Protokoli

Metodi popravljanja napak? (130)

<u>Popravljanje na sprejemni strani</u> - (ang. FEC - Forward Error Correction) oddajnik sporočila zaščiti s kanalsko kodo z dovolj veliko minimalno Hammingovo razdaljo, ki sprejemniku samemu, brez sodelovanja oddajnika, omogoča popravljanje napak, ob predpostavki da teh ni preveč.

<u>Popravljanje s ponovnim pošiljanjem</u> (ang. BEC - Backward Error Correction) - oddajnik uporabi kanalsko kodo z manjšo Hammingovo razdaljo, ki sprejemniku omogoča le odkrivanje napak, le-ta pa potem implicitno ali eksplicitno zahteva od oddajnika ponovno pošiljanje pokvarjenih sporočil; protokole ki uporabljajo tako metodo popravljanja napak pogosto imenujemo protokoli s ponavljanjem ali protokoli ARQ (ang. Automatic Repeat reQuest - AR

Glede odkrivanja in popravljanja napak imamo štiri osnovne možnosti:
- napak ne odkrivamo, in ne popravljamo
- napake odkrivamo, vendar napačno prenesenih sporočil ne popravljamo, ampak jih le zavrže
- napake popravljamo na sprejemni strani (torej jih oddajnik tako zaščiti, da jih lahko sprejemnik popravi)
- napake popravimo s ponovnim pošiljanjem, torej uporabimo enega izmed protokolov ARQ

<u>Rezidualna napaka</u> (preostala pogostnost napak, rezidualna pogostnost napak) - je napaka, katere ne odkrije določena metoda kodiranja. Pogostnost rezidualne napake je veliko manjša od pogostnosti napak, katero odkrijemo s pomočjo kodirnih metod.

<u>FEC</u> - protokol, ki detektira in popravi napako

<u>BEC</u> - protokol, ki samo detektira napako

Na dolge razdalje je boljši protokol s popravljanjem na sprejemni strani, ker protokol s ponavljanjem vnaša velike zakasnitve, ker mora zaradi napak ponovno pošiljati sporočila. To pa predstavlja problem za aplikacije ki potekajo v realnem času (govor ali video v živo). Zato pri takih aplikacijah uporabimo UDP protokol za prenašanje, kateri ne popravlja napak, medtem ko protokol TCP jih.

Protokol s popravljanjem na sprejemni strani - pomeni da oddanemu sporočilu dodaš režijo (dodatne bite s pomočjo katerih odkriva sprejemnik napako) npr. pri pariteti metodi je režija par. bi

Protokol s ponavljanjem - ko sprejemnik dobi sporočilo, vidi če so napake in sporoči oddajniku, da še enkrat pošlje sporočilo ki ga je napačno sprejel (to pomeni da rabimo dvosmerno komunikacijo) od oddajnika do sprejemnika in obratno!

Pri satelitih so velike razdalje in zaradi tega so zakasnitve velike. In ko oddajnik pošlje sporočilo mora čakati potrditev preden nadaljuje. Zato porabi preveč časa.

Ena razlika med protokoloma je v režiji, ki jo nosita s seboj.
Protokol s ponavljanjem nosi le toliko režije s seboj da je zmožen detektirat napako.
Protokol s popravljanjem pa nosi s seboj toliko režije, da je zmožen detektirat in odpravit napako.

	popravljanje na sprejemni strani	popravljanje s ponavljanjem
režija	velika	majhna
kanal	simpleksni	(pol) dupleksni
zakasnitve	konstantne	spremenljive
sinhroni promet	primerno	neprimerno

Protokoli s ponavljanjem! (132) (ARQ)

$t_{rt} = t_i + t_a + 2t_p$

t_{rt} = čas do potrditve
t_i = čas oddajanja informacijskega protokolnega sporočila
t_a = čas oddajanja potrditve
t_p = čas širjenja elektromagnetnega signala skozi kanal

1. Protokol s čakanjem – ABP (pogosto tudi protokol z alternirajočim bitom).
Najpreprostejša različica s ponavljanjem je protokol ABP. Pri tovrstnem protokolu oddajnik odda informacijsko protokolno sporočilo in nato čaka na njegovo potrditev. Šele ko oddajnik prejme potrditev lahko odda naslednje informacijsko protokolno sporočilo.

$M = 2$ → modul štetja infor. protokolnega sporočila

$$\eta = \frac{t_i}{t_{rt}} = \frac{t_i}{t_i + t_a + 2t_p}$$

→ učinkovitost protokola če v kanalu ni izgub

2. Protokol z drsečim oknom (kontinuirani protokol s ponavljanjem) Pri tem protokolu oddajni protokolni osebek pošilja inf. protokolna sporočila neposredno eno za drugim, počasno pa mu sprejemni protokolni osebek nazaj pošilja potrditve. Tukaj potrebujemo dvosmerni kanal za komunikacijo med osebkoma. Imamo tri vrste potrditve:
- individualna (potrjuje pravilen sprejem enega samega, v potrditvi specificiranega informacijskega protokolnega sporočila)
- kumulativna (potrjuje pravilen sprejem informacijskih protokolnih sporočil od prvega poslanega do vključno tistega, ki je v potrditvi specificirano)
- intervalna (potrjuje pravilen sprejem enega ali več zaporednih informacijskih protokolnih sporočil)

$$\eta = \frac{W_s \cdot t_i}{t_{rt}} = \frac{W_s \cdot t_i}{t_i + t_a + 2t_p}$$

W_s = širina oddajnega okna

3. Protokol s ponavljanjem N sporočil – GBN (=GBP) – Tukaj morajo sporočila prihajati do sprejemnega protokolnega osebka v pravilnem vrstnem redu. Če se katero protokolno sporočilo med prenosom pokvari ali izgubi mora oddajnik po izteku časovnika ponovno poslati ne le to sporočilo, ampak tudi vsa sporočila, ki mu sledijo, čeprav so nekatera izmed njih morda brez napake prispela do sprejemnika.

