Višja strokovna šola Velenje – Informatika Murska Sobota

Računalniške komunikacije in omrežja II

VODNIK V TCP/IP

Predavanja

Druga popravljena in razširjena izdaja

Pripravil: dr. **Iztok Fister**
E-mail: iztok.fister@mdi2.net

Murska Sobota
Januar, 2009

Kazalo

1 ARHITEKTURA TCP/IP 1

1.1 ARHITEKTURNI MODEL TCP/IP 2
1.1.1 *PROTOKOLNI SKLAD TCP/IP* 3
1.1.2 *MODEL ODJEMALEC/STREŽNIK* 4
1.1.3 *MOSTOVI, USMERJEVALNIKI IN PREHODI* 5
1.2 INTERNETNI PROTOKOL 5
1.2.1 *NASLAVLJANJE IP* 5
1.2.2 *NASLOV IP* 6
1.2.3 *POSEBNI NASLOVI IP* 7
1.2.4 *PODOMREŽJA IP* 8
1.2.5 *USMERJANJE IP* 10
1.2.6 *USMERJEVALNA TABELA IP* 10
1.2.7 *USMERJEVALNI ALGORITEM IP* 12
1.2.8 *METODE DOSTAVE – UNICAST, BROADCAST, MULTICAST IN ANYCAST* 12
1.2.9 *INTRANET* 14
1.2.10 *DATAGRAM IP* 14
1.3 ICMP 18
1.3.1 *APLIKACIJE ICMP* 18
1.4 IGMP 20
1.5 ARP 20
1.5.1 *GENERIRANJE NASLOVOV ARP* 20
1.5.2 *KONCEPT PROXY-ARP* 20
1.6 RARP 21
1.7 VRATA IN VTIČNICE 22
1.7.1 *VRATA* 22
1.7.2 *VTIČNICE* 23
1.8 UDP 23
1.8.1 *UDP FORMAT* 24
1.8.2 *UDP API* 24
1.9 TCP 25
1.9.1 *KONCEPT TCP* 25
1.9.2 *PRINCIP DRSNEGA OKNA* 26
1.9.3 *FORMAT TCP SEGMENTA* 28
1.9.4 *NADZOR NASIČENJA* 29

2 USMERJEVALNI PROTOKOLI 33

2.1 OSNOVNO USMERJANJE IP 33
2.1.1 *USMERJEVALNI PROCESI* 34
2.1.2 *AVTONOMNI SISTEMI* 34
2.2 USMERJEVALNI ALGORITMI 35
2.2.1 *STATIČNO USMERJANJE* 35
2.2.2 *USMERJANJE NA OSNOVI VEKTORJA RAZDALJ* 36
2.2.3 *USMERJANJE NA OSNOVI STANJA POVEZAV* 37

2.3 INTERNI USMERJEVALNI PROTOKOLI 39
2.3.1 *RIP* 39
2.3.2 *RIP-2* 41
2.3.3 *OSPF* 42
2.4 ZUNANJI USMERJEVALNI PROTOKOLI 57
2.4.1 *ZUNANJI USMERJEVALNI PROTOKOL* 57
2.4.2 *MEJNI USMERJEVALNI PROTOKOL* 57

3 VARNOST V TCP/IP....61

3.1 REŠITVE OMREŽNE VARNOSTI....61
3.1.1 *IMPLEMENTACIJE VARNOSTNIH REŠITEV* 62
3.1.2 *POLITIKA OMREŽNE VARNOSTI* 63
3.2 KRIPTOGRAFIJA 63
3.2.1 *SIMETRIČNI ALGORITMI ALI ALGORITMI TAJNIH KLJUČEV* 65
3.2.2 *ASIMETRIČNI ALGORITMI ALI ALGORITMI JAVNIH KLJUČEV* 65
3.2.3 *ZGOŠČEVALNA FUNKCIJA* 67
3.3 POŽARNI ZID 71
3.3.1 *KOMPONENTE POŽARNEGA ZIDU* 72
3.3.2 *USMERJEVALNIK S FILTRIRANJEM PAKETOV* 72
3.3.3 *PREHOD PROKSI* 74
3.3.4 *PREHOD NA NIVOJU VODA* 75
3.3.5 *PRIMERI POŽARNIH ZIDOV* 76
3.4 PREVAJANJE OMREŽNIH NASLOVOV IP 80
3.4.1 *MEHANIZEM PREVAJANJA* 81
3.4.2 *OMEJITVE NAT* 82
3.5 VARNOSTNA ARHITEKTURA IP....83
3.5.1 *AVTENTIKACIJSKA GLAVA* 85
3.5.2 *VAROVANJE KORISTNE VSEBINE Z OVIJANJEM* 87
3.5.3 *KOMBINIRANJE PROTOKOLOV IPSEC* 91
3.5.4 *PROTOKOL INTERNETNE IZMENJAVE KLJUČEV* 96
3.6 PLAST VARNIH VTIČNIC 98
3.6.1 *PROTOKOL SSL* 100
3.7 SECURE MULTIPURPOSE INTERNET MAIL EXTENSION....104
3.8 NAVIDEZNO ZASEBNO OMREŽJE 104
3.8.1 *UVOD V VPN IN PRAKTIČNA UPORABNOST* 104
3.8.2 *REŠITVE VPN NA TRŽIŠČU* 105
3.9 AVTENTIKACIJSKI PROTOKOLI ZA ODDALJENI DOSTOP 105
3.10 TUNELSKI PROTOKOL NA DRUGI PLASTI 107
3.10.1 *TERMINOLOGIJA* 108
3.10.2 *ZNAČILNOSTI PROTOKOLA* 108
3.10.3 *VARNOSTNI PROBLEMI L2TP* 110

DODATEK 1: DIJKSTROV ALGORITEM ISKANJA NAJKRAJŠE POTI 111

DODATEK 2: EULERJEVI GRAFI....113

LITERATURA....115

1 ARHITEKTURA TCP/IP

Danes so omrežja postala najpomembnejši del sodobnih informacijskih sistemov (krajše IS). Predstavljajo hrbtenico za izmenjavo informacij v podjetjih, vladnih organizacijah in raziskovalnih skupinah. Te informacije, ki jih obdelujemo z računalniki, lahko prenašamo v obliki sporočil, dokumentov ali podatkov.

Večina omrežij, nastalih v zgodnjih 60-ih in 70-ih letih, je bilo izvedenih na zapleten način z različnimi tehnologijami. V 70-ih se je pojavila potreba po medsebojnem povezovanju aplikacij, ki so tekle na različnih IS. Ta potreba je izpostavila zahtevo po standardizaciji komunikacijske opreme. Različni proizvajalci so predstavili celo paleto komunikacijskih protokolov, za katere je značilna plastnost funkcionalnih sklopov. Rezultat tega je, da imamo danes omrežne rešitve na različnih tehnologijah, od poceni asinhronih linij brez odpravljanja napak pri prenosu (angl. error recovery) pa do dragih lastniških omrežij SNA podjetja IBM. Če želimo izmenjavo informacij med dvema aplikacijama na različnih IS, morata obe podpirati isti protokol. V začetku 70-ih let se je v ZDA pod imenom ARPANET pojavila skupina raziskovalcev, ki se je ukvarjala z novim komunikacijskim pojmom, t.j. medmrežjem (internetworking). Ta skupina je poskušala definirati množico protokolov, razdeljenih v funkcionalne plasti tako, da bi aplikacija lahko komunicirala z drugo aplikacijo, ne glede na kakšni omrežni tehnologiji ali operacijskem sistemu tečeta.

Združenje raziskovalcev ARPANET je leta 1971 z delovanjem prenehalo. Začeto delo je nadaljevalo združenje DARPA, ki je na podlagi protokola NCP (angl. Network Control Program) leta 1978 razvila protokol TCP/IP v svoji sedanji podobi. Prva implementacija spletnega omrežja (Internet) je bila okrog leta 1980, ko je DARPA začela opremljati računalnike v raziskovalnem omrežju ARPANET z novim protokolom TCP/IP. Ta postopek je trajal tri leta. Združenje DARPA je z univerzo Berkeley v Kaliforniji sodelovala tudi pri razvoju protokola TCP/IP za operacijski sistem UNIX na računalnikih VAX. Univerza Berkeley je te računalnike opremljala z brezplačno distribucijo svojega operacijskega sistema UNIX BSD. Protokol TCP/IP se je tako hitro razširil med univerzami in raziskovalnimi ustanovami. S širjenjem protokola TCP/IP so se posamezna podomrežja po vsem svetu začela povezovati v skupno povezano omrežje. Rezultat tega povezovanja je nastanek sodobnega spletnega omrežja.

Splet sestoji iz naslednjih vrst omrežij (Parziale, 2006):

- hrbtenice: velika omrežja, ki služijo povezavi podomrežij, npr. NSFNET (angl. National Science Foundation Network) v ZDA, EBONE (angl. European Backbone) v Evropi,
- regionalnih omrežij: povezujejo regionalna omrežja, npr. univerzitetno omrežje,
- komercialnih omrežij: omogočajo posameznikom ali podjetjem dostop do hrbteničnega omrežja, npr. SIOL v Sloveniji,
- lokalnih omrežij: povezujejo lokalna omrežja (angl. local area network, krajše LAN), npr. univerzitetno lokalno omrežje.

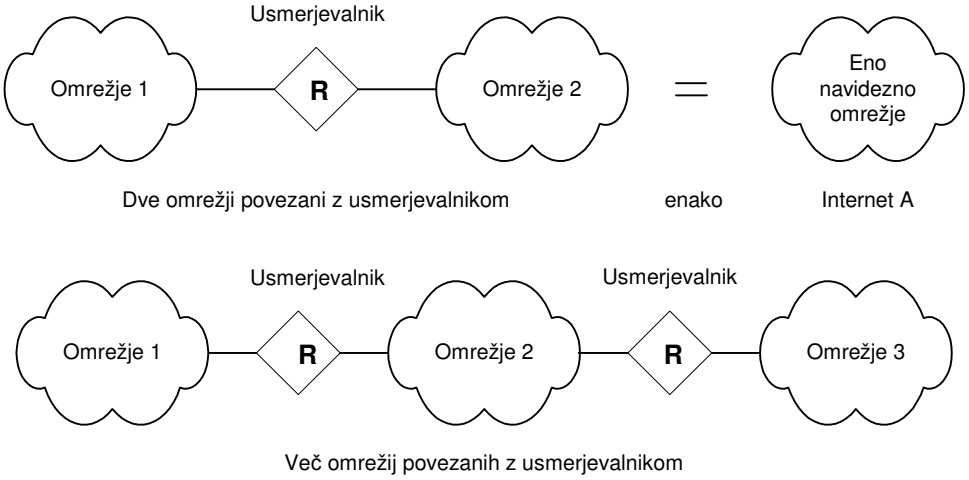
V večini primerov je prenos med temi omrežji in spletnim omrežjem omejen. Še posebej to velja za vojaška, komercialna in vladna omrežja.

1.1 Arhitekturni model TCP/IP

TCP/IP je sveženj protokolov, ki so svoje ime dobili po dveh najpomembnejših protokolih, ki delujeta na različnih funkcionalnih plasteh. To sta protokola TCP (angl. Transmission Control Protocol) in IP (angl. Internet Protocol). Cilji razvoja protokola TCP/IP so naslednji:

- izgradnja povezanega omrežja, ki omogoča univerzalne komunikacijske storitve, t.j. spletno omrežje,
- povezava različnih fizičnih omrežij v obliko, ki se končnemu uporabniku predstavlja kot eno samo veliko omrežje.

Za povezavo dveh omrežij potrebujemo napravo, ki je povezana na obe omrežji in lahko pošilja pakete iz enega omrežja v drugega in obratno. Taki napravi pravimo tudi usmerjevalnik (angl. router). Ker je usmerjanje funkcija omrežne plasti, na kateri deluje protokol IP, zanj uporabljamo tudi izraz usmerjevalnik IP (slika 1).



Slika 1: Spletna povezana omrežja

Osnovne lastnosti usmerjevalnika IP so:

- s stališča omrežja je usmerjevalnik navaden *gostitelj* (angl. host),
- s stališča uporabnika je usmerjevalnik neviden, t.j. uporabnik vidi eno samo veliko navidezno omrežje.

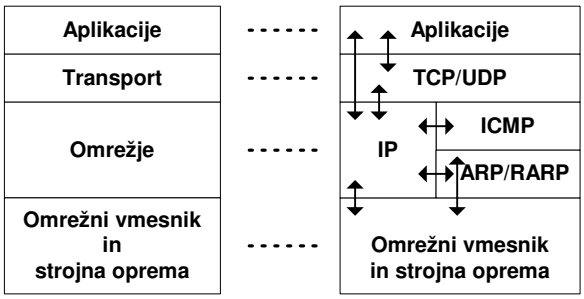
Za identifikacijo gostitelja na spletu vsakemu gostitelju prirejamo njegov naslov IP. Če je gostitelj opremljen z več *omrežnimi vmesniki* (angl. adapter), ima vsak vmesnik unikatni naslov IP. Naslov IP sestoji iz dveh delov:

IP naslov = <številka omrežja><številka gostitelja>.

Številko omrežja določamo centralno in je unikatna v spletnem omrežju. Za unikatnost naslovov IP skrbijo posebne avtorizirane organizacije (v Sloveniji Arnes).

1.1.1 *Protokolni sklad TCP/IP*

Protokolni sklad TCP/IP se je razvijal 30 let in je kot večina omrežne programske opreme po strukturi plastovit. Internetni protokol je sestavljen iz štirih plasti (slika 2).



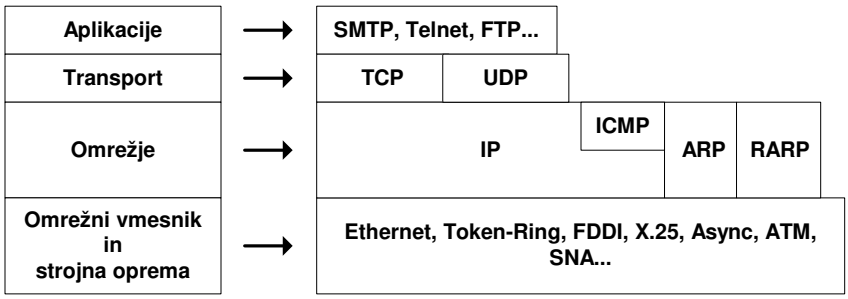
Slika 2: Protokolni sklad TCP/IP

Funkcionalne plasti so naslednje (Vidmar, 2002):

- *Aplikacijska plast*: določena s programom in za komunikacijo uporablja protokol TCP/IP. Aplikacija je uporabniški proces, ki komunicira z drugim procesom na lokalnem ali oddaljenem gostitelju (primer: Telnet, FTP, SMTP). Vmesnik med aplikacijo in transportno plastjo definiramo s številko vrat (angl. port) in vtičnice (angl. socket).
- *Transportna plast*: določa prenos podatkov med dvema aplikacijama. Istočasno lahko skrbi tudi za več aplikacij hkrati. Transportna plast zagotavlja zanesljivo izmenjavo informacij. Glavni transportni protokol je TCP. Poleg tega uporabljamo tudi protokol UDP. Ta omogoča nepovezane storitve v nasprotju s protokolom TCP, ki je povezavno orientiran. Pogosto se protokol UDP uporablja pri aplikacijah, ki potrebujejo hitri transportni mehanizem (aplikacije v realnem času).
- *Omrežna (Internetna) plast*: najpomembnejši protokol na tej plasti je IP, ki je nepovezan protokol in od spodnjih plasti ne zahteva zanesljivosti ali popravljanja napak. Te funkcije morajo biti realizirane na višjih plasteh. Najpomembnejši del te plasti predstavlja funkcija usmerjanja, ki zagotavlja, da sporočila pridejo do pravega naslovnika. Ostali omrežni protokoli, ki lahko nastopajo v omrežni plasti, so še: IP, ICMP, IGMP, ARP in RARP.
- *Plast računalniškega omrežja* (angl. link layer ali data-link layer): je vmesnik med omrežno plastjo in omrežno strojno opremo. Vmesnik lahko določa zanesljivo ali nezanesljivo povezavo in je lahko paketno ali znakovno (angl. stream) orientiran. V principu TCP/IP na tej plasti ne določa posebnega protokola,

dopušča pa uporabo katerega koli omrežnega vmesnika (primer: IEEE 802.2, X.25, ATM, FDDI in celo SNA).

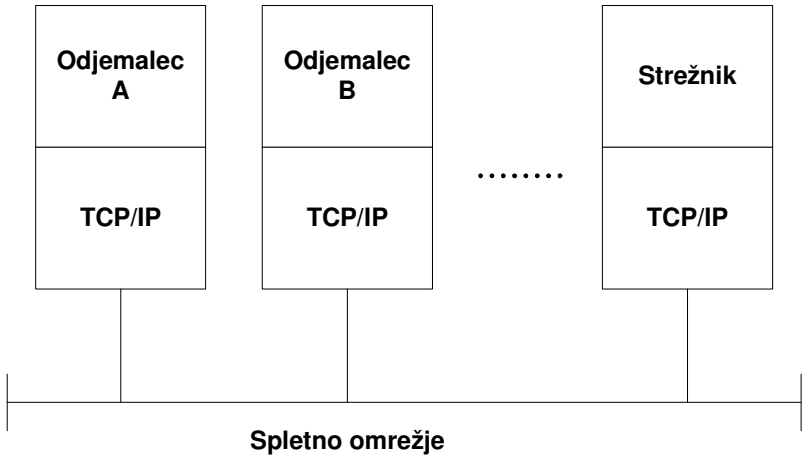
Povezave med plastmi in protokoli, ki delujejo v njih, prikazuje slika 3.



Slika 3: Podrobnejši arhitekturni model TCP/IP

1.1.2 *Model odjemalec/strežnik*

TCP je entitetno povezavno orientiran protokol, ki ne omogoča relacij strežnik/odjemalec (angl. master/slave). Strežnik je aplikacija, ki uporabnikom ponuja spletne storitve. Odjemalec storitev zahteva. Aplikacija sestoji iz obeh, tako strežniškega kot odjemalnega dela, ki lahko tečeta na istem ali različnih sistemih. Uporabnik običajno kliče odjemalni del aplikacije, ki zgradi zahtevo za določeno storitev in to pošlje strežniški aplikaciji. Protokol TCP/IP tukaj uporabljamo kot transportno sredstvo med odjemalcem in strežnikom. Strežnik zahtevo sprejme, izvrši zahtevano storitev in pošlje rezultate odjemalcu. Strežnik običajno lahko zadovoljuje več zahtev hkrati (slika 4).



Slika 4: Model odjemalec/strežnik

Nekateri strežniki čakajo zahteve na določenih vratih tako, da njihovi odjemalci vedo, preko katerega IP vtiča usmeriti svojo zahtevo. Odjemalci, ki želijo komunicirati s

strežnikom prek neznanih vrat, morajo uporabiti drugačen mehanizem, da bi se naučili, kam nasloviti zahtevo. Primer takega mehanizma je npr. proces *portmap*.

1.1.3 *Mostovi, usmerjevalniki in prehodi*

Oblikovanje spletnega omrežja s povezavo večjega števila podomrežij je mogoče s pomočjo usmerjevalnikov. Pri tem povezovanju je pomembno razlikovati med pojmi *most* (angl. bridge), *usmerjevalnik* (angl. router) in *prehod* (angl. gateway). Te pojme pojasnjujemo v nadaljevanju.

- *Most*: povezuje segmente LAN na plasti računalniškega omrežja in usmerja podatkovne *okvire* med njimi. Med omrežji prenaša naslove MAC (angl. Media Access Control) in je neodvisen od kakršnega koli protokola višje plasti. Most je transparenten protokolu IP, t.j. če gostitelj IP pošlje *datagram* IP (enota prenosa v protokolu IP) drugemu gostitelju na omrežje povezano z mostom, ga pošlje direktno gostitelju in datagram prečka most ne da bi se tega pošiljatelj zavedal.
- *Usmerjevalnik*: povezuje omrežja na omrežni plasti in usmerja *pakete* med njimi. Usmerjevalnik mora poznati strukturo naslavljanja in odloča o tem, kdaj in kako usmeriti pakete. Osnovna usmerjevalna funkcija je implementirana v plasti IP. Usmerjevalnik je dosegljiv prek številke IP. Če gostitelj pošlje IP datagram drugemu gostitelju na mreži, povezani z usmerjevalnikom, ga pošlje vmesniku in ne neposredno gostitelju.
- *Prehod*: povezuje omrežja na višjih plasteh kot most ali usmerjevalnik. Običajno podpira preslikavo naslovov iz enega omrežja v drugega in omogoča pošiljanje podatkov med okolji, ki podpirajo povezavo od konca do konca (angl. end-to-end). Prehod je nepropusten za protokol IP, t.j. gostitelj preko njega ne more poslati datagrama IP.

1.2 Internetni protokol

Internetni protokol (angl. Internet Protocol, krajše IP) oblikuje navidezni pogled na omrežje in z njim zakrije pod njim ležeče plasti. Je nezanesljiv in nepovezan paketno orientiran protokol. Paketi poslani prek protokola IP se lahko izgubijo ali celo pomnožijo. Takih situacij pa IP ne rešuje sam, ampak to nalogo prepusti protokolom višjih plasti. Eden izmed razlogov za uporabo nepovezanega omrežnega protokola je bilo zmanjšanje odvisnosti omrežja od računalniških centrov, ki so značilni za povezano orientirana omrežja.

1.2.1 *Naslavljanje IP*

IP naslov je predstavljen kot 32-bitna vrednost brez predznaka, ki jo običajno izražamo v decimalni obliki, kjer štiri bajte med seboj ločimo s piko. Vrednost posameznega bajta ne more presegati števila $2^8 - 1$ ali 255. Primer: 192.168.120.5. Preslikavo med naslovom IP

in simboličnim imenom, ki si ga lažje zapomnimo, opravlja strežnik DNS (angl. Domain Name System).

1.2.2 *Naslov IP*

Za identifikacijo gostitelja (angl. host) na spletu uporabljamo naslov IP oz. Internetni naslov. Če je gostitelj priključen na več kot eno omrežje, ima en naslov IP za vsak omrežni vmesnik. IP naslov sestoji iz naslednjega para števil:

IP naslov = <številka omrežja><številka gostitelja>

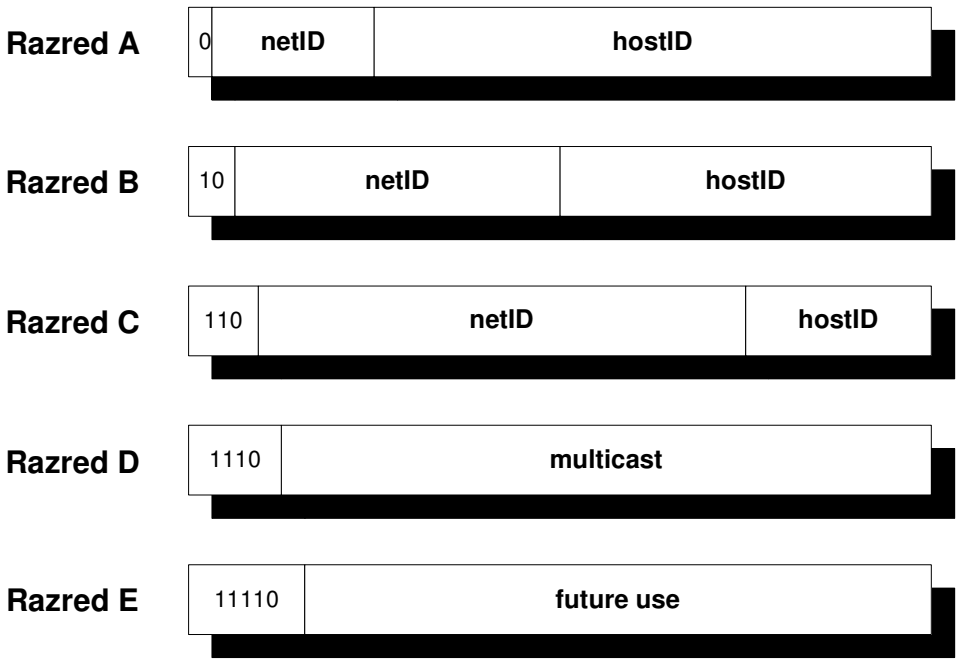
Številke omrežij centralno upravljajo Internetni omrežni informacijski centri (angl. Internet Network Information Center, krajše InterNIC) in so unikatne za celoten splet. Naslov IP je 32-bitno število, ki ga prikazujemo kot decimalne vrednosti štirih bajtov med seboj ločenih s piko.

Protokol IP uporablja naslove IP za unikatno določitev gostitelja na spletu. Po fizičnem omrežju gostitelja se prenašajo datagrami IP. Vsak datagram IP vsebuje izvorni in ponorni naslov IP. Ponorni naslov IP moramo pri pošiljanju preslikati v fizični naslov gostitelja, t.j. naslov MAC. Ta preslikava povzroči iskanje ponornega fizičnega naslova v omrežju. V ta namen uporabljamo protokol ARP (angl. Address Resolution Protokol).

Prvi biti naslova IP določajo, kako se preostanek bitov deli na omrežni in gostiteljski del. Omrežni naslov označujemo tudi kot *netID*, gostiteljski pa kot *hostID* (slika 5). Obstaja pet razredov naslovov IP:

- *razred A*: 7 bitov se uporablja za omrežni, preostalih 24 za gostiteljski del naslova IP, kar omogoča $2^7 - 2$ (ali 126) omrežij z $2^{24} - 2$ (ali 16.777.214) gostitelji na omrežje,
- *razred B*: 14 bitov za omrežni in 16 bitov za gostiteljski del naslova IP, kar omogoča $2^{14} - 2$ (ali 16.382) omrežij z $2^{16} - 2$ (ali 65534) gostitelji na omrežje,
- *razred C*: uporablja 21 bitov za omrežni in 8 bitov za gostiteljski del naslova IP, kar omogoča $2^{21} - 2$ (ali 2.097.150) omrežij z $2^8 - 2$ (ali 254) gostitelji na omrežje,
- *razred D*: naslovi so rezervirani za multicasting (vrsti *broadcastinga*, vendar v omejenem področju samo do gostiteljev, ki uporabljajo isti razred naslovov D),
- *razred E*: rezerviran za prihodnost.

Razred A uporabljamo za omrežja z velikim številom gostiteljev, razred C je primeren za omrežja z majhnim številom gostiteljev. Za srednje velika omrežja lahko uporabljamo naslove razreda B. Število manjših in srednjih omrežij se je v zadnjih letih zelo hitro povečalo in obstajala je bojazen, da bi naslovov razreda B sredi 90-ih zmanjkalo. Ta problem je znan pod imenom *IP Address Exhaustion*. Eden izmed načinov, kako rešiti ta problem, je t.i. *omrežno prevajanje naslovov NAT* (angl. Network Address Translation), ki ga bomo spoznali v nadaljevanju.



Slika 5: Razredi naslovov IP

Razlog razdelitve naslovov IP v dva dela je ta, da se s tem hkrati razdeli odgovornost določanja naslovov IP - številko omrežja določa mednarodna organizacija InterNIC, za številke gostiteljev v lokalnem omrežju pa je zadolžen omrežni skrbnik.

1.2.3 *Posebni naslovi IP*

Naslovi IP, z vsemi biti postavljenimi na 0 ali 1, imajo poseben pomen:

- Vsi biti postavljeni na 0: pomenijo, *ta gostitelj* (IP naslov <naslov gostitelja>=0) ali *to omrežje* (IP naslov <naslov omrežja>=0). Če želi gostitelj komunicirati prek omrežja in ne pozna svojega IP naslova, lahko pošlje paket z <naslov omrežja>=0. Ostali gostitelji na mreži bodo interpretirali ta naslov kot *to omrežje*.
- Vsi biti postavljeni na 1: pomenijo *vsa omrežja* in *vse gostitelje*. Primer: vse gostitelje na mreži 128.2 (razred B) zapišemo kot 128.2.255.255. Ta oznako imenujemo tudi neposredni naslov *broadcast*, saj vsebuje naslov določenega omrežja in naslov *broadcast* gostitelja.
- Povratna povezava (angl. loopback): razred A v omrežju 127.0.0.0 definiramo kot omrežje *povratnih povezav*. Naslove tega omrežja prirejamo vmesnikom, ki obdelujejo podatke na lokalnem sistemu in nikoli ne dostopajo do fizičnega omrežja.

1.2.4 *Podomrežja IP*

Zaradi hitre rasti spleta postane način prirejanja naslovov IP preveč nefleksibilen, da bi dopuščal enostavne spremembe v konfiguracijah lokalnih omrežij. Te spremembe lahko nastanejo, ko:

- na lokaciji namestimo novi tip fizičnega omrežja,
- rast števila gostiteljev zahteva delitev lokalnega omrežja v dve ali več podomrežij,
- povečevanje razdalj linij zahteva delitev omrežja v manjša podomrežja.

Da bi se izognili zahtevi po dodatnih omrežnih naslovih IP, se je pojavil koncept *podomrežij* (angl. subnetting). Gostiteljski del naslova IP ponovno razdelimo na številko omrežja in na številko gostitelja. Drugo omrežje imenujemo podomrežje (angl. subnet). Naslov IP je sedaj sestavljen iz treh delov:

naslov IP = <številka omrežja><številka podomrežja><številka gostitelja>

Kombinacijo številke podomrežja in gostitelja pogosto imenujemo *lokalni naslov* ali lokalni del naslova IP. Podomrežje je transparentno glede na zunanje omrežje, t.j. gostitelj znotraj omrežja, ki podpira podmreže, se podmreže zaveda, gostitelj z drugega omrežja pa lokalni del naslova IP obravnava enako kot številko gostitelja.

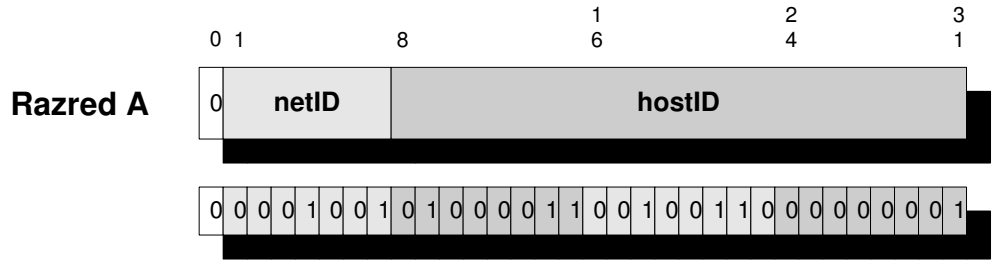
Delitev lokalnega dela naslova IP v številko podomrežja in številko gostitelja je narejena s pomočjo maske podomrežja. Maska podomrežja je 32-bitno število, kjer biti 0 označujejo pozicije bitov v naslovu IP, ki jih prištevamo k številki gostitelja, biti 1 pa pozicije bitov istega naslova, ki jih prištevamo k številki podomrežja. Bitne pozicije v maski podomrežja, ki pripadajo številki omrežja, postavimo na 1, vendar jih ne uporabljamo.

Primer: omrežje z naslovi iz razreda B, ki ima 16-bitni lokalni del, lahko uporablja naslednji shemi naslavljanja:

- Prvi bajt je številka podomrežja, drugi bajt pa številka gostitelja. To nam daje $2^8 - 2$ (ali 254) možnih podomrežij vsako z $2^8 - 2$ (ali 254) gostitelji. Maska podomrežja je 255.255.255.0.
- Prvih 12 bitov uporabljamo za številko podomrežja in zadnje štiri za številko gostitelja. To nam daje $2^{12} - 2$ (ali 4094) možnih podomrežij s samo $2^4 - 2$ (ali 14) gostitelji na mrežo. Maska podomrežja v tem primeru je 255.255.255.240.

Čeprav skrbnik omrežja lahko povsem poljubno določi del podomrežja lokalnega naslova, pa velja dogovor, da za številko podomrežja v maski uporabljamo zaporedne bite. Zato je zaželeno, da za masko podomrežja uporabljamo zaporedje enic na začetku lokalnega dela IP naslova. S tem postanejo naslovi IP bolj čitljivi (kanonična oblika zapisa IP naslova: 192.168.120.5/24).

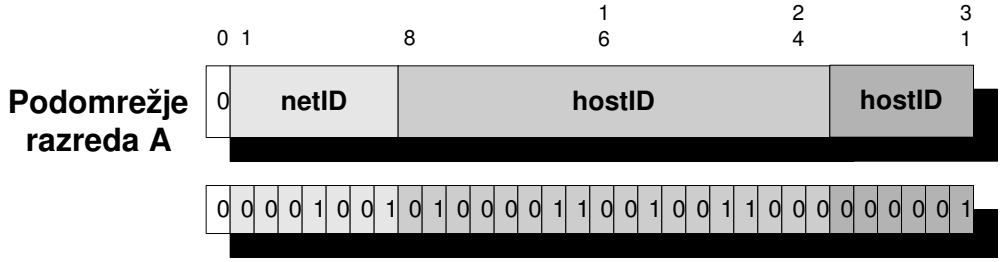
Primer: vzemimo omrežje z naslovi iz razreda A (slika 6).



Slika 6: Omrežje z naslovi iz razreda A brez podomrežij

```
00001001 01000011 00100110 00000001    32-bitni naslov
    9       67        38        1       decimalni zapis
             9.67.38.1                   naslov IP razreda A
             9                            <številka omrežja>
               67.38.1                   <številka gostitelj.>
```

Podomrežja so razširitev omenjene sheme, ki interpretira del naslova gostitelja kot naslov podomrežja. Podomrežno naslavljanje dobimo, če npr. biti 8 do 25 IP naslova iz razreda A označujejo naslov podomrežja in biti 26-31 dejanski naslov gostitelja (slika 7).



Slika 7: Naslov razreda A z masko in naslovom podomrežja

Za identifikacijo bitov v originalnem naslova IP gostitelja, ki označujejo številko podomrežja, uporabljamo masko podomrežja. V našem primeru uporabljamo masko 255.255.255.192 v decimalnem zapisu, ki razdeli lokalni naslov na $2^{18}-2$ (ali 262.143) podomrežij z maksimalno $2^{6}-2$ (ali 62) gostiteljev na podomrežje. Tehnika izračuna osnovnega IP naslova v omrežju s podomrežjem je naslednja:

```
00001001 01000011 00100110 00000001 = 9.67.38.1 (32-bitni naslov)
11111111 11111111 11111111 11------    255.255.255.192 (maska podomr.)
================================    logični_IN
00001001 01000011 00100110 00------ = 9.67.38 (osnovni naslov podomr.)
```

Naslov gostitelja izračunamo z negacijo maske podomrežja in operacijo logični_IN med masko in naslovom podomrežja, ali drugače

```
00001001 01000011 00100110 00000001 = 9.67.38.1 (32-bitni naslov)
--------- --------- --------- --111111 =  0.0.0.63 (maska gostitelja)
==============================    logični_IN
--------- --------- --------- --000001 = 1 (naslov gostitelja)
```

Številko podomrežja izračunamo tako, da v maski bite, ki označujejo pozicije omrežja gostitelja, zanikamo in s tako dobljeno masko na originalnem naslovu IP izvedemo operacijo logični_IN, ali drugače

```
00001001 01000011 00100110 00000001 = 9.67.38.1 (32-bitni naslov)
-------- 11111111 11111111 11------ = 0.255.255.192 (maska podomr.)
==============================    logični_IN
-------- 01000011 00100110 00------ = 68.760 (številka podomrežja)
```

1.2.5 *Usmerjanje IP*

Pomembna funkcija plasti IP je usmerjanje, ki določa osnovni mehanizem povezovanja različnih fizičnih omrežij z usmerjevalniki. Vsak gostitelj IP lahko poleg svojega normalnega dela opravlja istočasno tudi funkcijo usmerjanja (angl. IP forwarding). Obstajata dve vrsti usmerjanja IP: neposredno in posredno.

Neposredno usmerjanje

Če sta tako ponorni kot izvorni gostitelj priključena na isto fizično omrežje, lahko pošljemo datagram IP neposredno naslovniku. Ta način neposredne komunikacije imenujemo tudi neposredno usmerjanje.

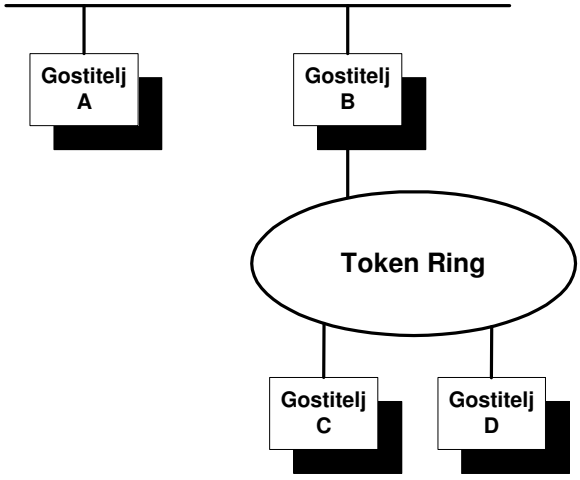
Posredno usmerjanje

Posredno usmerjanje nastopa, ko ponorni in izvorni gostitelj nista neposredno povezana na isto omrežje. Edina pot, kako doseči naslovnika na drugem omrežju, je prek enega ali več usmerjevalnikov. Naslov prvega izmed usmerjevalnikov (prvi skok, angl. hop) s stališča algoritma za usmerjanje IP imenujemo posredni prehod. Ta naslov je tudi edina informacija, ki jo potrebuje izvorni gostitelj. Tudi v primeru, da je na istem omrežju definiranih več podomrežij, potrebujemo za paket, ki potuje skozi različna podomrežja, posredno usmerjanje (slika 8).

1.2.6 *Usmerjevalna tabela IP*

Ob inicializaciji protokol IP avtomatično tvori seznam lokalnih vmesnikov, s pomočjo katerega določa neposredne smeri do gostiteljev, ki so na voljo. Istočasno se z uporabo usmerjanja IP ta seznam dopolnjuje z naslovi drugih omrežij in pripadajočih usmerjevalnikov. Vsak gostitelj vzdržuje množico preslikav med ponornimi omrežnimi

naslovi IP in potmi do bližnjih usmerjevalnikov. Te preslikave gostitelji hranijo v t.i. usmerjevalni tabeli IP.

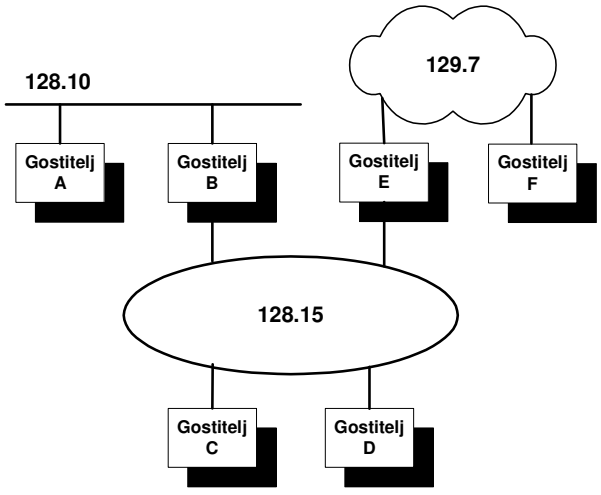


Slika 8: Neposredne in posredne poti

Vrste preslikav v teh tabelah so naslednje:

- neposredne poti za lokalno priključena omrežja,
- posredne poti za omrežja dosegljiva preko enega ali več usmerjevalnikov,
- privzeta pot, ki določa prehod v primeru, da ponornega omrežja IP ni mogoče najti s pomočjo točk 1 in 2.

Primer: izgradnja usmerjevalne tabele IP (slika 9).



IP usmerjevalna tabela gostitelja D

cilj	usmerjevalnik	vmesnik
129.7.0.0	E	lan0
128.15.0.0	D	lan0
128.10.0.0	B	lan0
default	B	lan0
127.0.0.1	loopback	lo

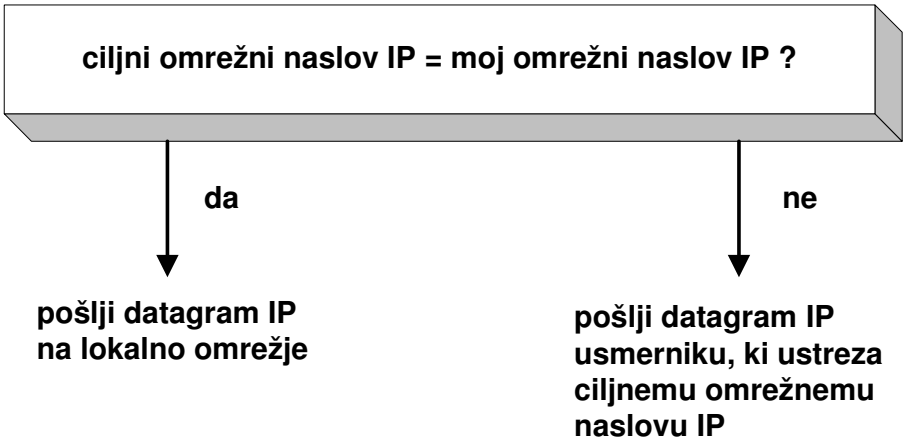
IP usmerjevalna tabela gostitelja F

cilj	usmerjevalnik	vmesnik
129.7.0.0	F	wan0
default	E	wan0
127.0.0.1	loopback	lo

Slika 9: Usmerjevalne tabele IP - scenarij

1.2.7 *Usmerjevalni algoritem IP*

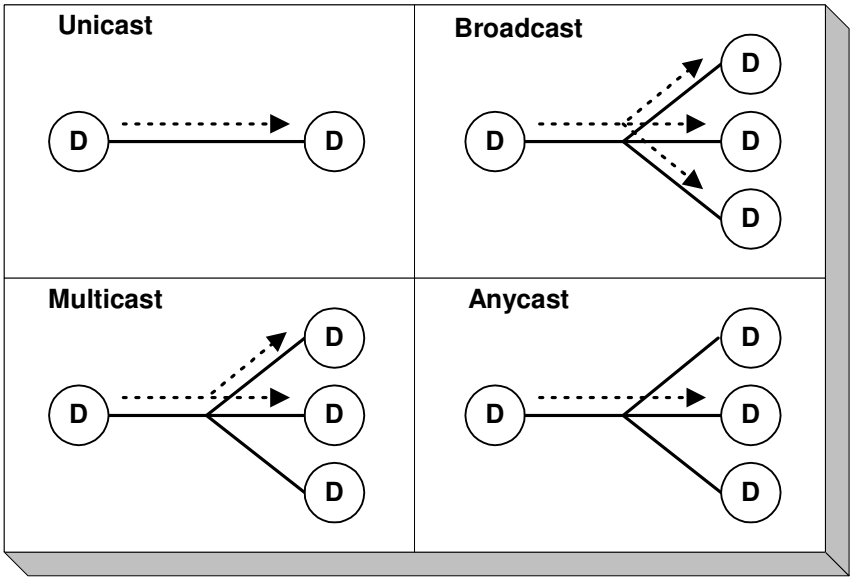
Protokol IP za usmerjanje datagrama IP uporablja usmerjevalni algoritem, katerega diagram poteka v primeru pošiljanja datagrama IP prikazuje slika 10.



Slika 10: Usmerjanje v omrežju IP brez podomrežij

1.2.8 *Metode dostave – Unicast, Broadcast, Multicast in Anycast*

Večina naslovov IP se sklicuje na enega prejemnika, t.j. njegov naslov *unicast*. Poleg tega obstajajo še tri vrste naslovov IP, ki jih uporabljamo za naslavljanje več prejemnikov hkrati: *broadcast*, *multicast* in *anycast* (slika 11). Iz slike 11 vidimo, da naslov *broadcast* naslavlja vse gostitelje v omrežju, *multicast* samo nekatere, *anycast* pa samo določenega specifičnega gostitelja.



