. Emiţătorul nu va trimite mai mulţi octeţi decât dimensiunea ferestrei trimisă de receptor. Există o singură excepţie: dacă receptorul a semnalizat dimensiunea ferestrei 0, emiţătorul poate trimite un singur octet; aceasta se face pentru cazul în care receptorul a anunţat o fereastră de dimensiune 0 şi apoi a anunţat o fereastră mai mare dar pachetul ce anunţa mărirea ferestrei s-a pierdut.

Pe lângă mecanismul sus-menţionat, emiţătorul TCP reduce debitul de date emise şi în cazul în care constată pierderi de pachete. Ideea este că pachetele IP se pot pierde fie ca urmare a erorilor la nivel fizic, fie ca urmare a congestiei nodurilor intermediare. Cu excepţia transmisiei radio, erorile la nivel fizic sunt rare, astfel încât pierderile de pachete IP se datorează în majoritatea cazurilor congestiei nodurilor.

10.3.1.9. Stabilirea time-out-ului pentru retransmiterea pachetelor

Aşa cum am văzut, pachetele TCP neconfirmate într-un anumit interval de timp sunt retransmise. Valoarea aleasă a timpului după care se face retransmiterea influenţează performanţele transferului de date. O valoare prea mică duce la repetarea inutilă a unor pachete ce ajung la destinaţie însă într-un timp mai lung şi, ca urmare, la generarea unui trafic inutil. O valoare prea mare duce la detectarea cu întârziere a pachetelor pierdute.

Modulul TCP încearcă să estimeze durata de timp necesară unui pachet emis să ajungă la destinaţie, să fie procesat şi să se întoarcă şi să se recepţioneze confirmarea. Acest timp se numeşte *timp dus-întors* (engl. *round-trip time*, *RTT*). Timpul dus-întors nu este fix, ci depinde de perechea emiţător–receptor considerată şi de încărcarea reţelei în momentul considerat. Modulul TCP estimează statistic media şi dispersia timpului dus-întors pentru fiecare conexiune deschisă şi fixează timpul după care se retrimit pachetele neconfirmate la o valoare ceva mai mare decât media timpului dus-întors.

© 2008, Radu-Lucian Lupşa

10.3.1.10. Algoritmul lui Nagle şi optimizarea numărului de pachete

La primirea datelor de la programul aplicaţie, prin apelul sistem `send()`, modulul TCP poate trimite imediat un pachet sau poate aştepta; aşteptarea este utilă dacă astfel se pot plasa mai multe date în acelaşi pachet IP.

Algoritmul lui Nagle prevede că, la primirea unor date de la utilizator prin `send()`:

- dacă nu sunt date trimise şi neconfirmate, sau dacă datele noi sunt suficient de mari pentru a umple un pachet, datele se trimit imediat;
- altfel, modulul TCP aşteaptă până la primirea confirmării sau expirarea timpului de retransmitere, şi abia atunci trimite datele primite între timp de la aplicaţie.

O altă optimizare constă în a întârzia câteva fracţiuni de secundă trimiterea confirmării pentru un pachet TCP primit. Ideea este că, dacă aplicaţia care recepţionează datele din acel pachet are de trimis date ca răspuns la datele primite, datele ce constituie răspunsul să fie trimise în acelaşi pachet cu confirmarea datelor primite.

10.3.1.11. Trimiterea datelor speciale (out of band)

TCP prevede un mecanism de transmitere, în cadrul fluxului normal de date, a unor date cu un marcaj special. Mecanismul este întrucâtva echivalent cu a avea două fluxuri de date ataşate conexiunii, unul pentru datele „obişnuite“ şi celălalt pentru date „speciale“. Datele speciale poartă denumirea, în terminologia angloamericană, *out of band data* (*OOB*).

O posibilă utilizare este următoarea: presupunem o aplicaţie care transferă fişiere. Presupunem că emiţătorul trimite mai întâi lungimea fişierului şi apoi conţinutul. Dacă utilizatorul lansează trimiterea unui fişier mare (să zicem câţiva gigaocteţi) şi apoi se răzgândeşte şi doreşte să abandoneze operaţia, partea de emiţător a aplicaţiei nu poate semnaliza receptorului în nici un fel abandonarea transmiterii. Aceasta deoarece octeţii transmişi sunt interpretaţi de receptor ca fiind conţinutul fişierului. Aici intervin datele cu marcajul *special* (*out of band*): emiţătorul trimite conţinutul fişierului ca date normale, iar o eventuală semnalizare de abandonare a transferului este trimisă ca date speciale.

Datele speciale se consideră că trebuie să fie livrate cât mari repede cu putinţă aplicaţiei destinaţie. În terminologia TCP, ele sunt denumite *date urgente* (engl. *urgent data*). Transmiterea lor se face astfel:

- datele speciale se plasează într-un pachet TCP fără a fi precedate de date normale;

- pachetul respectiv are flagul *urgent* setat;
- câmpul *poziţie date urgente* conţine dimensiunea datelor speciale rămase de transmis (în pachetul curent şi în următoarele).

De principiu un pachet conţinând date speciale va avea setat şi flagul *push*, care indică faptul că modulul TCP receptor ar trebui să livreze datele către aplicaţia destinaţie cât mai repede.

10.3.2. Datagrame nesigure: UDP

Există aplicaţii care se descurcă acceptabil cu datagrame nesigure. Pentru acestea, nivelul transport trebuie să rezolve doar problema găsirii aplicaţiei căreia îi este destinată datagrama. Această problemă este rezolvată de protocolul UDP.

Fiecărei aplicaţii ce utilizează UDP i se acordă, de către modulul UDP al sistemului de operare local, un *port UDP*. Un port UDP alocat unei aplicaţii nu va fi acordat şi altei aplicaţii decât după ce este eliberat de prima.

O datagramă UDP poartă ca adrese sursă şi destinaţie perechi formate din câte o adresă IP şi un număr de port. Adresa IP este adresa nodului pe care rulează aplicaţia sursă, respectiv destinaţie, iar portul este portul alocat de sistem aplicaţiei.

O datagramă UDP este construită dintr-un pachet IP cu identificatorul protocolului UDP în *protocol* şi cu zona de date utile conţinând un *antet UDP* şi datele datagramei UDP. Antetul UDP conţine portul sursă şi portul destinaţie. Adresele IP sursă şi destinaţie sunt cele plasate în câmpurile corespunzătoare ale pachetului IP.

Deoarece o datagramă UDP trebuie să fie plasată într-un singur pachet IP, dimensiunea datelor utile ale unei datagrame UDP este limitată de dimensiunea maximă a pachetului IP.

Programul utilizator cere trimiterea unei datagrame UDP prin intermediul apelului `sendto()` sau `sendmsg()`. Programul trebuie să specifice adresa destinaţie — adresa IP şi portul. Adresa sursă este adresa asociată socket-ului de pe care se emite pachetul. Dacă nu s-a asociat în prealabil o adresă (prin apelul `bind()`), sistemul alocă automat un număr de port liber.

Deoarece, conform protocolului UDP, recepţia datagramei trimise nu este confirmată în nici un fel, funcţiile `sendto()` sau `sendmsg()` nu au cum să returneze programului apelant informaţii despre livrarea pachetului. Dealtfel, ambele funcţii se termină (returnează controlul apelantului) înainte ca pachetul să fie efectiv livrat.

Funcţiile sistem `recvfrom()` şi `recvmsg()` aşteaptă recepţia unei datagrame având ca adresă destinaţie adresa asociată socket-ului dat ca argu-

© 2008, Radu-Lucian Lupşa

ment. Funcţia returnează datele utile din datagramă precum şi adresa sursă a datagramei.

10.4. Identificarea nodurilor după nume: sistemul DNS

Utilizarea adreselor IP direct de către utilizatorii umani este dificilă deoarece:

- adresele IP sunt greu de ţinut minte de către oameni;
- adresele IP sunt legate de topologia reţelei şi se schimbă odată cu aceasta; spre exemplu, dacă o firmă schimbă furnizorul de Internet folosit, adresele staţiilor firmei este probabil să se schimbe.

De fapt, adresele IP sunt similare numerelor de telefon. Ca urmare, este utilă o „carte de telefon a Internetului“.

