TEKNILLINEN KORKEAKOULU

# Kuljetuskerros

Kirja sivut: 280-301, 326-330

TEKNILLINEN KORKEAKOULU

# Kuljetuskerroksen tehtävä

- Kuljetuskerros yhdistää sovelluksia
  - Verkkokerros välittää viestejä koneelta toiselle
  - Kuljetuskerros lisää tarkemman osoitteen koneen sisällä
- Kuljetuskerros tarjoaa sovelluksille palveluita
  - Luotettava/epäluotettava tiedonsiirto
  - Viestinvälitys (datagrammi) tai tavuvirta (virtuaalinen yhteys)

- Kuljetuskerros toteutetaan eri protokollilla, jotka ovat vaihtoehtoisia
  - TCP tarjoaa luotettavan tavuvirran palveluna sovellukselle
  - UDP tarjoaa epäluotettavan viestinvälityksen palveluna

# UDP

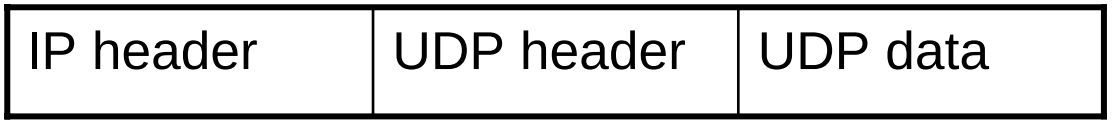
- User Datagram Protocol
- Standardi RFC-768
- Paketin syntaksi

| Source port | Destination port |
|---|---|
| Length | UDP checksum |
| Data | |

- Port on 16-bittinen numero, joka viittaa sovellukseen
- Tarkistussummaan lasketaan sekä otsake että data
  - Ei ole välttämätön

# ...UDP

- UDP-viesti kapseloidaan IP-viestiin:




- UDP tarjoaa epäluotettavan datagrammi-orientoituneen kuljetuskerrosprotokollan
  - Olennainen lisäarvopalvelu IP:n päälle on porttiosoitteet
  - Kevyt, ei tilaa, ei yhteydenmuodostusta, helppo toteuttaa, nopea (ei uudelleenlähetystä)
- UDP-sovelluksia: DNS, Radius, NTP, SNMP, RTP (VoIP)

TEKNILLINEN KORKEAKOULU

# UDP-kaappaus (DNS)

```
23 riku@mole $ dig a tapas.nixu.fi @194.197.118.20
;; got answer:
;; QUESTIONS:
;;      tapas.nixu.fi, type = A, class = IN
;; ANSWERS:
tapas.nixu.fi.  3600    A       194.197.118.24
;; AUTHORITY RECORDS:
nixu.fi.        3600    NS      ns2.tele.fi.
nixu.fi.        3600    NS      ns.nixu.fi.
nixu.fi.        3600    NS      ns.tele.fi.
;; ADDITIONAL RECORDS:
ns2.tele.fi.    35619   A       193.210.19.190
ns.nixu.fi.     3600    A       193.209.237.29
ns.tele.fi.     555991  A       193.210.19.19
ns.tele.fi.     555991  A       193.210.18.18
;; Total query time: 88 msec
;; FROM: mole.nixu.fi to SERVER: 194.197.118.20
;; MSG SIZE  sent: 31  rcvd: 175
24 riku@mole $
```

TEKNILLINEN KORKEAKOULU

# DNS-haku, Ethernet-kehys

```
ETHER:  Packet 1 arrived at 11:19:24.80
ETHER:  Packet size = 73 bytes
ETHER:  Destination = 8:0:20:74:f1:2c, Sun
ETHER:  Source      = 0:0:3b:80:e:93,
ETHER:  Ethertype = 0800 (IP)
```

TEKNILLINEN KORKEAKOULU

# DNS-haku, IP-otsake

```
IP:   Version = 4
IP:   Header length = 20 bytes
IP:   Type of service = 0x00
IP:         xxx. .... = 0 (precedence)
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:   Total length = 59 bytes
IP:   Identification = 35734
IP:   Flags = 0x4 (do not fragment)
IP:   Fragment offset = 0 bytes
IP:   Time to live = 255 seconds/hops
IP:   Protocol = 17 (UDP)
IP:   Header checksum = 7e65
IP:   Source address = 194.197.118.22
IP:   Destination address = 194.197.118.20
IP:   No options
```

TEKNILLINEN KORKEAKOULU

# DNS-haku, UDP-otsake

```
UDP:  Source port = 38325
UDP:  Destination port = 53 (DNS)
UDP:  Length = 39
UDP:  Checksum = E34A
```