Slika 11: Načini dostave paketov

Vsak nepovezan protokol lahko pošlje sporočila na naslove *broadcast*, *multicast*, *anycast* ali *unicast*. Povezavno orientirani protokoli lahko uporabljajo samo naslove *unicast*, ker povezava med določenim parom gostiteljev že obstaja.

Broadcast

Vrste naslovov *broadcast*:

- Omejeni naslovi *broadcast*: naslov 255.255.255.255 uporabljamo v omrežjih, ki podpirajo t.i. *broadcasting* (npr. LAN) in se nanaša na vse gostitelje v podomrežju. Vsi gostitelji priključeni na lokano omrežje ta naslov razpoznajo, vendar ga usmerjevalniki ne usmerjajo naprej.
- Omrežno usmerjeni naslovi *broadcast*: če biti na pozicijah, ki v naslovu IP predstavljajo številko omrežja in so vsi biti na pozicijah, ki označujejo številko gostitelja, postavljeni na 1, se naslov *broadcast* nanaša na vse gostitelje v določeni mreži (npr. protokol ARP).
- Podomrežno usmerjeni naslovi *broadcast*: če biti na pozicijah, ki v naslovu IP predstavljajo številki omrežja in podomrežja, označujejo veljavno omrežje in podomrežje in so vsi biti na pozicijah, ki označujejo številko gostitelja postavljeni na 1, se naslov *broadcast* nanaša na vse gostitelje določenega podomrežja.
- vsi podomrežno usmerjeni naslovi *broadcast*: če biti na pozicijah, ki v naslovu IP predstavljajo številko omrežja, označujejo veljavno omrežje in so vsi biti na pozicijah, ki označujejo lokalni del naslova, postavljeni na 1, se ta naslov *broadcast* nanaša na vsa podomrežja določenega omrežja.

Multicast

Velika pomanjkljivost pošiljanja naslovov *broadcast* je pomanjkanje selektivnosti. Če namreč podomrežno usmerjeni naslov *broadcast* pošljemo v določeno podomrežje, ga bo sprejel vsak na omrežje priključeni gostitelj, ga obdelal in preveril, ali je protokol, po katerem sprašujemo, aktiven. V primeru, da je ta protokol aktiven, gostitelj odgovori pritrdilno, v nasprotnem primeru pa poslano sporočilo uniči. Pošiljanje pritrdilnih odgovorov lahko predstavlja veliko obremenitev omrežja. *Multicast* se z uporabo skupin naslovov IP tej obremenitvi omrežja izogiba. Vsaka skupina naslovov *multicast* je sestavljena z 28-bitnim številom, ki je vključena v naslovni razred D. Skupina naslovov *multicast* je v območju 224.0.0.0 do 239.255.255.255. 28-bitna števila so standardizirana. Primer: naslov 224.0.0.1 naslavlja vse usmerjevalnike.

Anycasting

Včasih isto storitev IP ponuja več različnih strežnikov. Na primer, da želi uporabnik s protokolom FTP prenesti datoteko, ki je na voljo na dveh različnih FTP strežnikih, vendar ne ve, katera izmed povezav s strežnikom je hitrejša. Odjemalec lahko v takem primeru pošlje v omrežje naslov *anycast*. Strežnika, ki ponujata zahtevano storitev, odjemalcu na

ta naslov odgovorita. Odjemalec se poveže s strežnikom, ki prvi odgovori na njegovo povpraševanje. *Anycast* je del Ipv6.

1.2.9 *Intranet*

Zaradi velikega porasta števila omrežij na spletu se je pojavil problem zasičenja naslovnega prostora IP. Eden izmed načinov, kako zmanjšati ta naslovni prostor, je naslavljanja zasebnih omrežij. Organizacije, ki ne potrebujejo povezave IP s spletom, lahko uporabljajo standardizirana območja IP naslovov, ki jih je mednarodna organizacija IANA rezervirala v ta namen. Ta območja naslovov so prikazana v tabeli 1.

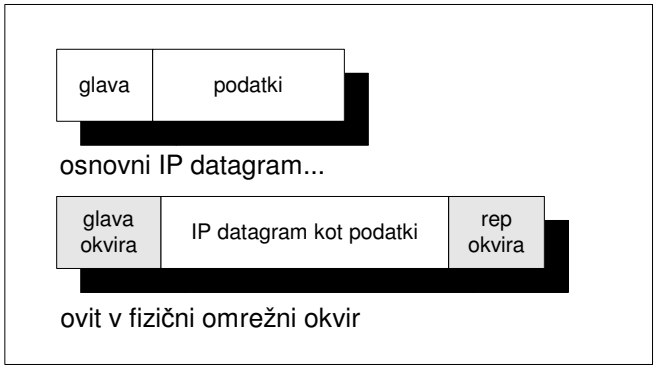
Tabela 1: Zasebni IP naslovi

Razred	Naslovno območje	Število omrežij
A	10	1
B	172.16-172.31	16
C	192.168.0-192.168.255	256

Vsaka organizacija, ki nima neposredne povezave z drugo, lahko uporablja katerekoli naslove v tem območju. Ker globalno ti naslovi niso unikatni, se ne morejo nanašati na gostitelje z istimi naslovi v drugi organizaciji. Po drugi strani pa usmerjevalniki v lokalnih omrežjih teh zasebnih naslovov ne smejo usmerjati v spletno omrežje. Gostitelji, ki imajo zasebne naslove IP, torej ne smejo imeti neposredne povezave na splet, t.j. od zunaj niso vidni. Vse povezave na splet morajo omogočati višje, t.j. aplikacijske plasti (npr. aplikacijski prehod, angl. proxy). Organizacija je od zunaj vidna s podmnožico zunanjih IP naslovov, za katero se lahko skriva celotno zasebno omrežje.

1.2.10 *Datagram IP*

Enoto prenosa podatkovnih paketov v protokolu TCP/IP imenujemo *datagram* IP. Ta je sestavljen iz glave, ki vsebuje informacije za omrežno plast, in podatkov, ki so pomembni za protokole višjih plasti (slika 12).



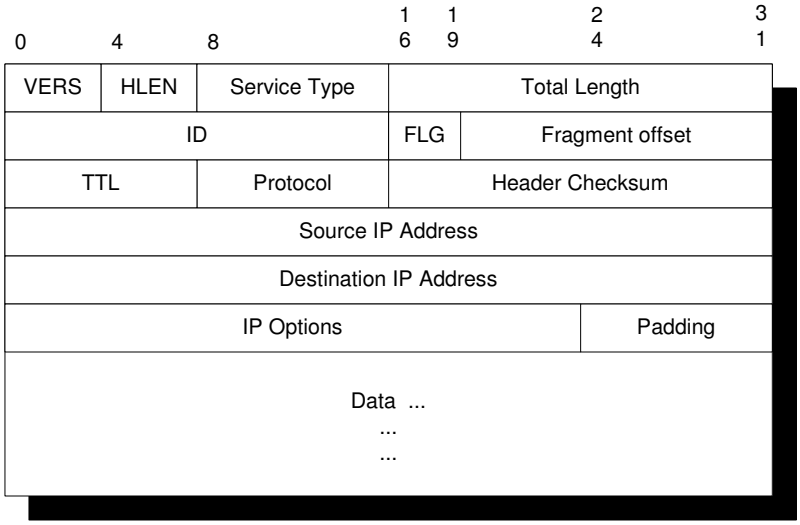
Slika 12: Format osnovnega datagrama IP

Datagram IP lahko protokol IP na izvoru fragmentira v zaporedje okvirov in na cilju zopet sestavi v originalen datagram. Maksimalna dolžina datagrama je 65.535 bajtov (ali

oktetov). Datagrame IP z velikostjo do največ 576 bajtov gostitelji TCP/IP obdelujejo brez fragmentiranja. Vsi fragmentirani datagrami imajo glavo, ki je kopija glave originalnega datagrama in podatke, ki glavi sledijo. Fragmenti se obnašajo kot celota, saj v primeru, da se izgubi samo eden fragment v verigi, izgubimo celoten datagram.

Glava datagrama IP

Glava datagrama IP je dolga najmanj 20 bajtov (slika 13).



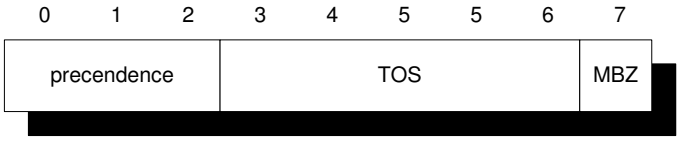
Slika 13: Format glave IP datagrama

Pomen posameznih polj s slike 13 je naslednji:

VERS
Verzija protokola IP (4 je IPv4, 5 je poskusna, 6 je IPv6).

HLEN
Dolžina glave IP zaokrožene na 8 bajtov.

Service Type
Kakovost storitve, kot jo zahteva ta datagram IP (slika 14).



Slika 14: IP tip storitve

Polja na sliki 14 pomenijo:

Precendence
Predstavlja merilo, s katerim označujemo naravo in prioriteto tega datagrama (rutina, prioriteta, neodložljiv, kritičen,…).

TOS (Type of Service)

Določa tip storitve (minimalna zakasnitev, maksimalna hitrost, maksimalna zanesljivost, minimalna cena, normalna storitev).

MBZ (must be zero)

Rezerviran za bodočo uporabo.

Total Length

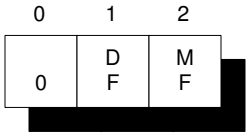
Celotna dolžina datagrama, t.j. glava in podatki, merjeno v bajtih.

Identification

Unikatna številka, ki jo prireja pošiljatelj in pomaga pri ponovnem sestavljanju fragmentiranih datagramov. Fragmenti datagrama imajo isto identifikacijsko številko.

Flags

Različne nadzorne zastavice (slika 15).



Slika 15: IP zastavice

Na sliki 15 pomenijo:

0 Rezervirano, mora biti 0.

DF (Don't Fragment) 0 pomeni dopuščeno fragmentiranje, 1 pomeni fragmentiranje ni dovoljeno.

MF (More Fragments) 0 pomeni, da je to zadnji fragment v tem datagramu, 1 pomeni, da to ni zadnji fragment v datagramu.

Fragment Offset

Uporabljen pri fragmentiranih datagramih in pomaga pri zbiranju fragmentov v celoten datagram.

Time to Live (TTL)

Določa čas (v sekundah), ki ga datagram lahko uporabi za potovanje po omrežju do naslovnika. Vsak usmerjevalnik, skozi katerega ta datagram gre, mora odšteti od tega polja čas, ki ga je porabil za njegovo procesiranje. Čeprav ga usmerjevalnik obdela v manj kot eno sekundi, pa kljub vsemu odšteje od polja TTL celo sekundo in s tem polje namesto časovne metrike postane števec korakov (hop-count metric). Ko vrednost polja doseže vrednost nič, datagram uničimo.

Protokol Number

Označuje višje-nivojski protokol, kateremu naj bi IP dostavil podatke v tem datagramu (na primer 1 = ICMP, 2 = IGMP, 3 = GGP, 4 = IP, 6 = TCP, 8 = EGP, 17 = UDP, 89 = OSPF…).

Header Checksum

Kontrolna vsota glave datagrama IP. Kontrolno vsoto izračunamo kot 16-bitni 1'komplement 1'komplementa vsot vseh 16-bitnih besed v glavi datagrama. Če se kontrolna vsota glave ne ujema s kontrolno vsoto vsebine, datagram uničimo, saj je vsaj eden izmed bitov glave pokvarjen.

Source IP Address
32-bitni naslov IP gostitelja, ki je poslal ta datagram.

Destination IP Address
32-bitni naslov IP gostitelja, kateremu je ta datagram namenjen.

Options
Polje variabilne dolžine, namenjeno nadzoru, odpravljanju napak in obračunavanju.

Padding
Če uporabljamo to možnost, datagram zaokrožimo do naslednje 32-bitne besede in polje inicializiramo z dvojiškimi 0.

Podatki
Vsebovani v datagramu se posredujejo višje-nivojskemu protokolu.

Fragmentiranje

Ko datagram IP potuje od enega gostitelja do drugega, lahko križa različna fizična omrežja. Fizično omrežje ima določeno maksimalno velikost okvira MTU (angl. Maximum Transmission Unit), ki omejuje dolžino datagrama kopiranega v fizični okvir. Dolgi datagrami IP se na izvornem gostitelju fragmentirajo v manjše okvire in ponovno zberejo na ponornem gostitelju. Običajna vrednost MTU za omrežja Ethernet je 1500 bajtov. Postopek fragmentiranja je naslednji:

- Protokol IP preveri, če je v glavi IP datagrama postavljena zastavica DF, ki onemogoča fragmentiranje. Če je temu tako, datagram uniči, izvornemu gostitelju pa preko paketa ICMP javi napako.
- Glede na vrednost MTU razdeli podatke na dva ali več fragmentov. Dolžine podatkov zaokroži na dolžino, ki je večkratnik števila 8 in napolni z binarno 0 (angl. padding).
- Vse podatke kopira v fragmentirane datagrame. Glave teh kopira iz originalnega datagrama in jih pri tem nekoliko modificira, t.j. prilagodi dolžino glave in skupno dolžino fragmenta ter ponovno preračuna kontrolno vsoto glave fragmentiranega datagrama.
- Vsak izmed teh fragmentiranih datagramov pošlje naprej kot normalni datagram IP. Plast IP obravnava vsak fragment neodvisno, t.j. fragmenti lahko do zahtevanega ponora potujejo po različnih poteh in so lahko predmet nadaljnjega fragmentiranja, če potujejo preko omrežij, ki imajo manjše MTU-je.

V ponornem gostitelju se fragmentirani podatki zberejo v en datagram IP. V ta namen gostitelj na sprejemni strani dodeli interni pomnilnik (angl. buffer) in čaka, dokler ne pride prvi fragment. V tem trenutku starta časovno rutino, t.j. timer, z nastavljenim števcem TTL. Če pade vrednost števca TTL na nič, in vsi fragmenti datagrama še niso bili sprejeti, se datagram uniči. Vrednost števca TTL je odvisna od implementacije.

Primer: na operacijskem sistemu IBM AIX 4.3 je števec TTL postavljen na 60 sekund.

Če zaporedni fragmenti datagrama prispejo do ponornega gostitelja pred časovno omejitvijo, plast IP podatke enostavno kopira v interni pomnilnik na pozicijo določeno s poljem Fragment Offset.

1.3 ICMP

Ko mora usmerjevalnik oz. ponorni gostitelj sporočiti izvornemu gostitelju, da je pri procesiranju datagrama prišlo do napake, uporabi protokol ICMP (angl. Internet Control Message Protocol). Uporabljamo ga v primerih, ko gostitelj:

- sporoča omrežne napake,
- sporoča omrežne preobremenitve,
- pomaga pri odpravi napak (angl. troubleshoting),
- obvešča o zakasnitvah.

ICMP ima naslednje lastnosti:

- Uporablja protokol IP tako kot vsi protokoli višjih plasti, t.j. sporočila ICMP so ovita (angl. encapsulate) v datagrame IP. ICMP je integralni del protokola IP in mora biti implementiran v vsakem modulu IP.
- Sporoča nekatere napake, protokola IP pa ne dela zanesljivejšega. Datagram se še vedno lahko izgubi, ne da bi se pri tem njegova izguba zabeležila. Zanesljivost morajo zagotavljati protokoli višjih plasti.
- Sporoča napake o vsakem datagramu IP z izjemo sporočil ICMP, saj se s tem izognemo rekurziji.
- Za fragmentirane datagrame IP se v primeru napake ICMP pošlje sporočilo za fragment 0.
- Sporočila ICMP se nikoli ne pošiljajo kot odgovor na naslove *broadcast* ali *multicast*.

1.3.1 *Aplikacije ICMP*

Poznamo dve enostavni in zelo razširjeni na protokolu ICMP temelječi aplikaciji: *ping* in *traceroute*. Za ugotavljanje dosegljivosti gostitelja uporablja ping sporočili ICMP *Echo* in *Echo Replay*. Traceroute pošilja datagrame IP, da bi določil pot od izvornega po ponornega gostitelja.

Ping

Ping je najenostavnejša aplikacija TCP/IP, ki pošlje enega ali več datagramov IP določenemu ponornemu gostitelju, od njega zahteva odgovor in med čakanjem na odgovor meri odzivni čas. Običajno velja, če naredimo *ping* do gostitelja, bi do njega morale priti tudi zahteve do ostalih aplikacij kot npr. Telnet ali FTP. Z razvojem varnostnih meril na spletu, posebno požarnih zidov (angl. Firewall, krajše FW), ki nadzoruje dostop do omrežja prek aplikacijskega protokola in številke vrat, gornja trditev

ne drži več, t.j. FW lahko prepreči dostop do aplikacije *ping*. Sintaksa aplikacije je odvisna od operacijskega sistema, na katerem teče.

Primer: primer izvajanja aplikacije *ping* na sistemu Windows/XP je prikazan na sliki 16.

```
C:\>ping 192.168.225.101

Pinging 192.168.225.101 with 32 bytes of data:

Reply from 192.168.225.101: bytes=32 time=1ms TTL=128
Reply from 192.168.225.101: bytes=32 time<1ms TTL=128
Reply from 192.168.225.101: bytes=32 time<1ms TTL=128
Reply from 192.168.225.101: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.225.101:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

Slika 16: Izvajanje aplikacije *ping* na Windows/XP

Traceroute

Program *traceroute* je lahko zelo uporaben pri odpravljanju napak in omogoča določiti pot, po kateri potuje datagram IP od izvornega do ponornega gostitelja. Implementiran je v večini produktov TCP/IP in sloni na protokolih ICMP in UDP.

Primer: primer izvajanja aplikacije *traceroute* na sistemu Windows/XP je prikazan na sliki 17. Opazimo lahko, da je klicanje aplikacije na sistemu Windows/XP nekoliko skrajšano, t.j. *tracert*.

```
C:\ >tracert www.google.com

Tracing route to www.l.google.com [64.233.183.99]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  192.168.225.1
  2    30 ms    30 ms    30 ms  192.168.10.5
  3    30 ms    30 ms    30 ms  193.189.186.102
  4    30 ms    30 ms    30 ms  193.77.12.97
  5    31 ms    31 ms    31 ms  213.250.18.89
……..
 18    77 ms    77 ms    77 ms  64.233.175.244
 19   168 ms    79 ms   159 ms  216.239.49.9
 20    83 ms    84 ms    82 ms  216.239.43.34
 21    93 ms    83 ms    83 ms  64.233.183.99