$M \geq W_s + 1$
$W_s \leq M - 1$

$$\eta = \frac{W_s \cdot t_i}{t_{rt}} = \frac{W_s \cdot t_i}{t_i + t_a + 2t_p}$$

4.) Protokol s selektivnim ponavljanjem – SRP – Pri tem protokolu pa oddajni protokolni osebek ponovno pošilja tista sporočila (ki so se pokvarila in izgubila), ne pa tista ki mu sledijo (kot je pri GBN). Da bo sprejemni protokolni osebek oddal uporabniku sporočila v vrstnem redu, na voljo na sprejemu še pomnilnik (nek dodaten pomnilnik). Tako bodo sporočila predana v pravilnem vrstnem redu.

$M \geq W_s + W_r = 2 \cdot W_s$ ($= 2W_r$), ker velja, da je $W_s = W_r$

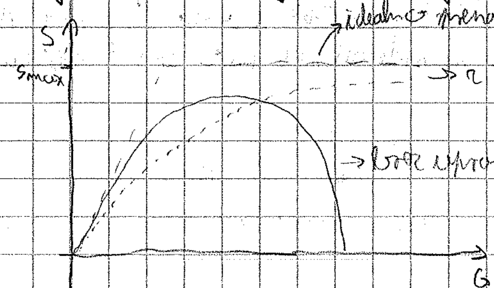
W_r = širina sprejemnega okna

	W_s	W_r
ABP	1	1
GBP	>1	1
SRP	W_s =	W_r

Krmilni pretok!

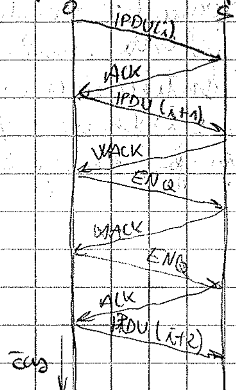
Vplivanje protokolnih entitet na pretok informacije imenujemo krmiljenje pretoka informacije. Krmiljenje pretoka nastopi pri zaporednih čakalnih vrst. Služi za usklajevanje pretoka podatkov zaradi različnih hitrosti in zgostitev prometa v omrežju.

Pomembno se je zavedati, da se telekomunikacijski sistemi brez krmiljenja pretoka informacij začne v primeru zgostitve obnašati nestabilno. Ko namreč entitete s polnimi čakalnimi vrstami začno izgubljati sporočila se izvorom teh sporočil začno iztekati časovniki. Ker pa izvori ne prejmejo potrdila za izgube v skladu z uporabljeno metodo ARQ izgubljena sporočila ponovno pošiljajo. S tem se ponujeni prometni pretok skozi vozlišče z zgostitvijo še povečuje, kar ima za posledico še večje izgube. Opravljeni prometni pretok skozi zgoščeno vozlišče se manjša in lahko v skrajnem primeru celo pade na nič. V tem primeru govorimo o mrtvi točki (deadlock). Poglavitna naloga krmiljenja pretoka je preprečevanje izgub in seveda tudi mrtve točke.



Metode krmiljenja pretoka v omrežju delimo na eksplicitne in implicitne.

Eksplicitne metode krmiljenja pretoka:
Ena od najpreprostejših metod se imenuje XON-XOFF metoda. V primeru, ko sprejemnik ni več mogel sprejemati podatkov, je poslal oddajniku tako imenovani XOFF znak. Ko pa je bil spet pripravljen na sprejem je to sporočil oddajniku z znakom XON.
Ob uporabi ARQ metode s čakanjem je lahko krmiljenje pretoka zelo preprosto: sprejemnik pač nekaj ne potrjuje prispelo sporočilo, vse dokler se toliko ne sprosti, da lahko spet sprejema. Da pa se ne bi medtem oddajniku potekel časovnik, sprejemnik eksplicitno sporoči oddajniku, da ne more več sprejemati, in sicer s sporočilom, ki ga označimo z oznako WACK (Wait Acknowledge). Ko oddajnik sprejme sporočilo WACK, preneha oddajati infor. protokolna sporočila in oddaja le poizvedovalna sporočila ENQ, vse dokler od sprejemnika ne prejme potrditvenega sporočila ACK. Nato nadaljuje z oddajanjem infor. protokolnih sporočil.



Implicitne metode krmiljenja pretoka so tiste, ki poleg tega da krmilijo pretok, zraven se odvijajo metode popravljanja napak ARQ.
Poznamo dve metodi:

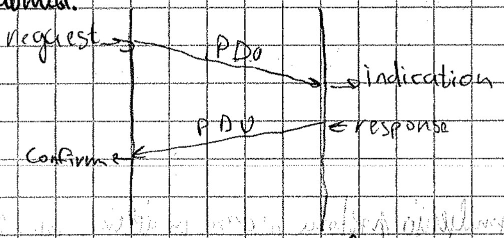
- pošiljanje potrditev in metoda drsečega okna - Sprejemnik ob veliki obremenitvi zavlačuje s pošiljanjem potrditve prejetih paketov, s tem tudi oddajnik zmanjša oddajanje. Vendar pri tem moramo biti pazljivi pri zavlačevanju, da se oddajniku ne izteče časovnik. Ker stvar samo še poslabša
- Metoda drsečega okna s spremenljivo širino okna - Tukaj sprejemnik poleg potrditve zraven še oddaja informacijo o zasedenosti njegovega pomnilnika, oddajnik pa na podlagi teh informacij krmili oddajanje

15 # Mrtve duše!

