

Tietoliikenteen perusteet

Luento 5: Kuljetuskerros
luotettavan tiedonsiirron periaatteet

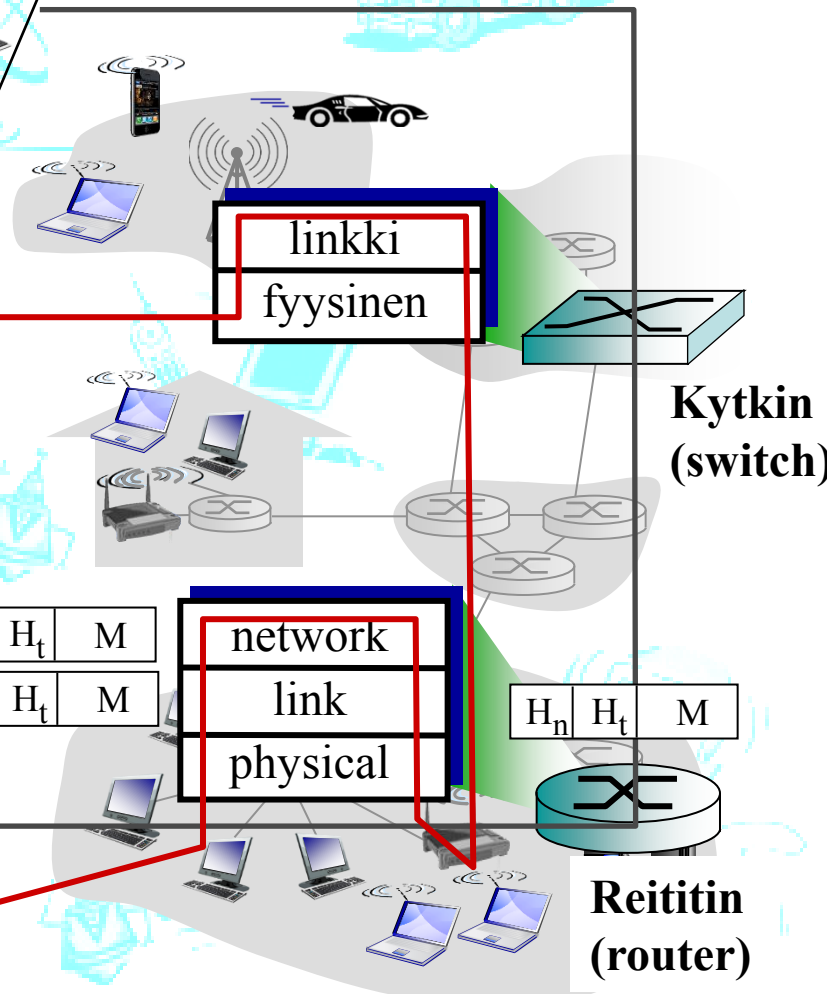
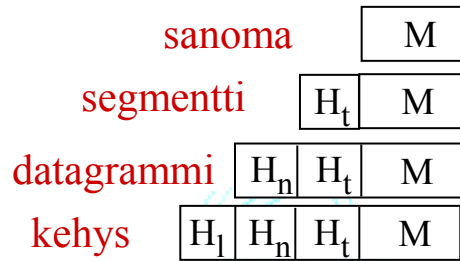
Syksy 2017, Timo Karvi

Kurose&Ross:
Ch3

Pääasiallisesti kuvien
© J.F Kurose and K.W. Ross, All
Rights Reserved

Lähettäjä (source)

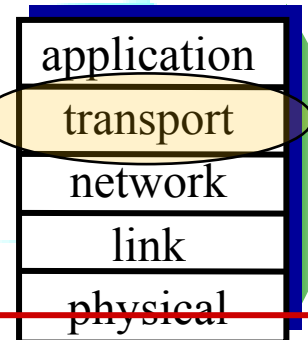
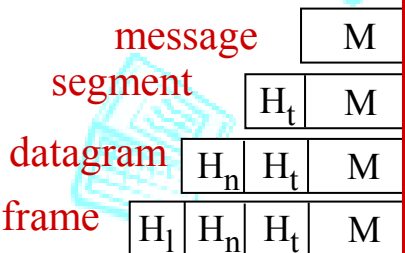
Luennon sisältöä



