Drago Hercog

Telecommunication Protocols

(Aplicative Electrotechnics, Telecommunications, 5th semester)

Laboratory Exercises Guidelines

Telekomunikacijski Protokoli

(Aplikativna elektrotehnika, Telekomunikacije, 5. semester)

Navodila za laboratorijske vaje

Guidelines are Temporary!

Navodila so začasna!

Contents:

Nazaj na domaco stran predmeta

Technical Information on the Web

Iskanje strokovnih informacij na spletu

Use your favourite Web browser to see the following technical information:

- Use the system COBISS to find all books of William Stallings available from any slovenian library!
- Find the list of International Telecommunication Union – Telecommunications Sector (ITU-T) recommendations that appear in the I-Series (http://www.itu.int/)!
- Find the list of 802 series standards issued by the Institute of Electrical and Electronics Engineers (IEEE) (http://www.ieee.org)!
- Find books discussing (tele)communication protocols issued by Prentice-Hall (http://www.prenhall.com) and Addison Wesley (http://www.awl.com) publishers!
- Read the description of terms HTML and URL in the Wikipedia encyclopedy (http://www.wikipedia.org)!

- - - - - - - - - - - - -

Uporabite poljubni spletni brskalnik za iskanje naslednje strokovne informacije:

- S pomočjo sistema COBISS poiščite vse knjige Williama Stallingsa, dostopne v slovenskih knjižnicah!

- Poiščite seznam priporočil serije I Mednarodne zveze za telekomunikacije (International Telecommunication Union – Telecommunications Sector - ITU-T) (http://www.itu.int/)!
- Find the list of 802 series standards issued by the Institute of Electrical and Electronics Engineers (IEEE) (http://www.ieee.org)!
- Find books discussing (tele)communication protocols issued by Prentice-Hall (http://www.prenhall.com) and Addison Wesley (http://www.awl.com) publishers!
- Read the description of terms HTML and URL in the Wikipedia encyclopedy (http://www.wikipedia.org)!

top

Protocol Efficiency

Učinkovitost protokola

Two protocol entities exchange protocol data units with average length of 100 o which includes 5 o of overhead. The channel interconnecting the two entities is duplex, the transmission rate is 64 kb/s. This communication process is simulated so that the input queues between users and protocol entities can never become empty. After 15 s of simulation time each of the two protocol entities has successfully received 900 protocol data units. Which is the protocol efficiency and which is the efficiency experienced by users?

- - - - - - - - - - - - -

S simulatorjem simuliramo komunikacijo med protokolnima osebkoma, ki si preko dupleksnega kanala z nazivno hitrostjo 64 kb/s izmenjujeta informacijska protokolna sporočila s povprečno dolžino 100 o, ki vsebujejo 5 oktetov režije. Pri tem skrbimo, da čakala vrsta med uporabnikoma in protokolnima osebkoma ni nikoli prazna. Po preteku simulacijskega časa 15 s vsak izmed protokolnih osebkov uspešno sprejme 900 protokolnih sporočil. Kolikšen je izkoristek protokola in kolikšen izkoristek vidita uporabnika?

top

Packet Delay

Zakasnitev paketov

Transmitter and receiver are interconnected by a 100 km long physical channel with transmission rate 1 Mb/s and electromagnetic wave propagation speed 0,8×c. After the transmitter has transmitted a packet it waits for the acknowledgment and then immediately transmits the next packet, or retransmits the packet as soon as possible if it has received no acknowledgment. The length of packets is 200 o and the length of acknowledgments is 5 o. 80 % of all packets are successfully received in the first attempt, 15 % in the second, and 5 % in the third attempt. Which is the average delay of packets transfer?

- - - - - - - - - - - - -

Oddajnik oddaja in sprejemnik sprejema pakete z dolžino 200 o z nazivno hitrostjo 1Mb/s. Oddajnik in sprejemnik povezuje fizični kanal dolžine 100 km, po katerem se elektromagnetni

val širi s hitrostjo 0,8×c. Oddajnik po oddaji čaka na potrditev paketa in oddajo ponovi po najkrajšem možnem času, če potrditve ne prejme, sicer takoj odda naslednji paket. Dolžina potrditve je 5 o. 80 % vseh paketov je uspešno prenešenih v prvem poskusu, 15 % v drugem in 5 % v tretjem. Kolikšna je povprečna zakasnitev paketov?

top

Server and Linux Operating System

Strežnik in operacijski sistem Linux

In our laboratory experiments a server will occasionally be used. The domain name of this server computer is »protokoli.fe.uni-lj.si«. The Linux operating system (Ubuntu distribution) runs on it. On this server computer, http, ssh, ftp, smtp, pop3... server processes (daemons) run. Each student present in the lab will be assigned a username on this computer. The user name will normally consist of the student's last name and the first letter of the student's first name, all lowercase letters (e.g., John Smith would have a user name smithj). A default password will initially be assigned to a new user; however, he/she will be forced to change the password at first login. A user will have to remember his/her password to be able to access the server. It will be the student's responsibility not to reveal his/her password to any other person. So the user will be responsible for any inappropriate action on or from the server on behalf of him/her. The user's home directory will be /users/*username*. This operating system does not offer a graphical user interface (GUI) for usual work, therefore a character-oriented user interface will be used via

the ssh application-layer protocol, using the »putty« application. The most important linux commands are shown in the following table.

command	**functionality**
passwd	set or change user password
pwd	display working directory
cd *directory*	change working directory to *directory*
cd ..	change working directory to parent directory
cd $HOME	change working directory to user's home directory
mkdir *directory*	create (make) directory *directory*
rmdir *directory*	delete (remove) directory *directory*
ls -l	list the contents of the working directory
ls –l *directory*	list the contents of the directory *directory*
cp *file1 file2*	copy file *file1*into file *file2*
mv *file1 file2*	rename (move) *file1*into *file2*
cat *file*	list contents of a text file *file*
more *file*	list contents of a text file *file*, one screen at a time
chmod *ugo +/-rwx file*	change access permissions of user/group/others, adding/removing read/write/execute permission, for *file*
exit	exit the session

- - - - - - - - - - - - -

Pri laboratorijskih vajah bomo občasno uporabljali strežnik z domenskim imenom »protokoli.fe.uni-lj.si«. Na njem teče operacijski sistem Linux distribucije Ubuntu. Na tem

strežniškem računalniku tečejo strežniški procesi (demoni) http, ssh, ftp, smtp, pop3... Vsak študent, ki bo prisoten na laboratorijskih vajah, bo na tem računalniku dobil uporabniško ime, ki bo normalno sestavljeno iz priimka in prve črke imena uporabnika (same male črke, vendar brez šumnikov); tako bi npr. uporabnik Janez Kovač imel uporabniško ime kovacj. Vsak novi uporabnik bo v začetku dobil privzeto geslo, ki pa ga bo moral ob prvi prijavi na strežnik spremeniti. Geslo si bo moral zapomniti, če bo želel do strežnika dostopati, ne bo ga pa smel razkriti nikomur drugemu, saj bo sam odgovoren za kakršno koli neprimerno dejanje, izvedeno na ali s strežnika. Domači direktorij uporabnika bo /users/*uporabniskoime*. Ta operacijski sistem za običajno rabo ne ponuja grafičnega vmesnika (GUI), ampak le znakovni vmesnik, ki ga bomo uporabljali preko protokola aplikacijskega sloja ssh, za kar bomo uporabili aplikacijo »putty«. Naslednja tabela prikazuje najobičajnejše ukaze operacijskega sistema Linux.

ukaz	**funkcionalnost**
passwd	določi ali spremeni geslo uporabnika
pwd	prikaži delovni direktorij
cd *directory*	spremeni delovni direktorij v *directory*
cd ..	spremeni delovni direktorij v višje ležeči direktorij
cd $HOME	spremeni delovni direktorij v domači direktorij
mkdir *directory*	ustvari (make) direktorij *directory*
rmdir *directory*	izbriši (remove) direktorij *directory*
ls -l	prikaži vsebino delovnega direktorija
ls –l *directory*	prikaži vsebino direktorija *directory*
cp *file1 file2*	kopiraj datoteko *file1* v datoteko *file2*
mv *file1 file2*	preimenuj (move) datoteko *file1* v datoteko *file2*

cat *file*	prikaži vsebino tekstovne datoteke *file*
more *file*	prikaži vsebino tekstovne datoteke *file*, stran za stranjo
cmod *ugo +/-rwx file*	spremeni dovoljenja za dostop lastnika/skupine/ostalih do datoteke *file*, tako da dodaš/odvzameš dovoljenje za branje/pisanje/izvajanje
exit	končaj sejo

top

Virtual Terminal

Navidezni terminal

Virtual terminal is a distributed application that allows a client to work interactively on a remote server just as if the user's terminal were directly connected to the server. A virtual terminal provides for a user interface between a user on a client computer and the operating system on a distant server computer and can be either character- or graphic-oriented. A virtual terminal distributed application is supported by an application-layer protocol. The protocol relays data between the user's i/o devices (keyboard, mouse, screen) and the operating system of the distant computer. The most frequently used application-layer protocols are telnet (character-oriented), ssh (character-oriented, secure), rlogin (character-oriented, developed for unix), x windows (graphic-oriented). You will use the virtual terminal application putty.exe and ssh protocol to connect to the protokoli.fe.uni-lj.si server, login with your user name / password and explore the linux environment (see the

previous paragraph!). Try out the above listed linux commands!

