Komunikačné protokoly

Komunikačná funkcia: vymedzená činnosť, ktorá realizuje časť riadenia komunikácie

Vrstvenie (Layering):
- rozdelenie, kde každá vrstva má nejakú úlohu
- vrstvy sú radené vertikálne
- vyššia vrstva požaduje komunikačnú službu nižšej
- nižšia vrstva pridáva vždy dodatočnú informáciu
- je to vlastne princíp štrukturovania riadenia komunikácie

Reference Model Open System Intercommection (RMOSI): referenčný model z 80. rokov

Sieťová architektúra: štruktúra riadenia komunikácie

Zákl. princípy tvorby vrstiev:
- každá vrstva plní jednoznačné a špecifické funkcie
- vnútorná výstavba vrstvy je nezávislá od jej funkcií
- výmena informácií prebieha len medzi susednými
- nižšia vrstva poskytuje služby vyššej vrstve
- počet vrstiev je dosť veľký na to, aby odlišné funkcie neboli v jednej vrstve
- počet vrstiev je dosť malý na to, aby bol model ešte prehľadný

Okopčiť: autonómny systém + FLOW CONTROL (1)
WAN (2)
MPLS (3)

Štruktúra RMOSI

1) Fyzická vrstva (physical layer)
- definuje komponenty, rozhrania, médiá pre prenos, modulácie, linkové kódy
- prenosové rýchlosti, topológie, bitová synchro
- je najnižšou vrstvou
- zabezpečuje prenos informácie z bodu A do bodu B
- vysielanie a detekcia informácie v bitoch

2) Linková vrstva (link layer) ~~dosť~~ pokrývajú obe sústy
- DSC, ~~HDLC~~, FR, MPLS, ~~X.25~~ + 802.2 LLC
- ETHERNET, ~~TOKEN RING~~, ~~TOKEN BUSS~~
  802.3, 802.5, 802.4
- zabezpečuje eliminácia chýb fyzického prenosu KÓDOVANIE
  EC
- riadi prístup na médium MAC
- adresovanie, flow control, framing

3) Sieťová vrstva (network layer)
- rieši sieťové aspekty komunikácie IP
- adresovanie, smerovanie

4) Transportná vrstva (transport layer)
- má na starosti prenos správ medzi koncovými bodmi
- riadenie chybovosti (error control) TCP, UDP

5) Relačná vrstva (session layer)
- riadi dialóg dvoch koncových bodov
- zabezpečuje možnosť návratu k určitému synchronizačnému bodu v prípade nejakej chyby

6) Prezentačná vrstva (presentation layer)
- transformuje dáta podľa potreby

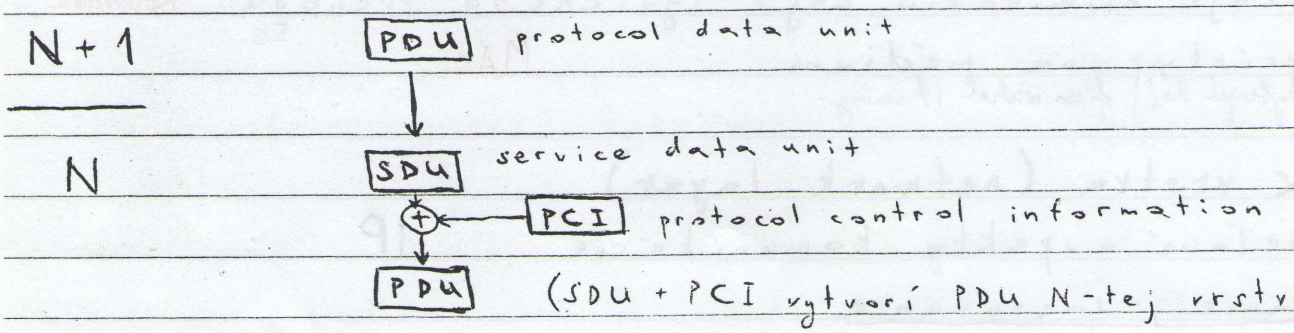
7) Aplikačná vrstva (application layer)
- sprostredkováva aplikáciu užívateľovi
  FTP, HTTP