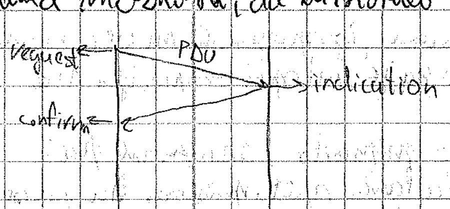
Ko neki omrežje pošljemo sporočilo se nam zgodi da nek določenih delih omrežja sporočilo predolgo zadržijo. Ker oddajnik ne dobi potrditve se mu izteče časovnik in pošlje še enkrat isto sporočilo, ki ga sprejemnik sprejme, zatem dobi pa še prvotno sporočilo (zakasnjeno) in pride do zmede (tem sporočilom bomo rekli mrtve duše). To je še posebej nevarno pri sporočilih, ki niso oštevilčena. To odpravimo tako, da posameznemu sporočilu določimo življenjsko dobo sporočila.

20 # Dvojni dogovor!

Najenostavnejši način vzpostavljanja ali prekinjanja zveze in podobnih zahtevnih postopkov med dvema protokolnima entitetama v ne preveč zahtevnih okoljih je dvojni dogovor, ki temelji na medsebojni izmenjavi dveh nasprotnih protokolnih sporočil. Tu gre vedno za storitev s potrditvijo, kar pomeni, da dobi tisti uporabnik, ki je podal zahtevo storitve, s sprožilom storitev potrdilo; z drugimi besedami pa to pomeni, da se postopek zaključi s primitivom confirm. Ločimo dve varianti: pri prvi ima tisti uporabnik, ki je prejemnik storitve, možnost storitev sprejeti ali ne (to mu omogoča primitiv response). To je storitev s potrditvijo uporabnika.

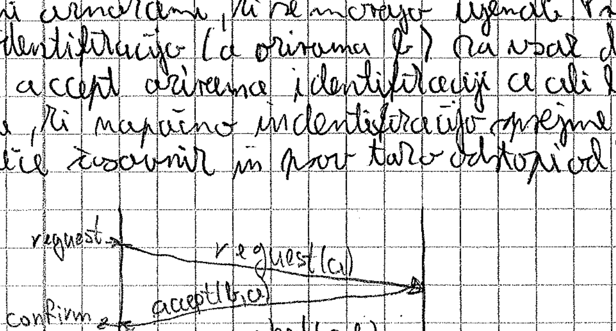


Druga možnost pa predstavlja storitev s potrditvijo izvajalca, kjer tisti uporabnik, ki je prejemnik storitve, nima možnosti, da bi storitev sprejel ali ne; v tem primeru odloči izvajalec storitve.



20 # Trojni dogovor!

Postopek dvojnega dogovora lahko da tudi neželene rezultate, če se v sistemu pojavijo mrtve duše. V primeru, ko pričakujemo takšne težave, ki pa je najpogosteje v slabe protokolnega sklada, pod katerim uporabljamo nepovezavno orientirane protokole, uporabimo metodo trojnega dogovora. Pri tem postopku si obe entiteti izmenjata 3 protokolna sporočila. Označimo jih s posebnimi oznakami, ki se morajo ujemati. Pri tem protokolni entiteti izbereta vsaka svojo identifikacijo (a oziroma b) za vsak dogovor posebej. Če se identifikacija a v sporočilu accept oziroma identifikaciji a ali b v sporočilu check ne ujemajo s pričakovanimi bo entiteta, ki napačno identifikacijo sprejme, prenehala z dogovarjanjem, drugi entiteti pa potem izteče časovnik in prav tako odstopi od dogovora.



10 # Protokolni sklad TCP/IP

TCP/IP je protokolni sklad, ki je nastal in se razvijal skupaj z Internetom.
Internet je omrežje za prenos informacije na paketni način.
Protokolni sklad ki se uporablja v omrežju Internet, se razlikuje od OSI-referenčnega modela in ima definirane le 4 sloje:

1) Sloj za dostop do omrežja (Network access layer) omogoča dostop končnega računalnika ali usmerjevalnika do omrežja

2) Internetni sloj (Internet layer) skrbi za usmerjanje informacije skozi Internet od enega končnega računalnika do drugega glede na naslov izvora in naslov ponora informacije, torej za usmerjanje informacije med omrežji. V tem sloju se uporablja protokol IP (Internet Protocol), ki deluje med končnim računalnikom in usmerjevalnikom oziroma med dvema usmerjevalnikoma v Internetu. Protokol IP definira tudi naslavljanje, tako je vsakemu priključku na omrežje Internet dodeljen IP naslov oziroma IP števila. V internetnem sloju deluje tudi protokol ICMP. Naloga ICMP je sporočanje in nadzor nad protokolom IP.