- - - - - - - - - - - - -

Navidezni terminal je porazdeljena aplikacija, ki uporabniku omogoča delo na oddaljenem računalniku tako, kot bi bil uporabnikov terminal neposredno priključen na oddaljeni računalnik, in tako omogoča odjemalcu interaktivno delo na oddaljenem strežniku. Navidezni terminal torej predstavlja uporabniški vmesnik med uporabnikom na odjemalskem računalniku in operacijskim sistemom na strežniškem računalniku; lahko je znakovno ali grafično orientiran. Porazdeljeno aplikacijo navideznega terminala podpira ustrezni protokol aplikacijskega sloja protokolnega sklada. Ta protokol le posreduje podatke med vhodno/izhodnimi napravami uporabnika (tipkovnico, miško, zaslonom) in operacijskim sistemom oddaljenega računalnika. Najpogostejši protokoli aplikacijskega sloja, ki podpirajo navidezni terminal, so telnet (znakovno orientiran), ssh (znakovno orientiran, zagotavlja varen prenos), rlogin (znakovno orientiran, razvit za dostop do operacijskega sistema unix), x windows (grafično orientiran). Uporabili bomo aplikacijo navideznega terminala putty.exe in protokol aplikacijskega sloja ssh ter se prijavili na strežniku protokoli.fe.uni-lj.si s svojim uporabniškim imenom in geslom. Z uporabo linux ukazov, ki so našteti v predhodnem razdelku, bomo raziskali okolje operacijskega sistema linux in preizkusili zgoraj naštete ukaze.

top

File Transfer, ftp Protocol

Prenos datotek, protokol ftp

To transfer files from a client to a server computer or vice versa, special application-layer protocols are used that support file transfer applications. One of the most frequently used application-layer protocols supporting file transfer in the Internet is FTP (File Transfer Protocol). A usual arrangement of usage of file transfer application over the FTP protocol consists of two computers, a client and a server. In both of them, two layers of communication can be considered. In the above layer the distributed application runs, including two different applications in client and server computers, respectively (the application in server computer is usually a part of the server operating system); additionally, the operation of the distributed application is often controlled on the client side by a human user over a user interface between the user and the client application. In the lower layer the FTP protocol runs; an ftp protocol entity consists of two parts. One of them is the control function which controls an ftp session in which a user can browse the file system on the server, modify it (e. g. by renaming or deleting files) or transfer files between file systems of client and server computers; however, when setting up a new session, a user must be authenticated by the server with his/her username/password pair. Normally, the language used by a user to interact with the application over the user interface is different (although not necessarily) from the protocol used for communication between the two application-layer protocol entities, and depends on the application at the client side. The other part of the ftp protocol entity is the file transfer function which controls the transfer of files between the two file systems. For session control and file

transfer respectively, two different channels (implemented as separate TCP connections) are used (the control TCP connection uses port no. 21, and the data connection uses port no. 20 on the server side. As an application-layer protocol of the TCP/IP protocol stack, the ftp protocol also provides for data format translation between the two computers. Apart from translation of character encoding, ftp also must translate the encoding of end-of-line indications in text files: in Windows operating system, the end-of-line marker consists of two control characters (CR-LF), while in unix and linux operating systems, the end-of-line marker is represented by a single \n (newline) character (hence text files must be transferred in the ASCII mode); binary files need no translation during transfer (hence binary files must be transferred in binary mode).

Use the ftp application on your personal computer to connect to the server and try out the ftp application commands that are listed in the lower table. Use the »debug« option to see which ftp protocol data units correspond to different user interface commands.

command	**functionality**
ascii	set transfer mode to ACII (default)
binary	set transfer mode to binary
cd *directory*	change remote working directory
debug	set debugging mode on/off (default is off)
delete *file*	delete remote *file*
dir	display the contents of remote working directory
disconnect	close ftp session
get *file*	copy *file* from server to client machine

	(both working dirs)
lcd *directory*	change working directory on local machine
mkdir *directory*	create a new *directory*
open *server*	open a new ftp session, connecting to *server*
put *file*	copy *file* from client to server machine (both working dirs)
pwd	display the name of remote working directory
quit	close a session end exit from application

Find or generate a simple text file and a simple »jpeg« image and copy them into the working directory on your PC. Transfer them to Linux machine and back to PC in both correct and incorrect transfer mode and control file integrity and length!

- - - - - - - - - - - - -

Za prenos datotek med odjemalcem in strežnikom uporabljamo posebne protokole aplikacijskega sloja, ki podpirajo aplikacije za prenos datotek. Eden izmed najpogosteje uporabljanih protokolov za podporo prenosu datotek v Internetu je FTP (ang. File Transfer Protocol). Sistem za prenos datotek sestoji iz dveh računalnikov, enega odjemalca in enega strežnika. V obeh nas bo tu zanimala komunikacija, ki poteka v dveh slojih. V zgornjem sloju teče porazdeljena aplikacija, ki jo predstavljata dve različni aplikaciji v odjemalskem oziroma strežniškem računalniku (aplikacija v strežniku je najpogosteje sestavni del strežniškega operacijskega sistema); delovanje te porazdeljene aplikacije pogosto krmili človek-uporabnik, ki z aplikacijo na strani odjemalca interagira preko uporabniškega vmesnika. V

spodnjem sloju pa teče protokol FTP; protokolni osebek ftp sestavljata dva dela. En del predstavlja krmilna funkcija (control function), ki krmili sejo ftp, v kateri lahko uporabnik pregleduje datotečni sistem na strežniku, ga spreminja (npr. preimenuje ali briše datoteke), ali pa prenaša datoteke med odjemalcem in strežnikom; vsekakor pa mora ob vzpostavljanju seje strežnik overoviti uporabnika s pomočjo njegovega uporabniškega imena in gesla. Jezik, v katerem uporabnik preko uporabniškega vmesnika interagira z aplikacijo, se običajno razlikuje od protokola aplikacijskega sloja, ki teče med osebkoma ftp (ni pa to nujno); seveda je pa ta jezik odvisen od aplikacije na odjemalski strani. Drugi del protokolnega osebka ftp pa predstavlja prenosna funkcija, ki krmili prenos datotek med datotečnima sistemoma obeh računalnikov. Za krmiljenje seje oziroma prenos datotek uporabljamo dva različna kanala, ki ga implementirata dve ločeni zvezi TCP (krmilna zveza uporablja na strani strežnika številko vrat 21, podatkovna zveza pa številko vrat 20). Kot protokol aplikacijskega sloja protokolnega sklada TCP/IP protokol ftp po potrebi izvaja tudi pretvarjanje formata podatkov. Poleg pretvarjanja kode, ki se uporablja za zapis znakov, ftp prevaja tudi oznake, s katerimi so v tekstovni datoteki označeni konci vrstic: v operacijskem sistemu Windows v ta namen uporabljamo zaporedje dveh krmilnih znakov ASCII (CR in LF), v operacijskih sistemih unix in linux pa en sam znak \n (»newline«), zato moramo tekstovne datoteke vedno prenašati v načinu ASCII; binarne datoteke moramo prenašati v binarnem načinu, saj v tem primeru ni potrebna nikakršna pretvorba.

