**UVOD & PONOVITEV**

**Plasti v ISO/OSI modelu**

|  |  |
| --- | --- |
| **Aplikacijska plast** | Je najbližja uporabniku, omogoča interakcijo aplikacije s omrežnimi storitvami. Standardne storitve so FTP, SMTP, SNMP, HTTP, DNS itd. |
| **Predstavitvena plast** | Določa pomen podatkov med entitetnima paroma aplikacijske plasti. Skrbi za sintakso in semantiko, določa kodiranje, kompresijo podatkov in varnostne mehanizme. |
| **Sejna plast** | Nadzor pogovora (množice povezav) med aplikacijama, logično povezovanje med aplikacijami, običajno je vgrajena v aplikacije. |
| **Transportna plast** | Skrbi za učinkovit, zanesljiv in transparenten prenos podatkov med uporabnikoma, te storitve zagotavlja višjim plastem, za naslovitev procesov, kontrolo pretoka, nadzor zamašitev itd. |
| **Omrežna plast** | Usmerjanje, prenos podatkov od izvornega do ciljnega računalnika |
| **Povezavna plast** | Okvirjanje datagramov, zaznavanje in odpravljanje napak pri prenosu (CRC, checksum, parity), fizično naslavljanje (MAC naslov). Protokoli: Ethernet, PPP, Frame Relay |
| **Fizična plast** | Prenos bitov po kanalu (baker/optika/brezžično). Bluetooth, 802.11 etd. |

**Posebne vrste IP naslovov**

|  |  |  |
| --- | --- | --- |
| **IP različica** | **Naslov** | **Pomen tega naslova** |
| **IPv6** | 0000:0000:0000:0000:0000:0000:0000:0001 | localhost |
| **IPv6** | 0000:0000:0000:0000:0000:0000:0000:0000 | nedefiniran |
| **IPv6** | FE80:0000:0000:0000:0000:0000:0000:0000 / 64 | Link-local naslovi |
| **IPv6** | FC00:0000:0000:0000:0000:0000:0000:0000 / 7 | Unique local (zasebni naslovi, dodeli jih registrar |
| **IPv6** | FF02:0000:0000:0000:0000:0000:0000:0001 | Link local – VSI VMESNIKI |
| **IPv6** | FF02:0000:0000:0000:0000:0000:0000:0002 | Link local – VSI USMERJEVALNIKI |
| **IPv4** | 127.0.0.1 | Localhost |
| **IPv4** | 169.254.0.0 – 169.254.255.255 | Rezerviran za APIPA (dobimo, ko nimamo static in NI DHCPja) |
| **IPv4** | 10.0.0.0 – 10.255.255.255 | Privatna notranja omrežja |
| **IPv4** | 172.16.0.0 – 172.31.255.255 | Privatna notranja omrežja |
| **IPv4** | 192.168.0.0 – 182.168.255.255 | Privatna notranja omrežja |

**Varnost: Kako lahko vdiralec škoduje našem sistemu?**

- Prisluškovanje (prestrezanje sporočil) - Aktivno ponarejanje - Kraja identitete (ponaredi vsebino paketa, npr src)

- Prevzem povezave (prevzame vlogo) - Onemogocanje nudenja storitve (DOS)

**Varnost: Zagotavljanje zanesljivosti**

* Nadzor (zbiranje podatkov o delovanju, uporabi, dnevniki)
* Upravljanje (ukrepanje na podlagi zbranih podatkov, diagnostika, administracija)
* Sistematičnost (imeniki, seznami in kazala, SNMP)
* Načrtovanje (zmogljivosti, razvoj, testiranje in upravljanje)
* Razpršenost zaščite (integriteta povezav, virov, vsebine, uporabnikov, sporočil)

**Varnost: Elementi varne komunikacije**

* Zaupnost (kdo sme prebrati)
* Avtentikacija (dokaži, da si to res ti)
* Razpoložljivost in nadzor dostopa (preprečevanje nelegitimne rabe virov): Avtorizacija (kaj smeš ti narediti) in Accounting (kaj je kdo naredil)
* Integriteta sporočila (ali je bilo med prenosom spremenjeno)
* Onemogočanje zanikanja (res si to poslal / res si to prejel)

**PRIKLOP IN ZAGON NAPRAVE**

**BIOS**

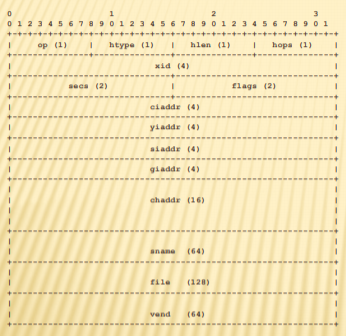
* Sestoji iz dveh sklopov (koda ki se začne izvajati ob zagonu & gonilniki za V/I enote)
* Koda izkoristi gonilnike za dostop do zunanjih enot
* BIOS v resnici naloži nek program, ki ga začne izvajati, najde ga na prvem bloku V/I enote (Master boot record - MBR)

