Subiecte Retele

1. Arhitectura retelelor:
    a. Tipuri de retele (cu difuzare, punct la punct)
    b. Retele locale, retele metropolitane, retele larg raspandite geografic (topologie, caracteristici)
    c. Niveluri, protocoale, modelul de referinta OSI (8 niveluri); descriere, caracteristici
2. Nivelul legatura de date
    a. Aspecte (caracteristici) ale proiectarii nivelului legaturii de date
    b. Detectarea si corectarea erorilor (Hamming+CRC)
    c. Protocoale elementare pentru legatura de date (simplex fara restrictii, stop and wait, protocol cu confirmare si retransmitere)
    d. Protocoale cu fereastra glisanta si protocoale cu revenire cu n pasi (Go back n)
3. Nivelul retea
    a. Aspecte (caracteristici, cerinte) ale proiectarii nivelului retea
    b. Algoritmi de dirijare (calea cea mai scurta, inundarea, dirijarea centralizata, dirijarea izolata, dirijarea distribuita, dirijarea ierarhica)
    c. Controlul si evitarea congestionarii si a blocarii
    d. Protocolul IP (descriere)
4. Nivelul Transport
    a. Caracteristici ale nivelului Transport (notiuni de baza: adresarea, stabilirea si eliberarea conexiunii, controlul fluxului, multiplexarea)
    b. Protocolul TCP

* Cu rosu e mai mare probabilitate sa dea la examen

1. ARHITECTURA RETELELOR

1.A. TIPURI DE RETELE

RETELE CU DIFUZARE

Retelele cu difuzare au un singur **canal de comunicatii** care este partajat de toate masinile din retea. Orice masina poate trimite mesaje scurte, numite in anumite context **pachete**, care sunt primate de toate celelalte masini. Un camp de adresa din pachet specifica masina careia ii este adresat pachetul. La receptionarea unui pachet, o masina controleaza campul de adresa. Daca pachetul ii este adresat, masina il prelucreaza. Daca pachetul este trimis pentru o alta masina, pachetul este ignorat.

Sistemele cu difuzare permit in general si adresarea unui pachet catre toate destinatiile, prin folosirea unui cod special in campul de adresa. Un pachet transmis cu acest cod este primit si prelucrat de toate masinile din retea. Acest mod de operare se numeste **difuzare**. Unele sisteme cu difuzare suporta de asemenea transmisia la un subset de masini, operatie cunoscuta sub numele de trimitere multipla.

RETELE PUNCT-LA-PUNCT

Retelele punct-la-punct dispun de numeroase conexiuni intre perechi de masini individuale. Pentru a ajunge de la sursa la destinatie pe o retea de acest tip, un pachet s-ar putea sa fie nevoit sa treaca prin una sau mai multe masini intermediare. Deseori sunt posibile trasee multiple, de diferite lungimi, si de aceea descoperirea drumurilor celor mai potrivite este foarte importanta. Transmisiile punct la punct cu un singur transmitator si un singur receptor sunt numite uneori si **unicasting**.

IN GENERAL

Ca o regula generala, desi exista numeroase exceptii, retelele mai mici, localizate geographic, tind sa utilizeze difuzarea, in timp ce retelele mai mari sunt de obicei punct-la-punct.

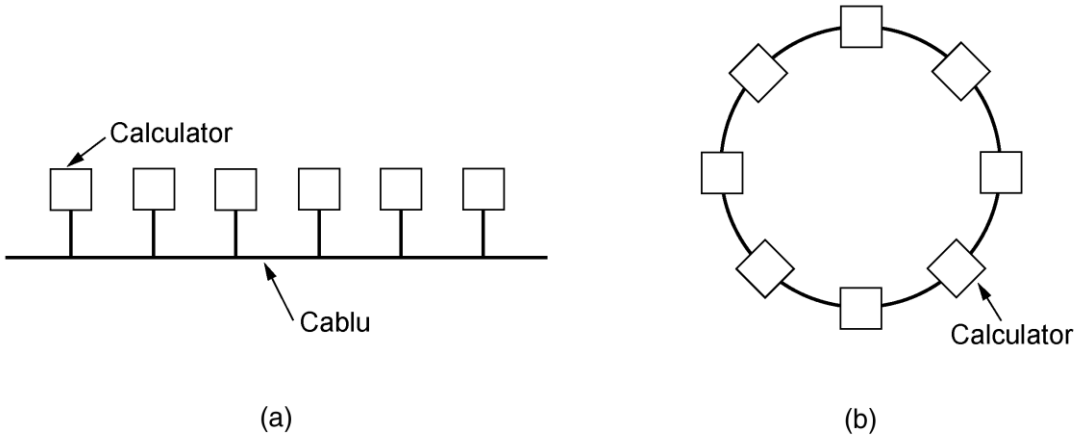
1.B. CLASIFICAREA RETELELOR

RETELE LOCALE

Retelele locale (Local Area Networks), denumite in general **LAN**-uri, sunt retele private localizate intr-o singura cladire sau intr-un campus de cel mult cativa kilometri. Ele sunt frecvent utilizate pentru a conecta calculatoarele personale si statiile de lucru din birourile companiilor si fabricilor, in scopul de a partaja resurse si de a schimba informatii. **LAN**-urile se disting de alte tipuri de retele prin trei caracteristici: marime, tehnologie de transmisie si topologie.

LAN-urile au dimensiuni restranse, ceea ce inseamna ca timpul de transmisie in cazul cel mai favorabil este limitat si cunoscut dinainte. Cunoscut aceasta limita, este posibil sa utilizam anumite tehnici de proiectare care altfel nu ar fi fost posibile.

LAN-urile utilizeaza frecvent o tehnologie de transmisie care consta dintr-un singur cablu la care sunt atasate toate masinile, asa cum erau odata cablurile telefonice commune in zonele rurale. **LAN**-urile traditionale functioneaza la viteze cuprinse intre 10 si 100 Mbps, au intarzieri mici si produc erori foarte putine. **LAN**-urile mai noi pot opera si la viteze mai mari, pana la 10 Gbps.

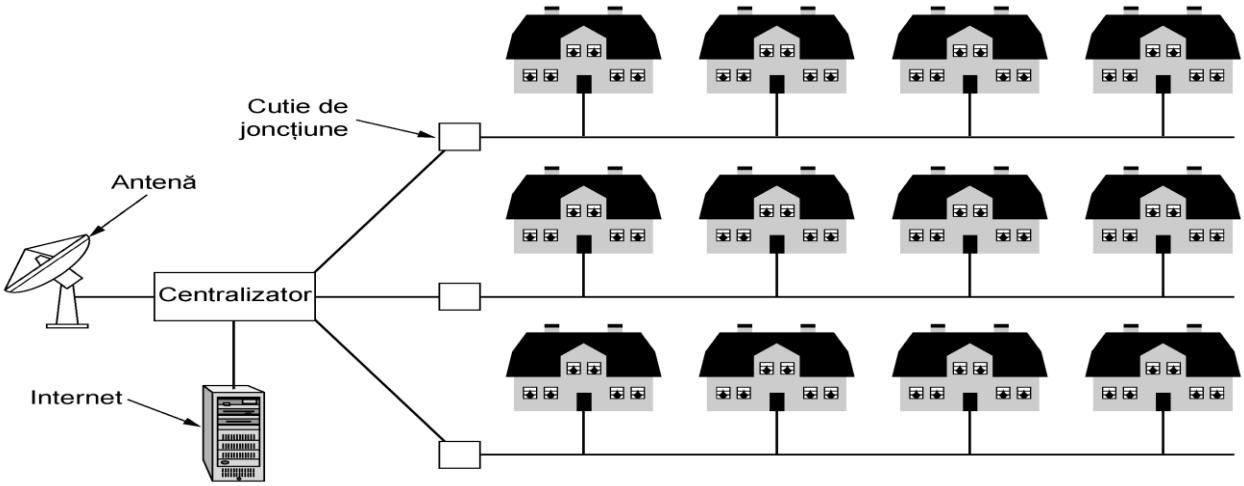


Pentru **LAN**-urile cu difuzare sunt posibile diverse topologii. Figura de mai sus prezinta doua dintre ele. Intr-o retea cu magistrala (cu cablu linear), in fiecare moment cel mult una dintre masini este master si are dreptul sa transmita, restul masinilor nu pot. Cand 2 sau mai multe masini vor sa transmita simultan, este necesar un mecanism de arbitrare.

Un al doilea tip de retea cu difuzare este reteaua in inel. Intr-un inel fiecare bit se propaga independent de ceilalti, fara sa astepte restul pachetului caruia ii apartine. In mod tipic, fiecare bit navigheaza pe circumferinta intregului inel intr-un interval de timp in care se transmit doar cativa biti, de multe ori inainte chiar ca intregul pachet sa fi fost transmis. Ca in orice alt sistem cu difuzare, este nevoie de o regula pentru a arbitra accesele simultane la inel.