3.) Transportni sloj (Transport Layer) skrbi za prenos informacije neposredno med dvema končnima računalnikoma, ki med seboj komunicirata preko Interneta. V tem sloju se alternativno uporabljata dva protokola: TCP (Transport Control Protocol) je povezavno orientiran, UDP (User Datagram Protocol), pa nepovezavno orientiran

4.) Aplikacijski nivo (Application layer) tu se uporablja vrsta protokolov, ki nudijo neposredno podporo porazdeljenim aplikacijam na Internetu.
Aplikacijski protokol Telnet omogoča uporabniku enega računalnika, da preko Interneta dela na nekem drugem računalniku, kot da bi bil terminal uporabnika neposredno povezan z oddaljenim računalnikom.
FTP (File Transfer Protocol) je protokol, ki omogoča manipuliranje z datotekami na oddaljenem ali lokalnem računalniku ter prenos datotek v obeh smereh, seveda pri pogoju, da ima uporabnik za to potrebna dovoljenja.
TFTP (Trivial File Transfer Protocol) protokol, ki omogoča prenos datotek z oddaljenega na lokalni računalnik ali obratno na preprost način in z minimalno obremenitvijo virov (procesorja in spomina). Elektronska pošta je sistem, ki uporablja protokol aplikacijskega sloja SMTP (Simple Mail Transfer Protocol) za pošiljanje elektronskih pisem v poštni predal in protokol aplikacijskega sloja POP3 (Post Office Protocol version 3) za dostop do poštnega predala.
Najbolj znana in priljubljena aplikacija v Internetu brskanje po omrežju informacijskih virov svetovnem spletu (World Wide Web - WWW); prenos spletnih dokumentov omogoča protokol aplikacijskega sloja HTTP (HyperText Transfer Protocol). Vsi našteti aplikacijski protokoli delujejo na principu odjemalec - strežnik (client - server)

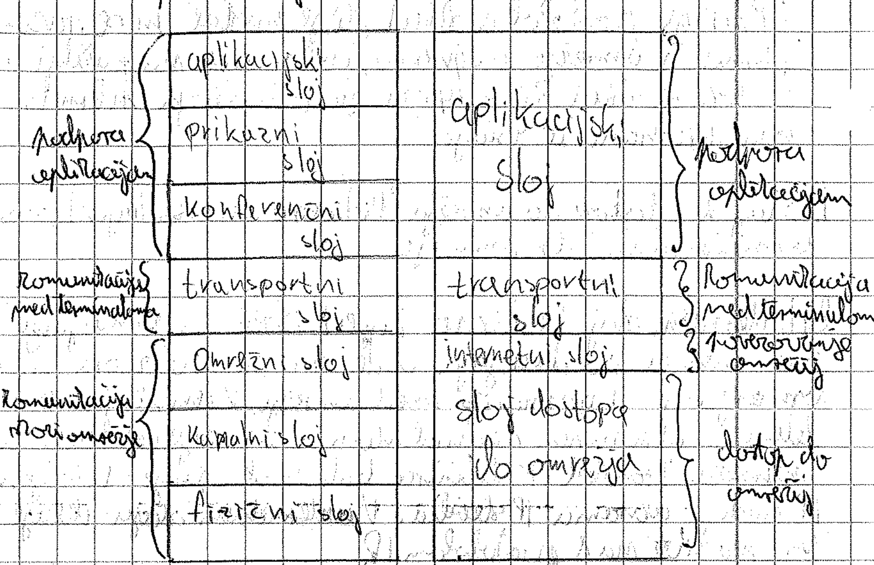
V primeru, da imamo več povezav k enim FTP strežnikom potem sporočila multipleksiramo. Strežniki uporabljajo t.i. vnaprej znane številke vrat (well known port numbers), ki so različne za različne aplikacijske protokole (ftp = 21, telnet = 23, ...) in imajo dodeljene decimalne vrednosti med 1 in 1023. Ti porti so stalni pri strežniku, medtem ko naključni port odjemalca se nekje dodeli le začasno (npr. za čas ko bomo zdownloadali neko datoteko → potem se ta port dodeli nekomu drugemu). Številka porta odjemalca, ki se začasno uporablja, ga dodeli transportni sloj in mora biti nad 1023. Poleg št. porta je potreben še naslov in to dvoje skupaj imenujemo vtičnica = naslov + št. porta.
Odjemalec in strežnik sta v odnosu podrejenosti.

S pomočjo naslova vemo komu gre katero sporočilo (demultipleksiranje)

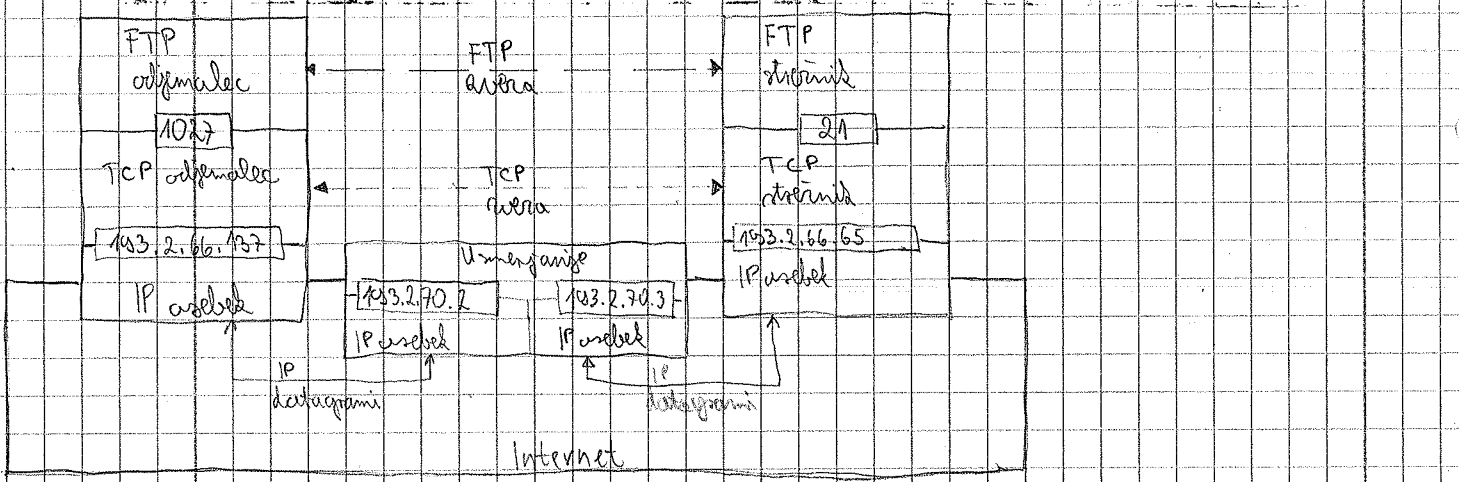
Vnaprej znane številke vrat

vnaprej znana številka vrat	aplikacijski protokol	transportni protokol
21	FTP	TCP
23	telnet	TCP
25	SMTP	TCP
69	TFTP	UDP
80	HTTP	TCP
110	POP3	TCP