Serviciul numelor de domeniu — DNS (engl. *Domain Name Service*) — este un mecanism ce permite identificarea unui nod de reţea printr-un nume uşor de memorat de către om şi care să fie independent de topologia reţelei.

DNS este construit ca o bază de date ce cuprinde înregistrări ce asociază unui nume o adresă IP. Această bază de date este împărţită între mai multe *servere de nume*, care pot fi interogate de oricine. O aplicaţie care doreşte să comunice şi să folosească nume în loc de adrese IP interoghează întâi nişte servere de nume, până ce află adresa IP corespunzătoare numelui dat, după care lucrează cu adresa astfel obţinută.

Baza de date DNS este stocată distribuit (pe mai multe servere, pentru a nu avea un server supraaglomerat) şi redundant (există mai multe servere capabile să răspundă la o cerere dată).

DNS este proiectat în ideea că informaţiile se citesc frecvent şi se modifică rar. Cu ocazia modificărilor se admite să existe temporar incoerenţe — un utilizator să obţină informaţia nouă în timp ce altul deţine informaţia veche.

10.4.1. Numele de domeniu

Numele de domeniu [RFC 1034, 1987] sunt numele ce pot fi date nodurilor şi altor obiecte, în cadrul DNS. Un nume de domeniu este compus dintr-un şir de componente. Fiecare componentă este un şir de caractere; RFC 1034 nu impune restricţii, însă recomandă ca fiecare componentă să fie formată din cel mult 63 de caractere, ce pot fi litere, cifre sau caracterul *minus* (`-`), cu restricţia ca primul caracter să fie literă şi ultimul caracter să

nu fie minus. Literele mari sunt echivalente cu literele mici corespunzătoare. Componentele au asociată o ordine ierarhică.

Scrierea în text a unui nume de domeniu se face scriind componentele, începând cu cea mai de jos, din punct de vedere al ierarhiei, şi terminând cu cea mai de sus. După fiecare componentă se scrie un caracter *punct* (`.`). În particular, numele vid (format din zero componente) se scrie „`.`“ (un caracter punct).

Exemplul 10.12: În adresa `nessie.cs.ubbcluj.ro.` componentele sunt, în ordine descrescătoare ierarhic:

- `ro` — România,
- `ubbcluj` — Universitatea Babeş-Bolyai Cluj-Napoca,
- `cs` — Departamentul de Informatică (din engl. *Computer Science*),
- `nessie` — numele staţiei.

Această scriere este inspirată din scrierea adreselor poştale, care încep cu numele destinatarului şi se termină cu ţara.

În majoritatea cazurilor, aplicaţiile acceptă specificarea numelor de domenii fără punctul final. În lipsa punctului final, interpretarea este însă diferită. Anume, dacă numele nu este terminat cu punct, aplicaţia va încerca să adauge la nume şiruri de componente superioare ierarhic dintr-o listă configurată de administratorul sistemului. Primul nume de domeniu, astfel construit, care există în DNS este considerat ca fiind semnificaţia numelui dat de utilizator.

Exemplul 10.13: Presupunem că lista de căutare configurată de administrator pentru un sistem conţine, în ordine, `scs.ubbcluj.ro`, `cs.ubbcluj.ro` şi `ubbcluj.ro`. O căutare pentru numele `www` va duce la căutarea numelui `www.scs.ubbcluj.ro.` care va fi găsit şi vor fi returnate informaţiile despre acest nume. O căutare pentru numele `www.cs` va duce la căutarea, în ordine, a numelor `www.cs.scs.ubbcluj.ro.`, `www.cs.cs.ubbcluj.ro.` şi `www.cs.ubbcluj.ro.`; acesta din urmă este găsit şi căutarea este încheiată.

Structurarea numelui în mai multe componente serveşte la administrarea ierarhică a spaţiului de nume. O organizaţie care dobândeşte un nume de domeniu poate crea şi administra după voie numele formate prin adăugare de componente ierarhic inferioare. De exemplu, Universitatea Babeş-Bolyai din Cluj-Napoca a obţinut numele `ubbcluj.ro.` . Crearea numelui `nessie.cs.ubbcluj.ro.` este decizia exclusivă a Universităţii Babeş-Bolyai.

O instituţie care doreşte un nume de domeniu trebuie să contacteze instituţia care administrează domeniul părinte şi să ceară alocarea unui nume.

© 2008, Radu-Lucian Lupşa

Alocarea numelui se plăteşte — fie o taxă plătită o singură dată, fie o taxă anuală. Instituţia ce a obţinut numele este responsabilă de întreţinerea unui server de nume (vezi § 10.4.3 şi § 10.4.4) pentru domeniul ce i-a fost alocat.

Adesea firmele care oferă acces Internet oferă gratuit clienţilor nume de domeniu ca subdomenii ale domeniului firmei. Exemplu ipotetic, firma XYNet, posesoarea domeniului `xynet.example`, oferă clientului ABC s.r.l. domeniul `abc.xynet.example`. Din păcate, numele maşinilor firmei ABC sunt legate în acest caz de furnizorul de acces Internet, iar dacă firma ABC s.r.l. va schimba furnizorul de Internet, va fi nevoită să-şi schimbe numele de domeniu ale serverelor sale, ceea ce poate duce la pierderea clienţilor ce nu află noul nume al site-ului firmei.

De remarcat că, deşi organizarea ierarhică a numelor de domeniu seamănă cu organizarea numelor de fişiere şi directoare, nu există o restricţie similară cu aceea că într-un director nu e permis să aibă acelaşi nume un fişier şi un subdirector. Anume, ca exemplu, numele `ubbcluj.ro.` şi `cs.ubbcluj.ro.` pot desemna simultan noduri în reţea.

10.4.2. Structura logică a bazei de date DNS

Să ignorăm deocamdată problemele legate de implementarea DNS.

DNS se prezintă ca un tabel cu cinci coloane: *numele de domeniu*, *tipul*, *clasa*, *valoarea* şi *termenul de valabilitate*. Tipul şi clasa se utilizează pentru a putea pune în DNS şi alte informaţii în afară de adrese IP. Înregistrările corespunzătoare adreselor IP au tipul *A* (de la *address*=adresă) şi clasa *IN* (de la *Internet*). Câmpul *valoare* a unei înregistrări cu tipul *A* şi clasa *IN* conţine adresa IP a nodului cu numele de domeniu dat.

DNS permite căutarea unei înregistrări pentru care s-au specificat numele de domeniu, clasa şi tipul.

Este permis să existe mai multe înregistrări pentru acelaşi nume, clasă şi tip, dacă au valori diferite. Cineva care obţine prin interogarea DNS mai multe adrese IP pentru un nume dat poate folosi oricare din adrese; acestea se presupune că sunt ale aceluiaşi calculator sau ale unor calculatoare ce furnizează servicii echivalente.

O listă a tipurilor de înregistrări DNS mai des utilizate este dată în tabelul 10.7.

Remarcăm în mod deosebit tipul *CNAME* (nume canonic). O înregistrare având numele *nume1*, tipul *CNAME* şi valoarea *nume2* defineşte *nume1* ca pseudonim pentru *nume2*. În acest caz, *nume2* este considerat numele canonic al acelui obiect. Dacă pentru un nume există o înregistrare *CNAME*, nu este permis să mai existe vreo altă înregistrare pentru acel nume.

Tip	Valoare	Observaţii
A	adresă IPv4	adresa corespunzătoare numelui solicitat
AAAA	adresă IPv6	adresa IPv6 corespunzătoare numelui solicitat ([RFC 3596, 2003])
CNAME	nume de domeniu	numele canonic corespunzător numelui solicitat
PTR	nume de domeniu	numele canonic al nodului cu adresa IP solicitată, vezi § 10.4.6
SOA	date de identificare ale informaţiilor despre zonă	vezi § 10.4.5
NS	nume de domeniu	numele canonic al serverului de domeniu pentru zona având ca rădăcină numele solicitat
MX	nume de domeniu şi prioritate	serverele de poştă electronică pentru domeniul solicitat, § 11.1

Tabelul 10.7: Tipuri de înregistrări DNS mai des folosite

Mai mult, numele canonic din câmpul *valoare* al unei înregistrări *CNAME* nu este permis să apară ca nume de domeniu în altă înregistrare *CNAME*.

