Dalyko turinys

21. **OSI kanalinis lygis. Projektavimo klausimai.**

22. **MAC polygis, jo sprendžiamos problemos. FDM, TDM. Protokolai (pure ALOHA, slotted ALOHA, CSMA, CSMA/CD (su collision detection), protokolai be kolizijų, riboto varžymosi protokolai, adaptive tree walk protokolas).**

23. **MAC polygio protokolai naudojami su bevielėmis komunikacijomis.**

24. **IEEE standartai 802.3 - CSMA/CD (Ethernet), Mančesterio kodai, 802.4 - token bus, 802.5 - token ring.**

25. **Binary exponentional backoff algoritmas.**

26. **Switcho veikimo principas.**

27. **Tinklų standartų 802.3, 802.4, 802.5 palyginimas – pranašumai, trūkumai.**

28. **LLC – Logical Link Control 802.2. Bridges**

29. **Tinklo lygis. Pagrindinės funkcijos, serviso transporto lygiui pateikimas, vidinės potinklio struktūros variantai**

30. **Maršrutizavimo algoritmai. Maršrutizavimas trumpiausiu keliu, užtvindymas, nuotolių lentelės metodas, hierarchinis maršrutizavimas.**

31. **Maršrutizavimas mobiliems kompiuteriams.**

32. **Persipildymo (congestion) valdymas. leaky bucket, token bucket, load shedding, choke paketai, RSVP.**

33. **Internetworking. Tunneling, paketų fragmentacija.**

1. KT

– tarpusavyje sujungtų savarankiškų kompiuterių aibė. Kompiuteriai tarpusavyje sujungti, jei jie gali keistis informacija. Sujungimo būdas (laidas, lazeris, mikrobangos) nėra svarbu. Nepriklausomi – jei nėra susiję master/slave sąryšiu, jei nė vienas sistemos kompiuteris negali priverstinai valdyti kito. Mainframe su terminalais nėra kompiuterių tinklas. Kompiuterių tinklas nėra paskirstyta sistema. Paskirstyta sistema yra programinė

įranga, veikianti naudodama kompiuterių tinklų galimybes. Kompiuterių tinklų klasifikacija.

Klasifikacija pagal duomenų perdavimo technologijas:

Broadcast tinklai turi vieną komunikacijos kanalą, kurį dalinasi visi tinklo kompiuteriai. Paketą, išsiustą bet kurio kompiuterio, gauna visi kiti esantys tinkle. Pakete yra laukas su adresu, kuris nurodo, kam skirtas pranešimas. Gaudamas pranešimą, kompiuteris patikrina šį lauką, ir jei paketas skirtas jam tai priima jį, o jei ne- ignoruoja. Taip pat Broadcast leidžia adresuoti paketą visiems, naudojant spec kodą adreso lauke.

Point-to-point tinklai susideda iš daug sąryšių tarp individualių kompiuterių porų. Prieš patekdamas į reikiamą kompiuterį, paketas apeina vieną ar daugiau tarpinių kompiuterių. Broadcast tinklai pagal topologiją skirstomi: bus, ring.

Point- to- point pagal topologiją skirstomi: complete, ring, tree, star, interseeting rings, irregular. Klasifikacija pagal tinklų skalę:

1m -personal area network; 10m- 1km LAN (local area network); 1km- 100km MAN (metropolian area network); 100km- 1000km WAN (wide area network); 10000km Internet

2. KT architektūros pagrindinės sąvokos, jų tarpusavio sąryšis

Tinklai yra organizuoti kaip lygmenų (sluoksnių) serijos. Lygmenų skaičius, pavadinimai, realizacija, funkcijos skirtinguose tinkluose gali skirtis. Tačiau visur kiekvieno lygmens paskirtis yra pateikti tam tikrus servisus aukštesniam lygmeniui, apsaugant jį nuo detalių kaip tie servisai iš tiesų realizuoti.

Lygmuo n vienoje mašinoje komunikuoja su lygmeniu n kitoje. Komunikavimui yra naudojamas n lygio protokolas. Protokolas – susitarimų ir taisyklių rinkinys, naudojamas komunikacijai. To paties lygio bendraujančios esybės skirtingose mašinose yra vadinamos peers. Protokolas yra serviso realizacija.

Tarp kiekvienos poros gretimų lygmenų yra interfeisas. Kiekvienas aukštesnis lygmuo naudojasi žemesnio lygmens pateiktu interfeisu. Interfeisas apibibrėžia kurias operacijas ir servisus žemesnis lygmuo padaro prieinamus aukštesniam. Lygis pateikia tam tikrus servisus, kuriuos galima pasiekti naudojantis interfeisu.

Kompiuterių tinklų architektūra – lygių ir protokolų rinkinys. Protokolų rinkinys konkrečioje sistemoje vadinamas protokolų steku (protocol stack), pvz. TCP/IP protokolų stekas.

3. Paslaugų tipai, primityvai, partnerių sąveikos modelis. Serverių ir protokolų sąryšis

Servisai yra dviejų tipų: 1) kontaktiniai (telefono modelis). Serviso vartotojas pirma sukuria ryšį, tada juo naudojasi ir galiausiai atsijungia. Dažniausiai bitai gaunami tokia tvarka, kokia buvo išsiųsti. 2) Nekontaktiniai. (pašto sistemos modelis). Kiekviename pranešime yra pilnas gavėjo adresas ir kiekvienas keliauja sistema nepriklausomai nuo kitų. Įprastai, kai du pranešimai išsiųsti tam pačiam gavėjui, jie bus gauti ta pačia eilės tvarka, tačiau pirmas pranešimas gali būti užlaikytas ir tokiu atveju antras pasieks gavėją anksčiau.

Kiekvienas servisas gali būti charakterizuojamas serviso kokybe (quality of service). Servisai gali būti patikimi (neprarandantys duomenų) arba nepatikimi. Paprastai patikimo serviso realizacijoje gavėjas turi patvirtinti, kad gavo pranešimą. Patvirtinimo procesas gali sukelti užlaikymus, kurie pageidautini ne kiekvienu atveju.

Servisas formaliai nusakomas primityvų (operacijų) aibe, pateikiama vartotojui ar kitai esybei. Primityvai pasako servisui atlikti kokį nors veiksmą, arba informuoja apie partnerio (peer) atliktą veiksmą.

Primityvų klasifikacija: Užklausimas (REQUEST) - objektas reikalauja serviso atlikti tam tikram darbui; Požymis (INDICATION) – objektas informuojamas apie įvykį; Atsakymas (RESPONSE) – objektas nori reaguoti į įvykį. Patvirtinimas (CONFIRM) – objektas turi būti informuotas apie paklausimą.

Servisų ir protokolų sąryšis. Servisas – primityvų aibė, kurią lygis pateikia aukštesniam lygiui. Servisas apibrėžia, kokius veiksmus lygis yra pasiruošęs atlikti vartotojams, bet nieko nenurodo apie realizaciją. Servisas susijęs su interfeisu tarp dviejų lygių, kur žemesnysis lygis – serviso tiekėjas, o aukštesnysis – serviso vartotojas. Protokolas – taisyklių aibė, reguliuojanti freimų, paketų, pranešimų, kuriais apsikeičia objektai partneriai, formatą ir prasmę. Objektai naudoja protokolus tam, kad realizuoti savo pateikiamą servisą. Jie gali pakeisti protokolą, svarbu, kad nekeistų serviso, kuris matomas jų vartotojams. Servisas apibrėžia operacijas, bet nenusako kaip jos realizuotos.

4. OSI sluoksninis modelis. Sluoksnių funkcijos ir charakteristika

Pats OSI modelis nėra tinklo architektūra, nes nespecifikuoja tikslių servisų ir protokolų, specifikuoja tik kiekvieno lygio funkcijas, ISO yra išleidę atskirus standartus kiekvienam lygiui. OSI lygių funkcijos:

Fizinis lygis (physical layer): Bitų perdavimas komunikacijos kanalu. Projektavimo problemos susijusios su tuo, kad kai viena pusė siunčia 0, kita gautų 0, o ne 1, ir atvirkščiai, perdavimo priemonės, jų fizinės charakteristikos, duomenų kanalo perdavimo greitis. Kanalinis lygis (data link layer): Švarios ryšio linijos tarp dviejų mazgų pateikimas tinkliniam lygiui, informacijos skirstymas "freimais"(grupavimas,pasiuntimas, patvirtinimas, klaidas randantys ir taisantys kodai), freimų unikalumo ir korektiškumo užtikrinimas, greito siuntėjo ir lėto priėmėjo problema – srauto kontrolė, bendrai naudojamo duomenų perdavimo kanalo kontroliavimas, jei naudojamas dupleksas, patvirtinimai gali susikirsti su duomenimis.