**Prednosti in slabosti nalaganja s omrežja Koraki za uspešno nalaganje računalnika s omrežja**

* Znati poiskati strežnik, s katerega bo naložil OS
* Znati se postaviti kot zahteva / svetuje srežnik
* Prenesti OS k sebi
* Namestiti OS in ga zagnati

|  |  |
| --- | --- |
| Ne potrebujemo diska na računalniku | Ranljivost |
| OS zamenjamo samo enkrat za vse naprave | Počasnost |

**BOOTP Protokol (RFC 951)**

* Lahko je hkrati prisotnih več strežnikov in lahko hkrati več odjemalcev želi naložiti OS. Če je več strežnikov, se moramo odločiti za enega.
* Na prenosni plasti uporablja protkol UDP
* Na začetku so bili vsi protokoli narejeni na UDP, ker je glava manjša in tudi koda manjša – čim manjši pomnilnik
* Koračni protokol (odjemalec vpraša, strežnik odgovori)
  + Odjemalec se priklopi in nima naslova IP. Na lokalno mrežo prek broadcast pošlje zahtevo po želji nalaganja OS
  + ****Strežnik mu dodeli IP naslov (lahko tudi ne) in mu pove, kje se nahaja OS (lahko izven lokalne mreže)

**op** zahteva ali odgovor

**htype** vrsta medija

**hlen** dolžina naslova

**chaddr** odjemalčev naslov plasti 2

**hops** število skokov (če dobivamo nastavitve glede na reference, preko enega drugega strežnika do drugega strežnika nas zanima število skokov do tega strežnika, glede na to se odločimo ali ga bomo vzeli ali je predaleč za hitrost)

**xid** id zahteve (razločanje med odgovori na določenezahteve)

**secs** koliko časa je minilo od prvega pošiljanja

**flags** zastavice – samo razpošiljanje ali ne

**ciaddr** odjemalčev naslov (željen naslov)

**yiaddr** strežnikov naslov (določen naslov)

**guaddr** naslov prehoda

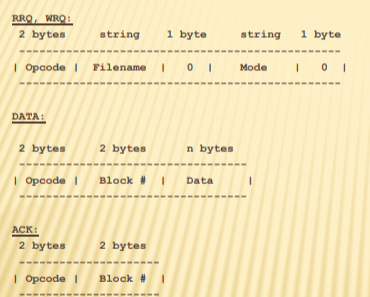
**sname** ime strežnika z OS, na IP naslov tega strežnika

**file**  datoteka z OS

**vend** možne razširitve (vendor specific, RFC ponuja poljubne podatke, prvi 4 bajti so kakšni so podatki, HP bo dal tako število, itd, vsak ima svojo 4 številčno kodo, v naslednjih vendor specific podatkih so takšni in takšni podatki, v protokol vgradimo svoj protokol)

**TFTP Protokol (RFC 1350)**

* Zelo poenostavljena funkcionalnost FTP protokola – ohranjena predvsem možnost prenosa podatkov
* Ni izpisa imenika, avtentikacije, kriptiranja, dovoljuje zelo velike pakete, največ 1TB
* Odjemalec na začetku pozna IP strežnika, saj mu ga je povedal BOOTP
* UDP na prenosni plasti
* Potek pogovora (branje)
  + Odjemalec pošlje zahtevo po branju (RRQ)
  + Strežnik odgovori s DATA paketom in podatki, ki so bili zahtevani
  + Na vsak prejet paket odjemalec odgovori s ACK paketom, nato strežnik pošlje naslednjega
  + Zadnji paket je manjši od največje dovoljene velikosti, če pade tako da je enak, pošljemo še en prazen paket

**opcode** zahteva

**filename 0** ime datoteke

**mode 0** oblika zapisa

**block #** številčenje poslanih podatkov

* Kako vemo da so zaupanja vredni podatki, kako vemo da ni prišlo do napake?
  + Datoteki, ki jo prinašamo dodamo CRC kodo, zaščitno kodo, ki skrbi za celovitost podatkov.
  + Kako zaupamo? DODAMO PODPIS
  + PODPIS: datoteko pošljemo čez HASH (MD5, SHA 256), DOBIMO NEKO ZGOŠČENO VREDNOST

Vzamemo asimetrično kriptografijo, zgoščeno vrednost obdelamo z zasebnim ključem in pošljemo, na podlagi tega, lahko odjemalec ki prejme tako datoteko, celotno datoteko spet poda čez isto zgoščevalno funkcijo in jo primerjamo z dešifriranim izvlečkom pošiljatelja ( vzamemo pošiljateljem javni ključ )

Podpis vsebuje poleg zašifrirane zgoščene datoteke tudi katero zgoščevalno funkcijo je uporabil ter potrdilo javnega ključa.