Kytkin (switch)

Reititin (router)

Vastaanottaja (destination)



Sisältöä

- Kuljetuspalvelut
- Luotettavan kuljetuspalvelun periaatteet
- Yhteydetön kuljetuspalvelu, UDP
- Yhteydellinen kuljetuspalvelu, TCP
- Ruuhkanhallinta TCP:ssä

Oppimistavoitteet:

- Tuntee Internetin kuljetusprotokollien (UDP/TCP) toiminnallisuuden ja periaatteet
- Osaa kuvata luotettavan kuljetuspalvelun ja vuonvalvonnan periaatteet ja toteutukset
- Osata selittää TCP-ruuhkanhallinnan perusidean



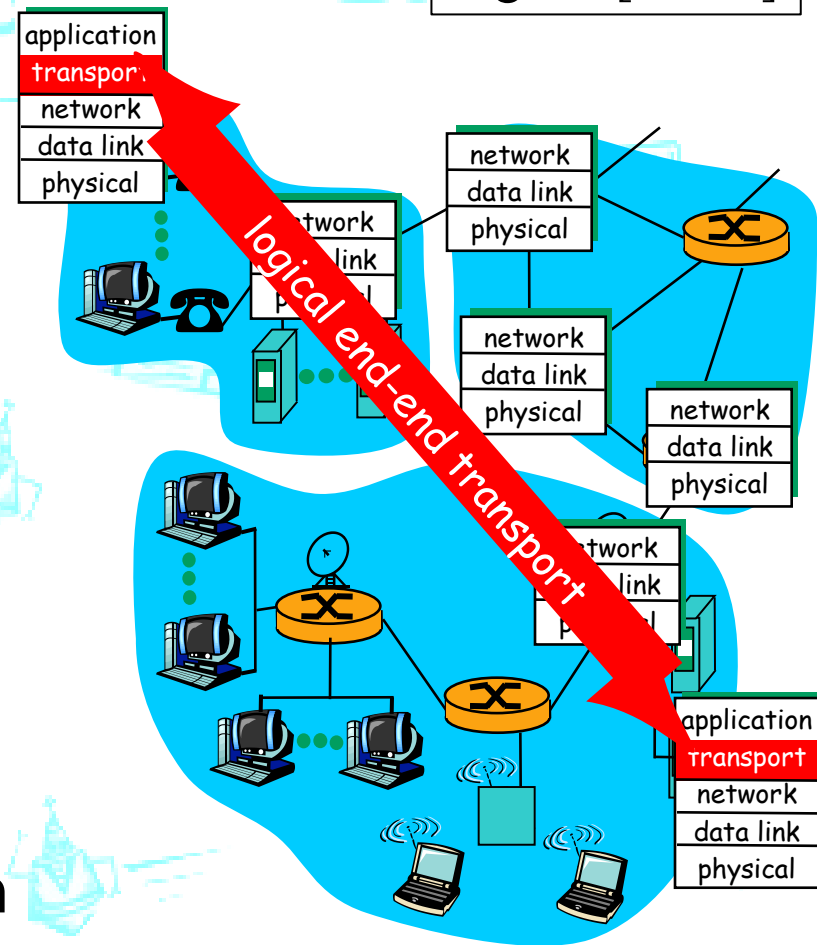


KULJETUSPALVELU

Kuljetuskerros

- Tarjoaa kuljetuspalvelun **prosessien välille**
- Vain isäntäkoneissa
 - Lähetys: Pilko sovelluskerroksen sanoma pienemmiksi **segmenteiksi**, jotka verkkokerros toimittaa perille.
 - Vastaanotto: Kokoa segmentit **sanomaksi**, jonka sovellus lukee.
- **Verkkokerros reitittää** koneesta koneelle
- Segmentin koko s.e. verkkokerros pystyisi välittämään sen sellaisenaan

Fig 3.1 [KR12]



Sovelluksen vaatimukset vs verkon ominaisuudet

- **Sovellus edellyttää kuljetuspalvelulta**

- Virheettömyyttä, luotettavuutta
- Järjestyksen säilymistä
- Kaksoiskappaleiden karsimista
- Mielivaltaisen pitkien sanomien sallimista
- Vuonvalvonnan mahdollistamista