# DNS-haku, otsakkeet ja data

```
0: 0800 2074 f12c 0000 3b80 0e93 0800 4500
.. t.,..;.....E.
16: 003b 8b96 4000 ff11 7e65 c2c5 7616 c2c5
.;..@...~e..v...
32: 7614 95b5 0035 0027 e34a 000a 0100 0001
v....5.'.J......
48: 0000 0000 0000 0574 6170 6173 046e 6978
.......tapas.nix
64: 7502 6669 0000 0100 0100
u.fi.....
```

- Tästä eteenpäin näytetään otsakkeista vain olennaiset kentät

# DNS-vastaus otsakkeet

```
ETHER:  Packet size = 217 bytes
ETHER:  Destination = 0:0:3b:80:e:93,
ETHER:  Source      = 8:0:20:74:f1:2c, Sun
ETHER:  Ethertype = 0800 (IP)
IP:   Total length = 203 bytes
IP:   Flags = 0x4 (do not fragmnet)
IP:   Protocol = 17 (UDP)
IP:   Header checksum = 8ed6
IP:   Source address = 194.197.118.20
IP:   Destination address = 194.197.118.22
UDP:  Source port = 53
UDP:  Destination port = 38325
UDP:  Length = 183
UDP:  Checksum = AD48
```

TEKNILLINEN KORKEAKOULU

# DNS-vastaus otsakkeet ja data

```
0: 0000 3b80 0e93 0800 2074 f12c 0800 4500
..;..... t.,..E.
16: 00cb 7a95 4000 ff11 8ed6 c2c5 7614 c2c5
..z.@.......v...
32: 7616 0035 95b5 00b7 ad48 000a 8580 0001
v..5.....H......
48: 0001 0003 0004 0574 6170 6173 046e 6978
.......tapas.nix
64: 7502 6669 0000 0100 01c0 0c00 0100 0100
u.fi............
--- some reply data deleted ---
208: 087b db00 04c1 d212 124f
.{.......
```

# TCP

- Transmission Control Protocol
- Standardi RFC-793
- Tarjoaa palveluna virtuaaliyhteyden, luotettavan tavuvirran
- Sovelluksen lähettämä tavuvirta jaetaan pienempiin lohkoihin, jotka välitetään IP-viesteinä
- Ominaisuuksia
  - Tarkistussummat, aikakatkaisu, vuonohjaus ruuhkatilanteissa
  - Datan järjestyksen seuraaminen, duplikaattien hylkääminen
- TCP on useimpien sovellusten käyttämä: SMTP, HTTP (WWW), NNTP (News),...

# TCP-otsake

| Source port number | | | Destination port number |
|---|---|---|---|
| Sequence number | | | |
| Acknowledgment number | | | |
| Hdrlen | Reserv. | Flags | Window size |
| TCP checksum | | | Urgent pointer |
| Options (if any) | | | |
| Data (if any) | | | |

- Portit edustavat lähettävää ja vastaanottavaa sovellusta
- Sekvenssinumero on datan (jos on) ensimmäisen tavun numero sekvenssissä
- Kuittausnumero kertoo seuraavaksi odotettavan tavun numeron

# …TCP-otsake

- Liput (bittejä)
  - URG viestin urgent pointer osoittaa dataan, joka on aiheellista lukea ohi jonossa olevan datan
  - ACK ilmoittaa acknowledgment-numeron edustavan vastaanotetun datan määrää
  - PSH kertoo että tämän viestin jälkeen ei ole tulossa välittömästi uutta dataa, vaan datan voi väittää eteenpäin
  - RST resetoi yhteydet
  - SYN merkitsee yhteyden synkronointia avauksen yhteydessä
  - FIN sulkee yhteyden