**DHCP Protokol (RFC 2131)**

* Je razširitev BOOTP Protokola, možnost vend je preimenovana v options
* Na prenosni plasti uporablja UDP protokol
* Tipi sporočil
  + **DHCPDISCOVER -** ISKANJE STREŽNIKA
  + **DHCPOFFER -** PONUDBA ODJAMALCA
  + **DHCPREQUEST -** ODJEMALEC POTRJUJE PREJETE NASTAVITVE, TUDI ŽELJA PO PODALJŠANJU SPOSOJE IP
  + **DHCPACK, DHCPNAK -** STREŽNIKOVA POTRDITEV/ZANIKANJE
  + **DHCPDECLINE -** ODJEMALCU STREŽNIKU, DA JE IP NASLOV ŽE V UPRABI
  + **DHCPRELEASE -** ODJEMALEC VRAČA NASLOV PRED POTEKOM
  + **DHCPINFORM -** ODJEMALEC ŽELI SAMO OSTALE PODATKE, NASLOV ŽE IMA
* Nevarnosti
  + DHCP ne predvideva avtentikacije
* Možni napadi
  + Neavtorizirani strežniki posredujejo napačno informacijo (podpis)
  + Neavtorizirani odjemalci pridobijo dostop do virov, do katerih ne bi smeli
  + Izpraznenje virov s strani neavtoriziranih odjemalec

**DHCPv6 Protokol (RFC 3315)**

**SOLICIT** prošnja za nastavitev **ADVERTISE** oglašanje naslova  **REQUEST** zahteva za nastavitvene parametre

**CONFIRM** preverjanje, ali je naslov, ki ga je dobil odjemalec, še vedno ok **RENEW** zahteva za obnovitev

**REBIND** zahteva za ohranitev **REPLY** odgovor odjemalcu **RELEASE** sprostitev naslova

**DECLINE** zavrnitev dodeljenega naslova **RECONFIGURE** strežnik odjemalcu sporoča, naj obnovi nastavitve

**INFORMATION-REQUEST** zahteva za nastavitve brez IP naslova **RELAY-FORW** prepošiljanje

**RELAY-REPL** potrdilo prepošiljatelju, ki vsebuje odgovor odjemalcu

**NADZOR & UPRAVLJANJE S OMREŽJI**

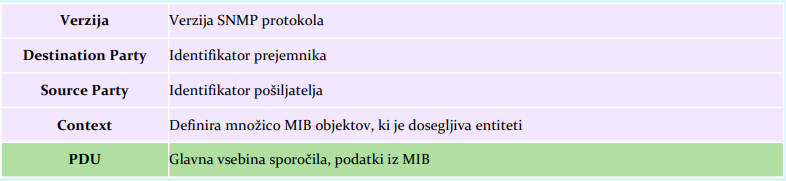
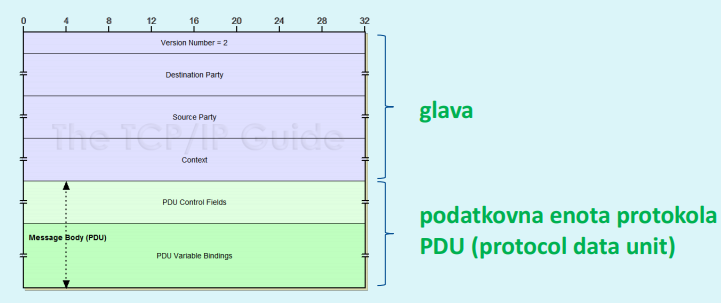
* Primeri aktivnosti upravljanja
  + zaznavanje napake na vmesniku računalnika ali usmerjevalnika: programska oprema lahko sporoči administratorju, da je na vmesniku prišlo do težave (celo preden odpove!)
  + nadzorovanje delovanja računalnikov in analiza omrežja
  + nadzorovanje omrežnega prometa: administrator lahko opazuje pogoste smeri komunikacij in najde ozka grla
  + zaznavanje hitrih sprememb v usmerjevalnih tabelah: ta pojav lahko opozarja na težave z usmerjanjem ali napako v usmerjevalniku
  + nadzorovanje nivoja zagotavljanja storitev: ponudniki omrežnih storitev nam lahko jamčijo razpoložljivost, zanasnitev in določeno prepustnost storitev; administrator lahko meri in preverja
  + zaznavanje vdorov: administrator je lahko obveščen, če določen promet prispe iz sumljivih virov; zaznava lahko tudi določen tip prometa (npr. množica SYN paketov, namenjena enem samem vmesniku)
* Primeri
  + Popis IP naslovov, diagnostika in odkrivanje napak, odkrivanje topologije omrežja profiliranje, pretok podatkov
* Definicija
  + Upravljanje z omrežjem vklučuje vpeljavo, integracijo in koordinacijo s strojno opremo, programsko opremo človeškimi viri z namenom opazovanja, testiranja, konfiguriranje, analiziranja in nadzorovanja omrežnih virov
* Področja upravljanja
  + Upravljanje s napakami, upravljanje s konfiguracijami (npr. spremeni DNS vsem hkrati), upravljanje s beleženjem dostopov (beleženje sej, uproabe, sprememb v sistemu), upravljanje s varnostjo
* V primetu velikega omrežja s veliko usmerjevalniki, sta protokola YANG in NETCONF bolj učinkovita