A
P
S
T
N
DL
Ph

Komunikačný protokol: súbor pravidiel pre komunikáciu na určitej vrstve (peer-to-peer)

$KP = \{KA_1, KA_2, S, F(S)\}$ (slová)

- KP – komunikačný protokol
- KA_1, KA_2 – konečný automat
- S – slová
- $F(S)$ – formát



PDU (N) = packet // PDU sieťovej vrstvy
PDU (DL) = frame (rámec) // PDU linkovej vrstvy

a) One to one: jeden protokol vyššej vrstvy využíva jeden protokol nižšej vrstvy

b) Multiplexing (many to one): viacero vyšších vrstiev využíva protokol nižšej vrstvy

c) Splitting (one to many): inverzný prípad k multiplexingu, napr. internet cez ISDN bežiaci cez dva B kanály

Komunikácia medzi vrstvami: realizuje sa pomocou servisných primitív (request → indication → response → confirmation)

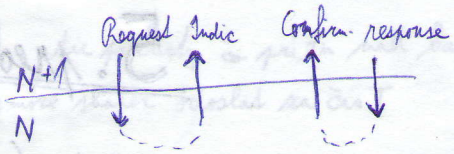
(4) ŠTRUKTÚRA TCP/IP

1) HARDWARE (physical) layer
   - physical transmissions, bit synchronization, ...
   - MODEMS
2) network interface (data link) layer - LAN, PPP
   - p2p communication
   - handling of transmissions errors
   - BRIDGES
3) Internet (network) layer
4) Transport layer (TCP, UDP)
5) aplication layer (SMTP, FTP, HTTP ..)

2. prednáška 25.9.

(segmentácia)
- fragmentácia dát ak je informácia moc veľká pre kanál (segmentation and reassembly)
- združovanie dát do väčšej protokolovo-dátovej jednotky (blocking and deblocking)

- multiplexing – viacero spojení vyššej vrstvy beží na 1 kanáli nižšej vrstvy
- inverz. multiplexing – opak (1 spojenie nižšej vrstvy beží na viacerých kanáloch vyššej vrstvy – za účelom prenosu väčšej dát)
- primitívy – typ – request, spät indication
  response, spät confirmation
  – slúžia na komunikáciu medzi vrstvami

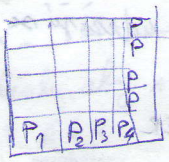


- layered communication TCP/IP architektúre (5 vrstiev)

Error control (riadenie chybovosti)

- komunikačná funkcia na linkovej úrovni (zabezpečenie)
- FEC – forward EC – dopredná kontrola dát – sú to kódy (samoopravné)
- BEC – backward EC – zabezpečenie dát so spätnou väzbou – ak obdržiaci dáta nieslo v poriadku, vyžiadať ich vyslať opäť
  – stačí použiť iba detekčný kód

- detekčné kódy – parita
  – block sum check BCC →
  – lineárne cyklické kódy



– paritné bity v stĺpcoch aj v riadkoch

ARQ – automatic repeat request

1. metóda – Send & Wait (Idle Repeat request)
- vysielač vyšle dáta, prijímač ich vyhodnotí a
  a) správne – vyšle ACK (acknowledge) – vysielač pokračuje
  b) nesprávne – nevyšle nič – vysielač čaká istý čas (timer) vyšle opäť tie isté dáta

Toto je P schéma
– positive (existujú len pozitívne overenia)

– A-schéma – prijímač vysiela opätovné vyžiadanie pokaziených dát (aj tu je timer u vysielača)