O aplicaţie care caută o înregistrare de un anumit tip pentru un nume trebuie să caute şi o înregistrare *CNAME* pentru acel nume. Dacă găseşte o înregistrare *CNAME*, trebuie să caute o înregistrare cu numele canonic găsit şi având tipul căutat iniţial. Valoarea astfel găsită trebuie utilizată ca şi când ar fi fost găsită pentru numele original.

10.4.3. Împărţirea în domenii de autoritate

Mulţimea numelor de domeniu este împărţită în *zone*. Pentru fiecare zonă există unul sau mai multe *servere de nume* sau *server DNS* care deţin toate înregistrările corespunzătoare numelor din acea zonă.

Privim spaţiul de nume ca un arbore în care rădăcina este domeniul rădăcină şi fiecare nume are ca părinte numele obţinut din el prin înlăturarea celei mai din stânga componente. O zonă este o submulţime de nume care, împreună cu legăturile dintre ele, formează un arbore. De remarcat că rădăcina zonei face parte din zonă.

Un server care este responsabil de o zonă trebuie să deţină toate înregistrările corespunzătoare numelor din zonă. Faptul că un nume care ar

© 2008, Radu-Lucian Lupşa

aparţine zonei nu figurează în tabela ţinută de acel server înseamnă că numele respectiv nu există.

Din motive de toleranţă la pene, pentru fiecare zonă există de regulă cel puţin două servere responsabile. De principiu, tabelele despre o zonă dată ale serverelor responsabile de acea zonă trebuie să fie identice; pot exista însă temporar incoerenţe între tabele cu ocazia modificărilor unor informaţii.

Pe lângă înregistrările despre zonele pentru care este responsabil, un server de nume trebuie să mai deţină înregistrări care să permită regăsirea serverelor de nume ale zonelor adiacente — zona imediat superioară ierarhic şi zonele imediat subordonate, dacă există — şi a serverelor de nume pentru zona rădăcină. Aceste înregistrări se numesc *glue records* — înregistrări lipici — deoarece crează legătura cu zonele învecinate.

Pentru numele rădăcină al fiecărei zone trebuie să existe următoarele înregistrări:

- O înregistrare *SOA* (*Start Of Authority*), care conţine nişte date administrative despre zonă (vezi § 10.4.5);
- Una sau mai multe înregistrări *NS* (*Name Server*) care conţin numele serverelor de nume responsabile de zonă. De remarcat că serverele de nume nu este obligatoriu să fie ele însele membre ale zonei.

Înregistrările *NS* ale rădăcinii unei zone împreună cu înregistrările *A* ale acelor nume sunt „înregistrările lipici“ ce trebuie ţinute de un server cu privire la zonele vecine.

Un server poate ţine şi alte înregistrări, în afară de înregistrările din zonele pentru care este responsabil şi de înregistrările de legătură.

10.4.4. Mecanismul de interogare a serverelor

Protocolul utilizat pentru interogarea serverelor de nume este descris în [RFC 1035, 1987].

Un server de nume aşteaptă cereri prin datagrame UDP trimise pe portul 53 şi prin conexiuni TCP pe portul 53. Clientul trimite cererea întâi ca datagramă UDP. Dacă răspunsul este prea lung pentru a încape într-o datagramă UDP atunci clientul repune întrebarea printr-o conexiune TCP.

Interogarea cuprinde numele de domeniu căutat, tipul şi clasa. Răspunsul conţine interogarea şi un şir de înregistrări, împărţite în trei categorii: înregistrări din zonele pentru care este responsabil, înregistrări de legătură şi alte înregistrări.

Dacă numele din interogare este dintr-o zonă pentru care serverul este responsabil, serverul va răspunde cu înregistrarea sau înregistrările care constituie răspunsul la interogare — eventual va spune că nu există înregistrări.

Dacă numele căutat este din afara zonei de responsabilitate, există două comportamente posibile pentru server:

- *iterativ*. Serverul răspunde cu înregistrările de legătură către zona căutată de client. Clientul urmează să interogheze alte servere.
- *recursiv*. Serverul interoghează (el însuşi) un server pentru zona vecină mai apropiată de zona numelui căutat de client, eventual interoghează în continuare servere din înregistrările returnate, până ce află răspunsul la întrebarea clientului. Înregistrările găsite sunt plasate în categoria a treia — alte înregistrări.

  Un server recursiv poate fi configurat să reţină înregistrările astfel obţinute, astfel încât la o interogare ulterioară să poată răspunde direct, fără a mai căuta un server responsabil pentru numele cerut de client. O înregistrare astfel memorată este ţinută un timp cel mult egal cu termenul de valabilitate al înregistrării, după care se consideră că informaţia s-ar fi putut modifica şi ca urmare înregistrarea este aruncată.

Căutarea adresei corespunzătoare unui nume se face de către programul utilizator care are de contactat nodul cu numele dat. Interogarea DNS se face de regulă prin intermediul unor funcţii de bibliotecă, cum ar fi `gethostbyname()`. Aceste funcţii de bibliotecă trebuie să găsească un prim server de nume pe care să-l interogheze. Pentru aceasta, în sistemul de operare există un loc standardizat unde administratorul scrie adresele IP ale unuia sau mai multor servere de nume. În sistemele de tip Unix, locul este fişieru `/etc/resolv.conf`, iar în Windows este în registry.

10.4.5. Sincronizarea serverelor pentru un domeniu

Protocolul de interogare DNS prevede posibilitatea de-a cere toate înregistrările dintr-o anumită zonă ([RFC 1035, 1987], [RFC 1995, 1996]). Acest lucru se utilizează pentru a putea întreţine uşor mai multe servere responsabile de o anumită zonă. Toate înregistrările privind zona se scriu în baza de date a unuia dintre servere, denumit *master*. Celelalte servere, numite *slave*, sunt configurate să copieze periodic informaţiile de pe *master*.

De notat că, într-o astfel de configuraţie, atât serverul *master* cât şi serverele *slave* sunt considerate responsabile de zonă; mecanismul este invizibil pentru cineva care face o interogare DNS pentru un nume din zonă.

Momentele la care un server *slave* copiază datele de pe serverul *master* depind de următorii parametri din înregistrarea *SOA* a zonei:

- *serial* este numărul de ordine al datelor; administratorul zonei trebuie să crească numărul serial oridecâteori modifică vreo înregistrare din zonă.

© 2008, Radu-Lucian Lupşa

Un *slave* nu execută copierea dacă numărul serial curent al *master*-ului coincide cu numărul serial propriu.

- *refresh* este timpul după care un server *slave* trebuie să interogheze serverul *master* pentru a vedea dacă s-a modificat vreo înregistrare;
- *retry* este timpul de aşteptare după o încercare eşuată de-a contacta serverul *master* înainte de-a încerca din nou;
- *expire* este timpul după care, în cazul în care nu a reuşit să contacteze serverul *master*, serverul *slave* trebuie să nu se mai considere responsabil de zonă.

Există şi un protocol ([RFC 1996, 1996]) prin care serverul *master* cere explicit unui server *slave* să copieze datele despre zonă.

10.4.6. Căutarea numelui după IP

DNS nu permite căutarea unei înregistrări după valoare, deoarece găsirea serverului ce deţine înregistrarea ar necesita interogarea tuturor serverelor DNS din Internet.

Pentru a putea răspunde la întrebări de tipul *cine are o adresă IP dată,* se utilizează următorul mecanism:

Fiecărei adrese IP versiunea 4 i se asociază un nume de domeniu astfel: se scrie adresa în formă zecimală cu punct, cu componentele în ordine inversă, şi se adaugă `in-addr.arpa.` . Astfel, adresei IP 193.0.225.34 îi corespunde numele `34.225.0.193.in-addr.arpa.` .

Pentru aceste nume se pun în DNS înregistrări cu tipul *PTR* şi având ca valoare numele de domeniu al nodului respectiv. Interogarea şi administrarea acestor nume de domeniu se fac întocmai ca şi pentru numele obişnuite.

De principiu, un subdomeniu din `in-addr.arpa.` corespunde unui bloc de adrese IP alocat unei instituţii; subdomeniul corespunzător din domeniul `in-addr.arpa.` este administrat de aceeaşi instituţie.