Trace complete.
```

Slika 17: Izvajanje aplikacije *traceroute* na Windows/XP

1.4 IGMP

Podobno kot ICMP je tudi IGMP (angl. Internet Group Management Protocol) integralni del plasti IP. Namen tega protokola je, da dopušča ali prepoveduje gostiteljem na omrežju udeležbo pri naslavljanju *multicast*. Poleg tega usmerjevalnikom omogoča preverjanje ali na določenem omrežju obstaja kakšen gostitelj, ki odgovarja na naslov *multicast*.

1.5 ARP

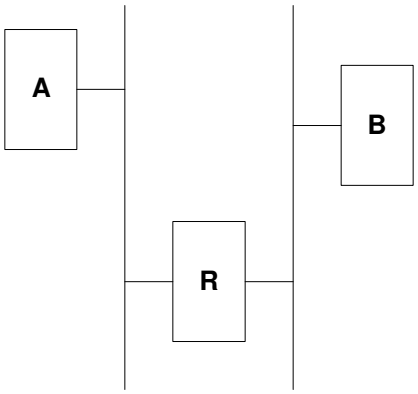
Protokol ARP (angl. Address Resolution Protocol) je omrežno specifičen standardni protokol, ki je zadolžen za konverzijo višje-nivojskih naslovov IP v fizične omrežne naslove MAC. Posamezni gostitelj se na enovitem fizičnem omrežju predstavlja s svojim naslovom MAC, ki ga višje-nivojski protokol naslavlja simbolično z naslovom IP (simbolični naslov). Ko plast IP želi poslati datagram na ponorni naslov IP w.x.y.z, fizičnega naslova pripadajočega gostitelja ne pozna, zato za odgovor prosi modul ARP. Ta modul omogoča prevajanje naslova IP w.x.y.z ponornega gostitelja v naslov MAC. Pri tem ARP uporablja posebno translacijsko tabelo. Če naslova ne najde v tabeli, pošlje na mrežo naslov *broadcast*.

1.5.1 *Generiranje naslovov ARP*

Ko aplikacija želi poslati podatke na določeni naslov IP, usmerjevalni mehanizem IP najprej določi naslova IP in MAC naslednjega prehoda (koraka) na poti do naslovnika. Prehod je lahko sam ponorni gostitelj ali usmerjevalnik. Modul ARP poskuša najti naslov MAC v svoji translacijski tabeli ARP. Če ga najde, 48-biten fizični MAC naslov vrne naslovniku, če pa ARP ustreznega para <IP-naslov, MAC-naslov> ne najde, paket uniči in v omrežje na naslov *broadcast* pošlje zahtevo ARP v obliki peterčka <*protokol*, *izvorni*-IP, *izvorni*-MAC, *ponorni*-IP, *ponorni*-MAC >. Ko gostitelj iz omrežja sprejme paket, ponorni gostitelj pošlje paket modulu ARP, ki pregleda, ali govori isti protokolni jezik z izvornim gostiteljem, in ali par <*izvorni*-IP, *izvorni*-MAC> že obstaja v tabeli ARP. Če zapis že obstaja, ga popravi, v obratnem primeru doda. Na koncu ARP v peterčku zamenja izvorne in ponorne naslove, doda svoj naslov MAC in pošlje odgovor izvornemu gostitelju. Izvorni gostitelj sledi istemu scenariju, po katerem ARP doda želenega gostitelja v svojo translacijsko tabelo. Naslednjič, ko bo nek višje-nivojski protokol zahteval isti naslov, bo ta že v ustrezni tabeli ARP.

1.5.2 *Koncept proxy-ARP*

Predpostavimo, da je omrežje IP razdeljeno na dve podomrežji povezani z usmerjevalnikom (slika 18).



Slika 18: Gostitelja ARP povezana z usmerjevalnikom

Ko gostitelj A želi poslati datagram IP gostitelju B, z uporabo protokola ARP najprej določi fizični naslov omrežja gostitelja B. Ker gostitelj A ne razlikuje fizičnih omrežij, njegov usmerjevalni algoritem predpostavlja, da je gostitelj B na istem lokalnem omrežju in pošlje zahtevo ARP na naslov *broadcast*. Gostitelj B seveda te zahteve ne bo sprejel, toda sprejel jo bo usmerjevalnik R. Usmerjevalnik R, ki loči podomrežja, ve, da je ponorni gostitelj B na drugem fizičnem omrežju. Na zahtevo ARP odgovori tako, kot če bi bil sam gostitelj B, t.j. naslov MAC gostitelja B v datagramu IP je isti kot naslov MAC usmerjevalnika R. Gostitelj A, ki sprejema ta odgovor ARP, bo naslednji paket IP za gostitelja B naslovil na usmerjevalnik R, ta pa ga bo pravilno posredoval v drugo podomrežje. Usmerjevalnik R ima vlogo strežnika proxy-ARP. Rezultat transparentnega omrežja:

- navadni gostitelji pojma podomrežja ne poznajo in lahko uporabljajo standardni usmerjevalni algoritem IP.
- usmerjevalniki med podomrežji morajo:
  - uporabljati podomrežni usmerjevalni algoritem IP,
  - uporabljati modificiran modul ARP, ki lahko odgovarja na zahtevo drugih gostiteljev.

1.6 RARP

Nekateri omrežni gostitelji, npr. delovne postaje brez diskov, omrežni računalniki (angl. Network Computer, krajše NC), v procesu nalaganja operacijskega sistema še ne poznajo svojega naslova IP. Za določitev njihovega naslova IP uporabljajo mehanizem podoben protokolu ARP, vendar je v tem primeru znan naslov gostitelja MAC in zahtevani parameter naslov IP. Od protokola ARP se RARP razlikuje tudi po tem, da mora v omrežju obstajati strežnik RARP s konfigurirano podatkovno bazo o preslikavah naslovov MAC v naslove IP. Koncept RARP se izvaja na podoben način kot ARP.

1.7 Vrata in vtičnice

Koncept vrat (angl. port) in vtičnic (angl. socket) je potreben, da lahko natančno določimo, kateri proces na lokalnem gostitelju komunicira z določenim procesom na oddaljenem gostitelju in po katerem protokolu poteka komunikacija med njima. Vzroki, da v ta namen ne moremo uporabiti identifikacijske številke procesa (angl. Process Identifier, krajše PID) na določenem operacijskem sistemu, so naslednji:

- PID se spremeni vsakokrat, ko proces na novo zaženemo.
- PID se med operacijskimi sistemi razlikuje, t.j. ni enolično določen.
- strežniški proces lahko ima več povezav z več odjemalci hkrati, t.j. tudi identificiranje povezave ni unikatno.

Koncept vrat in vtičnic omogoča, da enolično identificiramo povezave, aplikacije in gostitelje neodvisno od PID.

1.7.1 *Vrata*

Vsak proces, ki nudi storitev svojim odjemalcem, v protokolu TPC/IP identificiramo z vrati. Vrata so 16-bitno število, ki ga uporablja transportni protokol za določanje, kateremu višje-nivojskemu protokolu oz. aplikaciji mora dostaviti vhodno sporočilo. Obstajata dva tipa vrat:

- Splošno znana: pripadajo standardnim strežnikom. Strežnik Telnet, na primer, uporablja vrata 23. Splošno znana vrata uporabljajo števila v območju od 1 do 1023. Splošno znane številke vrat so običajno liha števila, saj so starejši sistemi uporabljali za dvosmerne operacije liho/sodi par vrat. Večina strežnikov zahteva danes samo eno številko vrat. Izjema sta strežnika BOOTP, ki uporablja vrata 67 in 68 ter FTP z vrati 20 in 21. Splošno znana vrata nadzoruje in prireja organizacija IANA in za delovanje zahtevajo določene sistemske programe ali procese. Vzrok za nastanek splošno znanih vrat je bil omogočiti odjemalcem, da v omrežju najdejo strežnike brez dodatnega konfiguriranja.
- Občasna: odjemalci za komunikacijo s strežniki ne potrebujejo splošno znanih številk vrat, saj lahko to dobijo ob inicializaciji komunikacije v ustreznem UDP datagramu od ustreznega strežnika. Ta vsakemu odjemalcu za čas trajanja komunikacije dodeli določeno številko vrat. Občasna vrata uporabljajo števila višja od 1023, t.j. običajno v območju 1024 do 65.535. Odjemalec lahko uporablja poljubno številko vrat, vendar mora biti kombinacija <*transportni protokol*, IP-*naslov*, *številka vrat*> unikatna. Organizacija IANA številil občasnih vrat ne nadzoruje in jih na večini sistemov uporabljajo navadni uporabniški programi.

1.7.2 *Vtičnice*

Vtičnica je eden izmed več aplikacijskih programskih vmesnikov (angl. Application Programming Interface, krajše API), ki jih sveženj TCP/IP ponuja za dostop do komunikacijskih protokolov. Prvič se je aplikacijski vmesnik pojavil na operacijskem sistemu BSD 4.2 UNIX. Čeprav še vedno ni standardiziran, je postal standard *de facto*. Pri definiciji vtičnic uporabljamo naslednje pojme:

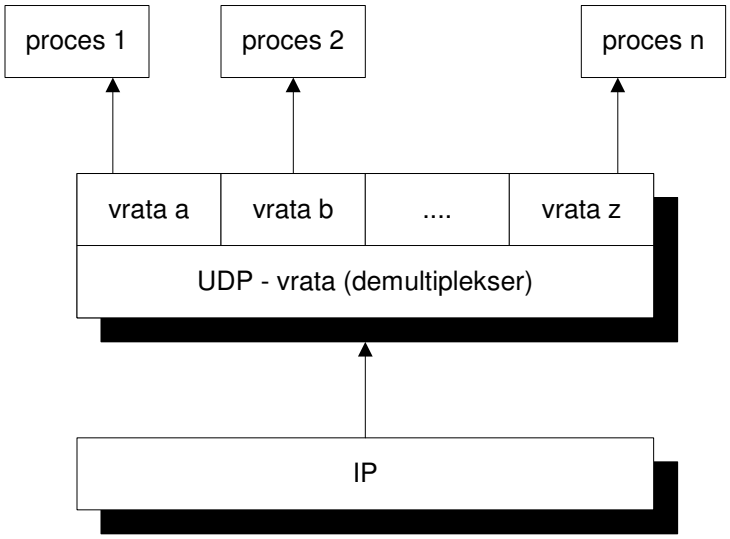
- *Vtičnica*: je posebna vrsta datoteke, ki jo proces uporablja pri zahtevanju omrežnih storitev od operacijskega sistema.
- *Naslov vtičnice*: je trojica {protokol, lokalni naslov, lokalni proces}. V primeru, da gre za protokol TCP/IP, velja {tcp, 193.189.186.65, 12345}.
- *Pogovor* (angl. conversation): je komunikacijska povezava med dvema procesoma.
- *Zveza*: je peterica {protokol, lokalni naslov, lokalni proces, oddaljeni naslov, oddaljeni proces}, ki v celoti določa dva procesa vključena v pogovor.
- *Pol-zveza*: je lahko {protokol, lokalni naslov, lokalni proces} ali {protokol, oddaljeni naslov, oddaljeni proces}.

Pol-zvezo imenujemo tudi *vtičnica* ali *transportni naslov*. Vtičnica je torej končna točka (angl. endpoint) komunikacije, ki jo lahko naslavljamo v omrežju. Dva procesa komunicirata prek vtičnic TCP. Model vtičnic omogoča procesu na lokalnem računalniku dvosmerno znakovno (angl. stream) povezavo do procesa, ki teče na oddaljenem računalniku. Aplikaciji se tako ni potrebno ukvarjati z upravljanjem povezave, saj to zanjo počne protokol TCP.

1.8 UDP

UDP je v osnovi aplikacijski vmesnik do protokola IP, ki pa mu UDP ne dodaja dodatne zanesljivosti, nadzora pretoka ali odpravljanja napak. UDP nastopa kot multipleksor/demultipleksor za pošiljanje in sprejemanje datagramov, ki usmerja datagrama preko določenih vrat (slika 19).

UDP je mehanizem, ki omogoča lokalni aplikaciji pošiljanje datagramov oddaljeni aplikaciji. Plast protokola UDP je izredno tanka, zato zelo hitra, vendar od aplikacije zahteva odgovornost glede zanesljivosti, pretoka podatkov in odpravljanja napak. Aplikacije, ki pošiljajo datagram gostitelju, za identifikacijo ponora potrebujejo več kot samo naslov IP, saj so datagrami običajno usmerjeni določenemu procesu in ne sistemu kot celoti. UDP izbira določen proces z uporabo vrat.

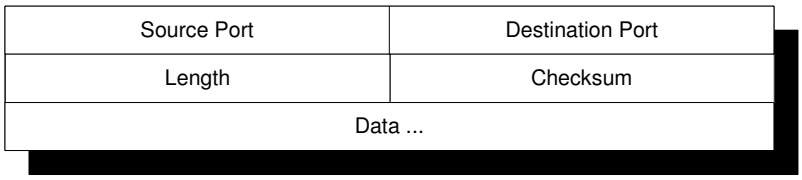


Slika 19: Demultipleksiranje UDP glede na vrata

1.8.1 *UDP format*

Vsak datagram UDP pošiljamo znotraj enega datagrama IP. Čeprav datagram IP med prenosom fragmentiramo, ga ponorna plast IP pred posredovanjem plasti UDP sestavi. Vse implementacije IP datagramov IP manjših od 576 bajtov ne fragmentirajo. Če predpostavimo, da je maksimalna velikost glave IP 60 bajtov, lahko zaključimo, da je dopuščena velikost ne-fragmentiranega datagrama UDP 516 bajtov.

Datagram UDP ima 16-bajtno glavo (slika 20).



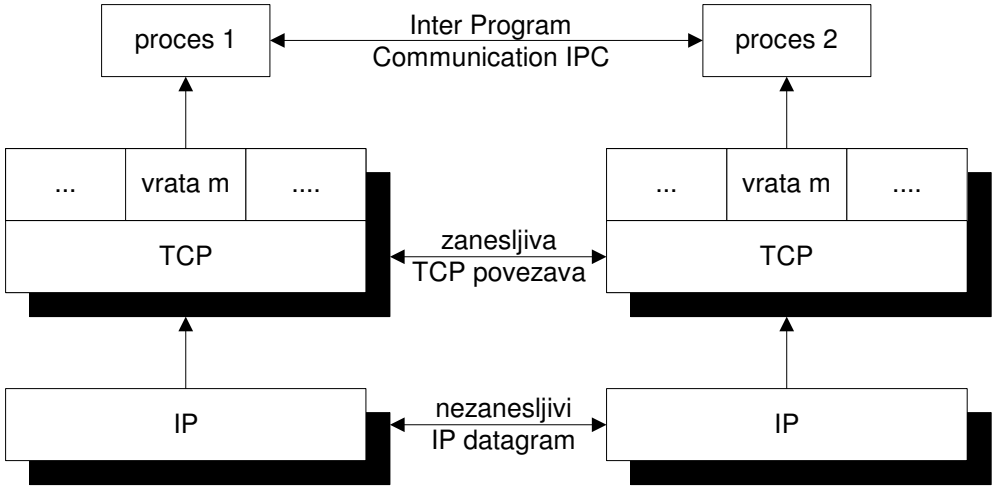
Slika 20: Datagram UDP

1.8.2 *UDP API*

Protokola UDP in IP ne zagotavljata dostave paketov, nadzora pretoka (angl. flow control) in odpravljanja napak (angl. error recovery). Vse te varnostne mehanizme mora omogočati aplikacija. Standardne aplikacije, ki uporabljajo UDP, so: TFTP, DNS, RPC (uporablja NFS), SNMP, LDAP.

1.9 TCP

Protokol TCP omogoča aplikacijam veliko več možnosti od protokola UDP, saj zagotavlja dodatno zanesljivost, odpravljanje napak in nadzor pretoka. TCP je povezavno orientiran protokol, kar pomeni, da mora biti v času trajanja komunikacije med odjemalcem in strežnikom vzpostavljena povezava (slika 21). Večina uporabniških aplikacijskih protokolov, kot npr. Telnet in FTP, uporablja protokol TCP.



Slika 21: Povezava TCP med procesoma

1.9.1 *Koncept TCP*

Osnovni namen protokola TCP je omogočiti zanesljiv logični tokokrog oz. povezano storitev med pari procesov. Protokol TCP ne pričakuje zanesljivosti od nižje-nivojskih protokolov (kot npr. IP), zato mora zanesljivost zagotavljati sam. Lastnosti TCP so naslednje:

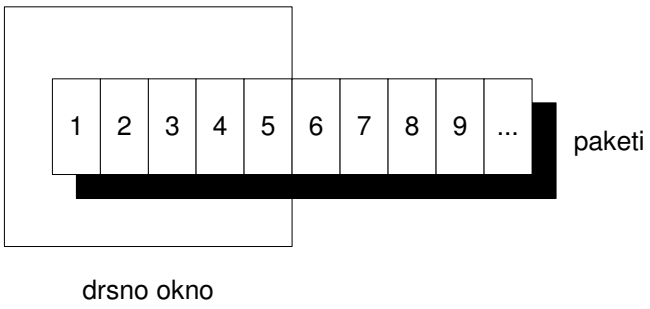
- *Znakovni prenos podatkov*: s stališča aplikacije protokol TCP prenaša zvezen tok (angl. stream) bajtov skozi omrežje. Aplikaciji ni potrebno skrbeti za sekanje podatkov v osnovne bloke ali datagrame. Za vse to poskrbi TCP. Včasih aplikacija želi vedeti ali so bili protokolu TCP posredovani podatki tudi dejansko dostavljeni. V ta namen služi definirana funkcija *potisni* (angl. push), ki pošlje vse segmente TCP iz pomnilnika ponornemu gostitelju.
- *Zanesljivost*: TCP vsakemu prenesenemu bajtu priredi zaporedno številko in od ponornega protokola TCP pričakuje potrditev sprejema (angl. acknowledgement, ACK). Če ACK izvorni TCP ne sprejme v določenem časovnem intervalu, podatke ponovno pošlje. Ker podatke pošiljamo v blokih, pošljemo samo zaporedno številko prvega podatkovnega bajta v segmentu. Ponorni TCP uporabi zaporedno številko za odstranitev dvojnih segmentov in njihovo preureditev, če prispejo neurejeno.

- *Nadzor pretoka*: ponorni TCP pri pošiljanju ACK izvornemu gostitelju označi število bajtov, ki jih lahko še sprejme, preden pride do prekoračitve velikosti njegovega internega pomnilnika. To sporoči v obliki najvišje zaporedne številke segmenta, ki ga lahko še sprejme. Ta mehanizem označujemo tudi kot mehanizem okna (angl. window).
- *Multipleksiranje*: izvršeno z uporabo vrat podobno kot pri protokolu UDP.
- *Logične povezave*: zanesljivost in nadzor pretoka zahtevata, da TCP za vsak podatkovni tok inicializira in vzdržuje določene statusne informacije. Kombinacijo tega statusa, ki vključuje vtičnico, zaporedno številko in velikost oken, imenujemo logična povezava. Vsaka povezava je enolično določena s paroma vtičnic uporabljenih v procesih pošiljanja in sprejemanja.
- *Dvosmerne povezave*: TCP za hkratne podatkovne tokove omogoča prenos v obe smeri.

1.9.2 *Princip drsnega okna*

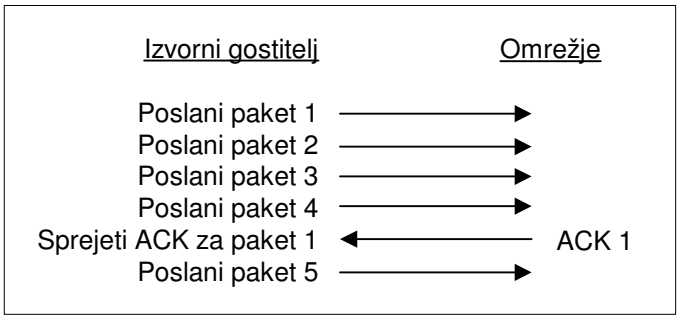
Enostavni transportni protokol uporablja naslednji princip: pošlje paket in preden pošlje naslednji paket od strežnika pričakuje potrditev ACK. Če ACK v določenem času ne sprejme, paket pošlje ponovno. Čeprav ta mehanizem zagotavlja zanesljivost, izkoristi samo del omrežnega prenosnega pasu. Zmogljivost tega lahko izboljšamo, če predpostavimo, da pošiljatelj uvrsti svoje pakete za prenos v skupino, t.j. drsno okno (angl. slide window) in uporabi sledeča pravila:

- izvor lahko pošlje vse pakete v drsnem oknu, ne da bi sprejel ACK, vendar mora zagnati timeout timer za vsakega posebej.
- ponor mora potrditi vsak prejeti paket in ga označiti z zaporedno številko zadnjega dobro sprejetega paketa.
- izvor pomakne drsno okno ob vsaki prejeti potrditvi ACK (slika 22).



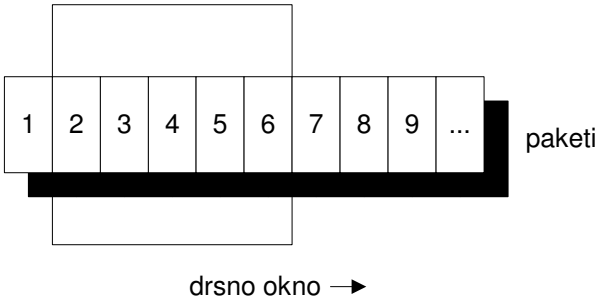
Slika 22: TCP sporočilni paketi

Primer: izvor pošlje pakete od 1 do 5 ne da bi počakal na potrditev (slika 23).



Slika 23: Potrditev sporočila TCP

V tem trenutku sprejme ACK 1 in lahko pomakne drsno okno za en paket desno (slika 24).



Slika 24: Pomik drsnega okna

V tem trenutku lahko izvor pošlje paket 8. Nastopita lahko dve različni situaciji:

- Paket 2 izgubimo: izvor ne dobi ACK 2 in drsno okno ostane na poziciji 1. Ker ponorni gostitelj ni prejel paketa 2, potrdi pakete 3, 4, in 5 z ACK 1, saj je bil paket 1 sprejet zadnji v zaporedju. Na izvorni strani paket 2 povzroči časovni iztek (angl. timeout), zato ga ta pošlje znova. Sprejem tega paketa ponorni gostitelj potrdi z ACK 5, saj je uspešno prejel vse pakete od 1 do 5, izvor pa je drsno okno premaknil za 5 pozicij desno.
- Paket 2 prispe, vendar izgubimo potrditev: izvor ni sprejel ACK 2, vendar je sprejel ACK 3, kar je potrdil, saj je sprejel vse tri pakete, zato izvor lahko pomakne drsno okno do paketa 4.

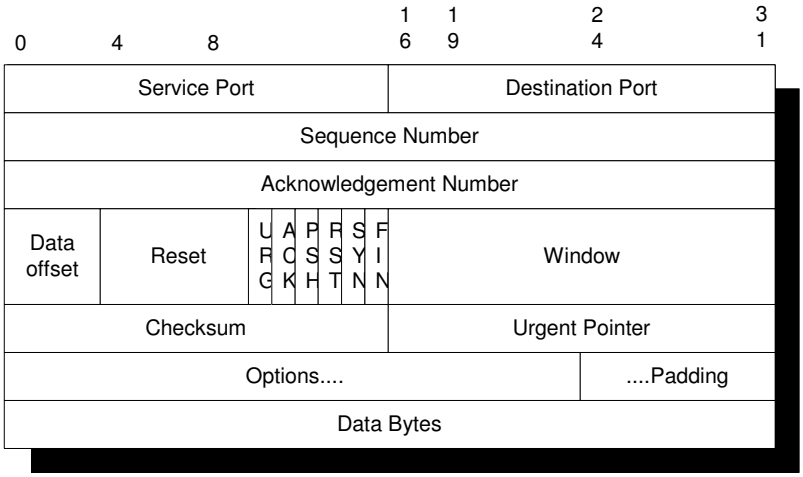
Drsno okno zagotavlja:

- zanesljiv prenos,
- boljši izkoristek prenosnega pasu omrežja,
- nadzor pretoka, ker lahko ponorni gostitelj počaka s potrditvijo glede na velikost prostega internega pomnilnika in velikost drsnega okna.

V protokolu TCP je drsno okno realizirano na bajtni, in ne na paketni, osnovi.

1.9.3 *Format TCP segmenta*

Format TCP segmenta je prikazan na sliki 25.



Slika 25: Format TCP segmenta

Na sliki 25 pomenijo:

Source Port
16-bitna številka izvornih vrat, ki jih uporablja ponorni gostitelj za pošiljanje odgovorov.

Destination Port
16-bitna številka ponornih vrat.

Zaporedna številka
Zaporedna številka prvega podatkovnega bajta v tem segmentu. Če je nadzorni bit SYN postavljen in ima zaporedna številka vrednost *n*, je prvi podatkovni bajt *n*+1.

Acknowledgement number
Če je nadzorni bit ACK postavljen, to polje označuje vrednost naslednje zaporedne številke, ki jo ponorni gostitelj pričakuje.

Data offset
Število 32-bitnih besed v TCP glavi, ki označujejo začetek podatkov.

Reserved
6-bitov rezerviranih za prihodnjo uporabo.

URG
Označuje, da je polje *urgent pointer* za ta segment pomembno.

ACK
Označuje, da je polje *acknowledgement number* za ta segment pomembno.

PSH
Funkcija potisni.

RST
Resetira povezavo.

SYN

Sinhronizira zaporedna števila.

FIN

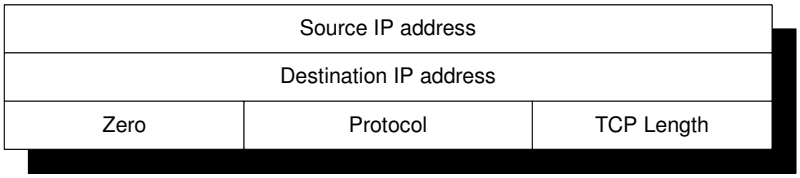
Konec podatkov.

Window

Uporabljamo ga v segmentih s postavljenim bitom ACK, kjer določa število podatkovnih bajtov, ki se začenjajo z vrednostjo polja *acknowledgement number* in katere lahko ponorni gostitelj še sprejme.

Checksum

16-bitni 1' komplement 1' komplementov vsote vseh 16-bitnih besed v glavi psevdo-IP. Glava psevdo-IP, ki se uporablja samo za izračun kontrolne vsote, je prikazana na sliki 26.



Slika 26: Glava prevdo-IP

Urgent Pointer

Kaže na prvi bajt, ki sledi podatkom *urgent* in ima pomen samo, če je nadzorni bit URG postavljen.

Options

Podobno kot v primeru datagrama IP so opcije lahko:

i. bajti, ki označujejo številko opcije,
ii. opcije s spremenljivo dolžino v formatu <opcija, dolžina, opcijski podatki>.

Padding

Vsi bajti z vrednostjo 0, ki se uporabljajo zato, da se dolžina glave TCP zaokroži na večkratnik števila 32 in dopolni z 0.

1.9.4 *Nadzor nasičenja*

Ena izmed velikih razlik med protokoloma TCP in UDP je nadzor nasičenja, ki izvornemu gostitelju preprečuje, da bi zasedel vse kapacitete omrežja. TCP lahko hitrost pošiljanja prilagodi zmogljivosti omrežja in se s tem poskuša izogniti nasičenju. Sčasoma se je razvilo več izboljšav nadzora nasičenja, vendar večina sodobnih implementacij uporablja štiri med seboj prepletene algoritme, ki predstavljajo osnovo protokola TCP:

- počasni zagon (angl. slow start),
- izogibanje nasičenju (angl. congestion avoidance),
- hitro ponovitev (angl. fast retransmit),
- hitra obnovitev (angl. fast recovery).

Počasni zagon

Algoritem deluje na dejstvu, da je število segmentov, ki jih v določenem časovnem obdobju izvorni gostitelj lahko odda v omrežje, enako številu prejetih potrditev , ki jih ta v istem časovnem obdobju prejme od ponornega gostitelja. Počasni zagon izvornemu protokolu TCP doda dodatno okno, t.j. okno nasičenja. Ko se med izvornim in ponornim gostiteljem vzpostavi nova povezava, TCP postavi okno nasičenja na vrednost 1. Vsakokrat, ko TCP sprejme potrditev ACK, se okno nasičenja poveča za en segment. Izvorni gostitelj lahko v omrežje pošlje število segmentov, ki ga določa nižja izmed velikosti okna nasičenja in drsnega okna. Velikost okna nasičenja nadzira izvorni, velikost drsnega okna pa ponorni gostitelj. Velikost okna nasičenja je torej odvisna od prenosne kapacitete omrežja, velikost drsnega okna pa od velikosti internega pomnilnika ponornega gostitelja.

Izogibanje nasičenju

Algoritem predpostavlja, da segmentov ne izgubljamo zaradi poškodb (manj kot 1%), ampak zaradi nasičenja, kar pomeni, če segment ni prišel do cilja, je nekje na poti prišlo do nasičenja. Izguba segmentov se kaže kot

- časovni iztek (angl. timeout) ali
- podvojena potrditev.

Čeprav sta algoritma izogibanje nasičenju in počasni zagon neodvisna, sta medsebojno povezana. Ko namreč nastopi nasičenje, mora TCP upočasniti prenos paketov v omrežju in potem za normalno delovanje uporabiti počasni zagon. V praksi sta oba algoritma običajno implementirana skupaj. Algoritma izogibanje nasičenju in počasni zagon za vsako povezavo zahtevata dve spremenljivki:

- okno nasičenja in
- prag nasičenja.

TCP, ki uporablja oba algoritma skupaj, deluje po naslednji logiki:

- algoritem za vsako povezavo inicializira spremenljivko okno nasičenja na 1 (segment) in prag nasičenja na 65535 (bajtov).
- TCP nikoli ne pošlje več segmentov kot je nižja vrednost velikosti okna nasičenja in velikosti drsnega okna.
- ko nastopi nasičenje (časovni iztek ali podvojena potrditev ACK), se vrednost praga nasičenja zmanjša za polovico. Če gre za časovni iztek, algoritem postavi vrednost okna nasičenja na 1.
- ko ponorni gostitelj potrdi nove podatke, algoritem poveča vrednost okna nasičenja. Način povečanja je seveda odvisen od tega ali TCP izvaja počasni zagon ali izogibanje nasičenju. Če je okno nasičenja manjše ali enako pragu nasičenja, je TCP v počasnem zagonu, v nasprotnem primeru pa se izogiba nasičenju.

Hitra ponovitev

Pospeši delovanje TCP ob izgubah segmentov. Ob napačnem vrstnem redu segmentov TCP pošilja potrditev s številko tistega, ki ga je pričakoval (pride do podvojitve potrditev).

2 USMERJEVALNI PROTOKOLI

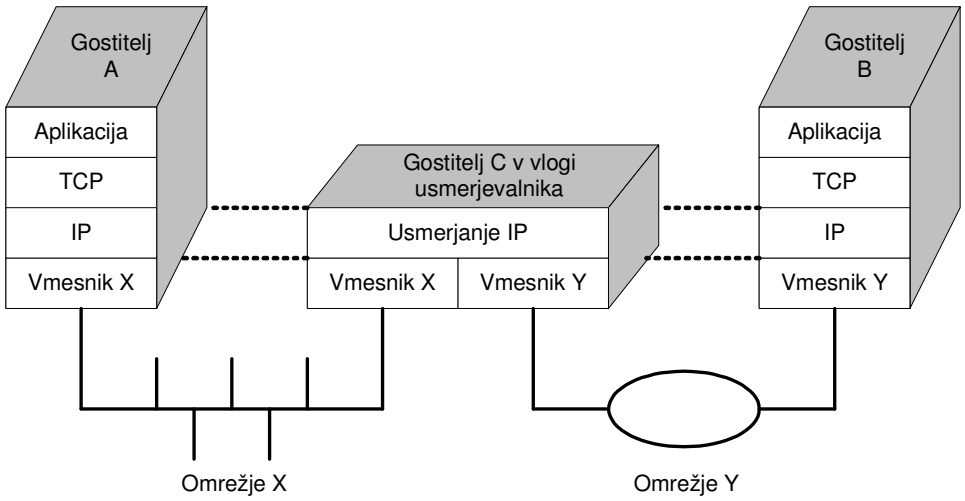
Ena izmed osnovnih funkcij protokola IP je povezovanje različnih fizičnih omrežij. To povezovanje je s protokolom IP zelo enostavno, saj je ta zelo fleksibilen, podpira večino fizičnih omrežij in uporablja usmerjevalni algoritem IP.

2.1 Osnovno usmerjanje IP

Osnovno funkcijo usmerjanja, ki je prisotna v vseh IP implementacijah, lahko definiramo kot:

> *Vhodni datagram IP, ki zahteva drugačen ponorni naslov IP od lokalnega naslova gostitelja, obravnava usmerjevalnik kot izhodni datagram IP.*

Ta izhodni datagram je predmet usmerjevalnega algoritma lokalnega gostitelja (vmesni gostitelj), ki izbere naslednji skok (angl. hop) za ta datagram. Ponorni gostitelj se lahko nahaja v omrežju, na katerega je priključen vmesni gostitelj. Če je fizično omrežje ponornega gostitelja drugačno od omrežja, na katerega je priključen izvorni gostitelj, je naloga vmesnega gostitelja, da posreduje (angl. forward) datagram IP iz enega fizičnega omrežja v drugega (slika 26).



Slika 26: Usmerjanje IP

Običajna usmerjevalna tabela IP vsebuje informacije o lokalno priključenih gostiteljih in naslovih IP ostalih usmerjevalnikov v tem omrežju. Razširimo jo lahko z informacijami o oddaljenih omrežjih IP, vendar pa še vedno ostajajo usmerjevalne tabele z omejenimi informacijami, saj prikazujejo samo del omrežja IP. Tako vrsto usmerjevalnikov, ki se nahajajo na običajnih delovnih postajah in strežnikih, imenujemo tudi usmerjevalniki z omejenimi informacijami. Zanje velja:

- poznajo samo del omrežja IP,
- lokalnim gostiteljem omogočajo avtonomijo pri vzpostavljanju in spreminjanju prehodov,
- napaka v zapisu prehoda enega izmed usmerjevalnikov lahko povzroči težavo, zaradi katere določeni del omrežja lahko postane nedosegljiv.

Naprednejše usmerjevalnike potrebujemo, če usmerjevalnik:

- mora poznati vsa dosegljiva omrežja IP, npr. spletno omrežje,
- uporablja dinamične usmerjevalne tabele, ki jih mora posodabljati brez posredovanja skrbnikov,
- mora naznaniti lokalne spremembe v omrežju drugim usmerjevalnikom.

Naprednejši usmerjevalniki za komunikacijo med seboj uporabljajo dodatne protokole, ki jih bomo spoznali v nadaljevanju.

2.1.1 *Usmerjevalni procesi*

Usmerjevalne protokole na operacijskih sistemih, ki podpirajo protokol TCP/IP, implementiramo običajno z dvema strežniškima procesoma (angl. daemons):

- *routed* (beri route D): je osnovni usmerjevalni proces za notranje usmerjanje, ki ga implementiramo na večini operacijskih sistemov, ki podpirajo protokol TCP/IP, s protokolom RIP (angl. Routing Information Protocol),
- *gated* (beri gate D): je zahtevnejši usmerjevalni proces na operacijskem sistemu UNIX za notranje in zunanje usmerjanje, ki uporablja dodatne protokole kot OSPF (angl. Open Shortest Path First) in BGP (angl. Border Gateway Protocol).

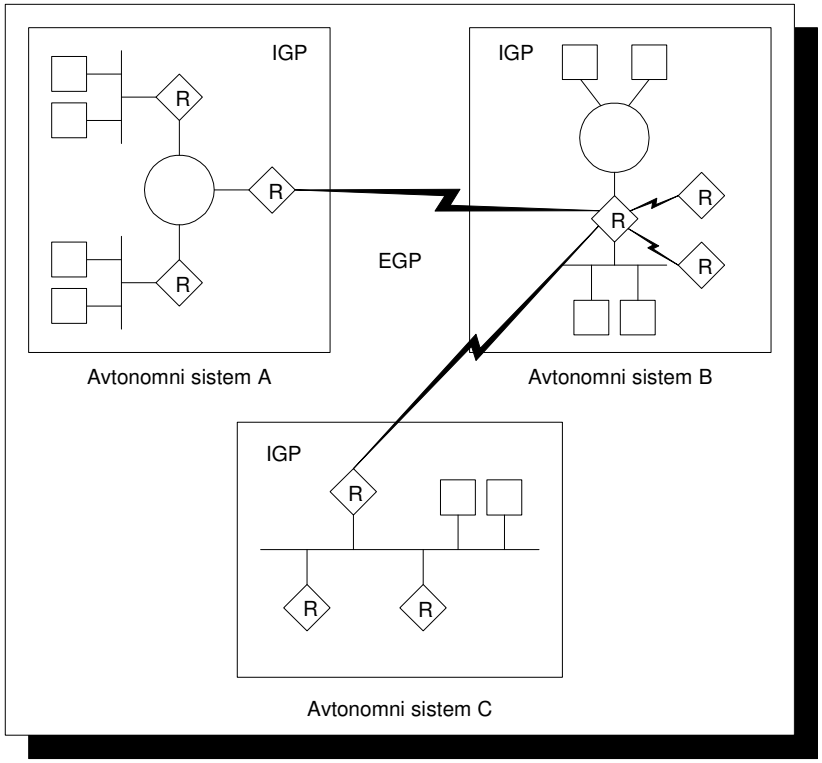
V napravah, ki so namenjene izključno usmerjanju, kot npr. Cisco IOS, usmerjevalne protokole implementiramo v samem operacijskem sistemu.

2.1.2 *Avtonomni sistemi*

Dinamične usmerjevalne protokole delimo v dve skupini:

- *notranji usmerjevalni protokol* (angl. Interior Gateway Protocol, krajše IGP): npr. OSPF in RIP,
- *zunanji usmerjevalni protokol* (angl. Exterior Gateway Protocol, krajše EGP): npr. BGP (angl. Border Gateway Protocol).

Avtonomni sistem (angl. Autonomus System, krajše AS) definiramo kot logični del večjega omrežja IP, ki predstavlja administrativno celoto. Ta podpira intranet znotraj organizacije in je zgrajen tako, da omogoča komunikacijo preko javnih naslovov IP z avtonomnimi sistemi, ki pripadajo drugim organizacijam (slika 27).



Slika 27: Avtonomni sistemi

Usmerjevalni protokol je lahko notranji (znotraj AS), ali zunanji (med AS). Notranji usmerjevalni protokol omogoča izmenjavo usmerjevalnih informacij znotraj avtonomnih sistemov, zunanji pa izmenjavo informacij med administrativno ločenimi avtonomnimi sistemi.

2.2 Usmerjevalni algoritmi

Dinamični usmerjevalni algoritmi omogočajo usmerjevalnikom izmenjavo informacij o prehodih ali povezavah, med katerimi poskušajo poiskati najboljšo pot do ponornega gostitelja v omrežju.

2.2.1 *Statično usmerjanje*

Statično usmerjanje zahteva, da se prehodi na vsakem usmerjevalniku konfigurirajo ročno. Ima to slabost, da dosegljivost omrežja ni odvisna od njegovega stanja. Če je ponorni gostitelj pri statičnem usmerjanju nedosegljiv, prehod do njega še vedno ostaja v usmerjevalni tabeli. Promet do gostitelja še vedno usmerjamo prek neobstoječega prehoda, ne da bi do gostitelja poiskali alternativna pot. Statično usmerjanje uporabljamo za:

- definicijo privzetih prehodov ali prehodov, ki niso bili naznanjeni v omrežju,
- dopolnjevanje ali nadomeščanje protokola EGP, ko:
    a. je cena linij med avtonomnimi sistemi previsoka in želimo znižati stroške prometa usmerjevalnega protokola,
    b. implementiramo kompleksno usmerjevalno politiko,
    c. se izognemo prekinitvam nastalih zaradi napak zunanjih usmerjevalnikov v drugih avtonomnih sistemih.

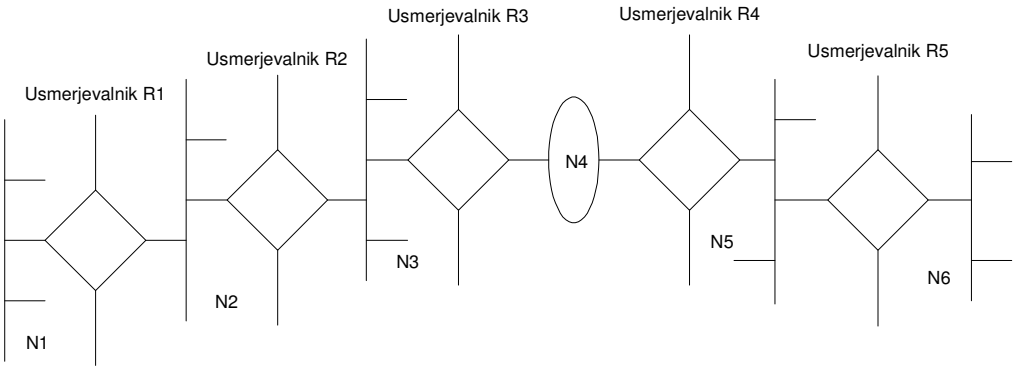
2.2.2 *Usmerjanje na osnovi vektorja razdalj*

Princip usmerjanja na osnovi vektorja razdalj (angl. Distance Vector Routing) je enostaven - vsak usmerjevalnik v omrežju hrani informacije o razdaljah, ki ga ločijo do nekega znanega ponora, v t.i. tabeli *vektorjev razdalj*. Te tabele sestojijo iz zaporedja vozlišč (vektor) in pripadajočih cen (razdalja). Iz njih usmerjevalnik v času prenosa izračuna minimalno ceno, ki je potrebna, da paket pride do ponora. Razdalje v tabelah izračunava algoritem iz informacij, ki jih dobi od sosednjih usmerjevalnikov. Vsak usmerjevalnik v omrežje pošlje lastno tabelo vektorja razdalj. Zaporedje operacij za ta izračun je naslednje:

- Vsak usmerjevalnik konfiguriramo s ceno za vsakega izmed omrežnih povezav. Ceno običajno fiksiramo na vrednost 1, kar označuje en skok (angl. hop), da pridemo v novo omrežje, pri izračunu cene pa lahko upoštevamo tudi druge parametre, kot npr. hitrost linije, promet, itd.
- Vsak usmerjevalnik inicializira tabelo vektorja razdalj za lokalnega gostitelja na 0, za neposredno povezavo na 1 in za vse ostale ponore na vrednost nedefinirano.
- Vsak usmerjevalnik periodično (običajno vsakih 30 sek) pošlje svojo tabelo vektorja razdalj vsakemu sosedu.
- Vsak usmerjevalnik shrani najnovejše tabele, ki jih dobi od sosedov in te informacije uporabi za izračun lastne tabele vektorjev razdalj.
- Skupno ceno do vsakega izmed ponorov usmerjevalnik izračuna s prištevanjem cene, ki jo je dobil v tabeli vektorjev razdalj soseda s ceno povezave do njega.
- Tabelo vektorja razdalj usmerjevalnik izračuna tako, da vzame pot z minimalno izračunano ceno do vsakega ponora (slika 28).

Usmerjevalni algoritem na osnovi vektorja razdalj konstruira v določenem časovnem obdobju stabilno usmerjevalno tabelo. To časovno obdobje, ki je odvisno od števila usmerjevalnikov v omrežju, imenujemo tudi *konvergenčni čas.* V velikih omrežjih lahko ta čas postane predolg in s tem vprašljiva tudi uporabnost algoritma. Kljub enostavnosti ima algoritem naslednje slabosti:

- nestabilnosti, ki jih povzročajo obstoječi mrtvi prehodi v omrežju,
- dolg konvergenčni čas na velikih omrežjih,
- omejena velikost omrežja zaradi maksimalnega števila skokov,
- prenašanje tabele vektorja razdalj, čeprav se vsebina v njej ne spremeni.



Usmerjevalnik R2: Tabela vektorja razdalj

Mreža	Naslednji skok	Metrika
N1	R1	2
N2	-	1
N3	-	1
N4	R3	2
N5	R3	3
N6	R3	4

Usmerjevalnik R3: Tabela vektorja razdalj

Mreža	Naslednji skok	Metrika
N1	R2	3
N2	R2	2
N3	-	1
N4	-	1
N5	R4	2
N6	R4	3

Usmerjevalnik R4: Tabela vektorja razdalj

Mreža	Naslednji skok	Metrika
N1	R3	4
N2	R3	3
N3	R3	2
N4	-	1
N5	-	1
N6	R5	2

Slika 28: vektor razdalj – izračun usmerjevalne tabele

2.2.3 *Usmerjanje na osnovi stanja povezav*

Algoritem usmerjanja na osnovi stanja povezav (angl. Link State Routing) pomeni zamenjavo usmerjanja na osnovi vektorja razdalj in temelji na *stanju povezav*. Imenujemo ga tudi algoritem *najprej najkrajša pot* (angl. Shortest Path First, krajše SPF). Najboljšo izvedbo tega algoritma predstavlja OSPF (angl. Open Short Path First), t.j. algoritem *najprej odprta najkrajša pot*. Princip usmerjanja na osnovi stanja povezav je v bistvu enostaven, sama implementacija pa lahko postane zelo kompleksna:

- usmerjevalniki povezujejo sosede in spoznavajo njihovo identiteto,
- usmerjevalniki kreirajo pakete stanja povezav, ki vsebujejo seznam mrežnih povezav in pripadajočih cen,
- pakete stanja povezav algoritem pošlje vsem usmerjevalnikom v mreži,
- vsi usmerjevalniki imajo podoben seznam povezav v omrežju in lahko konstruirajo podobno topološko karto,
- topološko karto algoritem uporabi za izračun najkrajše poti do vseh ponornih gostiteljev v omrežju.

Usmerjevalniki navežejo stike z usmerjevalniki v skupnem omrežju tako, da pošljejo v omrežne vmesnike paket *hello*. Ta paket usmerjevalni algoritem neposredno pošlje usmerjevalnikom na linijah PPP (angl. Point-to-Point) in na omrežja, ki ne podpirajo naslovov *broadcast*. V lokalna omrežja LAN pošlje paket na naslov *multicast*, ki ga

sprejmejo vsi usmerjevalniki v skupnem omrežju. Usmerjevalnik, ki sprejme paket *hello*, odgovori s paketom istega tipa, ki vključuje njegovo identiteto. Usmerjevalnika, ki sta stopita v kontakt s paketom *hello*, lahko izmenjata informacije o stanju povezav. Te informacije izmenjata v obliki paketov LSP (angl. Link State Packet). Pakete LSP tvorimo iz podatkov podatkovne baze, v kateri je shranjena topološka karta omrežja in jih pošiljamo pod pogojem, da:

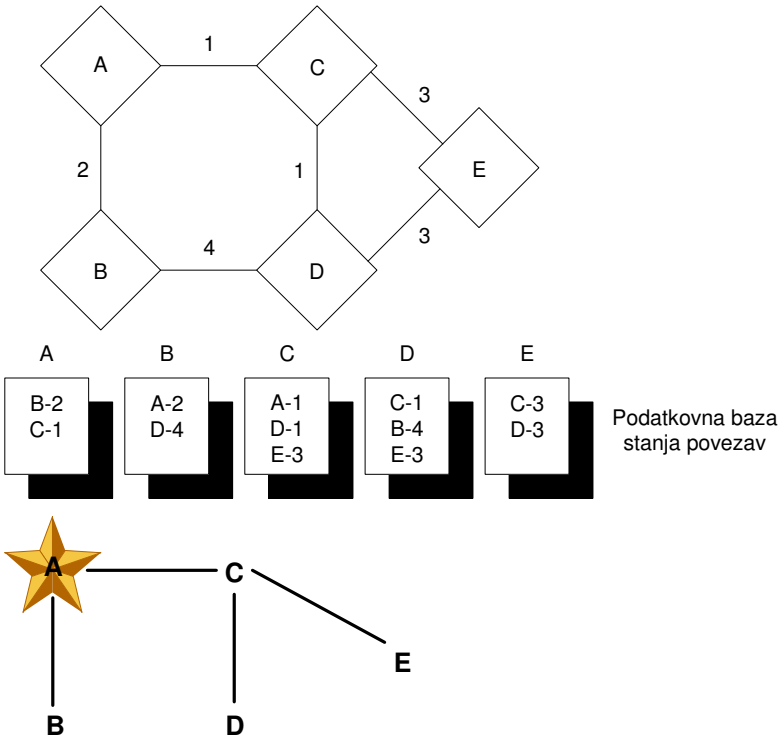
- usmerjevalnik odkrije nov usmerjevalnik v skupnem omrežju,
- povezava s sosednjim usmerjevalnikom pade,
- povezavi spremenimo ceno,
- gre za osnovne osveževalne pakete, ki jih običajno pošiljamo vsakih 30 minut.

Ko usmerjevalnik ustvari paket LSP, je zelo pomembno, da ga sprejmejo vsi ostali usmerjevalniki v omrežju. Če se to ne zgodi, usmerjevalniki izračunajo topološko karto omrežja osnovano na napačnih informacijah o stanju povezav. Razpošiljanje paketov LSP temelji na pravilnih usmerjevalnih tabelah. Tukaj pa se pojavi problem koklje in jajca, t.j. kreiranje usmerjevalnih tabel temelji na paketih LSP, paketi LSP pa so odvisni od usmerjevalnih tabel. Ta problem rešuje enostavna shema, ki jo imenujemo *poplava* (angl. flooding). Shema zagotavlja, da pakete LSP uspešno prenesemo na vse usmerjevalnike v omrežju. Usmerjevalnik, ki je paket LSP sprejel, tega pošlje vsem usmerjevalnikom v skupnem omrežju, razen pošiljatelju. Ti usmerjevalniki morajo uspešni sprejem paketov LSP potrditi.

Ko usmerjevalnik od usmerjevalnika v skupnem omrežju sprejme paket LSP, najprej v podatkovni bazi preveri zaporedno številko zadnjega paketa LSP, ki ga je dobil od njega. Če je zaporedna številka sprejetega paketa LSP ista kot v podatkovni bazi, paket LSP uniči. Proces *poplave* s tem zagotavlja, da imajo vsi usmerjevalniki v omrežju iste informacije o stanju povezav, kar zagotavlja, da lahko določeni usmerjevalnik točno izračuna najkrajšo pot v topološkem drevesu.

Najprej najkrajša pot

Usmerjevalni algoritem najprej najkrajša pot (angl. Shortest Path First, krajše SPF) je algoritem, pri katerem ima vsak usmerjevalnik v istem avtonomnem sistemu identično podatkovno bazo stanja povezav, kar pripelje do identične grafične predstavitve strukture drevesa z lokalnim usmerjevalnikom kot korenom drevesa. Iz tega drevesa lahko algoritem izračuna najkrajšo pot do ponornega omrežja ali gostitelja. Strukturo drevesa (vpeto drevo) imenujemo tudi drevo najkrajših poti. Slika 28 prikazuje najkrajše poti drevesa s korenom usmerjevalnikom A.



Slika 28: primer SPF

2.3 Interni usmerjevalni protokoli

Danes obstaja veliko standardnih in zaščitenih internih usmerjevalnih protokolov (angl. Interior Gateway Protocol, krajše IGP). Standardni IGP protokoli so:

- protokol za izmenjavo usmerjevalnih informacij (angl. Routing Information Protocol, krajše RIP),
- protokol za izmenjavo usmerjevalnih informacij verzije 2 (angl. Routing Information Protocol Version 2, krajše RIP-2),
- najprej odprta najkrajša pot (angl. Open Short Path First, krajše OSPF).

2.3.1 *RIP*

RIP je protokol usmerjanja na osnovi vektorja razdalj, primeren za manjša omrežja. Obstajata dve verziji tega protokola:

- *Verzija 1*: zelo razširjen protokol s številnimi znanimi omejitvami (v nadaljevanju RIP).
- *Verzija 2*: izboljšana verzija, razvita zato, da bi ublažila omejitve predhodne verzije in ostala z njo kompatibilna (v nadaljevanju RIP-2).

RIP je zelo razširjen, saj je bila koda protokola (znana kot proces *routed*) vključena v operacijske sisteme BSD UNIX in v vse operacijske sisteme UNIX, ki so temeljili na njem.

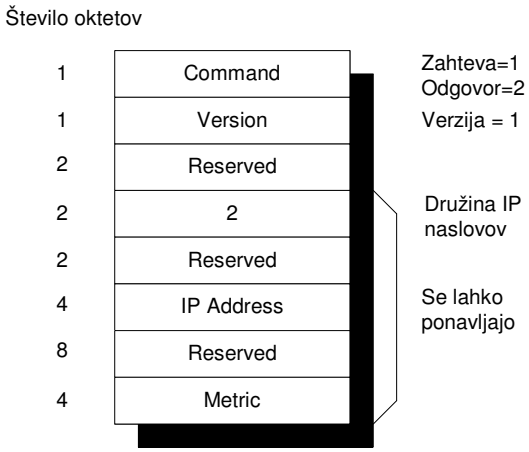
Opis protokola

Usmerjevalniki pošiljajo pakete RIP v omrežje kot datagrame UDP, ki jih nosijo datagrami IP. RIP pošilja in sprejema datagrame UDP prek vrat 520. Maksimalna velikost teh je 512 oktetov, zato sporočila, ki so večja od 512 oktetov, razbije v več zaporednih datagramov UDP. Datagram RIP je običajen paket z naslovom *broadcast*, ki ga lahko pošljemo v LAN. RIP v usmerjevalnikih teče običajno v aktivnem načinu, t.j. sporoča svojo lastno tabelo vektorjev razdalj in jo spreminja glede na informacije, ki jih dobi od sosedov. Končna vozlišča (angl. end-nodes) običajno delujejo v pasivnem (angl. silent, t.j. nemem, tihem) načinu, kar pomeni, da spreminjajo svoje tabel vektorjev razdalj glede na informacije, ki jih dobijo od sosedov, toda brez povratnega naznanjanja.