Tinklo lygis (network layer): Darbo potinklyje valdymas, paketų siuntimas iš "source" į "destination" hostus – maršrutizacija, tinklo užsikimšimo valdymas (congestion), tinklo transporto apskaita, heterogeniškų tinklų apjungimas – internetworking (adresavimo klausimai, paketų dydžiai, protokolų skirtumai), Broadcast tinkluose maršrutizavimo problemos nėra, todėl šis lygis ten dažniausiai "plonas". Transporto lygis (transport layer): Duomenų perdavimas tarp programinių esybių galutinėse sistemose (end-point hosts). Tai pirmas lygis, kuriame betarpiškai bendrauja galutinės sistemos (ar procesai). Aukštesni lygiai izoliuojami nuo potinklio realizacijų skirtumų, Kanalo pateikimas kiekvienai aplikacijai, derinimasis prie tinklo lygio ryšio kanalų, unikalumo ir korektiškumo užtikrinimas, serviso kokybės (QoS) valdymas, srauto kontrolė ir buferizavimas.

Seanso lygis (session layer): Nustatymas seansų tarp aplikacijų, transporto lygio servisų praplėtimas, dialogo kontrolė, markerio (token) valdymas, sinchronizacija (checkpoints) t.y seanso lygis turi būdus įterpti kontrolinius taškus, kad būtų galima pradėti darbą nuo ten, kur nutraukė.

Duomenų pavaizdavimo lygis (presentation layer): Susijęs su informacijos sintakse ir semantika, skirtas suvienodinti duomenų formatus, duomenų spaudimas (compression), kodavimas.

Taikomasis lygis (application layer)- Visa tinklų programinė įranga – tinklo virtualūs terminalai, failų perdavimas, e-mail, failų serveriai, DB serveriai, objektų serveriai, transcakcijų serveriai, etc

5. TCP/IP modelis, jo sluoksniai

Kuriant TCP/IP, daugiausia dėmesio buvo skiriama patikimumui. Apibrėžtas 1974 metais. (Host-to- network layer)- labiau interfeisas, nei lygis. Tinklo lygis(internet layer): Nekontaktinis (connection-less) servisas, kiekvienas paketas perduodamas

nepriklausomai, paketų tvarkos problema, apibrėžia paketo formatą ir IP protokolą. Transporto lygis (transport layer): TCP – patikimas, pateikia kontaktinį (CO) servisą, UDP – nepatikimas, pateikia nekontaktinį (CL) servisą Taikomasis lygis (aplication layer)- Telnet, FTP, SMTP, DNS, etc

6. TCP/IP ir OSI modelių palyginimas, OSI kritika, TCP/IP kritika

OSI konceptualiai apibrėžia servisus, protokolus ir interfeisus, o TCP/IP neturi tikslaus šių sąvokų atsiskyrimo. OSI yra bendresnis, nes TCP/IP tinka tik TCP/IP tinklams, OSI pirma atsirado modelis, poto protokolai, o TCP/IP modelis nusako protokolus. OSI transporto lygis- kontaktinis servisas, tinklo lygis- ir CO ir LC. TCP/IP- transporto lygis- CO ir LC, tinklo lygis- nekontaktinis servisas. OSI kritika. Blogai parinktas laikas, per daug sudėtinga technologija, apibrėžia nereikalingus lygmenis, prastos, sudėtingos, neefektyvios realizacijos, prasta politika.

TCP/IP kritika. Nėra apibrėžtų bendrų sąvokų, modelis nėra bendras, Host-to-network – labiau interfeisas, o ne lygis, nėra išskirtų fizinio ir kanalinio lygių, kai kurie taikomieji protokolai neišbaigti.

7. TCP/IP adresacija: adreso struktūra, adresų klasė, specialūs IP adresai, potinklis, potinklis, potinklio šablonas (subnet mask)

Adreso struktūra. Sudaro IP adresas ir subnetmaskas (po 32 bitus). Žymėjimas x.y.z.w po 8 bitus. Adresą sudaro dvi dalys – tinklo dalis ir hosto dalis. Tinklo dalis = IP AND subnetmask. (pvz., IP: 192.168.0.15, subnetmask:255.255.255.0, tinklas yra C tipo: 192.168.0). Adresų klasės. A: 1|7 bits network|24 bits host; B: 10|14 bits network|16 bits host; C: 110|21 bits network|8 bits host; D: 1110|Multicast address; E: 1111|rezervuota ateičiai.

Specialūs IP: 0.0.0.0 – šitas hostas. 0|host (network|host) – hostas šiame tinkle. 1.1.1.1 – broadcast vietiniame tinkle. Network|1 – broadcast kitame tinkle. 127.x.y.z – loopback savo hoste.

Potinkliai. Tinklas gali būti padalintas į keletą potinklių, taip kad iš išorės tai būtų nepastebima. Kuriant potinklius yra pasiimamas reikiamas bitų kiekis iš hosto dalies ir jame nurodomas potinklis. Pvz., B klasės adresas: 10|14 bits network|4 bits subnet|10 bits host. Subnetmaskas nurodo, kuri adreso dalis yra tinklas + subnetas, o kuri – hostas. Pagal pavyzdį subnetmask būtų 255.255.240.0. Potinklių pvz. būtų x.y.16.0, x.y.32.0. atitinka x.y.00010000.0 ir x.y.00100000.0. Routeris, atlikęs IP AND subnemask, nustatys, kur reikia routinti paketus.

8. Adresų rezoliucijos problema, ARP protokolas

Adresų rezoliucija. Techinė įranga (hardware) atpažįsta ir naudoja tik MAC adresus. IP ir aukštesni lygiai naudoja tik IP adresus. Pasekmė – reikalinga programinė įranga adresų transliacijai. Tai tinklo interfeiso dalis, žinoma kaip adresų rezoliucija. Kanalinio lygio protokolas.

Duota: Lokalus tinklas T, IP adresas K kompiuterio tinkle T. Reikia surasti MAC adresą kompiuteriui K. Tam naudojamas ARP – Address Resolution Protocol. Pateiksime 2 fizinių adresų pavyzdžius: Ethernet su dideliais ir fiksuotais fiziniais adresais bei proNet su mažais, lengvai konfigūruojamais fiziniais adresais.

ProNet'e MAC adresas sukonfigūruojamas taip, kad jis būtų paprasta funkcija nuo IP adreso. Pavyzdžiui, galima paprastai pasirinkti, kad MAC adresas būtų IP adreso host_id dalis. Tada kiekvienas gali paprastai paskaičiuoti reikalingą adresą pagal žinomą IP. Kitas paprastas būdas – laikyti visus IP-MAC adresų atitikimus statinėje lentelėje. Bet netinkamas keičiant tinklo interfeisus, IP adresus ir t.t. Todėl lentelių įrašai turi būti nusutatomi dinamiškai, ir kažkiek laikomi "cache".

Address Resolution Protocol ARP – adresų rezoliucija per dinaminį surišimą (dynamic binding). Jau žinomi adresai saugomi lentelėje, kurios įrašas yra pora – IP adresas ir MAC adresas. Lentelės saugomos tik lokaliam tinklui ir generuojamos automatiškai. Protokolas veikia po tinklo lygiu kaip interfeiso dalis tarp OSI tinklo ir OSI Data link lygių.

ARP algoritmas: 1) Ieškoti reikalingo IP adreso T ARP lentelėje. 2) Jei nerastas: a) Nusiųsti ARP pranešimą su adresu T, b) Gauti atsakymą su T MAC adresu; c) Prisidėti įrašą į lentelę 3) Grąžinti reikalingą MAC adresą iš lentelės. Užklausimas siunčiamas broadcastu, bet į jį reaguoja tik tas kompiuteris, kieno IP adresas yra pranešime. Atsakymas siunčiamas unicastu. Kadangi ARP yra tinklo interfeiso programinės įrangos dalis, visi aukštesnių lygių protokolai gali naudoti išimtinai IP adresus, nesirūpindami fiziniais adresais.

ARP pranešimo enkapsuliacija. ARP message talpinamas į Frame data area, o pats freimas atrodo taip: Frame header | Frame data area | CRC Taigi ranešimas siunčiamas freime, kad tai ARP, atskiriama pagal freimo tipą (Ethernete 0806 – 16-tainėj sistemoj). Dest. Address | Source Address | Frame type | Data in frame 806 complete ARP message

ARP protokolo formatas. (lentelė) Pirma eilutė: 0-15 bitukai Hardware Address Type, 16-31 Protocol Address Type, antra eilutė: 0-7 Haddr Len, 8-15 Paddr Len, 16-31 Operation, trečia eilutė: Sender Haddr (first 4 octets), ketvirta eilutė: 0-15 Sender Haddr (last 2 octets), 16-31 Sender Paddr (first 2 octets), penkta eilutė: Sender paddr (last 2 octets),

target haddr (first 2 octets), 6a eilutė: Target haddr (last 4 octets), 7a eilutė: Target paddr (all 4 octets).