Uporabite aplikacijo ftp na vašem osebnem računalniku in vzpostavite sejo ftp s strežnikom; preizkusite ukaze aplikacije ftp, ki jih prikazuje naslednja tabela. Uporabite tudi opcijo

»debug« in opazujte, katera protokolna sporočila ftp implementirajo ukaze aplikacije ftp.

ukaz	**funkcionalnost**
ascii	prenosni način naj bo ASCII (privzeto)
binary	prenosni način naj bo binary
cd *directory*	spremeni delovni direktorij na strežniku
debug	vključi ali izključi opcijo “debug” (privzeto je izključena)
delete *file*	izbriši datoteko *file* na strežniku
dir	prikaži vsebino delovnega direktorija na strežniku
disconnect	zaključi sejo ftp
get *file*	kopiraj datoteko *file* s strežnika na odjemalca (delovni dir)
lcd *directory*	spremeni delovni direktorij na lokalnem računalniku
mkdir *directory*	ustvari novi direktorij z imenom *directory*
open *server*	vzpostavi novo sejo ftp s strežnikom *server*
put *file*	kopiraj datoteko *file* s klienta na strežnik (delovna dir)
pwd	prikaži ime delovnega direktorija na strežniku
quit	zaključi sejo ftp in zapri aplikacijo

Najdi ali ustvari preprosto tekstovno datoteko in preprosto sliko v formatu »jpeg« in ju kopiraj v delovni direktorij osebnega računalnika. Prenesi ju na strežnik in nazaj na PC v pravilnem in nepravilnem načinu prenosa ter kontroliraj pravilnost in dolžino prenešenih datotek!

top

World Wide Web and HTML

Svetovni splet in HTML

The World Wide Web (WWW) is a network of information resources that reside on servers spread throughout the world and are reachable via some application-layer protocols of the TCP/IP protocol stack. The task of these protocols is to transfer a resource (in a form of a file) from the server to the client computer; however, it is up to the software on the client computer to interprete the resource and to decide what to do with it. Any information resource on the web is uniquely indicated with the address referred to as the Unified Resource Locator (URL). The general form of the URL is

protocol://*server*/*resource*

or

protocol://*server*/*resource*#*bookmark*

or

protocol:*user*@*server*

where *protocol* indicates the application-layer protocol that is to be used to retrieve the resource and hence also the type of resource; *server* indicates the server where the resource resides in the form of domain name or IP address; *resource* indicates the location on the server (usually the file within the file system of the server) where the resource is stored; *bookmark* indicates a place within a document which acts as a

target of a link; *user* indicates the user of the server who owns the resource (e. g. mailbox or IP telephone).

The most usual resource on the web is a web document. Most usually, it is accessed by the protocol http (HyperText Transfer Protocol). The applications that implement the World Wide Web on server and client sides are http server (or web server) and web browser, respectively. The usual task of a web browser is to retrieve (download) a web document from the web server, as requested by a human user, and to present it on the user's terminal in the way appropriate for the type of the information. Of course, it is also necessary to publish (upload) web documents on the server; however, most of the web traffic is in the server-to-client direction. The most usual type of a web document is a hypertext document which consists of text and other types of multimedia information (e.g. image or video) and can also contain links to other documents or even links between different places within the same document. The classical language in which a hypertext document is described is the HTML (HyperText Markup Language) which is standardised by the organisation W3C (World Wide Web Consortium). HTML documents are written as plain text and can hence be edited with simple text editors.

Use a notepad editor and the HTML language to define a simple web document on your local personal computer. This document shall contain at least a title, a header, some regular text, an inserted figure, and a link. If necessary, you can learn the HTML language from a short tutorial, the David Ragget's tutorial, HTML specification at W3C web page, and/or other sources.

- - - - - - - - - - - - -

Svetovni splet (ang. World Wide Web – WWW) je omrežje informacijskih virov, ki so shranjeni na strežnikih širom sveta in dostopni s pomočjo nekaterih protokolov aplikacijskega sloja protokolnega sklada TCP/IP. Naloga teh protokolov je prenesti informacijski vir v obliki datoteke s strežniškega na odjemalski računalnik; interpretacija in uporaba tega vira pa sta stvar programske opreme na odjemalskem računalniku. Vsak informacijski vir na spletu enolično označuje naslov, ki ga imenujemo naslov vira v enotni obliki (ang. Uniform Resource Locator – URL). URL ima splošno obliko

protocol://*server*/*resource*

or

protocol://*server*/*resource*#*bookmark*

or

protocol:*user*@*server*

kjer oznaka *protocol* označuje protokol aplikacijskega sloja, s katerim informacijski vir pridobimo s strežnika, s tem pa označuje tudi tip vira; oznaka *server* označuje strežnik, na katerem je vir shranjen, v obliki domenskega imena ali naslova IP; oznaka *resource* označuje mesto na strežniku, kjer je vir shranjen (običajno datoteko v datotečnem sistemu strežnika); oznaka *bookmark* označuje mesto znotraj spletnega dokumenta, ki predstavlja cilj povezave; oznaka *user* označuje uporabnika strežnika, ki je lastnik vira (npr. poštnega predala ali telefona IP).

Najpogostejši informacijski vir v svetovnem spletu je spletni dokument. Do njega najpogosteje dostopamo po protokolu http (ang. HyperText Transfer Protocol). Aplikaciji, ki implementirata svetovni splet na strani strežnika in odjemalca,

imenujemo strežnik http (spletni strežnik) oziroma spletni brskalnik. Običajna naloga spletnega brskalnika je, da v skladu z zahtevami uporabnika prenese spletni dokument s strežnika na odjemalca (ang. download) in ga prikaže na uporabniškem terminalu na način, ki ustreza tipu informacije, vsebovane v dokumentu. Seveda je včasih tudi potrebno prenesti spletne dokumente z odjemalca na strežnik (ang. upload), vendar pa se velika večina spletnega prometa odvija od strežnika proti odjemalcu. Najpogostejši tip spletnega dokumenta je hipertekstni dokument, ki poleg besedila vsebuje še druge tipe multimedijske informacije (npr. slike in video), poleg tega pa vsebuje povezave (ang. link) med različnimi spletnimi dokumenti in tudi povezave med različnimi mesti znotraj istega dokumenta. Klasični jezik, v katerem opisujemo hipertekstne dokumente, je HTML (ang. HyperText Markup Language), ki ga je standardizirala organizacija W3C (World Wide Web Consortium). Spletni dokumenti so v jeziku HTML opisani v obliki navadnega besedila, ki ga lahko urejamo s tekstovnimi urejevalniki.

Uporabite tekstovni urejevalnik notepad (beležnica) in v jeziku HTML definirajte preprost spletni dokument na osebnem računalniku. Ta dokument naj vsebuje vsaj naslov poglavja, nekaj navadnega besedila, vstavljeno sliko in kakšno povezavo. Po potrebi si pomagajte s preprostim opisom jezika, opisom Davida Ragget-a, specifikacijami jezika HTML na straneh W3C in/ali z drugimi spletnimi ali tiskanimi viri.

top

Publishing a web page on the server

Objava spletne strani na strežniku

Publish your web page on the server protokoli.fe.uni-lj.si! The entry document must be stored as the file index.htm in the subdirectory public_html of your home directory. The directory must be readable and executable by anybody, and all web documents must be readable by anybody. Text files must be transferred in ASCII mode, and binary files must be transferred in binary mode! So, the web page of the user smithj will be accessible under the URL http://protokoli.fe.uni-lj.si/users/smithj/public_html/index.htm or just http://protokoli.fe.uni-lj.si/~smithj/, as by default the web server will search for the file index.htm in the subdirectory public_html of the user's home directory as the user's default web page.

web pages 2011/2012

- - - - - - - - - - - - -

Objavite svojo spletno stran na strežniku protokoli.fe.uni-lj.si! Vstopna stran mora biti shranjena v poddirektoriju vašega domačega direktorija public_html pod imenom index.htm. Direktorij mora biti berljiv in eksekutabilen za kogarkoli, spletni dokumenti morajo biti berljivi za kogarkoli. Tekstovne datoteke prenesite na strežnik v ASCII načinu, binarne datoteke pa v binarnem načinu! Spletna stran uporabnika kovacj bo tako dostopna prek URL-ja http://protokoli.fe.uni-lj.si/users/kovacj/public_html/index.htm ali krajše http://protokoli.fe.uni-lj.si/~kovacj/, saj spletni strežnik na strežniku protokoli.fe.uni-lj.si privzeto išče spletno stran tega uporabnika v datoteki index.htm v poddirektoriju public_html domačega direktorija tega uporabnika.