RIP omogoča pošiljanje dveh vrst paketov: *zahtevo* in *odgovor*. Usmerjevalnik pošlje usmerjevalnikom v skupnem omrežju zahtevo po delu ali celotni vsebini njihovih tabel vektorjev razdalj. Usmerjevalniki pošiljajo svoje tabele v naslednjih primerih:

- avtomatsko, npr. vsakih 30 sekund,
- kot odgovor na zahtevo,
- ko se spremeni vsebina tabele vektorjev razdalj.

Aktivna in pasivna vozlišča poslušajo vse odgovore naslovljene na njih in glede na vsebino sporočil spreminjajo lastne tabele vektorjev razdalj. Pot do določenega ponornega gostitelja, ki ga je usmerjevalnik izračunal s pomočjo tabele vektorjev razdalj, ostane v tabeli tako dolgo, dokler usmerjevalnik ne najde boljše alternativne poti ali pa te poti ni v vsaj šest zaporednih ciklih odgovorov RIP. V tem primeru usmerjevalnik predpostavlja, da gostitelj ni več na voljo in pot do njega zbriše. Format paketa RIP prikazuje slika 29.



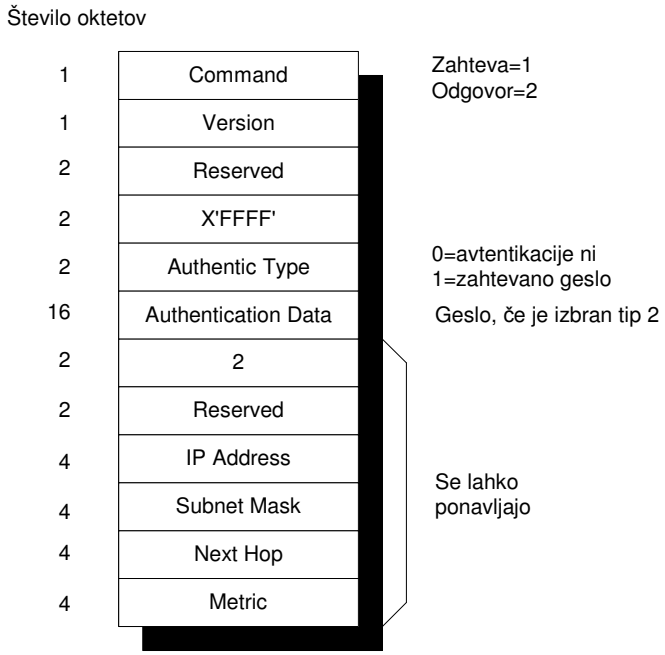
Slika 29: Format sporočila RIP

Protokol RIP iz tabel vektorjev razdalj ne zagotavlja prenosa mask podomrežja. Usmerjevalnik, ki sprejme odgovor RIP, mora informacije o maski podomrežja že imeti, da bi lahko pravilno interpretiral naslov IP. Ta pomanjkljivost pa pomeni, da RIP ne moremo uporabiti v omrežjih z variabilno dolžino mask podomrežij.

2.3.2 *RIP-2*

RIP-2 je razširitev protokola RIP. Cilj protokola RIP-2 je, da omogoča enostavno zamenjavo protokola RIP, ki ga uporabljamo v manjših in srednje velikih omrežjih, da je uporaben v omrežjih z variabilno masko podomrežja in kar je najvažnejše, da je *interoperabilen* s protokolom RIP, kar pomeni, da oba protokola delujeta na istih podatkovnih strukturah.

RIP-2 izkorišča dejstvo, da je polovica bajtov v RIP sporočilih rezerviranih (morajo biti postavljene na 0) in da je originalna specifikacija RIP že v zasnovi predvidevala razširitve formata sporočil. Format sporočil RIP-2 prikazuje slika 30.



Slika 30: Format sporočila RIP-2

Prvo polje, ki sledi 32-bitni glavi obeh sporočil RIP je lahko avtentikacijski zapis, kot v primeru RIP-2, ali prehod, kot v primeru RIP. Če je zapis avtentikacija, v sporočilo lahko vključimo 24 prehodov, v nasprotnem primeru pa 25 (RIP). Polja sporočil RIP-2 so enaka kot polja sporočil RIP, razen naslednjih:

Version
Verzija, ki pove usmerjevalnikom RIP, da rezervirana polja ignorirajo.

Address Family
Polje mora biti postavljeno na vrednost X'FFFF', kar pomeni, da gre za avtentikacijski zapis.

Authentication Type
Definira, kako uporabimo naslednjih 16 bajtov. Edina definirana tipa sta 0, ki označuje, da ni avtentikacije in 2, ki označuje, da polje vsebuje geslo.

Authentication Data
Geslo je 16-bitno, tekstovno, levo poravnano in z ASCII X'00' dopolnjeno polje.

Subnet Mask
Maska podomrežja, ki se nanaša na ta zapis.

Next Hop
Priporočilo usmerjevalniku o naslednjem skoku, ki naj bi ga uporabil za pošiljanje datagrama do podomrežja ali gostitelja v tem zapisu.

RIP-2 namesto naslavljanja *broadcasting* raje uporablja *multicasting*, saj s temi paketi ne obremenjuje gostiteljev, ki ne čakajo sporočil RIP-2. *Multicasting* lahko konfiguriramo za vsak vmesnik posebej.

2.3.3 *OSPF*

Usmerjevalni algoritem OSPF, t.j. najprej odprta najkrajša pot, (angl. Open Shortest Path First, krajše OSPF) je interni usmerjevalni protokol pomemben zaradi številnih možnosti, ki jih ostali notranji protokoli ne podpirajo. Te dodatne možnosti algoritmu OSPF dajejo prednost pri novih spletnih implementacijah, posebno še v velikih omrežjih. OSPF podpira naslednje možnosti (Chappell, 1999):

- podpora usmerjanju glede na tip storitve (angl. Type of Service, krajše TOS),
- izenačevanje obremenitve (angl. load balancing),
- krajevno delitev omrežja v podomrežja z uporabo območij (angl. area),
- izmenjava informacij med usmerjevalniki zahteva avtentikacijo,
- definiranje navideznih povezav (angl. virtual links) za podporo nezveznih območij,
- uporaba mask podomrežja spremenljive dolžine,
- vključevanje prehodov RIP in EGP v svoje podatkovne baze.

Terminologija OSPF

Protokol OSPF uporablja določeno terminologijo, ki jo pred samim opisom delovanja protokola podrobneje obdelamo v nadaljevanju.

Območja (angl. Areas)
Internetna omrežja OSPF so organizirana v območja. Ta območja sestojijo iz števila omrežij in usmerjevalnikov, ki predstavljajo celoto. Definiramo jih lahko

krajevno, funkcionalno ali administrativno. Vsako omrežje OSPF sestoji iz vsaj enega območja – hrbtenice in več dodatnih območij, kot to zahteva topologija omrežja ali kakšni drugi kriteriji. Vsi usmerjevalniki znotraj območja OSPF vzdržujejo isto topološko podatkovno bazo - izmenjujejo informacije o stanju povezav, da bi se med seboj sinhronizirali. To omogoča, da vsi usmerjevalniki računajo z isto topološko karto tega območja.

Informacije o omrežjih zunaj območja OSPF zbirajo mejni usmerjevalniki (angl. Area Border ali AS Boundary Routers) in jih razlijejo prek območja. Notranji usmerjevalniki, t.j. usmerjevalniki znotraj območja OSPF, ne poznajo topologije izven tega območja, ampak samo prehode do ponorov, ki jih določajo mejni usmerjevalniki. Pomembna lastnost koncepta območja je, da omejuje velikost topološke podatkovne baze, ki jo morajo vzdrževati usmerjevalniki. Velikost topološke podatkovne baze ima neposredni vpliv na čas procesiranja, ki ga mora vsak usmerjevalnik izvesti in na količino informacij, ki se preliva preko omrežja.

Hrbtenica OSPF (angl. Backbone)

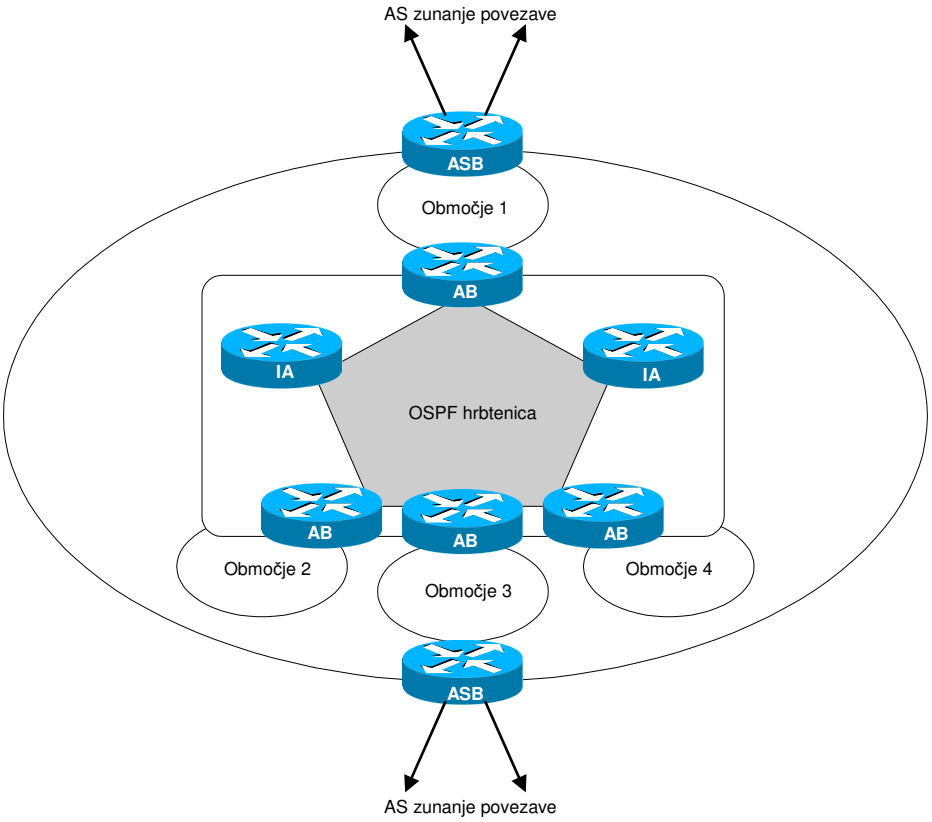
Vsako omrežje OSPF ima vsaj eno območje – hrbtenico, ki ji priredi naslov 0.0.0.0 (ta vrednost se razlikuje od naslova IP 0.0.0.0). Hrbtenica ima vse lastnosti območja in dodatno nalogo pošiljanja informacij o prehodih med območji, ki so ji pridruženi. Hrbtenica OSPF je običajno zvezna, t.j. vsa območja so preko usmerjevalnikov priključena nanjo. To pa zaradi mrežne topologije večkrat ni mogoče. V takem primeru simuliramo zveznost s t.i. navideznimi povezavami (angl. Virtual Links).

Notranji, področni in mejni usmerjevalniki (angl. Intra-Area, Area Border and AS Boundary Routers): V omrežjih OSPF obstajajo trije tipi usmerjevalnikov: notranji, področni in mejni usmerjevalniki (slika 31).

Notranji usmerjevalniki (angl. Intra-Area Routers, krajše IA): imajo vse vmesnike v istem območju OSPF. Vsi notranji usmerjevalniki pošiljajo naznanila stanja svojih povezav v območju, da bi definirali povezave, na katere je priključen. Če so usmerjevalniki označeni kot *določeni usmerjevalniki* (angl. Designated Router, krajše DR) ali *rezervni določeni usmerjevalniki* (angl. Backup Designated Router, krajše BDR), pošiljajo naznanila stanja povezav, da bi ostali usmerjevalniki lahko definirali identiteto vsakega usmerjevalnika priključenega na omrežje. Notranji usmerjevalniki skrbijo za topološko podatkovno bazo območja, v katerem se nahajajo.

Področni usmerjevalniki (angl. Area Border Router, krajše AB): povezujejo dve ali več območji. Področni usmerjevalniki vzdržujejo ločene topološke podatkovne baze za območja, na katera so priključeni in usmerjajo promet, ki ima svoj ponor v drugih območjih ali prihaja iz drugih območij. Področni usmerjevalnik je izhodna točka območja, t.j. usmerjevalne informacije, ki imajo ponor v drugih območjih, lahko gredo tja samo preko njega. Poleg tega povzema informacije iz

topološke baze svojih pridruženih območij in jih distribuira v hrbtenico. Hrbtenični AB posreduje te informacije do vseh ostalih pridruženih območij.



Slika 31: Omrežje OSPF

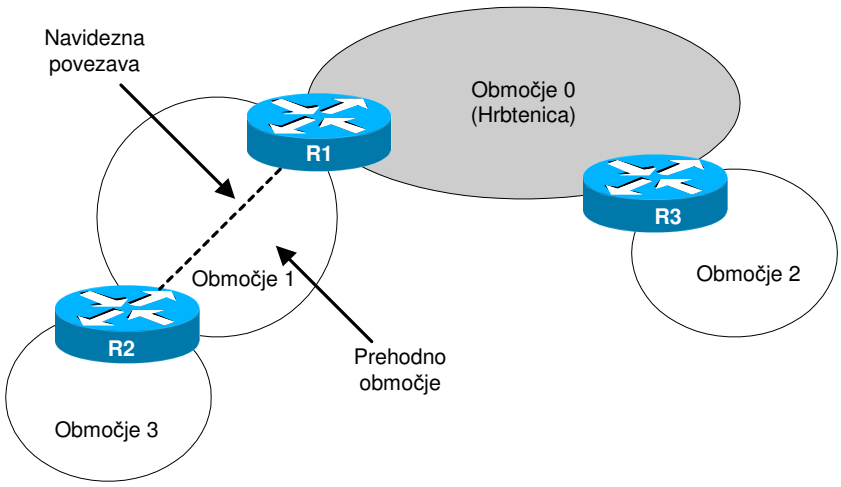
Mejni usmerjevalniki (angl. Autonomous System Boundary Router, krajše ASB): imajo vsaj enega izmed vmesnikov v zunanjem omrežju (drugem AS), npr. omrežju, ki ne podpira protokola OSPF. Ti usmerjevalniki lahko vključujejo (angl. redistribution) omrežne informacije, ki niso OSPF, na omrežje OSPF in obratno. Usmerjevalniki, ki vključujejo statične prehode ali prehode iz drugih notranjih usmerjevalnih protokolov IGP, npr. RIP, v OSPF omrežje, so prav tako mejni usmerjevalniki. Mejni usmerjevalnik posreduje naznanila stanja zunanjih povezav v vsa področja AS.

Navidezna povezava (angl. Virtual Link)

Večkrat se lahko zgodi, da v delujoče omrežje OSPF dodamo novo območje, ki ga ni mogoče neposredno povezati na hrbtenično omrežje. V tem primeru lahko definiramo navidezno povezavo (slika 32), ki temu neodvisnemu področju omogoča logično pot do hrbtenice. Vsa območja na hrbtenici morajo biti povezana neposredno ali prek prehodnega območja, ki nastane z navidezno povezavo. Navidezna povezava ima naslednji dve zahtevi:

- vzpostavljena mora biti med dvema usmerjevalnikoma, ki si delita skupno območje,

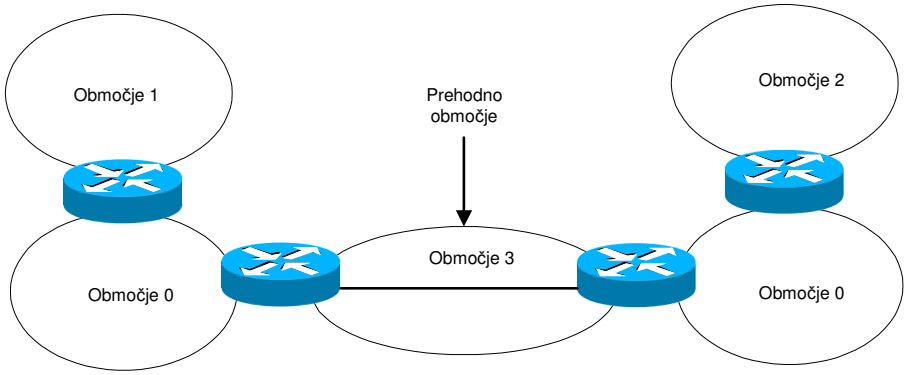
- eden izmed usmerjevalnikov mora biti povezan na hrbtenico.



Slika 32: Navidezna povezava

Navidezne povezave so namenjene:

- povezovanju območja, ki nima fizične povezave na hrbtenico. To se lahko zgodi, ko se dve organizaciji združujeta,
- popravljanje hrbtenice v primeru, da nastopi delitev območja 0 (slika 33).



Slika 33: Prehodno območje

Slika 33 prikazuje drugo možnost. Delitev hrbtenice lahko nastopi v primeru, ko dve organizaciji, ki uporabljata protokol OSPF, poskušata združiti dve ločeni omrežji v eno skupno hrbtenično omrežje. Naslednji razlog kreiranja navideznih povezav je dodajanje redundance v primerih, ko napaka usmerjevalnika povzroči delitev hrbtenica na dva dela.

Prehodno območje

Področje, skozi katerega je navidezna povezava fizično povezana (slika 33).

Usmerjevalnik v skupnem omrežju

Za usmerjevalnika, ki imata vmesnike v skupnem omrežju, pravimo, da sta usmerjevalnika v skupnem omrežju. Usmerjevalniki preiskujejo svojo okolico dinamično s protokolom *Hello*. Vsak usmerjevalnik v skupnem omrežju lahko opišemo s končnim avtomatom[1] (angl. finite state machine). Stanja, ki se lahko pojavijo pri pogovoru med dvema usmerjevalnikoma v skupnem omrežju, so naslednja:

Down (brez povezave): začetno stanje pogovora z usmerjevalnikom v skupnem omrežju, ki označuje, da od njega nismo dobili še nobene informacije.

Attempt (poskus): usmerjevalnik brez povezave v omrežju *non-broadcast* poskuša stopiti v stik z usmerjevalnikom v skupnem omrežju s pošiljanjem paketov *Hello*.

Init (začetek pogovora): usmerjevalnik sprejme paket *Hello* od usmerjevalnika v skupnem omrežju, vendar dvosmerna povezava med njima še ni vzpostavljena.

2-way (dvosmerno): v tem stanju je komunikacija med obema usmerjevalnikoma v skupnem omrežju že dvosmerna, t.j. začetek vzpostavljanja soseščine.

ExStart (ponovni začetek): oba usmerjevalnika v skupnem omrežju tvorita soseščino.

Exchange (izmenjava): oba usmerjevalnika v skupnem omrežju izmenjata informacije iz svojih topoloških podatkovnih baz.

Loading (nalaganje): oba usmerjevalnika v skupnem omrežju sinhronizirata svoji topološki podatkovni bazi.

Full (polna povezava): oba usmerjevalnika sta popolna soseda, njuni topološki podatkovni bazi sta sinhronizirani.

Spremembo stanja povzročijo različni dogodki. V primeru, da usmerjevalnik v stanju *down* sprejme od usmerjevalnika v skupnem omrežju paket *Hello*, se njegovo stanje spremeni v *init* in zažene se časovnik. Če v času, ki ga ta določa, usmerjevalnik od usmerjevalnika v skupnem omrežju ne dobi nobenega paketa OSPF, se vrne v prvotno stanje.

Sosednji usmerjevalnik

Usmerjevalnik v skupnem omrežju lahko danemu usmerjevalniku postane sosed, če oba sinhronizirata topološki podatkovni bazi preko izmenjave informacij o stanju povezav. Te informacije se izmenjujejo samo med sosednjimi, ne pa tudi vsemi ostalimi usmerjevalniki v skupnem omrežju. Vsi usmerjevalniki ne postanejo sosedje, t.j. na PPP linijah se to zgodi vedno, na več-dostopnem omrežju pa samo *interni* usmerjevalniki postanejo sosedje z DR in BDR.

Določeni in rezervni določeni usmerjevalniki (Designated Routers and Backup Designated Routers, krajše DR in BDR)

Vsa več-dostopna omrežja imajo določene (angl. Designated Router, krajše DR) in rezervne določene usmerjevalnike (angl. Backup Designated Router, krajše BDR). Te usmerjevalnike za vsako omrežje avtomatsko izvoli paket *Hello* v fazi

[1] Končni avtomat je model sistema sestavljen iz končnega števila stanj, prehodov med njimi in akcij, s katerimi lahko spreminjajo stanje sistema.

odkrivanja usmerjevalnikov v skupnem omrežju. DR imajo dve ključni vlogi v omrežju:

- generirajo naznanila omrežnih povezav, ki popišejo usmerjevalnike priključene na več-dostopno omrežje,
- oblikujejo soseščino z vsemi usmerjevalniki na več-dostopnem omrežju in s tem postanejo osrednje točke za posredovanje vseh naznanil stanja povezav.

BDR oblikuje soseščino z istimi usmerjevalniki kot določeni. Tako ima isto topološko podatkovno bazo in lahko prevzame funkcije določenega v primeru, da ta pade.

Vrste fizičnih omrežij

Vsa območja OSPF sestojijo iz skupine omrežij, ki so med seboj povezane z usmerjevalniki. OSPF razporeja omrežja v naslednje tipe:

Od točke do točke (angl. Point-to-Point, PPP, tudi med dvema točkama): omrežje neposredno povezuje dva usmerjevalnika.
Več-dostopno omrežje (angl. Multi-Access Network): omogoča priključitev več kot dveh usmerjevalnikov. Nadalje ga delimo v dva tipa:
- *broadcast*,
- *non-broadcast*.

Od točke do več točk (angl. Point-to-Multipoint, P-MP): opisuje posebno vrsto več-dostopnega omrežja *non-broadcast*, kjer ne zahtevamo, da ima vsak usmerjevalnik neposredno povezavo z drugimi usmerjevalniki. Predstavimo ga lahko tudi kot zbirko omrežij PPP.

Omrežja *broadcast* so sposobna paket OSPF usmeriti na vse priključene usmerjevalnike z uporabo znanega naslova *broadcast*. Primeri več-dostopnega omrežja *broadcast* so Ethernet oz. Token-Ring lokalna omrežja.

Omrežja *non-broadcast* nimajo možnosti pošiljanja naslovov *broadcast*, zato morajo biti vsi paketi naslovljeni na ustrezne usmerjevalnike v omrežju, kar zahteva, da usmerjevalnike konfiguriramo z ustreznimi naslovi.

Vmesnik

Povezava med usmerjevalnikom in enim od priključnih omrežij. Vsak vmesnik ima pridružene informacije o svojem stanju, ki jih dobi od spodnjih nižje-ležečih protokolov in protokola OSPF samega. Vmesnik je lahko v naslednjih stanjih:

Down (nepovezan): vmesnik ni na voljo. To je začetno stanje vmesnika.
Loopback (v zanki): vmesnik je sklenjen v zanko in ga ne moremo uporabiti za običajni promet.
Waiting (čakanje): usmerjevalnik poskuša določiti identiteto DR ali BDR.

Point-to-Point (med dvema točkama): vmesnik je omrežje med dvema točkama (PPP) ali navidezno omrežje. Usmerjevalnik tvori soseščino z usmerjevalnikom na drugi strani.
DR other (drugi usmerjevalnik): vmesnik je na več-dostopnem omrežju, toda ta vmesnik ni niti DR niti BDR. Ta usmerjevalnik oblikuje soseščino z DR ali BDR.
Rezerva: je BDR, ki ga lahko izvolimo (angl. promote) v DR, če obstoječi DR pade. Ta usmerjevalnik oblikuje soseščino z vsemi drugimi usmerjevalniki v stanju *DR other* na omrežju.
DR: je določeni usmerjevalnik, ki oblikuje soseščino z vsemi drugimi usmerjevalniki na omrežju.

Tip storitev (angl. Type of Service Metrics, TOS)

V vsakem naznanilu stanja povezave lahko za vsak tip IP storitve sporočamo drugačno ceno. Ceno za hrbtenico, t.j. TOS 0, moramo določiti vedno, za ostale pa v primeru, da ni določena, privzamemo vrednost cene za TOS 0.

Drevo najkrajših poti (angl. Shortest Path Tree)

Vsak usmerjevalnik izvaja algoritem SPF na podatkovni bazi stanja povezav, da dobi drevo najkrajših poti. Struktura drevesa opisuje najkrajše poti do vseh ponornih omrežij ali gostiteljev iz območja OSPF. To strukturo uporabljamo za gradnjo usmerjevalnih tabel.

Usmerjevalna tabela (angl. Routing Table or Forwarding Database)

Vsebuje zapise za ponorna omrežja, podomrežja in gostitelje. Za vsak tip ponornega gostitelja imamo informacije o eni ali več cen. Tabelo tvori algoritem SPF iz drevesa najkrajših poti.

ID območja (angl. Area ID)

ID območja je 32-bitno število, ki označuje določeno območje omrežja OSPF. Hrbtenica omrežja ima ID območja enak 0.

ID usmerjevalnika (angl. Router ID)

ID usmerjevalnika je 32-bitno število, ki označuje določen usmerjevalnik. Vsak usmerjevalnik znotraj AS ima svoj ID.

Prioriteta usmerjevalnika (angl. Router Priority)

8-bitna število brez predznaka, ki ga določimo na osnovi vmesnikov in označuje prioriteto tega usmerjevalnika v izboru za rezervni določeni usmerjevalnik. Prioriteta 0 določa, da usmerjevalnik ni primeren za izvolitev v BDR.

Naznanila stanja povezave (angl. Link State Advertisements)

Usmerjevalniki OSPF v soseščini izmenjujejo informacije stanja povezav med seboj, da omogočajo usmerjevalnikom vzdrževati topološke podatkovne baze in obveščati o notranjih in zunanjih poteh. Informacije o stanju povezav sestojijo iz

štirih tipov naznanil stanja povezav (opisanih v nadaljevanju), ki skupaj zagotavljajo vse informacije potrebne za opis omrežij OSPF in njegove zunanje okolice:

- usmerjevalniške povezave,
- omrežne povezave,
- skupne povezave,
- zunanje povezave AS.

Naznanila usmerjevalniških povezav: generirajo vsi usmerjevalniki OSPF in opisujejo stanja usmerjevalniških vmesnikov v območju OSPF. Ta naznanila se širijo samo skozi notranje območje.
Naznanila omrežnih povezav: generirajo določeni usmerjevalniki v več-dostopnem omrežju OSPF in popišejo usmerjevalnike priključene na dano omrežje. Ta naznanila se širijo samo skozi notranje območje.
Naznanila skupnih povezav: generirajo področni usmerjevalniki in so lahko dveh tipov: (1) opisuje poti do ponorov v drugih območjih, (2) opisuje poti do mejnih usmerjevalnikov AS. Ta naznanila se širijo samo skozi notranje območje.
Naznanila zunanjih povezav AS: generirajo mejni usmerjevalniki AS in opisujejo poti do zunanjih ponorov v omrežju OSPF. Ta naznanila se širijo skozi vsa območja v omrežju OSPF.

Opis protokola OSPF

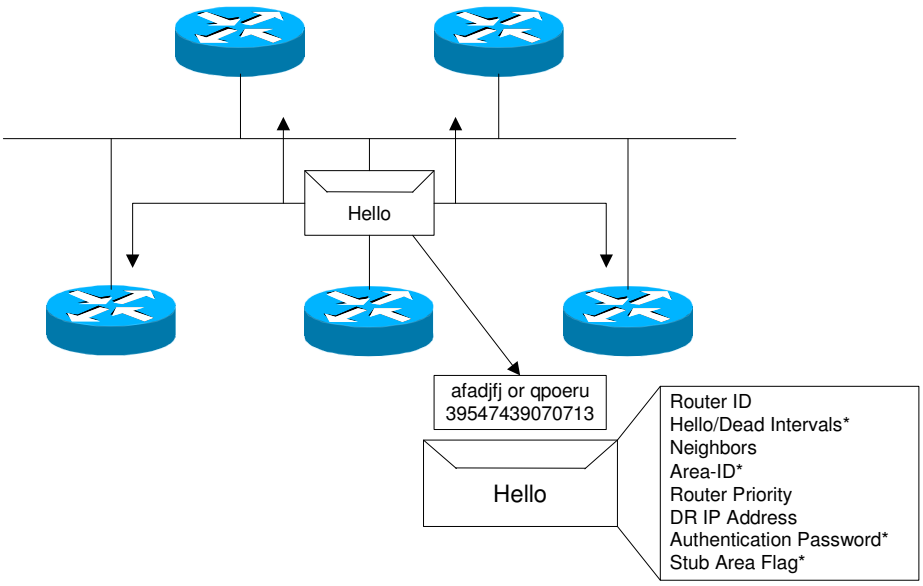
Protokol OSPF je sestavljen iz naslednjih korakov:

Korak 1: Vzpostavitev soseščine usmerjevalnikov.
Korak 2: Izvolitev DR in BDR.
Korak 3: Odkrivanje poti.
Korak 4: Izbiranje primerne poti.
Korak 5: Vzdrževanje usmerjevalnih informacij.

V nadaljevanju opisujemo posamezne korake podrobneje.

Korak 1: Vzpostavitev soseščine usmerjevalnikov.

Ker je usmerjanje OSPF odvisno od stanja povezav med dvema usmerjevalnikoma, se morata usmerjevalnika v skupnem omrežju, preden začneta izmenjevati informacije med seboj, najprej prepoznati. Usmerjevalnika v skupnem omrežju sta po definiciji tista usmerjevalnika, ki sta priključena na isto omrežje (slika 34).



Slika 34: Pošiljanje paketov Hello

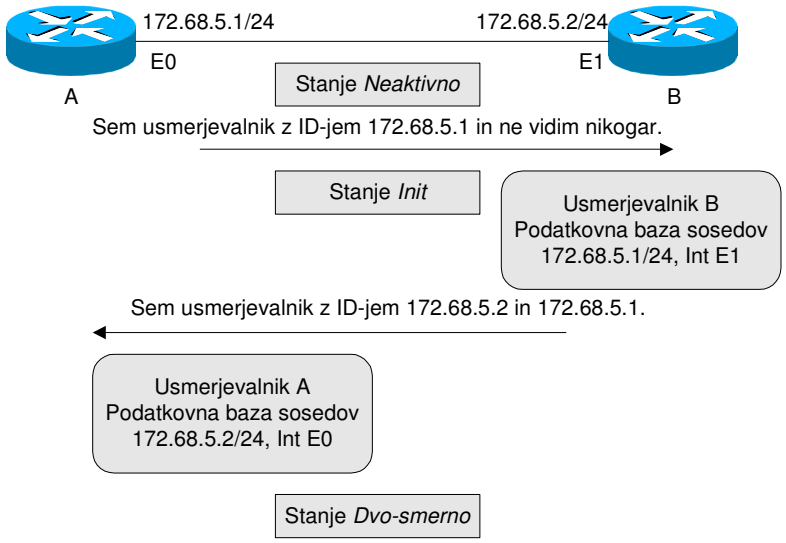
Prepoznavanje med dvema usmerjevalnikoma izvedemo s protokolom *Hello*, ki je del standardnega protokola OSPF. Ta protokol omogoča usmerjevalnikom v skupnem omrežju vzpostavljanje povezav (soseščine) in zagotavlja dvosmerno medsebojno komunikacijo pred izmenjavo informacij stanja povezav.

POMNI: *območje OSPF je zbirka omrežij in usmerjevalnikov, ki imajo isti ID območja.*

Pakete *Hello* periodično pošiljamo prek vseh vmesnikov z uporabo naslovov IP *multicast*. Naslove *multicast* uporabljamo za pošiljanje paketov skupini naprav. Proces izmenjave med aktivnimi usmerjevalniki je naslednji (slika 35):

- Usmerjevalnik A je na omrežju LAN aktiven, vendar je stanje OSPF neaktivno, ker nima odprte izmenjave informacij z ostalimi usmerjevalniki. Usmerjevalnik A pošlje paket *Hello* na omrežje.
- Vsi usmerjevalniki OSPF prek naslova IP *multicast* sprejmejo od usmerjevalnika A paket *Hello* in ga dodajo v svojo podatkovno bazo sosedov. To je stanje *Init*.
- Vsi usmerjevalniki, ki so sprejeli poslani paket, pošljejo usmerjevalniku A odgovor, t.j. paket *Hello* s svojimi pripadajočimi informacijami.
- Ko usmerjevalnik A sprejme pakete od usmerjevalnikov v omrežju OSPF, doda vse usmerjevalnike, ki imajo isti ID usmerjevalnika v njegovem paketu, v svojo podatkovno bazo sosedov. Ta postopek imenujemo tudi kot *dvo-smerno* stanje. V tej točki vsi usmerjevalniki, ki imajo izvorni naslov paketa že v podatkovni bazi, z njim vzpostavijo dvosmerno komunikacijo.
- Usmerjevalniki izvolijo DR in BDR. Ta proces mora nastopiti preden se začne proces izmenjave informacij stanja povezav.

- Periodično (privzeto vsakih 10 sekund) usmerjevalniki v omrežju izmenjujejo pakete *Hello*, da bi se prepričali ali komunikacija še vedno teče. Ti periodični paketi vsebujejo tudi ID določenega usmerjevalnika DR in seznam usmerjevalnikov, katerih paket *Hello* je usmerjevalnik, ki je oddal sporočilo, sprejel.

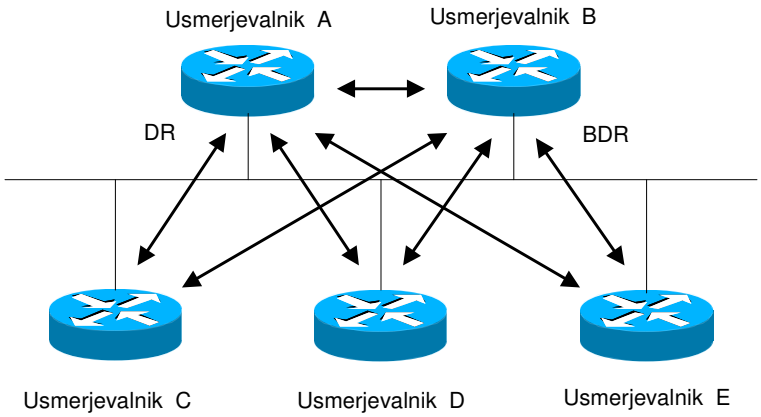


Slika 35: vzpostavitev soseščine usmerjevalnikov

Korak 2: Izvolitev DR in BDR.

Ko se usmerjevalniki prvič priključijo na omrežje, najprej raziščejo omrežje s pošiljanjem paketov *Hello*. Istočasno morajo izvoliti DR ali BDR in ga vpisati v podatkovno bazo stanj povezav. DR in BDR izboljšata delovanje omrežja na sledeči način:

- *zmanjšata promet usmerjevalnih informacij*: DR in BDR igrata vlogo osrednjih točk izmenjave stanj povezav na danem omrežju, zato mora vsak usmerjevalnik vzpostaviti soseščino z DR/BDR (slika 36).

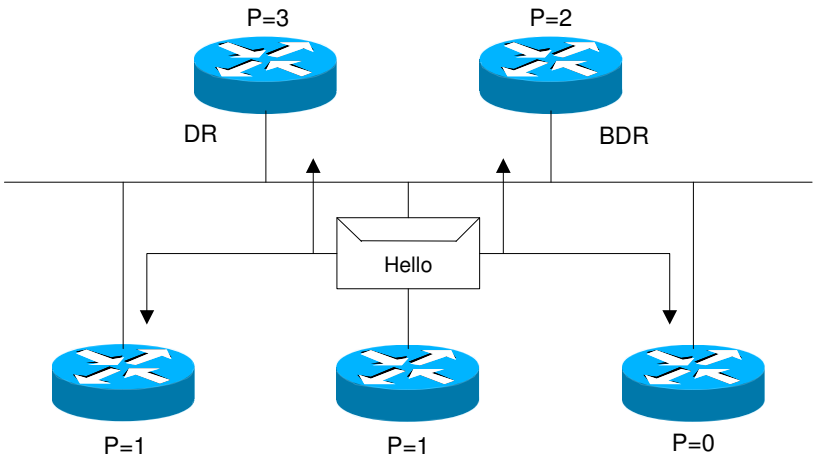


Slika 36: formiranje soseščine usmerjevalnikov z DB in BDR

Namesto, da bi usmerjevalnik izmenjeval informacije stanja povezav z vsemi preostalimi usmerjevalniki v segmentu, pošlje informacije o svojem stanju DR in BDR. DR vsem usmerjevalnikom v omrežju pošlje informacije stanja povezav. Ta proces poplave informacij občutno zmanjša promet usmerjevalnih informacij v segmentu.

- *vodita sinhronizacijo stanja povezav*: DR in BDR zagotavljata, da si vsi usmerjevalniki na omrežju delijo iste informacije o stanju povezav, s čimer število napak zmanjšamo.

BDR ne opravlja funkcije DR, dokler je DR operativen. V tem času sicer sprejema vse informacije, vendar dopušča, da DR opravlja posle posredovanja in sinhronizacije. BDR opravlja posle DR samo v primeru, da DR izpade. Za izvolitev DR in BDR usmerjevalniki med procesom izmenjave paketa *Hello* pregledajo vrednosti prioritet (slika 37). Usmerjevalnik s prioriteto 3 na sliki 37 ima najvišjo prioriteto in postane določeni usmerjevalnik DR. Usmerjevalnik z naslednjo najvišjo prioriteto (prioriteto 2) je rezervni določeni usmerjevalnik BDR. Usmerjevalnik s prioriteto 0 ne more postati niti DR niti BDR.



Slika 37: prioritete usmerjevalnikov

Za določitev, kateri usmerjevalnik izvolimo za DR ali BDR, so pomembni naslednji pogoji:

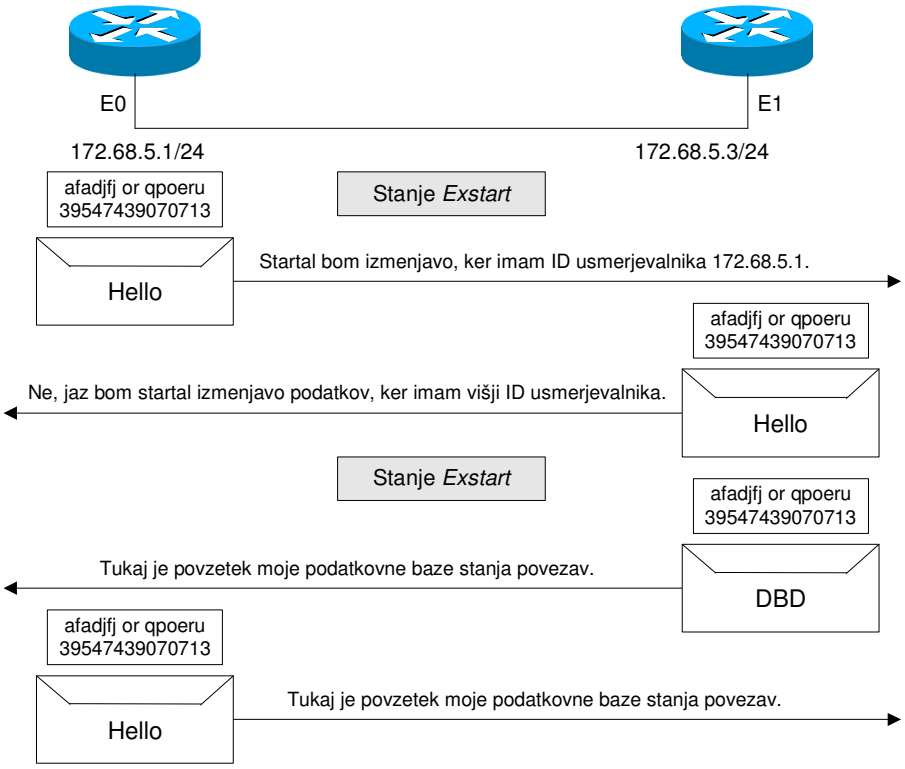
- Usmerjevalnik z najvišjo prioriteto postane DR.
- Usmerjevalnik z naslednjo najvišjo prioriteto postane BDR.
- Privzeta prioriteta vmesnika OSPF je 1. V primeru izenačenja postane DR usmerjevalnik z višjim ID usmerjevalnika,
- Če v omrežje dodamo usmerjevalnik z višjo vrednostjo prioritet, se DR in BDR ne spremenita. Edini način, da se zamenjata je, da izklopimo enega izmed njiju. Če izklopimo DR, prevzame njegovo funkcijo BDR, če pa izklopimo BDR, izvolimo drugi BDR.

- Da bi BDR ugotovil ali je DR aktiven, postavi časovnik. Če BDR ne sprejme posredovanega naznanila stanja povezave od DR preden časovnik poteče, BDR predpostavlja, da DR ni več aktiven.

Vsak omrežni segment v okolju *multicast* ima svoj lasten DR in BDR. Usmerjevalnik, ki je priključen na več omrežij, je lahko določen usmerjevalnik v enem segmentu in navaden usmerjevalnik v drugem.

Korak 3: Odkrivanje poti.

Ko DR in BDR izvolimo, so usmerjevalniki v stanju *Exstart* (slika 38). Pripravljeni so za odkrivanje informacij o stanju povezav v omrežju in kreiranje svoje podatkovne baze stanja povezav.

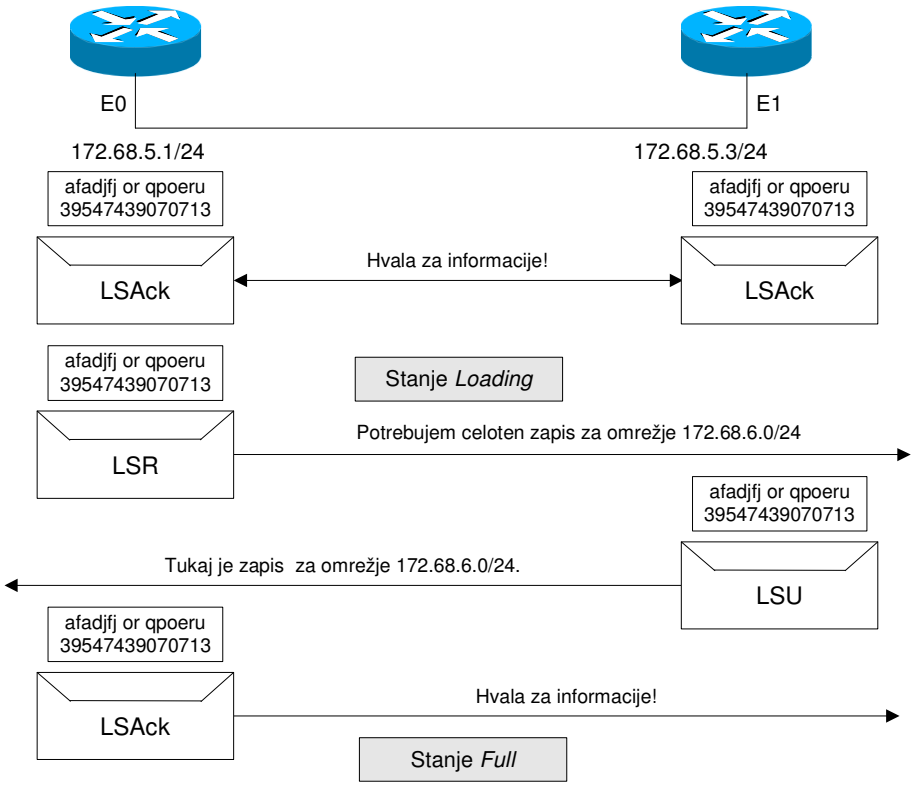


Slika 38: odkrivanje poti

Proces, ki ga uporabljamo pri odkrivanju omrežnih poti, imenujemo protokol *izmenjave.* Ta postavi usmerjevalnik v komunikacijsko stanje *Full*. Ko je usmerjevalnik v stanju *Full*, protokola izmenjave ne uporablja, dokler se stanje ne spremeni. Protokol izmenjave deluje na naslednji način:

- DR in BDR v stanju *Exstart* vzpostavita soseščino z vsakim usmerjevalnikom v omrežju. Med tem procesom DR/BDR z vsakim usmerjevalnikom tvori relacijo gospodar/hlapec (angl. master/slave). Usmerjevalnik z višjo številko ID igra vlogo gospodarja. Informacije stanja povezav izmenjujemo in sinhroniziramo med DR/BDR in usmerjevalniki, ki imajo vzpostavljeno soseščino.

- Usmerjevalniki v vlogi hlapcev z gospodarjem izmenjajo enega ali več opisnih paketov podatkovne baze (angl. DataBase Description Packet, krajše DBD, ali Datagram Delivery Protocol, krajše DDP). Ta proces poteka v stanju izmenjave. DBD vključuje zapise naznanil stanja povezav LSA, ki se pojavljajo v gospodarjevi podatkovni bazi stanja povezav. Zapisi lahko zajemajo povezave ali omrežja. Vsak zapis LSA vključuje stvari, kot so: tip stanja povezav, naslov obveščevalnega usmerjevalnika, cena povezave in zaporedna številka. Zaporedna številka je način, kako usmerjevalnik določi »novosti« v sprejetih informacijah stanja povezav in jo definira gospodar, uporablja pa hlapec.
- Ko usmerjevalnik v vlogi hlapca sprejme paket DBD, naredi naslednje:
  - s ponovitvijo zaporedne številke zapisa stanja povezave v paketu LSA potrdi sprejem tega paketa DBD (slika 39).
  - primerja sprejete informacije z obstoječimi informacijami. Ponovimo, da se začetni zapisi v podatkovno bazo napolnijo iz podatkovnih baz sosedov. Če ima paket DBD več posodobljenih zapisov stanja povezav, hlapec gospodarju pošlje zahtevo stanja povezav.
  - usmerjevalnik v vlogi gospodarja odgovori s celotno informacijo o zahtevanem zapisu v paketu spremenjenega stanja povezave (angl. link-state update, krajše LSU). Ko usmerjevalnik hlapec sprejme paket LSU, pošlje potrditev, t.j. paket LSAck. Proces pošiljanja paketa LSR imenujemo tudi stanje *Loading*.



Slika 39: tvorjenje podatkovne baze v omrežju OSPF

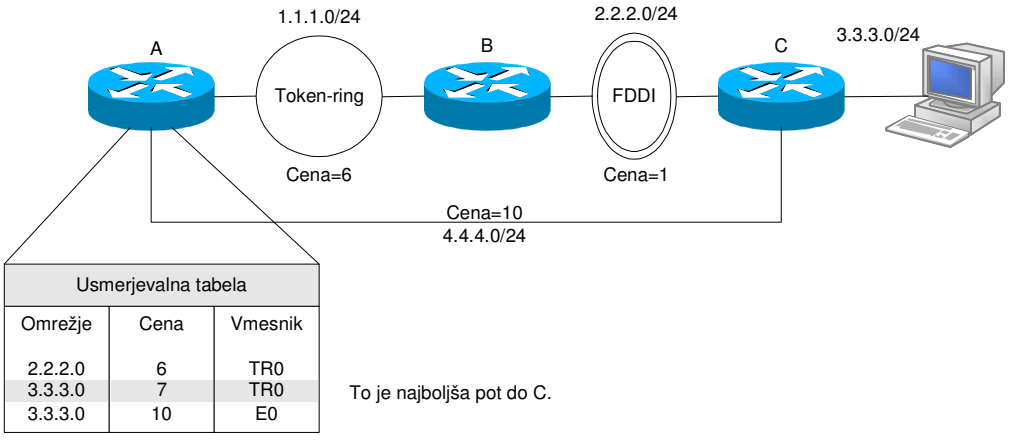
- Vsi usmerjevalniki dodajo nove zapise stanja povezav v podatkovno bazo.
- Ko določeni usmerjevalnik sprejme vse pakete LSR, pravimo, da je sosednji usmerjevalnik sinhroniziran, t.j. v stanju *Full*. Usmerjevalniki morajo biti v stanju Full, preden lahko začnejo usmerjati promet. V tej točki morajo vsi usmerjevalniki imeti identično podatkovno bazo stanja povezav.

Korak 4: Izbiranje primerne poti.

Ko usmerjevalnik napolni podatkovno bazo stanja povezav, je pripravljen za kreiranje usmerjevalne tabele namenjene usmerjanju prometa. Usmerjevalni protokoli na osnovi vektorja razdalj, kot npr. RIP, izbirajo najboljšo pot do ponornega gostitelja glede na metriko števila skokov in za izračun poti z najmanjšim številom skokov uporabljajo Bellman-Fordov algoritem.

Usmerjevalni protokoli na osnovi stanja povezav za izračun najboljše poti do ponornega gostitelja uporabljajo metriko cene, ki temelji na pasovni širini prenosnega medija. Ethernet 10-Mbps omrežje ima npr. nižjo ceno od 56 Kbps serijske linije, saj je 10-Mbps hitrejša povezava od 56 Kbps. Usmerjevalni protokoli na osnovi stanja povezav, kot npr. OSPF, za izračun najnižje cene do ponornega gostitelja uporabljajo Dijkstrov (1959) algoritem, ki skupni ceni poti med lokalnim usmerjevalnikom in ponornim omrežjem dodaja ceno poti od gostitelja do usmerjevalnika. Če do ponornega gostitelja vodi več poti, izberemo pot z najmanjšo ceno.

Dijkstrov algoritem (imenovan po Edsgar Dijkstri) zamenja celotno sliko omrežja z drevesno strukturo, kjer koren drevesa predstavlja lokalni usmerjevalnik. Drevesno strukturo določi algoritem iskanja najmanjšega vpetega drevesa, ki poveže vsa vozlišča omrežja (usmerjevalnike) tako, da ta ne tvorijo ciklov. Algoritem za iskanje najkrajše poti potuje po vpetem drevesu od korena do ponora in ugotavlja skupno ceno poti (slika 40).



Slika 40: Prikaz Dijkstrovega algoritma na omrežju OSPF

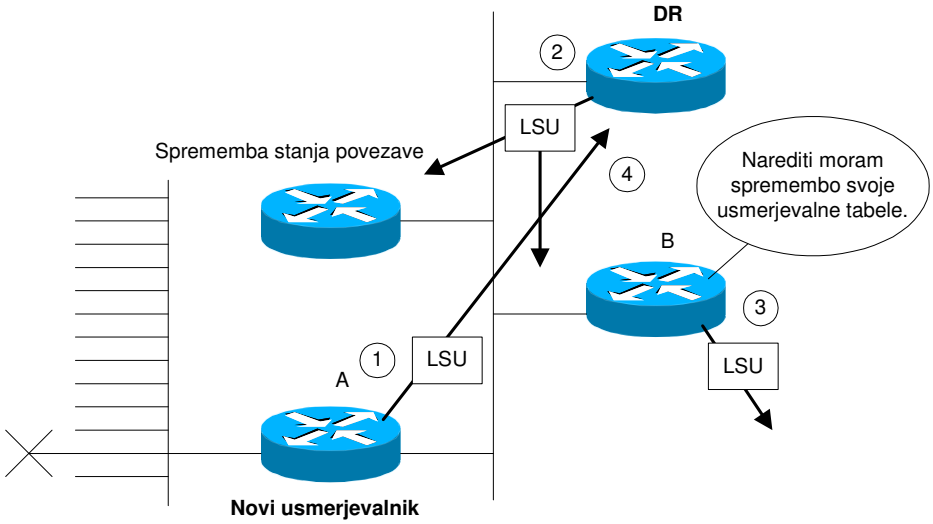
Povezava se lahko včasih, npr. pri serijski liniji, zelo hitro aktivira in deaktivira (ta pojav imenujemo angleško tudi *flapping*), kar lahko povzroči nenadno spremembo stanja povezave, ki vpliva na druge povezave. V tem primeru usmerjevalnik generira zaporedje

paketov LSU, ki povzroči novo preračunavanje usmerjevalne tabele. Proces *flapping* povzroči, da usmerjevalniki nikoli ne izmenjajo vseh podatkov med podatkovnimi bazami, t.j. ne konvergirajo. Da bi zmanjšali ta problem, vsakokrat, ko usmerjevalnik sprejme paket LSU, pred preračunavanjem svoje usmerjevalne tabele, počaka določeno obdobje (angl. *holdtime*).

Korak 5: Vzdrževanje usmerjevalnih informacij.

Ko se stanje povezave spremeni, usmerjevalnik uporabi proces *poplave* (angl. flooding), ki o spremembi v omrežju obvesti ostale usmerjevalnike. V splošnem je proces *poplave* naslednji (slika 41):

- Usmerjevalnik opazi spremembo stanja povezave in na naslov *multicast* 224.0.0.6, t.j. »vsi OSPF DR«, pošlje paket LSU.
- DR potrdi sprejem paketa LSU in ga pošlje drugim usmerjevalnikom v omrežju z uporabo naslova *multicast* 224.0.0.5 protokola OSPF. Po sprejemu paketa LSU vsak usmerjevalnik odgovori DR s potrditvenim paketom LSAck.



Slika 41: DR pošlje novo stanje povezave na multicast naslov 224.0.0.5

- Če je usmerjevalnik povezan na drugo omrežje, pošlje paket LSU drugim omrežjem tako, da ga posreduje DR v drugem več-dostopnem omrežju. Ta DR na naslov *multicast* pošlje paket LSU drugim usmerjevalnikom v omrežju.
- Ko usmerjevalnik sprejme paket LSU, ki vključuje spremembe, posodobi svojo podatkovno bazo stanja povezav. Algoritem SPF potem preračuna novo drevo najkrajših povezav, iz katerega generiramo novo usmerjevalno tabelo. Po kratki zakasnitvi proces usmerjanja poteka z novo usmerjevalno tabelo.

Vsak paket LSA ima svoj lasten časovnik staranja. Privzeta vrednost tega je 30 sekund. Ko paket LSA zastara, originalni usmerjevalnik v omrežje pošlje paket LSU, da bi

preveril, ali je povezava še vedno aktivna. Ta metoda preverjanja prihrani pasovno širino prenosnega medija v primerjavi z usmerjevalniki, ki usmerjajo na osnovi vektorja dolžin in po omrežju razpršijo svojo celotno usmerjevalno tabelo.

2.4 Zunanji usmerjevalni protokoli

Zunanje usmerjevalne protokole (angl. Exterior Gateway Protocol, krajše EGP) uporabljamo za izmenjavo usmerjevalnih informacij med usmerjevalniki v različnih avtonomnih sistemih. Danes uporabljamo dva protokola EGP:

- zunanji usmerjevalni protokol (angl. Exterior Gateway Protocol, krajše EGP),
- mejni usmerjevalni protokol (angl. Border Gateway Protocol, krajše BGP).

Značilnosti obeh opisujemo v nadaljevanju.

2.4.1 *Zunanji usmerjevalni protokol*

Zunanji usmerjevalni protokol EGP je protokol, ki ga uporabljamo za izmenjavo usmerjevalnih informacij med zunanjimi usmerjevalniki, t.j. usmerjevalniki, ki ne pripadajo istemu avtonomnemu sistemu. EGP zahteva enotno hrbtenico in s tem eno samo pot med dvema avtonomnima sistemoma. Praktično je EGP danes omejen na podjetja, ki želijo zgraditi zasebno spletno omrežje. EGP hitro nadomešča BGP.

EGP temelji na periodičnem vabljenju sosedov s pomočjo izmenjave paketov *Hello I Hear You*, s katerimi nadzoruje njihovo dosegljivost in sprejema njihove odgovore. EGP omejuje zunanje usmerjevalnike tako, da jim dovoljuje objavljati samo tista ponorna omrežja, ki so v celoti dosegljiva znotraj opazovanega avtonomnega sistema. Zunanji usmerjevalnik, ki uporablja EGP, sporoča informacije svojim sosedom EGP, ne pa tudi svojim sosedom (usmerjevalnika sta soseda, če izmenjujeta usmerjevalne informacije) izven avtonomnega sistema. Usmerjevalne informacije znotraj avtonomnega sistema zbira usmerjevalnik EGP pogosto preko notranjega usmerjevalnega protokola (IGP).

2.4.2 *Mejni usmerjevalni protokol*

Mejni usmerjevalni protokol BGP je zunanji usmerjevalni protokol, ki ga uporabljamo za izmenjavo informacij o dosegljivosti omrežja med avtonomnimi sistemi. BGP se razlikuje od IGP po naslednjih značilnostih:

- BGP temelji na usmerjevalnem protokolu, ki usmerja pakete na osnovi omrežne politike in ne na osnovi tehničnih meril.
- spremembe BGP se dogajajo v segmentih TCP, kar pomeni, da morata sosednja usmerjevalnika, preden lahko izmenjata informacije med seboj, vzpostaviti sejo.

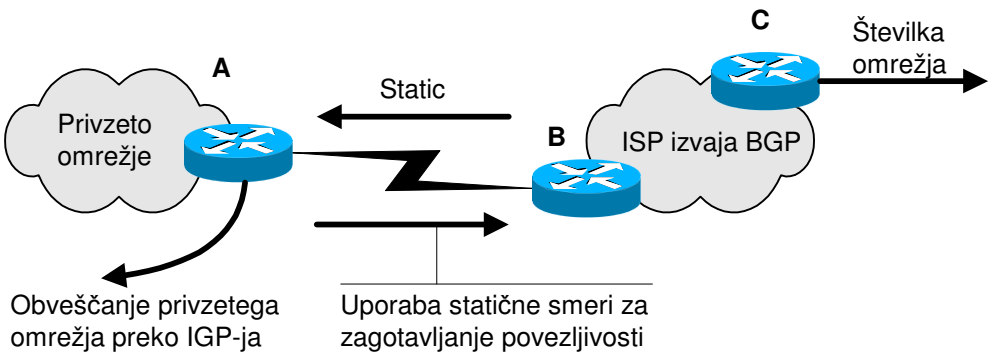
Ti dve ključni razliki pomenita vrh ledene gore vprašanja, zakaj je BGP tako težko konfigurirati. Na srečo najtežji del konfiguriranja nosijo ponudniki spletnih storitev (ISP). To je tudi eden izmed razlogov, zakaj tem ponudnikom plačujemo priključnino za dostop na splet, namreč zato, da nimamo opravka z BGP in vsemi njegovimi zahtevami.

Kdaj ne uporabljati BGP

Omrežna politika, ki jo implementiramo v avtonomnih sistemih, je v večini primerov usklajena s politiko implementirano pri ponudnikih spletnih storitev. V takih primerih ni potrebno, oziroma sploh ni zaželeno, konfigurirati BGP v avtonomnem sistemu. Tehnologija BGP in njegova implementacija je kompleksnejša kot pri IGP, npr. IGRP ali OSPF. Podobno povezljivost pa lahko dosežemo s kombinacijo statičnih poti in privzetega omrežja.

Če se priključujemo na dva različna ponudnika spletnih storitev, pogosto potrebujemo BGP. Redundanca, porazdelitev obremenitve (angl. load sharing) in nižje tarife v določenem času dneva ali noči, so razlogi, zakaj se nekatera podjetja odločajo za dva različna ponudnika. Če imamo redundančno linijo, lahko namesto protokola BGP uporabimo kombinacijo statičnih in privzetih prehodov, če pa želimo, da sta obe povezavi aktivni istočasno, potrebujemo BGP. Vsakokrat, ko se naša omrežna politika razlikuje od politike našega ponudnika spletnih storitev, je BGP nujno potreben.