RETELE METROPOLITANE

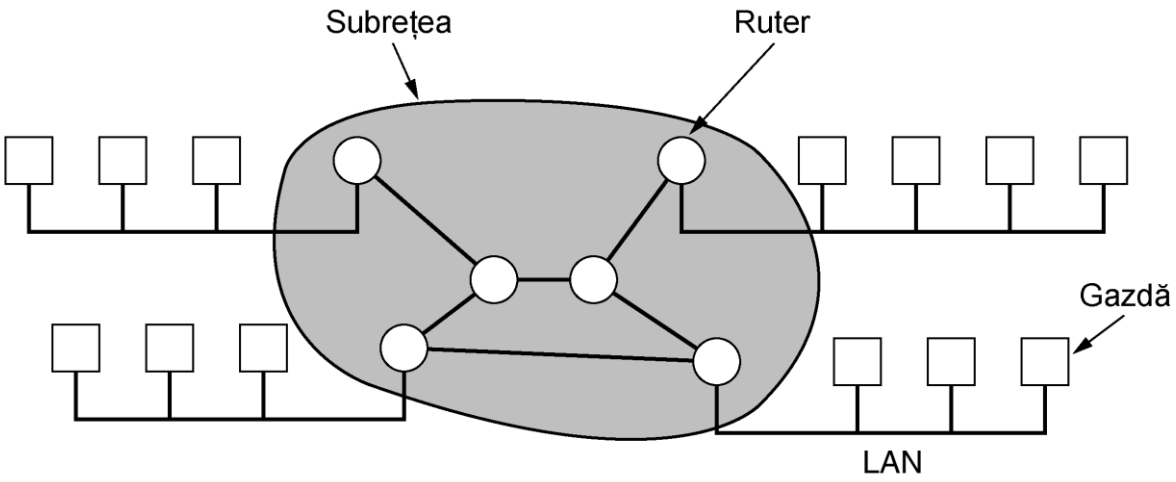
O **retea metropolitana (Metropolitan Area Network)**, sau **MAN**, deserveste un oras. Cel mai bun exemplu de **MAN** este reteaua de televiziune prin cablu disponibila in cele mai multe orase. Acest sistem s-a dezvoltat de la primele antene colective folosite in zone in care semnalul receptionat prin aer era foarte slab. In aceste sisteme timpurii, o atenta foarte mare era amplasata pe varful celui mai apropiat deal si semnalul captat era retransmis catre casele abonatilor.



RETELE LARG RASPUNDITE GEOGRAFIC

O **retea larg raspandita geografic (Wide Area Network)**, sau **WAN**, acopera o arie geografica intinsa. Reteaua contine o colectie de masini utilizate pentru a executa programele utilizatorilor. Aceste masini se numesc **gazde**. Gazdele sunt conectate printr-o **subretea de comunicatie** sau, pe scurt, **subretea**. Gazdele apartin clientilor (ex. PC-urile oamenilor), desi subreteaua apartine si este exploatata de o companie de telefonie sau de un furnizor de servicii Internet. Sarcina subretelei este sa transporte mesajele de la gazda la gazda.

In majoritatea retelelor larg raspandite geografic, subreteaua este formata din doua componente distincte: **liniile de transmisie** si **elementele de comutare**. Liniile de transmisie transporta bitii intre masini. Elementele de comutare sunt calculatoare specializate, folosite pentru a conecta doua sau mai multe linii de transmisie. Cand sosesc date pe o anumita linie, elementul de comutare trebuie sa aleaga o noua linie pentru a retransmite datele mai departe.



In aceasta figura, fiecare cazda este de cele mai multe ori conectata la un LAN in care exista un router, desi in anumite cazuri o gazda poate fi legata direct cu un router. Colectia de linii de comunicatie si de reoutere (dar nu si gazdele) formeaza **subreteaua**.

In cazul celor mai multe WAN-uri, reteaua contine numeroase linii de transmisie, fiecare din ele legand o pereche de routere. Daca 2 routere nu impart un acelasi cablu, dar doresc sa comunice, atunci ele trebuie sa faca acest lucru indirect, prin intermediul altor routere. Cand un pachet este transmis de la un router la altul prin intermediul unuia sau mai multor routere, pachetul este primit in intregime de fiecare router intermediar, este retinut acolo pana cand linia de iesire ceruta devine libera si apoi este retransmis. O subretea care functioneaza astfel se numeste subretea **memoreaza-si-retransmite** sau subretea **cu comutare de pachete**. Aproape toate WAN-urile au astfel de subretele.

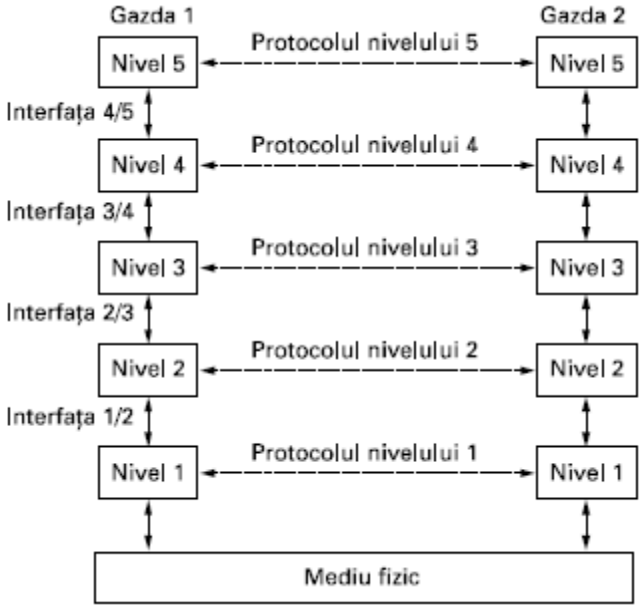
Atunci cand un proces al unei gazde are un mesaj de transmis catre un proces de pe o alta gazda, gazda care transmite va sparge mesajul in pachete, fiecare dintre ele retinandu-si numarul de ordine din secventa. Aceste pachete sunt apoi transmise in retea unul cate unul intr-o succesiune rapida. Pachetele sunt transportate individual prin retea si depozitate la gazda receptoare, unde sunt reasamblate in mesajul initial si furnizate procesului receptor.

1.C. NIVELURI, PROTOCOALE, MODELUL DE REFERINTA OSI

IERARHIILE DE PROTOCOALE

Pentru a reduce din complexitatea proiectarii, majoritatea retelelor sunt organizate sub forma unei serii de **straturi** sau **niveluri**, fiecare din ele construit peste cel de dedesubt. In toate retelele, scopul fiecarui nivel este sa ofere anumite servicii nivelurilor superioare, protejandu-le totodata de detaliile privitoare la implementarea efectiva a serviciilor oferite. Intr-un anumit sens, fiecare nivel este o masina virtuala, oferind anumite servicii nivelului de deasupra lui.

Nivelul n de pe o masina converseaza cu nivelul n de pe alta masina. Regulile si conventiile utilizate sunt cunoscute sub numele de **protocolul** nivelului n. In principal, un **protocol** reprizinta o intelegere intre partile care comunica, asupra modului de realizare a comunicarii. Incalcarea protocolului va face comunicarea mai dificila, daca nu chiar imposibila.



In imaginea de mai sus este ilustrata o retea cu 5 niveluri. Entitatile din niveluri corespondente de pe masini diferite se numesc **egale**. Entitatile egale pot fi procese, dispozitive hardware sau chiar fiinte umane. Cu alte cuvinte, entitatile egale sunt cele care comunica folosind protocolul.

In realitate nici un fel de date nu sunt transferate direct de pe nivelul n al unei masini pe nivelul n al altei masini. Fiecare nivel transfera datele si informatiile de control nivelului imediat inferior, pana cand se ajunge la nivelul cel mai de jos. Sub nivelul 1 se afla **mediul fizic** prin care se produce comunicarea efectiva. In figura de mai sus, comunicarea virtuala este reprezentata prin linii punctate, iar comunicarea fizica prin linii continue. Intre doua niveluri adiacente exista o **interfata**. Interfata defineste ce operatii si servicii primitive ofera nivelul de jos catre nivelul de sus.

O multime de niveluri si protocoale este numita **arhitectura de retea**. O lista de protocoale utilizate de catre un anumit sistem, cate un protocol pentru fiecare nivel, se numeste **stiva de protocoale**.

MODELUL DE REFERINTA OSI

O arhitectura de retea importanta este **modelul de referinta OSI**. Desi protocoalele asociate cu modelul OSI nu sunt folosite aproape deloc, modelul in sine este destul de general si inca valabil.

	Serviciu	Exemplu
Orientate pe conexiuni	Flux de mesaje sigur	Secvență de pagini
	Flux de octeți sigur	Conectare la distanță
	Conexiune nesigură	Voce digitizată
Fără conexiuni	Datagramă nesigură	Publicitate prin e-mail
	Datagramă confirmată	Scrisori cu confirmare
	Cerere-răspuns	Interogări baze de date

Modelul OSI este prezentat in figura de mai sus (mai putin mediul fizic). **Modelul OSI** curpinde 7 niveluri. Principiile aplicate pentru a se ajunge la cele 7 niveluri sunt urmatoarele:

1. Un nivel trebuie creat atunci cand este nevoie de un nivel de abstractizare diferit.
2. Fiecare nivel trebuie sa indeplineasca un rol bine definit.
3. Functia fiecarui nivel trebuie aleasa acordandu-se atentie definirii de protocoale standardizate pe plan international.
4. Delimitarea nivelurilor trebuie facuta astfel incat sa se minimizeze fluxul de informatii prin interfete.
5. Numarul de niveluri trebuie sa fie suficient de mare pentru a nu fi nevoie sa se introduca in acelasi nivel functii diferite si suficient de mic pentru ca arhitectura sa ramana functionala.