Primerjava referenčnega modela OSI in protokolnega sklada TCP/IP.



Primer komunikacije med FTP strežnikom in FTP odjemalcem.



[illegible] stanja imenujemo tudi konvergenca sistema.

Učinkovitost protokola pomeni, da protokol čimbolj izkoristi dane kapacitete kanala in protokolnih osebkov.

Standardiziranost protokola je pogoj za njegovo širšo uporabo, saj zagotavlja, da lahko po teh protokolih skladno med seboj komunicirajo naprave, ki so izdelki različnih proizvajalcev in v lasti različnih v upravljanju različnih operaterjev.

Detekcija in popravljanje napak

Paritetna metoda (lahko samo detektira napako) je ena izmed prvih in tudi najpreprostejših metod za zaščito podatkov pred napakami v telekomunikacijskem kanalu. Poznamo 2 različici - metodo sode in metodo lihe paritete. V obeh primerih dodamo izvornemu sporočilu predvsem eno binarno vrednost tako, da je celotno število logičnih enic v sporočilu (s paritetnim bitom vred!) sodo v primeru sode paritete oziroma liho v primeru lihe paritete.

Metoda seštevanja - pri metodi seštevanja (checksum) zaščitimo prenašano sporočilo tako, da dodamo zaščitne bite po nekoliko kompleksnejšem algoritmu kot pri paritetni metodi, temu ustrezno pa metoda seštevanja nudi močnejšo zaščito, kar pomeni, da je delež po tej metodi detektiranih napak večji - OMOGOČA SAMO ODKRIVANJE NAPAK.

Vnaprej znane številke vrat | Primerjava referenčnega modela OSI in [illegible]

Lastnosti protokola

<u>Logična pravilnost</u> protokolnih pravil pomeni, da ta pravila ne vsebujejo napak, ki bi vodile k napačnemu delovanju komunikacijskega sistema.

<u>Popolnost, nedvoumnost in skladnost</u> specifikacije protokolnih pravil pomenijo, da obstaja protokolno pravilo za vsakršno situacijo, v kateri se lahko znajde protokolna entiteta, da si teh pravil ni mogoče razlagati na dvoumen način in da si pravila niso v medsebojnem nasprotju.

<u>Pravičnost</u> protokola pomeni, da daje protokol vsem udeležencem v komunikacijskem procesu enake možnosti, da odpošiljajo svoja protokolna sporočila.

<u>Robustnost</u> protokola pomeni, da protokol predvidi tudi možnost nastopa situacij, do katerih pri normalnem delovanju sicer ne pride. Lastnost sistema, da se je sam sposoben vrniti iz nenormalnih stanj v katera je zašel zaradi napak ali napada, nazaj v normalna stanja imenujemo tudi konvergenca sistema.

<u>Učinkovitost</u> protokola pomeni, da protokol čimbolj izkorišča dane kapacitete kanala in protokolnih osebkov.

<u>Standardiziranost</u> protokola je pogoj za njegovo širšo uporabo, saj zagotavlja, da lahko po tem protokolu skladno med seboj komunicirajo naprave, ki so izdelki različnih proizvajalcev in v lasti oziroma v upravljanju različnih operaterjev.

Detekcija in popravljanje napak

Paritetna metoda (lahko samo detektira napako) je ena izmed prvih in tudi najpreprostejših metod za zaščito podatkov pred napakami v telekomunikacijskem kanalu.
Poznamo 2 različici - metodo sode in metodo lihe paritete. V obeh primerih dodamo izvornemu sporočilu predvidoma eno binarno vrednost tako, da je celotno število logičnih enic v sporočilu (s paritetnim bitom vred!) sodo v primeru sode paritete oziroma liho v primeru lihe paritete.

Metoda seštevanja - pri metodi seštevanja (checksum) zaščitimo prenašano sporočilo tako, da dodamo zaščitne bite po nekoliko kompleksnejšem algoritmu kot pri paritetni metodi, temu ustrezno pa metoda seštevanja nudi močnejšo zaščito, kar pomeni, da je delež po tej metodi detektiranih napak večji - OMOGOČA SAMO ODKRIVANJE NAPAK.

Metoda krožnega redundančnega preizkusa - to je na videz še bolj zapleten način izračunavanja zaščitne kode, ki pa nam prinese še učinkovitejšo detekcijo napak, poleg tega pa ne zahteva razdelitve sporočila na besede, kot to zahteva metoda seštevanja. Ta metoda je znana tudi pod kratico CRC (Cyclic Redundancy Check). Kode CRC štejemo med polinomske kode, katerih teoretična razlaga temelji na predstavitvi zaporedij binarnih vrednosti kot polinomov.

Seštevanje po modulu 2.