web pages 2011/2012

top

Electronic Mail System

Sistem elektronske pošte

Electronic mail represents a typical store-and-forward communication system which operates asynchronously. The first term means that electronic mail that is to be conveyed between different users is temporarily stored and relayed by mail servers; asynchronism means that mail transfer between mail sender and server is not simultanuous and not carried out with the same speed as between server and mail recipient. Here, we will consider a simple mail system consisting of a sending client, mail server and a receiving client. To transfer a mail from a client to a mailbox on a server, the SMTP (Simple Mail Transfer Protocol) application layer protocol is used over a TCP connection that uses the port number 25 at the server side. To transfer a mail from a mailbox on a server to a client, the POP3 (Post Office Protocol version 3) application layer protocol can be used over a TCP connection that uses the port number 110 at the server side. A mail box on a server is indicated by an address of the form *user@server*.

In our experiments we will transfer mail from one client to the server and from the server to another client. The two different clients will be implemented by two different students who will manually type SMTP and POP3 protocol data units (both these protocols are character oriented). A TCP connection between a client and a user to transfer characters will be set up using the telnet application with the following arguments:

`telnet` *server port*

where the *server* argument indicates the server computer (its domain name or IP address) and the *port* argument indicates the port number at which the application-layer server is reachable. More specifically, we will use the »telnet protokoli.fe.uni-lj.s 25« command to set up a TCP connection with the SMTP server and the »telnet protokoli.fe.uni-lj.s 110« command to set up a TCP connection with the POP3 server.

- - - - - - - - - - - - -

Sistem elektronske pošte je tipičen sistem, ki deluje asinhrono in po načelu »shrani in posreduj«. Asinhronost delovanja pomeni, da komunikacija med enim odjemalcem in strežnikom ter strežnikom in drugim odjemalcem ne poteka niti sočasno niti z enako hitrostjo. Načelo »shrani in posreduj« pa pomeni, da strežnik informacijo, ki jo prejme od enega odjemalca, shrani in kasneje posreduje naprej. Tu bomo obravnavali preprost sistem elektronske pošte, ki ga sestavljajo oddajni odjemalec, poštni strežnik in sprejemni odjemalec. Za pošiljanje elektronske pošte z odjemalca na strežnik uporabimo protokol aplikacijskega sloja SMTP (ang. Simple Mail Transfer Protocol); ta uporablja kanal v obliki TCP zveze, ki na strani strežnika uporablja standardno številko vrat 25. Za pošiljanje elektronske pošte s strežnika do odjemalca uporabimo protokol aplikacijskega sloja POP3 (ang. Post Office Protocol version 3); ta uporablja kanal v obliki TCP zveze, ki na strani strežnika uporablja standardno številko vrat 110. Poštni predal uporabnika na strežniku označuje naslov v obliki *user@server*.

V naših eksperimentih bomo prenašali elektronsko pošto od enega odjemalca v poštni predal na strežniku in od tod do drugega odjemalca. Oba odjemalca bosta implementirala dva

študenta, ki bosta ročno tipkala protokolna sporočila po protokolu SMTP oziroma POP3 (oba protokola sta namreč znakovno orientirana). Zvezo TCP med odjemalcem in strežnikom, prek katere se bodo prenašala zaporedja znakov, bomo vzpostavili s pomočjo aplikacije telnet z naslednjima argumentoma:

`telnet` *server port*

kjer argument *server* označuje strežniški računalnik (z njegovim domenskim imenom ali naslovom IP), argument *port* pa označuje številko vrat, na katerih je dosegljiv strežnik aplikacijskega sloja. Če povemo bolj konkretno, zvezo TCP s strežnikom SMTP bomo vzpostavili z ukazom »telnet protokoli.fe.uni-lj.si 25«, zvezo TCP s strežnikom POP3 pa z ukazom »telnet protokoli.fe.uni-lj.si 110«.

top

Electronic Mail Format

Format elektronskega sporočila

The electronic message (letter) must be composed in the standardised way, as specified in the standard RFC 822. Here, only a very simple description of the format will be given. The message consists of two parts, namely the message header and the message body, the two of them being separated by an empty line. The header itself consists of several lines, defining the date/time the message was sent, the sender and receiver of the message and the subject; while the above specifications are mandatory, according to the standard (although many servers do not check the validity or even the presence of header),

other optional lines can be added. The message body consists of one or more lines of text. Although the electronic message was originally strictly character-oriented, non-alphanumeric contents can also be encoded into character-like contents and sent as a mail according to newer standards. The format of the header will be explained here with an example (which includes the DATA keyword and the terminating period):

```
DATA
Date: 04 Nov 11 18:07 +0100
From: Drago.Hercog@fe.uni-lj.si
To: protokoli@protokoli.fe.uni-lj.si
Subject: synopsis of message

The user's message
comes here...
.
```

Most of the above is self-explanatory; the only item that needs explanation is the time zone specification (+0100) which defines the time zone (+1 hour and 0 minutes in the example) that allows the universal time to be specified.

- - - - - - - - - - - - -

Elektronsko sporočilo mora biti oblikovano v skladu s standardom RFC 822. Vendar pa bomo tukaj podali le poenostavljen opis tega formata. Sporočilo sestavljata dva dela, namreč glava in telo, loči pa ju prazna vrstica. Tudi samo glavo tvori več vrstic, ki definirajo datum/čas pošiljanja sporočila, pošiljatelja in prejemnika sporočila ter predmet sporočila; medtem ko so naštete specifikacije po standardu obvezne (čeprav mnogi strežniki pravilnosti in celo prisotnosti glave sploh ne preverjajo), lahko dodamo v glavo še dodatne

opcijske specifikacije. Telo sporočila sestavlja ena ali več vrstic. Medtem ko so bila po prvotnem standardu elektronska sporočila strogo znakovno orientirana, lahko po novejših standardih v elektronska sporočila vključujemo tudi binarne vsebine, če so te vsebine kodirane v format, ki je podoben zaporedju znakov. Format glave elektronskega sporočila bomo najlažje pojasnili s primerom, ki vključuje tudi ključno besedo DATA in zaključno piko:

```
DATA
Date: 04 Nov 11 18:07 +0100
From: Drago.Hercog@fe.uni-lj.si
To: protokoli@protokoli.fe.uni-lj.si
Subject: predmet sporočila

Tu zapišemo
elektronsko sporočilo...
.
```

Večina gornjega primera je jasna sama po sebi; dodatno je treba pojasniti le navedbo časovnega pasu (v primeru +0100, torej + 1 ura 0 minut), ki omogoča specifikacijo univerzalnega časa.

SMTP Protocol

Protokol SMTP

The protocol SMTP (Simple Mail Transfer Protocol) is a character-oriented protocol; this means that SMTP protocol data units are written as sequences of characters. Normally, end-of-line markers also delimit protocol data units that follow one another. An exception is the protocol data unit

DATA which is an information protocol data unit (it contains a user message, and this can consist of more than one line of text). Therefore, the protocol data unit DATA is always concluded with a single period in a line, hence with the sequence CR-LF-».«-CR-LF, where the sequence CR-LF indicates the end-of-line. However, it may well happen that the user message (electronic mail) contains this same sequence; therefore, a single period in a line is always added another period by the SMTP transmitter, and the SMTP receiver removes this additional period.

Use the telnet application to establish a TCP connection between the local computer and port 25 on the server protokoli.fe.uni-lj.si. Whatever you then type on the keyboard is transferred through the TCP connection to the SMTP server on the computer protokoli.fe.uni-lj.si. Your messages are interpreted by the SMTP server as SMTP protocol data units; the server executes the operations you request and replies to them. Thus, you shall send an electronic message (formatted according to the description in the previous paragraph) to your colleague's mailbox on the server protokoli.fe.uni-lj.si.