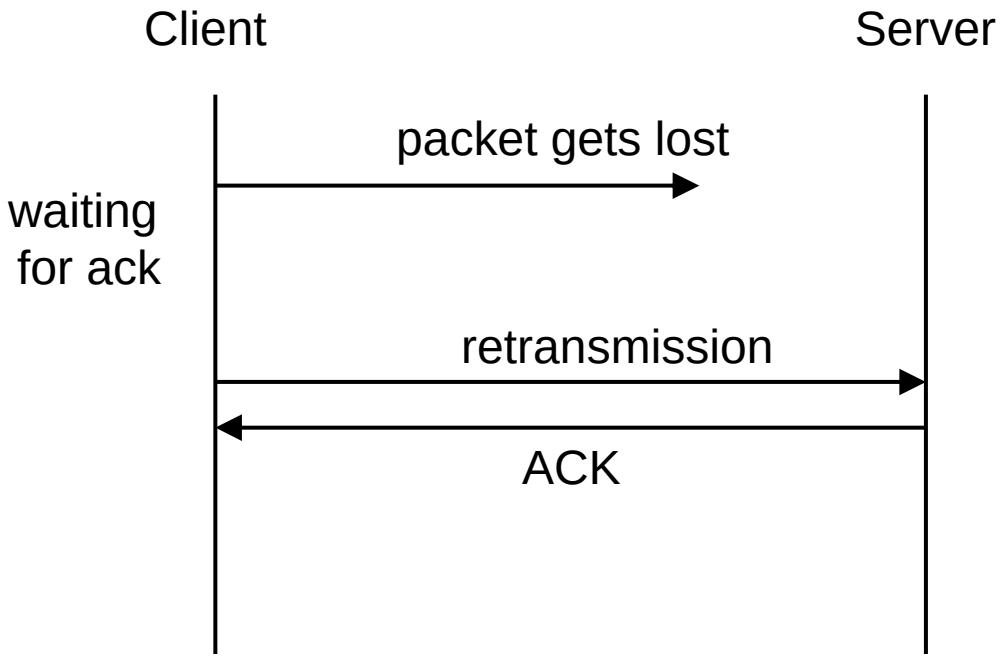
# …TCP-otsake

- Ikkunan koko kertoo montako tavua vastaanottaja on suostuvainen ottamaan vastaan, eli montako tavua lähettäjä voi lähettää odottamaan kuittausta
  - Tätä käytetään ruuhkanhallintaan
- Urgent pointer kiireellisen datan viimeiseen tavuun. Tätä käytetään esim. jos käyttäjä näppäilee interrupt-komennon kesken telnet-istunnon
- Optioita käytetään mm. selvittämään suurimman sallitun segmentin koko tai sopimaan lähetysikkunan käyttäytymisestä

TEKNILLINEN KORKEAKOULU

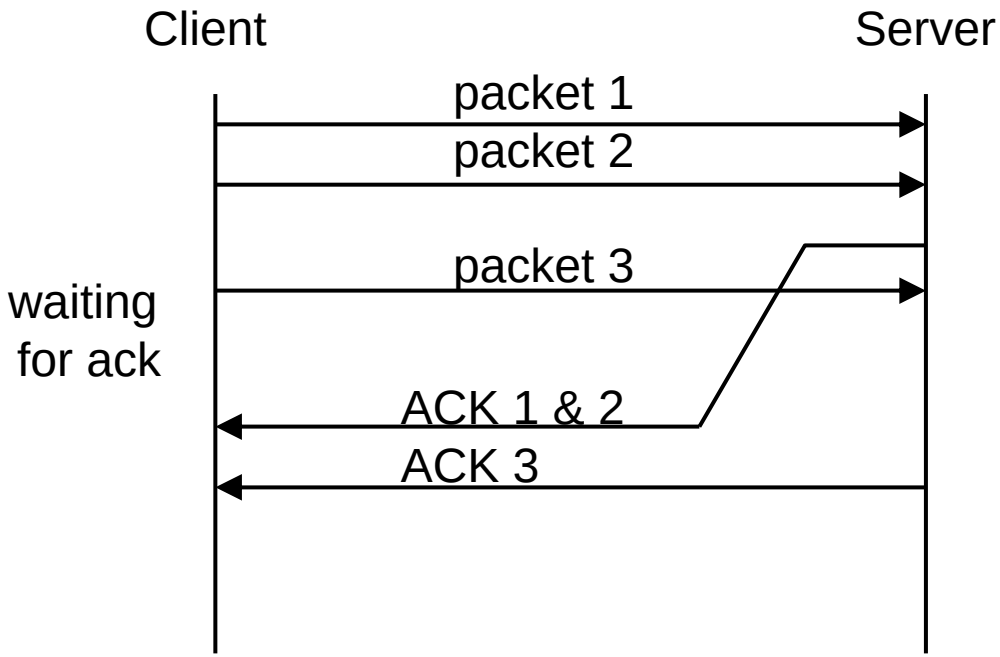
# TCP tietovuo

- Vastaanottaja lähettää kuittauspaketteja, joissa kerrotaan mihin tavuun saakka dataa on saatu
- Paketin kadotessa aikakatkaisu (timeout) saa sen lähtemään uudestaan