**SNMP Protokol**

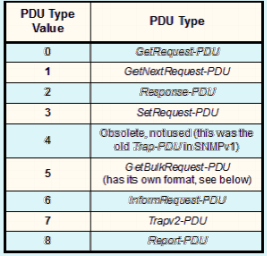
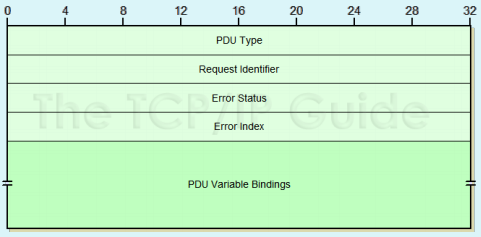
* Simple Network Management Protocol je protokol za izmenjavo nadzornih informacij med upravljalcem in nadzorovanimi objekti. Podatki o nadzorovanih objektih se prenašajo med nadzorovano opremo in upravljalcem skladno z definicijo MIB
* Dva načina delovanja:
  + zahteva-odgovor (request-response): bere in nastavlja vrednosti v MIB
  + obvestilo (trap message): naprava obvesti upravljalca o dogodku
* Na prenosni plasti uporablja protkol UDP, ker ne zaseže toliko prostora
* Ni potrebno hranjenje stanj
* Zaradi UDP težji DOS napad
* Tipi sporočil

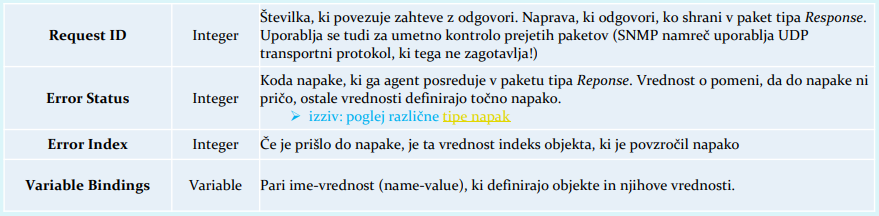
|  |  |  |
| --- | --- | --- |
| **Sporočilo** | **Smer** | **Pomen** |
| GetRequest  GetNextRequest  GetBulkRequest | Upravljalec -> agent | Get MiB |
| SetRequest | Upravljalec -> agent | Set MiB |
| Response | Agent -> upravljalec | Returns MiB |
| Trap | Agent -> upravljalec | Obvestilo upravljalcu o izrednem dogodku |
| InformRequest | Upravljalec -> upravljalec | Medsebojno posredovanje vrednosti iz MiB |

* Pametno je imeti več upravljalcev, saj če eden odpove, se poruši vse
* Imamo posebna vrata za obvestila (162), vrata 161 pa poslušajo po SNMP zahtevah
* Ker nimamo TCP, so pogoste izgube. Po določenem času se paket pošlje ponovno.
* Implementacija SNMP mora reševati naslednje težave:
  + velikost paketov: SNMP paketi lahko vsebujejo obsežne informacije o objektih v MIB, UDP pa ima zgornjo mejo velikosti segmenta (TCP nima)
  + ponovno pošiljanje: ker se uporablja UDP, nimamo zagotovljene dostave in potrjevanja. Nadzor dostave je torej potrebno reševati na višjem OSI nivoju
  + problem z izgubljenimi obvestili: če se obvestilo pri prenosu izgubi, pošiljatelj o tem nič ne ve; prejemnik pa ga tudi ne dobi. SNMP pošlje znova po določenem času.
* Glava glavnega paketa:

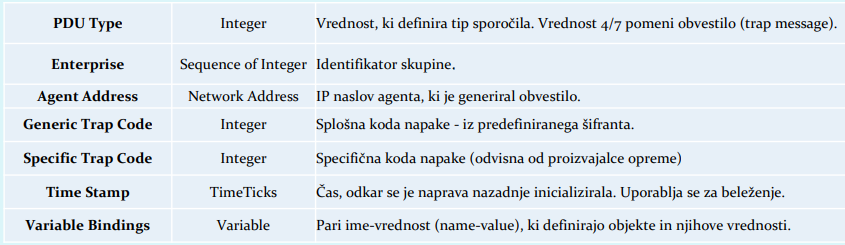
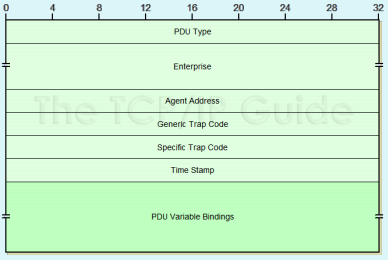


* Glava paketa tipa zahteva-odgovor:

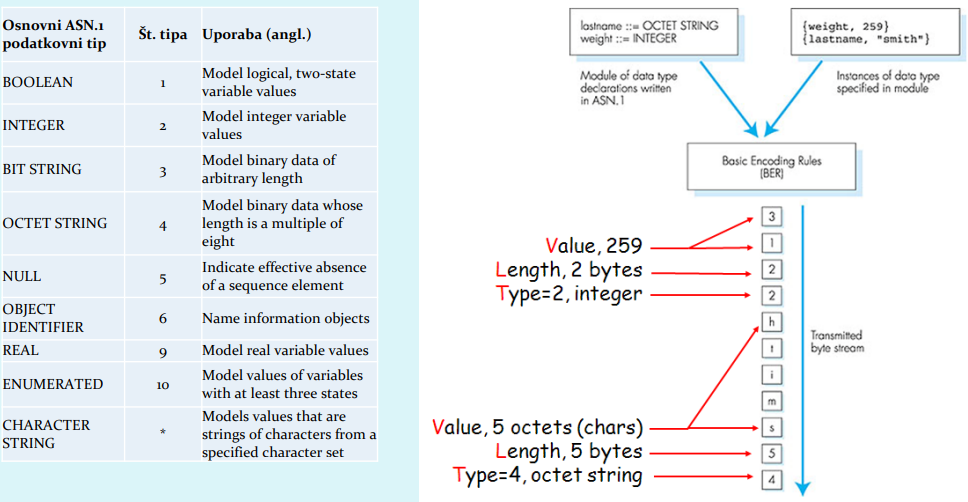




* Glava tipa obvestilo (trap message)

* Različice protokola SNMP
  + SNMP v1
    - definiran konec 80-ih let. Izkazal se je za prešibek za implementacijo vseh potrebnih zahtev (omejen pri sestavi PDU paketov)
  + SNMP v2
    - izboljšan SNMPv1 na področjih hitrosti (dodan GetBulkRequest), varnosti (vendar prekompleksna implementacija), komunikacij med upravljalci
    - RFC 1901, RFC 2578
    - uporablja SMIv2 (izboljšan standard za strukturiranje informacij)
  + SNMP v3
    - Varnostni elementi vpeljani šele tukaj, prejšnji dve različici nista imeli
    - izboljšan SNMPv2 - ima dodane varnostne mehanizme
    - omogoča kriptografijo, zagotavlja zaupnost, integriteto, avtentikacijo
    - tudi uporablja SMIv2
* SNMP varnostni mehanizmi
  + kriptiranje vsebine paketov (PDU): uporablja se DES (ključa je predhodno potrebno izmenjati)
  + integriteta: uporablja se zgoščanje sporočila s ključem, ki ga poznata pošiljatelj in prejemnik. S preverjanjem poslane zgoščene vrednosti imamo kontrolo pred aktivnim ponarejanjem sporočil
  + zaščita proti ponovitvi že opravljene komunikacije (replay attack): uporaba enkratnih žetonov (angl. nonce): pošiljatelj, mora sporočilo kodirati glede na žeton, ki ga določa sprejemnik (to je običajno število vseh zagonov system pošiljatelja in čas, ki je minil od zadnjega zagona)
  + nadzor dostopa: kontrola dostopa na osnovi uporabiških imen. Pravice določajo, kateri uporabniki lahko berejo/nastavljajo katere informacije. Podatki o uporabnikih se hranijo v bazi Local Configuration DataStore, ki ima ravno tako nadzorovane objekte s SNMP!
* Kako kodirati vsebino paketa, da bo razumljiva na vseh platformah (različna dolžina podatkovnih tipov, debeli / tanek konec)
  + ASP.1 standard poleg podatkovnih tipov definira tudi standarde kodiranja
  + TLV notacija
  + Možne rešitve:
    - Pošiljatelj upošteva obliko podatkov, ki jo uporablja prejemnik: podatke pretvarja v njegovo obliko in nato šele pošlje
    - Pošiljatelj pošlje podatke v svoji obliki, prejemnik pretvori v lastno obliko
    - Pošiljatelj pretvori v neodvisno obliko in nato pošlje. Prejemnik neodvisno obliko pretvori v svojo lastno obliko
  + ASN.1 uporablja tretjo rešitev – BER po principu TLV
  + Alternative rešitve so XML & SOAP, CORBA



* RMON (Remote Monitoring) (dodatni mehanizem): Klasični SNMP lahko nadzoruje omrežje iz nadzorne postaje. RMON zbira in analizira meritve lokalno, rezultate pošlje oddaljeni nadzorni postaji. Ima svoj MIB z razširitvami za različne tipe medijev

**STVARNI ČAS IN KOMUNIKACIJE**

**NTP Protokol (RFC 5905)**

* NTP (Network Time Protocol) uporabljamo za sinhroniziranje časa na več napravah v omrežju. NTP dobi čas iz nekega vira in nastavi lokalni čas na čas, ki ga dobi. Vsaka naprava je lahko strežnik in odjemalec.
* NTP strežniki se delijo na sloje (Stratum). Strežnik Stratum 1 dobi svojo uro iz atomske ure. Ti strežniki niso dostopni javnosti. So na Internetu, a do njih nimamo dostopa. Stratum 1 dajo čas Stratum 2 strežnikom itd. Če imamo veliko število računalnikov v omrežju (npr. 1000), vzamemo 2 ali 3 od teh in vzamemo čas od Stratum strežnika, nato pa naše ostale naprave konfigurirajo, da dobijo čas od teh, ki so dobili čas od Stratumov.
* Najprej se naprave sinhronizirajo vsako minuto, nato ta interval vsakih 17 minut naraste.
* Protokol uporablja UDP, ker TCP bi v primeru izgube paketa naredil NTP paket manj zanesljiv, zaradi mehanizma za ponovno pošiljanje