Pavyzdys – Ethernet tipo tinklams, bet formatas pakankamai bendras. HARDWARE ADDRESS TYPE = 1 for Ethernet PROTOCOL ADDRESS TYPE = 0x0800 for IP OPERATION = ARP request, ARP response, RARP request or RARP response Turi laukus ir siuntėjo, ir ieškomo kompiuterio fiziniams ir IP adresams. Request metu ieškomas fizinis adresas nustatomas į 0. Galima išsitraukti siuntėjo tiek fizinį, tiek IP adresą. Atsakymo metu sender ir target sukeičiami vietomis.

9. RARP protokolas

RARP protokolas reikalingas IP adresui rasti sistemos startavimo metu. Paprastai IP adresas laikomas kietajame diske, kur operacinė sistema startavimo metu jį suranda. Išimtis: darbo stotys, neturinčios disko. Joms reikia bendrauti su file serveriu naudojant IP adresą. IP adresą reikia gauti prieš startuojant operacinei sistemai. Sprendimas – pasinaudojant tinklo komunikacijomis, reikia sukontaktuoti su serveriu ir gauti savo IP adresą. Paradoksas: kompiuteris komunikuoja su nutolusiu serveriu tam, kad gautų adresą, reikalingą komunikacijoms. Faktoriai, kurie tai padaro įmanomu: 1) Komunikacija vykdoma tik lokaliame fiziniame tinkle 2) Mašina save gali identifikuuoti unikaliu fiziniu adresu 3) Serveris gali turėti lenteles su MAC-IP atitikimais.

IP adreso radimo idėja: Pasiunčiamas užklausimas serveriui ir laukiama atsakymo. Užklausime mašina turi būti identifikuojama unikaliai. Užklausimas yra broadcastinamas – nes fizinis serverio adresas nėra žinomas. Unikalus identifikatorius – fizinis adresas, nes: a) Jis visada surandamas iš tinklo interfeiso įrangos, nepriklausomas nuo OS; b) Visos mašinos lokaliame tinkle turės vienarūšius unikalius fizinius adresus.

Tai realizuoja RARP – Reverse Address Resolution Protocol. RARP naudoja bediskės darbo stotys. Tik tos mašinos, kurios autorizuotos kaip RARP serveriai gali apdoroti užklausimą ir pasiųsti atsakymą. Tinkle turi būti bent vienas RARP serveris. RARP serveris palaiko IP-MAC adresų atitikimų lenteles. RARP serveriai gali būti primary arba backup.

10. DIP protokolas – pagrindiniai veikimo principai. Datagramų perdavimas, fragmentacija. Datagramos formatas, laukų paaškinimas.

IP protokolas yra to paties lygio kaip ir tinklo protokolas. Šių protokolų dėka turime internetą tokį, koks jis dabar yra. IP protokolas - abstrakcija arba virtualus tinklas, nekontaktinis, nepatikimų duomenų paketų pristatymo servisas. IP apibrėžia bazinį duomenų perdavimo vienetą ir nustato taisykles paketų apdorojimui. IP programinė įranga atlieka maršrutizavimo funkcijas. IP yra nekontaktinis, nes datagrama turi savyje

galutinį adresatą, kiekviena datagrama siunčiama ir apdorojama nepriklausomai. Maršrutai gali pasikeisti bet kuriuo metu. IP leidžia datagramoms vėluoti, kartotis, būti pristatomoms ne ta tvarka, pasimesti. Todėl IP vadinamas best effort delivery. Motyvacija – apjungti visus įmanomus tinklus.

IP datagramos idėja atitinka žemesnių lygių freimus. Jos pagrindinės dalys yra header ir duomenys. IP datagrama naudoja IP adresus.

Datagramos formatas. (lentelė) Pirma eilutė: 0-3 Version, 4-7 IHL, 8-13 Type of service, 14-15 nenaudojami, 16-31 Total length, antra eilute: 0-15 Identification, 16 nenaudojamas, 17 DF, 18 MF, 19-31 Fragment Offset, trečia eilutė 0-7 Time to live, 8-15 Protocol, 16-31 Header checksum, Ketvirta eilutė 0-31 Source address, penkta eilutė 0-31 destination source, 5a eilutė Options (0 or more words), 6a ir kitos eilutės Data.

Version lauke nurodoma versija (dabartinė yra 4). Kol hederio ilgis nėra konstanta, IHL laukas pasako, kokio ilgio hederis yra (in 32-bit words). Minimali reikšmė 5 būna tada, kai nėra nustatytos jokios opcijos. Maksimali reikšmė 15. Type of service laukas reikalingas atskirti skirtingas serviso klases. Įmanomos įvairios patikimumo kombinacijos. Pirmieji 3 bitai nuo dešinės vadinami Precedence lauku ir nusako datagramos prioritetą (reikšmė nuo 0 iki 7). Kiti 3 bitai yra flagai D, T, R (Delay, Throughput, Reliability). Teoriškai šie laukai leidžia maršrutizatoriui apsispręsti tarp, pvz., palydovinio ryšio su aukštu pralaidumu ir kabelio su žemu. Tačiau iš tikrųjų dauguma maršrutizatorių tiesiog ignoruoja Type of service lauką. Time to live laukas yra kaunteris, ribojantis paketo gyvavimo laiką.maksimali jo reikšmė – 255 hop. Kiekvienas routeris Time to live lauko reikšmę sumažina vienetu. Kai jo reikšmė pasiekia 0, siuntėjui nusiunčiamas pranešimas apie nesėkmingą paketo nuėjimą. Taip išvengiama paklydusių paketų. Header checksum laukas tik patikrina hederį. Taip išvengiama klaidų, kurių priežastis routerio viduje bloga atmintis. Source address ir Destination address laukai saugo tinklo ir hosto numerius. Options laukas turi tokius laukelius: COPY flag, Option class (Datagram or network control arba Debugging and measurement), Option number (Record route option, Source route option (Strict, Loose), Timestamp option). Tačiau Options laukas gali būti ir tuščias.

Datagramų perdavimas.

Datagrama siunčiama per fizinius tinklus nuo siuntėjo iki maršrutizatoriaus, tarp tarpinių maršrutizatorių ir galiausiai nuo galutinio maršrutizatoriaus adresatui. Tinklo įranga neatpažįsta datagramos formato ir IP adresų. Reikalinga enkapsuliacija. Visa datagrama yra laikoma duomenimis - persiunčiama freimo duomenų lauke. Freimo tipas pasako, kad tai IP datagrama. Freimo gavėjo adresas nurodo sekantį žingsnį. Datagrama enkapsuliuojama į freimą tam, kad ją perduotų fiziniu tinklu. Freimo gavėjo adresas yra sekančio žingsnio, kur turi būti pasiųsta datagrama, adresas; jis gaunamas verčiant sekančio žingsnio IP adresą atitinkamu fiziniu adresu. Datagrama išlieka visos kelionės per internetą metu, o freimas gyvena tik vieną žingsnį. Kiekviename žingsnyje yra ištraukiama datagrama, ir kuriamas naujas freimas.

Datagramų fragmentacija

Problema atsiranda, kai datagrama yra didesnė už freimą, nes kiekviena tinklo technologija apibrėžia maksimalų freimo dydį (Maximum Transmission Unit (MTU)). MTU skirtinguose tinkluose skiriasi. Internetas – heterogeniška aplinka, nes apjungia įvairias skirtingas technologijas.

Taigi datagramos kartais turi būti skaidomos į fragmentus. Fragmentacija atliekama hostuose ir maršrutizatoriuose. Fragmentuoti reikia tada, kai datagrama didesnė, nei tinklo, per kurį siunčiama, MTU. Kiekvienas fragmentas turi datagramos headeri, beveik identišką originalios datagramos headeriui, išskyrus flagus. Fragmentai persiunčiami atskirai, galutinis adresatas vėl surenka fragmentus į vientisą datagramą.

Fragmentų surinkimas.