Nivelurile **modelului de referinta OSI** sunt urmatoarele:

1. **Nivelul fizic** – se ocupa de transmiterea bitilor printr-un canal de comunicatie.
2. **Nivelul legatura de date** – sarcina sa principala este de a transforma un mijloc oarecare de transmisie intr-o linie care sa fie disponibila nivelului retea fara erori de transmisie nedetectate.
3. **Nivelul retea** – se ocupa de controlul functionarii subretelei.
4. **Nivelul transport** – rolul sau principal este sa accepte date de la nivelul sesiune, sa le descompuna, daca este cazul, in unitati mai mici, sa transfere aceste unitati nivelului retea si sa se asigure ca toate fragmentele sosesc corect la celalalt capat.
5. **Nivelul sesiune** – permite utilizatorilor de pe masini diferite sa stabileasca intre ei sesiuni.
6. **Nivelul prezentare** – se ocupa de sintaxa si semantica informatiilor transmise.
7. **Nivelul aplicatie** – contine o varietate de protocoale frecvent utilizate. Un exempu de protocol utilizat pe scara larga este HTTP, care sta la baza WWW.

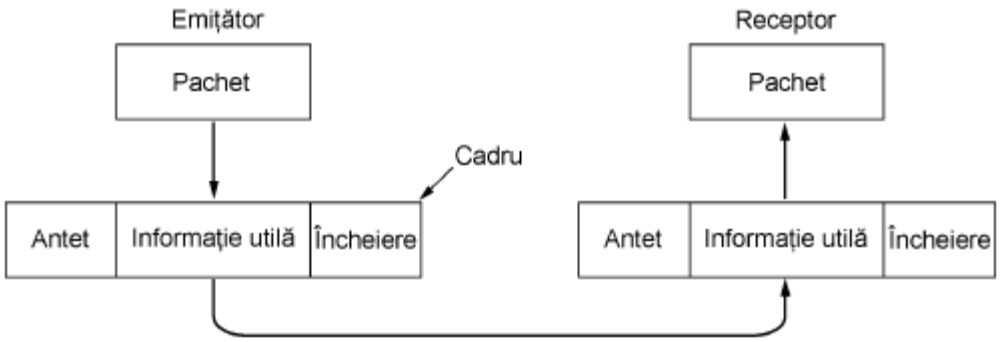
2. NIVELUL LEGATURA DE DATE

2.A. CARACTERISTICI ALE PROIECTARII NIVELULUI LEGATURII DE DATE

Nivelul legatura de date are un numar de functii specifice pe care trebuie sa le indeplineasca. Aceste functii includ:

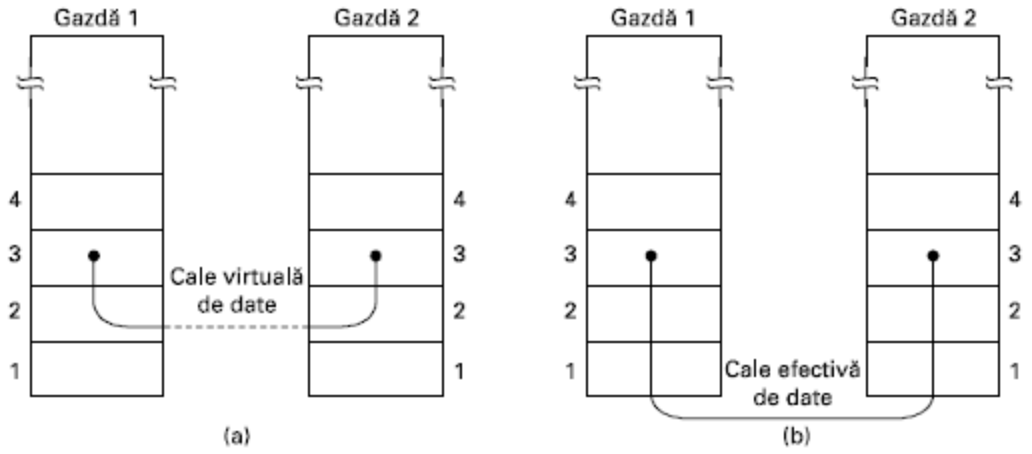
1. Furnizarea unei interfete bine-definite catre nivelul retea
2. Tratarea erorilor de transmisie
3. Reglarea fluxului cadrelor in asa fel, incat receptorii lenti sa nu fie inundati de catre emitatori rapizi

Pentru a indeplini aceste scopuri, nivelul legatura de date primeste pachete de la nivelul retea, pe care le incapsuleaza in **cadre** in vederea transmiterii. Fiecare **cadru** contine un antet, un camp de informatie utila pentru pachet si incheiere, dupa cum se vede in figura de mai jos. Gestionarea cadrelor reprezinta esenta a ceea ce face nivelul legatura de date.



SERVICII OFERITE NIVELULUI RETEA

Functia nivelului legatura de date este sa ofere servicii nivelului retea. Principalul serviciu este transferul datelor de la nivelul retea al masinii sursa la nivelul retea al masinii destinatie. La nivelul retea al masinii sursa exista o entitate, sa-i spunem proces, care trimite biti catre nivelul legatura de date, pentru a fi transmisi la destinatie. Funcia nivelului legatura de date este sa transmita bitii spre masina destinatie, pentru ca acolo sa fie livrati nivelului retea, asa cum se arata in figura (a) de mai jos. Transmisia efectiva urmeaza calea din figura (b).



Nivelul legatura de date poate fi proiectat sa ofere diferite servicii. Serviciile oferite pot varia de la sistem la sistem. Trei posibilitati de baza, oferite in mod curent, sunt:

1. Serviciu neconfirmat fara conexiune
2. Serviciu confirmat fara conexiune
3. Serviciu confirmat orientat-conexiune

INCADRAREA

In vederea furnizarii unui serviciu nivelului retea, nivelul legatura de date trebuie sa utilizeze serviciul furnizat de catre nivelul fizic. Sarcina nivelului fizic este sa primeasca un flux de biti si sa incerce sa-l trimita la destinatie. Nu se garanteaza ca acest flux de biti nu contine erori. Numarul de biti receptionati poate fi mai mic, egal cu, sau mai mare decat numarul de biti transmisi si pot avea valori diferite. Este la latitudinea nivelului legatura de date sa detecteze si, daca este necesar, sa corecteze erorile.

2.B. DETECTAREA SI CORECTAREA ERORILOR

Exista 2 strategii de baza pentru tratarea erorilor. O modalitate este ca pe langa fiecare bloc de date trimis sa se includa suficienta informatie redundanta astfel incat receptorul sa poata deduce care a fost caracterul trimis. O alta solutie este sa se includa suficienta redundanta pentru a permite receptorului sa constate ca a aparut o eroare, dar nu care este eroarea, si sa ceara o retransmisie. Prima strategie utilizeaza **coduri corectoare de erori**, iar cea de-a doua utilizeaza **coduri detectoare de erori**.

CODURI CORECTOARE DE ERORI

In cazul canalelor de comunicatie fara fir, este indicat sa adaugam destula informatie redundanta fiecarui bloc, in loc sa ne bazam pe retransmisie, care poate sa fie la randul sau afectata de erori.

In mod normal, un cadru contine *m* biti de date si *r* biti redundanti sau de control. Sa consideram lungimea totala *n* (adica *m+r*). O unitate formata din n biti, care contine date si biti de control, este numita frecvent **cuvand de cod** de n biti.

Date fiind doua cuvinte de cod, este posibil sa determinam cati biti corespunzatori difera. Pentru a determina cati biti difera, aplicam operatorul SAU EXCLUSIV intre cele doua cuvinte de cod si numaram bitii 1 din rezultat.

Exemplu:
10001001
10110001
00111000

Numarul de pozitii binare in care doua cuvinte de cod difera se numeste **distanta Hamming**. Daca doua cuvinte de cod sunt despartite de o distanta Hamming *d*, sunt necesare *d* erori de un singur bit pentru a-l converti pe unul in celalalt.

Proprietatile detectoare si corectoare de erori ale unui cod depinde de distanta sa Hamming. Pentru a **detecta** *d* erori, este nevoie de un cod cu distanta *d+1*. Atunci cand receptorul vede un cuvant de cod incorect, poate spune ca s-a produs o eroare de transmisie. Pentru a **corecta** *d* erori, este nevoie de un cod cu distanta *2d+1*.

METODA HAMMING

Bitii cuvantului de cod sunt numerotati consecutiv, incepand cu bitul 1 de la marginea din stanga. Bitii care sunt puteri ale lui 2 (1,2,4,8,16,etc) sunt biti de control. Restul sunt completati cu cei m biti de date. Fiecare bit de control forteaza ca paritatea unui grup de biti, inclusiv el insusi, sa fie para (sau impara).

Un bit poate fi inclus in mai multe calcule de paritate. Pentru a vedea la care biti de control contribuie bitul de date din pozitia k, rescriem k ca o suma de puteri ale lui 2. De exemplu, 11=1+2+8 si 29=1+4+8+16. Un bit este verificat de acei biti de control care apar in dezvoltarea sa.

