LTAT.06.001 Operatsioonisüsteemid

Võrk II jätk. Riistvarasuhtlus

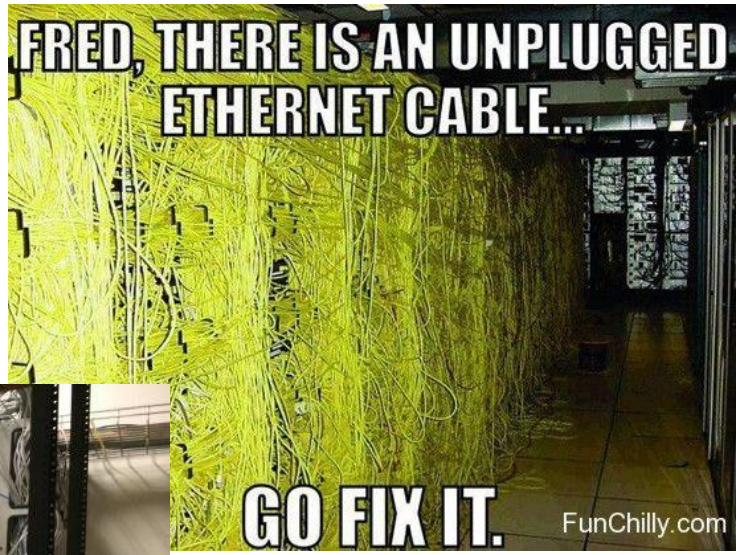
Artjom Lind

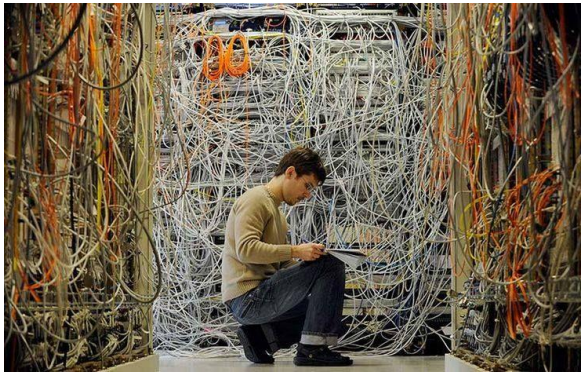
artjom.lind@ut.ee

21.11.2025

Loengu plaan

- Võrguprotokoll
- OSI mudell
- Võrguliidesed
- Edastusviisid
- Ethernet
- IP
- TCP
- UDP
- NAT
- IPv6
- Sockets





Võrguprotokoll

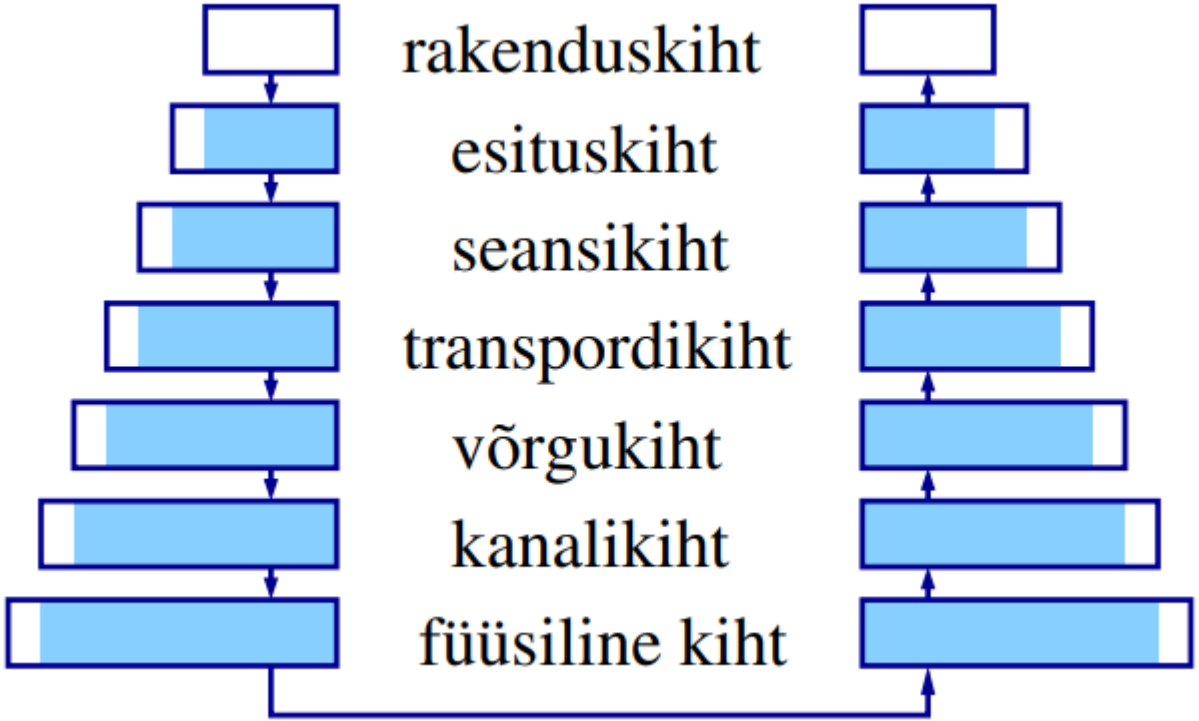
Võrguprotokoll on reeglite ja protseduuride kogum, mis määratlevad, **kuidas andmed liiguvad läbi võrkude**. Need protokollid võimaldavad erinevatel võrguseadmetel, nagu arvutid ja serverid, omavahel suhelda ning andmeid edastada. **Protokollid määratlevad, kuidas andmeid pakendatakse, saadetakse, vastu võetakse ja dekodeeritakse.**

Näiteid võrguprotokollidest:

- TCP (Transmission Control Protocol) - Tagab andmete usaldusväärse edastamise, korrigeerides vigu ja haldades andmevoogu.
- IP (Internet Protocol) - Vastutab andmepakettide saatmise eest ühest arvutist teise, määrates siht- ja lähteaadressid.
- HTTP (Hypertext Transfer Protocol) - Kasutatakse veebisaitide sisu edastamiseks veebiserveritest veebibrauseritesse.
- SMTP (Simple Mail Transfer Protocol) - Võimaldab e-kirjade saatmist ja vastuvõttu.
- FTP (File Transfer Protocol) - Kasutatakse failide ülekandmiseks arvutite ja serverite vahel.

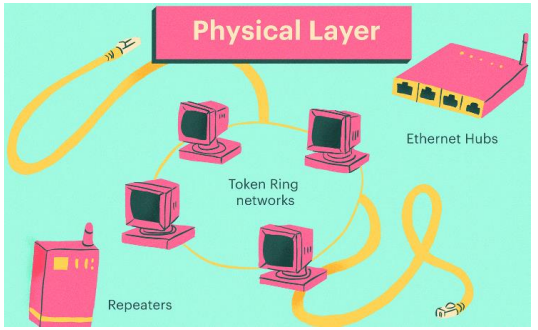
Iga protokoll täidab oma kindlat rolli võrgusuhtluses, tagades andmevahetuse sujuvuse ja efektiivsuse.

ISO/OSI 7-kihiline mudel



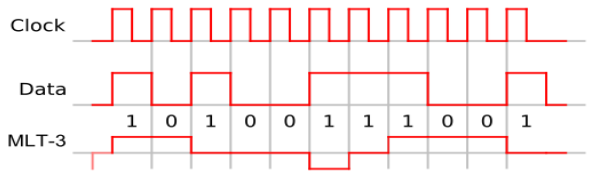
Füüsiline kiht (physical layer)

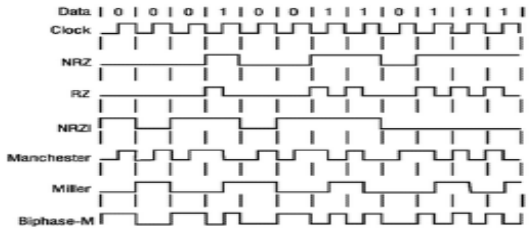
- **Eesmärgid:** Füüsiline kiht võimaldab toorandmebittide edastamist füüsilise meedia kaudu.
    a. Kahe seadme vaheline side, kandja kujundus, elektrisignaalide määratlemine
- **Minimaalne ühik: 1 bitt**
- **Riistvara Tugi:** See kiht kasutab kaableid, hube, retranslaatoreid, võrgukaarte ja modemeid andmeedastuseks.
    a. Cat5e või Cat6 Etherneti kaableid või Wi-Fi ruuter, switch, hub, Ethernet adapter, ADSL modem, Wi-Fi adapter
- **Operatsioonisüsteemi Tugi:** Operatsioonisüsteemid sõltuvad füüsilise kihi riistvarast ja püsivarast andmeedastuseks.
    a. Windows või Linux kasutab võrgukaardi draivereid füüsilise kihi funktsioonideks.
- **Draiverite Tugi:** Füüsilise kihi seadmed vajavad nende tööks spetsiifilisi draivereid.
    a. Intel Gigabit Etherneti võrgukaardi draiverid (e1000)
    b. Broadcom, Realtek, Atheros, Quallcom
- **API Tugi:** Füüsilise kihi API-d on tavaliselt madala taseme ja ei ole lõppkasutajale nähtavad.
    a. Ethernet controller low level API
- **Protokollid:** Füüsiline kiht tegeleb andmete edastamise füüsiliste standardite ja spetsifikatsioonidega, mitte kõrgema taseme protokollidega.
    a. RZ, NRZ, Manchesterm Miller, MLT-3



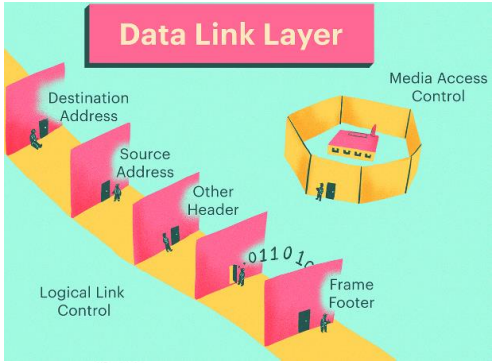
Cat5e Wire Diagram for T568B (Straight Through Cable)

RJ45 Pin #	Wire Color (T568A)	Wire Diagram (T568A)	10Base-T Signal 100Base-TX Signal	1000Base-T Signal
1	White/Orange		Transmit+	Bl_DA+
2	Orange		Transmit-	Bl_DA-
3	White/Green		Receive+	Bl_DB+
4	Blue		Unused	Bl_DC+
5	White/Blue		Unused	Bl_DC-
6	Green		Receive-	Bl_DB-
7	White/Brown		Unused	Bl_DD+
8	Brown		Unused	Bl_DD-

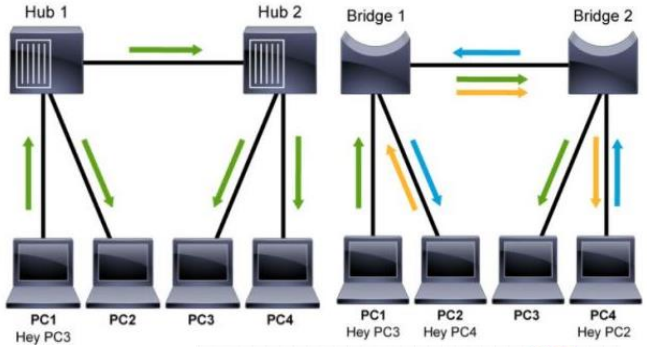


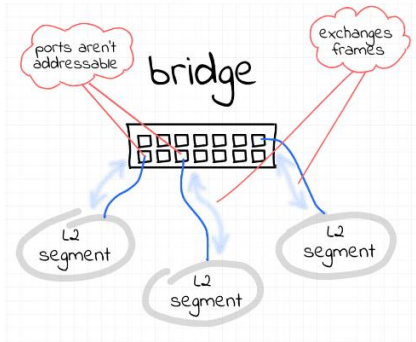


Kanalikiht (Link layer)