2. metóda – ISR (Ideal selective repeat) alebo (Continuous RQ)
- vysielač plynule vysiela dáta a dostáva potvrdenia; ak nedostane potvrdenie niektorého bloku, vyšle ho opäť a ukladá ich v bufferi – ak je vysoká rýchlosť vysielania dát a veľké oneskorenie, rastú nároky na buffer

3. metóda – GBN – ak nepríde potvrdenie N-tého bloku, vysielač začne vysielať opäť od N-tého bloku
- prijímač nemusí mať buffer (nemusí preusporiadavať dáta)

prípad, keď prijímač vyžaduje opätovné vyslanie poškodených dát, označujeme A-schéma (explicit request)
- aj tu však potrebujeme timer, aby sa vysielač mohol zotaviť pri poškodení potvrdzovacej správy

2) Continuous RQ - Ideal SelectiveRepeat (ISR)

- vysielač plynule posiela dáta a uchováva si ich vo vyrovnávacej pamäti, až pokým nedostane potvrdenie
- keď vysielač čaká na potvrdenie N-tého bloku a dostane potvrdenie na (N+1)-vý, pošle opätovne N-tý blok dát
- pri zvýšenej rýchlosti vysielania dát a zväčšenom oneskorení rastú nároky na buffer

3) Go-back-N (GBN)

- ak nepríde potvrdenie niektorého bloku, začne vysielač vysielať plynule od tohto bloku
- výhodou je, že prijímač nemusí mať buffer (nemusí preusporiadavať dáta)

Efektívnosť prenosu: závisí od chybovosti v kanáli, ale aj od dĺžky rámca → existuje ideálna dĺžka rámca pre danú chybovosť

NACK (N) - negatívne potvrdenie N-tého bloku sa väčšinou posiela až po prijatí (N+1)-vého bloku
ACK (N) - pozitívne potvrdenie s poradovým číslom N je v väčšine prípadov potvrdením všetkých blokov do (N-1)-vé vrátane

(7)

Hybrid ARQ: existuje niekoľko tried

1) má detekčné aj korekčné vlastnosti a ak nedokáže opraviť poškodenú správu, vysielač ju pošle celú odznova, pričom prijímač si môže pamätať všetky pokusy, alebo zlú správu vždy zahodiť

2) incremental redundancy - pri neúspešnom pokuse o opravenie pošle vysielač stále viac zabezpečovacej časti (nadbytočnosti)

Routing (smerovanie)

- hlavnou úlohou tejto komunikačnej funkcie je zabezpečiť smerovanie dát v sieti
- vyskytuje sa v sieťovej vrstve (N)

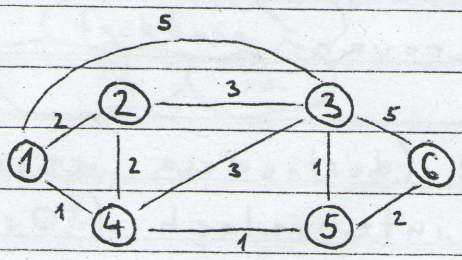
Typy konektivity v sieti:

1) connection-oriented network (ATM, MPLS)
- všetky uzly, cez ktoré dáta chodia, vedia, že medzi nimi existuje spojenie (virtuálne) - virtual circuit
- v takejto sieti musí byť signalizácia slúžiaca na vybudovanie, udržanie a zrušenie spojenia

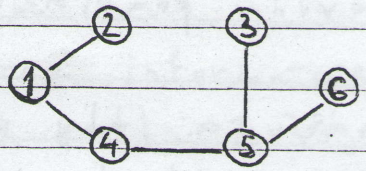
2) connection-less network (IP)
- sieť ale také nemá informáciu o existujúcom spojení
- každý paket preto musí mať cieľovú adresu, aby sa každý uzol mohol rozhodnúť kadiaľ ho pošle
- v takýchto sieťach sa paket zvykne označovať aj ako datagram

Scalability (scaling): vyjadruje schopnosť siete rozrastať sa (lepšie u sietí bez spojovej orientácie)