Cand soseste un cuvand de cod, receptorul initializeaza un contor la 0, Acesta examineaza apoi fiecare bit *k* (*k*=1,2,4,8,...) de control pentru a vedea daca are paritatea corecta. Daca nu, adauga *k* la contor. Daca, dupa ce au fost examinati toti bitii de control, contorul este 0, cuvantul de cod este acceptat ca valid. Daca valoarea este nenula, ea reprezinta numarul bitului incorect. De exemplu, daca bitii de control 1,2 si 8 sunt eronati, atunci bitul inversat este 11, deoarece este singurul verificat de bitii 1,2 si 8.

Codurile Hamming pot corecta numai erori singulare, insa exista un artificiu care poate fi utilizat pentru a permite codurilor Hamming sa corecteze erorile in rafala.

CODURI DETECTOARE DE ERORI

Uneori este mai eficient sa utilizam un cod detector de erori si sa retransmitem blocul in care s-au detectat erori. De exemplu pe canalele cu siguranta mare, cum ar fi fibra optica.

CODUL POLINOMIAL (CYCLIC REDUNDANCY CODE "**CRC**")

Codurile polinomiale sunt bazate pe tratarea şirurilor de biţi ca reprezentări de polinoame cu coeficienţi 0 şi 1. Un cadru de k biţi este văzut ca o listă de coeficienţi pentru un polinom cu k termeni, de la x^{k-1} la x^0 . Se spune că un astfel de polinom este de gradul k-1. Bitul cel mai semnificativ (cel mai din stânga) este coeficientul lui x^{k-1} ; următorul bit este coeficientul lui x^{k-2} ş.a.m.d. De exemplu, 110001 are şase biţi şi ei reprezintă un polinom cu şase termeni cu coeficienţii 1, 1, 0, 0, 0 şi 1: $x^5 + x^4 + x^0$. Aritmetica polinomială este de tip modulo 2, în conformitate cu regulile teoriei algebrice. Nu există transport la adunare şi nici împrumut la scădere. Atât adunările cât şi scăderile sunt identice cu SAU EXCLUSIV.

Atunci când este utilizată metoda codului polinomial, emiţătorul şi receptorul se pun de acord în avans asupra unui polinom generator G(x). Atât bitul cel mai semnificativ cât şi cel mai puţin semnificativ trebuie să fie 1. Pentru a calcula suma de control pentru un cadru cu m biţi, corespunzător polinomului M(x), cadrul trebuie să fie mai lung decât polinomul generator. Ideea este de a adăuga o sumă de control la sfârşitul cadrului, astfel încât polinomul reprezentat de cadrul cu sumă de control să fie divizibil prin G(x). Când receptorul preia cadrul cu suma de control, încearcă să-l împartă la G(x). Dacă se obţine un rest, înseamnă că a avut loc o eroare de transmisie.

2.C. PROTOCOALE ELEMENTARE PENTRU LEGATURA DE DATE

Un cadru de date este compus din patru câmpuri: kind, seq, ack şi info, dintre care primele trei conţin informaţii de control, iar ultimul poate conţine datele efective care trebuie transferate. Ansamblul acestor câmpuri de control este numit antetul cadrului **(frame header)**.

Câmpul kind (tip) spune dacă există sau nu date în cadru, deoarece unele protocoale fac distincţie între cadrele care conţin exclusiv informaţii de control şi cele care conţin şi date. Câmpurile seq şi ack sunt utilizate pentru numere de secvenţă şi, respectiv, confirmări (acknowledgements); utilizarea lor va fi descrisă în detaliu mai târziu. Câmpul info al unui cadru de date conţine un singur pachet de date; câmpul info al unui cadru de control nu este utilizat.

UN PROTOCOL SIMPLEX FĂRĂ RESTRICŢII

Într-un protocol foarte simplu, datele sunt transmise într-o singură direcţie. Cele două niveluri reţea, de transmisie şi de recepţie, sunt considerate tot timpul pre-gătite. Timpul de prelucrare poate fi ignorat. Memoria de stocare disponibilă este infinită. Şi, cel mai bun lucru dintre toate, canalul de comunicaţie între niveluri legătură de date nu pierde şi nu alterează niciodată cadrele. Acest protocol poate fi considerat total nerealist.

Protocolul constă din două proceduri distincte, una de emisie şi cealaltă de recepţie. Emiţătorul lucrează la nivelul legătură de date al maşinii sursă, iar receptorul la nivelul legătură de date al maşinii de destinaţie.

Emiţătorul este într-un ciclu infinit care doar inserează datele pe linie cât poate de repede. Ciclul constă din trei acţiuni: preluarea unui pachet de date de la nivelul reţea (care este întotdeauna

serviabil), construirea unui cadru de ieşire şi trimiterea cadrului pe drumul său. Acest protocol utilizează numai câmpul info al cadrului, deoarece celelalte câmpuri se referă la erori şi secvenţe de control, iar în acest caz nu există erori sau restricţii de control.

Receptorul este la fel de simplu. Iniţial el aşteaptă să se întâmple ceva, singura posibilitate fiind sosirea unui cadru nealterat. În cele din urmă, cadrul ajunge, iar procedura de tip wait_for_event se întoarce cu event setat la frame_arrival (care este oricum ignorat). Apelul from_physical_layer mută cadrul nou sosit din zona tampon a echipamentului într-o anumită variabilă. În cele din urmă pachetul de date este trimis nivelului reţea şi nivelul legătură de date revine la starea de aşteptare a cadrului următor, autosuspendându-se pur şi simplu până la sosirea unui nou cadru.

UN PROTOCOL SIMPLU STOP-AND-WAIT (PAS-CU-PAS)

În acest caz se renunţă la cea mai nerealistă restricţie utilizată în protocolul anterior (protocolul 1): posibilitatea ca nivelul reţea receptor să prelucreze datele de intrare cu viteză infinită (sau echivalent, prezenţa în nivelul legăturii de date receptor a unui tampon infinit în care să fie memorate, cât timp îşi aşteaptă rândul, toate cadrele sosite). Totuşi, se presupune în continuare că nu se produc erori pe canalul de comunicaţie şi că traficul de date este încă simplex.

Principala problemă care trebuie rezolvată aici este cum să se evite ca emiţătorul să inunde receptorul cu date care sosesc mai rapid decât poate acesta să prelucreze. În esenţă, dacă receptorul are nevoie de un timp Δt ca să execute from_physical_layer şi to_network_layer, atunci emiţătorul trebuie să transmită la o viteză medie mai mică de un cadru la fiecare interval de timp de Δt. Mai mult, dacă se presupune că echipamentul receptor nu realizează automat memorarea în zona tampon şi gestiunea cozii de aşteptare, atunci emiţătorul nu trebuie să transmită niciodată un nou cadru până când cel vechi nu a fost preluat de rutina from_physical_layer, ca nu cumva cel nou să se scrie peste cel vechi.

O soluţie mult mai generală a acestei dileme este ca receptorul să furnizeze o reacţie către emiţător. După trimiterea unui pachet către nivelul său reţea, receptorul trimite un mic cadru fictiv către emiţător care, de fapt, îi dă emiţătorului permisiunea să transmită următorul cadru. După ce a transmis un cadru, emiţătorul este obligat de protocol să intre în aşteptare un timp, până când soseşte micul cadru fictiv (deci confirmarea). Utilizarea reacţiei de la receptor pentru a anunţa emiţătorul că poate trimite date este un exemplu de control al fluxului menţionat anterior.

2.D.1. PROTOCOALE CU FEREASTRĂ GLISANTĂ

În protocoalele anterioare, cadrele cu date erau transmise într-o singură direcţie. În cele mai multe situaţii practice, este necesar să se transmită date în ambele direcţii. În acest model, cadrele cu date de la A la B sunt amestecate cu cadrele de confirmare de la A la B. Uitându-se la câmpul kind din antetul cadrului ce a sosit, receptorul poate spune dacă este vorba de un cadru de date sau de confirmare.

Cu toate că întrepătrunderea cadrelor de date şi control pe acelaşi circuit constituie o îmbunătă- ţire faţă de cazul utilizării a două circuite fizice separate, mai este posibilă încă o îmbunătăţire. Atunci când soseşte un cadru cu date, în locul emiterii imediate a unui cadru de control separat, receptorul stă şi aşteaptă până când nivelul reţea îi dă următorul pachet. Confirmarea este ataşată cadrului cu date de ieşire (utilizând câmpul ack din antetul cadrului). De fapt confirmarea este transportată pe gratis de către următorul cadru cu date de ieşire. Tehnica întârzierii confirmării, astfel încât să poată fi agăţată de următorul cadru de date, este cunoscută ca **ataşare (piggybacking)**.

Următoarele protocoale sunt protocoale bidirecţionale care aparţin unei clase de protocoale numite protocoale cu **fereastră glisantă (sliding window)**.