Usmerjevalnik A na sliki 42 naznanja spremembe v omrežju privzetemu omrežju v avtonomnem sistemu z uporabo določenega notranjega usmerjevalnega protokola, npr. protokola RIP. Statična smer ga povezuje z usmerjevalnikom B in ponudnikovim avtonomnim sistemom. Ponudnik spletnih storitev uporablja protokol BGP, tako da ga ostali usmerjevalniki BGP v spletnem omrežju prepoznajo.

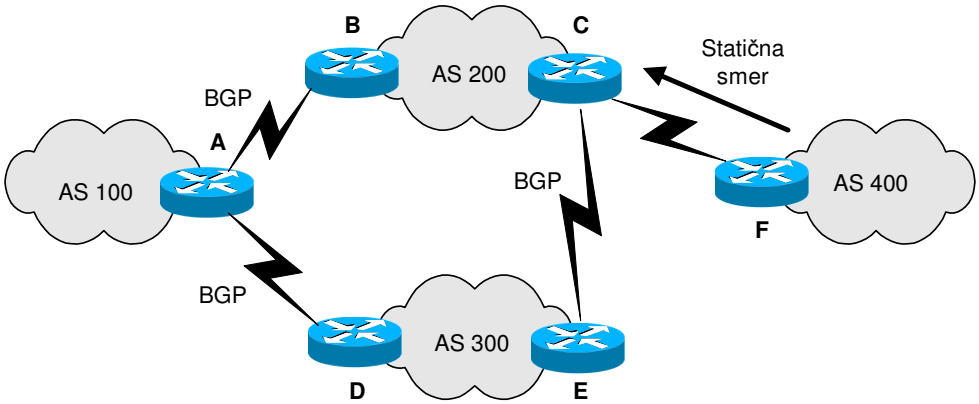


Slika 42: Izogibanje protokolu BGP

Pomni: *BGP za povezavo do ponudnika spletnih storitev uporabljamo samo v primeru, ko se naša omrežna politika razlikuje od ponudnikove.*

Omrežna politika vodi zahteve protokola BGP

Avtonomni sistem 400 na sliki 43 je do usmerjevalnika C povezan s statično potjo. Z vidika usmerjevalnika A imajo omrežja, ki jih povezuje usmerjevalnik F, isto politiko kot omrežja v avtonomnem sistemu 200 ponudnika spletnih storitev.



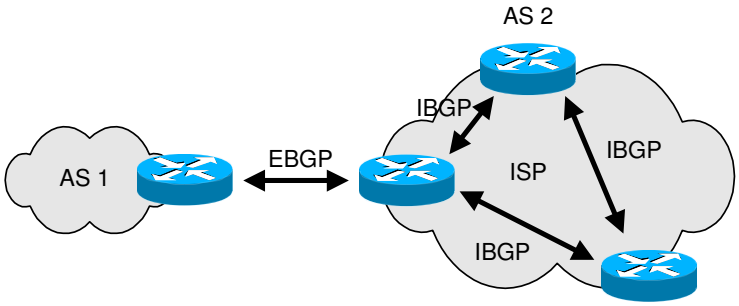
Slika 43: omrežna politika AS 100 do AS 400 vedno uporablja pot preko AS 300

Nove usmerjevalne politike v avtonomnem sistemu AS 400 ne moremo implementirati v AS 100, saj bi ta morala razlikovati AS 400 od drugih avtonomnih sistemov, t.j. če bi AS 100 hotel vedeti za obstoj AS 400, bi AS 400 moral biti povezan na omrežje prek protokola BGP. Za implementacijo te politike pa bi moral tudi usmerjevalnik F podpirati protokol BGP.

Seje BGP

Seje BGP izvedemo s protokolom TCP, t.j. zanesljivim transportnim mehanizmom. BGP podpira naslednji dve vrsti sej med usmerjevalnikom in njegovimi sosedi:

- *Zunanje seje BGP* (EBGP): nastopajo med usmerjevalniki v dveh avtonomnih sistemih. Ti usmerjevalniki so običajno sosednji in si delijo isti medij in podomrežje.



Slika 44: Seje BGP

- *Notranje seje BGP* (IBGP): nastopajo med usmerjevalniki v istem avtonomnem sistemu (slika 44) in jih uporabljamo za koordinacijo in sinhronizacijo omrežne politike med avtonomnimi sistemi. Sosedje se lahko nahajajo kjerkoli v avtonomnem sistemu, celo nekaj skokov daleč eden od drugega.

Notranje seje IBGP nastopajo med usmerjevalniki v istem avtonomnem sistemu pri ponudniku spletnih storitev in usklajujejo omrežno politiko BGP znotraj avtonomnega sistema.

3 VARNOST V TCP/IP

Že pred tisoč leti so ljudje stražili pred vrati zakladnic, v katerih so hranili svoje zaklade in dragocenosti. Če je bilo stražarjenje neuspešno, so roparji spraznili zakladnice, stražarje zakladnic pa je običajno čakalo omalovaževanje ali celo smrt. Čeprav se danes napadi v informacijske sisteme pogosto ne končajo tako dramatično, pa so lahko njihove posledice zelo hude. Vodje informacijskih centrov so spoznali, kako pomembna je zaščita njihovega komunikacijska omrežja pred napadalci in saboterji tako od zunaj kot od znotraj. Vrste napadov na omrežja, ki jih lahko danes najpogosteje pričakujemo, so naslednje:

- *prisluškovanje*: dobiti dostop do podatkov in gesel,
- *lažno predstavljanje*: dobiti neavtoriziran dostop do podatkov, kreiranje neavtorizirane elektronske pošte, ipd.,
- *zavrnitev storitve*: uničevanje omrežnih virov,
- *odgovor na sporočila*: dobiti dostop in spremeniti informacije v prometu,
- *uganiti gesla*: dobiti dostop do informacij in storitev, ki običajno ne bi bile dovoljene,
- *uganiti ključ*: dobiti dostop do kodiranih podatkov in gesel,
- *virusi*: uničiti podatke.

Čeprav ti napadi niso specifični samo za omrežja TCP/IP, predstavljajo potencialno nevarnost, s katero morajo računati vsi, ki gradijo svoja omrežja na protokolu TCP/IP. Istočasno z razvojem zaščitnih mehanizmov pa se razvijajo tudi mehanizmi napadov na omrežja, s katerimi vdiralci iščejo njihove šibke točke.

3.1 Rešitve omrežne varnosti

Z enako vnemo kot napadalci, ki iščejo nove poti, kako napasti računalniško omrežje, morajo lastniki teh omrežij iskati poti, kako se pred napadi zaščititi. Danes obstaja kar nekaj rešitev, ki omogočajo učinkovito zavarovanje pred njimi. Te rešitve pa običajno rešujejo samo enega oziroma samo omejeno število varnostnih problemov. Če želimo zagotoviti določen nivo varnosti in zaščite, potrebujemo celotno zbirko teh rešitev. Te rešitve so naslednje:

1. *šifriranje*: za zaščito podatkov in gesel,
2. *avtentikacija in avtorizacija*: za preprečevanje nedovoljenih dostopov,
3. *preverjanje integritete in sporočila avtentikacijskih kod*: za zaščito pred nepravilno spremembo sporočil,
4. *digitalni podpisi in certifikati*: za potrjevanje partnerjeve identitete,
5. *enkratna gesla in dvosmerna naključna preverjanja* (angl. handshakes): za medsebojno avtentikacijo partnerjev v pogovoru,
6. *skrivanje naslova*: za zaščito proti napadom na zavrnitev storitve.

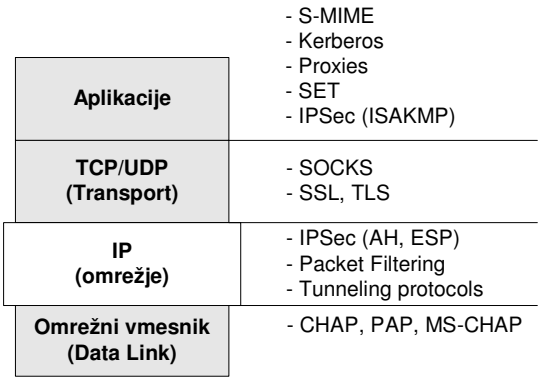
Pomni: *ohrani tesno povezavo z zunanjim svetom, toda bodi pozoren na notranjo mrežo, saj se večina napadov začenja v notranjem omrežju*.

3.1.1 *Implementacije varnostnih rešitev*

Protokoli in sistemi, ki jih lahko uporabljamo pri zagotavljanju določene stopnje varnosti storitev v računalniškem omrežju, so naslednji:

- *filtriranje IP*,
- *omrežno prevajanje naslovov* (angl. Network Address Translation ali NAT),
- *varnostna arhitektura IP* (angl. IP Security Arhitecture ali IPSec),
- *socks*,
- *plast varnih vtičnic* (angl. Secure Socket Layer ali SSL),
- *aplikacijski proksiji* (angl. Application Proxies),
- *požarni zidovi* (angl. Firewalls),
- *Kerberos in drugi avtentikacijski sistemi* (strežniki AAA),
- *varne elektronske transakcije* (angl. Secure Electronic Transactions ali SET).

Slika 45 prikazuje plasti v modelu TCP/IP, na katerih te varnostne rešitve deluje.



Slika 45: Varnostne rešitve v plasteh TCP/IP

Tabela 2 združuje značilnosti nekaterih že omenjenih varnostnih rešitev in jih primerja med seboj.

Tabela 2: implementacija varnostnih rešitev - primerjava

	Nadzor Dostopa	Šifriranje	Avtentikacija	Preverjanje integritete	Skrivanje naslova	Nadzor seje
IP filtriranje	D	N	N	N	N	N
NAT	D	N	N	N	D	D
IPSec	D	D (paket)	D (paket)	D (paket)	D	N
SOCKS	D	N	D (odjem./strež.)	N	D	D
SSL	D	D (podatki)	D (sistem/upor.)	D	N	D
Proxy	D	Običajno N	D (uporabnik)	D	D	D
AAA	D (uporabnik)	N	D (uporabnik)	N	N	N

Posamezna rešitev varuje omrežje pred določenimi napadi, zato lahko celostno varnostno rešitev zagotovimo samo s kombinacijo več rešitev, ki delujejo skupaj, npr. z uporabo požarnega zidu. Določene varnostne zahteve določimo v varnostni politiki.

3.1.2 *Politika omrežne varnosti*

Varnostno politiko celotne organizacije določimo z varnostno analizo in analizo potreb podjetja. Ker se požarni zid nanaša samo na varnost omrežja, za organizacijo nima velike vrednosti, dokler je ne povežemo s celovito varnostno politiko organizacije. Politika omrežne varnosti definira tiste storitve, ki jih eksplicitno dopuščamo ali prepovemo, kako te storitve uporabljamo, in za koga ta pravila ne veljajo. Vsako pravilo v politiki omrežne varnosti moramo implementirati v požarnem zidu in strežniku za oddaljeni dostop (angl. Remote Access Server ali RAS). Požarni zid v splošnem uporablja eno izmed naslednjih politik:

Vse, kar ni eksplicitno dopuščeno, je prepovedano

Ta pristop blokira ves promet med dvema omrežjema, razen prometa za tiste storitve in aplikacije, ki ga dovolimo. Vsako storitev ali aplikacijo, ki bi lahko pomenila potencialno luknjo v požarnem zidu, moramo prepovedati. To je najvarnejša zaščitna metoda, čeprav je s stališča uporabnika lahko preveč restriktivna in zato manj primerna.

Vse, kar ni eksplicitno prepovedano, je dopuščeno

Ta pristop omogoča ves promet med dvema omrežjema, razen prometa za tiste storitve in aplikacije, ki jih prepovemo. Vsako nevarno ali potencialno škodljivo storitev ali aplikacijo moramo prepovedati. Čeprav je to fleksibilna in primerna metoda za uporabnike, potencialno lahko povzroča nekatere resne varnostne probleme.

Strežnik ta oddaljeni dostop omogoča avtentikacijo (kdo je ta) in v idealnem primeru tudi avtorizacijo uporabnikov, t.j. določa, kaj lahko kdo dela znotraj skupnega intranet omrežja. Prav tako mora določati ali se uporabnik lahko prijavi iz različnih oddaljenih lokacij (angl. roaming) in če lahko strežnik za določene uporabnike, ki so se pravilno avtenticirali, uporabi povratni klic (angl. dial-back).

3.2 Kriptografija

Začnimo z definicijo nekaterih osnovnih konceptov.

Kriptografija

Kriptografija je znanost, ki se ukvarja z varovanjem naših podatkov in komunikacijskih poti. Za dosego tega cilja uporabljamo mehanizme kot *šifriranje*, *dešifriranje* in *avtentikacija*. Vsako proceduro, ki transformira podatke po

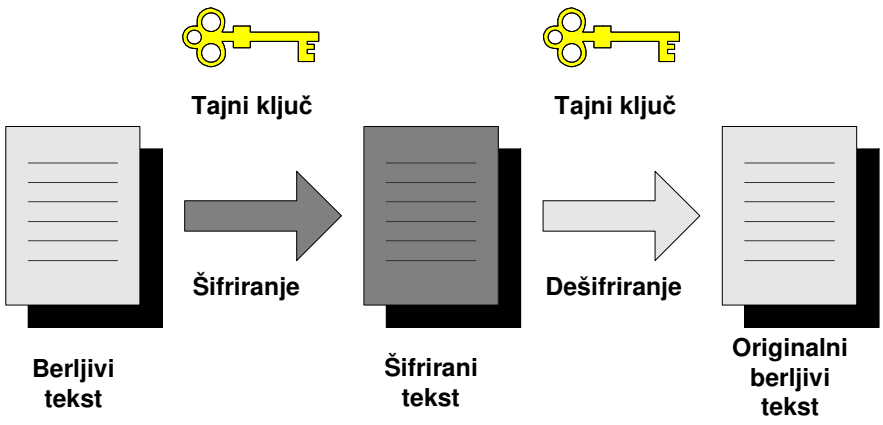
določenih metodah tako, da jih je težko povrniti v prvotno stanje, imenujemo *kriptografsko*. Ključni faktor močne kriptografije merimo s težavnostjo povratnega inženiringa, t.j. kako težko pridemo do originalnega stanja podatkov. Napad na npr. zaščiten Wordov dokument zahteva od izkušenega napadalca na običajnem osebnem računalniku nekaj minut dela. Močna kriptografija pomeni, da v primeru, ko ne vemo, kako je bilo sporočilo kodirano, sporočila ne moremo dekodirati v realnem času. Proces povratnega inženiringa imenujemo tudi *kriptoanaliza*.

Šifriranje in dešifriranje – kriptografski algoritmi

Šifriranje je transformacija berljivega teksta v neberljivo obliko, da bi skrili njegov pomen. Obratno transformacijo, ki iz neberljive oblike naredi berljivi tekst, imenujemo *dešifriranje*. Matematične funkcije, ki jih uporabljamo za šifriranje in dešifriranje, imenujemo *kriptografski algoritmi*. Kriptografski algoritmi so varni samo tako dolgo, dokler je njihovo delovanje tajno, zato jih imenujemo tudi omejeni kriptografski algoritmi, in imajo več slabosti. Ker jih uporablja veliko ljudi, je njihovo delovanje zelo težko ohraniti v tajnosti. Če so vključeni v komercialne produkte, je njihovo razkrinkanje samo vprašanje časa in denarja. Zaradi tega današnji algoritmi uporabljajo v procesu šifriranja in dešifriranja parametre, ki jih imenujemo *ključi*. Ključ lahko izberemo iz množice dopustnih vrednosti imenovane *prostor ključev*. Prostor ključev je običajno ogromen, oz. čim večji je tem teže razbijemo ključ. Varnost teh algoritmov temelji izključno na ključih in ne na notranjih algoritmih. V bistvu je algoritem kot tak javen in vseh njegovih šibkosti še ne poznamo.

Pomni: *splošno pravilo varnosti je, bodi konservativen. Ne zaupaj novim, neznanim ali nepreizkušenim algoritmom.*

Princip kriptografskih algoritmov na osnovi ključev prikazuje slika 46.



Slika 46: Kriptiranje na osnovi ključev

Avtentikacija, integriteta in lažno zanikanje

Šifriranje zagotavlja našim sporočilom verodostojnost. Ko komuniciramo prek nevarnega medija, kot je spletno omrežje, poleg verodostojnosti potrebujemo tudi:

- *Avtentikacijo*: metoda preveri, da je pošiljatelj sporočila v resnici tisti, ki ga pričakujemo. Avtentikacija odkrije vsakega napadalca, ki se skriva pod masko nekoga drugega.
- *Preverjenje integritete*: metoda preveri, da se sporočilo med komunikacijsko potjo ni spremenilo. Preverjenje integritete odkrije vse nepooblaščene posege v sporočila. Kot stranski efekt te je možnost odkrivanja komunikacijskih napak.
- *Lažno zanikanje*: metoda dokaže, da je pošiljatelj sporočilo resnično poslal. Pri uporabi tega algoritma pošiljatelj ne more zanikati, da tega sporočila ni poslal.

3.2.1 *Simetrični algoritmi ali algoritmi tajnih ključev*

Simetrični algoritem je kriptografski algoritem na osnovi ključev, kjer je dešifrirni ključ enak šifrirnemu. To je standardni kriptografski algoritem, kjer morata pred začetkom komunikacije pošiljatelj in sprejemnik izmenjati ključ (slika 46). Obstajata dva tipa simetričnih algoritmov: *blokovni*, ki operirajo na berljivem tekstu v bitnih blokih in *znakovni*, ki istočasno operirajo na posameznem bajtu berljivega teksta.

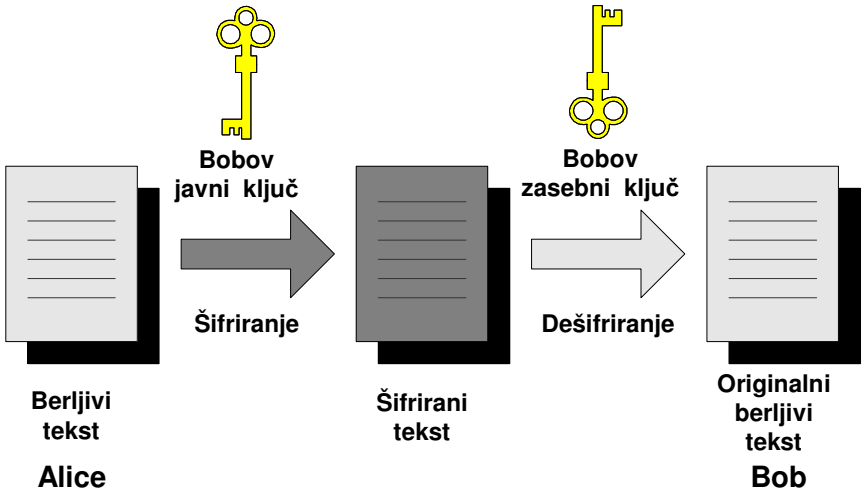
Najbolj znan blokovni kriptografski algoritem je standard za šifriranje podatkov (angl. Data Encription Standard, krajše DES), ki ga je razvil IBM. DES operira na 64-bitnih blokih in uporablja ključ dolžine 56 bitov, običajno izražen kot 64-bitno število. Vsak osmi bit v bajtu uporablja kot paritetni bit. Iz tega ključa algoritem izpelje 16 podključev, ki jih uporabi v 16 ciklih teka algoritma. DES tvori šifrirani tekst z dolžino, ki je enaka dolžini originalnega teksta. Dešifrirni algoritem deluje podobno kot šifrirni, vendar uporablja obratni vrstni red podključev. Zaradi tega je zelo primeren za implementacijo v strojni opremi. Kljub starosti (njegovi začetki segajo v zgodnja sedemdeseta leta) pa DES ostaja še vedno vodilni kritografski algoritem. Glavna odlika je bila njegova dolžina ključa, vendar bi ga lahko danes, z dovolj časa in denarja, zlahka razbili. Zato se je pojavila nova različica tega algoritma, t.j. 3DES (angl. triple-DES), pri katerem se originalni algoritem DES uporabi v treh ciklih z dvema ali tremi različnimi ključi.

Prednost simetričnih algoritmov je njihova učinkovitost. Enostavno jih lahko implementiramo v strojni opremi. Glavno njihovo slabost pa predstavlja težavno vodenje ključev, saj moramo najti varen način izmenjave ključev. To pa je v praksi ponavadi zelo težko izvesti.

3.2.2 *Asimetrični algoritmi ali algoritmi javnih ključev*

Glavna slabost simetričnih algoritmov predstavlja zahteva po varni izmenjavi ključev. Ta problem rešujejo t.i. asimetrični algoritmi. Pri asimetričnih algoritmih uporabljamo dva različna ključa: javni ključ, ki ga pozna vsak in zasebni ključ, ki ga uporabnik drži v strogi tajnosti. Zasebni ključ ne moremo določiti iz javnega ključa. Berljivi tekst, ki smo

ga šifrirali z javnim ključem lahko dešifriramo samo z ustreznim zasebnim ključem in obratno (slika 47).

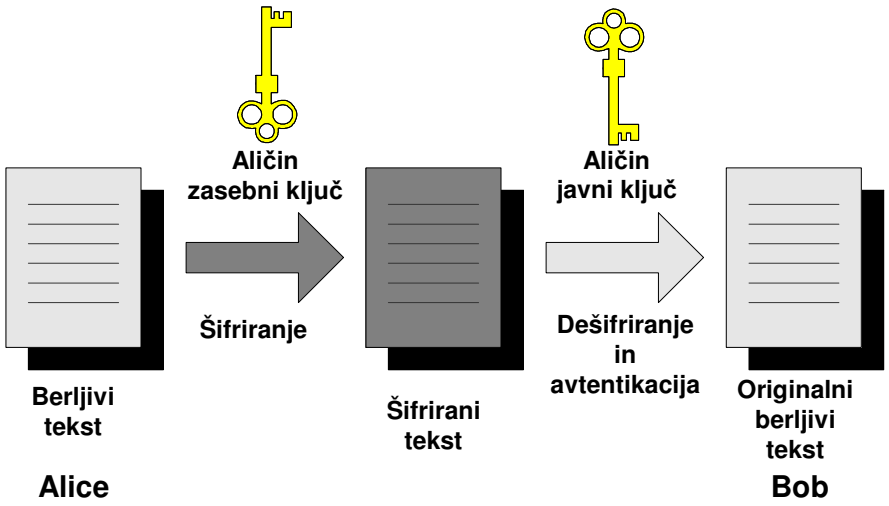


Slika 47: Šifriranje z uporabo prejemnikovega javnega ključa

Ker je javni ključ na voljo komurkoli, zagotavljamo zasebnost, ne da bi za to potrebovali dodaten kanal za izmenjavo ključev.

Avtentikacija in lažno zanikanje

Pomembna lastnost kriptografskih algoritmov na osnovi javnih ključev je, da omogočajo *avtentikacijo*. Zasebni ključ uporabljajo za šifriranje. Ker do ustreznega javnega ključa lahko dostopa in sporočila dešifrira vsak, ta ne omogoča zaščite. Seveda pa lahko sporočilo avtenticira. Če lahko nekdo z zahtevanim pošiljateljevim javnim ključem uspešno dešifrira to sporočilo, ga šifriramo z ustreznim zasebnim ključem, ki je znan samo realnemu pošiljatelju. S tem preverimo pošiljateljevo identiteto. Šifriranje z zasebnim ključem uporabljamo v *digitalnih podpisih* (slika 48).



Slika 48: Avtentikacija pri šifriranju s zasebnim ključem

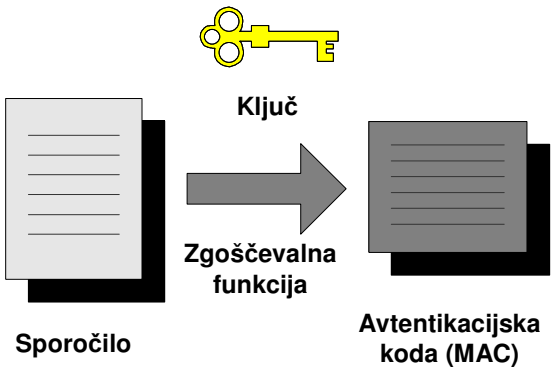
Iz slike 48 razberemo, da Alice šifrira sporočilo s svojim zasebnim ključem, da bi omogočila Bobu preveriti avtentikacijo sporočila. Če gremo še korak naprej, lahko ugotovimo, da šifriranje z zasebnim ključem daje tudi zaščito *lažnega zanikanja*. Samo slučajno bi se lahko zgodilo, da bi lahko tako šifrirano sporočilo poslal kdo drug, kot oseba z zasebnim ključem.

Tudi algoritma *RSA* in *Diffie-Hellmanova izmenjava ključev* temeljita na javnih ključih.

3.2.3 *Zgoščevalna funkcija*

Zgoščevalna funkcija (angl. Hash) predstavlja osnovo kriptografije. Vhodni podatki zgoščevalne funkcije so podatki variabilne dolžine, ki jih ta spremeni v podatek fiksne dolžine. Ta podatek lahko obravnavamo kot sled, oz. prstni odtis, vhodih podatkov. Če se sledi dveh sporočil ujemata, lahko z veliko gotovostjo trdimo, da sta enaki. Kriptografsko uporabna zgoščevalna funkcija mora biti *enosmerna*, t.j. mora se enostavno izračunati in težko dekodirati. Vsakodnevni primer enosmerne zgoščevalne funkcije je mečkanje krompirjev za pire - enostavno ga je narediti, iz pireja narediti originalni krompir pa je nemogoče.

Primer: zgoščevalna funkcija iz vhodnih podatkov in ključa zmeša kodirano sporočilo, ki predstavlja t.i. avtentikacijsko kodo (angl. Message Authentication Code, krajše MAC) (slika 49).

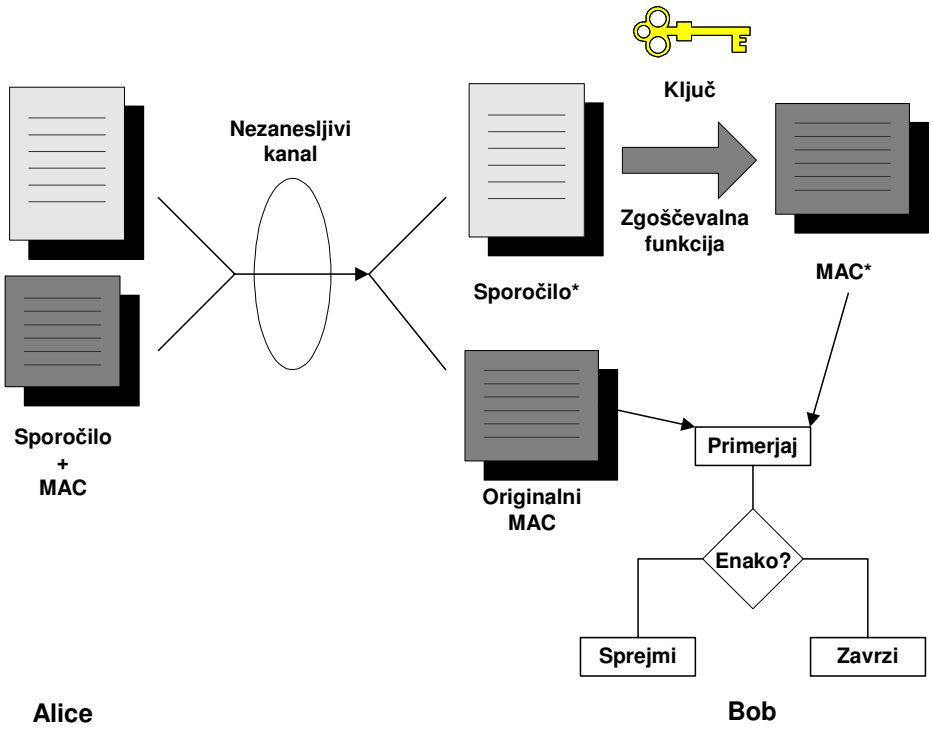


Slika 49: generiranje avtentikacijske kode MAC

V prvi vrsti uporabljamo zgoščevalno funkcijo v metodah preverjanja integritete in pri avtentikaciji. Poglejmo, kako lahko z njo implementiramo preverjanje integritete in avtentikacijo (slika 50):

- pošiljatelj z zgoščevalno funkcijo izračuna vrednost avtentikacijske kode MAC-1 in jo pripne v sporočilo,
- sprejemnik z zgoščevalno funkcijo izračuna vrednost avtentikacijske kode sprejetega sporočila MAC-2 in rezultate primerja s sprejeto vrednostjo zgoščevalne funkcije MAC-1,

- če se vrednosti zgoščevalnih funkcij MAC-1 in MAC-2 ujemata, sporočilo ni bilo napadeno,
- v primeru avtentikacijske kode MAC, kjer je kriptografski ključ lahko uporabil samo zanesljivi pošiljatelj, uspešno dešifriranje sporočila dokazuje, da sta zahtevani in dejanski pošiljatelj identična.



Slika 50: Preverjanje integritete in avtentikacija z avtentikacijsko kodo MAC

Opazimo lahko, da podoben rezultat, kot na sliki 50, lahko dobimo tudi z različnimi vrstami šifriranja. Če napadalec npr. spremeni šifrirano sporočilo, postane rezultat dešifriranja nesmiseln in s tem spremembo sporočila zlahka odkrijemo. Velikokrat pa ne potrebujemo avtentikacijo s šifriranjem celotnega sporočila, ampak samo določenih polj.

Šifriranje z zgoščevalno funkcijo in zasebnim ključem imenujemo *digitalni podpis* in ga lahko obravnavamo kot posebno vrsto avtentikacijske kode MAC. Uporaba digitalnega podpisa, namesto šifriranja celotnega sporočila z zasebnim ključem, vodi do znatnega povečanja učinkovitosti in novih lastnosti postopka šifriranja. Avtentikacijski del dokumenta lahko izdvojimo iz samega dokumenta. To lastnost npr. uporabljamo v protokolu varnih elektronskih transakcij SET.

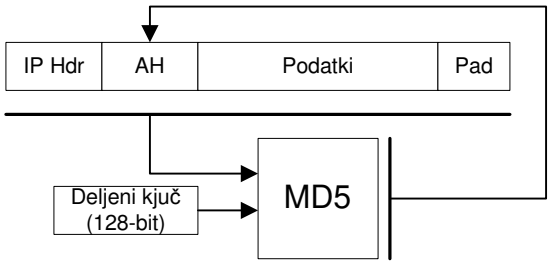
Primeri zgoščevalnih funkcij

Najpogosteje uporabljamo zgoščevalni funkciji *izvleček sporočila* (angl. Message-Digest Algorithm 5, krajše MD5) in *varni zgoščevalni algoritem* (angl. Secure Hash Algorithm 1, krajše SHA-1). MD5 je razvil Ron Rivest, SHA-1 pa National Institute of Standards

and Tehnology (NIST) in National Security Agency (NSA) za uporabo z Digital Signature Standard (DSS). MD5 generira 128-bitno, SHA-1 pa 160-bitno vrednost zgoščevalne funkcije. Obe funkciji v vrednosti zgoščevalne funkcije MAC kodirata tudi dolžino sporočila. SHA-1 velja za zanesljivejšega zaradi večje dolžine vrednosti zgoščevalne funkcije, ki jo kodira.

MD5 in SHA-1 s ključem

Princip delovanja zgoščevalnih funkcij MD5 in SHA-1 s ključem je naslednji. Vhodna parametra v zgoščevalno funkcijo MD5 sta tajni ključ in datagram, ki ga želimo zaščititi. Rezultat, t.j. vrednost zgoščevalne funkcije MAC, shranimo v polje avtentikacijski podatki (angl. Authentication Data) avtentikacijske glave (angl. Authentication Header, krajše AH) (slika 51).

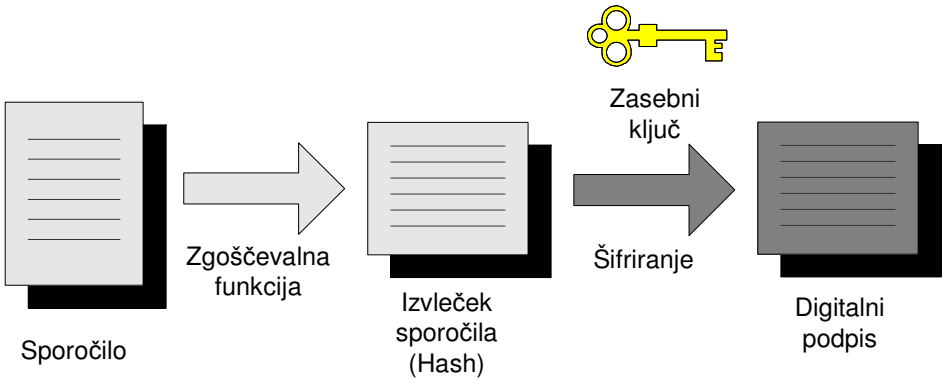


Slika 51: Procesiranje MD5 s ključem

Zgoščevalna funkcija SHA-1 s ključem deluje na podoben način kot MD5, razlika je samo v večji 160-bitni vrednosti zgoščevalne funkcije MAC.

Digitalni podpis

Vrednost zgoščevalne funkcije MAC, kriptirane z zasebnim ključem, imenujemo digitalni podpis (slika 52).



Slika 52: generiranje digitalnega podpisa

Avtentikacijska metoda, ki jo uporabljamo pri protokolu IKE (ISAKMP/Oakley), je digitalni podpis DSS. Tega sta organizaciji NIST in NSA izbrala kot digitalni avtentikacijski standard ameriške vlade. Ta standard opisuje algoritem digitalnega podpisa (angl. Digital Signature Algorithm, krajše DSA), ki ga uporablja za podpis in preverjanje digitalnega podpisa z izvlečkom sporočila narejenega z metodo SHA-1.

Digitalni certifikati in avtorizacija s certifikati

Pri avtentikaciji z javnimi ključi obe stranki, ki nastopata v komunikaciji, zamenjata svoja javna ključa. Ta izmenjava pa je lahko nevarna. Napadalec na omrežje bi lahko zamenjal nek javni ključ s svojim javnim ključem in potem namestil t.i. napad *vmesni človek* (angl. man-in-the-middle).

Primer: napadalec se vrine v komunikacijo med Alice in Bobom. Boba lahko prevara tako, da mu v imenu Alice pošlje enega izmed svojih javnih ključev. Podobno se zgodi pri komunikaciji z Alice, ki je prepričana, da uporablja Bobov javni ključ, v resnici pa komunicira z napadalcem. Na ta način lahko pameten napadalec dešifrira zaupni promet med obema in ostane v tajnosti.

Rešitev takih groženj je *digitalni certifikat*. Digitalni certifikat je datoteka, ki povezuje identiteto s pridruženim javnim ključem. To povezovanje preverja zaupni tretji partner, ki certifikate avtorizira (angl. Certification Authority, krajše CA). Digitalni certifikat CA avtorizira s svojim zasebnim ključem. Te certifikate lahko tudi avtenticiramo. Za razliko od javnih ključev digitalni certifikat vsebuje tudi druge informacije, kot so:

- datum izdaje certifikata,
- datum poteka certifikata,
- različne informacije, ki se tičejo izdajo certifikatov (npr. serijska številka).

Slika je sedaj povsem drugačna. Stranki zamenjata svoje digitalne certifikate in jih avtenticirata z uporabo javnih ključev pri partnerju, ki je certifikat izdal. Ta zagotavlja, da javni ključ resnično obstaja.

Vendar tudi digitalni certifikat ne pokriva vseh zahtev.

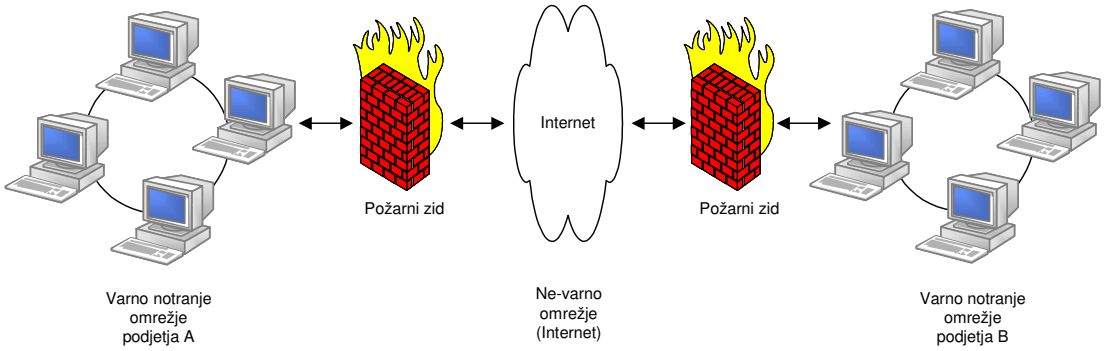
Primer: kaj se zgodi, če CA preverja Bobov certifikat za Alice in ga ne pozna? Ali lahko zaupa neznani stranki? Da bi postopek poenostavili, lahko CA oblikuje avtorizacijsko hierarhijo, ki jo imenujemo tudi *veriga zaupanja*. Vsak član verige ima certifikat, ki ga je avtoriziral nadrejeni partner. Višji kot je CA v verigi, močnejše varnostne procedure uporablja. Korenskemu certifikatu CA zaupa vsak in njegov zasebni ključ je strogo varovan.

Alice lahko potuje po verigi zaupanja navzgor dokler ne najde partnerja CA, ki Bobu zaupa. Ko ga najde, je ta prepričana, da je Bobov certifikat verodostojen. Implementacijo tega koncepta lahko najdemo v protokolu SET, kjer glavne znamke kreditnih kartic operirajo z lastno hierarhijo CA. Naslednji primer je avtentikacija produkta za pošiljanje

elektronske pošte Lotus Notes, ki prav tako temelji na certifikatih in lahko implementira lastno hierarhijo verig zaupanja.

3.3 Požarni zid

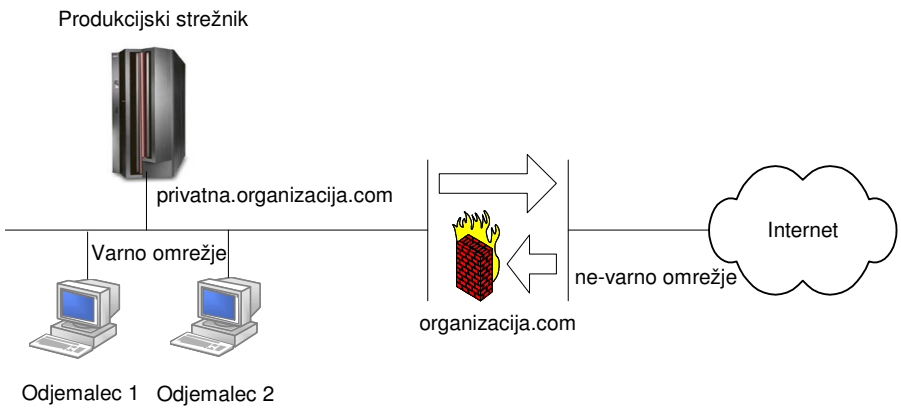
Požarni zid (angl. Firewall, krajše FW) je sistem, ki predpisuje varnostno politiko med varnim notranjim omrežjem in ne-varnim zunanjim omrežjem kot je npr. splet. Požarni zid predstavlja zaščito med sin zasebnim omrežjem. Globalno pa deli svet v dve ali več omrežij: eno ali več varnih (angl. secure) in eno ali več ne-varnih (angl. non-secure) (slika 53).



Slika 53: Ilustracija požarnega zidu

Požarni zid je lahko osebni računalnik, usmerjevalnik, centralni računalnik (angl. mainframe), delovna postaja UNIX ali kombinacija teh in določa, katere informacije oz. storitve so dostopne od zunaj ter komu dovoljujemo njihovo uporabo. Običajno požarni zid namestimo v točki, kjer se srečata varno notranje omrežje in ne-varno zunanje omrežje. To točko imenujemo tudi *točko dušenja* (angl. *choke* point). Da bi laže razumeli, kako požarni zid deluje, si zamislimo omrežje kot zgradbo, ki jo nadzorujemo. Zgradba ima kot edino vstopno točko vežo, v kateri receptorji pozdravljajo obiskovalce, stražarji jih nadzorujejo, video kamere beležijo njihovo gibanje in obiskovalce, ki vstopijo v zgradbo, receptorji označijo z značkami. Čeprav te akcije lahko dobro nadzorujejo dostop do zgradbe, pa v primeru, da se neavtorizirani osebi posreči vstop v zgradbo, ni načina, kako zgradbo zaščititi pred akcijami napadalca. Če njegovo premikanje nadzorujemo s kamero, lahko zaznamo njegove sumljive akcije.

Podobno, kot v opisanem primeru, tudi požarni zid z nadzorom dostopa med varnim notranjim in ne-varnim zunanjim omrežjem ščiti informacijske vire (slika 54). Tukaj je pomembno pripomniti naslednje: čeprav je požarni zid konstruiran tako, da dovoljuje prehod varnih podatkov preko omrežja, preprečuje ranljive storitve in ščiti notranje omrežje pred napadi od zunaj, ga lahko napadi, na katere ni pripravljen, npr. novi virusi, zlahka prebijejo. Da se lahko omrežni administrator uspešno pripravi na obrambo pred napadi od zunaj, mora preverjati vse loge in alarme, ki jih požarni zid generira,.



Slika 54: požarni zid nadzoruje promet med varnim omrežjem in Internetom

3.3.1 *Komponente požarnega zidu*

Požarni zid sestoji iz naslednjih komponent:

- *usmerjevalnikom s filtriranjem paketov* (angl. Packet-filtering router),
- *prehoda proksi* (angl. Proxy gateway),
- *prehoda na nivoju voda* (angl. circuit level gateway).

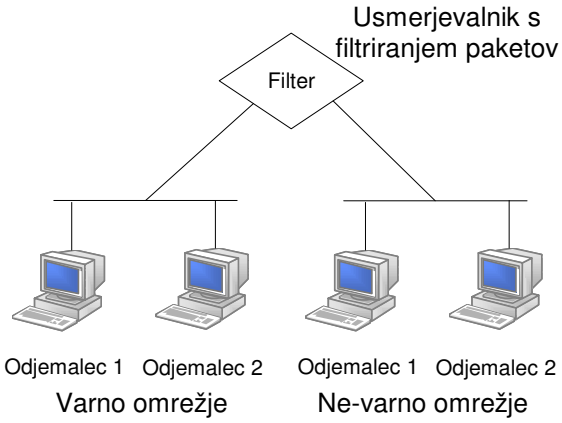
Vsaka od komponent ima svoje prednosti in slabosti. Za učinkoviti požarni zid morajo vse tri komponente delovati skupaj.

3.3.2 *Usmerjevalnik s filtriranjem paketov*

Filtriranje paketov izvajamo z usmerjevalnikom, ki zna posredovati pakete glede na filtrirna pravila. Ko paket prispe do usmerjevalnika s filtriranjem paketov, ta iz glave paketa potegne določene informacije in se glede na filtrirna pravila odloči ali ga bo posredoval naslovnika ali uničil (slika 55). Iz glave paketa lahko usmerjevalnik potegne naslednje podatke:

- izvorni naslov IP,
- ponorni naslov IP,
- izvorna vrata TCP/UDP,
- ponorna vrata TCP/UDP,
- tip sporočila ICMP.

Filtrirna pravila temeljijo na omrežni varnostni politiki. Paketno filtriranje izvajamo z uporabo teh pravil kot vhodno informacijo. Pri določanju filtrirnih pravil moramo upoštevati zunanje napadalce, omejitve na nivoju storitve in omejitve na nivoju izvora/ponora. V nadaljevanju obravnavamo omenjene vrste filtriranja podrobneje.



Slika 55: Usmerjevalnik s filtriranjem paketov

Filtriranje na nivoju storitve (angl. Service Level Filtering)

Ker večina storitev uporablja splošno znane številke vrat TCP/UDP, je mogoče dovoliti ali prepovedati storitve z uporabo ustreznih informacij vrat v filtru.

Primer: strežnik FTP posluša zahteve za storitev po protokolu TCP na vratih 20 in 21. Če želimo dovoliti povezavam FTP pot skozi varno omrežje, mora usmerjevalnik omogočiti paketom, ki vsebujejo v glavi segmenta TCP vrata 20 in 21 prosto pot.

Filtriranja na nivoju izvora/ponora (angl. Source/Destination Level Filtering)

Pravila filtriranja paketov usmerjevalnikom dovoljujejo ali prepovedujejo posredovanje paketov z ozirom na izvorne ali ponorne informacije v glavi datagrama. Če je v varnem omrežju na voljo neka storitev, običajno dovoljujemo zunanjim uporabnikom dostop samo na ta strežnik, t.j. vse pakete, ki imajo naslov IP različen od naslova IP tega strežnika zavržemo.

Napredno filtriranje:

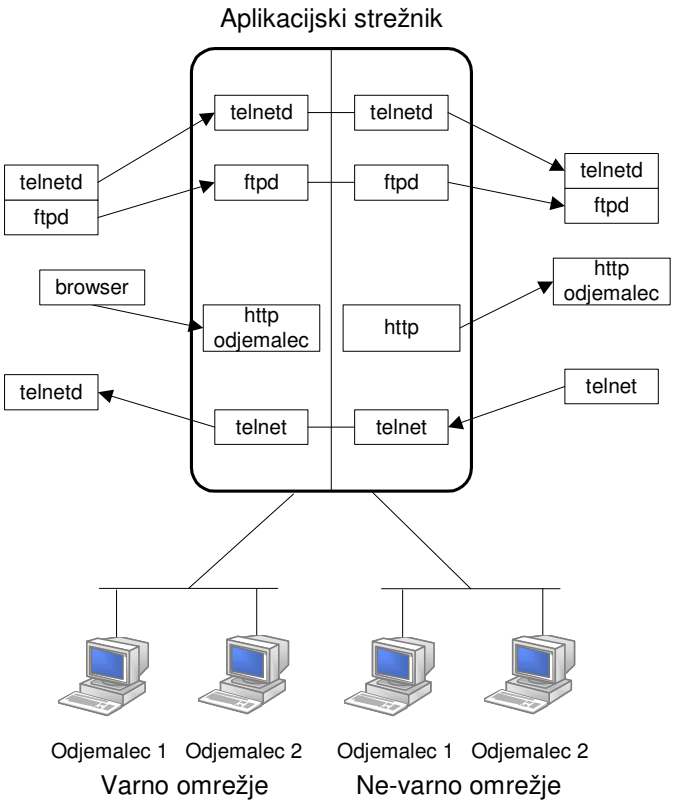
Obstaja več vrst napadov, ki ogrožajo zasebnost in varnost v omrežju. Nekaterim izmed njih se lahko izognemo z uporabo pravil naprednega filtriranja kot npr. preverjanje možnosti IP, polja fragment offset, ipd.

Omejitve filtriranja paketov

Pravila filtriranja paketov so pogosto zelo kompleksna, posebno še v primeru, da obstajajo izjeme v obstoječih pravilih. Kljub testnim orodjem pa še vedno obstajajo luknje v omrežni varnosti. Filtriranje paketov ne predstavlja absolutne varnost omrežja. V nekaterih primerih želimo omejiti samo neko množico informacij, npr. določen ukaz. Ker podatkov s filtriranjem paketov ne moremo nadzorovati, saj vsebine paketov ne poznamo, potrebujemo nadzor na aplikacijskem nivoju.

3.3.3 *Prehod proksi*

Aplikacijski strežnik na požarnem zidu pogosto imenujemo tudi *proksi*. Proksi omogoča višje nivojski nadzor nad prometom med dvema omrežjema tako, da vsebine določene storitve nadzorujemo in filtriramo glede na našo varnostno politiko. Če želimo storitev nadzorovati, moramo na strežnik namestiti ustrezno aplikacijsko kodo proksi (slika 56).



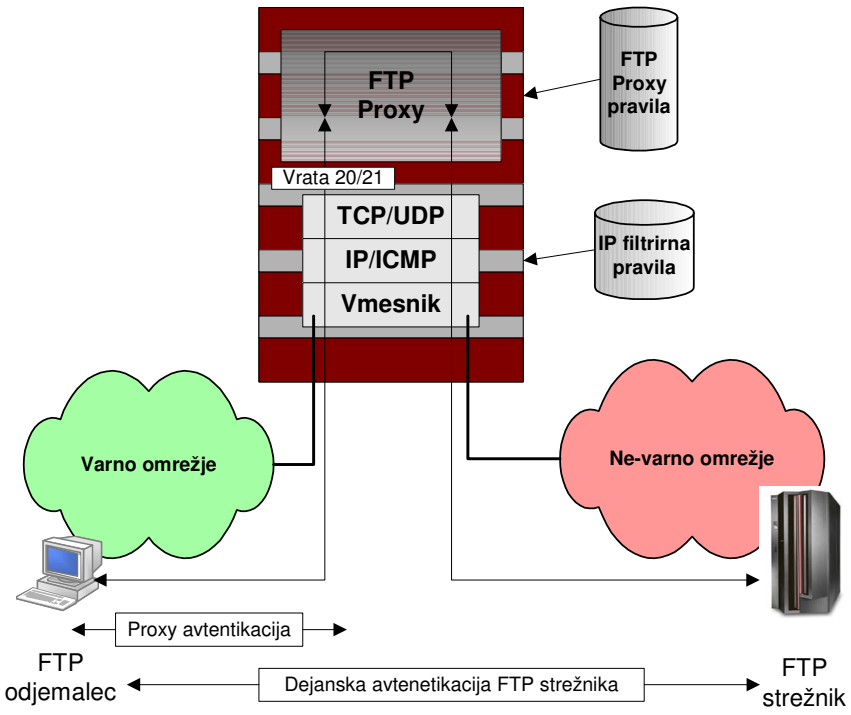
Slika 56: Aplikacijski strežnik proksi

Proksi deluje kot strežnik odjemalcu in kot odjemalec ponornemu strežniku. Vzpostavimo navidezno povezavo med odjemalcem in ponornim strežnikom. Čeprav je na prvi pogled proksi transparenten z vidika odjemalca in strežnika, je ta sposoben nadzora in filtriranja določenih vrst podatkov, kot npr. ukazov, preden jih pošlje na ponorni naslov.

Primer: strežniku FTP omogočimo dostop od zunaj. Da bi ga zaščitili pred možnimi napadi, proksi FTP v požarnem zidu prepoveduje uporabo ukazov PUT in MPUT.

Strežnik proksi je specifičen aplikacijski posredovalni strežnik (angl. relay) na gostitelju, ki povezuje varno in ne-varno omrežje. Namen strežnika proksi je nadzor izmenjave podatkov med dvema omrežjema na aplikacijski in ne na omrežni plati. Z uporabo strežnika proksi je mogoče onemogočiti usmerjanje IP med varnim in ne-varnim

omrežjem za aplikacijski protokol, ki ga strežnik proksi obravnava, toda še vedno izmenjavati podatke med mrežama z njihovim posredovanjem strežniku proksi (slika 57).



Slika 57: Proksi FTP

Če želimo odjemalcem omogočiti dostop do strežnika proksi, moramo programsko opremo na odjemalcu spremeniti, t.j. programska oprema tako na odjemalcu kot na strežniku mora podpirati povezave proksi. V primeru na sliki 57 se mora odjemalec FTP najprej avtenticirati na strežnik proksi. Po uspešno opravljeni avtentikaciji se seja FTP z znanimi omejitvami proksi lahko začne. Večina strežnikov proksi uporablja bolj zapletene avtentikacijske metode tako kot varnostna karta ID, ki za določeno povezavo generira unikatni ključ.

V primerjavi s filtriranjem IP strežnik proksi omogoča več izčrpnejših logov, ki temeljijo na aplikacijskih podatkih določene povezave. Nadaljnja njegova odlika je, da uporablja strogo avtentikacijo uporabnikov.

Primer: če uporabnik želi uporabljati storitve FTP ali Telnet iz ne-varnega omrežja, se mora najprej avtenticirati na strežniku proksi.

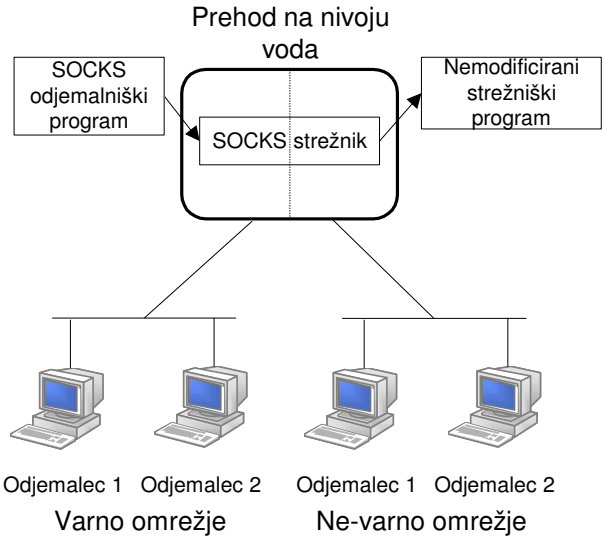
3.3.4 *Prehod na nivoju voda*

Prehod na nivoju voda nadomešča povezave TCP in UDP ter ne zahteva nobenega dodatnega procesiranja ali filtriranja paketov. Prehod na nivoju voda je posebna vrsta aplikacijskega strežnika, ki uporabniku omogoča transparentno uporabo storitev

požarnega zidu. Po uporabnosti je zelo podoben prehodu proksi, dejansko pa med obema obstajajo naslednje razlike:

- Prehod na nivoju voda lahko obdeluje več aplikacij TCP/IP ne, da bi za to potrebovali dodatne spremembe na strani odjemalca.
- Prehod na nivoju voda ne omogoča procesiranja paketov in filtriranja, zato ga označujemo tudi kot *transparentni* strežnik.
- Prehod na nivoju voda ne podpira protokola UDP.
- Prehode na nivoju voda pogosto uporabljamo za izhodne povezave, medtem ko strežnike proksi lahko uporabljamo tako za vhodne kot za izhodne povezave. Običajno v primerih, ko potrebujemo kombinacijo obeh tipov, prehod na nivoju voda uporabimo za izhodne, strežnike proksi pa za vhodne povezave.

Najbolj znan primer prehoda na nivoju voda je *socks* (slika 58). Ker pretok podatkov preko prehoda *socks* ne nadzorujemo ali filtriramo, lahko nastopi varnostni problem. Da zmanjšamo varnostne probleme morajo biti zaupne storitve in viri na zunanjem omrežju (ne-varnem omrežju).



Slika 58: Prehod na nivoju voda

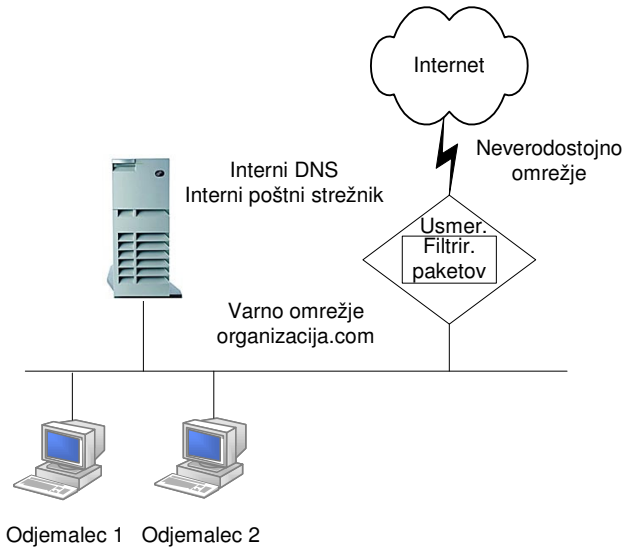
3.3.5 *Primeri požarnih zidov*

Požarni zid sestoji iz enega ali več programskih elementov, ki tečejo na enem ali več gostiteljih. Gostitelji so lahko splošni ali specializirani računalniški sistemi kot npr. usmerjevalniki. Poznamo štiri vrste požarnih zidov:

- *požarni zid s filtriranjem paketov* (angl. Packet-Filtering Firewall),
- *požarni zid z dvodomnim gostiteljem* (angl. Dual-Homed Gateway Firewall),
- *požarni zid z zaslonskim gostiteljem* (angl.Screened Host Firewall),
- *požarni zid z zaslonskim podomrežjem* (angl. Screened Subnet Firewall).

Požarni zid s filtriranjem paketov

Največkrat uporabljamo požarni zid s filtriranjem paketov, saj je to najcenejša vrsta požarnega zidu. Pri tej vrsti požarnega zidu postavimo med varno in ne-varno omrežje usmerjevalnik (slika 59). Usmerjevalnik s filtriranjem paketov za prenos med omrežji uporablja filtrirna pravila, ki dovoljujejo ali prepovedujejo promet. Običajno usmerjevalnik s filtriranjem paketov konfiguriramo tako, da prepovemo vse storitve, ki niso eksplicitno dovoljene.



Slika 59: požarni zid s filtriranjem paketov

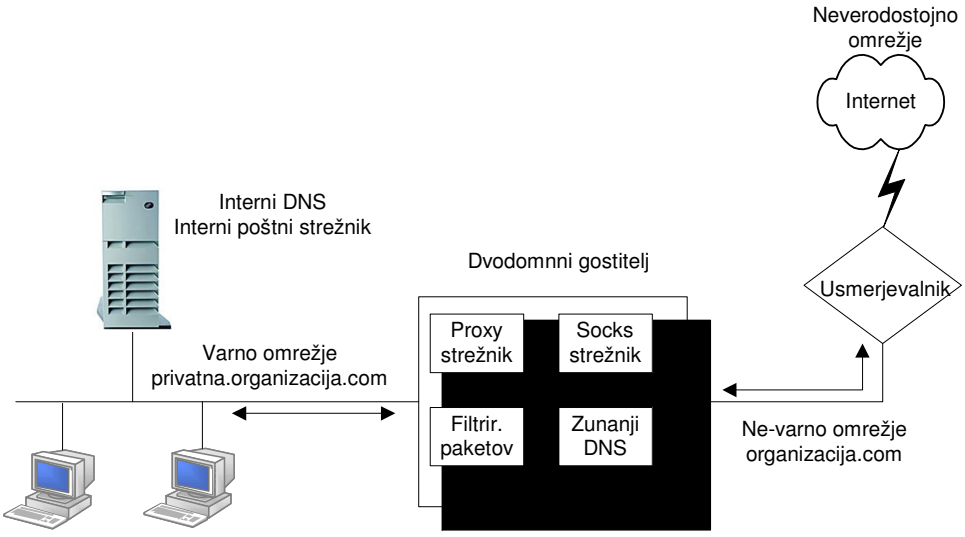
Ker je vsak gostitelj neposredno dostopen z zunanjega omrežja, mora vsak gostitelj v notranjem omrežju imeti lastni avtorizacijski mehanizem.

Požarni zid z dvodomnim gostiteljem

Dvodomni gostitelj ima vsaj dva omrežna vmesnika in vsaj dva naslova IP. Ker posredovanje IP (angl. IP forwarding) ni aktivno, ves promet IP na požarnem zidu prekinemo (slika 60). Tako paketi ne morejo skozi požarni zid, dokler na njem ne definiramo ustrezen prehod *proksi* oz. *socks*. Za razliko od požarnega zidu s filtriranjem paketov ta zagotavlja, da vsak napad, ki prihaja od neznane storitve, blokiramo. Požarni zid te vrste torej uporablja metodo, pri kateri prepovemo vse, kar ni eksplicitno dovoljeno.

Če želimo, da informacijski strežnik (npr. strežnik Web ali FTP) nudi storitve zunanjim uporabnikom, ga je potrebno namestiti v notranjem omrežju ali v relativno ne-varno omrežje med usmerjevalnikom in požarnim zidom. Če ga namestimo zunaj požarnega zidu, mora imeti ta ustrezne storitve proksi, ki omogočajo zunanjim uporabnikom dostop do notranjega informacijskega strežnika. Če pa ga instaliramo med požarnim zidom in usmerjevalnikom, mora usmerjevalnik omogočati filtriranje paketov. Ta tip

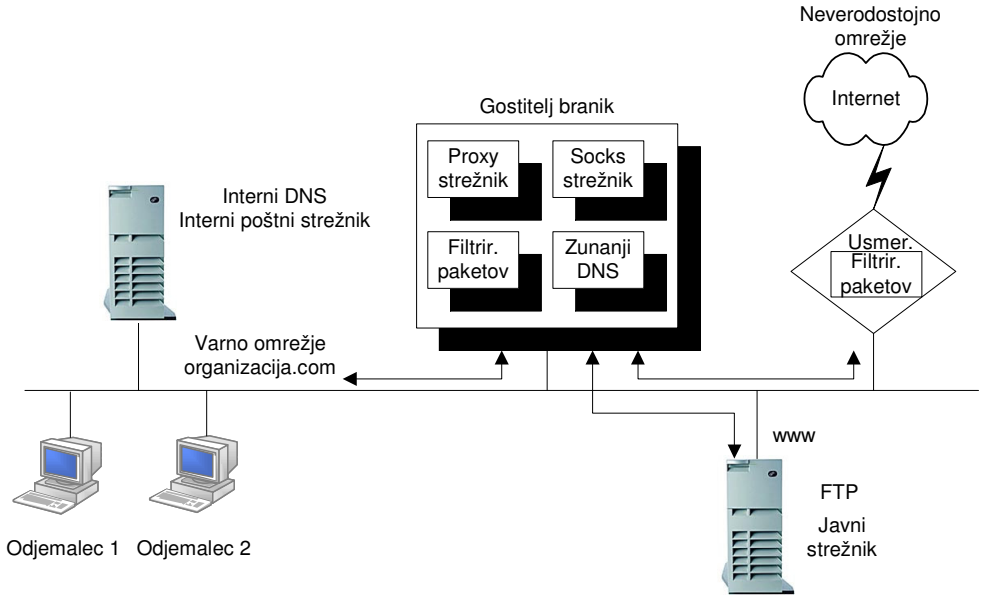
usmerjevalnika imenujemo tudi požarni zid s prikritim gostiteljem in ga obravnavamo v nadaljevanju.



Slika 60: požarni zid z dvodomnim gostiteljem

Požarni zid z zaslonskim gostiteljem