Quality Of Service (QOS): vyjadruje schopnosť siete garantovať dátam parametre, ktoré vyžadujú (lepšie u sietí so spojovou orientáciou)



sieť najkratších (najlacnejších) ciest z uzla č. 1:



ROUTING TABLE

	NEXT HOP	COST
1	–	0
2	2	2
3	4	3
4	4	1
5	4	2
6	4	4

Cost Matrix:

$$C = \begin{bmatrix} 0 & 2 & 5 & 1 & \infty & \infty \\ 2 & 0 & 3 & 2 & \infty & \infty \\ 5 & 3 & 0 & 3 & 1 & 5 \\ 1 & 2 & 3 & 0 & 1 & \infty \\ \infty & \infty & 1 & 1 & 0 & 2 \\ \infty & \infty & 5 & \infty & 2 & 0 \end{bmatrix}$$

Bellman's equation:

- najkratšia cesta do i je minimom súčtu optimálnej cesty do j a dĺžky medzi i a j

$$D_i = \min_j \left[D_j + c_{ij} \right]$$

Bellman - Ford - Moore Algorithm (BFMA): iterácie počítame lokálne

	1	2	3	4	5	6	Q
j = 1	(-1,0)	(1,2)	(1,5)	(1,1)	(-1,∞)	(-1,∞)	2,3,4
j = 2		(1,2)	(1,5)	(1,1)	(-1,∞)	(-1,∞)	3,4
j = 3		(1,2)	(1,5)	(1,1)	(3,6)	(3,10)	4,5,6
j = 4		(1,2)	(4,4)	(1,1)	(4,2)	(3,10)	5,6,3
j = 5		(1,2)	(5,3)	(1,1)	(4,2)	(5,4)	6,3
j = 6		(1,2)	(5,3)	(1,1)	(4,2)	(5,4)	3
j = 3		(1,2)	(5,3)	(1,1)	(4,2)	(5,4)	

- prvá položka je predchodca, druhá cena, do množiny Q idú uzly, ktorým sa zmenila v tom kroku nálepka

(9) Dijkstra - iteruj na dĺžku

Distributed BFMA:

- akoby obrátená logika - hľadáme najkratšiu cestu zo všetkých uzlov do i-tého uzla

$$D_i = \min_{j \in N(i)} (c_{ij} + D_j)$$ N(i) - susedia uzla i

Smerovací protokol: hovorí o tom, akým spôsobom si uzly vymieňajú informácie o smerovaní

(BFM)
Distance Vector: každý uzol vysiela dvojice (destinácia, cena) každému susedovi v pravidelných intervaloch (30s)

(pre DBFM)
Link - state: snaží sa, aby mal každý uzol prehľad o topológii siete a mohol si zostaviť cost matrix

Link - state routing protocols:

LSA - link state advertisement (informácia o stave siete)

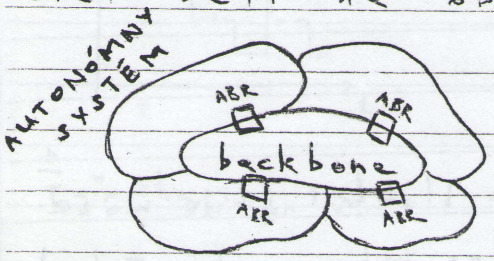
- objavenie susedov (Hello protocol)
- vybudovanie susednosti (adjacency)
- poslanie LSA do siete

Distribúcia LSA:

- flood(-ing) - záplava, najspoľahlivejší spôsob
- potvrdzovanie - posiela správu až kým nie je potvrdená

- správy sú číslované sekvenčnými číslami, aby sa dal rozlíšiť, ktorá je najaktuálnejšia
- každá správa má naviac určitú životnosť, preto router musí v polovici životnosti urobiť refresh tejto správy, aj keď nenastala žiadna zmena
- ak skončí životnosť správy, musí nanovo zbehnúť algoritmus

kompletný súbor všetkých LSA správ sa nazýva link state database a táto musí byť v každom ude synchrónna
- vo veľkých sieťach je problém s veľkou prevádzkou a nárokmi na pamäť a výpočtovú zložitosť, preto sa sieť delí na oblasti (areas)