Esenţa protocoalelor cu fereastră glisantă este aceea că, la orice moment de timp, emiţătorul menţine o mulţime de numere de secvenţă care corespund cadrelor pe care are permisiunea să le trimită. Se spune că aceste cadre aparţin **ferestrei de transmisie (sending window)**. Similar, receptorul menţine de asemenea o **fereastră de recepţie (receiving window)**, ce corespunde mulţimii de cadre care pot fi acceptate. Fereastra emiţătorului şi fereastra receptorului nu trebuie să aibă aceleaşi limite minime şi maxime şi nici măcar aceeaşi dimensiune. În unele protocoale ele au dimensiune fixă, dar în altele ele pot creşte sau scădea pe măsură ce cadrele sunt emise sau recepţionate.

Numerele de secvenţă din cadrul ferestrei emiţătorului reprezintă cadre transmise sau cadre ce pot fi transmise, dar încă neconfirmate. De fiecare dată când de la nivelul reţea soseşte un nou pachet, acestuia îi este atribuit următorul număr de secvenţă, iar marginea superioară a ferestrei este avansată cu unu. Atunci când soseşte o confirmare, creşte cu unu limita inferioară a ferestrei. În acest mod, fereastra menţine continuu o listă de cadre neconfirmate. Un exemplu este prezentat în figura de mai jos.

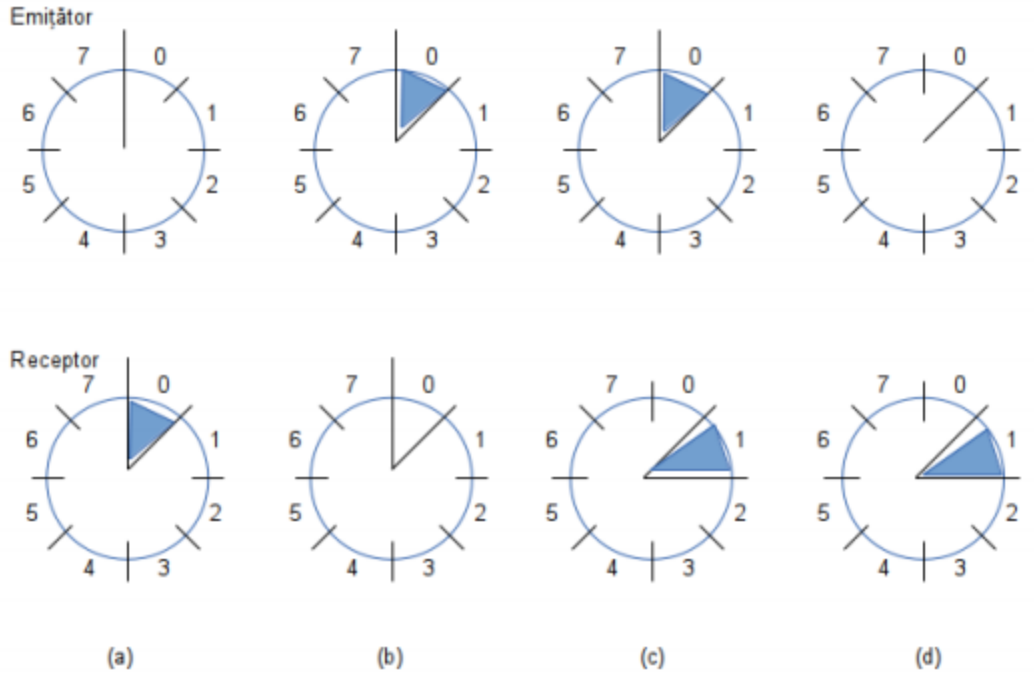


Figura 4-9 – O fereastră glisantă de dimensiune 1, cu număr de secvenţă de 3 biţi.

(a) Iniţial. (b) După ce a fost transmis primul cadru. (c) După ce a fost recepţionat primul cadru.

(d) După ce a fost recepţionată prima confirmare.

2.D.2. PROTOCOALE DE REVENIRE CU N PAŞI (GO BACK N)

Utilizarea benzii de asamblare în cazul unui canal de comunicaţie nesigur ridică probleme serioase. Mai întâi să vedem ce se întâmplă dacă un cadru din mijlocul unui şir lung este modificat sau pierdut. Multe cadre succesive vor ajunge la receptor înainte ca emiţătorul să observe că ceva este greşit. Atunci când un cadru modificat ajunge la receptor este evident că el trebuie eliminat, dar ce trebuie să facă receptorul cu toate cadrele corecte care urmează? Nivelul legătură de date receptor este obligat să livreze pachete către nivelul reţea în secvenţă. În Figura 4-10, se prezintă efectele utilizării benzii de asamblare asupra revenirii în caz de eroare.

Există două moduri de bază de tratare a erorilor în prezenţa benzii de asamblare. Un mod, numit **revenire cu n paşi (go back n)**, este ca receptorul să elimine pur şi simplu cadrele care urmează, netrimiţând confirmări pentru cadrele eliminate. Această strategie corespunde unei ferestre de recepţie de dimensiune 1. Cu alte cuvinte, nivelul legătură de date refuză să accepte orice cadru exceptându-l pe următorul care trebuie livrat către nivelul reţea. Dacă fereastra emiţătorului se umple înaintea expirării contorului de timp, banda de asamblare va începe să se golească. În cele din urmă, timpul emiţătorului va expira şi se vor retransmite toate cadrele neconfirmate, în ordine, începând cu

cadrul pierdut sau modificat. Dacă rata erorilor este mare, această abordare poate risipi o mare parte din lărgimea de bandă.

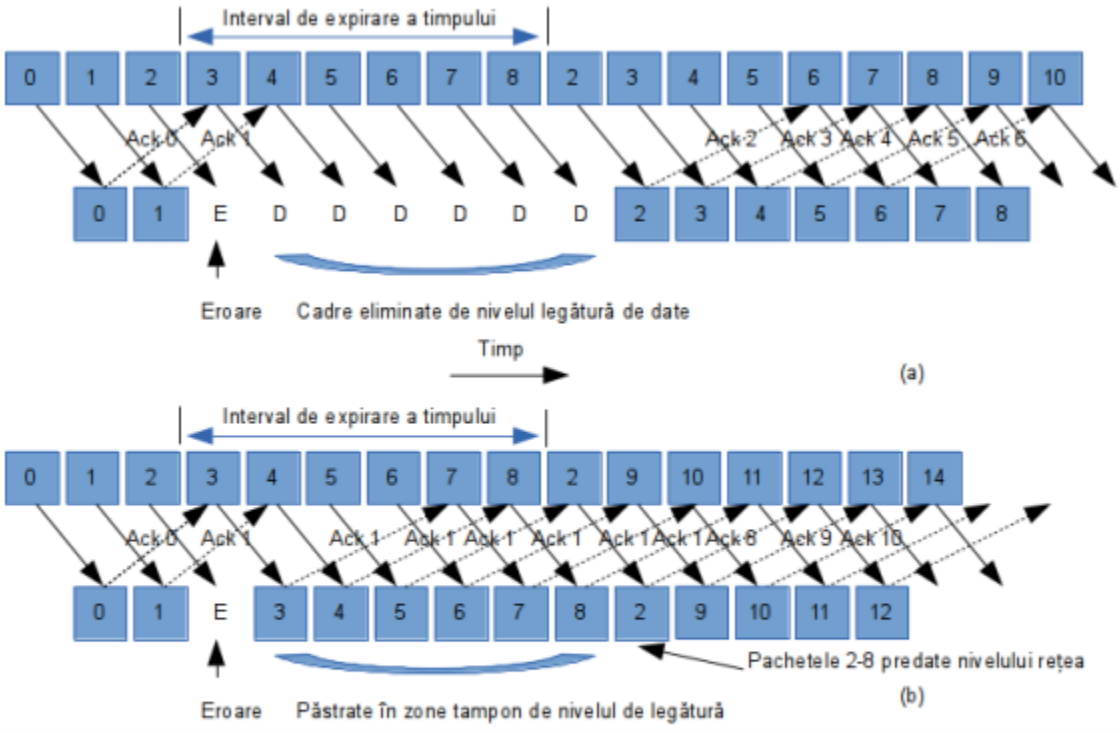


Figura 4-10 – Folosirea benzii de asamblare şi revenirea din eroare. Efectul unei erori când (a) dimensiunea ferestrei receptoare este 1şi (b) dimensiunea ferestrei receptorului este mare.

În Figura 4-10 (a) este prezentat protocolul de revenire cu n paşi pentru cazul în care fereastra receptorului are dimensiune unu. Cadrele 0 şi 1 sunt primite şi confirmate corect. Cadrul 2, totuşi, este alterat sau pierdut. Emiţătorul, care nu ştie de această problemă, continuă să trimită cadre până când timpul pentru cadrul 2 expiră. Apoi se întoarce la cadrul 2 şi o ia de la început cu el, trimiţând din nou cadrele 2, 3, 4 etc.

3. NIVELUL RETEA

3.A. CERINŢELE DE PROIECTARE ALE NIVELULUI REŢEA

COMUTARE DE PACHETE DE TIP MEMOREAZĂ-ŞI-RETRANSMITE (STORE-AND-FORWARD)

Componentele majore ale sistemului sunt echipamentul companiei de telecomunicaţii (rutere conectate prin linii de transmisie), prezentat în interiorul ovalului umbrit, şi echipamentul clientului, prezentat în afara ovalului. Gazda H1 este conectată direct la unul dintre ruterele companiei de telecomunicaţii, A, printr-o linie închiriată. În contrast, H2 este într-o reţea LAN cu un ruter, F, deţinut şi operat de către client. Acest ruter are, deasemeni, şi o linie închiriată către echipamentul companiei de telecomunicaţii.