Ta tip požarnega zidu sestoji iz usmerjevalnika s filtriranjem paketov in aplikacijskega strežnika. Usmerjevalnik konfiguriramo tako, da ves promet posreduje aplikacijskemu strežniku v vlogi obrambnega zidu (gostitelj branik) (slika 61). Ker je notranje omrežje na istem podomrežju kot branik, mora varnostna politika omogočati notranjim uporabnikom dostop do zunanjega omrežja neposredno ali jih prisiliti, da za ta dostop uporabljajo storitve proksi. To lahko dosežemo s konfiguriranjem usmerjevalnih pravil tako, da ta sprejema promet, ki ga generira branik.



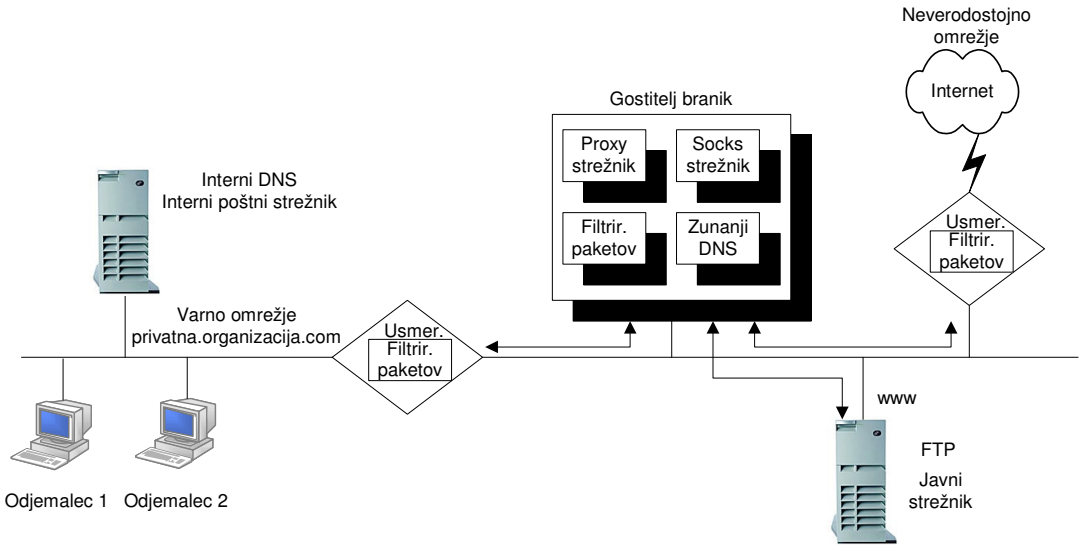
Slika 61: požarni zid s zaslonskim gostiteljem

Ta konfiguracija omogoča, da informacijski strežnik postavimo v omrežje med usmerjevalnikom in branikom. Varnostna politika določa, če bodo zunanji ali notranji uporabniki dostopali do informacijskega strežnika neposredno ali preko branika. Če imamo močno varnostno politiko, mora promet tako iz notranjega kot zunanjega omrežja do informacijskega strežnika prihajati prek branika.

V tej konfiguraciji je obrambni zid lahko običajni gostitelj ali v primeru, da želimo večjo varnost, požarni zid z dvodomnim gostiteljem. V tem primeru ves promet do informacijskega strežnika in do zunanjega omrežja preko usmerjevalnika avtomatsko pošiljamo prek strežnika proksi v dvodomnem gostitelju. Ker je branik edini sistem, ki ga lahko dosežemo od zunaj, nanj ne smemo dovoliti prijave od zunaj.

Požarni zid z zaslonskim omrežjem

Ta tip požarnega zidu sestoji iz dveh usmerjevalnikov s filtriranjem paketov in branika. Požarni zid s prikritim omrežjem omogoča najvišji nivo zaščite med vsemi že omenjenimi primeri (slika 62). Varnost dosežemo s kreiranjem demilitarizirane cone (angl. *Demilitarized Zone*, krajše DMZ) med zunanjim in notranjim omrežjem tako, da zunanji usmerjevalnik dovoljuje dostop iz zunanjega omrežja do branika in da notranji usmerjevalnik dovoljuje dostop iz notranjega omrežja do branika. Ker zunanji usmerjevalnik objavi naslove DMZ zunanjemu omrežju, sistemi v zunanjem omrežju ne morejo dostopati do notranjega omrežja. Notranji usmerjevalnik objavi naslove DMZ notranjemu omrežju, t.j. sistemi v notranjem omrežju ne morejo neposredno dostopati do strežnikov na spletu. Taka struktura omogoča močno varnost, saj se mora morebitni napadalec prebiti skozi tri ločene sisteme, da bi prišel na notranje omrežje.



Slika 62: požarni zid z zaslonskim omrežjem

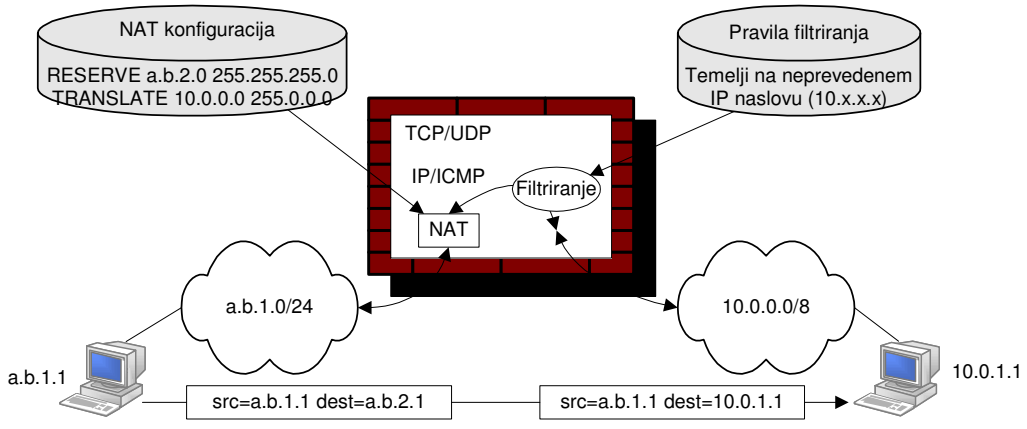
Ker usmerjevalnika prisilita sisteme tako v notranjem kot zunanjem omrežju, da uporabljajo branik, tukaj za branik ni potrebno uporabljati dvodomnega gostitelja, kar je ena izmed pomembnejših koristi cone DMZ. Taka postavitev istočasno omogoča veliko hitrejši odzivni čas, kot v primeru dvodomnega gostitelja.

3.4 Prevajanje omrežnih naslovov IP

Ideja prevajanja omrežnih naslovov (angl. Network Address Translation, krajše NAT) temelji na dejstvu, da samo manjši del gostiteljev v zasebnem omrežju komunicira z zunanjim omrežjem. Če vsakemu gostitelju priredimo naslov IP iz bazena javnih naslovov IP (angl. address pool) samo v času, ko ta komunicira z zunanjim omrežjem, potrebujemo samo manjše število javnih naslovov.

S prevajanjem omrežnih naslovov NAT lahko večja lokalna omrežja, ki želijo komunicirati z gostitelji na spletu, rešimo z manjšim območjem zasebnih naslovov IP. Odjemalci, ki dostopajo do strežnikov na spletu prek prehodov *proxy* ali *socks*, svojih naslovov IP ne izpostavljajo, zato njihovih naslovov ni potrebno prevajati. V primeru, da prehoda *proxy* in *socks* nista na voljo, lahko za upravljanje prometa med notranjim in zunanjim omrežjem, uporabimo NAT.

Predpostavimo, da notranje omrežje temelji na prostoru zasebnih naslovov IP in da uporabniki želijo uporabljati aplikacijski protokol, za katerega nimamo ustreznega aplikacijskega strežnika. V tem primeru je edina možnost komunikacije vzpostavitev povezave IP med gostitelji v notranjem omrežju in gostitelji na spletu. Usmerjevalnik na spletu ne more poslati paket IP na zasebni naslov IP, zato NAT rešuje ta problem tako, da vzame naslov IP izhodnega paketa in ga dinamično prevede v javni naslov IP (slika 63), pri vhodnih paketih prevede javni naslov IP v naslov IP zasebnega omrežja.

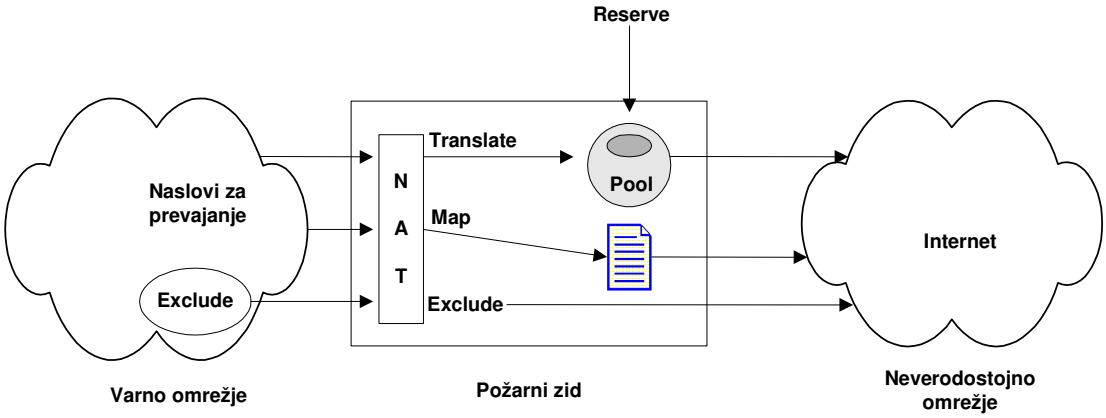


Slika 63: Prevajanje omrežnih naslovov

NAT z vidika dveh gostiteljev, enega v varnem in drugega v ne-varnem omrežju, ki izmenjujeta pakete IP med seboj, zgleda kot standardni usmerjevalnik IP, ki posreduje pakete IP med dvema omrežnima vmesnikoma.

3.4.1 *Mehanizem prevajanja*

Konfiguracijska pravila NAT preverjajo izvorni naslov vsakega izhodnega paketa IP. Če se pravilo ujema z izvornim naslovom IP, ga prevedemo v enega izmed javnih naslov IP iz bazena naslovov IP. Bazen naslovov IP vsebuje predefinirane naslove IP, ki jih NAT pri prevajanju lahko uporablja. NAT za vsak vhodni paket preveri ponorni naslov in če se s pravilom ujema, ga prevede v originalni zasebni naslov IP (slika 64).



Slika 64: NAT konfiguracija

NAT prevaja samo pakete TCP in UDP, protokola ICMP ne podpira.

Primer: *ping* do naslova NAT ne deluje, ker *ping* uporablja protokol ICMP.

Ker se sklad TCP/IP, ki ga uporablja NAT, ne razlikuje od sklada normalnega usmerjevalnika IP, za povezavo dveh omrežij ni potrebno dodatnega usmerjevalnika IP. Javne naslove iz bazena naslovov IP je potrebno pred uporabo rezervirati. Če s spletom vzpostavimo povezavo iz varnega omrežja, NAT vzame naslednji prosti naslov IP iz bazena naslovov IP in ga pridruži varnemu gostitelju. O preslikanih naslovih IP v določenem trenutku vodi evidenco. Ko na osnovi zahteve priredi določen naslov IP, mora vedeti, kdaj ga vrniti v bazen naslovov IP. V protokolu IP ne obstaja mehanizem, ki bi mu lahko pri tem pomagal. Ker je protokol TCP povezavno orientiran, je mogoče stanje povezave dobiti iz informacij v glavi segmenta TCP (ali je povezava prekinjena ali ne). UDP protokol takih informacij ne vključuje, zato potrebujemo neko časovno omejitev, ki obvesti NAT, da je ustrezen naslov IP potrebno vrniti v bazen javnih naslovov IP.

NAT pozna štiri oblike prevajanj, ki jih opisujemo z zapisi v konfiguracijski datoteki. Zapisi tipa *Reserve*, *Translate* in *Exclude* opisujejo dinamične odjemalce, z zapisom *Map* pa definiramo strežnike. V nadaljevanju poglejmo osnovne značilnosti posameznih zapisov.

Rezerviraj javne IP naslove

Ta zapis definira množico javnih, t.j. registriranih, naslovov IP, ki jih uporabljamo pri izhodnih povezavah. Ko varni gostitelj pošlje pakete v ne-varno omrežje, iz

bazena rezerviranih naslovov IP določi javni naslov IP. Ta naslov uporabimo za transport paketov IP med požarnim zidom in zunanjim omrežjem.

Primer: rezervirani javni naslov IP opišemo z naslednjim zapisom

RESERVE 195.9.5.0 255.255.255.0 30,

kjer rezerviranemu javnemu naslovu IP sledi časovna omejitev, t.j. po kolikšnem času neaktivnosti lahko NAT vrne določen naslov IP v bazen registriranih naslovov IP.

Prevedi varne IP naslove

Ta zapis definira množico varnih naslovov IP, ki zahtevajo prevajanje.

Primer: varni IP naslov prevedemo z naslednjim zapisom

TRANSLATE 126.1.2.0 255.255.255.0.

Izloči varne IP naslove

Zapis definira množico varnih omrežnih naslovov IP, ki ne zahtevajo prevajanja. NAT običajno izvaja prevajanje naslovov IP na vseh varnih naslovih IP.

Primer: varni naslov IP izločimo iz procesa prevajanja z naslednjim zapisom

EXCLUDE 128.1.2.0 255.255.255.0

Preslikaj varni IP naslov

Preslikava varnega naslova IP definira preslikavo ena-na-ena iz varnega naslova IP v javni naslov IP. Ta preslikava omogoča zunanjim aplikacijskim odjemalcem, kot npr. odjemalci FTP ali Telnet, vzpostaviti sejo TCP s strežnikom v varnem omrežju. Območje javnih naslovov IP v zapisu lahko prekrivamo z območjem naslovov IP določenih z zapisom RESERVE.

Primer: zapis statičnega prevajanja naslovov je naslednji

MAP 126.1.2.6 195.9.5.6

3.4.2 *omejitve NAT*

NAT deluje dobro za naslove IP v glavi paketa IP. Nekateri aplikacijski protokoli izmenjujejo informacije o naslovih IP v aplikacijskih podatkih, ki so del paketa IP in NAT pri teh aplikacijskih protokolih običajno ne more izvajati prevajanje naslovov IP uspešno. Implementacija NAT za specifične aplikacijske protokole, ki hranijo informacije IP v aplikacijskih podatkih je mnogo bolj zapletena od standardne implementacije. Danes največ implementacij NAT obravnava protokol FTP.

Naslednja pomembna omejitev je, da NAT spreminja nekatere informacije o naslovih v paketu IP. Če uporabljamo avtentikacijo IPSec, paket, katerega naslov spremenimo, test preverjanja integritete vedno zavrne, saj vsaka sprememba v datagramu IP spremeni vrednost avtentikacijske kode, s katero preverjamo integriteto izvornega gostitelja.

3.5 Varnostna arhitektura IP

Varnostna arhitektura IP (angl. IP Security Architecture, krajše IPSec) sestoji iz treh glavnih komponent: avtentikacijske glave (angl. Authentication Header, krajše AH), varovanja koristne vsebine z ovijanjem (angl. Encapsulating Security Payload, krajše ESP) in internetne izmenjave ključa (angl. Internet Key Exchange, krajše IKE). Pri delovanju IPSec uporablja dva glavna koncepta, t.j. varnostno zvezo (angl. Security Association, krajše SA) in tuneliranje (angl. tunneling). Oba koncepta v nadaljevanju opisujemo podrobneje.

Varnostna zveza

Varnostna zveza SA je enosmerna logična povezava med dvema sistemoma IPSec enolično določena s sledečo trojico:

<kazalo varnostnih parametrov, ponorni naslov IP, varnostni protokol>

Definicija članov trojice je naslednja:

Kazalo varnostnih parametrov (angl. Security Parameter Index, krajše SPI)
To je 32-bitno število, ki ga uporabljamo za identifikacijo različnih varnostnih zvez SA z istim ponornim naslovom IP in varnostnim protokolom. SPI prenašamo v glavi paketov pri varnostnih protokolih (AH ali ESP) in ima lokalni pomen, ki ga definira skrbnik SA. Mednarodna organizacija IANA je rezervirala vrednosti SPI v intervalu 1 do 255.

Ponorni naslov IP (angl. IP Destination Address)
To je lahko naslov *unicast*, *broadcast* ali *multicast*.

Varnostni protokol (angl. Security Protocol)
Ta je lahko AH ali ESP.

Varnostna zveza SA lahko deluje v prenosnem (transport) ali tunelskem načinu v odvisnosti od uporabljenega varnostnega protokola. Ker je varnostna zveza enosmerna, potrebujemo za dvosmerno povezavo med dvema sistemoma IPSec dve, t.j. eno za vsako smer. Za povezave, ki jih želimo zaščititi z obema varnostnima protokoloma AH in ESP, prav tako potrebujemo dve varnostni zvezi. Množico varnostnih zvez, ki definirajo povezavo, imenujemo sveženj (angl. bundle). Protokol IPSec vzdržuje dve podatkovni bazi, s katerimi opisujemo varnostne zveze:

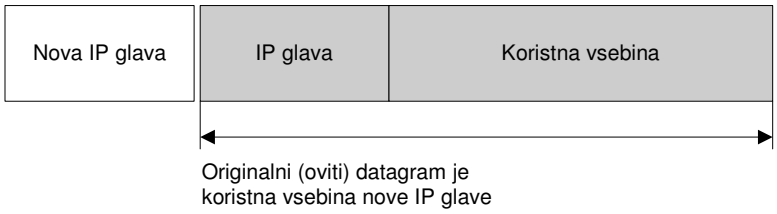
- podatkovna baza varnostne politike (angl. Security Policy Database, krajše SPD): določa, katere varnostne storitve, glede na faktorje kot npr. izvor, ponor, vhodni ali izhodni promet itd., ponujamo prometu IP. Podatkovna baza vsebuje urejen

seznam politik, ločeno za vhodni in izhodni promet. Te politike določajo, kateri promet ne sme skozi IPSec, katerega IPSec zbriše in katerega procesira. Zapisi v tej podatkovni bazi so podobni pravilom požarnega zidu ali filtriranju paketov.

- podatkovna baza SA (angl. Security Association Database, krajše SAD): vsebuje parametre o vsaki varnostni zvezi, o algoritmih in ključih varnostnih protokolov AH ali ESP, zaporednih številkah, metodah protokola in njeni življenjski dobi. Pri izhodnem procesiranju SPD določi, katero varnostno zvezo uporabimo za določen paket. Pri vhodnem procesiranju, SAD določi, kako moramo paket procesirati.

Tuneliranje

Tuneliranje ali ovijanje (angl. tunneling, encapsulation) je tehnika, ki jo poznamo v preklopnih omrežjih in temelji na ovijanju originalnega paketa v novega, t.j. originalnemu paketu priredimo novo glavo paketa, originalni paket pa postane koristna vsebina novega (slika 65).



Slika 65: Tuneliranje IP

V splošnem uporabljamo tuneliranje za prenos paketov po enega protokola prek omrežja, ki deluje po drugem protokolu.

Primer: protokola NetBIOS ali IPX ovijemo v protokol IP, da ga lahko prenesemo preko povezav TCP/IP WAN.

V primeru protokola IPSec protokol IP ovijamo v protokol IP zaradi drugačnega razloga, t.j. da bi omogočili popolno zaščito paketa vključno z glavo ovitega paketa. Če oviti paket šifriramo, napadalec ne more razbrati vsebine posameznih polj originalnega paketa, npr. ponornega naslova. Notranjo strukturo privatnega omrežja lahko na ta način skrijemo.

Tuneliranje zahteva vmesno procesiranje originalnega paketa na njegovi poti. To dodatno procesiranje pa odtehta dodatna varnost. Oviti paket s ponornim naslovom v zunanji glavi paketa, požarni zid ali usmerjevalnik IPSec običajno sprejme in ga pošlje ponornemu gostitelju.

Pomembna prednost tuneliranja IP je možnost izmenjave paketov med dvema intranetoma z zasebnimi IP naslovi prek javnega spletnega omrežja, ki zahteva globalne unikatne naslove. Ker glavo ovitega paketa IP spletni usmerjevalniki ne procesirajo, imata samo končni točki tunela globalno pridružene naslove IP, gostitelji v intranetu pa lahko uporabljajo privatne naslove.

3.5.1 *Avtentikacijska glava*

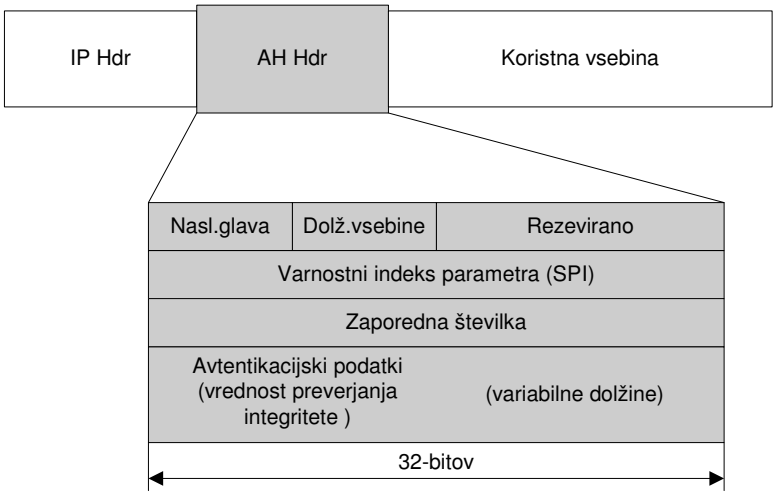
Varnostni protokol avtentikacijska glava AH omogoča datagramom IP integriteto in avtentikacijo. Čeprav njegova uporaba ni obvezna, pa ga moramo uporabljati, če želimo kompatibilnost z ostalimi sistemi IPSec. Nekatera polja v glavi datagrama IP se na poti do prejemnika spreminjajo, zato njihovih vrednosti ta ne more predvideti. Ta polja imenujemo *spremenljiva* in jih protokol AH ne zaščiti. Spremenljiva polja v IPv4 so:

- *vrsta storitve* (angl. Type of Service, TOS),
- *zastavice* (angl. Flags),
- *izravnava fragmenta* (angl. Fragment Offset),
- *življenjski čas* (angl. Time to Live, TTL),
- *kontrolna vsota glave paketa* (Header Checksum).

Ko zahtevamo zaščito teh polj, moramo uporabiti tuneliranje. Koristna vsebina paketa IP velja za nespremenljivo in jo protokol AH vedno zaščiti. Številka protokola AH, ki jo je dodelila organizacija IANA, je 51. Procesiranje paketov z avtentikacijsko glavo uporabimo samo na ne-fragmentiranih paketih IP. Paket IP z avtentikacijsko glavo lahko fragmetirajo vmesni usmerjevalniki. Pakete, ki jih ni mogoče avtenticirati, zavržemo. Ta način delovanja znatno zmanjša možnost uspešnega napada, t.i. *preklica storitve*, katerega namen je blokirati komunikacijo gostitelja s poplavo ponarejenih paketov.

Format avtentikacijske glave

Slika 66 predstavlja polja avtentikacijske glave v paketu IP.



Slika 66: Format paketa z avtentikacijsko glavo

Format avtentikacijske glave je naslednji:

Naslednja glava (angl. Next Header)

8-bitno polje, ki označuje tip naslednjega koristne vsebine, katera sledi zapisu avtentikacijske glave. Vrednosti tega polja predpisuje IANA.

Dolžina vsebine (angl. Payload Length)

8-bitno polje, ki vsebuje dolžino avtentikacijske glave izražene v 32-bitnih besedah minus 2. Privzeta vrednost tega polja je 4, t.j. 3x32-bit fiksne besede + 3x32-bit besede podatkov za avtentikacijo – 2.

Rezervirano (angl. Reserved)

16-bitno polje, rezervirano za bodočo uporabo.

Kazalo varnostnih parametrov (angl. Security Parameter Index, SPI)

32-bitno polje, ki smo ga opisali že v uvodu tega poglavja.

Zaporedna številka (angl. Sequence Number)

32-bitno polje je monotono naraščajoči števec, ki ga uporabljamo pri *zaščiti ponovitev* (angl. replay protection) pošiljanja istega paketa. Ta zaščita ni obvezna, kar pa ne velja za uporabo tega polja. Ob vzpostavitvi povezave postavimo zaporedno številko na 0. Prvi preneseni paket z uporabo SA ima zaporedno številko 1. Zaporedna številka se ne sme ponoviti. Najvišja zaporedna številka paketa IP, ki ga lahko prenesemo po dani SA je $2^{32}-1$.

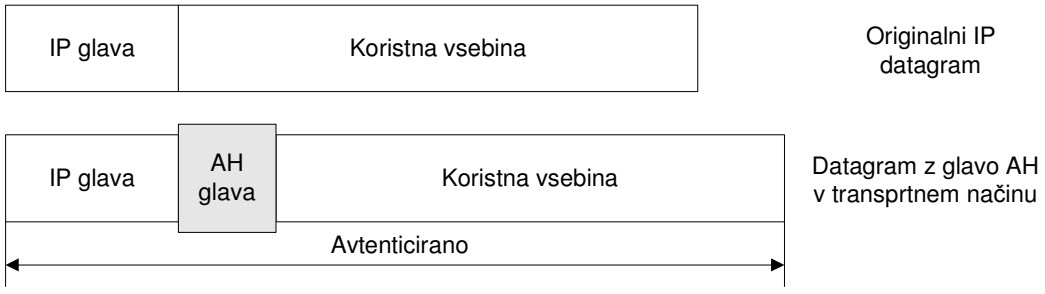
Aplikacijski podatki (angl. Authentication Data)

Polje variabilne dolžine, ki ga imenujemo tudi ICV (angl. Integrity Check Value). ICV za paket izračunamo z algoritmom izbranim ob inicializaciji SA. Dolžina avtentikacijskih podatkov je večkratnik 32-bitnega števila. Sprejemnik uporablja vrednost ICV za preverjanje integritete vhodnega paketa. Teoretično lahko za izračun vrednosti ICV uporabimo katerikoli algoritem MAC. V praksi najpogosteje uporabljamo algoritma MD5 in SHA-1 s ključem.

Načini uporabe avtentikacijske glave

Avtentikacijsko glavo lahko uporabljamo v transportnem ali v tunelskem načinu delovanja.

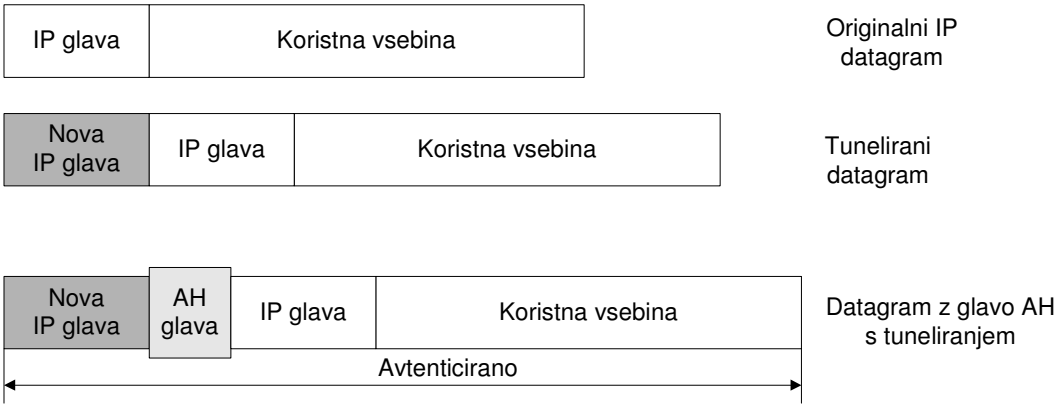
Avtentikacijska glava v transportnem načinu: v tem načinu vzamemo originalni IP datagram in vrinemo avtentikacijsko glavo poleg glave datagrama IP (slika 67). Če ima datagram že glavo IPSec, avtentikacijsko glavo vrinemo pred njo.



Slika 67: Avtentikacijska glava v transportnem načinu

Transportni način uporabljajo gostitelji in ne usmerjevalniki. Usmerjevalniki transportnega načina niti ne podpirajo. Prednost tega načina je manj procesiranja, pomanjkljivost pa, da spremenljivih polj ne avtenticiramo.

Avtentikacijska glava v tunelskem načinu: v tem načinu uporabljamo koncept tuneliranja (angl. tunneling), pri katerem tvorimo nov datagram IP, originalni pa postane njegova koristna vsebina. Na paketu, ki ga dobimo, uporabimo avtentikacijsko glavo v transportnem načinu (slika 68).



Slika 68: Avtentikacijska glava v tunelskem načinu

Tunelski način uporabljamo vedno, kadar je ena izmed točk komunikacije usmerjevalnik. Med dvema požarnima zidovoma ga uporabljamo vedno. Čeprav je tunelski način delovanja značilen za usmerjevalnike, lahko ti pogosto delujejo tudi v transportnem načinu. Ta način delovanja uporabljamo takrat, ko usmerjevalnik igra vlogo gostitelja, t.j. v primeru, ko promet naslovimo na njega. Primera takih zahtev sta ukaz SNMP ali ICMP.

Pri tunelskem načinu ni potrebno, da je zunanji naslov IP enak notranjemu.

Primer: dva požarna zidova uporabljata avtentikacijsko glavo za avtentikacijo celotnega prometa med omrežjema, ki ga povezujeta. Od gostiteljev v obeh omrežjih ne zahtevamo, da podpirajo tunelskega načina delovanja.

Prednost takega načina delovanja je celotna zaščita ovitih datagramov IP in možnost uporabe zasebnih naslovov, slabost pa dodatno procesiranje.

3.5.2 *Varovanje koristne vsebine z ovijanjem*

Protokol varovanje koristne vsebine z ovijanjem ESP uporabljamo za zagotavljanje integritete, avtentikacije in enkripcije datagramov IP. Omogoča tudi zaščito ponovitev. Omenjene storitve so nepovezane in delujejo na osnovi pošiljanja paketov po povezavi med dvema usmerjevalnikoma oz. požarnima zidovoma. Značilnosti povezave definiramo pri vzpostavljanju varnostne zveze. Pri tem veljajo naslednje omejitve:

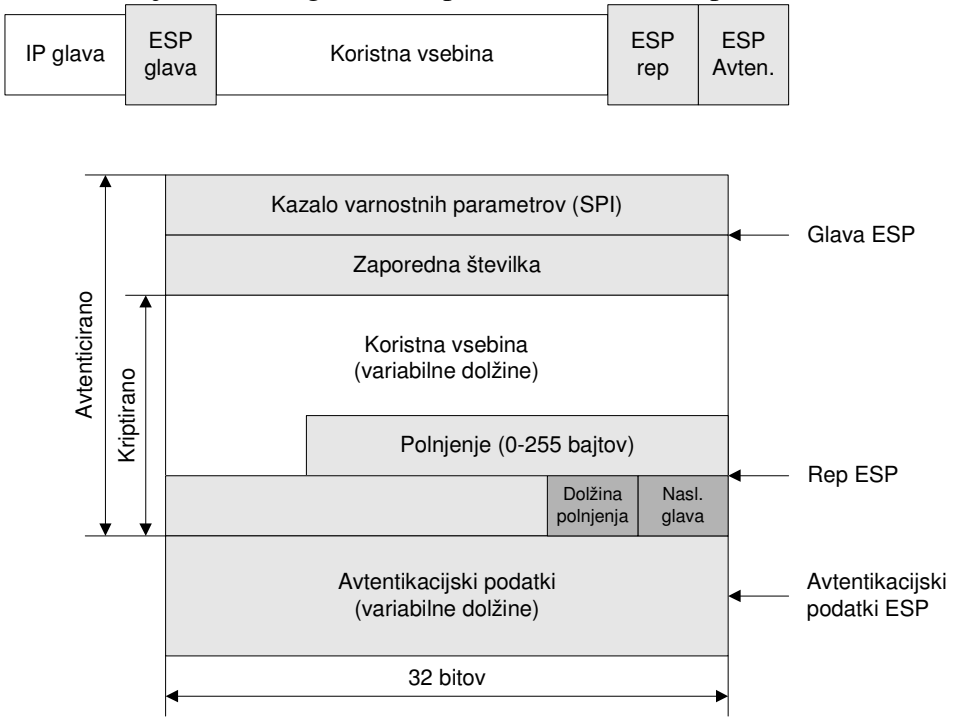
- če izberemo storitev preverjanja integritete, moramo izbrati tudi avtentikacijo,
- storitev zaščita ponovitev izberemo samo pri preverjanju integritete in avtentikacije,
- storitev zaščita ponovitev lahko izbira samo prejemnik.

Šifrirano povezavo lahko izberemo neodvisno od drugih storitev. Če izberemo šifriranje, je zelo priporočljivo izbrati tudi preverjanje integritete in avtentikacijo. Če izberemo samo šifriranje, lahko napadalec s kripto-analitičnimi napadi ponaredi pakete, kar ni mogoče v primeru uporabe preverjanja integritete in avtentikacije. Čeprav sta avtentikacija (s preverjanjem integritete) in šifriranje opcijska, pa ponavadi izberemo vsaj eno izmed njih. Če izberemo šifriranje in avtentikacijo s preverjanjem integritete, prejemnik najprej avtenticira paket in samo, če ta korak izvede uspešno, lahko nadaljuje z dešifriranjem. Ta način delovanja prihrani računalniške vire in zmanjša ranljivost napadov.

Varnostni protokol ESP uporablja številko vrat 50, ki jo je določila IANA. Protokol ESP procesira samo ne-fragmentirane pakete, fragmentirane pakete pa obravnavajo vmesni usmerjevalniki. V primeru fragmentiranja ponorni gostitelj najprej zbere paket in ga šele potem prepusti procesiranju protokola ESP. Če ta obdeluje fragmentirani paket IP, katerega vsebina polja offset ni enako nič, ga uniči.

Format paketa ESP

Format paketa ESP je veliko bolj zapleten od paketa z avtentikacijsko glavo. Poleg glave paketa ESP imamo tukaj tudi rep paketa ESP in avtentikacijske podatke paketa ESP (slika 69). Koristni tovor je ovit med glavo in repom. Od tu tudi ime protokola.



Slika 69: Glava in rep paketa ESP

Paket ESP sestavljajo naslednja polja:

Kazalo varnostnih parametrov (angl. Security Parameter Index, SPI)
32-bitno polje, ki smo ga opisali že v uvodu tega poglavja.

Zaporedna številka (angl. Sequence Number)
32-bitno monotono naraščajoči števec, ki smo ga opisali že pri formatu avtentikacijske glave.

Koristni tovor (angl. Payload Data)
To polje je obvezno in sestoji iz variabilnega števila bajtov podatkov opisanih s poljem *Next Header* (slo. sledeča glava). To polje kriptiramo s kriptografskim algoritmom izbranim med vzpostavljanjem SA. Specifikacija protokola ESP zahteva podporo algoritma DES v CBC načinu (preslikavo DES-CBC). Pogosto podpira tudi algoritma trojni-DES (angl. triple-DES) in CDMF.

Polnjenje (angl. Padding)
Večina šifrirnih algoritmov zahteva, da morajo biti vhodni podatki popolno število blokov. Dolžina šifriranega teksta mora biti delitelj števila 4 tako, da je polje *Next Header* desno poravnano. To je tudi razlog, zakaj vključimo polje variabilne dolžine. Šifriranje zajema polja: koristni tovor, polnjenje, dolžino polnjenja in sledeča glava.

Dolžina polnjenja
To 8-bitno polje vsebuje število predhodnih bajtov, s katerimi dopolnimo polje koristni tovor do dolžine, ki je delitelj števila 4. Število 0 označuje, da polnjenja nimamo.

Naslednja glava (angl. Naxt Header)
Naslednja glava je 8-bitno obvezno polje, ki določa podatkovni tip tovora, npr. višje-nivojski identifikator protokola kot recimo TCP. Zalogo vrednosti tega polja predstavljajo številke protokolov IP določene pri mednarodni organizaciji IANA.

Avtentikacijski podatki (angl. Authentication Data)
To je polje variabilne dolžine in vsebuje izračunani ICV za paket ESP od polja SPI do polja sledeče glave vključno. To polje je opcijsko. Vključimo ga samo v primeru, da izberemo ob inicializaciji SA tudi preverjanje integritete in avtentikacijo. Specifikacija ESP zahteva podporo dveh avtentikacijskih algoritmov: HMAC z MD5 in HMAC z SHA-1.

Načini uporabe ESP

Podobno kot pri varnostnem protokolu AH lahko tudi protokol ESP uporabimo na dva načina: transportni in tunelski način.

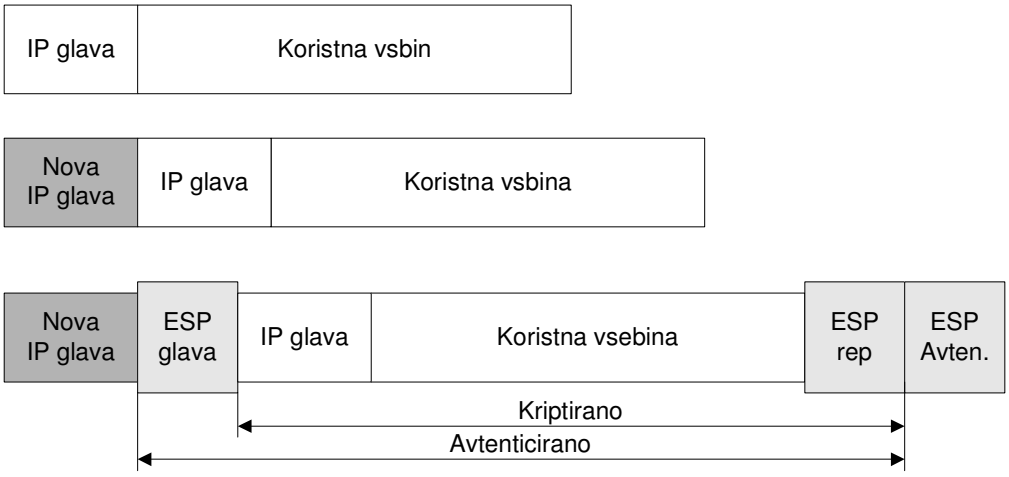
ESP v transportnem načinu: v tem načinu vzamemo originalni datagram IP in za glavo paketa IP vrinemo glavo paketa ESP (slika 70). Če datagram glavo IPSec že ima, glavo paketa ESP vrinemo pred njo. Rep paketa ESP in opcijske avtentikacijske podatke pripnemo za koristni tovor.

Protokol ESP v transportnem načinu za glavo paketa IP ne omogoča niti avtentikacije niti šifriranja. To je pomanjkljivost, ker lahko protokol ESP procesira napačne pakete.

Prednost transportnega načina pa so nižji stroški procesiranja. Podobno kot v primeru protokola AH tudi protokol ESP v transportnem načinu uporabljajo predvsem gostitelji, ne usmerjevalniki. Za podporo transportnega načina usmerjevalniki niso potrebni.

Slika 70: Protokol ESP v transportnem načinu

ESP v tunelskem načinu: v tem načinu protokol ESP uporablja princip tuneliranja. Novi paket IP tvorimo z novo glavo IP in nato uporabimo paket ESP v transportnem načinu (slika 71). Ker originalni datagram postane koristni tovor novega paketa ESP, v primeru, da izberemo šifrirano in avtentikacirano povezavo, zagotavljamo celovito zaščito. Nova glava IP tudi v tem primeru ni zaščitena.



Slika 71: Protokol ESP v tunelskem načinu

Tunelski način uporabljamo v primeru, da je vsaj ena izmed končnih točk komunikacije usmerjevalnik.

Primer: med dvema požarnima zidovoma uporabljamo tunelski način delovanja.

Čeprav je tunelski način delovanja značilen za usmerjevalnike, lahko ti pogosto delujejo tudi v transportnem načinu. Ta način uporabljamo takrat, ko usmerjevalnik igra vlogo gostitelja, t.j. v primeru, ko promet naslovimo na njega. Primera takih zahtev sta ukaz

SNMP ali ICMP. Pri tunelskem načinu delovanja ni potrebno, da je zunanji naslov IP enak notranjemu.

Primer: dva požarna zidova delujeta s tunelom AH, ki ga uporabljata za avtentikacijo celotnega prometa med omrežjema, ki ga povezujeta. Od gostiteljev v obeh omrežjih ne zahtevamo, da podpirajo tunelski način delovanja.

Prednost takega načina delovanja je celotna zaščita ovitih datagramov IP in možnost uporabe zasebnih naslovov, slabost pa seveda dodatni stroški procesiranja.

Zakaj dva avtentikacijska protokola?

Če poznamo značilnosti varnostnega protokola ESP, se lahko upravičeno vprašamo, ali res potrebujemo še protokol AH. Zakaj avtentikacija protokola ESP ne pokriva tudi glave datagrama IP? Na to dvoje vprašanj ne obstaja uradni odgovor, pač pa obstajajo nekatera spoznanja, ki opravičujejo obstoj dveh različnih varnostnih protokolov IPSec:

- Protokol ESP zahteva implementacijo močnih kriptografskih algoritmov, pa čeprav jih ne uporabljamo. Močna kriptografija je v nekaterih državah, kjer imajo stroge omejitvene predpise, občutljiva tema. V teh področjih je uporaba rešitev, ki temeljijo na protokolu ESP, težavna. Avtentikacija ni predpisana, zato lahko protokol AH uporabljamo po svetu brez omejitev.
- Pogosto uporabljamo samo avtentikacijo. Avtentikacija pri protokolu AH je v primerjavi z avtentikacijo pri protokolu ESP učinkovitejša zaradi enostavnejšega formata in nižjih stroškov procesiranja. V takih primerih je boljše uporabiti protokol AH.
- Imeti dva različna protokola pomeni preciznejšo stopnjo nadzora prek omrežja IPSec in bolj fleksibilne varnostne možnosti. Z gnezdenjem protokolov AH in ESP lahko, na primer, implementiramo tunele IPSec s kombinacijo prednosti obeh protokolov.

3.5.3 *Kombiniranje protokolov IPSec*

Protokola AH in ESP lahko uporabimo posamezno ali v kombinaciji. Za danima dvema načinoma uporabe vsakega protokola obstaja dovolj veliko število mogočih kombinacij. Sliko nadalje zaplete tudi dejstvo, da varnostna zveza z varnostnima protokoloma AH in ESP ne zahteva identičnih končnih točk. Na srečo v realnem življenju prihaja v poštev samo nekaj od teh možnosti.

Omenili smo že, da s svežnji SA realiziramo kombinacije protokolov IPSec. Za kreiranje svežnjev SA obstajata dva pristopa:

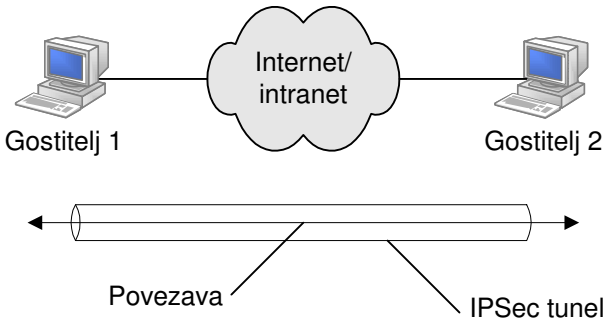
- *Transportna soseščina*: oba varnostna protokola AH in ESP uporabimo na istem datagramu IP v transportnem načinu. Ta metoda je uporabna za eno-nivojsko kombiniranje.

- *Iterativni (gnezdeni) tunelski način*: varnostna protokola AH in ESP uporabimo zaporedno v tunelskem načinu delovanja. Po delovanju enega protokola dobimo nov datagram IP, na katerem uporabimo naslednji protokol. Ta metoda nima omejitve v nivojih gnezdenja, vendar v praksi ne uporabljamo več kot treh nivojev.

Če gradimo navidezna privatna omrežja (angl. Virtual Private Network, krajše VPN), moramo procesiranje določenega paketa IPSec omejiti na razumen nivo. V večini praktičnih primerov sta dva koraka dovolj. Da bi tvorili sveženj SA, v katerem imajo SA različne končne točke, moramo uporabiti vsaj en tunelski nivo. Pri praktični uporabi kombiniranja protokolov velja princip, da po sprejemu paketa z obema glavama protokolov, po avtentikaciji uporabimo dešifriranje. Po tem principu pošiljatelj na izhodni promet najprej uporabi varnostni protokol ESP in potem protokol AH. To zaporedje je v principu eksplicitna zahteva za procesiranje IPSec v transportnem načinu. Ko uporabljamo oba varnostna protokola ESP in AH se pojavi naslednje vprašanje: ali uporabiti avtentikacijo s protokolom ESP? Protokol AH avtenticira paket v vsakem primeru. Odgovor je enostaven: avtentikacija s protokolom ESP je smiselna samo, če se SA s protokolom ESP razširi prek SA s protokolom AH. V tem primeru ne samo, da je smiselno uporabiti avtentikacijo s protokolom ESP, ampak celo zelo priporočljivo, saj se tem izognemo napadom *prevar* (angl. spoofing) v notranjem omrežju.

Primer 1: Varnost od točke do točke

Dva gostitelja priključimo prek spleta (ali na intranet) brez vsakršnega usmerjevalnika IPSec med njima (slika 72). Oba lahko uporabljata varnostna protokola ESP, AH ali oboje. Uporabimo lahko transportni ali tunelski način delovanja.



Slika 72: Varnost od točke do točke

Kombinacije, podprte pri vseh implementacijah IPSec, so naslednje:

Transportni način

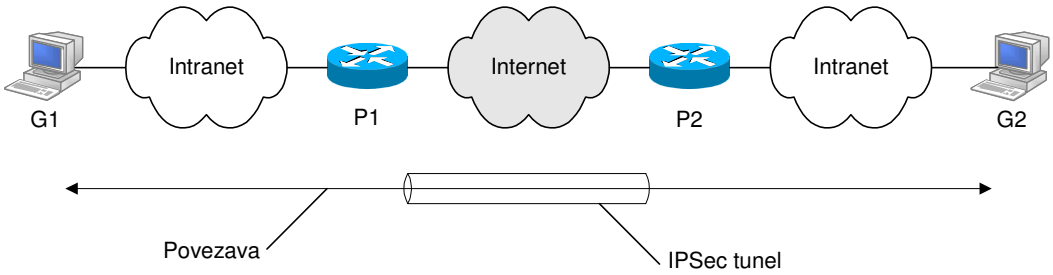
- samo protokol AH,
- samo protokol ESP,
- protokol AH uporabljen po protokolu ESP (transportna soseščina).

Tunelski način

1. samo protokol AH,
2. samo protokol ESP.

Primer 2: Osnovna podpora navideznih privatnih omrežij

Slika 73 prikazuje najenostavnejše navidezno privatno omrežje. Usmerjevalnika P1 in P2 delujeta na protokolnem skladu IPSec. Za gostitelja v intranetu ne zahtevamo podpore za IPSec.

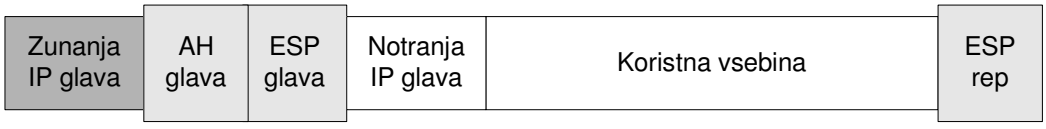


Slika 73: Osnovna podpora VPN

V tem primeru od usmerjevalnikov zahtevamo podporo tunelskega načina delovanja s protokolom AH ali ESP.

Kombinirani tunel med usmerjevalnikoma: čeprav od usmerjevalnikov zahtevamo samo podporo protokolov AH in ESP s tunelskim načinom delovanja, je pogosto zaželeno, da takšni tuneli znajo kombinirati odlike obeh varnostnih protokolov IPSec. IBM implementacije IPSec podpirajo kombinirani tip obeh. Uporabnik izbira vrstni red glav datagramov IP pri postavljanju politike tunela.

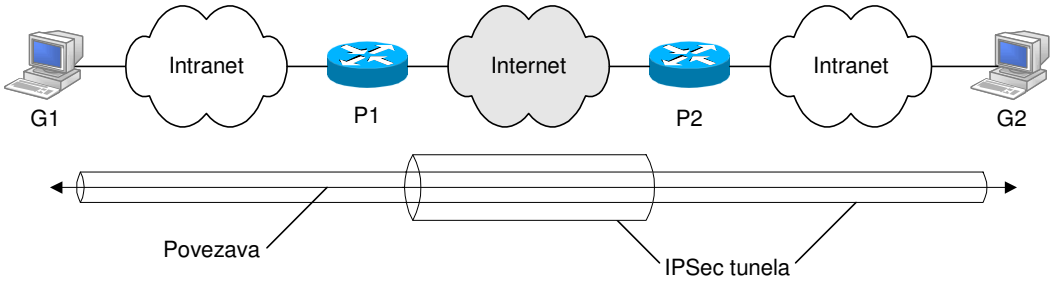
Kombinirani tunel med usmerjevalnikoma ne pomeni, da imamo opravka z iterativnim tunelskim načinom delovanja. Ker ima sveženj SA, ki obsega tunel, identični končni točki, je iterativni tunelski način lahko zelo neučinkovit. Namesto tega en varnostni protokol IPSec uporabimo v transportnem, drugega pa v tunelskem načinu delovanja, kar konceptualno pomeni kombinirani tunel AH-ESP. Slika 74 prikazuje format paketa v kombiniranem tunelu AH-ESP med dvema požarnima zidovoma IBM.



Slika 74: Kombinirani tunel AH-ESP

Primer 3: Varnost od točke do točke s podporo VPN

Ta primer je kombinacija primerov 1 in 2 ter od protokola IPSec ne zahteva dodatnih virov v primerjavi s primerom 2 (slika 73). Največja razlika v tem primeru je, da od gostiteljev zahteva podporo protokola IPSec.



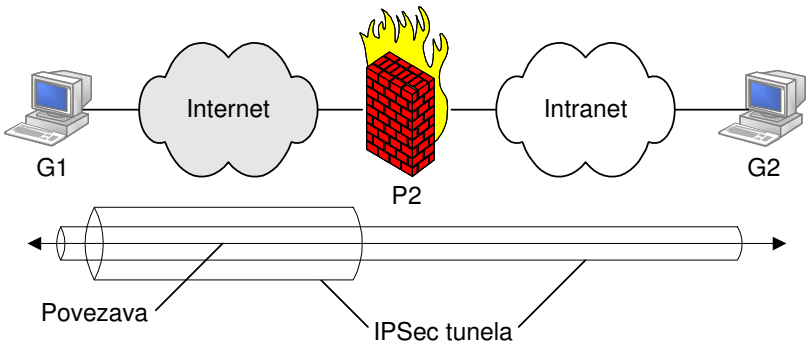
Slika 75: Varnost od točke do točke s podporo VPN

Običajno usmerjevalnika uporabljata varnostni protokol AH s tunelskim načinom delovanja, medtem ko gostitelja uporabljata protokol ESP v transportnem načinu.

Primer 4: Oddaljeni dostop

Oddaljeni dostop (angl. Remote Access) (slika 76) omogoča oddaljenim gostiteljem povezanih na spletno omrežje, dostop do strežnika v organizaciji zaščiteni s požarnim zidom. Oddaljeni gostitelj do ponudnika spletnih storitev običajno uporablja klicno povezavo PPP.

Med oddaljenim gostiteljem G1 in požarnim zidom P2 zahtevamo samo tunelski način delovanja. Izbire so enake kot v primeru 2. Med gostiteljema lahko uporabimo tunelski ali transportni način z istimi možnostmi kot v primeru 1.



Slika 76: Oddaljeni dostop

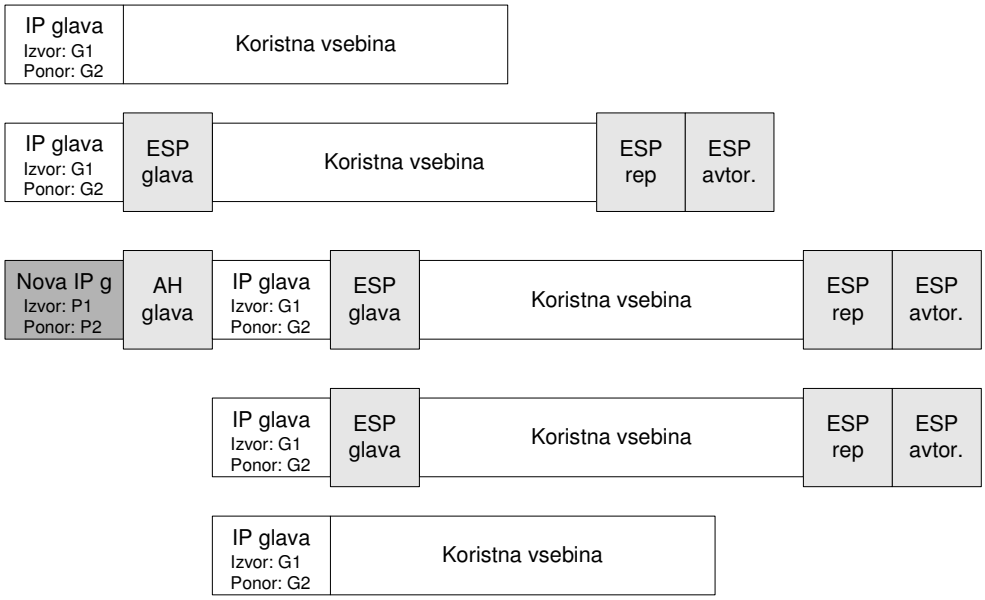
Običajno med gostiteljem G1 in požarnim zidom P2 uporabljamo protokol AH s tunelskim načinom delovanja ter med gostiteljem G1 in gostiteljem G2 pa protokol ESP v transportnem načinu. Prav tako običajno je kreiranje kombiniranega tunela AH-ESP med

oddaljenim gostiteljem G1 in požarnim zidom P2. V tem primeru lahko gostitelj G1 dostopa do celotnega spleta z uporabo enega samega svežnja SA. Če uporabimo nastavitve, prikazane na sliki 76, gostitelj G1 lahko dostopa samo do gostitelja z enim svežnjem SA.

Zaključek in primer

Čeprav kombinacije protokolov IPSec v teoriji vodijo do velikega števila možnosti, jih v praksi uporabljamo samo nekaj. Zelo pogosta je kombinacija protokola AH s tunelskim načinom delovanja, ki varuje promet protokola ESP v transportnem načinu. Zelo pogosti so med požarnimi zidovi tudi kombinirani tuneli AH-ESP.

Slika 77 detajlno prikazuje realizacijo prve kombinacije.



Slika 77: Gnezdenje protokolov IPSec

Predpostavimo, da gostitelj G1 s slike 77 pošilja paket IP gostitelju G2. Zgodi se naslednje:

- Gostitelj G1 tvori paket IP in na njem uporabi protokol ESP v transportnem načinu delovanja. Gostitelj G1 potem pošlje paket usmerjevalniku P1 z naslovom ponora G2.
- Usmerjevalnik P1 ugotovi, da je potrebno ta paket usmeriti do prehoda P2. Po pregledu podatkovnih baz IPSec (SPD in SAD), usmerjevalnik P1 zaključi, da mora, preden ga pošlje v omrežje, uporabiti protokol AH v tunelskem načinu. Usmerjevalnik P1 naredi zahtevano ovijanje. V tem trenutku dobi paket IP naslov svojega ponora P2, končni ponor G2 pa se začne ovijati.
- Usmerjevalnik P2 sprejme ovit paket IP z avtentikacijsko glavo. Njegov ponor kaže na samega sebe, tako da ga avtenticira in mu odreže zunanjo glavo.

Usmerjevalnik P2 opazi, da je koristna vsebina drugi IP paket (tisti, ki ga je poslal gostitelj G1) naslovljen na gostitelja G2, zato ga usmeri proti njemu. Usmerjevalnika P2 ne skrbi, da ta paket vsebuje tudi glavo ESP.

- Gostitelj G2 paket sprejme. Ker je ta hkrati tudi ponor, uporabimo ESP transportno procesiranje in dobimo originalni koristni tovor.

3.5.4 *Protokol Internetne izmenjave ključev*