**RTP Protokol (RFC 3350)**

* RTP (Real-time Transport Protocol) uporabljamo za prenos podatkov v stvarnem času. Je framing protokol za aplikacije, ki ne definira nobenih QoS mehanizmov. RTP teče v končnih sistemih, nikjer na omrežju
* Uporabljamo ga za PRENOS
* Enkapsuliran je v UDP segmentih
* RTP upravlja prenos podatkov v stvarnem času in specificira strukturo paketov za prenos avdia in videa
* Osnovne funkcionalnosti:
  + Skrbi za pravo zaporedje paketov
  + Skrbi za časovne značke dogodkov
* Dodatne funkcionalnosti:
  + Ena povezava lahko prenaša več podatkovnih tokov
  + Identifikator vira / seje
  + Poseben element – mešalec (mixer)
* Varnost
  + RTP protokol uporablja UDP prenos, ki nima SSL plasti
  + Zato moramo varnost dograditi sami
  + Uporabljamo kriptiranje s tokom šifer, zaradi izgube paketov ko si izmenjamo ključe

**Zakaj uporabljamo te protokole? Zakaj ne moremo enostavno prenašati s pomočjo TCP ali UDP?**

* TCP
  + Ob izgubi paketa se sproži mehanizem za ponovno pošiljanje paketov, kar povzroči zamudo
  + Zaradi ponovnih pošiljanj niha tok prihoda paketov
* UDP
  + Preveč enostaven
  + Zaradi varnostnih razlogov ga požarni zid velikokrat ovrže

UDP ponavadi uporabljamo za prenos čistega zvoka (npr. VoIP). TCP ponavadi uporabljamo, kjer imamo večji pomnilnik (buffer), npr. YouTube. V tem primeru, tudi ob zamiku ni nujno, da pride do prekinitve, saj imamo podatke v pomnilniku.

**Cilji pri oblikovanju**

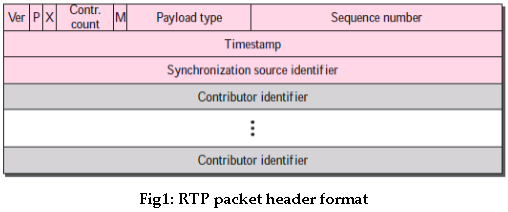
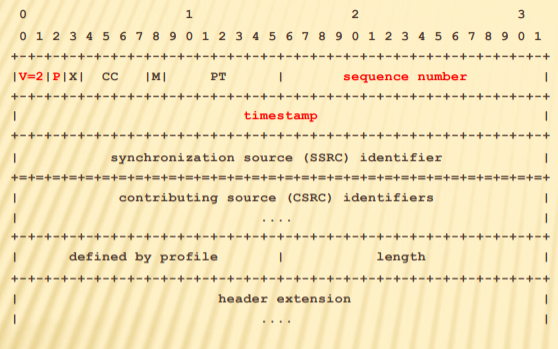
* Protokol mora biti fleksibilen, ne diktira mehanizmov (MPEG itd.)
* Neutralen glede na protokole na nižji (transportni) plasti
* Moral bi biti scalable (deluje na unicast, multicast itd.)

**Zahteve, ki jih mora izpolniti**

* Pogajanje pri kodnih shemah ipd. (lahko imamo dve aplikaciji v omrežjih s zelo različnimi pasovnimi širinami – možnost pogajanja)
* Pomoč prejemniku za zaznavanje časovnih relacij med različnimi podatki (časovne oznake, sinhronizacija različnih medijev, npr. ujemanje govora s videom, da ne zamuja)
* Obvestilo o izgubi paketov (ker teče na UDP, ni zagotovila da paketi prispejo. RTP mora omogočiti opozorilo, da je paket bil izgubljen, pa naj se nato aplikacija odloči kaj narediti glede tega)
* Identiteta pošiljatelja (neodvisna od IP vmesnika – ne glede na IP lahko vemo, kdo je pošiljatelj)

**Dve glavni ideji izpolnjevanja zahtev**

* Application level framing
  + Aplikacija ve, kaj potrebuje, sama naj okvirja in formatira podatke
  + Imamo več profilov, ki ustrezajo različnim aplikacijam
  + Prednost je fleksibilnost za specifične aplikacije
* End-to-end principle
  + Končni sistemi so odgovorni za ponujanje storitve, ne glede na omrežne sposobnosti
  + Vsa inteligenca je v aplikacijah, RTP omogoča le prenos

 **V** Verzija – 2

**P** Zapolnitev (padding)

**M** Marker – postavlja meje v podat. toku

**PT** Identifikacija protokola (npr H264)

**CC** Število mešanih virov

**Sequence Num** Številčenje paketov – zaznava izgub, rep. att.

**Timestamp** Oznaka dogodka (ne paketa)

**SSRC** Kakšen tip podatkov prenašam (ID vira)

**Source Identifier** Da vemo, kdo je vir podatkov

**Contributing src** Identifikator mešanih virov (če je več virov)

**X** Ali je prisotna razširitev glave

Mixer je posebna naprava, ki kombinira toke podatkov iz različnih virov. Mixer sprejme podatke vseh virov, jih kombinira in predvaja tako, da prejemnik lahko ustrezno gleda. Če je več virov podatkov (CSRCs), se dajo vsi v mixer in nato mixer postane SSRC, vsi ostali pa CSRCs.