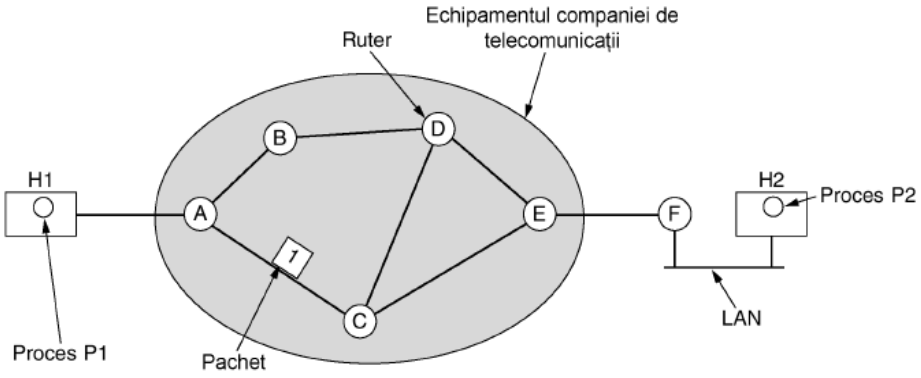


Fig. 5-1. Cadrul protocoalelor nivelului reţea.

O gazdă care are de transmis un pachet îl transmite celui mai apropiat ruter, fie in aceeaşi reţea LAN, fie printr-o legătură punct la punct cu compania de telecomunicaţii. Pachetul este memorat acolo până ajunge integral, astfel încât să poată fi verificată suma de control. Apoi este trimis mai departe către următorul ruter de pe traseu, până ajunge la gazda destinaţie, unde este livrat. Acest mecanism reprezint comutarea de pachete de tip memorează-şi-retransmit.

SERVICII FURNIZATE NIVELULUI TRANSPORT

Serviciile nivelului reţea au fost proiectate având în vedere următoarele scopuri:

1. Serviciile trebuie să fie independente de tehnologia ruterului.
2. Nivelul transport trebuie să fie independent de numărul, tipul şi topologia ruterelor existente
3. Adresele de reţea disponibile la nivelul transport trebuie să folosească o schemă de numerotare uniformă, chiar în cadrul reţelelor LAN şi WAN.

Problema centrală a discuţiei este dacă nivelul reţea trebuie să furnizeze servicii orientate pe conexiune sau servicii neorientate pe conexiune.

O tabără (reprezentată de comunitatea Internet) afirmă că scopul ruterului este de a transfera pachete şi nimic mai mult deoarece subreţeaua este inerent nesigură, indiferent cum ar fi proiectată.. Acest punct de vedere duce rapid la concluzia că serviciul reţea trebuie să fie neorientat pe conexiune, cu două primitive SEND PACKET şi RECEIVE PACKET şi cu foarte puţin în plus.

Cealaltă tabără (reprezentată de companiile de telefoane) afirmă că subreţeaua trebuie să asigure un serviciu orientat pe conexiune sigur.

IMPLEMENTAREA SERVICIULUI NEORIENTAT PE CONEXIUNE

Sunt posibile două organizări diferite, în funcţie de tipul serviciului oferit. Dacă este oferit un serviciu neorientat pe conexiune, atunci pachetele sunt trimise in subreţea individual şi dirijate independent de celelalte.

In acest context, pachetele sunt numite frecvent datagrame (datagrams) (prin analogie cu telegramele), iar subreţeaua este numită subreţea datagramă (datagram subnet). Dacă este folosit serviciul orientat conexiune, atunci, înainte de a trimite pachete de date, trebuie stabilită o cale de la ruterul sursă la ruterul destinaţie

Această conexiune este numită VC (virtual circuit, circuit virtual), prin analogie cu circuitele fizice care se stabilesc în sistemul telefonic, iar subreţeaua este numită subreţea cu circuite virtuale (virtual-circuit subnet).

Să presupunem că procesul P1 din fig. 5-2 are un mesaj lung pentru procesul P2. El transmite mesajul nivelului transport, cu instrucţiunile de livrare către procesul P2 aflat pe calculatorul gazdă H2. Codul nivelului transport rulează pe calculatorul gazdă H1, de obicei în cadrul sistemului de operare.

Acesta inserează la începutul mesajului un antet corespunzător nivelului transport şi transferă rezultatul nivelului reţea, probabil o altă procedură din cadrul sistemului de operare. Să presupunem că mesajul este de patru ori mai lung decât dimensiunea maximă a unui pachet, aşa că nivelul reţea trebuie să îl spargă în patru pachete, 1, 2, 3, şi 4 şi să le trimită pe fiecare în parte ruterului A, folosind un protocol punct-la-punct, de exemplu, PPP. Cum au ajuns la A, pachetele 1, 2 şi 3 au fost memorate pentru scurt timp (pentru verificarea
sumei de control). Apoi fiecare a fost trimis mai departe către C conform tabelei lui A. Pachetul 1 a fost apoi trimis mai departe către E şi apoi către F. Când a ajuns la F, a fost încapsulat într-un cadru al nivelului legătură de date şi trimis către calculatorul gazdă H2 prin reţeaua LAN.

IMPLEMENTAREA SERVICIILOR ORIENTATE PE CONEXIUNE

Pentru serviciile orientate conexiune, avem nevoie de o subreţea cu circuite virtuale. Ideea care se stă la baza circuitelor virtuale este evitarea alegerii unei noi căi (rute) pentru fiecare pachet trimis. În schimb, atunci când se stabileşte o conexiune, se alege o cale între maşina sursă şi maşina destinaţie, ca parte componentă a iniţializării conexiunii şi aceasta este memorată în tabelele ruterelor.

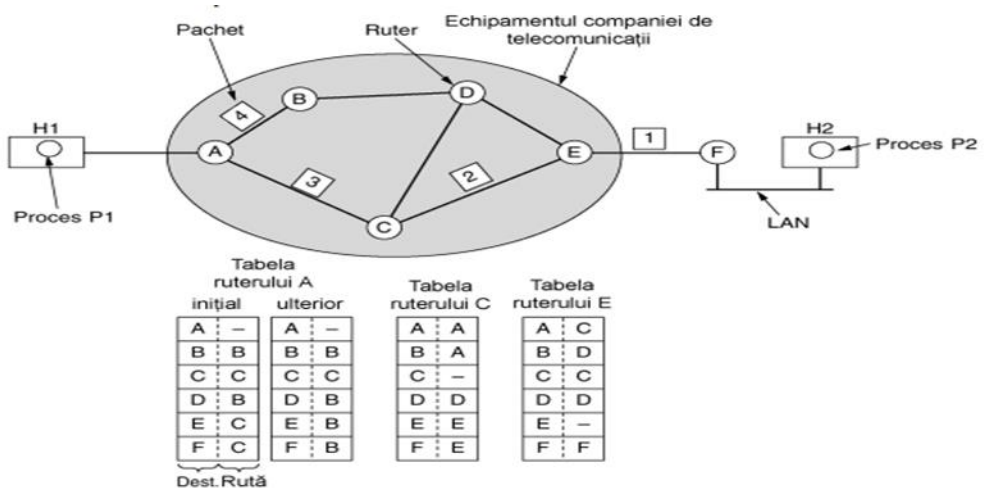


Fig. 5-2. Dirijarea într-o subrețea datagramă.

COMPARAŢIE ÎNTRE SUBREŢELE CU CIRCUITE VIRTUALE ŞI SUBREŢELE DATAGRAM

Problemă	Subrețea datagramă	Subrețea cu circuite virtuale (CV)
Stabilirea circuitului	Nu este necesară	Obligatorie
Adresare	Fiecare pachet conține adresa completă pentru sursă și destinație	Fiecare pachet conține un număr mic de CV
Informații de stare	Ruterele nu păstrează informații despre conexiuni	Fiecare CV necesită spațiu pentru tabela ruterului per conexiune
Dirijare	Fiecare pachet este dirijat independent	Calea este stabilită la inițierea CV; toate pachetele o urmează
Efectul defectării ruterului	Nici unul, cu excepția pachetelor pierdute în timpul defectării	Toate circuitele virtuale care trec prin ruterul defect sunt terminate
Calitatea serviciului	Dificil	Simplu, dacă pentru fiecare CV pot fi alocate în avans suficiente resurse
Controlul congestiei	Dificil	Simplu, dacă pentru fiecare CV pot fi alocate în avans suficiente resurse

Fig. 5-4. Comparație între subrețele datagramă și subrețele cu circuite virtuale.

În interiorul subreţelei există situaţii în care trebuie să se aleagă între facilităţi antagoniste specifice fie circuitelor virtuale, fie datagramelor. Un astfel de compromis este acela între spaţiul de memorie al ruterului şi lăţimea de bandă.

Alt compromis este cel între timpul necesar stabilirii circuitului şi timpul de analiză a adresei. Folosirea circuitelor virtuale presupune existenţa unei faze iniţiale de stabilire a căii, care cere timp şi consumă resurse.