ABR - area border router

- smerovače vo vnútri backbone oblasti fungujú na základe link-state princípu a z okolitých oblastí prijímajú od jednotlivých ABR sumárne informácie
- aby nebola prevádzka príliš veľká, vždy sa zvolí jeden designated router a ostatné prijímajú informácie od neho

Používané link-state protokoly:
OSPF - open shortest path first
IS-IS - intermediate system

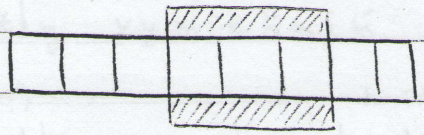
11

Flow control (riadenie toku)

smerovanie adresovanie

- ale vedia? Nalebo T? error control
- táto komunikačná funkcia zabezpečuje, aby v každom momente bolo v každom uzle siete také množstvo dát, ktoré je schopné zvládnuť

Metóda okna:



- hovorí o tom, koľko blokov naraz môže vysielač poslať bez toho, aby čakal na potvrdenie od prijímača
- nejde o ARQ - potvrdenie je povolením na ďalšie vysielanie

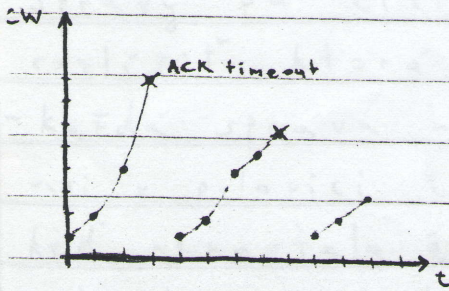
- povolenie na vysielanie (nemusí) môže byť v prijímači generované až po prijatí celého okna ⇒ pred poslaním ďalšieho bloku dát musí vysielač čakať na potvrdenie
- pri generovaní povolenia po prvej správe z bloku (okna) sa dá dosiahnuť kontinuálny prenos
- v mnohých protokoloch sa používa spojenie ARQ potvrdenia s povolením na vysielanie (piggy backing)

ide o tzv. sliding window

TCP:

- nečísluje správy, ale bajty (oktety)
- posiela okrem ACK aj hodnotu okna ⇒ TCP má dynamické okno (môže byť aj nula, čo znamená nevysielať)

ACK timeout - neprišlo potitívne potvrdenie, čo v TCP znamená, že segment sa stratil



slow start - začína od šírky okna jedna a v každom kroku ju zdvojnásobuje
- ak nastane ACK timeout, zapamätá si polovicu hodnoty (posledná dobrá), po ňu rastie exponenciálne a od nej lineárne

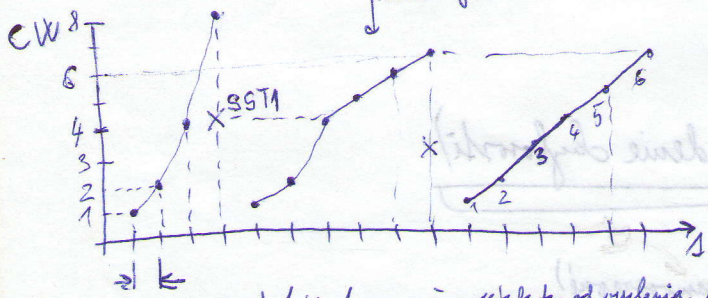
- **efektívnosť prenosu** - závisí od chybovosti na kanáli a od dĺžky rámca => existuje ideálna dĺžka rámca pre konkrétnu chybovosť

- NACK(N) - negatívne potvrdenie N-tého bloku, vyžiada sa väčšinou až po prijatí N+1 bloku
- ACK(N) - pozitívne potvrdenie s číslom N je väčšinou potvrdením všetkých predchádzajúcich blokov