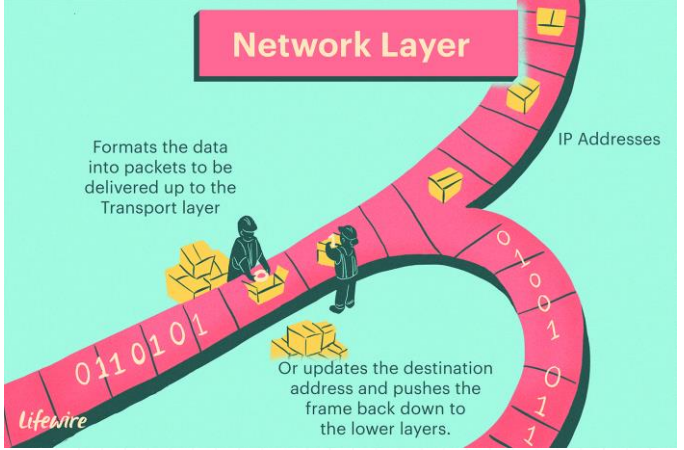
- **Eesmärgid:** Andmesidelinkkiht tagab otseühendatud sõlmede vahelise usaldusväärse andmeedastuse ja veaparanduse.
    a. Tüüpiline LAN over Ethernet: 1 switch ja mittu arvutit või 1 wi-fi AP ja mitu arvutit
- **Minimaalne ühik: 1 kaader (frame)**
- **Riistvara Tugi:** Kasutab võrgulüliteid, sildu ja võrgukaarte.
    a. Switch, ethernet adapter, wi-fi APN, wi-fi adapter
- **Operatsioonisüsteemi Tugi:** Operatsioonisüsteemid kasutavad andmesidelinkkihi jaoks spetsiifilisi draivereid ja tarkvara.
    a. Windows ja Linux kasutavad võrguadapteri draivereid linkkihi funktsioonide jaoks.
- **Draiverite Tugi:** Näiteks Etherneti lüliti draiverid.
    a. Intel, Broadcom, Realtek, Atheros, Quallcom
- **API Tugi:** Pakub API-sid võrgu seadmete juhtimiseks, nagu libpcap võrguliikluse jälgimiseks.
    a. libpcap - võimaldab võrguliikluse jälgimist ja analüüsimist.
- **Protokollid:** Andmesidelinkkihi protokollid hõlmavad Etherneti, PPP ja MAC aadressimist.
    a. Etherneti protokoll (IEEE 802.3) ja MAC (Media Access Control) aadressimine

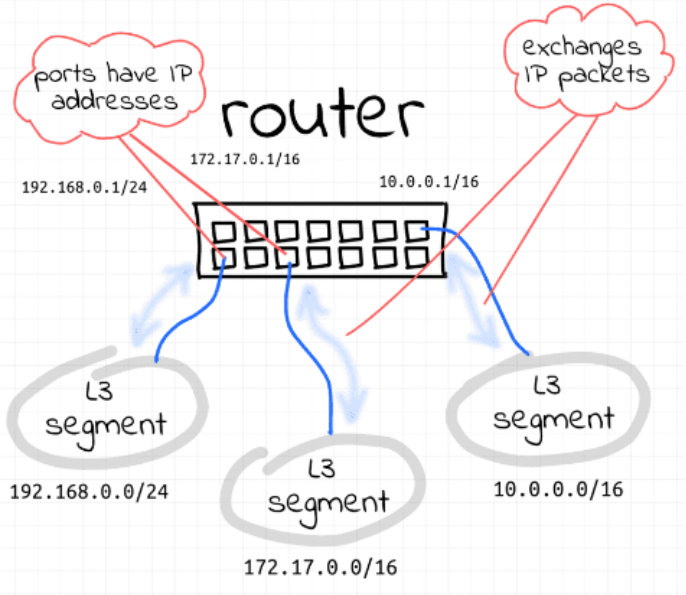




Võrgukiht (Network layer)

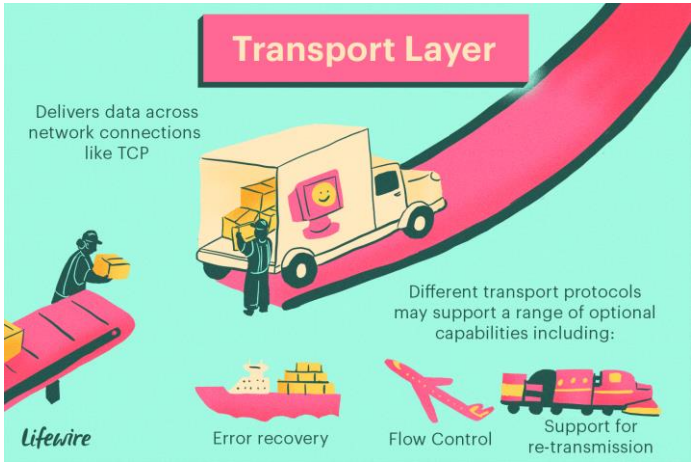
- Tee otsimine võrgus rohkem kui kahe seadme vahel
- Kommuteerimine ja marsruudi leidmine
  - Igale paketile eraldi
  - Ühe korra virtuaalse kanali loomisel
- Võrgusõlmede adresseerimine
- Pakettide edastamine erinevate võrkude vahel
- Pakettide tükeldamine ja kokkupanek





Transpordikiht (Transport layer)

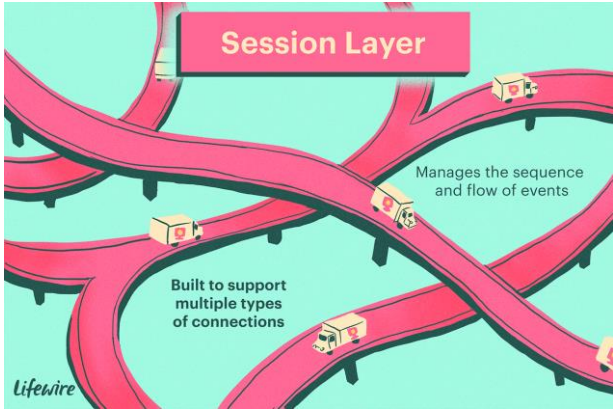
- Andmete läbipaistev transport kahe rakenduse vahel
- Vajadusel garanteerib andmete järjestuse
- Vajadusel garanteerib andmete uuestisaatmise
- Tegeleb otspunktide vahelise vookontrolliga
- Ummistuste lahendamine (congestion control)

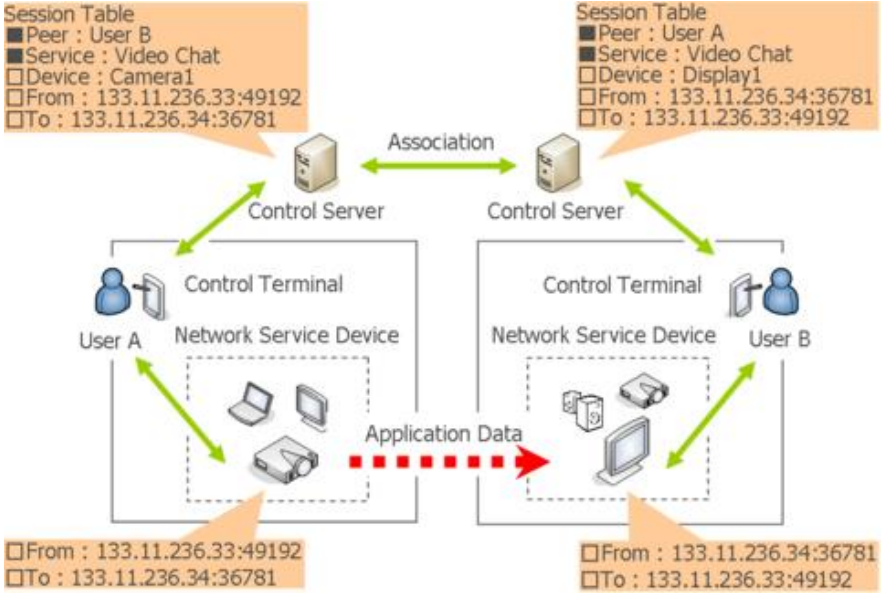


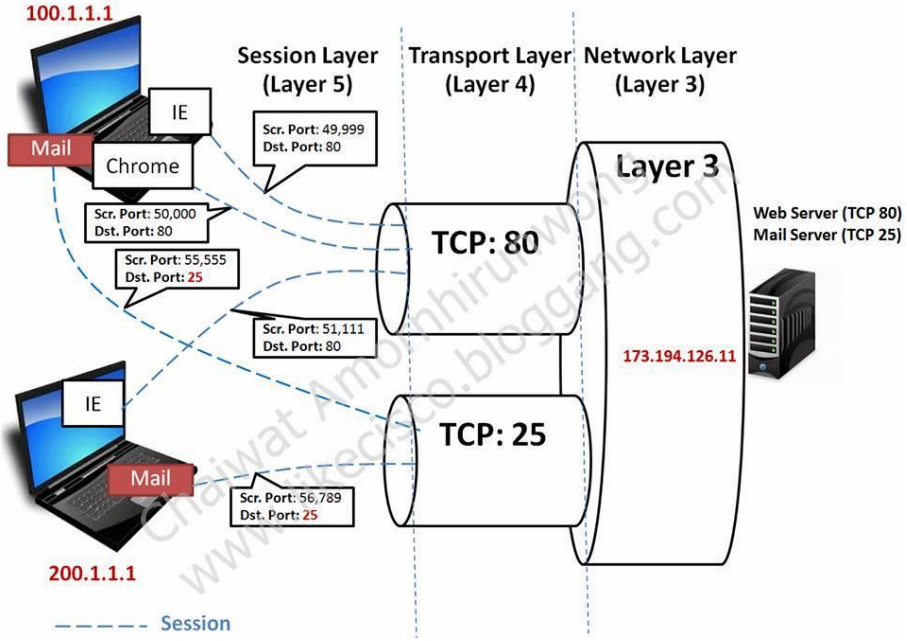
TCP	UDP
Secure	Unsecure
Connection-Oriented	Connectionless
Slow	Fast
Guaranteed Transmission	No Guarantee
Used by Critical Applications	Used by Real-Time Applications
Packet Reorder Mechanism	No Reorder Mechanism
Flow Control	No Flow Control
Advanced Error Checking	Basic Error Checking (Checksum)
20 Bytes Header	8 Bytes Header
Acknowledgement Mechanism	No Acknowledgement
Three-Way Handshake	No Handshake Mechanism
DNS, HTTPS, FTP, SMTP etc.	DNS, DHCP, TFTP, SNMP etc.