Sporočilo je M, generacijski polinom = G, ostanek = R, poslano sporočilo je T = M + R

x	y	x ⊕ y
0	0	0
0	1	1
1	0	1
1	1	0

Oddajnik doda zaporedju M dolžine m, ki ga želi zaščititi, r ničel in tako dobljeno zaporedje dolžine m+r pošlje sprejemniku. Le-ta sprejeto zaporedje deli z istim generacijskim polinomom G. Če pri tem deljenju dobi ostanek 0, sklepa, da pri prenosu ni prišlo do napake, če je ostanek od nič različen, je pri prenosu zagotovo prišlo do napake. Pri tem oddajnik in sprejemnik izvajata vse postopke deljenja, ki temeljijo na seštevanju po modulu 2 in ga imenujemo dolgo deljenje.

[illegible]

Hammingova razdalja

Sposobnost nekega postopka kodiranja pogosto ocenjujemo na podlagi Hammingove razdalje. Hammingova razdalja d_H med dvema kodnima besedama je definirana kot število binarnih vrednosti, v katerih se ti dve besedi med seboj razlikujeta.

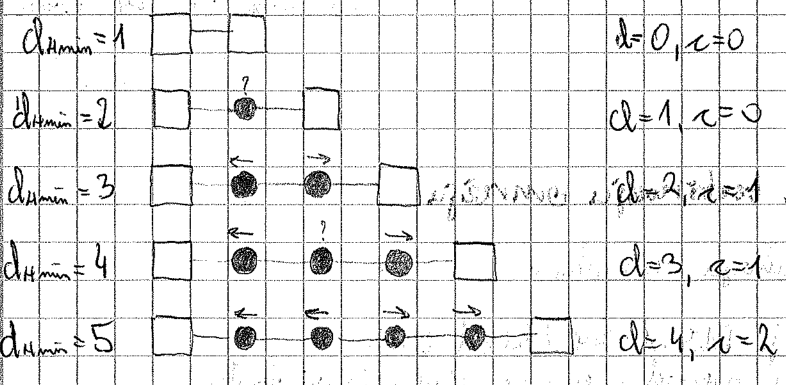
Minimalna Hammingova razdalja d_{Hmin} je definirana kot minimalno število binarnih vrednosti, v katerih se med seboj razlikujeta poljubni dve kodni besedi.

□ → veljavna kodna beseda — d = število napak v sporočilu, ki jih pri določeni kodi lahko detektiramo

● → neveljavna kodna beseda — c = število napak, ki jih lahko popravimo

← → - puščice označujejo napake, ki jih lahko popravimo — ? - vprašaji označujejo napake, ki jih lahko le odkrijemo

V primeru $d_{Hmin} = 4$ bomo lahko zanesljivo popravili sporočila z največ eno bitno napako in detektirali tista z največ tremi bitnimi napakami, v primeru $d_{Hmin} = 5$ pa bomo lahko zanesljivo popravili sporočila z največ dvema in detektirali sporočila z največ štirimi bitnimi napakami.



Zanesljivo bomo lahko detektirali napačno prenešena sporočila v primeru, če se je v sporočilu pojavilo največ d binarnih vrednosti $d = d_{Hmin} - 1$

Zanesljivo bomo lahko popravili sprejeto sporočila le v primeru, ko se je v sporočilu pojavilo največ c binarnih vrednosti, kjer ima c vrednost $c = \frac{d_{Hmin} - 1}{2}$. C je lahko le celo število, rezultat, ki ga dobimo po enačbi moramo zaokrožiti navzdol.

Metoda krožnega redundančnega preizkusa - to je na videz še bolj zapleten način izračunavanja zaščitne kode, ki pa nam prinese še učinkovitejšo detekcijo napak, poleg tega pa ne zahteva razdelitve sporočila na besede, kot to zahtevajo metode seštevanja. Ta metoda je znana tudi pod kratico CRC (Cyclic Redundancy Check). Kode CRC štejemo med polinomske kode, katerih teoretična razlaga temelji na predstavitvi zaporedij binarnih vrednosti kot polinomov

Metoda dvodimenzionalne paritete - Sedaj lahko sprejemnik preverja pravilnost prejetega sporočila dvodimenzionalno, torej po vrsticah in po stolpcih. Če se bo pri prenosu pokvaril i-ti bit j-te besede, bo sprejemnik našel paritetno napako v j-ti besedi, prav tako pa tudi v i-tem bitu paritetne besede. Sprejemnik bo torej ne le detektiral napako, ampak jo bo tudi lociral - natančno bo vedel, kateri bit v sporočilu se je pokvaril. Ker pa ima vsak bit lahko le vrednost 0 ali 1, bo sprejemnik z invertiranjem pokvarjenega bita popravil celotno sporočilo. To bo seveda vedno mogoče, če bo v sporočilu nastala ena binarna napaka. V primeru več napak v sporočilu napake ne bo vedno možno popraviti

Hammingova razdalja

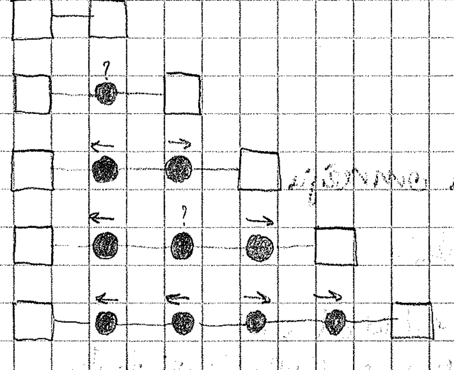
Sposobnost nekega postopka kodiranja pogosto ocenjujemo na podlagi Hammingove razdalje. Hammingova razdalja d_H med dvema kodnima besedama je definirana kot število binarnih vrednosti, v katerih se ti dve besedi med seboj razlikujeta.
Minimalna Hammingova razdalja d_{Hmin} je definirana kot minimalno število binarnih vrednosti, v katerih se med seboj razlikujeta poljubni dve kodni besedi.