Kad galutinis adresatas žinotų, kuriai datagramai gautas fragmentas priklauso, reikalingas Identification laukas. Visi tos pačios datagramos fragmentai turi tą pačią Identification reikšmę. Kur originalioje datagramoje buvo fragmentas pasako Fragment offset laukas. Kontroliniai flagai DF (Don‘t Fragment) ir MF (More Fragments). DF nurodo maršrutizatoriams nefragmentuoti datagramos, nes gavėjas negalės iš fragmentų sudėti originalios datagramos. MF – visų fragmentų, išskyrus paskutinį šis bitukas lygus 1. Tai reikalinga tam, kad gavėjas žinotų, kada visi fragmentai atėjo. Kad gavėjas nelauktų amžinai pamestų fragmentų, naudojami reassembly timers. Jei nors vienas fragmentas pasimeta, prarandama visa datagrama.

Fragmento fragmentavimas.

Kartais fragmentui gali tekti eiti per tinklą su dar mažesniu MTU, ir jis bus per didelis persiuntimui. Tokiu atveju reikalingas fragmentų fragmentavimas – maršrutizatorius padalina fragmentus į mažesnes dalis. Dalinama taip, kad visi fragmentai gautųsi to paties lygio. Duomenų poslinkiai nurodomi pagal originalią datagramą. Galutinis adresatas neskiria subfragmentų nuo fragmentų.

11. IP datagramų maršrutizavimas

Maršrutizavimas – procesas, kai yra parenkamas kelias, kuriuo siųsti paketus. Maršrutizatorius – kompiuteris, kuris tai atlieka. IP maršrutizavimas parenka kelią, kaip pasiųsti datagramą per skirtingus fizinius tinklus. Naudoja IP adresus. Sekantis žingsnis būna arba maršrutizatorius, arba galutinis adresatas. Skiriami du pristatymo tipai: Tiesioginis (direct delivery) ir Netiesioginis (indirect delivery).

Datagramos perdavimas lokaliame tinkle. Perduodant lokaliai, maršrutizatoriai nenaudojami. Norėdamas perduoti datagramą, hostas: 1) Enkapsuliuoja datagramą į fizinį

freimą, 2) Suranda gavėjo IP adresui atitinkamą fizinį adresą, 3) Nustato jį freimui kaip gavėjo adresą, 4) Naudoja tinklo techninę įrangą pristatyti freimui.

Kaip sužinoma, kad gavėjas yra lokaliame tinkle? Visi į vieną tinklą sujungti kompiuteriai turi vienodą network-prefix IP adrese – taigi reikia ištraukti tinklo dalį iš gavėjo adreso, ir palyginti su savo.

Netiesioginis pristatymas – hostas perduoda datagramą maršrutizatoriui, kuris ją perduoda toliau link gavėjo. Maršrutizavimas remiasi lentelėmis (IP routing table), esančiomis kiekvienoje mašinoje, ir saugončiamis informaciją apie galimus adresatus ir kaip juos pasiekti. Tokias lenteles turi tiek hostai, tiek maršrutizatoriai. Kad nereiktų saugoti visų galimų adresatų, saugoma tik IP adreso tinklo dalis, nes ji nusako visus tame tinkle esančius galimus adresatus.

Kitas principas, padedantis palaikyti nedideles lenteles yra default router opcija. Tokiu atveju, jei lentelėje nenurodytas konkretus kelias link gavėjo, naudojamas default maršrutizatorius.

Adresas, kuriuo datagrama siunčiama, vadinamas sekančio žingsnio (next - hop) adresu. Tačiau niekur datagramoje jis nėra saugomas – ten saugomas tik galutinis adresatas. Kai maršrutizavimo algoritmas grąžina sekantį IP adresą, kuriuo reikia išsiųsti, tas adresas ir datagrama perduodama tinklo interfeiso įrangai, atsakingai už perdavimą lokaliame fiziniame tinkle. Ten yra išrišamas fizinis adresas, kuris ir naudojamas. Kodėl tada lentelėse nesaugomi fiziniai adresai? 1) Kad būtų švariai atskirta tinklo įranga, kuri perduoda datagramas, ir aukštesnio lygio įranga, kuri manipuliuoja maršrutais. Be to, jei lentelėse būtų naudojami fiziniai adresai, jos būtų sunkiai skaitomos administratoriams. 2) Vienas pagrindinių IP tikslų yra sukurti abstrakciją, kuri paslepia žemiau esančių fizinių tinklų realizacijos detales.

12. Gaunamų datagramų apdorojimas

Kai hostas gauna IP datagramą, jis tikrina, ar tai jam skirti duomenys. Jei taip, datagrama perduodama aukštesnio lygio protokolams, jei ne, atmetama (discard). Hostams nepriklauso persiųsti datagramų, tai atlieka maršrutizatoriai. Pirmiausia vyksta patikrinimas, ar datagrama neatvyko pas galutinį adresatą. Jei neatvyko, naudojamas algoritmas bei lentelė nustatyti, kur toliau siųsti datagramą. Datagramos yra pasiekusios galutinį adresatą, kai adresatas datagramoje atitinka bent vieną iš kompiuterio turimų tinklo interfeisų adresų. Jei ta datagrama buvo broadcastinta lokaliame tinkle, adresatas yra broadcast adresas tam tinklui. Iškyla dar sudėtingesni patikrinimai multicastų atvejų. Jei datagrama neatitinka nė vieno adreso, IP sumažina time-to-live lauko reikšmę, paskaičiuoja naują checksum, ir išsiunčia datagramą (jei time to live lauko reikšmė ją sumažinus pasidarė lygi 0, datagrama atmetama).

Hostai negali persiuntinėti datagramų:1) Jei hostas gauna datagramą skirta ne jam, reiškia kažkur įvyko klaida ir pats hostas labai sunkiai ją aptiks jei bandys datagramą kažkur persiųsti. 2) Papildomas persiuntimas bereikalingai užims tinklą (be to, ir siuntėjo CPU laiką kitiems vartotojams). 3) Paprastos klaidos gali sukelti chaosą. Pvz.:, jei datagrama, skirta konkrečiam hostui, buvo per klaidą broadcastinta, tai jei visi hostai imsis ją persiųsti adresatui, jie užkimš tinklą ir gavėją. 4) Maršrutizatoriai, skirtingai nei hostai, ne tik persiunčia datagramas. Jie naudoja specialius protokolus raportuoti apie klaidas, palaikyti konsistentiškas lenteles ir pan. Jei hostai tik persiuntinėtų datagramas, neatlikdami visų funkcijų, kiltų įvairių anomalijų.

13. DICMP protokolas

Operacijos internete yra glaudžiai kontroliuojamos maršrutizatorių. Kai atsitinka kažkas netikėto, apie tai praneša ICMP (Internet Control Message Protocol) protokolas, kuris taip pat yra naudojamas internetui testuoti. Klaidos pranešimas siunčiamas siuntėjui (pradiniam, ne tarpiniams). Kiekvienas ICMP pranešimas yra enkapsuliuojamas IP pakete. Atraportuojamos tik klaidų sąlygos. Nenurodoma, kokių veiksmų turi būti imtasi klaidų taisymui. Pats IP protokolas ignoruoja visas klaidas, todėl ICMP – būtina IP dalis. Galimos klaidos (jų sprendimai): neteisingi, iškraipyti bitai (header laukų kontrolinė suma), neteisingi adresai (maršrutizavimo lentelės), maršrutizavimo ciklai (Time-To-Live (TTL) laukas), fragmentų praradimas (Time-out). ICMP klaidų pranešimų pavyzdžiai: echo (užklausa, ar mašina „gyva“), echo rep („Taip, aš gyva“), destination unreachable (paketas negali būti pristatytas), source quench, time exceeded (TTL = 0), redirect.

Echo ir Echo rep nėra klaidos. Tai pranešimai, naudojami patikrinti, ar adresatas yra pasiekiamas. Nusiunčiame adresatui Echo pranešimą, naudodami ping programą, o jis mums atsiunčia Echo rep per ICMP protokolą.

Destination unreachable pranešimas siunčiamas tada, kai adresato tinklas, hostas arba protokolo portas yra nepasiekiami. Source quench pranešimas siunčiamas hostui, kuris atsiuntė per daug paketų. Pranešimą siunčia maršrutizatorius, gavęs tuos paketus, bandydamas „apraminti“ hostą.

Time exceeded pranešimą gali siųsti tiek maršrutizatorius, tiek ir hostas. Maršrutizatorius siunčia tada, kai paketas yra atmestas dėl TTL = 0 (kas yra simptomas, kad paketas kažkur užsiciklino), tačiau nesitiki persiuntimo dar kartą. Hostas siunčia negavęs bent vieno fragmento.

Redirect pranešimą siunčia maršrutizatorius, nusiuntęs hostui paketą ir gavęs iš hosto pranešimą, jog šis paketas skirtas ne šiam tinklui, t.y. kažkur blogai nukreiptas. Tokiu atveju paketas atmetamas.