În situaţia în care alocarea blocurilor de adrese IP se face după schema CIDR, graniţele blocurilor nu coincid cu graniţele subdomeniilor lui `in-addr.arpa.` . De exemplu, dacă firma X are alocat blocul de adrese 193.226.40.128/28, ea nu va putea primi în administrare întregul domeniu `40.226.193.in-addr.arpa.`, deoarece acesta conţine şi alte adrese decâte cele ale firmei X. Administrarea numelor din `in-addr.arpa.` se face, în acest caz, conform [RFC 2317, 1998]: numele corespunzătoare IP-urilor se definesc ca pseudonime pentru nişte nume din domenii create special pentru blocurile alocate. Pentru exemplul dat, construcţia este următoarea:

- se crează domeniul `128/28.40.226.193.in-addr.arpa.`, asociat blocului

193.226.40.128/28, a cărui administrare este delegată firmei X;

- numele de domeniu de la `128.40.226.193.in-addr.arpa.` până la `143.40.226.193.in-addr.arpa.` se definesc ca pseudonime (*CNAME*) pentru numele de la `128.128/28.40.226.193.in-addr.arpa.` până la `143.128/28.40.226.193.in-addr.arpa.`

Pentru adresele IPv6 se foloseşte un mecanism asemănător, definit în [RFC 3596, 2003]. Anume, fiecărei adrese IPv6 i se asociază un nume în domeniul `ip6.arpa` . Numele se construieşte astfel:

- Se iau grupuri de câte 4 biţi din adresa IPv6 şi se scrie cifra hexa corespunzătoare ca o componentă separată.
- Ordinea ierarhică a componentelor astfel obţinute este aceea în care componentele corespunzătoare biţilor mai semnificativi din adresa IP sunt superioare ierarhic componentelor corespunzătoare biţilor mai puţin semnificativi.

Exemplul 10.14: Pentru adresa `4321:0:1:2:3:4:567:89ab`, numele de domeniu asociat este

```
b.a.9.8.7.6.5.0.4.0.0.0.3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.0.1.2.3.4.
        ip6.arpa
```

10.5. Legăturile directe între nodurile IP

10.5.1. Rezolvarea adresei — ARP

În multe cazuri, ceea ce, din punctul de vedere al unei reţele Internet este o legătură directă, este de fapt o legătură între două noduri într-o reţea de alt tip, de exemplu o reţea IEEE 802. O astfel de reţea joacă rolul nivelului legăturii de date în contextul protocolului Internet şi ca urmare o vom numi aici *reţea de nivel inferior*.

Transmiterea unui pachet IP de la un nod A către un nod B, în cadrul aceleiaşi reţele de nivel inferior, se face astfel:

- mai întâi, nodul A determină adresa, în cadrul reţelei de nivel inferior (adică adresa MAC, pentru IEEE 802), a nodului B;
- apoi A trimite pachetul IP nodului B, sub formă de date utile în cadrul unui pachet al reţelei de nivel inferior, pachet destinat adresei găsite anterior.

© 2008, Radu-Lucian Lupşa

Determinarea adresei MAC a nodului B se face cu ajutorul unui mecanism numit *ARP* (engl. *Address Resolution Protocol*, rom. *protocol de determinare a adresei*). Mecanismul funcţionează astfel:

- A trimite în reţeaua de nivel inferior un pachet de difuziune (broadcast) conţinând adresa IP a lui B. Acest pachet este un fel de întrebare „cine are adresa IP cutare?“.
- La un astfel de pachet răspunde doar nodul cu adresa IP specificată în pachet — adică doar nodul B în cazul de faţă. Pachetul de răspuns este adresat direct lui A (prin adresa MAC a lui A recuperată din interogare) şi conţine adresa IP şi adresa MAC ale lui B. Pachetul are semnificaţia „eu am adresa IP cutare şi am adresa MAC cutare“.

După determinarea corespondenţei IP $\rightarrow$ MAC prin mecanismul ARP, nodul A păstrează corespondenţa în memorie un anumit timp (de ordinul minutelor), astfel încât, dacă nodurile A şi B schimbă mai multe pachete în timp scurt, mecanismul ARP este invocat doar o singură dată.

Dacă nodul A primeşte mai multe răspunsuri ARP cu adrese MAC diferite pentru aceeaşi adresă IP, înseamnă că există mai multe noduri cărora li s-a dat din greşeală aceeaşi adresă IP. În această situaţie A ar trebui să semnalizeze situaţia:

- printr-un pachet ICMP cu tipul *destination unreachable* destinat sursei pachetului destinat lui B, şi
- printr-un mesaj afişat pe ecran şi înregistrat în fişierele jurnal ale sistemului de operare.

Mecanismul ARP este utilizat de asemnea la configurarea adresei unei interfeţe de reţea ca verificare că adresa configurată este unică în subreţea. Mai exact, dacă administratorul configurează o anumită adresă IP pentru o interfaţă, sistemul emite mai întâi o cerere ARP pentru adresa ce urmează a fi setată. Dacă cererea primeşte răspuns înseamnă că mai există un nod ce are adresa respectivă, caz în care sistemul tipăreşte un mesaj de avertisment şi eventual refuză configurarea adresei respective. Dacă cererea nu primeşte răspuns este probabil ca adresa să fie unică şi ca urmare sistemul o poate accepta ca adresă proprie.

Informaţiile despre asocierile IP $\rightarrow$ MAC cunoscute nodului curent se determină, pe sistemele de tip UNIX, cu ajutorul comenzii `arp`. Trimiterea, în vedera testării, a unei cereri ARP către o staţie se poate face cu ajutorul comenzii `arping`.

© 2008, Radu-Lucian Lupşa

10.6. Configurarea automată a staţiilor — DHCP

Sunt situaţii în care este util ca un nod să-şi determine propria adresă IP, împreună cu alţi câţiva parametri (masca de reţea, *default gateway*, servere DNS) prin interogări în reţea, în loc ca aceşti parametri să fie stocaţi într-o memorie nevolatilă (disc sau memorie flash) a nodului. Situaţii în care acest lucru este util sunt:

- pentru un calculator fără harddisc;
- pentru un laptop, PDA sau alt dispozitiv mobil, care este mutat frecvent dintr-o reţea în alta, unde parametrii trebuie configuraţi de fiecare dată în funcţie de reţeaua la care este conectat;
- într-o reţea mare în care este de dorit ca parametrii de reţea ai staţiilor să poată fi schimbaţi uşor de pe un calculator central.

Există trei protocoale ce au fost utilizate de-a lungul timpului pentru determinarea parametrilor de reţea: RARP, BOOTP şi DHCP. Vom studia mai în detaliu protocolul DHCP, celelalte două nemaifiind utilizate în prezent. De notat însă că protocolul DHCP este conceput ca o extensie a protocolului BOOTP.

Un nod care doreşte să-şi determine parametrii de reţea (adresa IP proprie, masca de reţea, *default gateway*-uri, numele propriu, adresele serverelor DNS) se numeşte *client DHCP*. Clientul DHCP trimite o cerere, la care răspunde un *server DHCP* stabilit pentru reţeaua respectivă. Răspunsul conţine parametrii solicitaţi. Vom studia în continuare:

- cum se transmit cererea şi răspunsul DHCP, în condiţiile în care modulul IP al clientului nu este configurat şi ca urmare nu este complet funcţional;
- cum determină serverul parametri clientului.

Transmiterea cererii şi a răspunsului DHCP. Presupunem în continuare că nodul client este conectat la o singură reţea de tip IEEE 802.

Cererea DHCP este transmisă ca un pachet UDP. Adresa IP destinaţie a pachetului este adresa de broadcast locală (255.255.255.255), iar portul destinaţie este portul standard pe care ascultă serverul DHCP, anume portul 67. Adresa IP sursă este 0.0.0.0 (valoarea standard pentru adresă necunoscută). Pachetul IP este încapsulat într-un pachet Ethernet destinat adresei de broadcast (FF:FF:FF:FF:FF:FF) şi purtând ca adresă sursă adresa plăcii de reţea locale. Serverul DHCP trebuie să fie în aceeaşi subreţea cu clientul

© 2008, Radu-Lucian Lupşa

(sau să existe în aceeaşi subreţea un server *proxy DHCP* care să preia cererea clientului şi s-o retrimită serverului).