□ → veljavna kodna beseda d = število napak v sporočilu, ki jih pri določeni kodi lahko detektiramo

● → neveljavna kodna beseda c = število napak, ki jih lahko popravimo

← → - puščice označujejo napake, ki jih lahko popravimo ? - vprašaji označujejo napake, ki jih lahko detektiramo

V primeru $d_{Hmin} = 4$ bomo lahko zanesljivo popravili sporočila z največ eno bitno napako in detektirali tista z največ tremi bitnimi napakami, v primeru $d_{Hmin} = 5$ pa bomo lahko zanesljivo popravili sporočila z največ dvema in detektirali sporočila z največ štirimi bitnimi napakami.

$d_{Hmin} = 1$  $d = 0, c = 0$

$d_{Hmin} = 2$ $d = 1, c = 0$

$d_{Hmin} = 3$ $d = 2, c = 1$

$d_{Hmin} = 4$ $d = 3, c = 1$

$d_{Hmin} = 5$ $d = 4, c = 2$

Zanesljivo bomo lahko detektirali napačno prenešena sporočila v primeru, če se je v sporočilu pokvarilo največ d binarnih vrednosti $d = d_{Hmin} - 1$

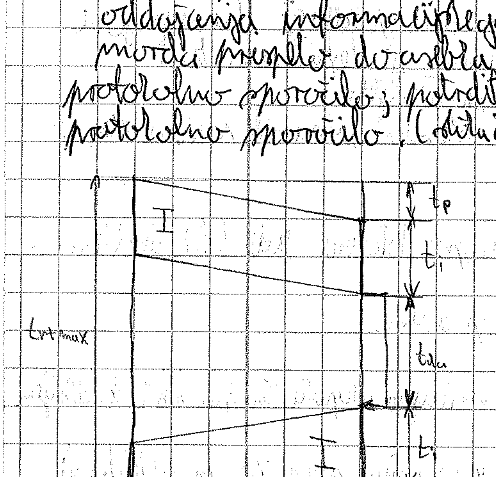
Zanesljivo bomo lahko popravili (prejeto) sporočila le v primeru, ko se je v sporočilu pokvarilo največ c binarnih vrednosti, kjer ima c vrednost $c = \frac{d_{Hmin} - 1}{2}$. C je lahko le celo število, rezultat, ki ga dobimo po enačbi moramo zaokrožiti navzdol.

Kvadratek označuje veljavno kodno besedo, krogec pa neveljavno kodno besedo. Puščica ka v isto smer, katera je bližje veljavni kodni besedi. Pokvarjen bit pomeni, da se bit pokvari, zato rečemo da je beseda neveljavna. Gledamo prvi krogec s puščico na levo (gledamo na d... ko pomeni da bo to neveljavno besedo popravil na veljavno besedo, ki je na levi strani, se je do veljavne besede na desni strani razdalja enaka 4, do leve pa samo 1. Drugi krogec s puščico na levo bo isto popravil na levo, saj je razdalja na levo 2 na desno pa 3. Enako velja za tretji in četrti krogec, le da je popravi na desno. V tem primeru se lahko zanašamo popravi sporočila z največ 2 bitnima napakama in detektira sporočila z največ 4 bitnimi napakami.

Natovarjanje

Protokolni osebek potrditve sporočil, ki jih je sprejel kot sprejemnik, neposredno kar vključi v informacijska protokolna sporočila, ki jih oddaja kot oddajnik. Tako potrditve uporabljajo sinhronizacijsko zaporedno in zaščitno kodo informacijskih sporočil, na tudi ne potrebujejo posebne identifikacije saj protokol določa na katerem mestu v informacijskem protokolnem sporočilu se nahaja potrditev informacijskih sporočil, ki potujejo v obratni smeri. Takšno tehniko oddajanja potrditev informacijskim sporočilom imenujemo natovarjanje (piggybacking)

Osebek se sam prilagodi prometnemu pretoku po kanalu; ko je ta velik natovarja potrditve v informacijska protokolna sporočila; ko je prometni pretok majhen, pa potrditve pošilja v posebnih nadzornih sporočilih.
Seveda moramo biti nastaviti oba časovnika (oddajnika in sprejemnika) med seboj uskladiti. Čas do potrditve je namreč v najugodnejšem primeru enak: $t_{rt} = 2 \cdot t_i + t_{da} + 2 \cdot t_p$
Izteka časovnika smo označili kot t_{da}, saj gre v dejstvu za zakasnjeno potrditev. (slika 1)
Pri tem pa se moramo zavedati, da čas izteka časovnika t_{da} ne sme biti krajši od časa oddajanja informacijskega protokolnega sporočila t_i, saj je sporočilo, ki ga moramo potrditi morda prispelo do osebka v trenutku, ko je le-ta pravkar začel oddajati neko informacijsko protokolno sporočilo; potrditev bomo lahko vključili šele v naslednje informacijsko ali nadzorno protokolno sporočilo. (slika 2)