ICMP pranešimų persiuntimas Siunčiama šaltiniui, pranešimai pernešami enkapsuliuoti į IP datagramas. Gaunama dviguba enkapsuliacija. Nes freimą sudaro hederis ir duomenys,

freimo duomenis sudaro IP hederis ir IP duomenys, o IP duomenis sudaro ICMP hederis ir ICMP duomenys. Kad būtų išvengta kaskadinių klaidų pranešimų apie pačius ICMP klaidų pranešimus, nepasisekus perduoti ICMP, klaidos negeneruojamos.

14. IPv6. Naujos versijos tikslai, naujovės. Adresai. IPv6 datagramų antraščių (headers) principai. Pagrindinės antraštės laukai

IPv4 adresų aibė jau praktiškai išeikvota. Tinklų skirstymas į A, B, C klases neatitinka šiandieninių poreikių ir sąlygoja adresų aibės išeikvojimo problemą. Maršrutizatorių lentelės auga nevaldomai, kadangi jose turi būti informacija apie kiekvieną tinklą, taigi dviejų lygių adresų hierarchija yra nepakankama. IPv6 tikslai: praktiškai neišsemiama adresų aibė; mažos maršrutizavimo lentelės; paprastesnis protokolas ir paketų formatas greitesniam jų apdorojimui; security fyčersai; rimčiau žiūrima į serviso tipą (ypač real time data); protokolo praplečiamumas; suderinamumas su IPv4. IPv6 tikslų realizacija: adresai 128 bitų; paprastas bazinis fiksuoto ilgio headeris (papildomos savybės, pvz. didelės datagramos, norimo routinimo nurodymas, realizuojomas papildomais headeriais); integruota security. IPv6 adresai: yra 2128 ≈ 3,4*1038 adresų (visam žemės paviršiui būtų po 7*1023 adresų kvadratiniame metre). Rašomi kaip 32 šešioliktainiai skaičiai, sugrupuoti po 4 į 8 grupes ir atskirti „:“. Grupės, kuriose yra vien tik nuliai, sutraukiamos iki „::“. Pagrindinė IPv6 paketo antraštė yra fiksuoto ilgio su tokiais laukais:versija (=6); prioritetas (0-15, 0-7 – paketą galima užlaikyti, 8-15 – paketas skubus); flow label – naudojamas žymėti IP pseudosusijungimui, t.y. grupė paketų su vienu flow label bus perduodami pagal tuos pačius susitarimus; payload ilgis – duomenų ilgis; sekantis headeris – nusako, kas seka už bazinio headerio: ar vienas iš papildomų headerių, ar TCP/UDP paketas; time to live; source adresas; dest adresas.

15. Transporto lygis, jo pateikiami servisai.

Kertinis lygis, naudodamasis tinklo lygio servisais, pateikia efektyvų ir patikimą servisą aplikacijų lygiui. Kaip ir tinklo lygyje, yra du serviso tipai: connection-oriented ir connectionless. Abu labai panašūs į atitinkamus tinklo lygio servisus, taip pat panašūs connection-oriented serviso adresavimas ir srauto kontrolė. Connection-oriented servise jungtys pereina tris etapus: susijungimo, duomenų perdavimo, atsijungimo. Nors transporto lygis ir panašus į tinklo lygį, tačiau visgi yra reikalingas atskiras lygis, kad 1) paslėptų tinklo lygio problemas, pvz. patikimumo stokos atveju tą kiek įmanoma kompensuotų. ir pan. 2) būtų bendras interfeisas virš potencialiai daug skirtingų tinklo lygio interfeisų. 3) adresuotų ne kompiuterius, o konkrečias programas (ne „machine-to-machine“, bet „end-to-end“ adresavimas)

16. Programų adresavimovproblema, portai.

TCP/IP atveju tinklo lygio adresas yra IP adresas, kuris adresuoja visą kompiuterį ir nėra kaip išskirti konkrečių programų jame. Taip pat negalima naudoti kažko, kas būtų pririšta prie konkrečios sistemos. Todėl TCP/IP transporto lygis įveda „porto“ abstrakciją. Portas kartu su IP adresu ir transporto protokolo tipu (TCP ar UDP) vienareikšmiškai identifikuoja konkrečią programą tinkle. Kiekvienai norinčiai programai priskiriamas unikalus skaičius nuo 1 iki 65536. Serveriams duodami rezervuoti portų numeriai iki 1024, klientai gauna didelius numerius. Pvz. DNS serveriui rezervuotas 53 portas, o jo klientui prisijungimo metu duodamas laisvai pasirinktas 35412 portas.

17. UDP protokolas

Transporto lygio protokolas, pateikiantis nepatikimą, orientuotą į pranešimus, connectionless duomenų perdavimą. Tai „lengvas“ protokolas, jam reikia mažai papildomų tinklo ir CPU resursų. Išsiunčiamiem duomenim prideda UDP headerį, jame yra source port, destination port, pranešimo ilgis bei checksumas.

18. TCP protokolas. Savybės, patikimo duomenų perdavimo užtikrinimo būdai . Ryšio užmezgimo ir nutraukimo principas.

Transporto lygio protokolas, pateikiantis patikimą (nėra jokio duomenų pasikartojimo, praradimo ar iškraipymo), connection-oriented, streamini, point-to-point protokolą. Kitos sąvybės: full-duplex; streamas suskaidomas į atskirus segmentus, kurių kiekvienas enkapsuliuojamas į IP paketą perdavimui.

Srauto kontrolė vykdoma TCP lango pagalba – ACK pranešimuose būna įrašytas lango dydis, t.y. kiek dar galima siųsti duomenų. Kai langas lygus 0, siuntėjas turi laukti.

Patikimas duomenų perdavimas užtikrinamas: 1) patikimu ryšio užmezgimu – klientas siunčia FIN+ACK, serveris atsako FIN+ACK, klientas atsako ACK; (FIN ir ACK – tai TCP headerio bitai) 2) patikimu duomenų perdavimu – gavėjas, gavęs duomenis, siunčia patvirtinimą (acknowledgment). Jei siuntėjas per tam tikrą laiką patvirtinimo negauna, persiunčia duomenis. Konkretus laukimo laikas priklauso nuo atstumo iki gavėjo ir transporto sąlygų. TCP naudoja adaptive retransmission, kuris nustato timeoutą pagal tai, kiek užtrukdavo ankstesni tos sesijos paketai; 3) grakščiu ryšio nutraukimu – nutraukinėtojas siunčia FIN, pareina jam FIN + ACK, siunčia ACK. Ryšio užmezgimo ir nutraukimo principas aprašytas 1) ir 3) punktuose aukščiau.

19. TCP bei UDP duomenų validavimas.

Skaičiuojama kontrolinė suma iš TCP/UDP headerio, duomenų ir pseudoheaderio, kuriame yra source ir dest adresai, protokolo versija (= 6), bei pranešimo ilgis. Skaičiavimo algoritmas: iš pradžių headerio checksumas := 0, jei duomenų yra nelyginis skaičius baitų, pridedamas dar vienas nulinis baitas, tada sudedama viskas kaip 16 bitų žodžiai vieneto papildymo formoje. Paimamas sumos vieneto papildymas, ir įrašomas į headerio checksumą. Vėliau skaičiuojant tokio viso paketo checksumą, turi gautis 0. („Vieneto papildymas“ duotam skaičiui yra visų jo bitų apvertimas).

20. Socket’ai. Jų tipai, pagrindinės operacijos. Programų tarpusavio bendravimo per socket’us schema.

Socketas – tai TCP/IP transporto lygio „end-point‘o“ abstrakcija. Socketų rūšys atitinka skirtingus transporto lygio siūlomus servisus: yra connectionless datagram (UDP) socketai ir connection-oriented stream (TCP) socketai. Pagrindinės operacijos: socket – sukuria naują socketą; bind – susieja lokalų adresą su socketu; listen – TCP socketo paruošimas priimti susijungimus; accept – TCP socketo prisijungimo priėmimas; connect – TCP ryšio inicijavimas, UDP parametrų nustatymas; send, receive – duomenų siuntimas/priėmimas; shutdown – ryšio nutraukimas, close – socketo atlaisvinimas.

Programų bendravimo schema UDP atveju: serveris: socket -> bind -> listen -> recvfrom -> ... -> shutdown -> close. Klientas: socket -> connect -> sendto -> ... -> shutdown -> close. TCP atveju serveris: socket -> bind -> listen -> accept -> recv -> ... -> shutdown -> close. Klientas: socket -> connect -> send -> ... -> shutdown -> close.