Ogrodje protokola Internetne izmenjave ključev (angl. The Internet Key Exchange Protocol, krajše IKE), ki se je prvotno imenoval ISAKMP/Oakley, podpira avtomatska pogajanja SA, avtomatsko generiranje in osveževanje kriptografskih ključev. Sposobnost, kako avtomatizirati te funkcije, je eden od ključnih elementov vsakega večjega razvoja IPSec. Preden začnemo z detajlnim opisom izmenjave ključev in posodabljanja sporočil, potrebujemo naslednje definicije osnovnih pojmov:

Internetno združenje za varnost in protokol vodenja ključev (ISAKMP) (angl. Internet Security Association and Key Management Protocol)
: Krovna organizacija, ki definira vodenje združenj za varnost (pogajanja, spremembe, brisanje), ključev ter definicije koristnih tovorov pri generiranju in avtenticiranju izmenjave ključev. ISAKMP ne definira nobenega protokola izmenjave ključev, vendar specifikacije, ki jih določa, lahko uporabimo pri varnostnih mehanizmih v omrežjih, prenosu ali aplikacijskem nivoju.

Oakley
: Protokol izmenjave ključev, ki ga lahko uporabimo v specifikaciji ISAKMP za izmenjavo in posodobitev ključnih materialov SA.

Interpretacijsko področje (angl. Domain of Interpretation)
: Definicija množice protokolov, ki jih uporabljamo s specifikacijami ISAKMP v določenem okolju in množica skupnih definicij, s katerimi protokoli opisujejo sintakso atributov SA, vsebin prenosnih tovorov itd.

Internetna izmenjava ključev (angl. Internet Key Exchange)
: Protokol je mešanica ISAKMP, Oakley in SKEME protokolov izmenjave ključev in določa vodenje ključev, varnostnih protokolov AH in ESP ter ISAKMP samega.

Pregled protokola

Organizacija ISAKMP zahteva, da morajo biti vse izmenjave informacij šifrirane in avtenticirane tako, da ključnega materiala ne more nihče pregledovati in da ga izmenjujemo samo z avtenticiranimi partnerji. Metode ISAKMP razvijamo z jasnim ciljem zaščite pred naslednjimi splošno znanimi nevarnostmi:

- *zavrnitev storitve* (angl. Denial-of-Service): ta sporočila tvorimo z unikatnimi piškotki (angl. cookies), ki jih lahko uporabimo za hitro identificiranje in zavračanje nepravilnih sporočil, ne da bi pri tem izvajali procesorsko intenzivne šifrirne operacije.

- *vmesni človek* (angl. Man-in-the-Middle): omogoča zaščito proti splošnim napadom tako kot brisanjem sporočil, spremembam sporočil, vračanje sporočil pošiljatelju, odgovarjanje na stara sporočila in preusmerjanje sporočil nenadejanim naslovnikom.
- *popolna tajnost dostave* (angl. Perfect Forward Secrecy, krajše PFS): sporazum o predhodnih ključih ne določa uporabnega pravila izklopa vsakega ključa, ki nastopa pred ali po sporazumnem ključu, t.j. vsak osveženi ključ izpeljemo neodvisno od predhodnih ključev.

Pri IKE definiramo naslednje avtentikacijske metode:

- skupni ključ,
- digitalni podpis (DSS ali RSA),
- šifrirani javni ključ (RSA in popravljen RSA).

Moč vsake kriptografske rešitve je odvisna veliko bolj od vprašanja, kako držati ključe v tajnosti, kot od tega, kako dober kriptografski algoritem izberemo. Zaradi tega je delovna skupina IETF IPSec predpisala množico zelo močnih protokolov izmenjave Oakley. Uporabljajo dvofazni pristop:

Faza 1: ta množica pogajanj vzpostavlja krovno varnost iz katere bomo zaporedoma izpeljevali vse kriptografske ključe za zaščito uporabniškega podatkovnega prometa. V bolj splošnem primeru kriptiran javni ključ uporabljamo pri vzpostavitvi SA ISAKMP med sistemi in pri vzpostavitvi ključev potrebnih za zaščito sporočil ISAKMP, ki bodo tekla v naslednji fazi 2 pogajanj. Faza 1 skrbi samo za vzpostavitev določene vrste varnosti za sporočila ISAKMP samih, ne pa tudi za vzpostavljanje kakršnekoli SA ali ključev za zaščito uporabniških podatkov.

Kriptografske operacije faze 1 so najbolj procesorsko intenzivne, vendar jih delamo zelo redko, t.j. eno izmenjavo faze 1 lahko uporabimo za podporo več zaporednih izmenjav faze 2. Pogajanja faze 1 tečejo enkrat dnevno ali morda enkrat mesečno, medtem ko pogajanja faze 2 tečejo vsakih nekaj minut.

Faza 2: izmenjave faze 2 so manj kompleksne, ker jih uporabljamo samo v primerih, ko določene vrste varnosti, za katere smo se dogovorili že v fazi 1, aktiviramo. Množica komunikacijskih sistemov se pogaja za SA in ključe, s katerimi bo zaščitila izmenjavo uporabniških podatkov. Sporočila ISAKMP faze 2 zaščitimo z SA ISAKMP generirano v fazi 1. Pogajanja faze 2 v splošnem potekajo bolj pogosto kot pogajanja faze 1. Primer: tipična aplikacija pogajanj faze 2 je osvežitev kriptografskih ključev vsake dve tri minute.

Stalni identifikatorji (angl. Permanent Identifiers): protokol IKE nudi rešitev tudi, ko IP naslov oddaljenega gostitelja ne poznamo vnaprej. ISAKMP omogoča oddaljenemu gostitelju, da se identificira s stalnim identifikatorjem, npr. imenom ali E-poštni naslov,

kar sam. Po tej identifikaciji faza 1 izmenjav ISAKMP avtenticira stalni identifikator oddaljenega gostitelja z uporabo javnega kriptografskega ključa:

1. certifikat tvori vez med stalnim identifikatorjem in javnim ključem. Na certifikatu temelječe izmenjave sporočil faze 1 ISAKMP lahko avtenticirajo stalno identiteto oddaljenega gostitelja.
2. ker IP datagrami nosijo sporočila ISAKMP, lahko partner ISAKMP (npr. požarni zid ali ponorni gostitelj) pridruži dinamični IP naslov oddaljenega gostitelja z njegovo avtenticirano stalno identiteto.

3.6 Plast varnih vtičnic

Plast varnih vtičnic (angl. Secure Socket Layer, krajše SSL) je varnostni protokol, ki ga je razvil Netscape Communications Corporation skupaj z RSA Data Security, Inc. Osnovni cilj protokola SSL je med komunikacijskimi aplikacijami določiti zasebni kanal, ki zagotavlja avtentikacijo, diskretnost in integriteto podatkov.

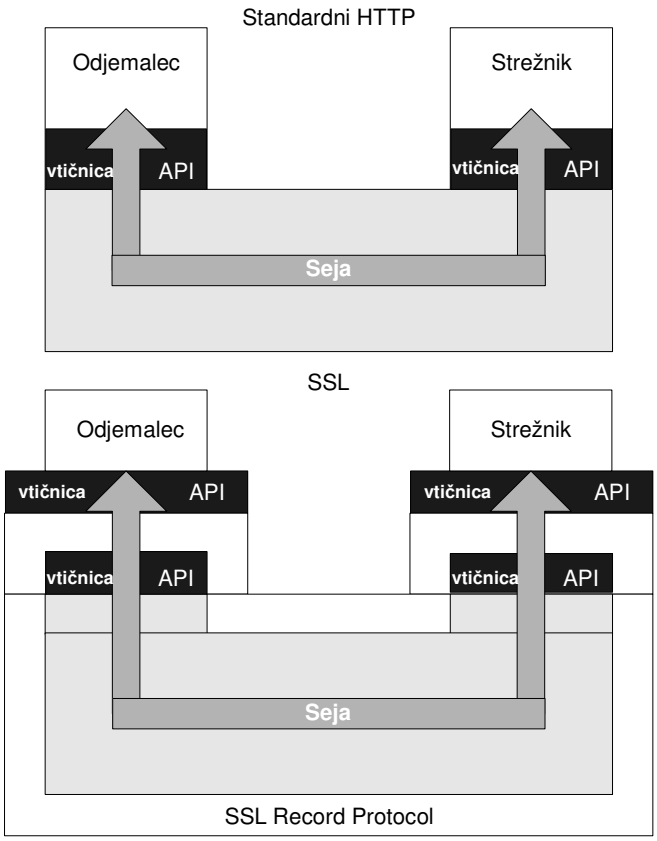
Protokol SSL določa alternativo standardnim vtičnicam TCP/IP (angl. socket API) z implementirano varnostjo. Teoretično lahko vsako aplikacijo TCP/IP zaganjamo na varen način, ne da bi jo pri tem spreminjali. Praktično protokol SSL najpogosteje uporabljamo za aplikacije, ki uporabljajo protokol HTTP, čeprav ga njegovi razvijalci želijo uporabiti tudi za druge aplikacije, kot npr. NNTP in Telnet, za katere obstaja že več brezplačnih implementacij na spletu. IBM uporablja protokol SSL za povečanje varnosti sej TN3270 v njihovih produktih Host On-Demand in eNetwork Communication Server.

Protokol SSL je sestavljen iz dveh plasti:

- na spodnji plasti, t.j. protokolu za prenos podatkov, uporablja različne predefinirane kriptografske in avtentikacijske kombinacije, ki jih s skupnim imenom imenujemo tudi *SSL Record Protocol*. Slika 78 prikazuje primerjavo protokola SSL s standardno povezavo HTTP z vtičnicami. Iz slike 78 lahko razberemo, da protokol SSL določa enostaven vmesnik z vtičnico, na katerem lahko ležijo tudi druge aplikacije. Trenutne implementacije v praksi imajo vmesnik z vtičnico vstavljen znotraj aplikacije, ki ga druge aplikacije ne morejo uporabljati.
- na višji plasti uporablja avtentikacijo in prenos kriptografskih ključev, ki jo s skupnim imenom imenujemo tudi *SSL Handshake Protokol*.

Začetek seje SSL zahteva naslednje faze:

- na odjemalcu (brskalniku) uporabnik zahteva dokument s posebnim URL, ki namesto protokola *http* požene protokol *https*.
- koda na odjemalcu razpozna zahtevo SSL in vzpostavi povezavo preko vrat TCP številka 443 s strežnikom SSL.
- odjemalec nato začne fazo usklajevanja s strežnikom SSL z uporabo *SSL Record Protocol* kot nosilcem. V tej točki povezava še ni šifrirana in ne avtenticirana.



Slika 78: Primerjava med standardnimi sejami in sejami SSL

Protokol SSL uporablja naslednje varnostne mehanizme:

Zasebnost (angl. Privacy)
: Pri začetnem usklajevanju zahtev zamenjamo simetrični ključ, s katerim šifriramo sporočila v nadaljevanju seje.

Integriteto (angl. Integrity)
: Sporočila vsebujejo avtentikacijsko kodo MAC, ki zagotavlja njegovo integriteto.

Avtentikacijo (angl. Authentication)
: Med usklajevanjem zahtev se odjemalec z uporabo asimetričnega ali javnega ključa avtenticira na strežniku. Ta avtentikacija lahko temelji tudi na certifikatih.

Protokol SSL zahteva, da vsako sporočilo šifriramo ali dešifriramo, kar ima lahko velik vpliv na zmogljivost sistema.

Razlike med protokoloma SSL V2.0 in SSL V3.0

Med obema protokoloma velja kompatibilnost navzdol, t.j. implementacija strežniškega protokola SSL V3.0 mora obravnavati tudi zahteve odjemalca s protokolom SSL V2.0. Glavne razlike med obema protokoloma pa so naslednje:

- protokol SSL V2.0 ne podpira avtentikacije odjemalcev,
- protokol SSL V3.0 podpira več kriptografskih algoritmov.

3.6.1 *Protokol SSL*

Protokol SSL se nahaja na vrhu transportne plasti. SSL je tudi sam plastni protokol, ki vzame podatke iz aplikacijske plasti, jih predela in pošlje v transportno plast. Sporočila obdeluje na naslednji način:

Pošiljatelj (angl. Sender)
- vzame sporočilo z višje plasti,
- fragmentira podatke v bloke, primerne za prenos po mreži,
- opcijsko stisne podatke,
- uporabi avtentikacijsko kodo MAC,
- šifrira podatke,
- rezultate prenese na spodnjo plast.

Prejemnik (angl. Receiver)
- vzame podatke s spodnje plasti,
- podatke dešifrira,
- preveri podatke z dogovorjenim ključem MAC,
- opcijsko raz-pakira podatke,
- ponovno zbere sporočilo,
- pošlje sporočilo na višjo plast.

Seja SSL lahko deluje v dveh različnih stanjih: stanju seje ali stanju povezave. Protokol usklajevanje zahtev SSL usklajuje stanja odjemalca in strežnika. Za usklajevanje šifriranja definiramo tudi stanja čitanj in pisanj. Ko eden izmed partnerjev v komunikaciji pošlje sporočilo *spremeni šifrirni algoritem*, spremeni stanje pisanja, ki je v teku, v trenutno stanje. Stanje seje vključuje naslednje komponente:

Identifikator seje (angl. Session Identifier)
: Poljuben bajt, ki ga izbere strežnik za identifikacijo aktivnega stanja seje.

Certifikat vrstnika (angl. Peer certificate)
: Certifikat vrstnika v paru. To polje je opcijsko in je lahko prazno.

Metoda stiskanja (angl. Compression Method)
: Algoritem stiskanja.

Šifrirni algoritem (angl. Cipher spec)
: Določa podatkovni enkripcijski algoritem (npr. 3DES) in algoritem MAC. (Cipher = skrivno pravilo za pretvorbo nekega sporočila v zaporedje znakov)

Glavna skrivna šifra (angl. Master Secret)
: 48-bajtna skrivna šifra, ki si jo delita odjemalec in strežnik med seboj.

Nadaljuj (angl. is resumable)
: Zastavica, ki določa, če sejo lahko uporabimo za nove povezave.

Stanje poveze vključuje naslednje komponente:

Strežnik in odjemalec naključno (angl. Server and client random)
Poljuben bajt, ki ga za vsako povezavo izbereta odjemalec in strežnik.
Strežniška skrivna šifra MAC za pisanje (angl. Server write MAC secret)
Skrivna šifra, ki jo uporablja strežnik pri operacijah MAC.
Odjemalna skrivna šifra MAC za pisanje (angl. Client write MAC secret)
Skrivna šifra, ki jo uporablja odjemalec pri operacijah MAC.
Strežniški ključ za pisanje (angl. Server write key)
Šifrirani ključ strežnika za šifriranje podatkov in odjemalca za dešifriranje teh.
Odjemalni ključ za pisanja (angl. Client write key)
Šifrirani ključ odjemalca za šifriranje podatkov in strežnika za dešifriranje teh.
Začetni vektorji (angl. Initialization vectors)
Začetni vektorji hranijo šifrirne informacije.
Zaporedne številke (angl. Sequence numbers)
Zaporedna številka določa številko sporočila poslanega od zadnje spremembe šifrirnega algoritma. Ta števila vzdržujeta tako odjemalec kot strežnik.

Protokol zamenjave šifrirnega algoritma

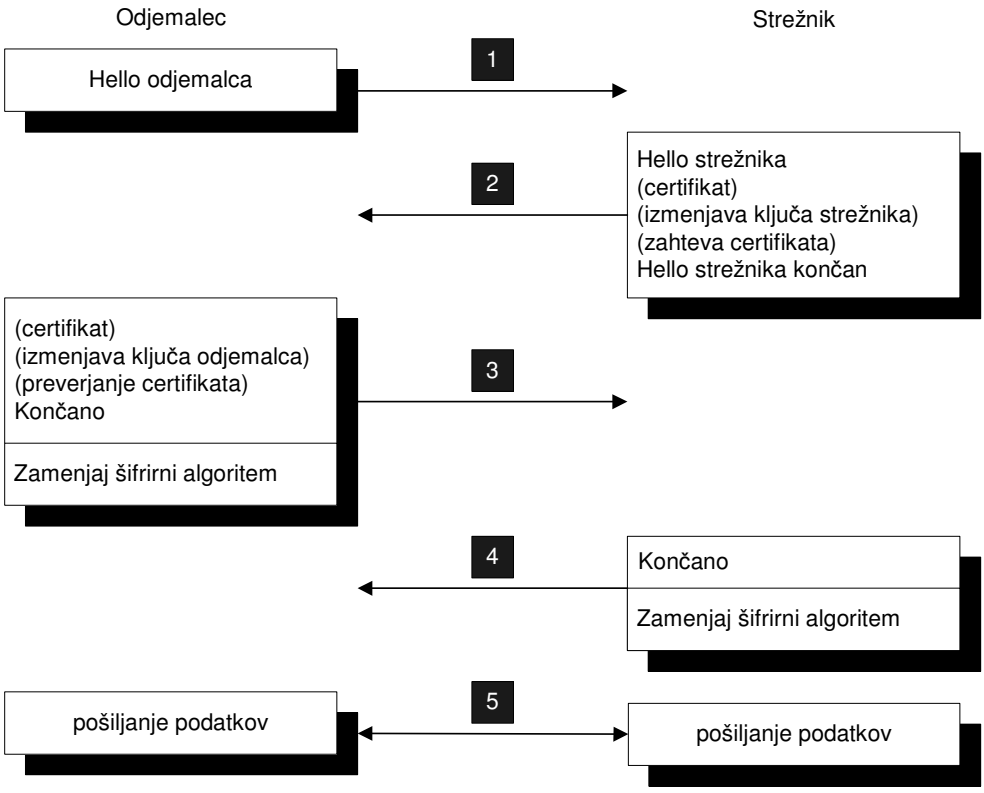
Protokol zamenjave šifrirnega algoritma je odgovoren za pošiljanje sporočil *zamenjaj šifrirni algoritem*. Odjemalec lahko kadarkoli zahteva izmenjavo trenutnih kriptografskih parametrov, kot npr. *uskladi izmenjavo ključa*. V skladu s potrditvijo sporočila *zamenjaj šifrirni algoritem*, odjemalec pošlje sporočilo *uskladi izmenjavo ključa* in če je na voljo, pošlje sporočilo *preveri certifikat.* Na to sporočilo po procesiranju sporočila *izmenjaj ključ* strežnik pošlje odjemalcu sporočilo *zamenjaj šifrirni algoritem*. Novi ključi se bodo uporabljali vse do naslednje zahteve *zamenjaj šifrirni algoritem*.

Protokol usklajevanja SSL

Protokol usklajevanja SSL omogoča odjemalcu in strežniku določiti potrebne parametre za vzpostavitev povezave SSL kot npr. verzijo protokola, kriptografske algoritme, opcijsko avtentikacijo odjemalca ali strežnika in šifrirne metode javnih ključev za generiranje deljenih šifer. Med tem procesom vsa usklajevalna sporočila pošiljamo v plast *SSL Record Layer*, kjer jih ovijemo v posebna sporočila SSL (slika 79).

1. Odjemalec pošlje zahtevo za povezavo s sporočilom *Hello*. To sporočilo vključuje:
    - Verzijo programa,
    - Časovne informacije: tekoči čas in datum v standardnem 32-bitnem formatu UNIX,
    - Opcijsko ID-seje: če ga ne določimo bo strežnik poskušal nadaljevati s predhodno sejo ali vrnil kodo napake,

- Sveženj šifrirnih pravil: seznam kriptografskih opcij, ki jih odjemalec podpira. To so način avtentikacije, metode izmenjave ključev, enkripcijski algoritmi in MAC algoritmi.
- Metode stiskanja, ki jih podpira odjemalec.
- naključna vrednost.



Slika 79: Proces usklajevanja SSL med odjemalcem in strežnikom

2. Strežnik oceni parametre, ki jih pošlje odjemalec v sporočilu *Hello* in vrne odgovor *Hello* z naslednjimi parametri:
   - Verzijo programa.
   - Časovne informacije: tekoči čas in datum v standardnem 32-bitnem formatu UNIX.
   - ID-seje.
   - Sveženj šifrirnih pravil.
   - Metode stiskanja.
   - Naključna vrednost.

   Strežnik po sporočilu *Hello* pošlje še naslednja sporočila:

   - strežnikov certifikat, če se mora avtenticirati,
   - sporočilo izmenjava ključev, če certifikatov ni na voljo,
   - zahteva certifikat, če se mora odjemalec avtenticirati.

3. Odjemalec pošlje naslednja sporočila:
    - Če je strežnik zahteval certifikat, ga mora odjemalec poslati ali odgovoriti s sporočilom *ni certifikata*.
    - Če je strežnik poslal sporočilo izmenjava ključev, odjemalec pošlje sporočilo izmenjave ključev, ki temelji na algoritmu javnih ključev določenim s sporočili *Hello*.
    - Če mora odjemalec poslati certifikat, preveri certifikat strežnika in pošlje sporočilo preveri certifikat, ki pokaže rezultat.

    Odjemalec nato pošlje sporočilo končano, ki pokaže, da se je pogajalski del končal. Odjemalec za generiranje skrivne šifre pošlje tudi sporočilo spremeni šifrirni algoritem.

4. strežnik pošlje sporočilo končano, ki nakaže, da se je pogajalski del končal. Strežnik nato pošlje sporočilo spremeni šifrirni algoritem.

5. Končno oba partnerja seje ločeno generirata enkripcijski ključ, t.j. glavni ključ iz katerega izpeljujemo ključe, ki jih uporabljamo v šifrirni seji. Protokol usklajevanja SSL stanju povezave spremeni stanje. Vse podatke, ki jih dobimo iz aplikacijske plasti, pošljemo kot posebna sporočila drugemu partnerju.

Pri proženju seje SSL v primerjavi z normalno povezavo HTTP nastanejo dodatni stroški. Protokol SSL se nekaterim stroškom izogne tako, da dovoli odjemalcu in strežniku obdržati sejne informacije ključev in nadaljuje to sejo, ne da bi se znova pogajal ali avtenticiral. Oba partnerja seje po končanem usklajevanju generirata glavni ključ. Iz tega ključa generirata ostale ključe seje, ki jih uporabljata za šifriranje podatkov seje s simetričnim ključem. Prvo sporočilo šifrirano na ta način je zadnje sporočilo poslano od strežnika. Če ga lahko odjemalec interpretira, pomeni to, da smo:

- Dosegli *zasebnost*, saj je sporočilo kriptirano s šifrirnim algoritmom, ki za kriptiranje uporablja simetrične ključe (npr. DES, 3DES).
- Zagotovili *integriteto sporočila*, saj to vsebuje avtentikacijsko kodo MAC, ki je izvleček tako sporočila samega kot tudi vsebine izpeljane iz glavnega ključa.
- *Avtenticirali* strežnik, saj je iz osnovnega glavnega ključa izpeljal glavni ključ. Ker smo ga poslali z uporabo strežniškega javnega ključa, ga je lahko dešifrira samo strežnik (z uporabo privatnega ključa).

Protokol zapisov SSL

Ko odjemalec in strežnik določita glavni ključ, ga lahko uporabljata za šifriranje podatkov. Protokol zapisov SSL določa format teh sporočil. Običajno vključuje izvleček sporočila MAC, ki zagotavlja, da se sporočilo na poti do sprejemnika ni spremenilo. Sporočilo dodatno šifrira z uporabo simetričnih šifer. Pogosto uporablja algoritem RC2 ali RC4, čeprav specifikacija omogoča podporo tudi za algoritme DES, 3DES in IDEA.

Agencija ameriške zvezne vlade NSA (agencija za nacionalno varnost, angl. National Security Agency) določa omejitve velikosti kriptografskega ključa, ki ga uporabljajo v programski opremi izvoženi zunaj ZDA. Danes uporabljamo 128-bitne ključe.

3.7 Secure Multipurpose Internet Mail Extension

S-MIME (krajše S-MIME) lahko obravnavamo kot zelo specifični protokol, ki je podoben SSL. Protokol S-MIME je varnostni konstrukt na aplikacijskem nivoju, vendar je njegova uporaba omejena zaščiti elektronske pošte prek šifriranja in digitalnih podpisov. Sloni na tehnologiji zasebnih ključev in uporablja certifikate X.509 za vzpostavljanje identitet komunikacijskih partnerjev. Protokol S-MIME lahko implementiramo v komunikacijskih končnih točkah, t.j. vmesni usmerjevalniki in požarni zidovi ga ne uporabljajo.

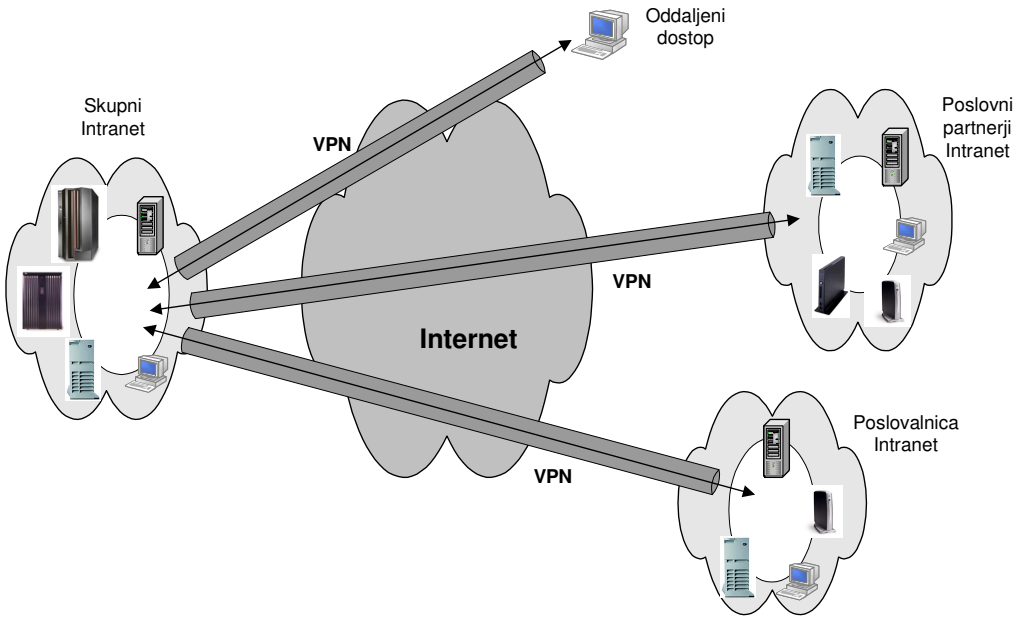
3.8 Navidezno zasebno omrežje

Splet postaja popularna in poceni hrbtenična infrastruktura. Njena univerzalna širina je veliko podjetij prisilila, da zgradijo varna navidezna zasebna omrežja (angl. Virtual Private Networks, krajše VPN) prek javnega spletnega omrežja. Izziv pri načrtovanju VPN za današnjo globalno podjetniško okolje je, kako izkoriščati javno spletno hrbtenico za komunikacije znotraj in zunaj podjetja ter hkrati zagotoviti varnost tradicionalnih zasebnih omrežij.

3.8.1 *Uvod v VPN in praktična uporabnost*

Z eksplozivno rastjo spletnega omrežja so se podjetja začela spraševati, kako je mogoče najboljše izkoristi splet v poslovne namene. V začetku so podjetja z omogočanjem dostopa do skupnih spletnih strani uporabljala spletno omrežje za reklamiranje svojih produktov in storitev. Današnji potencial spletnega omrežja je neskončen in podjetja so se osredotočila na elektronsko poslovanje, ki uporablja globalno razsežnost svetovnega spleta za enostaven dostop do ključnih poslovnih aplikacij in podatkov. Te aplikacije in podatki se nahajajo običajno v tradicionalnih informacijskih sistemih. Podjetja lahko sedaj varno in poceni razširijo dostopnost svojih aplikacij in podatkov prek sveta s pomočjo implementacije VPN.

VPN je razširitev podjetniškega zasebnega intraneta prek javnih omrežij kot je splet, gradnje varnih zasebnih povezav, posebej skozi zasebne tunele. VPN omogoča varno sporočanje informacij prek spleta s povezovanjem oddaljenih uporabnikov, poslovalnic in poslovnih partnerjev v razširjeno skupno privatno omrežje (slika 80). Ponudniki spletnih storitev (angl. Internet Service Providers, krajše ISP) ponujajo poceni dostop do spletnega omrežja (prek neposrednih vodov ali lokalnih telefonskih številk), podjetjem omogočajo odstranitev njihovih trenutnih, dragih najetih vodov in oddaljenih dostopov prek klicnih linij.



Slika 80: Navidezno zasebno omrežje

Družba Infonetics Resaerch, Inc. je v *VPN Research Report* leta 1997 ocenila prihranke pri zamenjavi najetih vodov WAN omrežij z omrežjem VPN na 20% do 47%. Prihranki zamenjav oddaljenih klicnih povezav uporabnikov z odjemalci VPN so lahko od 60% do 80%. Dostop do spletnega omrežja je na voljo širom po svetu.

3.8.2 *Rešitve VPN na tržišču*

Prodajalce ponudb VPN lahko razdelimo na številne načine. Najpomembnejša je po našem mnenju razlika v protokolni plasti, na kateri VPN realiziramo. V tem smislu lahko različne pristope k implementaciji VPN razdelimo na:

- omrežni plasti bazirana omrežja VPN (osnova IPSec),
- povezavni plasti bazirana omrežja VPN (osnova L2TP).

Danes obstajajo še druge metode, ki operirajo na višjih protokolnih plasteh in dopolnjujejo rešitve VPN, npr. *socks*, SSL ali S-MIME. Nekatere rešitve na tržišču za konstrukcijo omrežij VPN uporabljajo samo višje protokolne plasti pogosto v kombinaciji s *socks* in SSL.

3.9 Avtentikacijski protokoli za oddaljeni dostop

Strežnik za oddaljeni dostop (angl. Remote Access Authentication Protocols, krajše RAS) je postal danes eden od najvitalnejših delov današnjega spletnega omrežja. Vse bolj mobilni uporabniki zahtevajo dostop ne samo do centralnih računalniških virov, ampak tudi informacijskih virov na spletu. Široka uporaba spleta in skupnega intraneta je

pospešila rast storitev za oddaljeni dostop in komunikacijskih naprav za njihovo podporo. Danes se povečujejo zahteve po enostavnejšem povezovanju do virov skupnega omrežja iz mobilnih računalniških naprav, kot npr. prenosnih računalnikov.

Oddaljeni dostop je povzročil pomemben razvoj na področju omrežne varnosti. V industriji je v ta namen nastal varnostni model AAA (angl. triple A), t.j. avtentkacija, avtorizacija in obračun (angl. accounting), ki odgovarjajo na vprašanja kdo, kaj in koliko časa. Sledi kratek opis varnostnega modela trojni A:

Avtentikacija, overjanje (angl. authentication)
: To je akcija, kjer določamo, *kdo* je uporabnik. Poznamo več vrst avtentikacije. Tradicionalna avtentikacija uporablja ime in fiksno geslo. Večina dostopov do računalnikov danes deluje na ta način. Fiksna gesla pa imajo omejitve, saj večina sodobnih avtentikacijskih mehanizmov uporablja enkratna gesla ali vprašanje geslo-odgovor.

Avtorizacija, pooblastitev (angl. authorization)
: To je akcija, kjer določamo, *kaj* uporabniku dovolimo delati na sistemu. Avtorizacijska zahteva običajno dokazuje, da se uporabnik uspešno avtenticiral. V primeru, da avtorizacija nastopi pred avtentikacijo, je odvisno od avtorizacijskega agenta ali dovoli neavtenticiranemu uporabniku izvesti storitev ali ne.

Obračun (angl. accounting)
: Običajno je to tretja akcija, ki sledi avtentikaciji in avtorizaciji, kjer zabeležimo, kdaj uporabnik dela in kaj je ta na sistemu počel.

V porazdeljenem varnostnem modelu odjemalec/strežnik s podatkovno bazo številni komunikacijski strežniki ali odjemalci avtenticirajo oddaljene uporabnike prek enotne, centralne podatkovne baze ali avtentikacijskega strežnika. Avtentikacijski strežnik shrani vse informacije o uporabnikih, njihovih geslih in dostopnih pravicah. Porazdeljena varnost določa centralno lokacijo za avtentikacijske podatke, kar je bolj varno, kot sejanje uporabniških informacij po različnih napravah v omrežju. En avtentikacijski strežnik lahko podpira stotine komunikacijskih strežnikov in zadovolji tudi do 10.000 uporabnikov.

Več prodajalcev strežnikov za oddaljeni dostop in združenje IETF (angl. Internet Engineering Task Force) so združili moči, da bi varnost za oddaljeni dostop standardizirali. Rezultata tega smelega podviga je nastanek protokolov RADIUS (angl. Remote Authentication Dial In User Service) in TACACS (angl. Terminal Access Controller Access Control System). Podobno kot RADIUS, tudi TACACS sprejme avtentikacijsko zahtevo od odjemalca NAS (angl. Network Access Server) in posreduje uporabniško ime in geslo do centralnega varnostnega strežnika. Centralni varnostni strežnik lahko poseduje bodisi interno podatkovno bazo TACACS ali ima dostop do zunanje podatkovne baze uporabnikov, npr. LDAP (angl. Lightweight Directory Access Protocol) oz. AD (angl. Active Directory). TACACS+ je razširitev podjetja Cisco, ki različnim strežnikom za dostop (strežnikom TACACS+) omogoča neodvisne avtentikacijske, avtorizacijske in obračunske storitve.

Čeprav RADIUS in TACACS avtentikacijske strežnike, v odvisnosti od varnostne sheme omrežje, ki ga zadovoljuje, lahko postavimo na različne načine, pa so osnovni procesi za avtentikacijo uporabnikov v bistvu enaki. Oddaljeni dial-in uporabnik se prek modema poveže na strežnik za oddaljeni dostop NAS z vgrajenim analognim ali digitalnim modemom. Ko je zveza vzpostavljena, NAS zahteva od uporabnika ime in geslo. NAS iz predanih podatkov tvori t.i. avtentikacijsko zahtevo. Ta je sestavljena iz informacij za identifikacijo določene naprave NAS, ki pošilja avtentikacijsko zahtevo, vrata za modemsko povezavo ter ime uporabnika in njegovo geslo.

Za zaščito pred napadalci NAS v vlogi varnostnega strežnika RADIUS oz. TACACS šifrira gesla, preden jih pošlje avtentikacijskemu strežniku. Če primarni avtentikacijski strežnik ni dosegljiv, lahko varnostni odjemalec usmeri zahtevo do alternativnega avtentikacijskega strežnika. Ko avtentikacijski strežnik sprejme zahtevo, jo preveri in dešifrira podatkovni paket, da lahko pride do informacij o uporabniškem imenu in geslu. Če sta uporabniško ime in geslo pravilni, strežnik pošlje paket o potrditvi uspešne avtentikacije. Ta paket lahko vsebuje dodatne filtre, npr. informacije o uporabnikovih potrebah po omrežnih virih in nivo avtorizacije. Avtentikacijski strežnik lahko, npr. informira NAS, da uporabnik na liniji PPP za povezavo na omrežje potrebuje protokol TCP/IP, IPX oz. SLIP. Prav tako lahko vključi informacije o določenih omrežnih virih, do katerih ima uporabnik dostop.

Za preprečitev napadov na omrežje varnostni strežnik pošlje avtentikacijski ključ ali podpis, s katerim identificira varnega odjemalca. Ko NAS sprejme te informacije, ustrezno konfigurira omrežne naprave in uporabniku tako omogoči pravice za dostop do omrežnih storitev in virov. Če na poljubni točki tega prijavnega procesa niso izpolnjeni vsi potrebni avtentikacijski pogoji, strežnik z varnostno podatkovno bazo pošlje varnostnemu strežniku NAS sporočilo *avtentikacija zavrnjena* in uporabniku prepove dostop do omrežja..

3.10 Tunelski protokol na drugi plasti

Tunelski protokol na drugi plasti (angl. Layer 2 Tunneling Protocol, krajše L2TP) omogoča protokol PPP v tunelskem načinu delovanja, t.j. vsak protokol, ki ga podpira PPP, lahko uporabimo v tunelskem načinu. Ta protokol širi uporabnost povezave PPP. Namesto povezave z začetkom na oddaljenem gostitelju in koncem pri ponudniku spletnih rešitev se navidezna povezava PPP sedaj razširi od oddaljenega gostitelja pa do skupnega lokalnega prehoda, t.j. usmerjevalnika v lokalnem omrežju. Protokol L2TP trenutno podpira protokol UDP.

L2TP je splošno sprejeti standard, ki je nastal z združitvijo dveh obstoječih protokolov v tunelskem načinu: tunelskega protokola od točke do točke (angl. Point-to-Point Tunneling Protocol, krajše PPTP) in posredovanja na drugi plasti (angl. Layer 2 Forwarding, krajše L2F). Oba obstoječa protokola ne omogočata tako popolne rešitve kot protokol L2TP, saj lahko s prvim rešujemo tunele, ki jih kreirajo ponudniki spletnih storitev in z drugim tunele, ki jih kreirajo oddaljeni gostitelji. Protokol L2TP podpira oba,

t.j. tunele, ki jih kreirajo oddaljeni gostitelji in tunele, ki jih kreirajo ponudniki spletnih storitev.

Protokol L2TP omogoča kreiranje virtualnih zasebnih omrežij, kjer dopuščamo več protokolov in zasebnih omrežij IP in IPX. Oddaljenim uporabnikom omogoča priključitev do lokalnega ponudnika spletnih storitev in tunel prek spletnega do domačega omrežja.

3.10.1 *Terminologija*

Preden opišemo protokol, potrebujemo definicije nekaterih pojmov iz terminologije L2TP.

Dostopni koncentrator (angl. Access Concentrator, krajše LAC)
: Naprava priključena na eno ali več linij javnega telefonskega omrežja (angl. Public Service Telephone Network, krajše PSTN) ali integriranega digitalnega omrežja (angl. Integrated Services Digital Network, krajše ISDN), ki omogoča tako operacije PPP kot tudi protokola L2TP. LAC je implementacija medija preko katerega L2TP operira. L2TP pošlje promet enemu ali več strežnikom L2TP (LNS).

Omrežni strežnik L2TP (angl. L2TP Network Server, krajše LNS)
: LNS operira na vsaki platformi na katero je priključena končna postaja PPP. LNS obravnava strežniško stran protokola L2TP. LNS ima lahko samo en LAN ali WAN vmesnik na katerem zaključuje kakršne koli vmesnike, ki jih LAC podpira, t.j. asinhrone, sinhrone, ISDN, V.120 itd.

Omrežni dostopni strežnik (angl. Network Access Server, krajše NAS)
: Naprava, ki omogoča začasni dostop do omrežja (dostop do omrežja na zahtevo). Ta dostop je od točke do točke in uporablja linije PSTN oz. ISDN.

Seja ali klic (angl. Session or Call)
: Protokol L2TP kreira sejo pri vzpostavljanju povezave PPP od točke do točke med klicnim uporabnikom in LNS. Datagrame za to sejo pošiljamo preko tunela med LAC in LNS. LNS in LAC vzdržujeta informacije stanja za vsakega uporabnika priključenega na LAC.

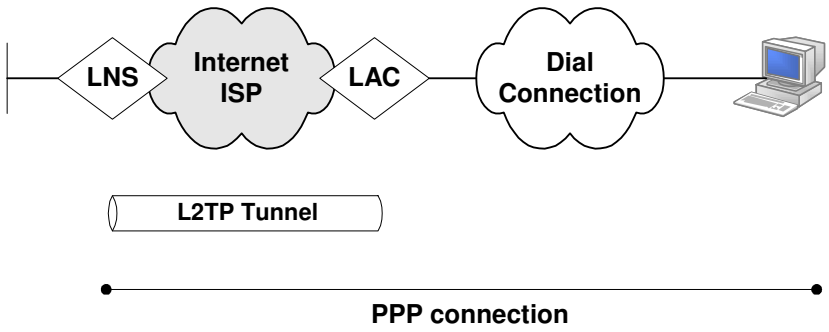
Tunel (angl. Tunnel)
: Tunel definira par LNS-LAC. Tunel nosi datagrame PPP med LAC in LNS. En tunel lahko multipleksira več sej. Nadzor povezave deluje na istem tunelu in nadzoruje vzpostavljanje, sproščanje in vzdrževanje vseh sej in tunela samega.

Par attribute-vrednost (Attribute Value Pair, krajše AVP)
: Uniformna metoda za kodiranje tipov in vsebine sporočil.

3.10.2 *Značilnosti protokola*

Ker gostitelj in prehod delita iste povezave PPP, lahko izkoristimo možnosti protokola PPP za prenos ne samo protokola IP, temveč tudi drugih protokolov.

Primer: tunele L2TP lahko uporabimo tako za podporo oddaljenih dostopov na LAN, kakor tudi oddaljenega dostopa IP (slika 82).

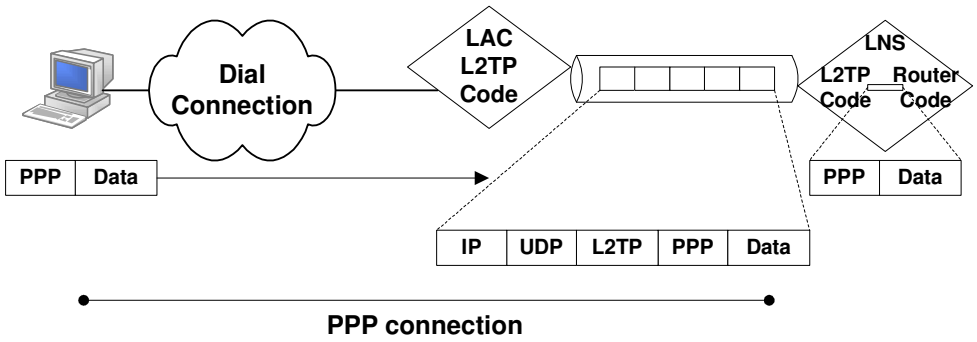


Slika 82: implementacija L2TP

Iz slike 82 razberemo, da:

- Oddaljeni uporabnik začne vzpostavljati povezavo PPP.
- NAS klic sprejme.
- NAS identificira oddaljenega uporabnika z uporabo avtorizacijskega strežnika.
- Če je avtorizacija uspela, NAS/LAC začneta vzpostavljati tunel L2TP do želenega LNS kot vstopa v podjetje.
- LNS s pomočjo avtentikacijskega strežnika avtenticira oddaljenega uporabnika in sprejme tunel.
- LNS potrdi sprejem klica in tunela L2TP.
- NAS sprejem zabeleži.
- LNS se z oddaljenim uporabnikom dogovori za pravila PPP.
- Podatke od točke do točke med oddaljenim uporabnikom in LNS prenašamo v tunelskem načinu.

L2TP lahko obravnavamo tudi kot na inačico enkapsulacijskega protokola IP. Iz slike 83 lahko razberemo, da tunel L2TP kreiramo z enkapsulacijo okvira L2TP v paket UDP, ki ga ponovno enkapsuliramo v paket IP. Naslov izvora in ponora v paketu IP definirata končni točki tunela. Ker je zunanji enkapsulacijski protokol IP, lahko uporabimo na njem protokole IPSec in s tem zaščitimo podatke, ki tečejo skozi tunel L2TP.



Slika 83: Spremembe paketa L2TP med prenosom

Protokol L2TP lahko deluje preko UDP/IP in podpira naslednje funkcije:

- tunelski način delovanja posameznih klicnih uporabnikov,
- tunelski način delovanja manjših usmerjevalnikov, npr. usmerjevalnik z enim statičnim prehodom, ki temelji na avtenticiranem uporabniškem profilu,
- vhodni klici do LNS od LAC,
- večkratni klici na tunel,
- avtentikacija proxy za protokola PAP in CHAP,
- proxy LCP,
- avtentikacija končne točke tunela,
- tunelski način delovanja z uporabo uporabniškega imena PPP, ki ga preverja sistem AAA.

3.10.3 *Varnostni problemi L2TP*

Čeprav protokol L2TP določa poceni dostop, več-protokolni prenos in oddaljeni dostop do lokalnega omrežja, ne omogoča kriptografskih varnostnih mehanizmov:

- Avtentikacija je določena samo za identifikacijo končnih točk tunela, na pa tudi za vsak posamezni paket, ki teče skozi tunel. Tunel je tako izpostavljen napadom tipa *mož-v-sredini* in *spoofing*.
- Brez integritete paketa je možen napad *odpoved storitve*, ki generira ponarejena nadzorna sporočila. Ta lahko prekinejo tunel oz. pod njim ležečo povezavo PPP.
- Protokol L2TP ne določa mehanizmov za šifriranje podatkovnega prometa.
- Čeprav lahko tovor paketov PPP šifriramo, protokolni sveženj PPP ne določa mehanizma za avtomatsko generiranje oz. osveževanje ključev.

Strateška usmeritev L2TP je realizacija novega protokola IPSec za uporabo s PPP in L2TP. Protokol L2TP je odlična usmeritev, saj omogoča ceneni oddaljeni dostop do notranjega omrežja. Če ga uporabimo skupaj s protokolom IPSec, je to odlična metoda določanja varnega dostopa do notranjega omrežja.

Dodatek 1: Dijkstrov algoritem iskanja najkrajše poti

Velikokrat želimo v aplikacijah najti najkrajšo pot med dvema vozliščema iz danega vozlišča (problem je znan tudi pod angl. imenom *single-source problem*). Če želimo najti eno samo najkrajšo pot med dvema danima vozliščema, v splošnem ne obstaja algoritem, ki bi v najslabšem primeru dajal boljše rezultate od Dijkstrovega (Aho et al., 1974). Večina teh algoritmov dela v času $O(e)$.

Dijkstrov algoritem ima časovno zahtevnost $O(n^2)$ in deluje tako, da poišče množico vozlišč *S*, katerih najkrajše poti od izvora poznamo. Na vsakem koraku dodamo množici *S* iz množice preostalih vozišč, t.j. vozlišč, ki jih še nismo obiskali, tisto vozlišče *v*, katerega razdalja od izvora je krajša od vseh preostalih vozlišč. Če imajo vse povezave nenegativno ceno, smo lahko prepričani, da poteka pot od izvora do vozlišča *v* samo skozi vozlišča v *S*.

Algoritem. Dijkstrov algoritem iskanja najkrajše poti.

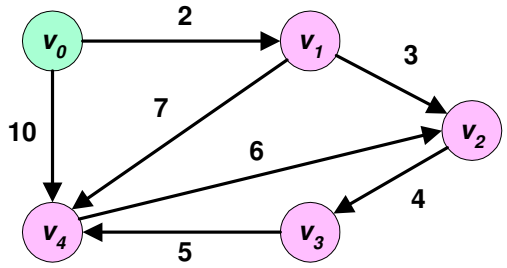
<u>*Vhodni podatki*</u>. Podan je usmerjeni graf $G=(V,E)$, z množico vozlišč *V* ter povezav *E*, izvorno vozlišče $v_0 \in V$ in funkcijo cene *l*, ki povezavo (v_i, v_j) preslika v realno vrednost $l(v_i, v_j) \in \mathbb{R}$. Predpostavimo, da velja $l(v_i, v_j) = +\infty$, če vozlišči (v_i, v_j) ne tvorita povezave, $v_i \neq v_j$ in $l(v_i, v_j) = 0$.

<u>*Izhod*</u>. Za vsak $v \in V$ minimalna izmed vseh poti *P* od v_0 do v glede na vsoto label povezav *E*.
<u>*Metoda*</u>. Tvorimo množico $S \subseteq V$ tako, da najkrajša pot od izvora do vsakega vozlišča $v \in S$ leži vsa v S. Polje $D[v]$ vsebuje ceno trenutne najkrajše poti od v_0 do v, ki gredo samo skozi vozlišča S (algoritem 1).