12

5. prednáška - 16.10.2006

TCP metóda okna - číslujú prenášané bajty (oktety) SST - slow start threshold

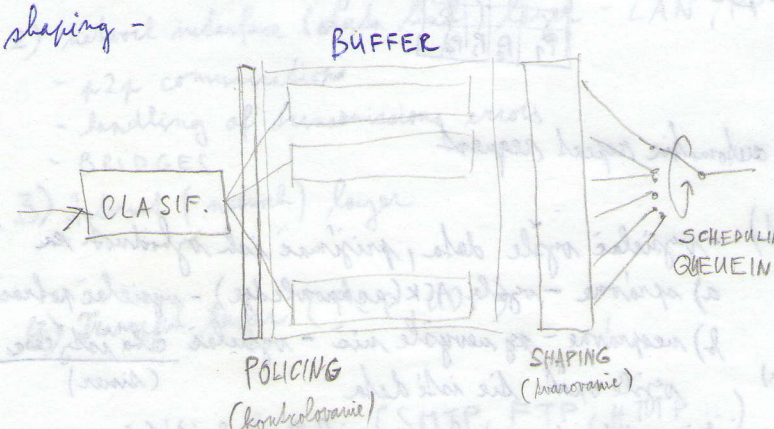


slow start - nárast okna na dvojnásobok

ack timeout - SST1 - 50% z maxima CW (congestion window)
- CW -> 1 a začína opäť slow start

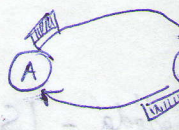
└ ak nedôjde potvrdenie o segmente z okna, čiže segment sa stratil

- klasifikátor - klasifikácia rozdelí pakety podľa dôležitosti a priorít
- buffer (vyrovnávacia pamäť) - udržiavajú sa tam pakety
- shaping -



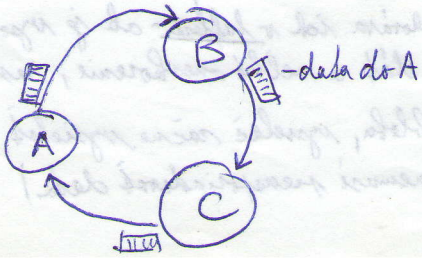
PROBLÉMY PRI ZAPLNENOM BUFFERI

direct deadlock - nastane ak je v oboch uzloch zaplnená pamäť dátami na vyžiadanie



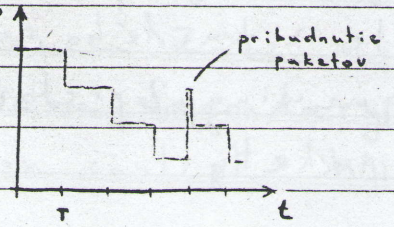
(priame uviaznutie)

indirect deadlock -



Leaky Bucket (deravé vedro):

- je to algoritmus pre shaping (tvarovanie prevádzky)



- za jednu časovú jednotku „vyteče" z vedra jeden paket
- ak do vedra príde viac paketov ako je jeho kapacita, budú zahodené

$\frac{b}{T}$ = shaping rate

Token Bucket:

- algoritmus pre policing (dohliadanie na prevádzku)
- tokeny sú akési povolenia na prenesenie a každý prenesen paket vezme z vedra zodpovedajúce množstvo tokenov
- tokeny pribúdajú vo vedre konštantnou rýchlosťou r
- keď príde paket, ktorý prekračuje kapacitu (nie je dosť tokenov vo vedre), môžeme ho zahodiť, alebo mu môžeme prideliť oranžovú / červenú farbu (čo znamená, že pri zahltení siete bude takýto paket zahodený ako prvý)

Queuing (riadenie fronty):

First In First Out (FIFO) Queuing:

- radíme pakety v takom poradí, v akom prichádzajú

Priority Queuing:

- radíme pakety do niekoľkých front podľa ich vopred určenej priority
- scheduler potom berie iba pakety z najprioritnejšieho radu a až keď tam žiadne nie sú, skočí o level nižšie a tak ďalej (tzv. striktná priorita)

Flow Based Queuing:

- vhodné, ak nemáme rozdielnu prioritu paketov, ale chce

14

aby mohli všetci useri zdielať linku „spravodlivo"
- classifier musí byť dosť inteligentný blok, ktorý pre každú prevádzku (napr. ftp download) vytvorí jeden rad
- scheduler potom berie po jednom pakete z každého radu
- nájde ale ešte o úplne spravodlivý queuing, pretože v každej fronte môžu byť rôzne dlhé pakety

Fair Queuing:
- použijeme Bit-by-bit Round-robin, teda z každého radu vezmeme vždy jeden bit
- takto sa ale dáta posielať nedajú, preto počítame pre každý paket jeho finish time, čo je čas, za ktorý by sa paket dostal celý, keby sme ho na otvej posielali bit za bitom
- pakety potom posielame od najnižšieho finish timeu po najvyšší
- pokiaľ máme priority paketov, realizujeme ich tak, že za jednu jednotku času zarátame v tom rade väčší počet bitov (označujeme Weighted Fair Queuing)

- praktická realizácia je väčšinou taká, že jeden rad. má striktnú prioritu (napr. hlasová prevádzka) a na ostatné aplikujem napr. Fair Queuing

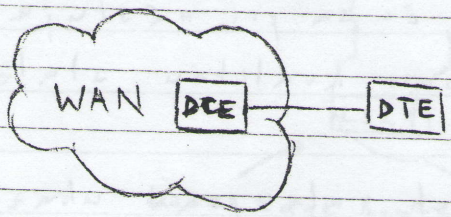
Link Layer

Binary Synchronous Communication (BSC):
- znakovo orientovaný protokol pracujúci v half-duplex
- definoval status master alebo slave (master riad. komunikáciu buď point-to-point, alebo point-to-multipoint
- používa Send & Wait ARQ
- módy: select; poll

igh Level Data Link Control (HDLC):
- bitovo orientovaný protokol
- môže pracovať v troch módoch - NRM, ARM a ABM
- stanica môže mať status Primary (posiela commands) alebo Secondary (posiela responses)
- flow control - Window, error control - GBN a SREJ

WAN - Wide Area Network

X25:



- X25 je prvý protokol, ktorý definoval rozhranie pre prístup do WAN
- definuje zariadenia DCE a DTE

DCE - Data Circuit Terminating Equipment
DTE - Data Terminating Equipment

- zariadenia pracujú na troch vrstvách - Physical, Data Link a Packet (táto je podobná sieťovej vrstve)

virtual circuit ⟨ permanent (PVC)
switched (SVC)

- permanentný virtuálny okruh sa naperno zapíše do uzlov, cez ktoré ide spojenie (niečo ako prenajatá linka)
- prepájaný virtuálny okruh na začiatku vytvorí spojenie a na konci spojenie zruší
- X25 definuje multiplex virtuálnych okruhov - je možné mať viacero VC súčasne

na riadenie toku sa používa metóda statického okna, na riadenie chybovosti GBN

- X25 definuje fragmentáciu pomocou M-bitu (more), ktorý označuje, či išlo o posledný paket

- DTE môže mať vlastnosť PAD, čo umožňuje pripojiť do X25 siete zariadenia, ktoré nerozumejú paketom (PAD = packet assembler / deassembler)
- slúži to na pripojenie malých zariadení, ako napr. čítačka kariet či bankomat

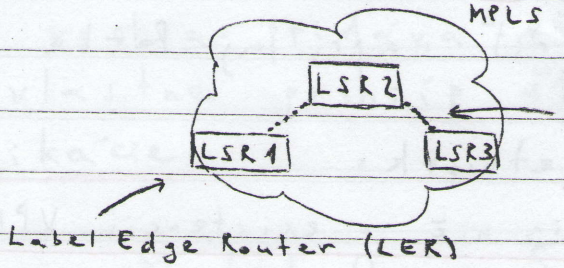
Frame Relay:

- X25 posiela množstvo nadbytočných dát, čo sa pri nástupe technológií s nižšou chybovosťou a vyššou kvalitou stalo zbytočným
- Frame Relay bol službou v ISDN sieťach, ale prax ukázala, že je to vhodný protokol pre rozľahlé siete
- FR takisto definuje rozhranie siete (nazýva sa UNI) používa rovnaký terminológiu ako X25
- FR definuje komunikáciu iba na linkovej vrstve (keďže je z rodiny HDLC protokolov) ⇒ oproti X25 odpadla sieťová vrstva
- jediné čo robí error control je kontrola prijatých rámcov a zahodenie chybných (vysporiadanie sa so stratenými rámcami nechávame na vyššie vrstvy)
- každý VC má dva parametre: CIR (commited information rate) - garantovaná prenosová rýchlosť a EIR (excess information rate) - rýchlosť nad CIR, ktorou môže užívateľ posielať dáta, ktoré sú ale doručované bez garancie
  - toto sa realizuje pomocou bitu DE, ktorý keď je nastavený na 1, dovoľuje zahodiť rámec
- zahltenie uzla v sieti sa signalizuje pomocou explicitného oznámenia o zahltení (jeden bit pre forward a jeden bit pre backward) v smere pohybu rámca, alebo v opačnom koncové zariadenie by malo zareagovať znížením rýchlosti

štandard síce hovorí aj o implementácii SVC, no v praxi sa používa PVC
- FR sa používa aj na prenos hlasu, čo bolo zabezpečené priradením vyššej priority hlasovým rámcom a fragmentáciou dát (aby hlasové rámce nečakali príliš dlho napr. pri prenose ethernetového rámca)

Multiprotocol Label Switching (MPLS):

- technológia, ktorá vnáša do IP siete prvky spojovej orientácie



- spojenie v MPLS sieti sa buduje vďaka Label Distribution Protocols, ktoré definujú, akým spôsobom si routre vymieňajú informácie o hodnote použitých labelov
- downstream router otvorí upstream routru akú hodnotu labelu má použiť na prenos paketu do požadovanej destinácie
- toto sa môže diať v unsolicited (keď mu to oznamuje bez vyžiadania) alebo on-demand (upstream router si vyžiada hodnotu labelu) móde
- label môže mať rôzny formát → MPLS header sa vloží medzi IP header a header linkovej vrstvy
- jeden paket môže niesť viacero MPLS headerov, čo sa nazýva label stacking

MPLS Traffic Engineering:

- klasické Shortest Path protokoly agregujú prevádzku do jednej časti siete, čím ju zaťažia oveľa viac ako „dlhšiu" cestu (tzv. hyperagregation problem)

vstupný router má právomoc rozhodnúť, kadiaľ má smerovať LSP (explicit source-based routed LSP)
na toto smerovanie sa používajú protokoly OSPF-TE a IS-IS-TE, čo sú rozšírené link-state protokoly rozšírené o TE (trafficengineering)
každý router si buduje TE databázu a má v nej všetky informácie, aby mohol vybudovať spojenie s požadovanou šírkou pásma
algoritmus na nájdenie takejto cesty sa nazýva Constrained Shortest Path (s obmedzeniami)
medzi obmedzenia môže patriť napr. aj farba linky

MPLS Virtual Private Network:
routre vo vnútri siete nevedia nič o existencii VPN, keďže táto informácia sa nachádza vo vnútornom labele
toto je výhodné, keďže pri pridaní novej pobočky alebo celej VPN nemusíme meniť konfiguráciu uzlov, cez ktoré táto VPN prechádza
informáciu o existencii VPN majú iba uzly PE (Provider Edge Router)

Mobilné dátové siete