21. OSI kanalinis lygis. Projektavimo klausimai.

Kanalo lygis (data link layer) yra tarp tinklo ir fizinio lygio. Jis atsakingas už duomenų perdavimą tarp dviejų hostų tarp jų esančiu „laido tipo“ kanalu. „Laido tipo“ reiškia, kad visi bitai, išsiųsti vieno hosto, ateina į kitą tą pačia tvarka. Taigi kanalo lygis atsako už perdavimo klaidų apdorojimą, duomenų srauto reguliavimą ir aiškaus interfeiso pateikimą tinklo lygiui. Tinklo lygio paketai čia yra enkapsuliojami į freimus. Teikiami servisai gali būti kelių rūšių: nepatvirtinantys connectionless, patvirtinantys connectionless, patvirtinantys connection-oriented. Freimai: kaip suskirstyti bitų srautą į juos? 1) daryti pauzes tarp freimų (visai nepatikima) 2) prieš kiekvieną freimą siųsti jo ilgį (klaidos atveju labai sunku tęsti darbą) 3) flag baitai su „byte stuffinimu“ (jei flag baitas pasitaiko normaliuose duomenyse, prieš jį įterpiamas specialus ESC baitas, o jei pasitaiko pastarasis – įterpiamas dar vienas ESC, ir t.t.). arba flag bitai (tas pats, tik bitų lygmenyje) 4) metodų kombinacija (ilgis + stufinimas). Klaidų kontrolė vykdoma patvirtinimais; timeoutais, (jei per tam tikrą laiką neatėjo patvirtinimas, freimas persiųnčiamas); kad freimai nesusipainiotų, ypač jei dar jie persiuntinėjami, tai prie kiekvieno prikabinamas sequence numeris; pačių freimų klaidos aptinkamos arba

ištaisomos Hamingo kodais. Srauto kontrolė vykdoma feedback based flow control pagalba: gavėjas praneša siuntėjui, kiek dar duomenų galima siųsti artimiausiu metu.

22. MAC polygis, jo sprendžiamos problemos. FDM, TDM. Protokolai (pure ALOHA, slotted ALOHA, CSMA, CSMA/CD(su collosion detection), protokolai be kolizijų, riboto varžymosi protokolai, adaptive tree walk protokolas).

MAC(Medium Access Control) polygio protokolai yra atsakingi už sprendimą, kas siųs duomenis, o kas lauks. Tai žemiausia datalink lygio dalis. Transliavimo laikas gali būti skirstomas statiškai ir dinamiškai.

Statiškai yra skirstoma pagal FDM (Frequency Division Multiplexing) arba TDM(Time Division Multiplexing). Pirmuoju atveju bus skirstomas kanalo plotis (pvz. bendras plotis 100Mbps bus išskirta tik 10) antruoju naudojimosi kanalu laikas. Trūkumai: išskyrimas statiškas, todėl netinka kintančiam vartotojų kiekiui. Kompiuterių tinkluose duomenys yra labai pliūpsniniai (1:1000), todėl didžiąją laiko dalį stotis nenaudos turimo resurso ir taip labai lėtins bendrą tinklo darbą

Sąlygos dinaminiam kanalo paskirstymui:

1.Stoties modelis – egzistuoja N nepriklausomų stočių, kiekviena turi programą(vartotoją), kuri generuoja duomenų (“freimus”). Pasiuntusi informaciją stotis blokuojama iki sėkmingo perdavimo.

2.Ryšio kanalas – egzistuoja vienintelis kanalas. Aparatūriškai visos stotys yra lygiateisės juo naudotis, galimi programiniai prioritetai

3.Susidūrimai (collisions) – jei du iš skirtingų stočių pasiųsti freimai nors truputį persikera kanale, jie abu sugadinami. Kiekviena stotis pati gali aptikti susidūrimus. Sugadintas freimas turi būti pakartotas. Kitokių klaidų nėra.

4.Laikas – 1) tolydus 2) diskretus (slotted) – kiekviekas slotas gali tureti 0, 1, >1 freimus – atitinkamai gali būti tuščias, sėkmingas, arba įvykti susidūrimas.

5.Nustatymas ar kanalas užimtas (Carrier Sense) 1) su nustatymu 2) be nustatymo

Pure ALOHA – laikas ištisinis be cs. Siunčia duomenis bet kada. Tinkamas jei tinklo užimtumas mažas. Po kolizijos laukia rand laiko tarpą ir siunčia iš naujo. Efektyvumas būtų S = Ge-2G, max su G=0.5 ir S=1/2e ~ 0.184 Slotted Aloha – laikas slotted be cs. Siunčia duomenis tik prasidėjus naujam slot’ui taip išvengiama kolizijos iš pure ALOHA, kurios gadindavo tris freimus

(„;;;”;;;„) - dvigubai trumpesnis galimo freimo pažeidimo intervalas. S = Ge-G, G=1 ir S=1/e ~ 0.368

CSMA (Carrier Sense Multiple Access) šeima: su cs, kai užimamas kanalas kolizijų nebus, tačiau kolizijos gali vykti rungomsi dėl kanalo(contention) periodu

- 1-persistent protokolas – siunčia iš karto, kai aptinka laisvą kanalą, tikrina visą laiką, čia kyla problema, jei stotys pasiruošia siųsti siunčiant kitai, atsilaisvinus kanalui siųs abi – gausim koliziją.
- nonpersistent CSMA – jei randa užimtą kanala – prieš tikrindama vėl palaukia atsitiktinį laiko tarpą ir tada vėl kartoja algoritmą iki kanalas laisvas. Kolizija gali atsirasti tik dėl signalo užlaikymo sklindant laiko. Minusas esant net laisvam tinklui signalas užlaikomas prieš išsiunčiant.
- p-persistent CSMA – jei randa laisvą kanalą – siunčia su tikimybe p, arba laukia kito laiko sloto su tikimybe q=1-p.
- CSMA/CD (Collision Detection). Įvedamas patobulinima – nutraukti siuntimą aptikus freimų susidūrimą. Taip sutaupomas kanalo resursas. Nutraukus siuntimą palaukiama atsitikrinį laiko tarpą, tada vel tikrinamas kanalas. Stotis, turi galėti ir klausyti ir siųsti vienu metu, kas gali būti apribojimas sistemai. Kolizija gali būti nustatoma po 2t laiko, kur t – laikas, reikalingas signalui nueiti tarp dviejų toliausių stočių .Pvz 1 km koaksialinio kabelio t = 5 milisekundes.

Protokolai be kolizijų - turi būti žinomas stočių sk. Be cs, su slot’ais A Bit-Map Protocol turi spec varžymosi laiko tarpas, kurio metu stotys pareiškia norą siųsti, po to užsiėmusios iš eilės siunčia paprastai stotims su mažesniais numeriais tenka laukti ilgiau. Efektyvumas prie didelio apkrovimo - d/(1+d), Vid d/(N+d) kur d – freimo ilgis

Binary Countdown metodas – stotims priskiriami vienodo ilgio binariniai adresai. Joms varžantis paskirtu metu su adresais atliekama OR operacija. Stotis pasitraukia, jei vietoje jos nuliuko įrašomas vienetas. Po rungimosi siunčiamas tik 1 freimas. Ir vėl rungiamasi. Stotys su aukštesniu numeriu turės prioritetą prieš žemesnio.

- Riboto varžymosi Stotys gali būti paskirstomos į grupes, taip kad tik stotys priklausančios 0 grupei varžosi dėl nulinio sloto. Jei kam nors pasiseka – tai freimas perduodamas, jei ne – pereinama prie sekančios grupės.
- Pagrindinė problema – kaip paskirstyti stotis. Jei jos paskirstomos po vieną į grupę – gaunamas bitmap protokolas, jei visos į vieną grupe – slotted ALOHA
- Reikalinga dinamika: kai apkrovimas mažas, tai grupę sudaro daug stočių, ir atvirkščiai.
- Prisitaikančio medžio algoritmas (Adaptive Tree Walk protocol)
- Del sloto 0 varžosi visos.Jei pasiseka – OK
- Jei ne, dėl sloto 1 varžosi pomedis 2. Jei pasiseka kam nors perduoti, tai stotys iš pomedžio 3 varžosi dėl 2, jei ne, varžosi pomedis 4.
- Principas – jei tyla, ar sėkmingas perdavimas – toliau gilyn algoritmas neina – visos norinčios stotys jau ten surastos.

Pradinio sloto klausimas – nuo kuio lygio pradėti ieškoti.