Seansikiht (Session layer)

- Seansihaldus kahe osapoole vahel:
  - Loob, haldab ja lõpetab loogilisi seansse
- Tegeleb ka seansside jätkamisega vea korral

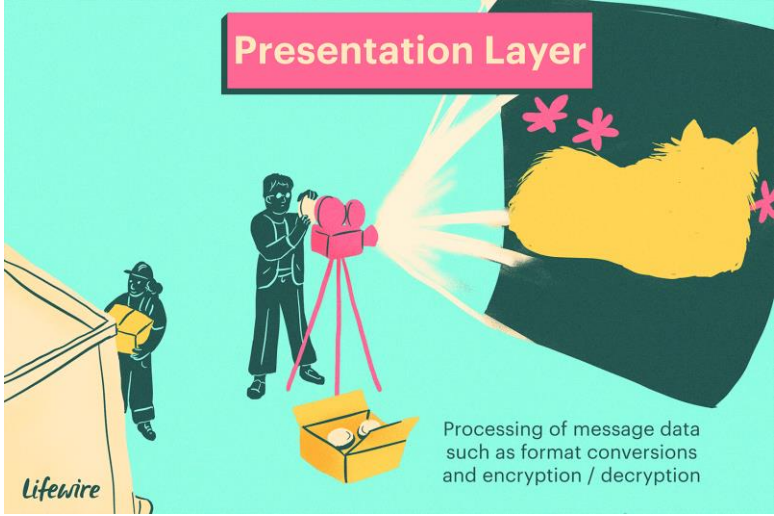


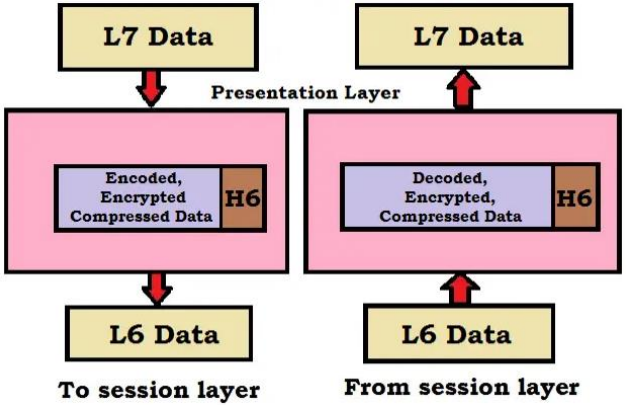


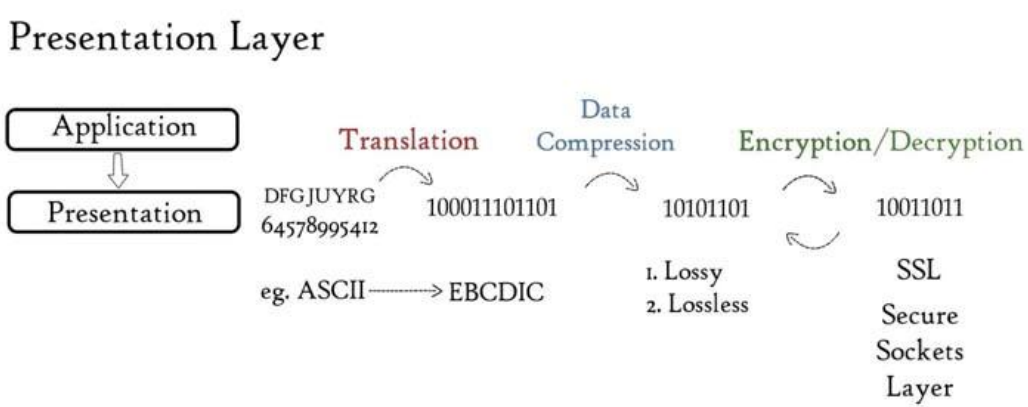


Esitluskiht (Presentation Layer)

- Andmete esituskujust sõltumatu tõlkekiht
- Tegeleb andmete kodeeringuga, struktuurse esitusega
- Krüpteerimine • Nn. süntaksikiht

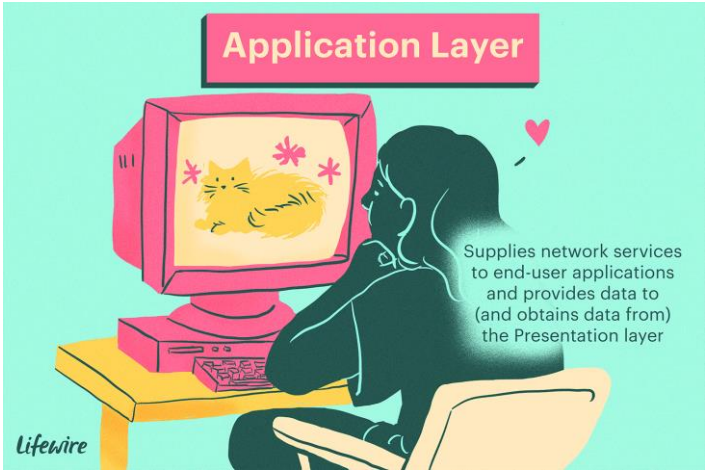


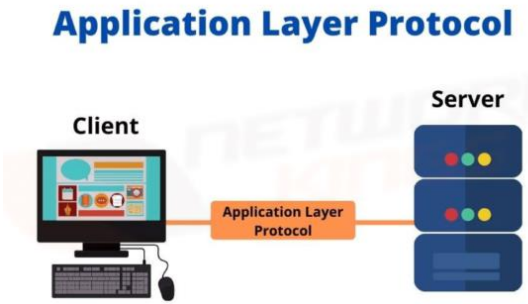


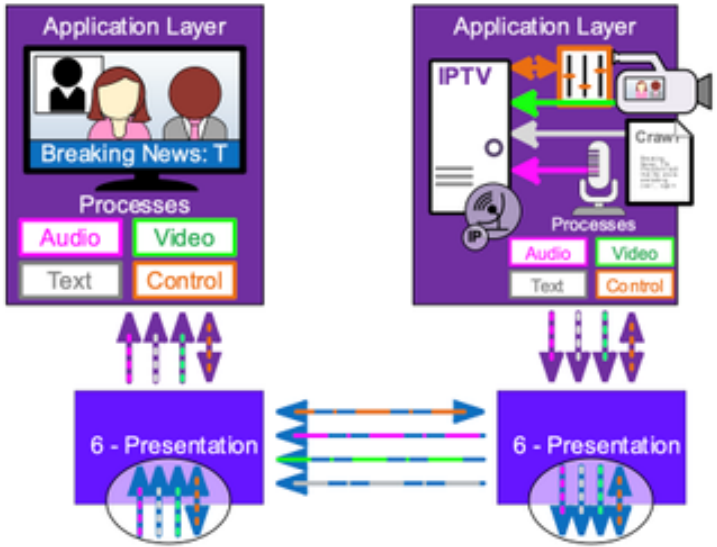


Rakenduskiht (Application layer)

- Rakendusprogrammide spetsiifilised protokollid
- Iga rakendus saab defineerida oma protokolli
- Kõik ülejäänud aspektid on rakenduskihi hallata



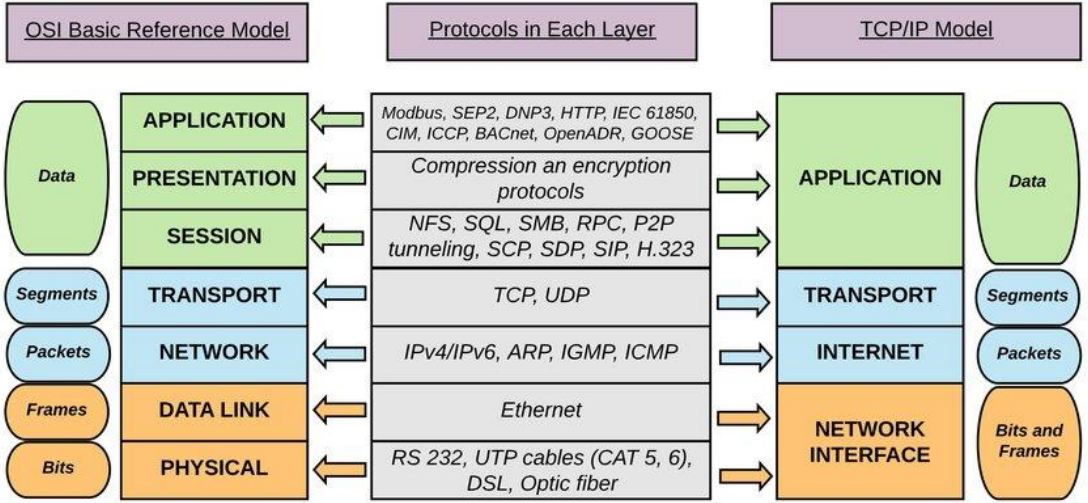




Internet ja OSI mudel

Füüsiline kiht — igasugused, näiteks Ethernet, WiFi

Kanalikiht — MAC aadressidega arvutite adresseerimine, Etherneti 802.3 kaadriformaat, ARP protokoll IP ja MAC vastavuse leidmiseks



Võrgukiht — IP: pakettide marsruutimine õigesse võrku

Transpordikiht: TCP (töökindel baidivoog), UDP (sõltumatute datagrammide saatmine)

Kolm ülemist kihti on kokku sulanud rakenduskihiks • Aegajalt on seansihaldust või esituskihi funktsionaalsust võimalik rakenduse protokollis eristada

Võrguliidesed

Ühel arvutil võib olla üks või mitu erinevat füüsilist võrguliidest

Lisaks on tavaliselt kasutusel lokaalne arvutisisene võrguliides (loopback)

Enamasti on igal liidesel oma aadress

- Mitmesse L3 võrku ühendatud arvutil peab olema igal liidesel aadress vastavast võrgust

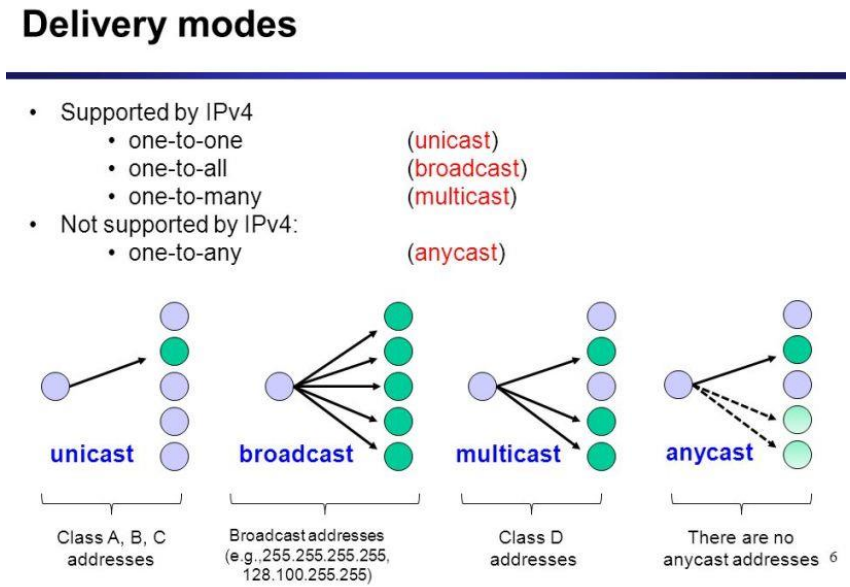
```
# Linux
me@myhost:~$ ip addr
…
me@myhost:~# ifconfig
…

# Windows
C:\Users\me> ipconfig
…
```

Mis on rakenduse ja mis OS-i realiseerida?

- Tavalahendus on realiseerida protokollivirna kihid 2-4 opsüsteemis
  - Rakenduste isoleerimine üksteise eest
  - Pordinumbrite hõivamine
  - Soklite jagamine protsesside vahel
- Alternatiiv: transpordikihi (4) kasutajarakendusse toomine
  - Vookontroll ja protokolli parsimine opsüsteemi seest protsessi sisse (end-to-end mudeli laiendamine parema skaleerimise huvides)
  - Opsüsteemi peab jääma minimaalne demultipleksimine ja jagatud ressursside haldus •
- Rakenduskiht on pea alati kasutaja tasemel, Internet mudelis koos sellega ka seansi- ja esituskiht

Edastusviisid

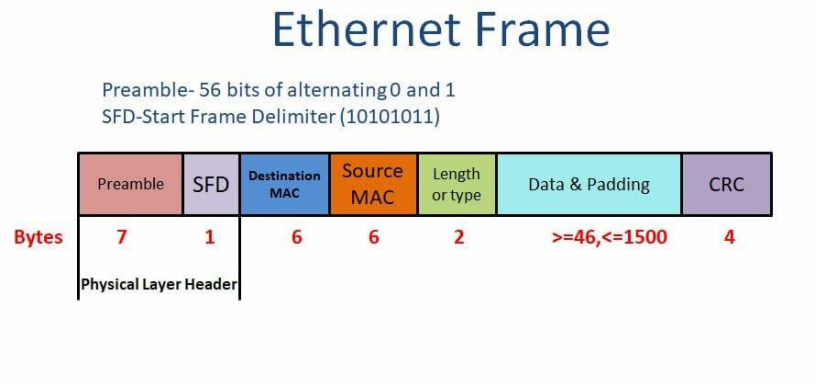
- Ainuedastus (unicast) — andmepaketi saatmine üle võrgu ühelt saatjalt ainult ühele vastuvõtjale
- Leviedastus (broadcast) — andmepaketi saatmine üle võrgu ühelt saatjalt kõigile võrgusõlmedele mingis piirkonnas
- Multiedastus (multicast) — andmepaketi saatmine üle võrgu ühelt saatjalt valitud vastuvõtjate rühmale
- Suvaedastus (anycast) — andmepaketi saatmine üle võrgu ühelt saatjalt rühma lähimale vastuvõtjale



Ethernet

Ethernet on tänapäeval protokollide pere, kus on sama kaadri formaat kuid erineva meedia ja kiirusega sidekanalid (kuni 100 Gbit/s seni standarditud)

- Etherneti kaader:
  - Preamble (fikseeritud baitide jadaa)
    - andmete edastuse alguses sünkroonida saatja ja vastuvõtja kellasid
  - Start frame delimiter
    - Kaadri alguse eristaja
    - selle väärtus on alati "10101011"
    - aitab vastuvõtjal eristada Preamble'it
  - Saaja MAC aadress
  - Saatja MAC aadress
  - Mittekohustuslik VLAN tag
    - (mitme loogilise võrgu tegemiseks
    - samas kaablis)
  - Andmeosa pikkus
  - Andmeosa (42-1500 baiti)
  - Kaadri kontrollsumma (32-bitine CRC)



MAC-aadress

- Teise arvuti adresseerimiseks 2. kihi (L2) võrgus peab tema aadressi teadma (näiteks MAC-aadress Etherneti-laadsete protokollide puhul)
- MAC — Medium Access Control
- 48-bitine idee poolest unikaalne seadme identifikaator
- Näiteks 00:0a:e4:7e:a5:e0
- Igal võrguliidesel on aadress tootja poolt sisse programmeeritud
- Koosneb tootja prefiksist ja unikaalsetest baitidest tootja piires
- MAC-aadress ei paista kohtvõrgust (LAN) kaugemale!