A client and a server communicate according to the SMTP protocol by exchanging commands (client) and replies (server). The commands of a client are composed of keywords and parameters, while the replies of a server are composed of number codes and textual explanations. Commands of a client and replies of a server in the normal course of communication are shown in the table and are used in that same order. Each command (except for DATA command) is exactly one line long. While the commands are written in the standard (RFC 822) exactly as given in this table (hence all uppercase letters and command arguments within angle brackets »<« and »>«), it depends on the server, how exactly it requires the format of

commands to comply with standard; our server is quite tolerant in this respect, as it alows both lowercase and uppercase letters, and even the angle brackets may be omitted. In any case, one must be aware that such protocols are normally implemented with software and are therefore not so tolerant to errors as man-machine interfaces. We have already mentioned that server's replies contain textual explanations; however, only numeric codes are standardised, so the explanations can differ from server to server, and can even be given in different languages.

client commands	**server replies**
connection setup	220
HELO <*client_domain*>	250
MAIL FROM: <*sender_address*>	250
RCPT TO: <*recipient_address*>	250
DATA	354
user message, possibli in more than one line	
. (a single period within a line)	250
QUIT	connection release

- - - - - - - - - - - - -

Protokol SMTP (ang. Simple Mail Transfer Protocol) je znakovno orientiran protokol, kar pomeni, da so protokolna sporočila SMTP zapisana kot zaporedja znakov. Praviloma konci vrstic predstavljajo tudi ločila med zaporednimi

sporočili SMTP. Izjemo predstavlja protokolno sporočilo DATA, ki je informacijsko protokolno sporočilo, torej nosi uporabniško sporočilo, ki ga lahko sestavlja več vrstic, zato se protokolno sporočilo DATA zaključi z eno samo piko v vrstici, torej z zaporedjem CR-LF-».«-CR-LF, kjer zaporedje krmilnih ASCII znakov CR-LF predstavlja konec vrstice. Ker pa se prav lahko zgodi, da uporabniško sporočilo (elektronsko pismo) vsebuje prav tako zaporedje, oddajni osebek SMTP eni sami piki na začetku vrstice doda še eno piko, sprejemni protokolni osebek SMTP pa zaporedju dveh pik na začetku vrstice odvzame eno piko.

S pomočjo aplikacije telnet vzpostavite TCP zvezo med lokalnim računalnikom in vrati 25 na strežniku protokoli.fe.uni-lj.si. Kar koli boste potem tipkali na tipkovnici lokalnega računalnika, se bo preneslo preko te TCP zveze do strežnika SMTP na računalniku protokoli.fe.uni-lj.si. Vaša sporočila bo strežnik SMTP interpretiral kot sporočila protokola SMTP, izvajal potrebne operacije v skladu s temi sporočili in nanja odgovarjal. Na ta način boste poslali elektronsko sporočilo (oblikovano v skladu s prejšnjim razdelkom) v poštni predal svojega kolega / svoje kolegice na strežniku protokoli.fe.uni-lj.si.

Po protokolu SMTP odjemalec in strežnik komunicirata z izmenjavo ukazov (odjemalec) in odgovorov (strežnik). Ukaze odjemalca sestavljajo ključne besede s parametri, odgovore strežnika pa številske kode s tekstovnimi pojasnili. Ukazi odjemalca in odgovori strežnika pri normalnem poteku komunikacije so prikazani v tabeli in jih tudi uporabimo v tem istem vrstnem redu. Vsak ukaz odjemalca (razen ukaza DATA) je dolg točno eno vrstico. Pri tem naj še pripomnimo, da so v standardu (RFC 822) ukazi napisani točno tako, kot jih vidimo v spodnji tabeli, torej s samimi velikimi črkami in

vključno z znakoma »<« in »>«, ki ograjujeta argumente ukazov; od strežnika pa je odvisno, kako natančno se tega standarda drži (naš strežnik protokoli.fe.uni-lj.si npr. je v tem pogledu dokaj prizanesljiv, saj poleg velikih črk dopušča tudi male, ne moti pa ga tudi, če znaka »<« in »>« izpustimo. Vsekakor pa se moramo zavedati, da so tovrstni protokoli namenjeni programski implementaciji in zato niso tako tolerantni do napak, kot uporabniški vmesniki, ki so namenjeni komunikaciji človek-stroj. Prej smo omenili, da odgovori strežnika poleg numeričnih kod vsebujejo še tekstovna pojasnila; ker pa so standardizirane le numerične kode, tekstovna pojasnila pa ne, so lahko le-ta pri različnih strežnikih različna in zapisana celo v različnih jezikih.

ukaz klienta	**odgovor strežnika**
vzpostavitev zveze s strežnikom	220
HELO <*domena_klienta*>	250
MAIL FROM: <*naslov_pošiljatelja*>	250
RCPT TO: <*naslov_prejemnika*>	250
DATA	354
uporabniško sporočilo, lahko v več vrsticah	
. (ena sama pika v vrstici)	250
QUIT	prekinitev zveze

top

POP3 Protocol

Protokol POP3

The protocol POP3 (Post Office Protocol, version 3) allows a user to manage the mailbox on a distant mail server and to transfer electronic messages from server to client computer. This protocol is normally used by software, not human users, and is therefore not very tolerant to user errors.

Similarly as in case of the protocol SMTP, POP3 protocol data units are lines of text which can therefore be very easily generated on keyboard and read on display. We will transfer these messages through a TCP connection between client and server computers with the port number 110 on the server side. To set up this connection, telnet application will be used.

Use the telnet application to establish a TCP connection between your local computer and the port 110 on the server protokoli.fe.uni-lj.si. Whatever you type on the keyboard of your local computer will be transferred through this connection to the POP3 server on the protokoli.fe.uni-lj.si computer. Your messages will be interpreted by the POP3 server as POP3 protocol data units; the server will execute necessary operations and reply to your messages. Server replies will be displayed on your local computer. In this way, you will examine the massages in your mailbox on the server, transfer them to your local computer and read them there.

According to the POP3 protocol, a client and a server communicate by exchanging commands (client) and responses (server). A client command consists of a keyboard and possibly a parameter. There are two possible server responses - "+OK" (success) and "-ERR" with textual explanation

(error). A POP3 session consists of three phases. In the first phase (Authorisation), the client is authenticated with a username/password pair for the user whose mailbox is to be examined; the server then locks the mailbox to prevent other users to access it. In the second phase (Transactions), the client can examine the mailbox and transfer electronic messages to the local computer. In the third phase (Update), the server deletes the messages marked for deletion, unlocks the mailbox and releases the connection. Client commands, server actions and the respective session phases are shown in the table.

phase	**client command**	**server operation**
	TCP connection setup	greet the client and go to phase A
A	`USER` *name*	username of the mailbox owner
A	`PASS` *geslo*	password of the mailbox owner; if correct, go to phase T
T	`STAT`	return the number of messages and mailbox size
T	`LIST` *n*	return the number and size of message No. *n*
T	`LIST`	return the numbers and sizes of all messages in the mailbox
T	`RETR` *n*	transfer the message No. *n* to the client (a multiline message and the concluding ».«
T	`DELE` *n*	mark the message No. *n* for deletion

T	RSET	cancel all marks for deletion
T	QUIT	go to the phase U and disconnect

- - - - - - - - - - - - -

Protokol POP3 (Post Office Protocol, verzija 3) omogoča upravljanje s sporočili v poštnem predalu uporabnika na oddaljenem računalniku ter prenos sporočil iz poštnega predala na lokalni računalnik. Ta protokol običajno uporablja poštni agent, ne pa uporabnik neposredno.

Podobno kot pri protokolu SMTP, tudi protokolne podatkovne enote protokola POP3 predstavljajo vrstice ASCII besedila, zato lahko ta sporočila preprosto oblikujemo in beremo. Za prenos teh sporočil od lokalnega računalnika do POP3 strežnika bomo uporabili TCP zvezo med odjemalcem in strežnikom, številka vrat na strani strežnika bo 110. TCP zvezo bomo vzpostavili s pomočjo programa telnet.