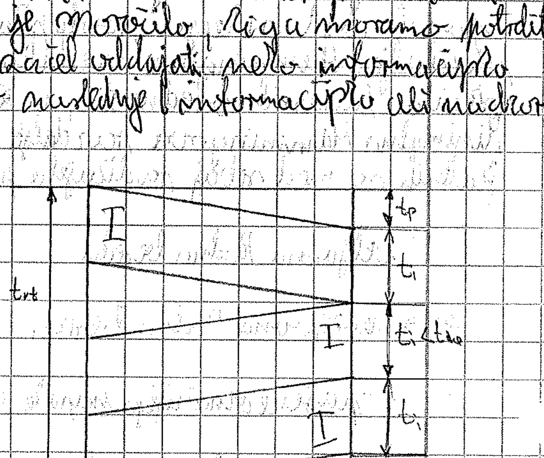


t_{da} = čas izteka časovnika

t_i = čas oddajanja IPDU-ja (informacijskega protokol. sporočila)

t_{rt} = čas do potrditve

slika 1.
Čas do potrditve pri natovarjanju



slika 2.
Minimalni čas izteka časovnika

$t_{da} = t_i \Rightarrow t_{rt} = 2t_i + t_i + 2 \cdot t_p = 3t_i + 2t_p$

Internetske aplikacije namenjene testiranju omrežja

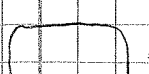
hostname ... pove ime lokalnega računalnika

nslookup ime
nslookup IP_številka
} S pomočjo DNS storitve dobi od DNS strežnika prevod imena v IP številko ali obratno

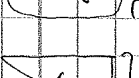
ping ime
ping IP_številka
} pošlje naslovljenemu računalniku datagram, čaka odgovor nanj in izračuna časovno razliko (čas do odgovora)

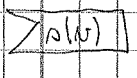
SDL - je nadgradnja končnega avtomata

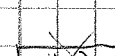
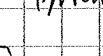
Nekateri izmed omenjenih podatkovnih tipov:
- Integer (omejitev celih števil)
- Duration (omejitev časovnih spremenljivk) - t - časovnik
- Time (omejitev časov)


- začetni simbol, začetni simbol definira začetek deklaracije procesa; prehodi tega simbola v simbol nekega drugega stanja predstavlja inicializacijo procesa


- stanje - V tem simbolu je zapisano ime stanja. Tak simbol uporabimo na začetku in koncu prehoda iz stanja v stanje

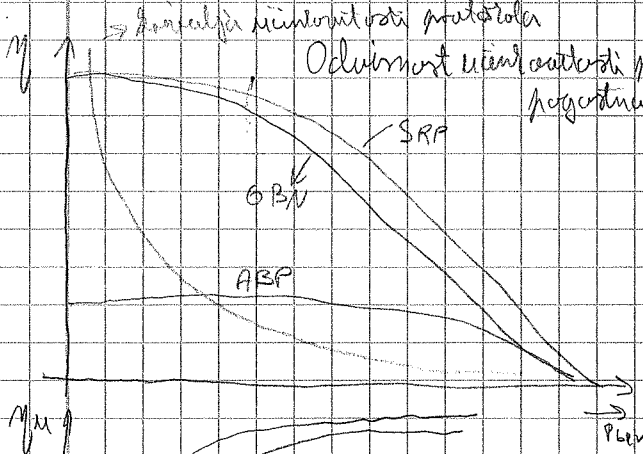

Vhodno sporočilo - simbol vhodnega sporočila pomeni, da je proces prejel določil iz vhodne čakalne vrste sporočilo (vhodni signal ali interni časovnik) S; temu lahko dodamo še, kateri proces je to sporočilo poslal, če se to ne razume samo po sebi; če signal nosi s seboj vrednost, lahko z zapisom S(v) v simbolu to vrednost shranimo s spremenljivko z imenom v.

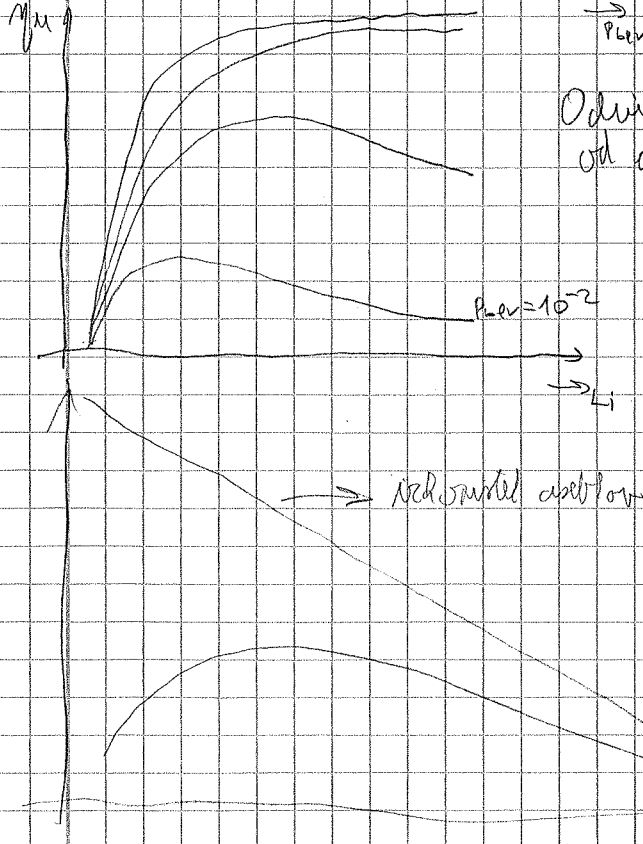

Izhodno sporočilo - Simbol izhodnega sporočila pomeni, da je proces oddal signal S; temu lahko dodamo še, kateremu procesu je to sporočilo namenjeno, če se to ne razume samo po sebi; če signal S nosi s seboj neko vrednost, določimo to vrednost kot vrednost spremenljivke z zapisom S(v) v simbolu.


ne sme biti ⇒  ⇒ S je signal, ki nosi s seboj vrednost v spremenljivki x

neomejena vrednost

X .... integer





→ različni avtorji v odvisnosti od dolžine sporočila