IP

- Internet Protocol
- Palju kohtvõrke on kokku ühendatud, IP-aadresside järgi leitakse tee läbi mitme võrgu õige seadmeni
- IPv4 kasutab 32-bitiseid aadresse
  - 127.0.0.1 = 01111111 00000000 00000000 00000001
  - 192.0.2.1 = 11000000 00000000 00000010 00000001
- IP-aadress jaguneb kaheks: võrguosa ja hostiosa
  - Võrguosa bitis on sama L2 võrgu piires kõigil hostidel samad
  - Hostiosa bitid tagavad sama L2 võrgu piires unikaalsuse

IP võrgumask

- Võrgumask näitabki, missugused bitid antud võrgus fikseeritud on
- Näiteks levinuim 255.255.255.0 = 3 baiti fikseeritud (/24)
- 255.255.254.0 = 11111111 11111111 11111110 00000000 = /23
- 255.255.255.240 = 11111111 11111111 11111111 11110000 = /28
- Igas võrgus on eritähendusega aadressid „kõik nullid“ (vanasti kasutusel leviaadressina) ja „kõik ühed“ (leviaadress tänapäeval)

Võrgumaski lihtne peast arvutamine

- Olgu meil võrgumask 255.255.255.224 kujul, tahame teada bittide arvu ja seda, mitu hosti on antud võrgus (koos leviaadressidega)
- Arvutus on lihtsalt peast tehtav:
  - 255 ja 0 väärtused on triviaalselt kõik 1-d või kõik 0-d, need saab baidi kaupa kokku arvutada
  - Neist erinev maski bait 224 = 256-32
  - Seega on antud võrgus 32 eri IP-d – 32 = 2^5 , seega on hostiosa pikkus 5 bitti
  - Maskis jääb võrguosale seega sellest baidist 8-5=3 bitti
  - Seega on numbriliseks maskiks /27 (sest 8+8+8+3=27).

ARP

- Kui L2 võrgus tahab arvuti IP-aadressiga Y arvutile IP-aadressiga X paketti saata, on vaja leida sihtarvuti MAC-aadress, et ainult talle saata teisi segamata
- Selleks on IPv4 juures ARP (Address Resolution Protocol)
- ARP päringu puhul kisab klient L2 leviaadressile päringu, et „Kes teist on X? Öelge seda palun Y-le.“
- Kui X enda kohta päringut kuuleb, vastab ta Y-le.
- Y saab nüüd X-le üle L2 IP-paketti saata.
- Lisaks hoiab Y seda kirjet mõne aja meeles oma ARP tabelis.

IP (mars)ruutimine

- (Mars)ruuterid edastavad liiklust mitme võrgu vahel
- Paketi teekonna igal sammul otsustab selle sammu ruuter, kuhu pakett edasi saata
- Selleks on ruutingutabelid sihtvõrkudega, kust valitakse iga paketi puhul sobivate hulgast kõige pikema maskiga kirje, näiteks:

sihtvõrk	mask	kuhu
192.168.0.0	255.255.255.0	eth0
192.168.1.0	255.255.255.0	10.0.0.2
192.168.0.0	255.255.0.0	10.0.0.5
10.0.0.0	255.255.255.0	eth1
0.0.0.0	0.0.0.0	10.0.0.1

IP-vahemike jaotamine

- Hierarhiline:
  - ISP saab suure ploki, näiteks /16 kuni /19
  - ISP jagab igale kliendile väiksema ploki, näiteks /24
  - Iga klient võib oma võrku jagada alamvõrkudeks, näiteks /28
- Klient võib osta otse teenusepakkujast sõltumatu IP ploki ja selle maailmale mitme ISP kaudu kättesaadavaks teha
  - Kõrgema käideldavuse jaoks
  - AS (Autonomous System)
  - osapool, kellel on otse oma teenusepakkujast sõltumatu IP-vahemik ja kes korraldab selle ühendamist teiste AS-ide kaudu

Ruutinguinfo levitamine

- Staatiline ruuting — seadmesse konfigureeritakse ruutingutabel administraatori poolt
- Dünaamiline ruuting — ruuterid vahetavad omavahel infot kättesaadavate võrkude kohta ja arvutavad ise, kust kaudu mingi võrk kõige otsem kättesaadav on
- „Core routers“ — Interneti tuumik, mis koosneb ruuteritest, mis teavad kõigi võrkude asukohti ilma vaikeruutingut kasutamata
- Sisemised ruutinguprotokollid (IGP — interior gateway protocol) — kasutamiseks organisatsiooni sees optimaalse tee leidmiseks, näiteks RIP ja OSPF
- Välised ruutinguprotokollid (EGP — exterior gateway protocol) — kasutamiseks organisatsioonide vahel suuremate plokkide granulaarsusega, näiteks BGP

IP-aadresside jagamine kohtvõrgus

- Staatiline — igale seadmele eraldatakse IP käsitsi ning konfigureeritakse seade seda kasutama
- Dünaamiline — seade saab käivitamisel omale ajutise aadressi automaatselt
- DHCP (Dynamic Host Configuration Protocol) — protokoll aadressi, domeeninime, staatiliste ruutingute ja muude parameetrite küsimiseks
- DHCP server teab, mis IP-aadress missugusele MAC-aadressile vastavuses on (seos võib olla ajutine või permanentne)

ICMP

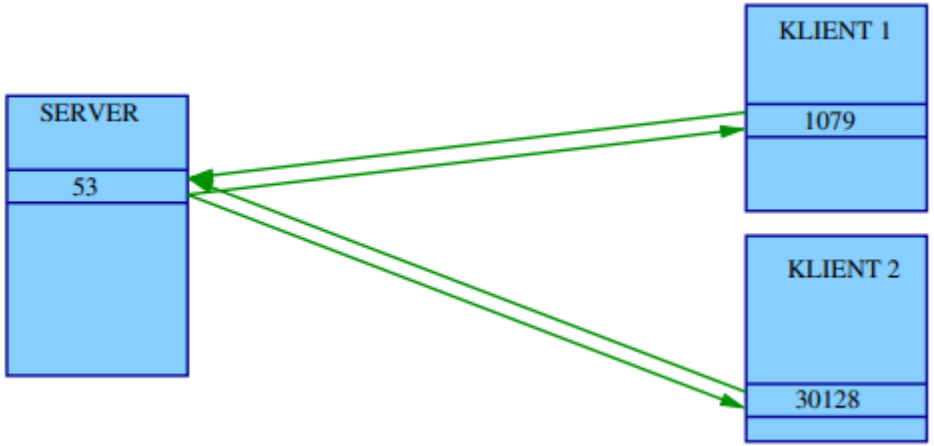
- Internet Control Message Protocol
- Ühelt arvutilt teisele saadetavate juhtsõnumite protokoll
- Näiteks:
  - echo request (ping)
  - echo reply (pong)
  - redirect (teine ruuter on otsem)
  - time exceeded
  - destination unreachable, network unreachable
  - destination unreachable, host unreachable
  - destination unreachable, administratively prohibited
  - destination unreachable, fragmentation needed but DF set (seda ei tohi filtreerida!)

UDP

- User Datagram Protocol
- Lähtearvuti mingilt protsessilt sihtarvuti mingile protsessile teate saatmine • Lihtne (ainult fragmenteerimine)
- Pole töökindel (iga pakett võib kaotsi minna)
- Olekuvaba (ei mingit korduvsaatmist ega järjestust) • Kiire (ei vaja puhverdamist)
- Sobib ka ühesuunaliseks suhtluseks (leviedastuse ja multiedastuse puhul) • UDP ise ei sisalda ummistuste vältimist (congestion control)
- Sobib nii lihtsate päring-vastus tüüpi protokollide kui reaalajavoogude jaoks

UDP port

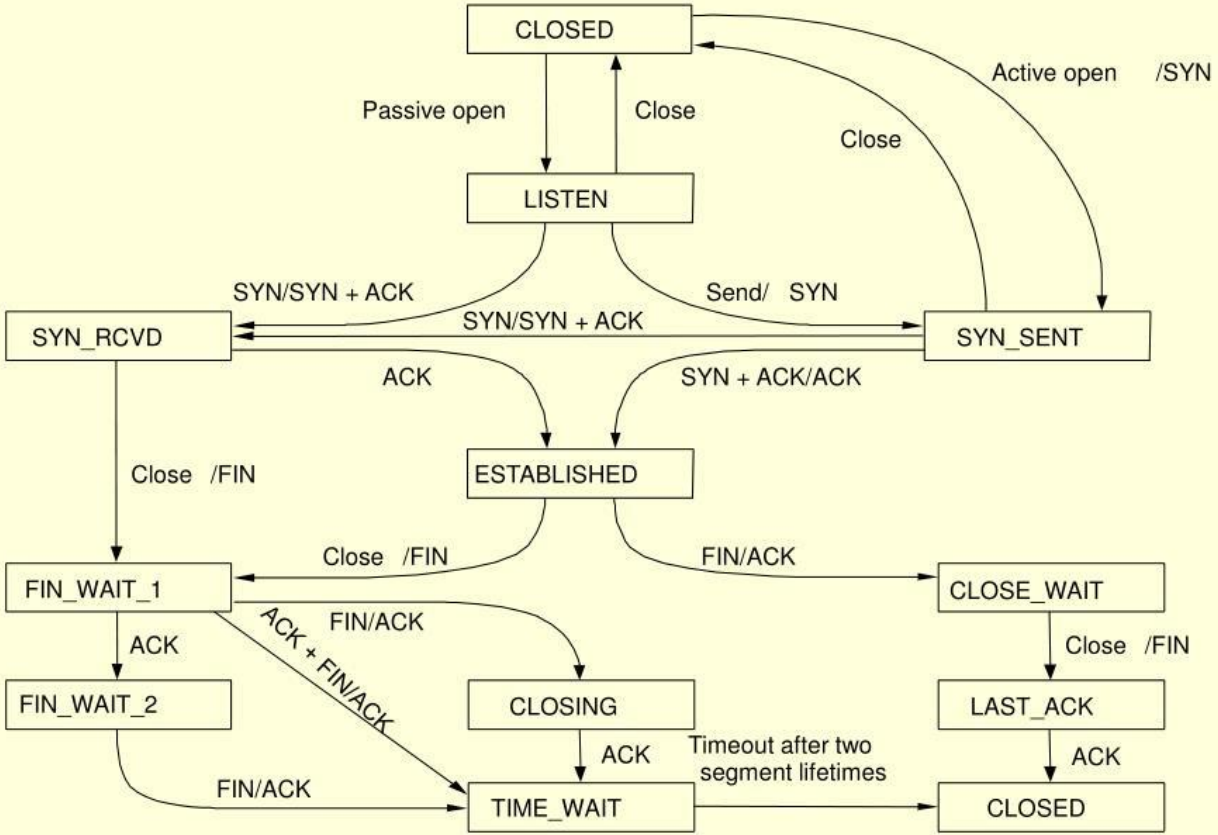
- Eristamaks arvutis mitut protsessi, on side kummaski otspunktis lisaks IP-aadressile kasutusel 16-bitised pordinumbrid
- Enamus UDP-d kasutavates rakendusprotokollides on teenuse pordinumber fikseeritud, kliendi pordinumber dünaamiline ning server vastab sellele kliendi pordile, kust päring tuli