S pomočjo aplikacije telnet vzpostavite TCP zvezo med lokalnim računalnikom in vrati 110 na strežniku protokoli.fe.uni-lj.si. Kar koli boste potem tipkali na tipkovnici lokalnega računalnika, se bo preneslo preko te TCP zveze do strežnika POP3 na računalniku protokoli.fe.uni-l.si. Vaša sporočila bo strežnik POP3 interpretiral kot sporočila protokola POP3, izvajal potrebne operacije v skladu s temi sporočili in nanja odgovarjal. Odgovori strežnika se bodo izpisali na zaslonu lokalnega računalnika. Na ta način boste pregledali elektronska sporočila v svojem poštnem predalu na strežniku protokoli.fe.uni-lj.si, jih prenesli na lokalni računalnik in tam prebrali.

Po protokolu POP3 odjemalec in strežnik komunicirata z izmenjavo ukazov (odjemalec) in odgovorov (strežnik). Ukaze

odjemalca sestavljajo ključne besede, ki jim lahko sledi parameter. Možna odgovora strežnika sta "+OK" v primeru uspeha in "-ERR" v primeru neuspeha s tekstovnimi pojasnili. Komunikacija med odjemalcem in strežnikom poteka v treh fazah. V prvi (avtorizacija - Authorisation) se odjemalec predstavi z imenom in geslom uporabnika, katerega poštni predal želi pregledovati, strežnik pa nato poštni predal zaklene za vse ostale možne odjemalce. V drugi fazi (transakcije - Transactions) odjemalec pregleduje pošni predal in po želji prenaša elektronska sporočila s strežnika na odjemalca. V tretji fazi (zaključek seje - Update) pa strežnik zbriše sporočila, ki so označena za brisanje, odklene poštni predal in prekine zvezo. Ukazi odjemalca, akcije strežnika in faze, v katerih se ukazi in akcije izvedejo, so prikazani v tabeli.

faza	**ukaz odjemalca**	**akcija strežnika**
	vzpostavitev TCP zveze	pozdrav odjemalcu in prehod v fazo avtorizacije
A	`USER` *ime*	uporabniško ime uporabnika poštnega predala
A	`PASS` *geslo*	geslo uporabnika poštnega predala; če je pravilno, prehod v fazo T
T	`STAT`	strežnik vrne število sporočil in velikost poštnega predala v oktetih
T	`LIST` *n*	strežnik vrne številko in velikost sporočila št. *n* v poštnem predalu
T	`LIST`	strežnik vrne številke in velikosti vseh sporočil v poštnem predalu
T	`RETR` *n*	strežnik prenese sporočilo *n* v več

		vrsticah do odjemalca; zadnja vrstica vsebuje samo znak "."
T	`DELE` *n*	strežnik označi sporočilo *n* za brisanje
T	`RSET`	vse označbe za brisanje sporočil se prekličejo
T	`QUIT`	prehod v fazo U in prekinitev zveze

top

E-Mail UserAgent

Poštni agent

On your personal computer, examine the settings of the mail user agent Windows Outlook Express!

- - - - - - - - - - - - -

Na osebnem računalniku si oglejte možne nastavitve poštnega agenta Windows Outlook Express!

top

SDL Specification of a Counting Clock

SDL opis ure, ki šteje

In the SDL language specify a clock system that counts time intervals! To create, edit and simulate the specification, use the Cinderella software!

- - - - - - - - - - - - -

V jeziku SDL specificirajte sistem ure, ki šteje časovne intervale! Specifikacijo ustvarite, uredite in simulirajte s pomočjo programske opreme Cinderella!

top

SDL Specification of a Simple Communication System

SDL opis preprostega sistema za komuniciranje

Specify the functionality of the transmitting and receiving protocol entities, which should operate according to the stop-and-wait protocol without / with sequence numbering of information protocol data units. Simulate the specification and find out why such a system without sequence numbering does not operate correctly. Use the system description and users and channel specifications contained in the file comm.cbf which was created and can be opened in Cinderella; this file is downloadable from the server.

- - - - - - - - - - - - -

Specificirajte funkcionalnosti oddajnega in sprejemnega protokolnega osebka, ki naj delujeta po protokolu s čakanjem brez / s sekvenčnim številčenjem informacijskih protokolnih sporočil. Simulirajte specifikacijo in ugotovite, kdaj opisani sistem brez uporabe sekvenčnih številk informacijskih protokolnih sporočil napačno deluje. Za opis sistema ter

specifikacijo uporabnikov in prenosnega kanala uporabite SDL datoteko comm.cbf, ki je bila ustvarjena in jo lahko odprete v programu Cinderella; to datoteko dobite na strežniku.

top

Residual Error Rate

Rezidualna pogostnost napak

The bit error rate in a physical channel is 10^{-8}. A uniform distribution of bit errors is assumed. The packet length is 1000 octets. Which is the packet error rate? 99,5 % of corrupted packets are corrected by the receiver. Which is the residual packet error rate? Which is the virtual residual bit error rate?

- - - - - - - - - - - - -

Pogostnost bitnih napak v fizičnem kanalu je 10^{-8}. Predpostavljamo, da so bitnih napake enakomerno porazdeljene. Dolžina paketov je 1000 o. Kolika je pogostnost napačno prenešenih paketov? 99,5 % napačno prenešenih paketov uspe sprejemnik popraviti. Kolikšna je rezidualna pogostnost napačno prenešenih paketov? Kolika je navidezna rezidualna pogostnost bitnih napak?

top

Two-Dimensional Even-Parity Channel Coding

Kanalsko kodiranje z dvodimenzionalno sodo pariteto

When transferring four texts encoded with the 7-bit ASCII code and channel coded with 2-dimensional even-parity coding method, zero or more errors occurred. Messages shown in the tables bellow were received. Find out which messages were corrupted and which corrupted messages can be corrected. Decode the uncorrupted and corrected messages. Parity bit is the rightmost in the byte, and parity byte is the bottommost in the message. For decoding use the ASCII table! (When solving this task try to behave as a machine, do not guess the solution and temporarily forget your excellent knowlege of English literature!)

10010000	10011010	10111001	10001110
10000010	10000010	10001011	10011001
10011010	10000101	10001010	10011111
10111001	10000100	10100101	10000100
10001011	10001011	00110101	10001011
10101001	10101001		10000111
00110011	10010000		
	10101100		

Pri prenosu štirih besedil, kodiranih s 7-bitno ASCII kodo in kanalsko kodiranih po metodi dvodimenzionalne sode paritete, je prišlo do nič ali več bitnih napak. Sprejemnik je sprejel sporočila, prikazana v zgornjih tabelah. Ugotovite, katera

sporočila so se pokvarila ter katera je mogoče enoumno popraviti in jih popravite. Nepokvarjena in popravljena sporočila dekodirajte v besedila! Paritetni bit je zadnji (desni) bit v zlogu, paritetni zlog je zadnji (spodnji) zlog v sporočilu. Pri tem uporabite ASCII tabelo! (pri reševanju naloge se poskusite obnašati kot stroj, torej ne uganjujte in začasno odmislite svoje odlično poznavanje svetovne književnosti!)

top

Checksum Method

Metoda seštevanja

Use the checksum method to protect the given sequence of seven four-bit words by adding an additional four-bit word. Simulate one or more bit errors and find out which errors can be detected by the receiver and which cannot!

```
1001

0110

1111

1011

0000

0111

1000
```

Po metodi seštevanja zaščitite niz sedmih štiri-bitnih besed z dodatno štiri-bitno besedo! Simulirajte eno ali več napak pri

prenosu in ugotovite, katere napake lahko sprejemnik odkrije in katerih ne more!

top

Cyclic Redundancy Check (CRC)

Metoda krožnega redundančnega preizkusa (CRC)

Use the Cyclic Redundancy Check method to protect the binary sequence `1001011001100101`! Generator polynomial is `10101`. Which sequence is transmitted? Imitate the receiver's verification in case of no corruption / corruption.

- - - - - - - - - - - - -

Po metodi krožnega redundančnega preizkusa zaščitite niz binarnih vrednosti `1001011001100101`! Generacijski polinom je `10101`. Katero binarno zaporedje je oddano? Posnemajte preverjanje sprejemnika v primeru, ko pri prenosu ni/je prišlo do napake!

top

Stop-and-Wait protocol

Protokol s čakanjem

The Stop-and-Wait Protocol is used with the connection over a physical channel with the following properties:

connection	1	2
nominal bit rate	16 kb/s	1Mb/s
user message length	60 o	60 o
overhead	8 o	8 o
acknowledgment length	8 o	8 o
physical channel length	1000 km	1000 km
propagation speed	c	c

For both connections, calculate minimum timer expiration time, as well as the protocol efficiency and the efficiency seen by the users; suppose a lossless channel.