23. MAC polygio protokolai naudojami su bevielėmis komunikacijomis

- Portable nėra mobile. Pasiekti mobilumą komunikavimui naudojamos radio ar infraraudonosios bangos.
- Paprasti CSMA protokolai netinka – yra problemos:

– Paslėptos stoties (hidden station) problema (pav a) A nemato C todėl gali būti kolizija jiems siunčiant kartu.

– Išstatytos stoties (exposed station) problema (pav b) C jaučia užimtumą ir nesiunčia D, nors iš tikrųjų jis B ir A netrukdys.

- Jei sistema naudoja trumpąsias bangas, vienu metu gali vykti keli perdavimai, jei jie nėra vienas kito pasiekimo zonoje.

Protokolai MACA ir MACAW.

- MACA (Multiple Access with Collision Avoidance) – 1990 m.
- Pagrindinė idėja – stimuliuoti gavėją išsiunčiant trumpą freimą, kad tai galėtų pastebėti aplinkui esančios stotys, ir susilaikytų nuo duomenų perdavimo per numatomą ilgo duomenų freimo siuntimą.
- Apsikeičiama trumpais freimais – RTS (Request To Send) ir CTS (Clear To Send). Jie praneša informaciją duomenų freimo ilgį. Visa aplinka sulaikoma nuo siuntimo siunčiančiajai ar priimančiajai stočiai.
- Kolizijų vis tiek gali įvykti – pavyzdžiui gali dingti RTS freimai. Tada siuntėjas po atsitiktinio laiko tarpo bando pakartoti
- MACAW – pagerinta versija 1994 m. (įvedė acknowledgment freimus, kanalo užimtumo nustatymą – carrier sensing, mechanizmą apsikeisti informacijai apie kanalo užsikimšimą tarp stočių, etc.)

24. IEEE standartai 802.3 – CSM/CD (Etherne), Mančesterio kodai, 802,4 – token bus, 802,5 – token ring.

- IEEE Standard 802.3 – tai 1-persistent CMSA/CD lokalus tinklas(LAN) Naudojami sujungimo būdai
- Galimi 6 baitų MAC adresai
- Multicast galimybė (aukščiausias bitas - 1)
- Jei adrese visi bitai yra 1 – tai broadcast, freimas pristatomas visoms tinkle esančioms mašinoms.

- Globalaus adreso unikalumas(46-as bitas žymi, ar tai lokalus ar globalus adresas) – iš 46 bitų galima sudaryti apie 7x1013 adresų.
- Ilgis – geri freimai turi būti bent 64 baitų ilgio. Trumpi freimai yra papildomi(padded), kad pasiektų tokį ilgį.

802.4 - token bus

- Taigi 802.4 realizuojama yra šitaip:

- Logiškai stotys sudaro žiedą. Fizinė topologija gali būti tiesinė ar medžio formos

- Kiekviena stotis turi numerį

- Kiekviena stotis žino savo kaimynų numerius

- Pirmą freimą siunčia aukščiausio numerio stotis

- Ji paskui perduoda leidimą siusti – markerį (token) – kaimynei. Perduoti duomenis gali tik stotis turinti markerį.

- 802.4 palaikomos prioritetų klasės

- 802.4 MAC protokolas yra sudėtingas (turi būti galimybės stotį įtraukti ir išmesti iš žiedo)

- Fiziniam lygiui 802.4 naudoja koaksialinį kabelį.

802.5 - token ring

- Žiediniai tinklai

- Tiesiog rinkinys nuo-iki (point-to-point) ryšių, kurie sudaro žiedą.

- Beveik visiškai skaitmeniniai

- Garantuotas maksimalus laukimo laikas

- Token ring tinkle žiedu juda specialus bitų rinkinys – markeris (token), kas turi markerį gali siųsti duomenis
- Stotis, norinti perduoti duomenis, pagriebia markerį ir pašalina jį iš žiedo
- Siųsti duomenis gali tik stotis, pas kurią tuo metu yra markeris

Mančesterio kodai

Tinklai negali perduoti informacijos binariniu kodavimu 0;5V (0 neskiriamas nuo laisvos linijos)o problema su –1 ir 1 yra sinchronizavimo pametimas esant ilgoms 1 ar 0 sekoms,

nesikreipiant į vidinį laikrodį. Manč. kod. aukštas įtampos signalas yra +0.85 voltų, žemas - -0.85 voltų. 1 vaizduojamas aukšta įtampa pradžioj žema pabaigojperdavimo periodo 0 atvirksčiai. Naudojmas Ethernet tinkluose. Differencialinis Manč kod. 1 kai nėrs perėjimo tarpe tarp periodų 0 kai yra naudojamas pvz.802.5 token ring. Vienintelis minusas- sk perduoti reikia 2 ciklų

25. Binary exponentional backoff algoritmas.

- Laiko randomizavimo algoritmas – nusako, kurį laiką stotis laukia, jei įvyksta kolizija.
- Po pirmos kolizijos laukiama 0 arba 1 slotą, po antros – 0,1,2,3
- Po i-tosios laukiama pasirinktą skaičių slotų iš intervalo 0 …2i-1.
- Po dešimties sustojama ties 1023 maksimumu.
- Po 16 – pasiduodama, ir klaida perduodama į aukštesnį lygį.

26. Switcho veikimo principas.

Kadangi kanalo pralaidumo didinimas yra brangus, stengiamasi mažinti kolizijų skaičių ir taip užtikrinti didesnį perdavimo greitį ir mažinti kolizijų sritis t.y. sumažinti stočių skaičių besinaudojančių vienu fiziniu kanalu, todėl svarbu mažinti besivaržančių stočių skaičių. Tai daroma Switcho pagalba:

- Switch - didelio greičio Gbps įrenginys su plug-in kortomis, kurios sudaro atskirą kolizijų sritį. Taip didelis tinkles sudalinamas į daug nedidelių dalių, kuriose kolizijos sprendžiamos lokaliai ir neturi įtakos visam tinklui.
- Korta pas save viduje tikrina ar freimas turi būti persiunčiamas kitur, ar jo gavėjas yra čia pat
- Į išorę perduodami tik “svetimi” freimai – skirti kitoje plug-in kortoje esančiam gavėjui

Vienas iš optimizavimo būdų – buferizuoti portai, turintys RAM ir dėl to galintys palaikyt net kelis perdavimus vienoje plug-in kortoje. Dėl Switch’ų nemažos kainos prie jų galima jungti ne vieną stotį, o kelias stotis sujungtas į tinklą pigesniu hub’u, taip kai kurie portai gali būti naudojami kaip koncentratoriai.

27. Tinklų standartų 802.3 802.4, 802.5 palyginimas – pranašumai, trūkumai.

- 802.3 trūkumai

- Analoginė komponentė perdavimo technologijoje (Carrier Sense/Collision Detection)

- Minimalus freimo ilgis 64 baitai

- Nedeterministinis ir neturi prioritetų

- Kabelis iki 2.5 km

Konfliktai apkrautame tinkle

- 802.3 pranašumai

- paplitęs

- paprastas algoritmas

- Stotys instaliuojamos nenutraukiant darbo

- Nėra užlaikymo

- 802.4 trūkumai

- Analoginės sistemos – modemai, stiprintuvai

- Sudėtingas protokolas

- Didelis užlaikymas prie mažo apkrovimo

- Netinkamas optiniams kabeliams

- 802.4 pranašumai

- Deterministinis, nors markerio praradimas gali kenkti

- Palaiko prioritetus

- Geras pralaidumas apkrautam tinkle

- Patikimos perdavimo priemonės

- optiniams kabeliams

• 802.5 pranašumai

- Nuo-iki (point-to-point) jungtys

- Skaitmeninė technika

- Efektyvus prie didelio apkrovimo

- Galimi prioritetai

- Kintamas freimų dydis (ilgi freimai)

- Didelis fizinių perdavimo priemonių pasirinkimas

• 802.5 trūkumai

- Loginis žiedo palaikymas – centralizuotas monitorius

- Užlaikymas prie nedidelio apkrovimo

28. LLC – Logical Link Control 802.2.Bridges

LLC. Parteikia vienodą interfeisą tinklo lygiui virš visų 802.x LAN. Serviso tipai: nepatikimas datagramų, patikimas, su patvirtinimais datagramų servisas, patikimas connection-oriented. Kanalinį lygį sudaro MAC ir LLC.

Bridge. Priežastys naudoti: sujungimas keleto skirtingų LAN, nutolusių LAN, apkrovimo mažinimas skaidant tinklą į potinklius, lokalaus tinklo ploto praplėtimas, didinimas lokalaus tinklo patikimumo, saugumo gerinimas. Bridge'ai routina naudodami kanalo sluoksnio adresus (o ne paketuose nurodytus adresus, kaip daro routeriai).