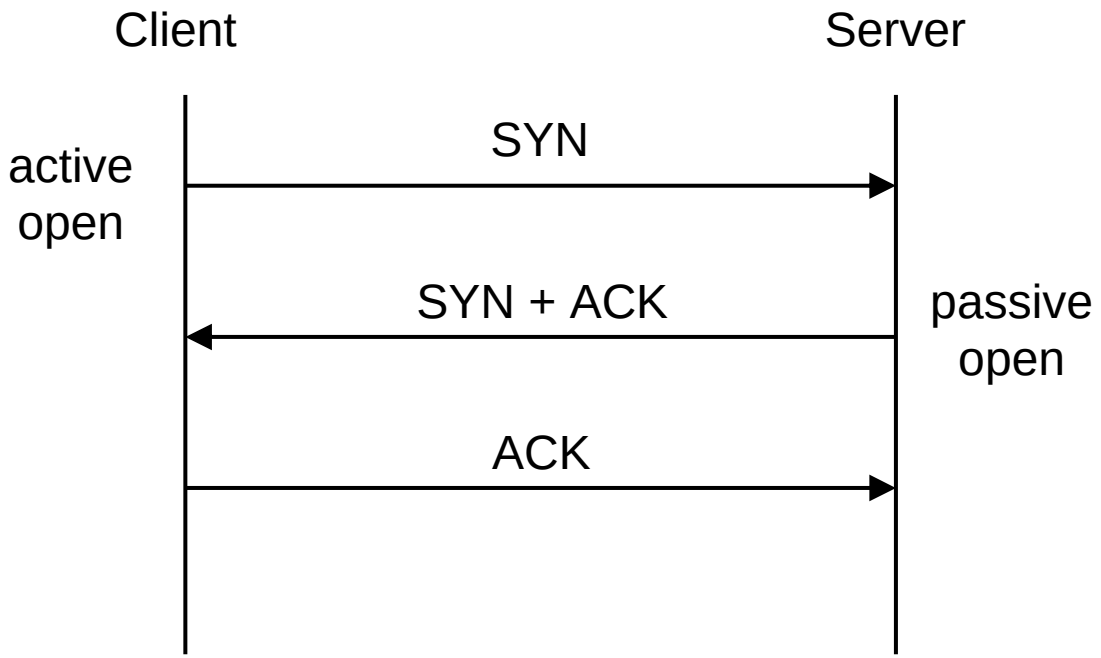
# …TCP tietovuo

- Liukuva ikkuna edistää siirtokapasiteetin hyödyntämistä
- Ikkunan koko riippuu asetuksista, ruuhkan hallinnasta, muistin määrästä jne.



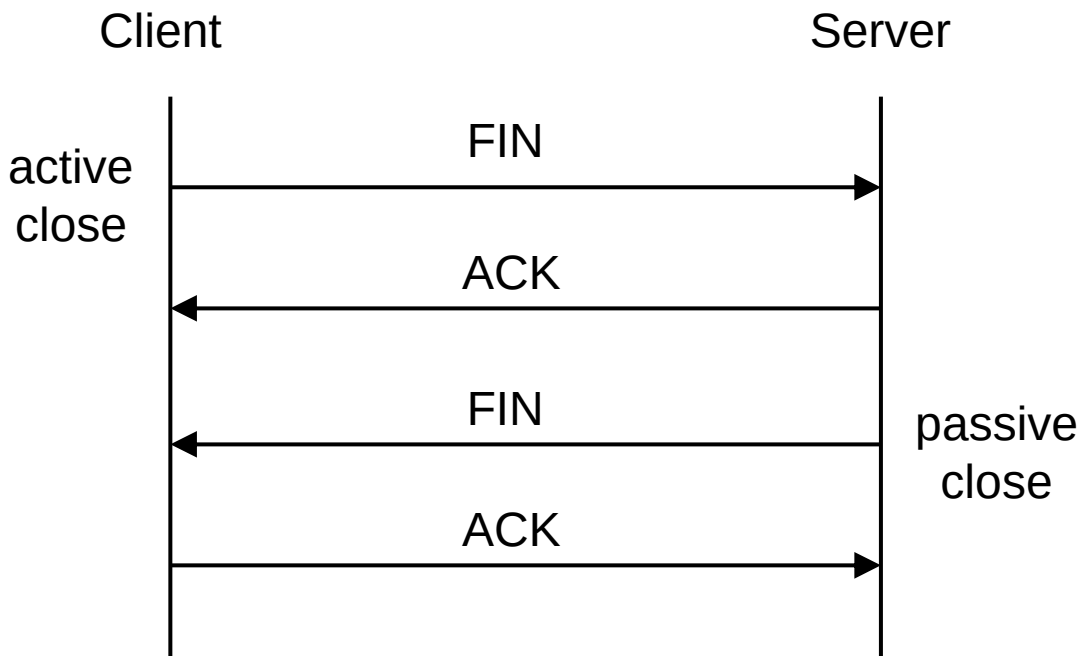
# TCP-yhteyden avaus

- Tunnetaan nimellä “three-way handshake”



TEKNILLINEN KORKEAKOULU

# TCP-yhteyden sulkeminen




- Kumpi tahansa osapuoli voi aloittaa sulkemisen
- Yleensä sovellustason protokollaistunto suljetaan ensin
- Huomaa, että TCP-yhteys on itse asiassa kaksi simplex-yhteyttä