Răspunsul DHCP este plasat la rândul lui într-un pachet UDP purtând ca adresă sursă adresa serverului DHCP şi ca adresă destinaţie adresa alocată clientului DHCP, cu portul destinaţie 68. Pachetul este încapsulat într-un pachet Ethernet destinat adresei MAC a clientului. Asocierea IP → MAC pentru transmiterea pachetului de către server nu se face prin mecanismul ARP (care ar eşua deoarece clientul DHCP nu poate încă răspunde la cererea ARP) ci este setată de către serverul DHCP prin intermediul unui apel sistem.

Determinarea parametrilor de către server. Pentru fiecare subreţea pentru care acţionează, un server DHCP trebuie să aibă două categorii de date: parametrii comuni tuturor nodurilor din subreţea (masca de reţea, gateway-uri, servere DNS) şi parametrii specifici fiecărui nod în parte (adresa IP a nodului).

Parametrii comuni sunt setaţi de administratorul serverului DHCP într-un fişier de configurare al serverului.

Pentru adresele IP ale nodurilor din reţea există două strategii ce pot fi folosite:

Alocare manuală: Adresa fiecărui nod este fixată manual de către administratorul serverului DHCP. Nodul client este identificat prin adresa MAC sau prin nume. În primul caz administratorul trebuie să scrie într-un fişier de configurare al serverului toate adresele MAC ale plăcilor de reţea şi adresele IP corespunzătoare. În al doilea caz, pe serverul DHCP se scriu numele staţiilor şi adresele corespunzătoare iar pe fiecare staţie se setează numele staţiei (a doua soluţie nu este aplicabilă pe calculatoare fără harddisc).

Alocare dinamică: Serverul dispune de o mulţime de adrese pe care le alocă nodurilor. Serverul păstrează corespondenţa dintre adresele MAC ale clienţilor şi adresele IP ce le-au fost alocate. Adresele alocate pot fi eliberate la cererea explicită a clientului sau la expirarea perioadei de alocare (vezi mai jos).

Adresele se atribuie de regulă pe o durată determinată. Perioada de alocare poate fi prelungită la solicitarea clientului printr-o cerere DHCP de prelungire. După expirarea perioadei de alocare, clientul nu mai are voie să utilizeze adresa.

Atribuirea adreselor pe perioadă determinată are două avantaje (faţă de atribuirea pe durată nedeterminată):

- la alocarea dinamică a adreselor, dacă clientul este scos din reţea fără

a elibera explicit adresa, adresa este eliberată automat la expirarea perioadei de atribuire;

- dacă este necesară modificarea strategiei de atribuire a adreselor se pot opera modificările necesare în configurarea serverului DHCP iar clienţii vor primi noile adrese cu ocazia încercării de prelungire a atribuirii adresei.

10.7. Situaţii speciale în dirijarea pachetelor

Vom studia în paragraful de faţă anumite procedee mai deosebite utilizate în dirijarerea pachetelor. Aceste procedee se aplică îndeosebi în reţelele interne ale unor instituţii.

10.7.1. Filtre de pachete (firewall)

Un *filtru de pachete* (engl. *firewall*) este un nod IP care nu transmite toate pachetele conform regulilor normale de funcţionare ale unui nod IP ci, în funcţie de anumite reguli, ignoră complet sau trimite pachete ICMP de eroare pentru anumite pachete.

Scopul unui filtru de pachete este de-a interzice anumite acţiuni în reţea, în special pentru a contracara anumite încercări de spargere a unui calculator.

Configurarea unui filtru de pachete constă în stabilirea unui ansamblu de *reguli de filtrare*. Prezentăm în continuare posibilităţile oferite de mecanismul *iptables* din sistemul Linux, cu menţiunea că facilităţile de bază se regăsesc în toate sistemele.

O *regulă de filtrare* este o pereche formată dintr-o *condiţie* şi o *acţiune*. Regulile sunt grupate în şiruri numite *lanţuri*. Există trei lanţuri predefinite:

- *INPUT* aplicat pachetelor destinate nodului curent,
- *OUTPUT* aplicat pachetelor generate de nodul curent,
- *FORWARD* aplicat pachetelor generate de alt nod şi având ca destinaţie alt nod (pentru care nodul curent acţionează ca ruter).

Pentru fiecare pachet ajuns la modulul IP, acesta aplică prima regulă, din lanţul corespunzător traseului pachetului, pentru care pachetul îndeplineşte condiţia specificată în regulă. Aplicarea regulii înseamnă executarea acţiunii specificate de regulă.

Principalele *acţiuni* ce pot fi specificate sunt:

- *ACCEPT* — pachetul este livrat normal,
- *DROP* — pachetul este ignorat (ca şi când nu ar fi fost primit),

© 2008, Radu-Lucian Lupşa

- *REJECT* — se trimite înapoi un pachet semnalând o eroare — implicit *ICMP destination unreachable*.

Condiţiile specificate într-o regulă pot privi:

- interfaţa prin care a intrat pachetul (cu excepţia lanţului *OUTPUT*),
- interfaţă prin care ar ieşi pachetul (cu excepţia lanţului *INPUT*),
- adresa IP sursă şi adresa IP destinaţie (se poate specifica şi un prefix de reţea, condiţia fiind satisfăcută de pachetele ce au adresă începând cu acel prefix sau, eventual, pachetele ce au adresă ce nu începe cu acel prefix),
- adresa MAC sursă sau destinaţie (pentru pachete ce intră, respectiv ies, prin interfeţe ce au conceptul de adresă MAC),
- protocolul (TCP, UDP, ICMP),
- portul sursă sau destinaţie (pentru protocoale care au noţiunea de port),
- tipul şi subtipul ICMP (pentru pachete ICMP),
- flag-uri ale diverselor protocoale,
- dimensiunea pachetului,
- starea conexiunii TCP căreia îi aparţine pachetul (vezi mai jos).

Un nod (intermediar) prin care trec toate pachetele asociate unei conexiuni TCP poate, examinând antetul TCP al fiecărui pachet, să ţină evidenţă stării conexiunii. Ca urmare, nodul poate stabili dacă un pachet deschide o conexiune nouă, aparţine unei conexiuni deschise sau este un pachet invalid.

Este adevărat, acest lucru înseamnă o încălcare a principiului separării nivelelor: TCP este un protocol de nivel transport, deasupra nivelului reţea. Ca urmare, modulele de reţea nu ar trebui să interpreteze protocolul TCP (antetele TCP ar trebui considerate pur şi simplu date utile). Ca urmare, nodurile intermediare, din care nu acţionează asupra pachetelor în tranzit decât modulul reţea şi modulele inferioare, nu ar trebui să „înţeleagă“ protocolul TCP.

Regulile de filtrare se configurează de către administratorul sistemului. Ca şi în cazul parametrilor IP:

- Pe sistemele Linux, regulile aplicate de nucleul sistemului de operare se examinează şi se modifică cu ajutorul unei comenzi — `iptables`. Regulile valabile la iniţializarea sistemului sunt configurate de *script*-urile invocate la pornire, fiind încărcate dintr-un fişier text.

- Pe sistemele Windows există o interfaţă grafică cu care se configurează simultan regulile curente aplicate de nucleu şi în acelaşi timp acele reguli sunt scrise în *registry* pentru a fi încărcate la repornirea sistemului.

Prin regulile de filtrare se urmăresc de obicei următoarele lucruri:

- Să fie blocate pachetele pentru care se poate determina că adresa sursă a fost falsificată. Aici intră:
  - pachete ce intră pe interfaţă către Internet şi au ca adresă sursă o adresă din reţeaua internă,
  - pachete ce au ca sursă o adresă de broadcast (clasa D) sau de clasă E,
  - pachete ce intră dinspre o anumită subreţea au ca sursă o adresă ce nu face parte din subreţeaua respectivă şi nici din alte subreţele din direcţia respectivă.

- Să fie interzise conexiunile din afara reţelei locale către servicii care sunt oferite doar pentru reţeaua locală. De exemplu, accesul la un *share* Windows este adesea de dorit să nu fie posibil din afara reţelei locale.

  Pentru aceasta există două strategii:
  - se blochează pachetele destinate porturilor pe care aşteaptă conexiuni serviciile respective;
  - se permit conexiunile către serviciile ce se doresc a fi accesibile din afară (web, mail, eventual ssh), se permit conexiunile iniţiate din interior şi se interzic toate celelalte pachete.