**Kako zaznava flow?**

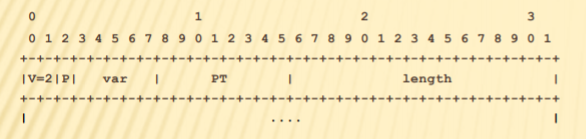
* Spremlja konstanten SSRC in naraščujočo zaporedno številko paketa

**RTCP Protokol**

RTCP (Real-Time Tranport Control Protocol) je pomožni protokol k RTP protokolu. Uporabljamo ga za prenos kontrolnih podatkov in poročil za RTP protokol. Če je veliko sodelujočih strank (multicast), potem je gostota poročanja manjša.

Uporabljena je stalna pasovna širina.

* Opravlja 4 funkcije:
  + Sporoča o kakovosti prenašanega prometa (RR – receiver report & SR – sender report)
  + Dodaten opis vira toka dogodkov (SDES: Source description items)
  + Skrbi za pravilno gostoto pošiljanja sporočil o kakovosti prenosa
  + Prenaša lahko še dodatne podatke za potrebe aplikacije (APP: Application-Specific functions)

**V** verzija

**P** zapolnitev

**PT** ukaz SR, RR, SDES (opisuje SSRC), BYE, APP

**Var** različne vrednosti odvisne od ukaza

**SRTP Protokol (RFC 3711)**

SRTP (Secure Real-Time Transport Protocol) uporablja kriptiranje s tokom šifer. Imamo simetrično šifriranje, ker je mnogo hitreje, poleg tega, pa bi za različne prejemnike bilo učinkovitejše, saj ne rabimo posebej za posameznega šifrirat z njihovimi javnimi certifikati.

* Ključ od prejšnjega bloka se uporablja za tvorjenje podpisa naslednjega bloka začetna vrednost (IV) je poznana obema stranema
* obema stranema je poznan tudi ključ
* vsak paket se ločeno zakriptira
* + je preprosti xor ali kakšen podoben algoritem
* če se paket izgubi, samo v prazno zavrtimo E

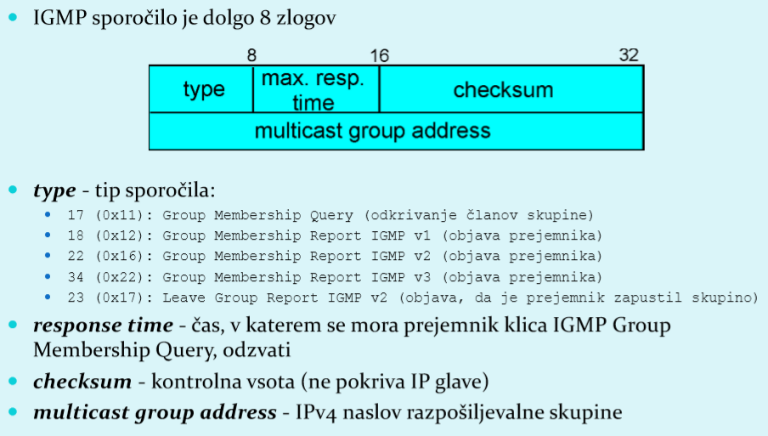
**RTSP Protokol**

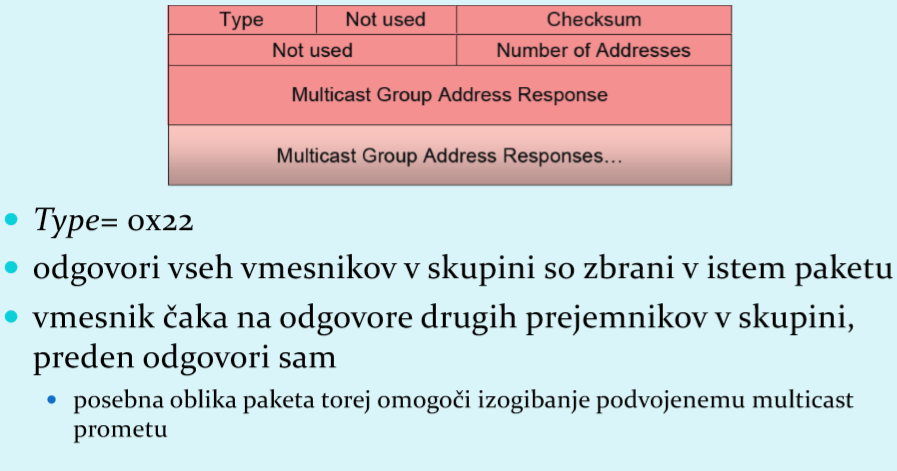
RTSP (Real Time Streaming Protocol) se uporablja za vzpostavljanje povezave in nadzorovanje sej med strežnikom in odjemalcem. Sam prenos podatkov ni delo RTSP protokola. RTSP uporablja TCP za ohranjevanje povezave med končnima sistemoma.