O altă problemă este cea a dimensiunii spaţiului necesar pentru tabela din memoria ruterului. O subreţea datagramă necesită o intrare pentru fiecare destinaţie posibilă, în timp ce o reţea cu circuite virtuale necesită o intrare pentru fiecare circuit virtual.

3.B. ALGORITMI DE DIRIJARE

Algoritmul de dirijare (routing algorithm) este acea parte a software-ului nivelului reţea care răspunde de alegerea liniei de ieşire pe care trebuie trimis un pachet recepţionat

CALEA CEA MAI SCURTA

Modelul topologic al unei retele este un graf in care nodurile corespund comutatoarelor de pachete, iar muchiile corespund liniilor de comunicatie.

Asociind fiecarei muchii o lungime, se poate calcula calea cea mai scurta intre oricare doua noduri, deci cea mai indicata pentru dirijarea pachetelor intre nodurile respective (algoritmul lui Dijkstra).

Lungimea poate avea diverse semnificatii. Daca toate liniile au lungimea unu, gasim caile cu numar minime de noduri intermediare.

Lungimea poate fi distanta geografica intre noduri, costul comunicatiei, intirzierea medie masurata etc

- Algoritmul lui Dijkstra gaseste caile cele mai scurte de la o sursa la toate celelalte noduri.
- El trebuie sa dispuna de informatii topologice generale asupra retelei: listele nodurilor si legaturilor,costurile asociate legaturilor.
- Prin natura sa el este centralizat
- Algoritmul este iteratiasi calculeaza la fiecare iteratie ë cea mai scurta cale de la sursa la un nod al retelei.

DIRIJAREA CENTRALIZATA

Varianta centralizata a algoritmului drumurilor minime (Floyd) utilizeaza un tablou A al distantelor minime, A[i][j] fiind distanta minima de la nodul i la nodul j. Initial,

Calculul drumurilor minime se face iterativ.

La iteratia k, A[i][j] va avea ca valoare cea mai buna distanta intre i si j, pe cai care nu contin noduri numerotate peste k (exceptind i si j).

Deficientele acestei metode sint determinate de:

- vulnerabilitatea retelei,dependenta de functionarea centrului de control (se recurge la dublarea lui);
- supraincarcarea traficului prin transmiterea rapoarelor si a tabelelor de dirijare;
- utilizarea in noduri,in anumite perioade, a unor tabele necorelate, datorita receptiei la momente de timp distincte a noilor tabele.

DIRIJAREA IZOLATA

Pachetul receptionat de nod este plasat in coada cea mai scurta. (O varianta ia in consideratie lungimea cozilor anumitor linii, selectate conforme cailor celor mai scurte)

Desi nepractic, algoritmul este folosit in aplicatii militare (datorita robustetii sale) sau in comparatii de performanta cu alte tehnici (deoarece are un timp de intirziere minim).

DIRIJAREA DISTRIBUITA

Varianta modificata a algoritmului lui Dijkstra care calculeaza drumurile minime de la toate nodurile catre o anumita destinatie.

Conduce in mod natural la o varianta descentralizata

Algoritmul este convergent,asigurind gasirea drumurilor minime intr-un numar finit de pasi

Poate fi utilizat doar pentru datagrame.

DIRIJAREA IERARHICA

Se utilizeaza pentru retele de mari dimensiuni la care tabelele de dirijare ar fi voluminoase.

Comutatoarele sint grupate in regiuni,fiecare comutator cunoscind detaliat caile din regiunea proprie,dar necunoscind structura interna a altor regiuni.

Doua regiuni sint legate prin conectarea unui anumit nod din prima regiune cu un anumit nod din a doua regiune.

Tabela de dirijare se poate reduce,ea avind cite o intrare pentru fiecare nod din regiunea proprie si cite o intrare pentru fiecare din celelalte regiuni.

INUNDAREA

Un alt algoritm static este inundarea (flooding), în care fiecare pachet recepţionat este trimis mai departe pe fiecare linie de ieşire, cu excepţia celei pe care a sosit.

Inundarea generează un mare număr de pachete duplicate, de fapt un număr infinit dacă nu se iau unele măsuri pentru a limita acest proces.

3.C. ALGORITMI PENTRU CONTROLUL CONGESTIEI

Atunci cand mai multe pachete sunt prezente intr-o subretea performatele se degradeaza. Aceasta situatie se numeste congestie. Congestia poate fi produsă de mai mulţi factori. Dacă dintr-o dată încep să sosească şiruri de pachete pe trei sau patru linii de intrare şi toate necesită aceeaşi linie de ieşire, atunci se va forma o coadă. Dacă nu există suficientă memorie pentru a le păstra pe toate, unele se vor pierde. Şi procesoarele lente pot cauza congestia.

SOLUTII:

1. **Prealocarea zonelor tampon**
    - Este aplicabila circuitelor virtuale si consta in rezervarea uneia sau mai multor zone tampon in fiecare nod intermediar, la deschiderea circuitului. In lipsa de spatiu, se alege o alta cale sau se rejecteaza cererea de stabilire a circuitului.
2. **Distrugerea pachetelor**
    - Daca nu exista spatiul necesar memorarii, pachetul receptionat de un nod este ignorat. Deoarece astfel se pot ignora pachete de confirmare, care ar duce la eliberarea spatiului ocupat de pachetele confirmate, se mentine cel putin un tampon de receptie pentru fiecare linie, permitindu-se inspectarea pachetelor primite. De asemenea, se poate limita numarul zonelor tampon de transmisie ale fiecarei linii.
3. **Pachete de permisiune.**
    - Se initializeaza reteaua cu pachete de permisiune (in numar fix). Cind un nod vrea sa transmita, el captureaza un pachet de permisiune si trimite in locul lui pachete de date. Receptorul regenereaza pachetul de permisiune. Se garanteaza astfel ca numarul maxim de pachete nu depaseste numarul de pachete de permisiune, fara a se asigura distribuirea lor conform necesitatilor nodurilor. In plus, pierderea pachetelor de permisiune conduce la scaderea capacitatii retelei.
4. **Pachete de soc.**

- Sunt transmise de comutatoare surselor de date pentru a micsora rata de generare a pachetelor.

EVITAREA BLOCARII DEFINITIVE.

Blocarea reprezinta o situatie limita a unei retele congestionate, cind lipsa de spatiu impiedica transmiterea vreunui pachet.

O solutie de evitare a blocarii definitive este utilizarea in fiecare nod a m+1 zone tampon, m fiind lungimea maxima a cailor retelei. Un pachet sosit de la calculatorul gazda local este acceptat in zona 0. In urmatorul nod trece in 1, apoi in 2 s.a.m.d. Zona "m" a unui nod poate fi goala, poate contine un pachet pentru gazda locala, care este livrat, sau are un pachet pentru un nod distant, care este distrus. In toate cazurile zona "m" se elibereaza, putind avansa un pachet din zona "m-1",apoi "m-2" etc.

Alta varianta pastreaza pentru fiecare pachet o informatie de vechime. La comunicarea dintre doua noduri A si B putem intilni situatiile urmatoare (presupunem ca A are de transmis lui B un pachet mai vechi decit B catre A):

- B are un tampon liber si poate primi cel mai vechi pachet al lui A catre B;
- B nu are un tampon liber, dar are un pachet pentru A si poate primi, prin schimb, cel mai vechi pachet al lui A catre B;
- B nu are nici un tampon liber si nici un pachet catre A; in acest caz, B este fortat sa transmita lui A un pachet la alegere si sa primeasca cel mai vechi pachet al lui A catre B

3.D. PROTOCOLUL IP

O datagramă IP constă dintr-o parte de antet şi o parte de text. Antetul are o parte fixă de 20 de octeţi şi o parte opţională cu lungime variabilă. El este transmis în ordinea **big endian** (cel mai semnificativ primul): de la stânga la dreapta, începând cu bitul cel mai semnificativ al câmpului Versiune. (Procesorul SPARC este de tip bigendian; Pentium este de tip **little endian** - cel mai puţin semnificativ primul). Pe maşinile de tip little endian, este necesară o conversie prin program atât la transmisie cât şi la recepţie.

- Utilizat de sisteme autonome in vederea interconectarii
- Serviciu de transmitere de pachete (host-to-host)
- Translatare dintre diferite protocoale legatura de date
- Ofera servicii neorientate- conexiune, nesigure: datagrame
- Fiecare datagrama este independenta de celelalte
- Nu se garanteaza transmiterea corecta a datagramelor (pierdere, multiplicare,...)
- Foloseste doar adresele logice ale gazdelor
- Adresele IP nu sunt identice cu cele ale nivelului MAC (e.g., adresele hardware ale placilor de retea) pentru ca IP trebuie sa suporte diferite implementari hardware (retele eterogene)

4. NIVELUL TRANSPORT

4.A. CARACTERISTICI ALE NIVELULUI TRANSPORT

ADRESAREA

Atunci când un proces aplicaţie (de exemplu, un proces utilizator) doreşte să stabilească o conexiune cu un proces aflat la distanţă, el trebuie să specifice cu care proces doreşte să se conecteze. . Metoda folosită în mod normal este de a defini adrese de transport la care procesele pot să aştepte cereri de conexiune.