- - - - - - - - - - - - -

Protokol s čakanjem uporabljamo pri zvezi prek fizičnega kanala z naslednjimi lastnostmi:

zveza	1	2
nazivna hitrost	16 kb/s	1Mb/s
dolžina uporabniških sporočil	60 o	60 o
dolžina režije	8 o	8 o
dolžina potrditev	8 o	8 o
dolžina fizičnega kanala	1000 km	1000 km
hitrost propagacije	c	c

Za oba primera izračunajte minimalni čas izteka časovnika ter učinkovitost protokola in učinkovitost, ki jo vidita uporabnika, ob predpostavki, da v kanalu ni izgub!

top

Continuous Protocol

Protokol s tekočim pošiljanjem

Continuous protocol is used in a connection over a physical channel with the following data:

connection	**1**	**2**
nominal bit rate	16 kb/s	1Mb/s
user message length	60 o	60 o
overhead	8 o	8 o
acknowledgment length	8 o	8 o
physical channel length	1000 km	1000 km
propagation speed	c	c

For both connections, calculate the minimum transmission window width necessary to achieve the 100 % efficiency over a lossless channel, and the amount of memory necessary for such a window; calculate the efficiency seen by the users in case of such a window and lossless channel!

- - - - - - - - - - - - -

Protokol s tekočim pošiljanjem uporabljamo pri zvezi prek fizičnega kanala z naslednjimi lastnostmi:

zveza	**1**	**2**
nazivna hitrost	16 kb/s	1Mb/s
dolžina uporabniških sporočil	60 o	60 o
dolžina režije	8 o	8 o
dolžina potrditev	8 o	8 o
dolžina fizičnega kanala	1000 km	1000 km

hitrost propagacije	c	c

Za obe zvezi izračunajte minimalno širino oddajnega okna, ki je potrebna, da bo učinkovitost protokola pri brezizgubnem kanalu enaka 1, in spomin, ki je za tako okno potreben; izračunajte tudi učinkovitost, ki jo pri takem oddajnem oknu in kanalu brez izgub vidita uporabnika!

top

SDL Description of the Alternating Bit Protocol

SDL opis protokola z alternirajočim bitom

Specify in SDL language the communication system which is based on the use of Alternating Bit Protocol and is described in a textual form! Use the Cinderella software! Use the Cinderella simulator to test the specification!

- - - - - - - - - - - - -

Komunikacijski sistem, ki temelji na uporabi protokola z alternirajočim bitom in je specificiran v tekstovni obliki, specificirajte v jeziku SDL! Uporabite orodje Cinderella! Testirajte pravilnost specifikacije s simulatorjem Cinderella!

top

Efficiency of Sliding Window Protocols

Učinkovitost protokolov z drsečim oknom

The simulator swp simulates sliding window protocols and calculates the protocol efficiency, as well as the efficiency seen by users.

Simulate the communication system based on sliding window protocols with the following data:

- nominal bit rate in the physical channel R=1 Mb/s,
- propagation time in the channel T_p=200 μs (200 microseconds),
- overhead length L_r=7 octets,
- acknowledgment length L_a=7 octets.

Calculate

- the minimum transmit window width Ws_0 which will allow the transmitter to transmit information protocol data units with length L_{sdu}=20 octets into the lossless channel all the time (yielding protocol efficiency 1), and the efficiency seen by users in this case; verify your results with simulator!

Use the simulator to find out the following results and display them in manually-drawn diagrams (use the paper with printed millimetre scale):

- the dependence of protocol efficiency η on the bit error probability p_b at L_{sdu}=20 octets for protocols ABP (alternating bit protocol) (Ws=Wr=1), GBP (go-back-N protocol) (Ws=Ws_0, Wr=1) and SRP (selective repeat protocol) (Ws=Wr=Ws_0), for p_b between 10^{-6} and 10^{-3}; simulate and show results in seven points p_b equidistant on logarithmic scale;

- the dependence of the efficiency seen by users when using alternating bit protocol on the user message length L_{sdu}, for L_{sdu} running from 10 to 200 octets in steps of 10 octets, for $p_b = 10^{-6}$ and $p_b = 10^{-3}$;
- optimum user message length $L_{sdu\ opt}$ for selective repeat protocol with $p_b = 10^{-6}$, 10^{-5} , 10^{-4} and 10^{-3}.

Simulator swp/swpe.exe runs in Windows environment. You will also need the file swp/cw3220.dll in the same folder as the simulator program.

- - - - - - - - - - - - -

Simulator swp simulira protokole z drsečim oknom ter izračuna učinkovitost protokola in učinkovitost, ki jo vidita uporabnika.

Simulirajte komunikacijski sistem, ki temelji na uporabi protokolov z drsečim oknom. Pri tem uporabite naslednje podatke:

- nazivna hitrost kanala R=1 Mb/s,
- propagacijski čas v kanalu T_p=200 μs (200 mikrosekund),
- dolžina režije L_r=7 oktetov,
- dolžina potrditev L_a=7 oktetov.

Izračunajte

- širino oddajnega okna Ws_0 tako, da bo lahko oddajnik ves čas oddajal v brezizgubni kanal informacijska protokolna sporočila z L_{sdu}=20 oktetov dolgimi uporabniškimi sporočili (kar bo pomenilo učinkovitost protokola 1), ter učinkovitost, ki jo v tem primeru vidi uporabnik. Oboje preverite s simulatorjem.

S pomočjo simulatorja ugotovite ter narišite ustrezni diagram na milimetrski papir:

- odvisnost učinkovitosti protokola η od verjetnosti bitne napake p_b pri L_{sdu}=20 oktetov za protokole ABP (protokol z alternirajočim bitom) (Ws=Wr=1), GBP (protokol s ponavljanjem zaporedja) (Ws=Ws_0, Wr=1) in SRP (protokol s selektivnim ponavljanjem) (Ws=Wr=Ws_0), pri čemer naj bo p_b med 10^{-6} in 10^{-3}; simulacijo izvedite in rezultate podajte v 7 enakomerno porazdeljenih točkah p_b v logaritemskem merilu;
- odvisnost učinkovitosti, ki jo vidita uporabnika pri komuniciranju po protokolu z alternirajočim bitom, od dolžine uporabniškega sporočila L_{sdu}, ki naj bo med 10 in 200 okteti, pri p_b =10^{-6} in p_b =10^{-3}; simulacijo izvedite v korakih po 10 oktetov;
- optimalno dolžino uporabniškega sporočila $L_{sdu\ opt}$ za protokol s selektivnim ponavljanjem pri p_b = 10^{-6} , 10^{-5} , 10^{-4} in 10^{-3}.

Simulator swp/swp.exe teče v okolju Windows. Potrebujete tudi datoteko swp/cw3220.dll, ki jo prenesete na osebni računalnik v isto mapo kot simulator.

top

LAPB Protocol

Protokol LAPB

Show and indicate the interchange of LAPB frames between LAPB protocol entities A and B. The protocol entity A sends to B 6 information frames (numbered 0÷5), and the protocol

entity B sends to A 7 information frames (numbered 0÷6). The frame No. 4 from B to A is corrupted, and the frame No. 2 from A to B is corrupted. The LAPB frames shall be specified in the form *I,N(S),N(R)*, *RR,N(R)*, and *REJ,N(R)*, respectively. All information (*I*) frames shall be equally long (5 divisions in the solution form), and all supervisory (*RR* and *REJ*) frames shall be 2 divisions long. The time needed for processing in protocol entities shall be neglected. Transmit window widths of both protocol entities shall be 7 frames. At initialisation time, all user packets to be transferred are already in the input queues of protocol entities. If a frame reception and a frame transmission appear to occur at the same sime, the reception is assumed to have occurred before the transmission. Both protocol entities transmit acknowledgments as soon as possible.