  Prima metodă este mai simplă însă necesită lista completă a serviciilor ce trebuie blocate. A doua metodă este mai sigură, întrucât serviciile sunt inaccesibile dacă nu s-a specificat explicit contrariul, însă este dificil de permis intrarea pachetelor de răspuns pentru conexiunile iniţiate din interior. Aceasta se întâmplă deoarece o conexiune iniţiată din interior are alocat un port local cu număr imprevizibil; ca urmare nu se poate stabili o regulă simplă pentru permiterea intrării pachetelor destinate acelui port.

  Soluţia uzuală este:
  - pentru conexiun TCP, se permit pachetele asociate unei conexiuni deja deschise, se permit pachetele către exterior, se permit pachetele destinate serviciilor publice şi se interzic toate celelalte pachete.

© 2008, Radu-Lucian Lupşa

- pentru UDP, unde nu se poate ţine evidenţa unor conexiuni, se interzic pachetele destinate unor servicii private, se permit pachetele spre exterior, se permit pachetele provenite de la serviciile ce se doreşte a fi accesate în exterior (serverele DNS sau NTP utilizate) şi se interzic toate celelalte pachete.

- Să fie interzise diferite alte pachete „dubioase“, cum ar fi:
  - pachete destinate adresei de broadcast a reţelei locale sau adresei de broadcast generale (255.255.255.255),
  - pachete având ca adresă sursă sau destinaţie adresa maşinii locale (127.0.0.1) sau o adresă privată.

Exemplul 10.15: Fie un ruter având în spate o reţea internă având adrese cu prefixul 193.226.40.128/28. Ruterul are interfaţa *eth0* cu adresa 193.0.225.20 către exterior şi interfaţa *eth1* cu adresa 193.226.40.129 către subreţeaua locală.

Din reţeaua locală dorim să se poată deschide orice fel de conexiuni TCP către exterior; din exterior dorim să nu se poată deschide alte conexiuni decât către serverul *http* şi *https* de pe 193.226.40.130 şi către serverele *ssh* de pe toate maşinile din reţeaua locală.

Din reţeaua locală dorim să putem accesa servicii DNS şi NTP. Acestea le furnizăm astfel:

- pe ruter instalăm un server DNS şi un server NTP, accesibile din reţeaua locală; acestea furnizează serviciile respective pentru reţeaua locală
- permitem cererile emise de serverele DNS şi NTP de pe ruter, precum şi răspunsurile corespunzătoare. Cererile NTP provin de pe portul UDP 123 al ruterului şi sunt adresate portului UDP 123 al unui nod din exterior, iar cererile DNS sunt emise de pe un port UDP mai mare sau egal cu 1024 şi sunt adresate portului DNS 53 de pe un nod extern.

Pentru diagnosticarea funcţionării reţelei vom mai permite trecerea pachetelor ICMP *ping*, *pong*, *destination unreachable* şi *time exceeded*.

Traficul în interiorul reţelei locale îl permitem fără restricţii.

Blocăm toate încercările de *spoofing* detectabile.

```
# blocare spoofing detectabil si alte pachete dubioase
iptables -A FORWARD -i eth0 -s 193.226.40.128/28 -j DROP
iptables -A FORWARD -i eth0 -s 193.0.225.20 -j DROP
iptables -A FORWARD -i eth0 -s 127.0.0.0/8 -j DROP
```

```
iptables -A FORWARD -i eth0 -s 0.0.0.0/8 -j DROP
iptables -A FORWARD -i eth0 -s 224.0.0.0/3 -j DROP
iptables -A FORWARD -i eth0 -s 10.0.0.0/8 -j DROP
iptables -A FORWARD -i eth0 -s 172.30.16.0/12 -j DROP
iptables -A FORWARD -i eth0 -s 192.168.0.0/16 -j DROP
iptables -A FORWARD -i eth1 -s ! 193.226.40.128/28 -j DROP
iptables -A FORWARD -d 255.255.255.255 -j DROP
iptables -A FORWARD -i eth0 -d 193.226.40.159 -j DROP
iptables -A INPUT -i eth0 -s 193.226.40.128/28 -j DROP
iptables -A INPUT -i eth0 -s 193.0.225.20 -j DROP
iptables -A INPUT -i eth0 -s 127.0.0.0/8 -j DROP
iptables -A INPUT -i eth0 -s 0.0.0.0/8 -j DROP
iptables -A INPUT -i eth0 -s 224.0.0.0/3 -j DROP
iptables -A INPUT -i eth0 -s 10.0.0.0/8 -j DROP
iptables -A INPUT -i eth0 -s 172.30.16.0/12 -j DROP
iptables -A INPUT -i eth0 -s 192.168.0.0/16 -j DROP
iptables -A INPUT -i eth1 -s ! 193.226.40.128/28 -j DROP
iptables -A INPUT -d 255.255.255.255 -j DROP
iptables -A INPUT -i eth0 -d 193.226.40.159 -j DROP
# celelalte restrictii
iptables -A INPUT -i eth1 -j ACCEPT
iptables -A FORWARD -i eth1 -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED -j ACCEPT
iptables -A INPUT -i eth0 -p udp --dport 1-1023 -j DROP
iptables -A INPUT -i eth0 -p udp --sport 53 -j ACCEPT
iptables -A INPUT -i eth0 -p udp --sport 123 --dport 123 -j ACCEPT
iptables -A FORWARD -d 193.226.40.130 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -d 193.226.40.130 -p tcp --dport 443 -j ACCEPT
iptables -A FORWARD -d 193.226.40.128/28 -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
iptables -A INPUT -p icmp --icmp-type destination-unreachable \
    -j ACCEPT
iptables -A INPUT -p icmp --icmp-type time-exceeded -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type echo-request -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type echo-reply -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type destination-unreachable \
    -j ACCEPT
iptables -A FORWARD -p icmp --icmp-type time-exceeded -j ACCEPT
iptables -A INPUT -j DROP
iptables -A FORWARD -j DROP
```

© 2008, Radu-Lucian Lupşa

10.7.2. Reţele private

O *reţea privată* este o reţea, conectată sau nu la Internet, a cărei calculatoare nu pot comunica direct cu calculatoarele din Internet.

O utilizare tipică este cea prezentată în figura 10.7. O instituţie A are o reţea proprie de calculatoare. Din această reţea proprie, o parte dintre calculatoare — să le numim *publice* — trebuie să comunice nerestricţionat cu alte calculatoare din Internet, în vreme ce restul calculatoarelor — le vom numi *private* — este acceptabil să poată comunica doar cu alte calculatoare din reţeaua internă.

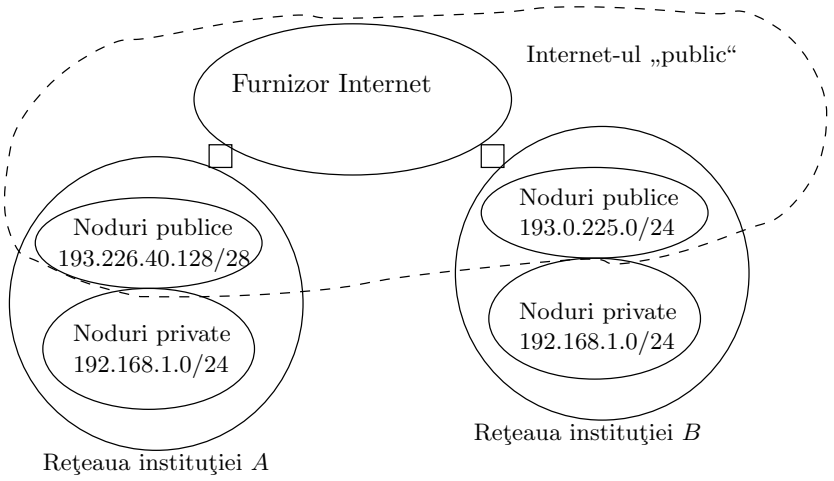

Figura 10.7: Două reţele locale având fiecare aceleaşi adrese private pentru o parte din calculatoare.

Calculatoarele private nu este necesar să aibă adrese unice în Internet. Adresele calculatoarelor private pot fi refolosite de către calculatoare private din alte astfel de reţele interne, de exemplu de cele ale instituţiei B din figură. De fapt, în general, o adresă trebuie să fie unică doar în mulţimea nodurilor cu care un anumit nod ar putea dori să comunice.