Un scenariu posibil pentru stabilirea unei conexiuni la nivel transport este următorul:

1. Un proces server care furnizează ora exactă şi care rulează pe gazda 2 se ataşează la TSAP 122 aşteptând un apel. Felul în care un proces se ataşează la un TSAP nu face parte din modelul de reţea şi depinde numai de sistemul de operare local. Poate fi utilizat un apel de tip LISTEN din capitolul precedent.
2. Un proces aplicaţie de pe gazda 1 doreşte să afle ora exactă; atunci el generează un apel CONNECT specificând TSAP 1208 ca sursă şi TSAP 1522 ca destinaţie. Această acţiune are ca rezultat în cele din urmă stabilirea unei conexiuni la nivel transport între procesele aplicaţie de pe gazda 1şi serverul 1 de pe gazda 2.
3. Procesul aplicaţie trimite o cerere o cerere pentru timp
4. Procesul server de timp răspunde cu timpul curent.
5. Conexiunea transport este apoi eliberată.

STABILIREA CONEXIUNII

La prima vedere, ar părea suficient ca o entitate de transport să trimită numai un TPDU CONNECTION REQUEST şi să aştepte replica CONNECTION ACCEPTED . Problema apare deoarece reţeaua poate pierde, memora sau duplica pachete. Acest comportament duce la complicaţii serioase. Se poate imagina o subreţea care este atât de congestionată încât confirmările ajung greu înapoi, şi, din această cauză, fiecare pachet ajunge să fie retransmis de câteva ori.

Punctul crucial al problemei este existenţa duplicatelor întârziate. El poate fi tratat în mai multe feluri, dar nici unul nu este într-adevăr satisfăcător. O posibilitate este de a utiliza adrese de transport valabile doar pentru o singură utilizare. O altă posibilitate este de a atribui fiecărei conexiuni un identificator (adică, un număr de secvenţă incrementat pentru fiecare conexiune stabilită), ales de cel care iniţiază conexiunea, şi pus în fiecare TPDU, inclusiv în cel care iniţiază conexiunea. Se poate încerca şi o altă soluţie. În loc să se permită pachetelor să trăiască la nesfârşit în subreţea, se poate inventa un mecanism care să elimine pachetele îmbătrânite.

ELIBERAREA CONEXIUNII

Există două moduri de a termina o conexiune: eliberare simetrică şi eliberare asimetrică. Eliberarea asimetrică este bruscă şi poate genera pierderi de date. După stabilirea conexiunii, gazda 1 trimite un TPDU care ajunge corect la gazda 2. Gazda 1 mai trimite un TPDU dar, înainte ca acesta să ajungă la destinaţie, gazda 2 trimite DISCONNECT REQUEST . În acest caz, conexiunea va fi eliberată şi vor fi pierdute date.

Eliberarea simetrică este utilă atunci când fiecare proces are o cantitate fixă de date de trimis şi ştie bine când trebuie să transmită şi când a terminat. În alte situaţii însă, nu este deloc uşor de determinat când trebuie eliberată conexiunea şi când a fost trimis tot ce era de transmis. S-ar putea avea în vedere un protocol de tipul următor: atunci când 1 termină, trimite ceva de tipul:

Am terminat. Ai terminat şi tu? Dacă gazda 2 răspunde: Da, am terminat. Închidem! conexiunea poate fi eliberată în condiţii bune. Din nefericire, acest protocol nu merge întotdeauna. Binecunoscuta problemă a celor două armate este similară acestei situaţi.

CONTROLUL FLUXULUI

Una din problemele cheie a apărut şi până acum: controlul fluxului. La nivel transport există asemănări cu problema controlului fluxului la nivel legătură de date, dar există şi deosebiri.

La nivel legătură de date, emiţătorul trebuie să memoreze cadrele transmise, pentru că poate fi necesară retransmiterea acestora. Dacă subreţeaua oferă un serviciu datagramă, atunci entitatea de transport emiţătoare va trebui să memoreze pachetele trimise din aceleaşi motive.

Pe scurt, dacă serviciul reţea nu este sigur, emiţătorul va trebui să memoreze toate TPDU-urile trimise, la fel ca la nivel legătură de date. Totuşi, folosind un serviciu la nivel reţea sigur sunt posibile unele compromisuri. În particular, dacă emiţătorul ştie că receptorul are întotdeauna tampoane disponibile, atunci nu trebuie să păstreze copiile TPDU-urilor trimise. Totuşi, dacă receptorul nu poate garanta că orice TPDU primit va fi acceptat, emiţătorul va trebui să păstreze copii. În ultimul caz, emiţătorul nu poate avea încredere în confirmarea primită la nivel reţea, deoarece aceasta confirmă sosirea TPDU-ului la destinaţie, dar nu şi acceptarea lui.

Chiar dacă receptorul va realiza memorarea temporară a mesajelor primite, mai rămâne problema dimensiunii tamponului. Dacă dimensiunea tampoanelor ar fi constantă, egală cu cel mai mare TPDU posibil, atunci va apărea o risipă de spaţiu ori de câte ori este primit un TPDU mai scurt. Dacă dimensiunea tampoanelor este aleasă mai mică decât cel mai mare TPDU posibil, atunci pentru memorarea unui TPDU mai lung vor fi necesare mai multe tampoane, iar complexitatea operaţiei va creşte.

O altă soluţie este utilizarea unor tampoane de dimensiune variabilă. Avantajul este o mai bună utilizare a memoriei, cu preţul unei gestiuni a tampoanelor mai complicată. O a treia posibilitate este alocarea unui singur tampon circular pentru fiecare conexiune. Această soluţie are de asemenea avantajul unei utilizări eficiente a memoriei, dar numai în situaţia în care conexiunile sunt relativ încărcate.

MULTIPLEXAREA

Multiplexarea mai multor conversaţii pe conexiuni, circuite virtuale şi legături fizice joacă un rol important în mai multe niveluri ale arhitecturii reţelei. În cazul nivelului transport, multiplexarea poate fi necesară din mai multe motive. De exemplu, dacă doar o singură adresă de reţea este disponibilă pe o gazdă, toate conexiunile transport de pe acea maşină trebuie să o folosească. Când un TDPU soseşte este necesar un mod de a spune cărui proces trebuie dat. Această situaţie numită multiplexare în sus.

Multiplexarea poate să fie utilă nivelului transport şi din alt motiv, legat de deciziile tehnice şi nu de politica de preţuri ca până acum. Să presupunem, de exemplu, că o subreţea foloseşte intern circuite virtuale şi impune rată de date maximă p fiecare dintre ele. Dacă un utilizator are nevoie de mai multă lăţime de bandă decât poate oferi un circuit virtual, o soluţie este ca nivelul transport să deschidă mai multe conexiuni reţea şi să distribuie traficul prin acestea (într-un sistem round-robin), .Acest mod de operare se numeşte multiplexare în jos.

4.B. PROTOCOLUL TCP

TCP (**Transport Communication Protocol** - protocol de comunicaţie de nivel transport) a fost proiectat explicit pentru a asigura un flux sigur de octeţi de la un capăt la celălalt al conexiunii într-o inter-reţea nesigură.

O caracteristică importantă a TCP, care domină structura protocolului, este aceea că fiecare octet al unei conexiuni TCP are propriul său număr de secvenţă, reprezentat pe 32 biţi.

Entităţile TCP de transmisie şi de recepţie interschimbă informaţie sub formă de segmente. Un segment TCP constă dintr-un antet de exact 20 de octeţi (plus o parte opţională) urmat de zero sau mai mulţi octeţi de date. Programul TCP este cel care decide cât de mari trebuie să fie aceste segmente.

Protocolul de bază utilizat de către entităţile TCP este protocolul cu fereastră glisantă. Atunci când un emiţător transmite un segment, el porneşte un cronometru. Atunci când un segment ajunge la destinaţie, entitatea TCP receptoare trimite înapoi un segment (cu informaţie utilă, dacă aceasta există sau fără, în caz contrar) care conţine totodată şi numărul de secvenţă următor pe care aceasta se aşteaptă să-l recepţioneze. Dacă cronometrul emiţătorului depăşeşte o anumită valoare înaintea primirii confirmării, emiţătorul retransmite segmentul neconfirmat.

Deşi acest protocol pare simplu, pot apărea multe situaţii particulare care vor fi prezentate mai jos. Segmentele pot ajunge într-o ordine arbitrară, deci octeţii 3072-4095 pot fi recepţionaţi, dar nu pot fi confirmaţi datorită absenţei octeţilor 2048-3071. Segmentele pot de asemenea întârzia pe drum un interval de timp suficient de mare pentru ca emiţătorul să detecteze o depăşire a cronometrului şi să le retransmită. Retransmisiile pot include porţiuni de mesaj fragmentate altfel decât în transmisia iniţială, ceea ce impune o tratare atentă, astfel încât să se ţină evidenţa octeţilor primiţi corect. Totuşi,

deoarece fiecare octet din flux are un deplasament unic faţă de începutul mesajului, acest lucru se poate realiza. TCP trebuie să fie pregătit să facă faţă unor astfel de situaţii şi să le rezolve într-o manieră eficientă. Un efort considerabil a fost dedicat optimizării performanţelor fluxurilor TCP, ţinându-se cont inclusiv de probleme legate de reţea.