solution form

- - - - - - - - - - - - -

Narišite in označite medsebojno pošiljanje okvirjev med osebkoma A in B po protokolu LAPB, če osebek A pošlje osebku B 6 informacijskih okvirjev (oštevilčenih 0÷5) in osebek B pošlje osebku A 7 informacijskih okvirjev (oštevilčenih 0÷6); osebek A odkrije napako v sprejetem okvirju št. 4, osebek B pa v sprejetem okvirju št. 2. Okvirje označujte v formatu *I,N(S),N(R)*, *RR,N(R)*, *REJ,N(R)*. Vsi informacijski okvirji (*I*) naj bodo enako dolgi (5 razdelkov v obrazcu za reševanje), nadzorni okvirji (*RR* in *REJ*) pa naj bodo dolgi po 2 razdelka. Čas procesiranja zanemarimo. Širina oddajnega okna obeh osebkov je 7 okvirjev. Ob začetku delovanja so vsi uporabniški paketi, ki naj bodo prenešeni, že v vhodnih čakalnih vrstah obeh protokolnih osebkov. Če se zdi, da je osebek oddal en okvir in prejel drugega ob istem

času, predpostavimo, da se je sprejem zgodil pred oddajo. Oba protokolna osebka oddata potrditve sprejetih okvirjev takoj, ko je to mogoče.

obrazec za reševanje

top

TCP Protocol

Protokol TCP

Draw and designate TCP segments to be transferred between TCP protocol entities A and B to establish a connection, transfer 480 user octets from A to B and release the connection. Connection establishment and release shall be initiated by A. The protocol entity A shall transmit 100 octets long information segments (including TCP and IP headers, and user information), and the protocol entity B shal transmit 40 octets long cumulative acknowledgments (including only TCP and IP headers). Segments used for connection setup and release shall be 40 octets long as well (TCP and IP headers). In the figure, information segments shall be 5 divisions long, and control segments shall be 2 divisions long; propagation time shall be 3 divisions. The window width advertised by the protocol entity B shall be very large. Maximum segment size (excluding TCP and IP headers) declared by both entities shall be 60 octets. The initial sequence number of A shall be 0, and the initial sequence number of B shall be 1000. Acknowledgments shall not be delayed. The fourth information segment transmitted by the entity A shall be lost. Processing times shall be neglected. An information segment

shall be given with the abstract syntax DATA,s=*SequenceNumber*,d=*DataLength*, an acknowledgment segment shall be given with the abstract syntax ACK,a=*Acknowledgment*, connection setup segment shall be given with the abstract syntax SYN,s=*SequenceNumber*,mss=*MaximumSegmentSize* or SYN,s=*SequenceNumber*,a=Acknowledgment,mss=*MaximumSegmentSize*, and a connection release segment shall be given with the abstract syntax FIN,s=*SequenceNumber*. In the figure, you shall also indicate the initial sequence number in the transmit window of A (W_{begin}) and the transmit window width of A (W_{width}), whenever their values change. Make use of the algorithms slow start, fast retransmit and fast recovery, whenever appropriate!

solution form

- - - - - - - - - - - - -

Narišite in označite prenos segmentov za vzpostavitev zveze, prenos uporabniške informacije in sprostitev zveze med TCP osebkoma A in B, če osebek A oddaja 100 oktetov dolge informacijske segmente (vključno z IP in TCP glavo), osebek B pa le 40 oktetov dolge kumulativne potrditve (ki vsebujejo le IP in TCP glavo). Segmenti za vzpostavitev in sprostitev zveze naj bodo prav tako dolgi 40 oktetov (IP in TCP glava). Postopka za vzpostavitev in sprostitev zveze naj začne osebek A. Osebek A naj osebku B pošlje 480 uporabniških oktetov. V sliki naj bodo informacijski segmenti dolgi 5 razdelkov, nadzorni segmenti 2 razdelka, čas propagacije pa naj bo 3 razdelke. Širina okna, ki ga oglešuje osebek B, naj bo zelo velika. Maksimalna dolžina segmenta (brez IP in TCP glave) naj bo 60 oktetov. Začetna sekvenčna številka osebka A naj bo 0, začetna sekvenčna številka osebka B pa 1000. Osebka ne

odlagata potrditev. Četrti informacijski segment, ki ga odda osebek A, se pri prenosu izgubi. Čase procesiranja zanemarimo. Informacijski segment podajte z abstraktno sintakso DATA,s=*SekvenčnaŠtevilka*,d=*Dolžina_Podatkov*, potrditveni segment z abstraktno sintakso ACK,a=*Potrditev*, segment za vzpostavitev zveze z abstraktno sintakso SYN,s=*SekvenčnaŠtevika*,mss=M*aksimalnaDolžinaSegmenta* ali SYN,s=*SekvenčnaŠtevika*,a=Potrditev,mss=M*aksimalnaDolžinaSegmenta* in segment za sprostitev zveze z abstraktno sintakso FIN,s=*SekvenčnaŠtevilka*. V sliki za osebek A označite tudi začetno sekvenčno številko v oddajnem oknu (W_{begin}) in širino oddajnega okna (W_{width}), kadar se ti vrednosti spremenita. Uporabite algoritme počasnega zagona, hitre ponovitve in hitrega zdravljenja, kadar je to potrebno!

Obrazec za reševanje

top

IP Addresses, Connectivity, Response Times and Routing Paths

Naslovi IP in časi odziva

Try out the following Windows commands/applications:

comm./appl.	**find out:**
ipconfig (W)	IP configuration of your computer
ifconfig (L)	IP configuration of our server
arp (W,L)	table of translations between IP and MAC

	addresses in your computer
nslookup (W,L) host (L)	domain name of your computer; IP addresses of the following computers: protokoli.fe.uni-lj.si, www.fe.uni-lj.si, www.ibis.fr, www.ieee.org, www.sydney.com
ping (W,L)	connectivity with and average response times from the following computers: protokoli.fe.uni-lj.si, www.fe.uni-lj.si, www.ibis.fr, www.ieee.org, www.sydney.com
tracert (W) tracepath (L)	routing paths to the following computers: protokoli.fe.uni-lj.si, www.fe.uni-lj.si, www.ibis.fr, www.ieee.org, www.sydney.com

- - - - - - - - - - - - -

Preizkusite naslednje ukaze/aplikacije v operacijskem sistemu Windows:

ukaz/apl.	**ugotovite:**
ipconfig	konfiguracijo IP vašega računalnika
arp	tabelo prevodov med naslovi IP in MAC v vašem računalniku
nslookup	domensko ime vašega računalnika; naslove IP naslednjih računalnikov: protokoli.fe.uni-lj.si, www.fe.uni-lj.si, www.ibis.fr, www.ieee.org, www.sydney.com
ping	povezljivost z in povprečni odzivni čas naslednjih računalnikov: protokoli.fe.uni-lj.si, www.fe.uni-lj.si, www.ibis.fr, www.ieee.org, www.sydney.com
tracert	usmerjevalne poti do naslednjih računalnikov: protokoli.fe.uni-lj.si, www.fe.uni-lj.si, www.ibis.fr, www.ieee.org, www.sydney.com

top

Protocol Analysis with TcpDump

Protokolna analiza z aplikacijo TcpDump

We will use the TcpDump application on the Protokoli.fe.uni-lj.si server to demonstrate the operation of ping and tracepath applications. For this purpose, we will use the TcpDump application with the following options and packet filter:

- tcpdump –vvvc10 icmp

This will be a demonstration only, as runing TcpDump requires superuser privileges. The version of TcpDump on Windows is called WinDump.

- - - - - - - - - - - - -

Na strežniku protokoli.fe.uni-lj.si bomo s pomočjo aplikacije TcpDump prikazali delovanje aplikacij ping in tracepath. V ta namen bomo uporabili aplikacijo TcpDump z naslednjimi opcijami in izbirnim filtrom:

- tcpdump –vvvc10 icmp

To bo le demonstracija, saj je poganjanje TcpDump mogoče le z administratorskimi privilegiji. Verzija TcpDump pod operacijskim sistemom Windows se imenuje WinDump.

top

Protocol Analysis with WireShark

Protokolna analiza z aplikacijo WireShark

We will use the WireShark application on a Windows computer to demonstrate the operation of ping and tracert applications. For this purpose, we will use the following packet filter:

- icmp

- - - - - - - - - - - - -

Na osebnem računalniku si bomo s pomočjo aplikacije WireShark ogledali delovanje aplikacij ping in tracert. V ta namen bomo uporabili naslednji izbirni filter:

- icmp

top