În situaţia descrisă, este necesar ca adresele din Internet-ul „public“ să fie unice, adresele din reţeaua locală să fie unice şi să nu existe suprapuneri între adresele din Internet şi adresele din reţeaua locală. Cerinţa din urmă este determinată de cerinţa ca nodurile cu adrese publice din reţeaua proprie să poată comunica şi cu nodurile private şi cu nodurile din Internet.

Un pachet a cărui adresă destinaţie este o adresă privată este dirijat de către rutere astfel:

- dacă ruterul face parte dintr-o reţea locală în care există acea adresă,

pachetul este dirijat către unicul nod din reţeaua locală purtând adresa respectivă;

- altfel, ruter-ul declară pachetul nelivrabil.

Aşa cum am văzut în § 10.2.4.1, următoarele blocuri de adrese IP sunt alocate pentru astfel de utilizări în reţele private: 10.0.0.0/8, 172.16.0.0/12 şi 192.168.0.0/16.

De cele mai multe ori, calculatoarelor private li se oferă posibilităţi limitate de-a comunica cu calculatoare din Internet, prin intermediul unor mecanisme descrise în paragrafele următoare.

10.7.3. Translaţia adreselor (NAT)

Translaţia adreselor este un mecanism prin care un ruter modifică adresa sursă sau adresa destinaţie a unor pachete.

10.7.3.1. Translaţia adresei sursă

Presupunem că avem o reţea privată şi dorim ca de pe calculatoarele cu adrese private să se poată deschide conexiuni către calculatoare din Internet — spre exemplu, pentru a putea accesa pagini web. Fără vreun mecanism special, acest lucru nu este posibil, din următorul motiv: Un calculator C cu adresă privată care doreşte să deschidă o conexiune către un calculator S din Internet trimite un pachet IP având ca adresă sursă adresa proprie (privată) C, ca adresă destinaţie adresa serverului S (adresă care este publică) şi conţinând o cerere de deschidere de conexiune TCP. Pachetul ajunge la destinaţie, iar serverul S vom presupune că acceptă conexiunea. Serverul S trimite înapoi un pachet IP având ca adresă sursă adresa proprie S şi ca adresă destinaţie adresa, privată, a clientului C. Deoarece adresa clientului nu este unică la nivelul Internet-ului (ci doar la nivelul propriei reţele interne), pachetul de răspuns nu poate fi livrat.

Translaţia adresei sursă rezolvă problema de mai sus în modul următor: În primul rând, trebuie să existe un nod (ruter) G în reţeaua internă, având adresă publică şi prin care să tranziteze toate pachetele de la C către S (vezi fig. 10.8). Un pachet provenind de la C către S este modificat de către G, acesta punând adresa proprie G în locul adresei lui C. Serverul S primeşte pachetul ca provenind de la G şi ca urmare răspunde cu un pachet (de acceptarea conexiunii) destinat lui G. Deoarece adresa lui G este publică, pachetul de răspuns ajunge la G. În acel moment, G trebuie să determine faptul că pachetul este răspuns la o cerere adresată de C. Ca urmare,G modifică adresa destinaţie a pachetului, punând adresa lui C în locul propriei adrese,

© 2008, Radu-Lucian Lupşa

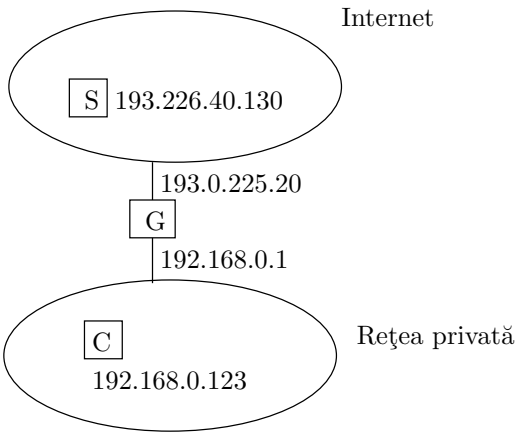

Figura 10.8: Reţea privată pentru exemplificarea mecanismului de translaţie a adresei sursă

după care trimite mai departe pachetul către C — acest lucru este acum posibil deoarece G este în reţeaua internă, şi ca urmare adresa lui C indică singurul nod din reţeaua proprie având acea adresă.

Pentru ca mecanismul de mai sus să fie realizabil, este necesar ca ruterul G să poată determina cărui nod privat îi este destinat în mod real fiecare pachet având ca adresă destinaţie adresa G. De regulă, acest lucru necesită identificarea fiecărui pachet venit din Internet ca răspuns la un pachet dinspre un nod privat spre Internet.

Acest lucru este cel mai simplu de făcut pentru conexiunile TCP. Pentru o conexiune TCP, ruterul G urmăreşte pachetele de deschidere a conexiunii. La primirea unui pachet de deschiderea conexiunii dinspre un nod privat C, de pe un port p_c, către un server S, ruterul G alocă un port TCP local p_g (de preferinţă $p_g = p_c$, însă dacă p_c nu este liber se poate aloca un p_g diferit). Pachetul de deschidere a conexiunii este modificat de G astfel încât adresa sursă să fie G şi portul sursă să fie p_g. G păstrează asocierea (C, p_c, p_g). La sosirea unui pachet cu adresa destinaţie G şi portul destinaţie p_g, ruterul G pune adresa destinaţie C şi portul destinaţie p_c; la primirea unui pachet cu adresa sursă C şi portul sursă p_c ruterul G pune adresa sursă G şi portul sursă p_g. Asocierea (C, p_c, p_g) este păstrată până la închiderea conexiunii, determinată prin schimbul corespunzător de pachete.

Exemplul 10.16: Pentru reţeaua ilustrată în figura 10.8, presupunem că clientul C având adresa (privată) 192.168.0.123 deschide o conexiune TCP de pe portul efemer 3456 către serverul S, având adresa 193.226.40.130, pe

© 2008, Radu-Lucian Lupşa

Sens	Între C şi G		Între G şi S	
	sursă	destinaţie	sursă	destinaţie
$C \to S$	192.168.0.123 port 3456 (p_c)	193.226.40.130 port 80	193.0.225.20 port 7890 (p_g)	193.226.40.130 port 80
$C \leftarrow S$	193.226.40.130 port 80	192.168.0.123 port 3456 (p_c)	193.226.40.130 port 80	193.0.225.20 port 7890 (p_g)

Tabelul 10.8: Adresele sursă şi destinaţie ale pachetelor schimbate între clientul C şi serverul S în exemplul 10.16

portul 80 (pentru a aduce o pagină web). Ruterul G având adresa publică 193.0.225.20 efectuează translaţia adresei sursă. Adresele pachetelor transmise între C şi S sunt date în tabelul 10.8.

Pentru alte protocoale, asocierea este mai dificil de făcut. Pentru UDP, există noţiunea de port şi ca urmare identificarea pachetelor primite de G pe baza portului destinaţie p_g este posibilă. Este însă dificil de determinat durata valabilităţii asocierii (C, p_c, p_g), înturcât nu există pachete de „închiderea conexiunii UDP“. Se poate însă utiliza un timp de expirare, de ordinul câtorva minute, de exemplu, în care dacă nu trec pachete asocierea este desfăcută. Pentru ICMP *ping* şi *pong* există un număr de identificare care poate fi folosit cu acelaşi rol ca şi un număr de port. Translaţia adreselor pentru astfel de pachete funcţionează întocmai ca şi în cazul UDP.

Mecanismul de translaţie a adresei sursă, descris mai sus, permite deschiderea unei conexiuni de la un nod cu adresă privată către un nod public din Internet. Nodul privat „are impresia“ că comunică direct cu serverul din Internet. Serverul din Internet „are impresia“ că comunică cu ruterul G pe portul alocat de acesta.

Faţă de utilizarea adreselor publice, utilizarea adreselor private şi a translaţiei adresei sursă are două limitări majore:

- nu permite deschiderea conexiunilor în sens invers, dinspre Internet către un nod privat;
- dacă pe conexiune sunt trimise, sub forma de date utile pentru protocolul TCP, informaţii privind adresa IP şi portul de pe client, vor fi constatate incoerenţe între IP-ul şi portul clientului văzute de către server (acestea fiind G şi respectiv p_g) şi IP-ul şi portul clientului văzute de client (acestea fiind C şi respectiv p_c).