TCP

- Transmission Control Protocol
- Lähtearvuti mingilt protsessilt sihtarvuti mingile protsessile baidivoo kahesuunaline edastamine
- Pakettide piirid pole fikseeritud, tükeldus võib teel muutuda
- Töökindel (iga andmetükki kviteeritakse, kaotsi minekul saadetakse mingi aja pärast uuesti)
- Järjestatud (andmed jõuavad rakendusele kohale samas järjekorras nagu teisest rakendusest saadeti, TCP puhverdab vajadusel kuni vahepealsed andmed kah kohal on)
- Raskekaalulisem kui UDP — vajab ühenduse loomist andmete saatmiseks, tegeleb ummistuste vältimisega
- Tegeleb vookontrolliga (et üks rakendus ei saadaks rohkem kui teine jõuab vastu võtta)

TCP State-Transition Diagram

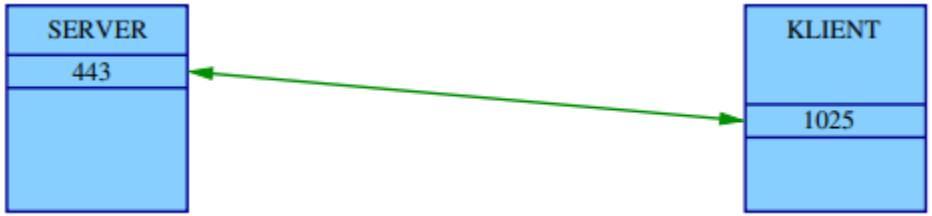


TCP detailid

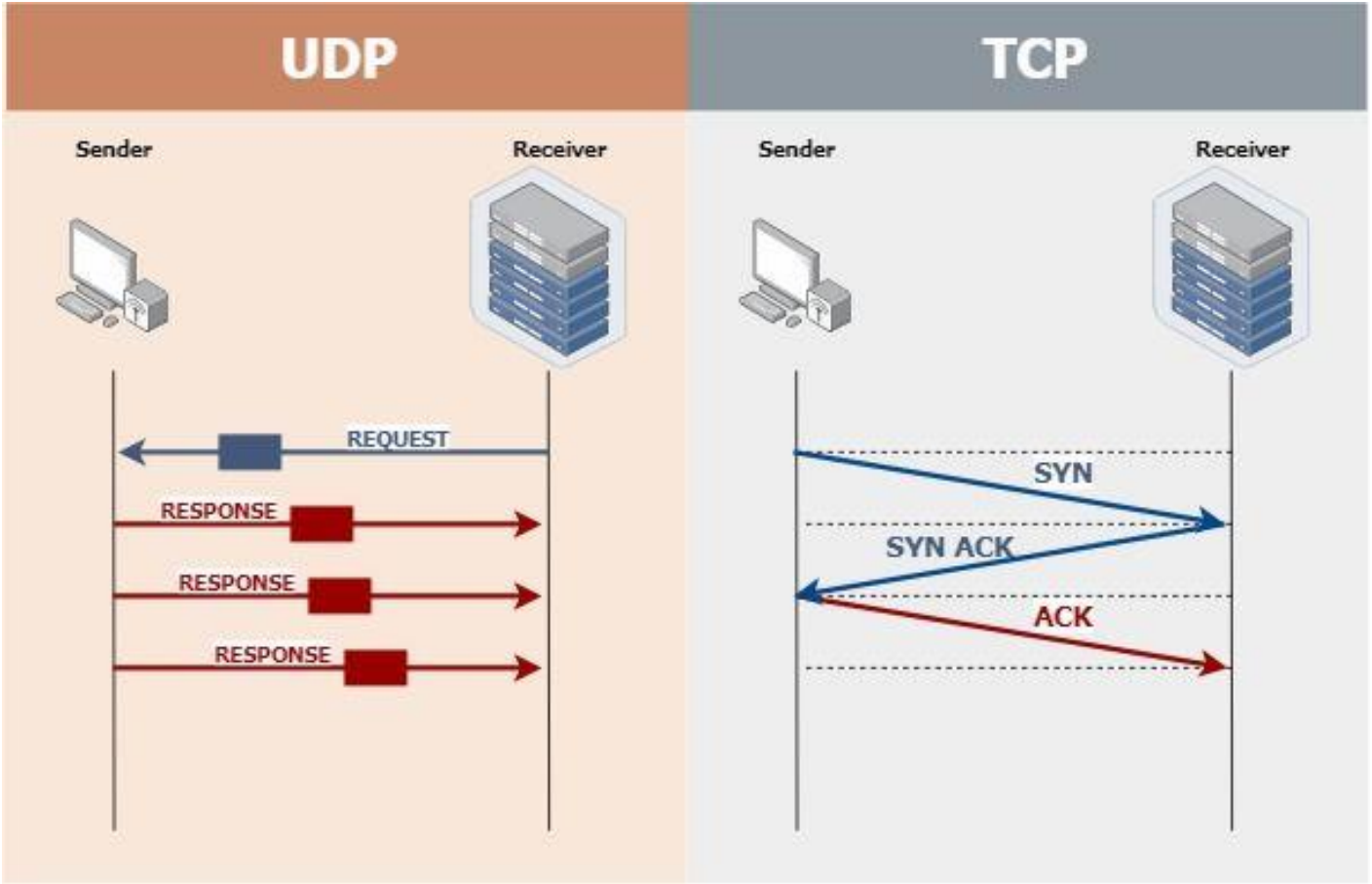
- Selgelt ainult kahele osapoolele (ainuedastus)
- Ühenduse loomine: SYN, ACK, SYN+ACK
- Aknaga protokoll (kui palju andmeid on korraga teel?) vookontrolliks
- Kummaski suunas sõltumatute loenduritega baidivoog
- Slow start — ühenduse või iga järgmise suurema mahu saatmise algul hakatakse kiirust järjest kasvatama, kuni mõõdetakse ära teekonna läbilaskevõime

TCP pordid

- Side kummaski otspunktis on protsesside eristamiseks lisaks IP-aadressile kasutusel 16-bitised pordinumbrid
- Kahe protsessi vahel võib olla mitu sõltumatut TCP ühendust erinevate kliendiportidega
- Igal teenusel on enamasti fikseeritud serveri pordinumber • Kliendi pordinumber on tavaliselt juhuslik



TCP vs. UDP sequence diagrams



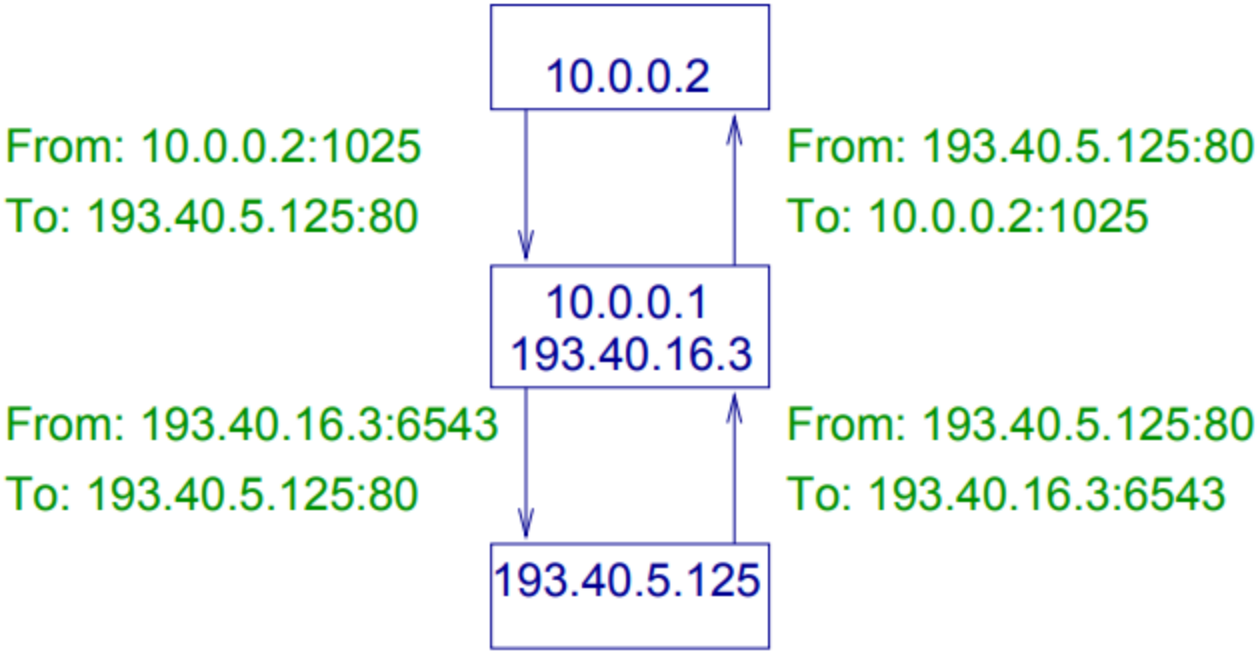
Võrguaadresside tõlkimine — NAT

- IPv4 aadressidega on kitsas käes, vaja on aadresside kasutamist optimeerida
- Tahame sisevõrgu struktuuri teiste eest ära peita
- Tahame, et sisevõrgu masinad ei oleks väljast otse nähtavad
- Lahendus(?): kasutame sisevõrgus privaataadresse, mis Internetis ei esine • Vahel on siiski vaja pakette sise- ja välisvõrgu vahet liigutada
- Lahenduseks on aadresside tõlkimine ruuteris. Tõlkimist on kolme moodi:
    - Staatiline: n – n — tõlgitakse terve aadressiplokk
    - Dünaamiline: n – m, m < n — avalikke aadresse on vähem
    - Tõlkimine porte kasutades: n – 1 — kõik siseaadressid 1 välisaadressiks, varieeritakse lähtepordi numbrit

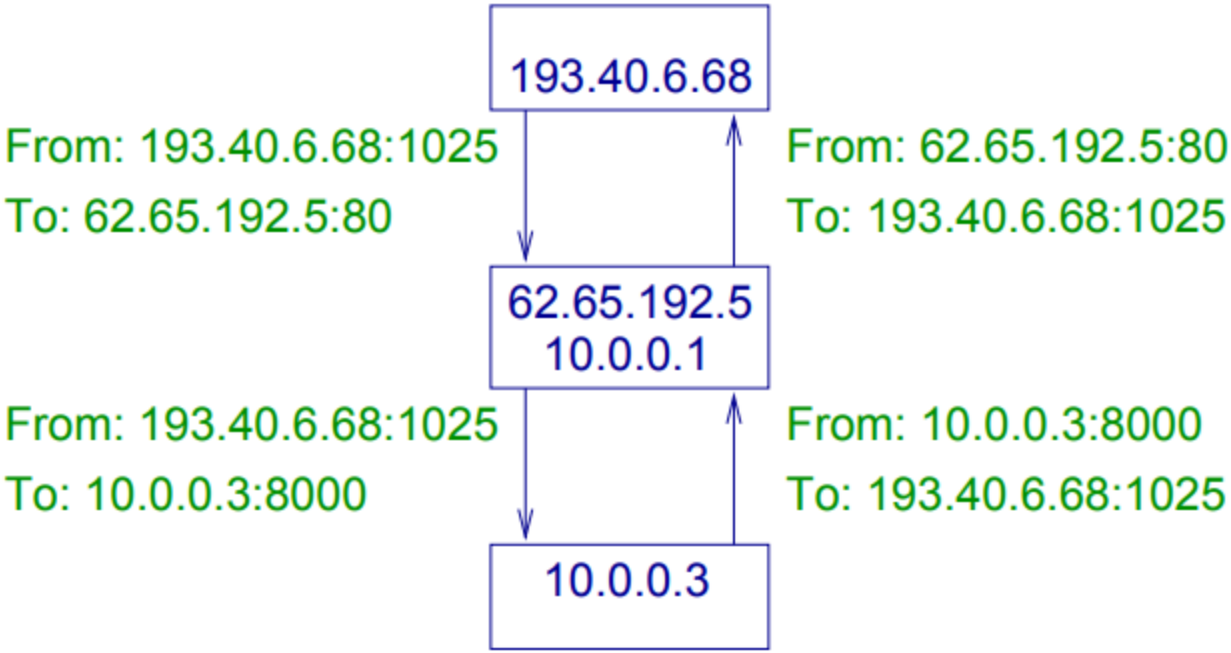
NAT tehnoloogia

- Standarditega on paika pandud aadressivahemikud, mida võib vabalt oma sisevõrkudes kasutada:
  - 10.*.*.*
  - 172.16.*.*
  - 172.31.*.*
  - 192.168.*.*
- Neid aadresse Internetis ei ruudita
- Aadresse tõlkiv ruuter modifitseerib ühe osapoole IP aadressi (tõlgib ühe suuna andmed ning tõlgib tagasi vastused)
- Lähteaadressi maskeerimise abil saame varjata klientarvutit (algatajat) → SNAT
- Sihtaadressi maskeerimise abil saame varjata serverarvutit → DNAT
- SNAT ja DNAT võib samas ruuteris ka järjest rakendada