TEKNILLINEN KORKEAKOULU

# TCP:n puutteet

- Lähetysikkunan koko
  - Standardin perusversiossa maksimikoko on 64 kt
  - Max bandwidth = max window size / round trip time
- Virheiden käsittely
  - TCP olettaa pakettien katoamisen johtuvan ruuhkasta verkossa
    - Internetin oletetaan koostuvan laadukkaista yhteyksistä
  - Ruuhkanhallinta (tarkemmin tietokoneverkot -kurssilla) toteutetaan pienentämällä lähetysikkunaa, joka vähentää liikennettä
  - Nykyään pakettien katoaminen saattaa johtua myös radioliikenteen häiriöistä
    - Jolloin oikea ratkaisu olisi lähettää data uudestaan

TEKNILLINEN KORKEAKOULU

# Kaapattu SMTP-istunto

```
24 riku@mole $ telnet jalopeno 25
Trying 194.197.118.20 ...
Connected to jalopeno.
Escape character is '^]'.
220-jalopeno.nixu.fi ESMTP Sendmail 8.6.12/8.6.12 ready at
   Sat, 5 Oct 1996
11:24:29 +0300
220 ESMTP spoken here
QUIT
221 jalopeno.nixu.fi closing connection
Connection closed by foreign host.
25 riku@mole $
```

TEKNILLINEN KORKEAKOULU

# 1. SYN asiakas->palvelin (Ethernet-otsake)

```
ETHER:  Packet 1 arrived at 10:58:0.62
ETHER:  Packet size = 60 bytes
ETHER:  Destination = 8:0:20:74:f1:2c, Sun
ETHER:  Source      = 0:0:3b:80:e:93,
ETHER:  Ethertype = 0800 (IP)
```

TEKNILLINEN KORKEAKOULU

# 1. SYN asiakas->palvelin (IP-otsake)

```
IP:   Version = 4
IP:   Header length = 20 bytes
IP:   Type of service = 0x00
IP:         xxx. .... = 0 (precedence)
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:   Total length = 44 bytes
IP:   Identification = 63629
IP:   Flags = 0x4 (do not fragment)
IP:   Fragment offset = 0 bytes
IP:   Time to live = 255 seconds/hops
IP:   Protocol = 6 (TCP)
IP:   Header checksum = 1188
IP:   Source address = 194.197.118.22, mole.nixu.fi
IP:   Destination address = 194.197.118.20, jalopeno.nixu.fi
IP:   No options
```

TEKNILLINEN KORKEAKOULU

# 1. SYN asiakas->palvelin (TCP-otsake)

```
TCP:  Source port = 35620
TCP:  Destination port = 25 (SMTP)
TCP:  Sequence number = 760886272
TCP:  Acknowledgement number = 0
TCP:  Data offset = 24 bytes
TCP:  Flags = 0x02
TCP:        ..0. .... = No urgent pointer
TCP:        ...0 .... = No acknowledgement
TCP:        .... 0... = No push
TCP:        .... .0.. = No reset
TCP:        .... ..1. = Syn
TCP:        .... ...0 = No Fin
TCP:  Window = 8760
TCP:  Checksum = 0x17a1
TCP:  Urgent pointer = 0
TCP:  Options: (4 bytes)
TCP:    - Maximum segment size = 1460 bytes
```

TEKNILLINEN KORKEAKOULU

# 1. SYN asiakas->palvelin (SMTP-ata)

```
SMTP: ""
```

- From now on only relevant portions of the headers will be displayed

TEKNILLINEN KORKEAKOULU

# 2. SYN+ACK palvelin->asiakas

```
ETHER:  Destination = 0:0:3b:80:e:93,
ETHER:  Source      = 8:0:20:74:f1:2c, Sun
ETHER:  Ethertype = 0800 (IP)
IP:   Flags = 0x4 (do not fragment)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.20, jalopeno.nixu.fi
IP:   Destination address = 194.197.118.22, mole.nixu.fi
TCP:  Source port = 25
TCP:  Destination port = 35620
TCP:  Sequence number = 2371738143
TCP:  Acknowledgement number = 760886273
TCP:  Flags = 0x12 (ACK, SYN)
TCP:  Options: (4 bytes)
TCP:    - Maximum segment size = 1460 bytes
SMTP:  ""
```

TEKNILLINEN KORKEAKOULU

# 3. ACK asiakas->palvelin

```
ETHER:  Destination = 8:0:20:74:f1:2c, Sun
ETHER:  Source      = 0:0:3b:80:e:93,
ETHER:  Ethertype = 0800 (IP)
IP:   Flags = 0x4 (do not fragment)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.22, mole.nixu.fi
IP:   Destination address = 194.197.118.20, jalopeno.nixu.fi
TCP:  Source port = 35620
TCP:  Destination port = 25 (SMTP)
TCP:  Sequence number = 760886273
TCP:  Acknowledgement number = 2371738144
TCP:  Flags = 0x10 (ACK)
TCP:  No options
SMTP:  ""
```