**RAZPOŠILJANJE (MULTICAST)**

**IGMP Protokol**

* IGMP (Internet Group Management Protocol) se uporablja za upravljanje s skupinami
* PRIMER: Poslati želimo 4 od 6 računalnikov v omrežju:
  + unicast (potrebujemo 6 kopij istega paketa, večkratno pošiljanje obremenjuje medij)
    - To ni dobro, ker preobremeni vir
  + broadcast (naslovi vse računalnike, filtriranje pravih prejemnikov prepustimo protokolom na višjih plasteh)
    - To ni dobro, ker preobremeni mrežo
  + multicast(pošljemo »posebnemu naslovu«, ki predstavlja skupino prejemnikov, ki posluša pakete naslovljene na ta naslov (podobno kot broadcast paket dobijo vsi, vendar filtriranje se izvede na omrežnem nivoju – IP (včasih lahko tudi na povezavnem)))
* Izzivi protokola:
  + postopek razpošiljanja
    - odkrivanje, kje so prejemniki paketa
    - razpošiljanje zahteva dodatno delo: usmerjevalni protokoli, posredovanje informacije o poslušateljih
    - razpošiljalni naslovi ne oblikujejo (pod)mrež -> maska ima 32 bitov. V usmerjevalnih tabelah zato zahtevajo posebne vnose
  + varnost
  + ali lahko paket razpošlje vsakdo ali samo član skupine?
  + prisluškovalec se lahko naroči na poslušanje paketov in postane legitimni prejemnik
  + zanesljivost
    - kaj narediti, če samo en prejemnik javi, da ni dobil paketa
* Aplikacije razpošiljanja
  + Pošiljanje velikih datotek preko omrežja (glavni urad podružnicam) – zanesljiv prenos SLABA IDEJA, če na višji plasti ni stvari, ki poskrbi za to.
  + Nadgradnja programske opreme v velikem omrežju
  + Podatkovni tokovi (data streaming) – npr. pošiljanje podatkov o delnicah vsem finančnim družbam) – slabo a manj pomembno da se nekaj zgodi
  + Audio/video tokovi podatkov (streaming) – tipičen primer uporabe
  + Video na zahtevo (spremnljanje TV programa)
  + Izvedba konferenc (pomislek: boljša uporaba konferenčnega cedntra, ki odloča, kdo lahko govori in čigave pakete posredovati drugim)
  + Aplikacije v realnem času z RTP, ki se uporablja za zagotavljanje tekoče in kakovostne dostave v okoljih, kjer se uporablja razpošiljanje
* Verzije IGMP
  + IGMPv1
    - Vmesniki se lahko pridružijo skupinam. Sporočila za izstop ne obstajajo. Usmerjevalniki uporabljajo mehanizem s pretekom časa, da odkrivajo skupine, ki za vmesnik niso zanimive
  + IGMPv2
    - Dodana sporočila za izstop iz skupine. S tem omogočeno hitrejše sporočanja usmerjevalniku o prekinitvi dostave nepotrebnega prometa
  + IGMPv3
    - Večje spremembe v protokolu. Vmesniki lahko določijo SEZNAM drugih vmesnikov, od koder želijo prejemati promet. Promet od ostalih vmesnikov omrežje prepreči





* Prijava na vir
  + Korak 1
    - za pridružitev skupini, se pošlje GMR (group membership request) sporočilo z vrednostjo TTL=1  dostava samo najbližjemu usmerjevalniku
    - usmerjevalnik evidentira, da mora skupinske pakete posredovati novemu naročniku
  + Korak 2
    - usmerjevalnik sporoči sosednjim usmerjevalnikom, da ima novega naročnika. Če bi vsak usmerjevalnik sporočil enako naprej, pride do problema -paketi bi se posredovali navzkrižno preko vseh povezav v omrežju. Rešitvi:
      * uporablja se RPL algoritem (Reverse Path Lookup): zavržemo vse multicast pakete, ki pridejo od usmerjevalnikov, ki ne povezujejo z izvorom paketa po najbližji poti
      * usmerjevalniki imajo posebne usmerjevalne protokole za multicast promet:npr. protokol PIM-SM (Protocol Independent Multicast -Sparse Mode)
* Usmerjanje Multicast
  + Naloga usmerjanja: najti drevo povezav, ki povezuje vse usmerjevalnike v isti razpošiljevalni skupini
  + Za komunikacijo med usmerjevalniki potrebujemorazpošiljevalne usmerjevalne protokole (aplikacijaki protokoli, ki delujejo na omrežni plasti), kot so: PIM, DVMRP, MOSFP in BGP
* Dve rešitvi iskanja razpošiljevalnega drevesa
  + uporaba enega samega drevesa za vseusmerjevalnike zausmerjanje razpošiljevalnega prometa se določi eno samo drevo (group-shared tree)
  + določitev ločenega drevesa za vsakega udeleženca v skupini (source-based tree); za N članov skupine imamo torej N dreves (za vsako razpošiljevalno skupino)