SNAT näide


10.0.0.2
From: 10.0.0.2:1025
To: 193.40.5.125:80
From: 193.40.5.125:80
To: 10.0.0.2:1025
10.0.0.1
193.40.16.3
From: 193.40.16.3:6543
To: 193.40.5.125:80
From: 193.40.5.125:80
To: 193.40.16.3:6543
193.40.5.125

DNAT näide


193.40.6.68
From: 193.40.6.68:1025
To: 62.65.192.5:80
From: 62.65.192.5:80
To: 193.40.6.68:1025
62.65.192.5
10.0.0.1
From: 193.40.6.68:1025
To: 10.0.0.3:8000
From: 10.0.0.3:8000
To: 193.40.6.68:1025
10.0.0.3

NAT probleemid

- Teeb katki TCP/IP mudeli, kus ainult ühenduse otspunktid teavad detaile
- Sunnib peale mingi osaliselt fikseeritud marsruudi otspunktide vahel
- Toob sisse ühe katki mineku punkti
- Toob sisse ühildumatuse paljude protokollidega
- Ei lahenda IPv4 aadresside kitsikust
- AGA:
  - Leevendab IPv4 aadresside kitsikust
  - Aitab lihtsalt ja praktiliselt võrku turvalisemaks teha

IPv6

- Uus võrgukihi protokoll 128-bitiste aadressidega:
  - 2001:bb8:2002:2400:209:3dff:fe11:e8c5/64
  - 2002:5abf:a1ac:2::1/64 – ::1/128
- ARP asemel multiedastusel põhinev neighbour discovery
- Samad TCP, UDP ja enamus rakendustaseme protokolle
- Uued ICMPv6, DHCPv6
- Lisaks DHCPv6-le võimalus olekuvabalt aadresse konfigureerida
- Igale kohtvõrgule /64, ruuter reklaamib prefiksit
- Host paneb oma aadressi kokku prefiksist ja 64-bit kohalikust osast (MAC või juhuslik)

IPv6 üleminek

- Ülemineku ajaks võetakse enamus võrkudes kasutusele nii IPv4 kui IPv6 aadressid
- Protokollid IPv6 tunneldamiseks üle IPv4 (automaatne tunneldamine: 6to4, Teredo, ISATAP)
- API tasemel suudab IPv6 sokkel teenindada ka IPv4 ühendusi
  - IPv4-mapped aadressid ::193.40.36.2 kujul IPv4-aadresside tähistamiseks API-s (aga mitte „traadil“)
  - Vastupidi ei saa
- IPv6 puhul on NAT tugevalt vastunäidustatud — aadresse jätkub, väldime NAT probleeme kui võimalik

Soklid

- Sokkel (socket) — programmeerimisliides (API) võrguga suhtlemiseks
- Sokkel on sidekanali otspunkt (näiteks TCP ühenduse kummaski otsas on vastaval rakendusel sokkel)
- Sokli-API on protokollist sõltumatu, disainitud OSI mudeli järgi
- Pärit BSD Unixist, tänapäeval üldlevinud (+kohalikud täiendused eri OS-ides)
- Toetab palju protokolliperekondi
- Toetab voosokleid (stream socket) ja paketisokleid (datagram socket)
- Blokeeruvad ja mitteblokeeruvad soklid
- Lisaks nimelahendus (gethostaddr() jt, Interneti puhul kasutab DNS)

Soklite näited

- Kombineerime protokolliperekonna ja sokli tüübi:
  - PF_INET, SOCK_STREAM — TCPv4
  - PF_INET6, SOCK_STREAM — TCPv6
  - PF_INET, SOCK_DGRAM — UDPv4
  - PF_UNIX, SOCK_STREAM — Unixi sisene stream-sokkel protsesside vahel
  - PF_RAW, SOCK_DGRAM — Etherneti kaadrid toorkujul

Soklite API

- socket()
- connect()
- send(), recv()
- sendto(), recvfrom()
- shutdown(), close()
- bind()
- listen()
- accept()
- ioctl()

Soklinäide: TCP

server	klient
socket()	
bind()	
listen()	
	socket()
	connect()
accept()	
send()	recv()
recv()	send()
. . .	. . .
close()	close()

Soklinäide: tavaline UDP

server	klient
socket()	
bind()	
	socket()
	bind()
recvfrom()	sendto()
sendto()	recvfrom()
. . .	. . .
	close()

Soklinäide: UDP ja sokli ühendamine

server	klient
socket()	
bind()	
	socket()
	connect()
recvfrom()	send()
sendto()	recv()
. . .	. . .
	close()

Küsimused ?

Riistavarasuhtlus

Plaan

- Riistvarasuhtlus (pordid, DMA, . . .)
- Katkestused
- Süsteemsed siinid
- Arvutivälised siinid ja pordid
- Salvestusseadmete liidesed
- Rakenduste liidesed sisendi ja väljundi jaoks

Storage	latency	
L1 cache reference	0.5	ns
Branch mispredict	5	ns
L2 cache reference	7	ns
Mutex lock/unlock	25	ns
Main memory reference	100	ns
Compress 1K bytes with Zippy	3,000	ns
Send 1K bytes over 1 Gbps network	10,000	ns
Read 4K randomly from SSD	150,000	ns
Read 1 MB sequentially from memory	250,000	ns
Round trip within same datacenter	500,000	ns
Read 1 MB sequentially from SSD	1,000,000	ns
Disk seek	10,000,000	ns
Read 1 MB sequentially from disk	20,000,000	ns
Send packet CA->Netherlands->CA	150,000,000	ns

Port I/O (Portsisend/Väljund)

- Port I/O hõlmab konkreetsete riistvaraaadresside (portaadresside) kasutamist välisseadmetega suhtlemiseks.
- Selles meetodis on I/O toimingute jaoks reserveeritud kindel hulk aadresse. Need aadressid on sageli eraldi aadressiruumis põhivahemälu suhtes.
- Riistvaraseadme lugemiseks või kirjutamiseks saadab CPU käsklusi ja andmeid nendesse konkreetsetesse portaadressidesse.
- Port I/O-d kasutatakse sageli x86 arhitektuuris ja seda saab tihti juurde pääseda kasutades asembleri keele IN ja OUT käske.

Näide: DOS-i INT 13h BIOS-i katkestus

Näide sellest, kuidas saate DOS-i assembleris kasutada INT 13h BIOS-i katkestust ekraanirežiimi initialiseerimiseks:

; Seome AH register 00h Video Teenuste funktsiooni jaoks

mov ah, 00h

; Seome AL register soovitud ekraanirežiimile (näiteks 03h 80x25 tekstirežiimiks)

mov al, 03h

; Kutsume INT 10h, et seada ekraanirežiim

int 10h

Memory Mapped I/O (Mälukaardistatud Sisend/Väljund)

- Mälukaardistatud I/O integreerib I/O seadmed süsteemi mälu aadressiruumi.
- Selle asemel, et olla pühendatud porti aadressidele, kaardistatakse riistvaraseadmed konkreetsetele mäluaadressidele.
- Seadmega suhtlemiseks loete või kirjutate mäluaadressidele, mis on reserveeritud selle seadme jaoks.
- Mälukaardistatud I/O lihtsustab programmeerimismudelit, käsitledes I/O seadmeid nagu mäluaadresse, võimaldades nende juurdepääsuks tavapäraseid laadimis- ja salvestamistoiminguid.
- See lähenemine on levinud kaasaegsetes arvuti arhitektuurides ja mikrokontrollerites.

Näide: set the display mode using Memory-Mapped I/O (MMIO) in C

```
// Open /dev/mem to access physical memory
int mem_fd = open("/dev/mem", O_RDWR | O_SYNC);
if (mem_fd == -1) {
    perror("Error opening /dev/mem");
    return 1;
}
```

Näide: set the display mode using Memory-Mapped I/O (MMIO) in C

```
// Map the physical memory to a user-space pointer
void* mmio_base =
     mmap(NULL, getpagesize(), PROT_READ | PROT_WRITE, MAP_SHARED, mem_fd,
                    GRAPHICS_BASE_ADDRESS);
if (mmio_base == MAP_FAILED) {
    perror("Error mapping memory");
    close(mem_fd);
    return 1;
}
```

Näide: set the display mode using Memory-Mapped I/O (MMIO) in C

```
// Access and modify the control and mode registers
volatile uint32_t* control_register = (volatile uint32_t*)((char*)mmio_base + CONTROL_REGISTER_OFFSET);
volatile uint32_t* mode_register = (volatile uint32_t*)((char*)mmio_base + MODE_REGISTER_OFFSET);

// Set the desired display mode (replace 0x03 with the desired mode)
*mode_register = 0x03;

// Unmap the memory and close /dev/mem
munmap(mmio_base, getpagesize());
close(mem_fd);
```

Andmete Ülekandmine

- Programmeeritud I/O (PIO)
- Otseste Mäljuurdepääsude (DMA)

kaks erinevat meetodit andmete ülekandmiseks perifeeriaseadmete (näiteks salvestusseadmete, võrguliideste) ja arvuti peamälu vahel

PIO vs. DMA Juhtimismehhanism

Programmeeritud I/O (PIO):

- Programmeeritud I/O puhul vastutab CPU andmeülekannete juhtimise eest perifeeriaseadmete ja mälu vahel.
- CPU väljastab käsklused perifeeriaseadmele, ootab, kuni perifeeriaseade toimingu lõpetab, ja seejärel loeb või kirjutab andmeid.
- See meetod on lihtne rakendada, kuid võib olla ebaefektiivne, kuna see seob CPU andmeülekannete ülesannetega.

PIO vs. DMA Juhtimismehhanism

DMA (Otseste Mäljuurdepääsude):

- DMA on riistvaral põhinev mehhanism, mis võimaldab perifeeriaseadmetel otse andmeid mälusse lugeda või sealt kirjutada ilma CPU sekkumiseta.
- DMA kontroller vastutab andmeülekannete haldamise eest perifeeriaseadme ja mälu vahel.
- CPU seadistab DMA kontrolleri lähteaadresside ja sihtkoha ning muude parameetritega ning seejärel laseb DMA kontrolleril tegeleda tegeliku andmeülekandega.
- DMA võib oluliselt vähendada CPU ülekoormust ja parandada andmeülekande efektiivsust.

PIO vs. DMA CPU sekkumine

Programmeeritud I/O (PIO):

- CPU on andmeülekande igas etapis aktiivselt kaasatud.
- CPU väljastab käsklusi, jälgib perifeeriaseadme olekut ja liigutab andmeid.

DMA (Otseste Mäljuurdepääsude):

- CPU seab üles DMA kontrolleri ja algatab ülekande, kuid ei ole andmeülekande käigus aktiivselt kaasatud.
- CPU saab tegeleda teiste ülesannetega, samal ajal kui DMA kontroller tegeleb andmeülekandega.