TEKNILLINEN KORKEAKOULU

# 4. Data palvelin->asiakas

```
ETHER:  Destination = 0:0:3b:80:e:93,
ETHER:  Source      = 8:0:20:74:f1:2c, Sun
ETHER:  Ethertype = 0800 (IP)
IP:   Flags = 0x4 (do not fragment)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.20, jalopeno.nixu.fi
IP:   Destination address = 194.197.118.22, mole.nixu.fi
TCP:  Source port = 25
TCP:  Destination port = 35620
TCP:  Sequence number = 2371738144
TCP:  Acknowledgement number = 760886273
TCP:  Flags = 0x18 (ACK, PSH)
SMTP:  "220-jalopeno.nixu.fi ESMTP Sendmail 8.6.12/8.6.12
   ready"
```

TEKNILLINEN KORKEAKOULU

# 5. Kuittaus asiakas->palvelin

```
ETHER:  Destination = 8:0:20:74:f1:2c, Sun
ETHER:  Source      = 0:0:3b:80:e:93,
ETHER:  Ethertype = 0800 (IP)
IP:   Flags = 0x4 (do not fragment)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.22, mole.nixu.fi
IP:   Destination address = 194.197.118.20, jalopeno.nixu.fi
IP:   No options
TCP:  Source port = 35620
TCP:  Destination port = 25 (SMTP)
TCP:  Sequence number = 760886273
TCP:  Acknowledgement number = 2371738258
TCP:  Flags = 0x10 (ACK)
SMTP:  ""
```

TEKNILLINEN KORKEAKOULU

# 6. Data asiakas->palvelin

```
ETHER:  Destination = 8:0:20:74:f1:2c, Sun
ETHER:  Source      = 0:0:3b:80:e:93,
ETHER:  Ethertype = 0800 (IP)
IP:   Flags = 0x4 (do not fragment)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.22, mole.nixu.fi
IP:   Destination address = 194.197.118.20, jalopeno.nixu.fi
TCP:  Source port = 35620
TCP:  Destination port = 25 (SMTP)
TCP:  Sequence number = 760886273
TCP:  Acknowledgement number = 2371738258
TCP:  Flags = 0x18 (ACK, PSH)
SMTP:  "QUIT\r\n"
```

TEKNILLINEN KORKEAKOULU

# 7. Kuittaus palvelin->asiakas (sisältää dataa)

```
ETHER:  Destination = 0:0:3b:80:e:93,
ETHER:  Source      = 8:0:20:74:f1:2c, Sun
ETHER:  Ethertype = 0800 (IP)
IP:   Flags = 0x4 (do not fragment)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.20, jalopeno.nixu.fi
IP:   Destination address = 194.197.118.22, mole.nixu.fi
TCP:  Source port = 25
TCP:  Destination port = 35620
TCP:  Sequence number = 2371738258
TCP:  Acknowledgement number = 760886279
TCP:  Flags = 0x18 (ACK, PSH)
SMTP:  "221 jalopeno.nixu.fi closing connection\r\n"
```

TEKNILLINEN KORKEAKOULU

# 8. palvelin alkaa sulkea

```
ETHER:  Destination = 0:0:3b:80:e:93,
ETHER:  Source      = 8:0:20:74:f1:2c, Sun
ETHER:  Ethertype = 0800 (IP)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.20, jalopeno.nixu.fi
IP:   Destination address = 194.197.118.22, mole.nixu.fi
TCP:  Source port = 25
TCP:  Destination port = 35620
TCP:  Sequence number = 2371738299
TCP:  Acknowledgement number = 760886279
TCP:  Data offset = 20 bytes
TCP:  Flags = 0x11 (ACK, FIN)
SMTP:  ""
```

TEKNILLINEN KORKEAKOULU

# 9. asiakas kuittaa aiemman data ja sulkemisen

```
ETHER:  Destination = 8:0:20:74:f1:2c, Sun
ETHER:  Source      = 0:0:3b:80:e:93,
ETHER:  Ethertype = 0800 (IP)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.22, mole.nixu.fi
IP:   Destination address = 194.197.118.20, jalopeno.nixu.fi
TCP:  Source port = 35620
TCP:  Destination port = 25 (SMTP)
TCP:  Sequence number = 760886279
TCP:  Acknowledgement number = 2371738300
TCP:  Flags = 0x10 (ACK)
SMTP:  ""
```

TEKNILLINEN KORKEAKOULU

# 10. asiakas alkaa sulkea myös

```
ETHER:  Destination = 8:0:20:74:f1:2c, Sun
ETHER:  Source      = 0:0:3b:80:e:93,
ETHER:  Ethertype = 0800 (IP)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.22, mole.nixu.fi
IP:   Destination address = 194.197.118.20, jalopeno.nixu.fi
TCP:  Source port = 35620
TCP:  Destination port = 25 (SMTP)
TCP:  Sequence number = 760886279
TCP:  Acknowledgement number = 2371738300
TCP:  Flags = 0x11 (ACK, FIN)
SMTP:  ""
```