29. Tinklo lygis. Pagrindinės funkcijos, serviso transporto lygiui peteikimas, vidinės potinklio struktūros variantai

Tinklo lygis rūpinasi paketų persiuntimu iš šaltinio galutiniam adresatui. Siuntimas adresatui gali reikalauti daug žingsnų per tarpinius routerius (tuo tarpu kanalinis lygis rūpinasi tik perduoti freimus iš vieno laido galo į kitą). Kad įgyvendintų savo tikslus, tinklo lygis turi žinoti potinklio (subnet) topologiją (visų maršrutizatorių aibę) ir parinkti atitinkamus kelius. Rūpinasi tinklo ir routerių apkrovimu.

Serviso transporto lygiui pateikimas. Servisai turi būti nepriklausomi nuo potinklio technologijos. Transporto lygis neturi rūpintis esančių potinklių skaičiumi, tipais bei tipologijomis. Tinklo adresai, kuriais naudojasi transporto lygis turi būti sudaryti pagal vienodą schemą, tiek LANuose, tiek WANuose. Tinklo lygis gali teikti tiek connection-oriented servisus, tiek ir connectionless (dvi skirtingos koncepcijos).

Vidinės potinklio struktūros variantai. Virtualių grandinių VG (veikia connection-oriented principu), datagramų D – nepriklausomų paketų (connection-less). Skirtumai: VG reikia

inicializuoti, D – ne, kiekvienas D paketas turi pilnus adresus, o VG tik trumpus identifikatorius. D atveju potinklis nelaiko informacijos, o VG – laiko vidines lenteles. D – kiekvienas paketas routinamas atskirai. VG – routinimo kelias bendras ir nustatomas inicializavimo metu. D – nesprendžia sangrūdų, VG – gali nesunkiai susitvarkyti.

30. Maršrutizavimo algoritmai. Maršrutizavimas trumpaisusiu keliu, užtvindymas, nuotolių lentelės metodas, hierarchinis maršrutizavimas.

Maršrutizavimo algoritmas – tinklo lygio programinės įrangos dalis, atsakinga už įeinančio paketo išsiuntimą tinkama išeinančia linija. Datagramų atveju geriausias maršrutas nustatomas kiekienam paketui iš naujo, virtualių grandinių atvejų – vienai sesijai (session routing). Pageidautinos savybės: korektiškumas, paprastumas, patikimumas, stabilumas, teisingumas, optimalumas. Maršrutizavimas trumpiausiu keliu. Visi maršrutizatoriai sudaro grafą. Briaunos gali turėti svorius – kainas. Ieškomas kelias su mažiausia kaina.

Užtvindymas. Kiekvienas įeinantis paketas yra išsiunčiamas visom išeinančiom linijom, išskyrus tą, per kurią atėjo. Generuoja daug pasikartojančių paketų. Juos galima apriboti pora būdų: nustatant paketo gyvavimo laiką, sekant, kurie paketai jau buvo išsiųsti ir atmetant pasikartojančius. Variacija – selective flooding – paketas siunčiamas tik tomis išeinančiomis linijomis, kurios apytiksliai eina teisinga kryptimi.

Nuotolių lentelės metodas. Kiekvienas maršrutizatorius palaiko lentelę su atstumais iki kiekvieno kito maršrutizatoriaus, bei kaip jį pasiekti. Ši informacija atnaujinama besikeičiant informacija su kaimynais.

Hierarchinis maršrutizavimas. Problema - dideliuose tinkluose didėja nuotolių lentelėms reikalingos atminties kiekis, taip pat skaičiavimai užima daugiau CPU laiko.

Hierarchiniu atveju maršrutizatoriai suskirstomi į regionus, kur kiekvienas žino regiono struktūrą, bet nežino apie kitus regionus. Bendravimui su kitais regionais paskiriami konkretūs maršrutizatoriai. Galima skirstyti į kiek nori hierarchinių lygių. Tai labai sumažina palaikomas nuotolių lenteles.

31. Maršrutizavimas mobiliems kompiuteriams

Visi vartotojai turi pastovų namų adresą, kuris nekeičiamas (home location). Kiekviena sritis turi savo agentus. Svetimi agentai vadinami foreign agents, namų – home agents. Tipinė veiksmų seka, kai naujas mobilus vartotojas atvyksta į svetimą tinklą:1) Foreign agent laikas nuo laiko broadcastina pranešimus apie savo egzistavimą ir adresą. Jei mobilus vartotojas tokio nesulaukia, jis pats gali broadcastinti užklausimą apie agentus. 2) Mobilus vartotojas užsiregistruoja pas agentą, nurodydamas jam namų adresą, taip apt

saugumo informaciją. 3) Svetimas agentas susisiekia su namų agentu, pasakydamas, kad pas jį yra vienas to tinklo vartotojas. Saugumo informacija naudojama pasitikrinimui. 4) Kai gaunamas atsakymas iš namų agento, mobilus vartotojas užregisturojamas naujame tinkle ir įtraukiamas į lenteles.

Kai mobiliam vartotojui siunčiamas paketas: 1) Paketas siunčiamas mob. vartotojo namų adresu. 2) Paketas persiunčiamas (tunneled) foreign agentui. 3) Siuntėjui duodamas foreign agento adresas. 4) Tolimesni paketai siunčiami tiesiogiai foreign agentui.

32. Persipildymo (congestion) valdymas. Leaky bucket, token bucket, load shedding, choke paketai, RSVP.

Kai per daug paketų yra siunčiama, sistema persipildo ir jos našumas ženkliai krenta. Leaky bucket. Susikaupę paketai sulaikomi, surikiuojami ir leidžiami toliau konstantiniu greičiu. Jei „kibiras“ persipildo, tai naujai atėję paketai numetami.

Token bucket. Kibire laikomi tokenai. Nauji atėję pranešimai praeina tik sunaikinę tokenus iš viedro. Likę laukia, kol bus sugeneruota daugiau tokenų. Paketai niekados neprarandami. Leidžia greičiau grupėmis išsiųsti netikėtus paketų prasiveržimus. Krūvio numetinėjimas (load shedding). Kai maršrutizatoriai perpildomi paketais, jie kai kuriuos gali tiesiog atmesti. Yra du atmetimo būdai: Wine – kai geriau palaikyti senesnį, nei naujesnį paketą (failo perdavimas), Milk – kai geriau palaikyti naujesnį, nei senesnį paketą (video perdavimas). Kad naudotų teisingą atmetimo politiką, turi padėti ir aplikacijos, nurodydamos paketo svarbumą. Užsikimšimo (choke) paketai. Maršrutizatorius gali išsiųsti choke paketą, kai nebesusidoroja su ateinančiais duomenimis. Siuntėjas, gavęs choke paketą, privalo sumažinti duomenų srautą tam maršrutizatoriui tam tikru procentu. Po kurio laiko, jei daugiau choke paketų neateina, srautas gali būti didinamas, bet lėčiau.

RSVP – Resource reSerVation Protocol. Vartotojai užsisako kurį nors siuntėją. Siunčiamo rezervavimo metu kiekvienas maršrutizatorius rezervuoja reikalingas linijas. Naudojami spaning tree, sudaromi pagal siuntėjus. Rezervavimo pranešimas siunčiamas atitinkamu medžiu.

33. Internetworking. Tunneling, paketų fragmentacija

Vieną tinklo lygių sprendžiamų problemų – duomenų perdavimas tarp skirtingų tinklų (TCP/IP, SNA, DECnet, Novell, ...). Internetworking būdai: Connection-oriented. Galima sujungti virtualią grandinę nuo siuntėjo iki adresato, ir visus paketus siųsti ja. Tinka, kai visi tarpiniai tinklai turi panašias savybes. Connection-less. Datagraminis modelis. Privalumai – nesvarbu ar tarpiniai tinklai palaiko kontaktinį servisą, ar ne. Problemos – paketų formatai, adresavimas.

Tunneling. Būdas išspręsti paketų nesuderinamumą. Paketas yra siučiamas sukurtu „tuneliu“. Kitaip tariant, jis yra įvelkamas į tarpinio tiklo pripažįstamą formatą (header, etc.) ir pasiučiamas. Perėjęs tarpinį tiklą jis būna išpakuojamas ir pristatomas adresatui.

Paketų fragmentacija

Sprendžia skirtingų paketų ilgių problemas. Skirtingus paketų ilgius gali sąlygoti Techninė įranga, operacinė sistema, protokolai ir pan. Jei paketas per didelis, galima išskaidyti jį į dalis, ir siųsti kaip atskirus paketus. Klausimas – kada surinkinėti paketus iš dalių – ar kiekviename tarpiniame routeryje, ar tik galutiniame hoste.