```
        begin
1.          S ← {v0};
2.          D[v0] ← 0;
3.          for vsak v v V − {v0} do D[v0] ← l(v0, vi);
4.          while S ≠ V do
                begin
5.                  izberi vozlišče v ∈ V − S tako, da je D[w] minimalen;
6.                  dodaj w k S;
7.                  for vsak w ∈ V − S do
8.                      D[v] ← min(D[v], D[w] + l(w, v))
                end
        end
```

Algoritem 1: Dijkstrov algoritem

Primer. Vzemimo graf na sliki 84. Začetno stanje programa je naslednje: $S=\{v_0\}$, $D[v_0]=0$ in $D[v_i]=\{2,+\infty,+\infty,10\}$ za $i=1,2,3,4$.



Slika 84: usmerjeni graf

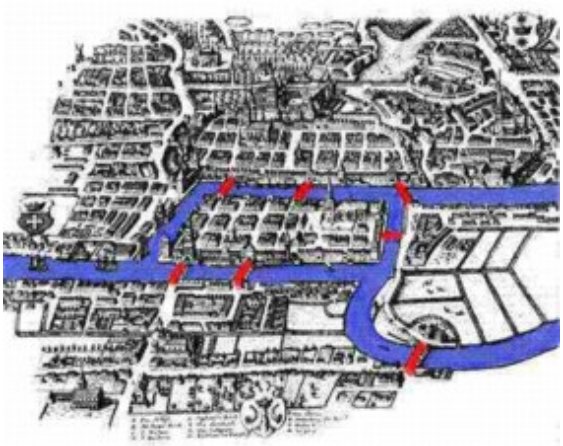
Zaporedja korakov algoritma 1 so zbrana tabeli 3.

Tabela 3: izračun algoritma 1 na grafu s slike 84

Iteracija	S	w	$D[w]$	$D[v_1]$	$D[v_2]$	$D[v_3]$	$D[v_4]$
Začetno	$\{v_0\}$	-	-	2	$+\infty$	$+\infty$	10
1	$\{v_0,v_1\}$	v_1	2	2	5	$+\infty$	9
2	$\{v_0,v_1,v_2\}$	v_2	5	2	5	9	9
3	$\{v_0,v_1,v_2,v_3\}$	v_3	9	2	5	9	9
4	Vsa	v_4	9	2	5	9	9

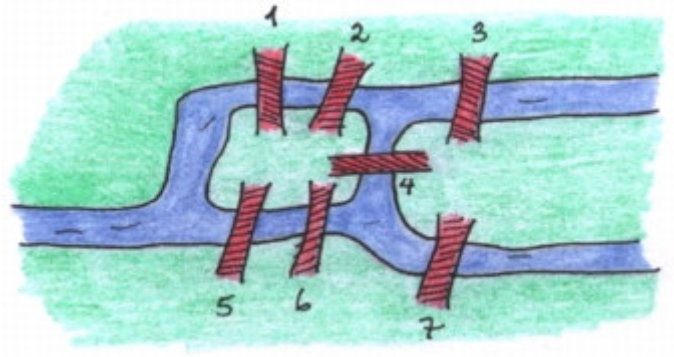
Dodatek 2: Eulerjevi grafi

Problem. V Koenigsbergu, Nemčija, teče reka skozi mesto tako, da je njegov center na otoku, za katerim se reka znova razdeli v dva strugi. Meščani so zgradili sedem mostov, tako da so lahko hodili iz enega dela mesta v drugega (slika 85).



Slika 85: Mesto Koenigsberg v Nemčiji

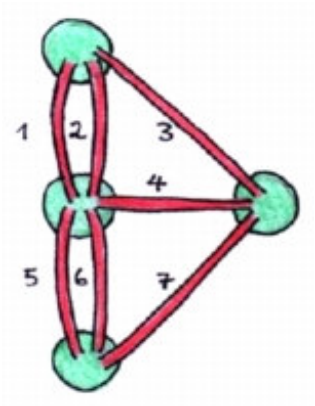
Meščani so brez uspeha poskušali najti obhod mesta, ki bi prečkal vsak most natanko enkrat in se vrnil v izhodišče. Začeli so verjeti, da naloga nima rešitve. Dokazati tega ni znal nihče, dokler se problema ni lotil Leonhard Euler (1707-1783). Eulerjev dokaz je bil objavljen v članku leta 1736. Čeprav ni bil napisan v jeziku grafov, pa so uporabljene ideje po naravi v bistvu grafsko-teoretične, zato ga upravičeno lahko štejemo za prvi članek iz teorije grafov. Problem Koenigsberških mostov (Wilson, 1996))shematično predstavlja slika 86.



Slika 86: Shema mostov v Koenigsbergu

Rešitev. Problem Koenigsberških mostov lahko prevedemo v jezik teorije grafov na sledeč način: štiri dele mesta uporabimo za točke grafa, sedem mostov pa za povezave grafa (slika 87). Problem poiskati *obhod mesta, ki bi prečkal vsak most natanko enkrat in se vrnil v izhodišče* ustreza problemu t.i. iskanja Eulerjevega *obhoda v tem grafu*.

Za Eulerjev obhod mora biti izpolnjen naslednji pogoj: *kadarkoli pridemo v neki del mesta, mora obstajati možnost, da ta del zapustimo po še neprehojenem mostu.*



Slika 87: Graf, ki ustreza problemu Koenigsberških mostov

Gornji pogoj pomeni, da vsakokrat, ko pridemo v neko točko, moramo imeti možnost, da jo zapustimo po drugi povezavi. Vsakič, ko obiščemo neko točko, prispevamo natanko 2 k stopnji te točke. Zato mora imeti v Eulerjevem grafu vsaka točka sodo stopnjo.

Pravilo, ki ga je v svojem dokazu predlagal Euler, je: *za preverjanje, ali je dani graf Eulerjev, moramo pogledati stopnje točk, t.j. če so vse soda števila, potem je graf Eulerjev; če niso, potem graf ni Eulerjev.*

Sklep. Problem Koenigsberških mostov nima rešitve.

Literatura

Aho, A., V., Hopcroft, J., E., Ullman, J., D. (1974). *The Design and Analysis of Computer Algorithms.* Addison-Wessley Publishing Company. Reading.

Chappell, L. ed. (1999). *Advanced Cisco Router Configuration*. Cisco Systems, Inc. Indianapolis.

Dijkstra, E., W. (1959). *A note on two problems in connexion with graphs.* Nummerische Mathematik 1, p. 269-271.

Parziale, L. in ostali (2006). *TCP/IP Tutorial and Technical Overview*. International Technical Support. Na Internetni strani http://www.redbooks.ibm.com.

Vidmar, T. (2002). *Informacijsko komunikacijski sistem.* Poglavja: 2, 3, 4, in 5. Pasadena. Ljubljana.

Wilson, J., W. (1996). *Introduction to Graph Theory.* Prentice Hall. Harlow, England.