TEKNILLINEN KORKEAKOULU

# 11.palvelin kuittaa sulkemisen

```
ETHER:  Destination = 0:0:3b:80:e:93,
ETHER:  Source      = 8:0:20:74:f1:2c, Sun
ETHER:  Ethertype = 0800 (IP)
IP:   Flags = 0x4 (do not fragment)
IP:   Protocol = 6 (TCP)
IP:   Source address = 194.197.118.20, jalopeno.nixu.fi
IP:   Destination address = 194.197.118.22, mole.nixu.fi
TCP:  Source port = 25
TCP:  Destination port = 35620
TCP:  Sequence number = 2371738300
TCP:  Acknowledgement number = 760886280
TCP:  Flags = 0x10 (ACK)
SMTP:  ""
```

TEKNILLINEN KORKEAKOULU

# Automatic Repeat Request

# ARQ

- Automatic repeat request on yleinen nimi joukolle tekniikkoja, joita mm. TCP hyödyntää
- ARQ-pohjaisia protokollia voidaan hyödyntää eri tasoilla ja niistä voidaan valita tarpeen mukaan sopiva
- Tämän esityksen tarkoitus on muistuttaa, että TCP:n käyttämä tapa ei ole ainoa vaihtoehto
  - Suunnittelija valitsee ratkaisun tilanteen mukaan

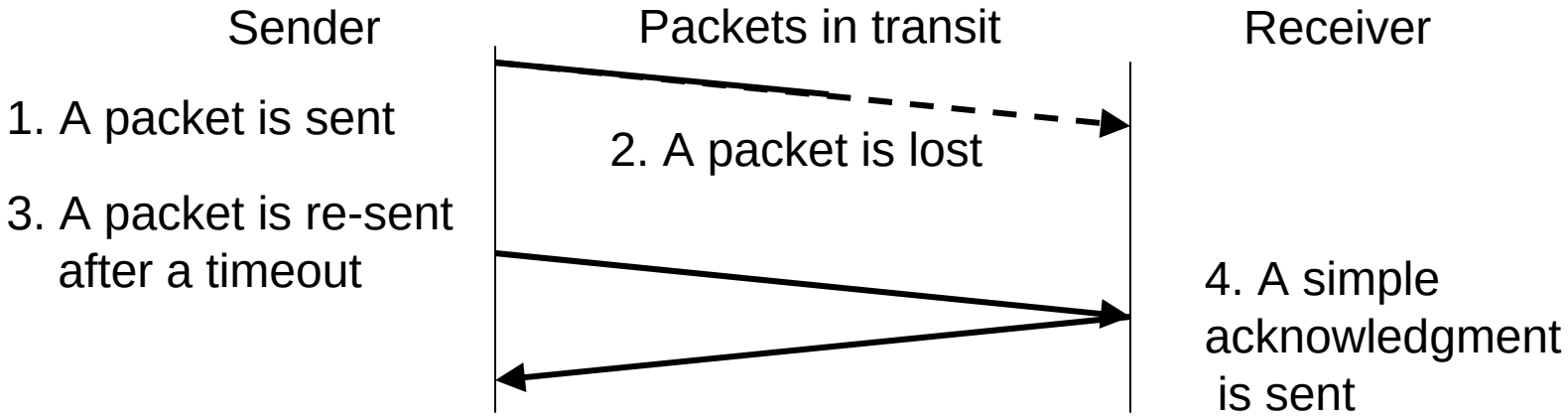
TEKNILLINEN KORKEAKOULU

# Tiedonsiirron luotettavuus uudelleenlähetyksillä

- Miten toteuttaa luotettava tiedonsiirto epäluotettavan kerroksen ylitse?
- Päästä päähän
  - Esim. TCP
- Vaihe vaiheelta
  - Esim. X.25, HDLC
- Eräs vastaus: ARQ
  - Automatic Repeat Request
  - ARQ on abstrakti konsepti tai periaate, ei protokolla itsessään
  - ARQ-tekniikkaa hyödynnetään useissa protokollissa
    - TCP, HDLC, Kermit, Xmodem...
  - ARQ-tekniikoita on kourallinen
  - Yhteistä on kadonnen datan korvaaminen lähettämällä se uudestaan

# Basic ARQ

- Data (SDU) kapseloidaan paketeiksi (PDU), joissa on otsake ja tarkistussumma
  - Kutsutaan myös informaatiokehyksiksi
- Merkinantoa (signalointi) varten on ohjauskehyksiä (control frame), joissa ei siirretä ylemmän tason informaatiota
- Lisäksi mekanismi aikakatkaisua (time-out) varten




- Mitä tapahtuu, jos kehys vastaanotetaan ja kuitataan sen jälkeen kun aikakatkaisu on tapahtunut lähettäjän päässä?