CPU vs. DMA Efektiivsus

Programmeeritud I/O (PIO):

- PIO võib olla vähem efektiivne, kuna see nõuab CPU pidevat tähelepanu, mis võib põhjustada kõrget CPU kasutust ja aeglast üldist süsteemi jõudlust.

DMA (Otseste Mäljuurdepääsude):

- DMA on efektiivsem, kuna see võtab andmeülekandetööd CPU-lt üle. CPU saab tegeleda teiste ülesannetega, samal ajal kui DMA kontroller tegeleb andmeülekannetega, mis viib parema süsteemi jõudluseni, eriti kõrge kiirusega andmeülekannete stsenaariumides.

PIO vs. DMA Kasutusjuhtumid

Programmeeritud I/O (PIO):

- PIO sobib olukordades, kus lihtsus on tähtsam kui jõudlus, näiteks madala kiirusega perifeeriaseadmete suhtluses.

DMA (Otseste Mäljuurdepääsude):

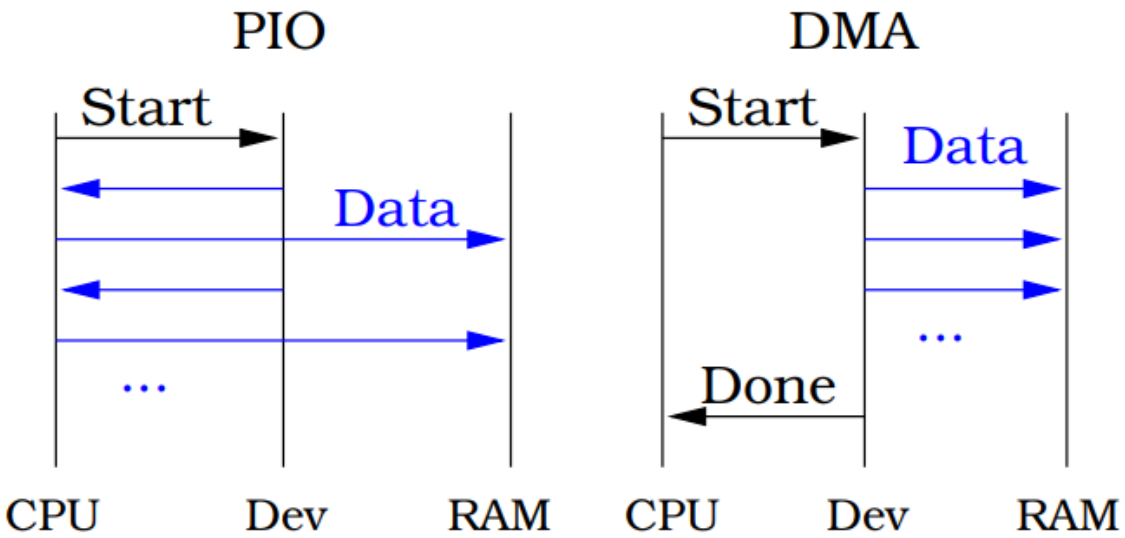
- DMA on ideaalne kõrge kiirusega andmeülekannete, nagu suurte andmeblokkide lugemine või kirjutamine salvestusseadmetesse, andmete edastamine võrgus või heli- ja videovoogude haldamiseks.

PIO vs. DMA

Kokkuvõttes toetub Programmeeritud I/O CPU-le andmeülekannete juhtimiseks perifeeriaseadmete ja mälu vahel, samal ajal kui DMA kasutab spetsiaalset DMA kontrollerit, et ülesandeid CPU-lt üle võtta, mis parandab efektiivsust ja süsteemi jõudlust, eriti kõrge kiirusega andmeülekannete stsenaariumides.

Riistvarasuhtlus

- Port IO vs MMIO (Memory Mapped I/O)
- Seadme registrid
- Vookontroll
- PIO (Programmed I/O) vs DMA / DVMA



Pollimine vs. katkestamine

Pollimine ja katkestused on kaks meetodit, mida kasutatakse arvutisüsteemides välise sündmuse või asünkroonsete sündmustega tegelemiseks.

Pollimine

Pollimine on meetod, kus CPU kontrollib pidevalt seadme või lipu olekut, et näha, kas see vajab tähelepanu.

Pollimisel küsitleb CPU pidevalt konkreetset riistvara registrit või tingimust, et teha kindlaks, kas teatud sündmus on toimunud või kas seade on valmis.

See hõlmab tavaliselt tsüklite ja tingimuslauseid kasutamist sündmuse oleku regulaarseks kontrollimiseks.

Pollimine võib olla lihtne ja otsekohene viis sündmuste käsitlemiseks, kuid see võib olla ebaefektiivne, kuna see tarbib CPU aega, eriti harvaesinevate sündmuste korral.

See võib põhjustada aktiivset ootamist, kus CPU kontrollib pidevalt ilma kasuliku töö tegemiseta, kuni sündmus toimub.

Katkestused

Katkestused on mehhanism, kus välised seadmed või sündmused saavad katkestada normaalse programmi käigu ja paluda CPU-lt kohest tähelepanu.

Katkestustel põhinevas lähenemisviisis saadavad seadmed või sündmused signaale CPU-le, et teavitada teda vajadusest teenust või tähelepanu.

Kui katkestus toimub, peatab CPU oma praeguse töö, salvestab oma oleku ja hüppab eelnevalt määratletud katkestusteenindusrutiini (ISR) juurde, mis tegeleb konkreetse katkestusega.

ISR-id on väikesed koodilõigud, mis on mõeldud konkreetsete katkestuste käitlemiseks, ja need käivituvad ainult vastava katkestuse korral.

Katkestused on efektiivsemad kui pollimine, kuna need võimaldavad CPU-l oodata sündmuste toimumist ja samal ajal teisi ülesandeid täita. Need minimeerivad raisatud CPU tsükleid.

Pollimine vs. katkestamine

Efektiivsus: Katkestused on üldiselt efektiivsemad kui pollimine, kuna need võimaldavad CPU-l olla jõude, kuni sündmus toimub, vähendades tarbetut CPU kasutust. Pollimine võib olla vähem efektiivne, kuna see kontrollib sündmusi pidevalt ja võib potentsiaalselt raisata CPU tsükleid.

Reaalajas reageerimine: Katkestused sobivad hästi reaalajas süsteemidele, kuna need võivad pakkuda välisele sündmusele kohest reageerimist. Pollimine võib viia viivitusteni, kuna CPU ei pruugi sündmuse olekut kohe kontrollida.

Komplekssus: Pollimine võib olla lihtsam rakendada ja mõista, eriti lihtsates süsteemides. Katkestused nõuavad keerulisemat käitlemist ja võivad hõlmata ülesannete vahel ümberlülitumist.

Ressursside kasutamine: Pollimine tarbib pidevalt CPU ressursse, samal ajal kui katkestused tarbivad ressursse ainult siis, kui tegelik sündmus toimub, muutes katkestused ressursisäästlikumaks.

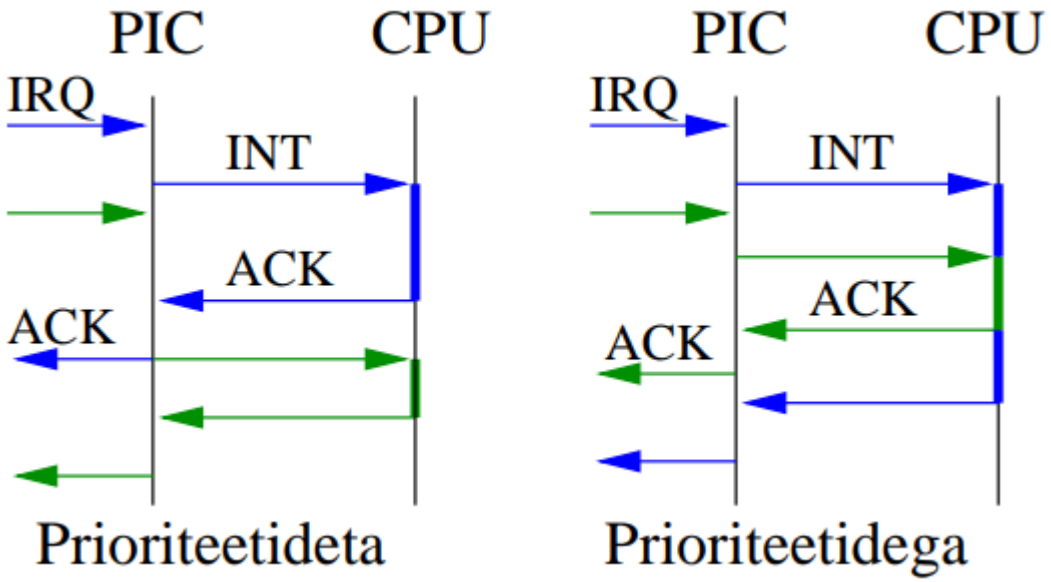
Pollimine vs. katkestamine

Kokkuvõttes hõlmab pollimine CPU aktiivset olekut sündmuste või seadmete oleku kontrollimisel, samas kui katkestused võimaldavad välissündmustel katkestada CPU tavapärase täitmise, mis viib efektiivsema ja reageerivama asünkroonsete sündmuste käsitlemise poole. Valik pollimise ja katkestuste vahel sõltub konkreetsetest süsteemi nõuetest ja omadustest.