Cea de-a doua limitare poate fi eliminată dacă G cunoaşte protocolul de nivel aplicaţie dintre C şi S şi modifică datele despre conexiune schimbate

© 2008, Radu-Lucian Lupşa

între C şi S. Prima limitare poate fi eliminată în măsura în care este vorba de conexiuni iniţiate în urma unor negocieri pe o conexiune anterioară (de exemplu, conexiunile de date din protocolul FTP); pentru aceasta, G trebuie, din nou, să urmărească comunicaţia dintre C şi S şi să modifice datele privitoare la adresa şi portul pe care clientul aşteaptă conexiune dinspre server.

10.7.3.2. Translaţia adresei destinaţie

Presupunem că avem o reţea privată, un server S cu adresă privată, un ruter G situat în reţeaua proprie dar având adresă publică şi un client C din Internet. Clientul C doreşte să deschidă o conexiune către serverul S. Desigur, comunicarea „normală“ nu este posibilă deoarece în contextul lui C adresa privată a lui S nu este unică.

Este posibil însă ca adresa publicată (în DNS-ul public) pentru serverul S să fie adresa lui G. În acest caz, C deschide conexiunea către G. Printr-un mecanism similar cu cel din paragraful precedent, G modifică de data aceasta adresa destinaţie, punând, în locul propriei adrese, adresa privată a lui S. Pachetul de răspuns de la S este de asemenea modificat de către G, care pune ca adresă sursă adresa proprie G în loc de S. Astfel, S „are impresia“ că comunică direct cu C, iar C „are impresia“ că comunică cu G.

Translaţia adresei destinaţie poate fi utilă în următoarele situaţii:

- dacă avem o singură adresă publică şi dorim să avem mai multe servere accesibile din exterior, servere ce nu pot funcţiona pe aceeaşi maşină. Putem furniza astfel un server *HTTP* şi un server *SMTP* care apar din Internet ca fiind la aceeaşi adresă IP dar sunt găzduite în realitate pe calculatoare distincte. Necesitatea utilizărea calculatoarelor distincte pentru aceste servicii poate rezulta din motive de securitate sau din motive legate de performanţele calculatoarelor utilizate.
- dacă dorim totuşi acces din afară către calculatoarele din reţeaua internă. De exemplu, pe fiecare calculator rulează un server *SSH*. Fiecărui calculator îi vom asocia un port pe ruterul G. O conexiune din afară, prin protocolul *SSH*, către un anumit port de pe G va fi redirecţionată către serverul *SSH* de pe calculatorul privat corespunzător.
- pentru a distribui cererile către un server foarte solicitat. În acest caz, serverul va avea ca adresă publicată în DNS adresa lui G, însă vor exista de fapt mai multe servere pe calculatoare S_1, S_2, ..., S_n în reţeaua internă. Conexiunile deschise din Internet către G vor fi redirecţionate echilibrat către serverele S_1, S_2, ... S_n. Mai exact, la deschiderea unei conexiuni către G, G alege un server S_k către care redirecţionează acea conexiune. Orice pachet ulterior de pe acea conexiune va fi redirecţionat

către S_k. Conexiuni noi pot fi redirecţionte către alte servere.

10.7.4. Tunelarea

Prin *tunelare* se înţelege, în general, transmiterea unor date aparţinând unui anumit protocol ca date utile în cadrul unui protocol de acelaşi nivel sau de nivel superior.

Ne vom ocupa în cele ce urmează de tunelarea pachetelor IP, adică de transmiterea pachetelor IP prin protocoale de nivel reţea sau de nivel aplicaţie.

O situaţie în care este necesară tunelarea este aceea în care există două reţele private şi se doreşte ca nodurile din cele două reţele să poată comunica nerestricţionat între ele. De exemplu, avem o instituţie care are două sedii şi are o reţea privată in fiecare sediu. Există mai multe soluţii pentru a realiza legătura:

- *Translaţia adreselor* realizează o legătură supusă unor restricţii, studiate în § 10.7.3.
- *Unificarea fizică* a celor două reţele private, ducând o legătură fizică între ele (fig. 10.9), oferă conectivitate completă, însă ducerea unui cablu specual pentru aceasta poate fi extrem de costisitor.

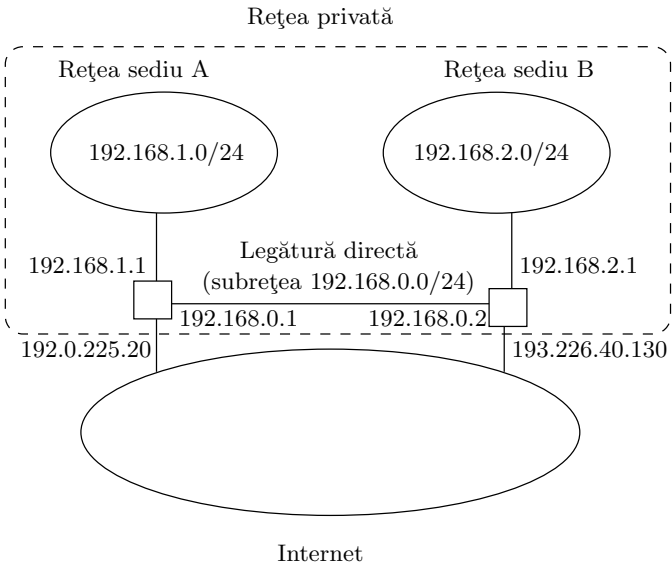

Figura 10.9: Unificarea reţelelor private printr-o legătură fizică directă.

© 2008, Radu-Lucian Lupşa

- *Tunelarea* oferă conectivitate completa ca şi legătura fizică folosind legăturile la Internet existente. Construcţia constă în realizarea unei conexiuni (de exemplu TCP) între două rutere cu adrese publice din cele două reţele interne (fig. 10.10). Conexiunea dintre rutere este un „cablu virtual“ ce preia rolul conexiunii fizice.

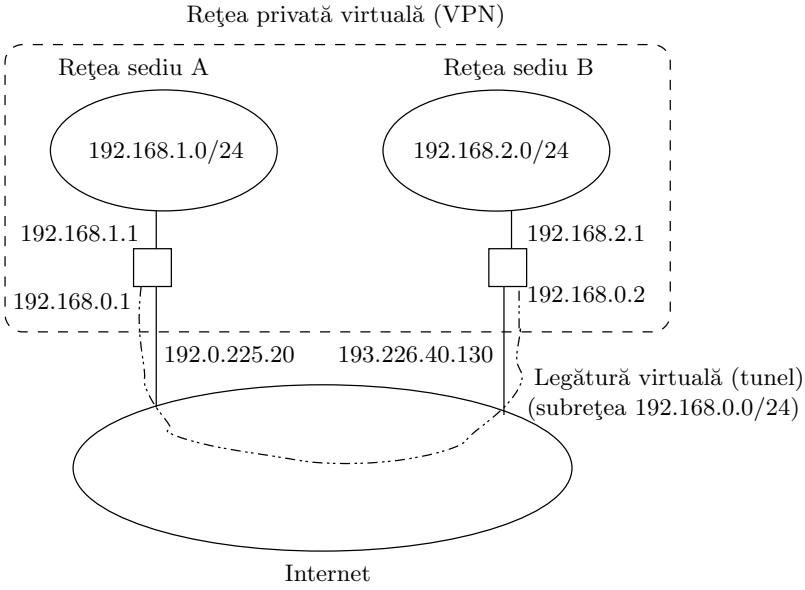

Figura 10.10: Unificarea reţelelor private printr-un tunel

Tunelul se prezintă faţă de nivelul reţea ca şi când ar fi o legătură fizică. Ca urmare, fiecare capăt al tunelului este o interfaţă de reţea, având o adresă IP şi o mască de reţea.

Pentru tunelarea propriu-zisă există mai multe protocoale. Unele dintre protocoale criptează pachetele tunelate; astfel de protocoale oferă securitatea unui cablu direct bine păzit.

Un tunel poate avea mai mult de două capete. Un tunel cu mai multe capete se comportă ca o subreţea cu mai multe interfeţe conectate la ea — de exemplu o reţea Ethernet.