TEKNILLINEN KORKEAKOULU

# ARQ ja sekvenssinumerot

- Lähettäjä ja vastaanottaja menettävät helposti synkronointinsa
  - Ongelma, johon kaikkien protokollien on otettava kantaa
- Synkronointi voidaan varmistaa lisäämällä kuhunkin kehykseen sekvenssinumero
- Teoriassa yksi-bittinen sekvenssinumero riittää yksinkertaiselle Stop-and-Wait ARQ:lle
  - Stop-and-Wait tarkoittaa, että vain yksi kehys kerrallaan on matkalla
  - Yksi sekvenssibitti ei riitä, mikäli verkko saattaa kopioida kehyksiä
  - Stop-and-Wait ei ole yleensä kovinkaan tehokasta
- Suurempi sekvenssinumeroavaruus sallii useiden kehysten olevan matkalla samanaikaisesti

TEKNILLINEN KORKEAKOULU

# ARQ:n ohjauskehykset

- Nämä kolme perusviestiä mahdollistavat joukon erilaisia ARQ-toteutuksia
  - Kaikki toteutukset eivät käytä kaikkia viestejä
  - Esim. pelkkä ACK riittää TCP:lle
- ACK, acknowledgment, kuittaus, positiivinen vastaus
- NAK, negative acknowledgment, negatiivinen vastaus
- ENQ, enquiry, tiedustelu

TEKNILLINEN KORKEAKOULU

# Miten Stop-and-Wait ARQ käsittelee kadonneet kehykset

- Vain yksi informaatio-kehys lähetetään kerrallaan
- Sääntö: vain informaatiokehykset kuitataan, ohjauskehyksiä ei kuitata
- Kun kehys katoaa,
  - Lähettäjä lähettää uudelleen aikakatkaisun jälkeen (ei ACK-viestiä) tai
  - ENQ-viestiin vastataan viimeksi lähetetyllä kehyksellä
    - Lähettäjä ei saa kuittausta ja lähettää ENQ-viestin
    - Vastaanottaja lähettää viimeksi lähettämänsä ACK-viestin
    - Nyt lähettäjä on tietoinen vastaanottajan tilasta ja tiedonsiirto on uudelleensynkronoitunut

# Go-Back-N ARQ

- Riittävä numeroavaruus sekvenssinumeroille ja liukuva ikkuna
- Vastaanottaja kuittaa vain oikeassa järjestyksessä saapuvat kehykset
- Informaatiokehyksen kadotessa se ja kaikki sen jälkeen tulevat kehykset on lähetettävä uudelleen
  - Lähettäjä havaitsee kehyksen katoamisen aikakatkaisulla tai
  - Vastaanottaja lähettää NAK-viestin saadessaan kehyksen, joka ei ole paikallaan sekvenssissä
  - Vastaanottaja tarvitsee yhden kehyksen kokoisen puskurin
  - Lähettäjällä on oltava ikkunan suuruinen puskuri (kaikki lähetetyt kehykset, joista ei ole vielä tullut kuittausta)
- Jos ACK-ohjauskehys katoaa matkalla, myöhempi ACK riittää korvaamaan sen
- Go-Back-N ARQ:n toiminta on tehokkaampaa kuin Stop-and-Wait ARQ:n
  - Ikkunointi auttaa pitämään median täynnä dataa
  - Latenssi syö Stop-and-Wait ARQ:n tehoa huomattavasti
- TCP muistuttaa Go-Back-N ARQ:ta
  - TCP:ssä ei ole NAK-viestiä
  - TCP:n sekvenssinumerot kertovat siirretyistä tavuista, ei kehyksistä

# Selective Repeat ARQ

- Vastaanottaja voi myös tyytyä lähettämään NAK-viestin puuttuvista kehyksistä, jolloin vain ne tarvitsee lähettää uudelleen
  - Vastaanottajalla on oltava riittävän suuri puskurimuisti
- Go-Back-N ARQ:ta tehokkaampi hyvin virheisillä siirtoteillä

# Yhteenveto

- Kuljetuskerros tarvitaan yhdistämään sovelluksia
  - Verkkokerros välittää viestejä koneelta koneelle
  - Sovellus per kone ei ole toimiva abstraktio
- Kokonaisarkkitehtuurin olisi voinut tehdä toisinkin
  - TCP ja IP kuvastavat suunnittelijoiden maailmankuvaa, on tietokoneita ja tietokoneissa sovelluksia
  - Asian pohdinta ei ole tämän kurssin sisältöä, mutta MPLS:n ja HIP:n tapaiset tekniikat viittaavaat vaihtoehtoisten paradigmojen olemassaoloon