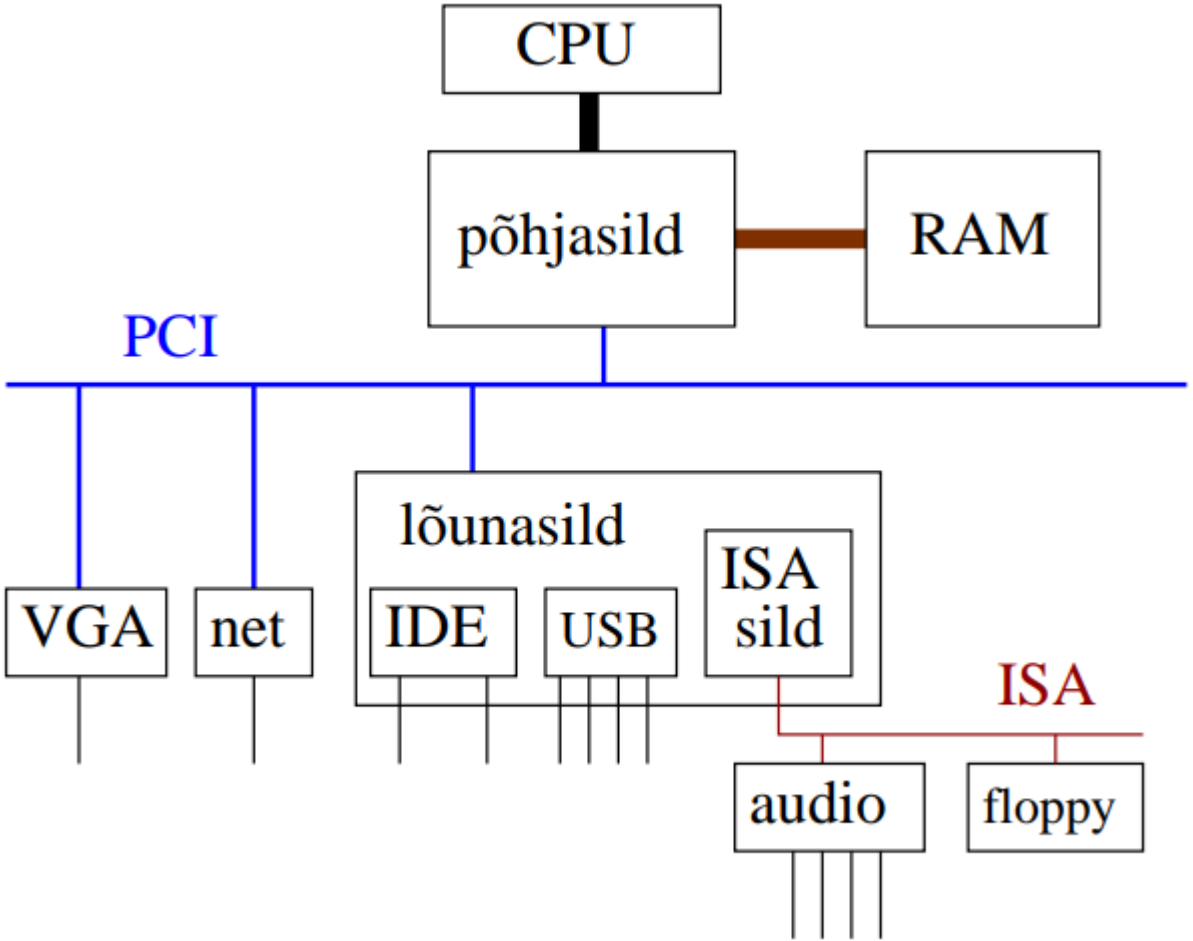
Katkestuste prioriteedid

Võime erinevatele katkestustele anda erinevad prioriteedid

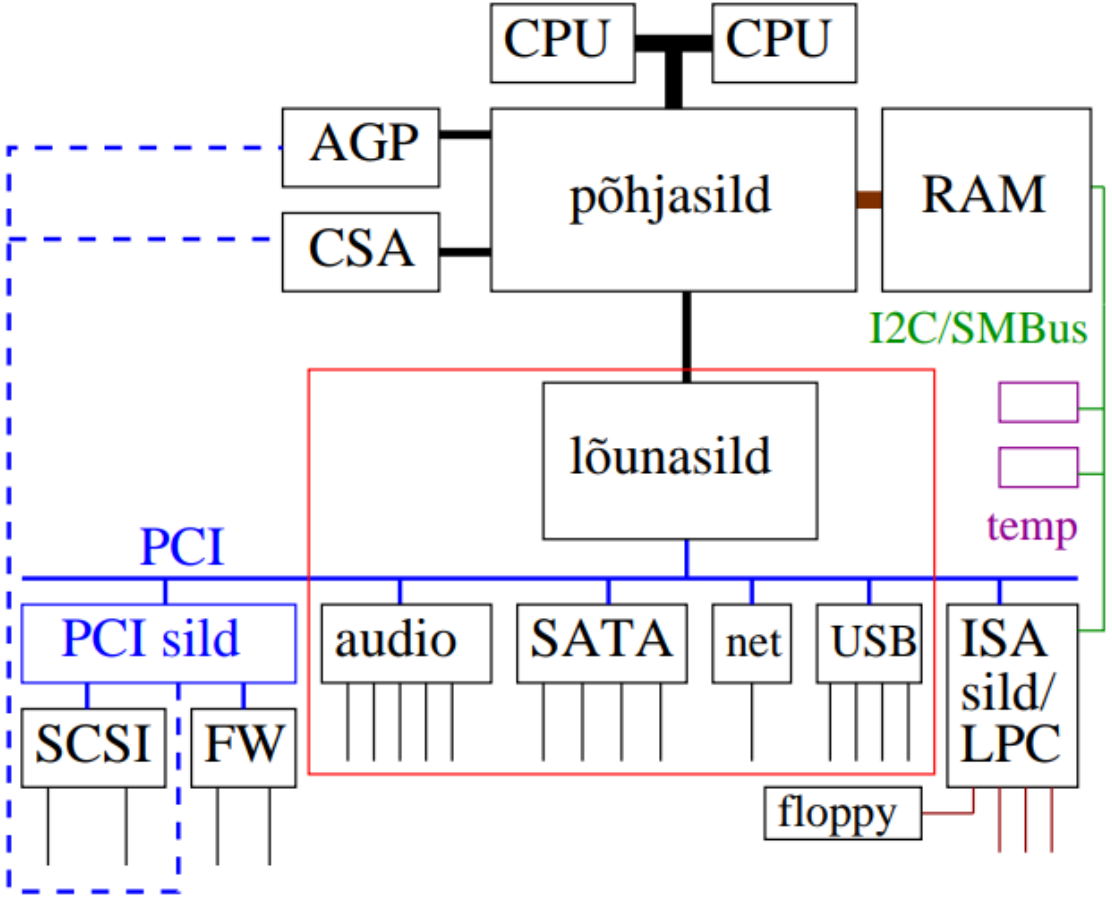
- Suurema prioriteediga katkestus võib vahele segada väiksema prioriteediga katkestuse teenindamisele
- Prioriteetidega katkestused kui vahend vookontrolliks tuumas
- Prioriteetide programmeerimine katkestuste kontrollerisse on lisakulu



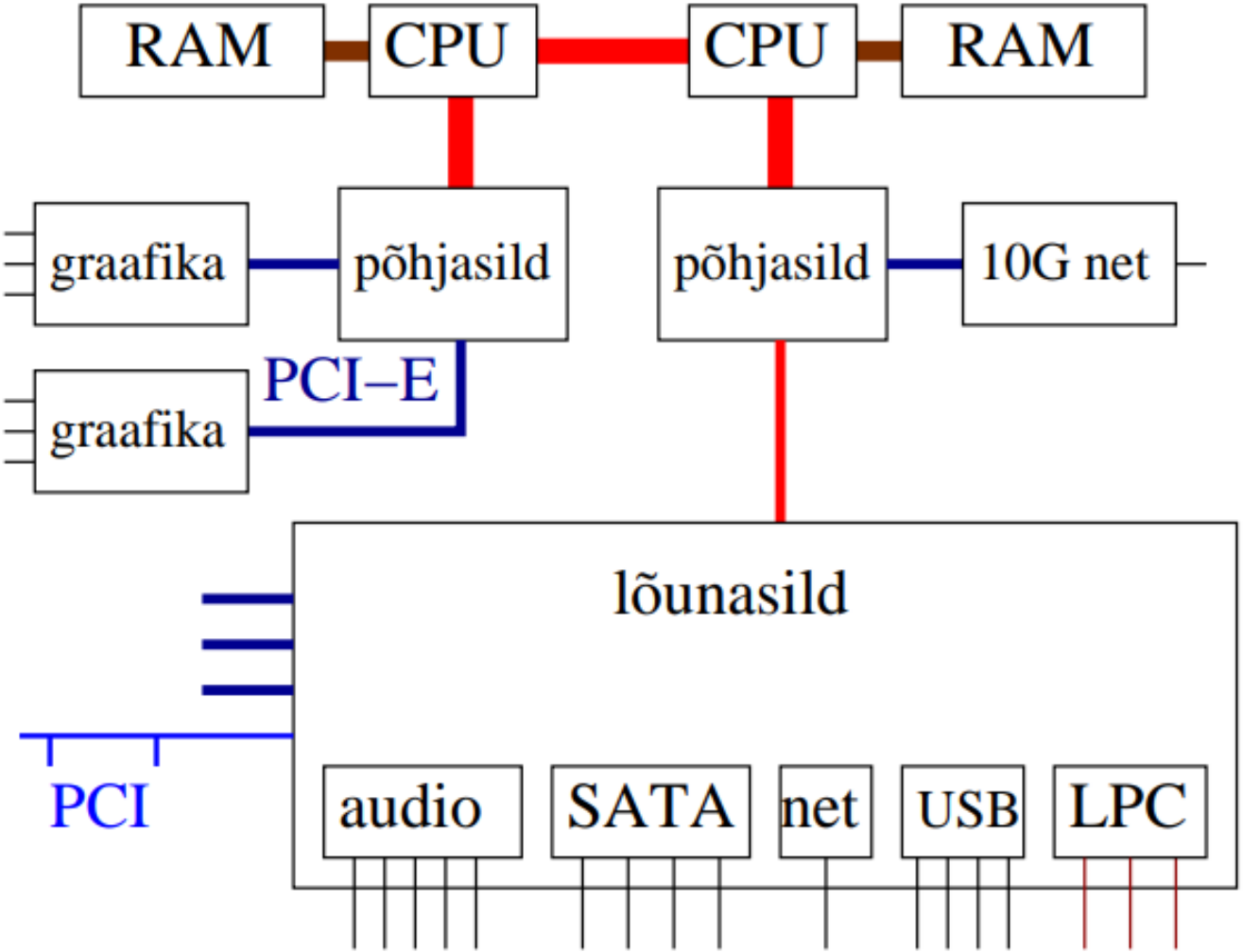
Ülevaade
vanast
PC-arvutist



Ülevaade PC-arvutist



Ülevaade kaasaegsest PC-arvutist



Süsteemsed siinid

- Vana PC: ISA, EISA, MCA, VLB
- PnP, ISAPnP, muu PnP
- Kaasaeg: PCI perekond
- PCMCIA / PC Card, ExpressCard, CompactFlash, . . .
- SoC (System on a Chip) sisemised siinid

PCI (Peripheral Component Interface)

- Algselt 32-bitine, 33 MHz
- 64-bitine 66 MHz PCI
- AGP — Advanced Graphics Port (PCI+IOMMU)
- Mitu pinget
- PCI-X (32 või 64-bitine; 33, 66, 100, 133, 266 ja 533 MHz)
- PCI Express (PCI-E) – Kuni 32 jadakanalit paralleelselt – Üks kanal (lane) 250 MB/s, 2.5 GT/s – PCI-E 2.0: 5GT/s, 3.0: 8 GT/s, 4.0: 16 GT/s
- CSA — Communication Streaming Architecture
- Hotplug, Thunderbolt, cPCI • SATA Express

Flash-mälu liidesed

- MTD (Memory Technology Devices) — flash-kivide madala taseme liidesed (NAND, NOR tavaliselt)
- (EEPROM,) Flash BIOS
- I2C, SPI stiilis liidesed
- CompactFlash
- MMC/xD
- SD/SDIO
- . . .
- Enamus liideseid on kasutatavad ka IO-seadmetele

Arvutivälised pordid

- Jadapordid — RS-232, RS-422 (0.15..460 kbps)
- Paralleelport — IEEE-1284 (kuni 8 Mbps)
- PS/2 — spetsiaalne jadaport
- Muud spetsiaalsed jadapordid (klaviatuurid, JTAG, . . .)

Arvutivälised siinid

- ADB — Apple Desktop Bus (10 kbps)
- USB — Universal Serial Bus
  - 1.1 ja 12 Mbps (UHCI, OHCI)
  - 480 Mbps (EHCI, USB 2.0)
  - 4.8 Gbps (XHCI, USB 3.0)
  - 10 Gbps (USB 3.1)
  - 20 Gbps (USB 3.2)
  - Wireless USB (WUSB) — 53..480 Mbps
- FireWire (IEEE-1394) – 400 Mbps, 800 Mbps, (1600 Mbps, 3200 Mbps)
- OHCI

Salvestusseadmete liidesed

- SCSI — Small Computer Systems Interface — tegelikult terve protokollivirn erinevate transpordikihtidega
- IDE (Integrated Drive Electronics) / ATA (AT Attachment)
- ATAPI — ATA Packet Interface
- FibreChannel — füüsiline kanal SCSI, IP jms jaoks; tegelikult omaette võrk
- Serial ATA – AHCI kui universaalne riistvaraliides SATA kontrolleritele
- SAS — Serial Attached SCSI
- Oma kontroller PCI-Express liidesega – SSD: AHCI, NVMe

SCSI

- Füüsiline kanal ja sideprotokoll on omavahel sõltumatud
- Füüsiliseks kanaliks oli esialgu paralleelkaabel
- FC, USB, FireWire, iSCSI, . . .
- Palju seadmeid samal siinil (tavaliselt kuni 8 või 16) – Adresseerimine: (bus, target, LUN)
- Märgistatud käskude järjekord (TCQ — Tagged Command Queuing)

IDE/ATA standardid

Nimetus	max kiirus	moodid
ATA (ATA-1)	8.3 MB/s	PIO 0-2, SWDMA 0-2, MWDMA 0
ATA-2	16.6 MB/s	PIO 0-4, SWDMA 0-2, MWDMA 0-2
ATA-3	16.6 MB/s	PIO 0-4, SWDMA 0-2, MWDMA 0-2
ATA-4	33 MB/s	+ UDMA 0,1,2
ATA-5	66 MB/s	+ UDMA 3,4
ATA-6	100 MB/s	+ UDMA 5
—	133 MB/s	mittestandardne
SATA-1	150 MB/s	eSATA, PMP
SATA-2	300 MB/s	NCQ
SATA-3	600 MB/s	
SATA-3.1	600 MB/s	mSATA, Zpodd, link power mgmt
SATA-3.2	1969 MB/s	SATA Express, M.2, microSSD