- **Verkkokerroksen palvelu** voi kuitenkin

- Muuttaa segmentin bittejä tai kadottaa segmenttejä
 - Toimittaa segmentit epäjärjestyksessä kuljetuskerrokselle
 - Viivyyttää segmenttejä satunnaisen pitkän ajan
 - Luovuttaa kuljetuskerrokselle useita kopioita samasta segmentistä
 - Rajoittaa segmentin kokoa
- Kuljetuskerros peittää verkkokerroksen puutteita ja parantaa sovelluksen näkemää palvelun laatua

Sovelluksen vaatimukset

Kuljetuskerros pyrkii paikkaamaan verkkokerroksen puutteita

Verkkokerroksen palvelut

Internetin kuljetusprotokollat

- **TCP:** luotettava, järjestyksen säilyttävä tavujen kuljetuspalvelu
 - **Virheenvalvonta** (error control): Huomaa ja korjaa virheet, hylkää kaksoiskappaleet
 - **Vuonvalvonta** (flow control): Älä ylikuormita vastaanottajaa
 - **Ruuhkanhallinta** (congestion control): Älä ylikuormita verkkoa
 - **Yhteyden** muodostaminen ja purku
- **UDP:** Ei-luotettava, ei-järjestyksen säilyttävä sanomien kuljetuspalvelu
 - Välittää vain sanomia, ei pyri mitenkään parantamaan verkkokerroksen tarjoamaa palvelun laatua
 - Luotettavuus jää sovelluskerroksen hoidettavaksi
- Kumpikaan kuljetuspalvelu ei anna takuita viiveelle tai siirtonopeudelle (“best effort”)

Mikä kone / Mikä prosessi?

- Kuljetuskerros tarjoaa **päästä-päähän yhteyden**
 - Prosessilta prosessille (= pistokkeesta pistokkeeseen)
 - Prosessi lukee ja kirjoittaa sanomia halutessaan
- Datan lisäksi on välitettävä osoitetietoja
 - Vastaanottajan ja lähettäjän tiedot
 - Eri koneiden prosessit voivat käyttää samaa palvelua
 - Saman koneen prosessit voivat käyttää eri palveluita
- Kuljetuskerros: mikä prosessi = mikä portti
- Verkkokerros: mikä kone = mikä IP-osoite
- Porttinumero
 - 16-bittinen: 0 – 65535
 - Portit 0 – 1024 on varattu kukin tietyille palvelulle (well known ports)
 - Esim. www-palvelulle portti 80, SMTP-postipalvelulle portti 25

Segmenttien lomitus (multiplexing) ja erottelu (demultiplexing)

Lomitus lähettäjällä:

Segmenttejä useasta pistokkeesta, liitä kuhunkin kuljetusotsake (käytetään erottelussa myöhemmin)

Erottelu vastaanottajalla:

Välitä saapuva segmentti oikeaan pistokkeeseen, tieto löytyy otsakkeesta

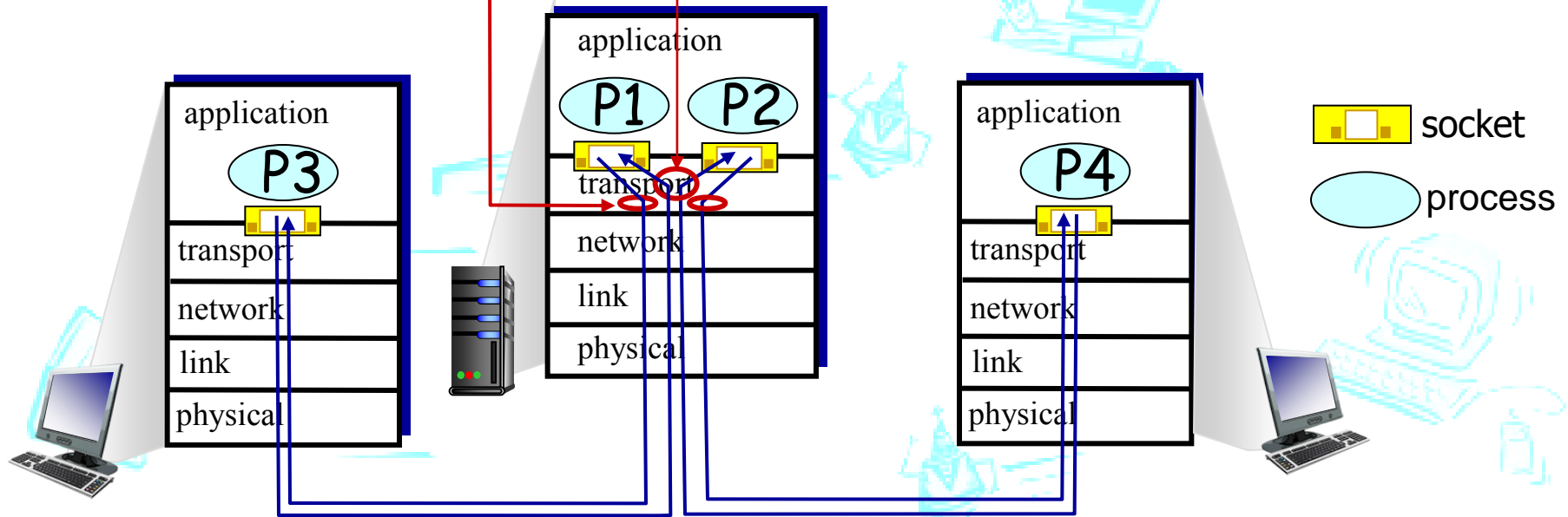


Fig 3.2 [KR12]

Mikä kone / Mikä prosessi?

Lähetys (asiakas)

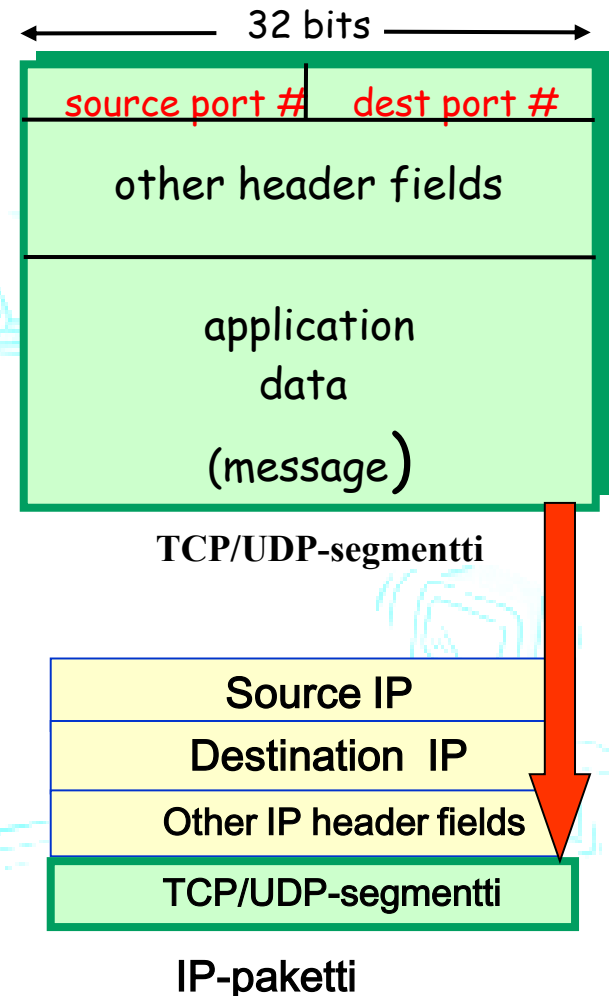
- Kuljetuskerros

- Segmentin otsakkeessa lähde- ja kohdeprosessin porttinumero
- Antaa segmentin verkkokerroksen välitettäväksi
- TCP: huolehtii myös luotettavuudesta
- UDP: tarjoaa pelkän välityspalvelun

- Verkkokerros

- Paketin otsakkeessa lähde- ja kohdekoneen IP-osoite,
- reitittimet osaavat ohjata oikealle koneelle

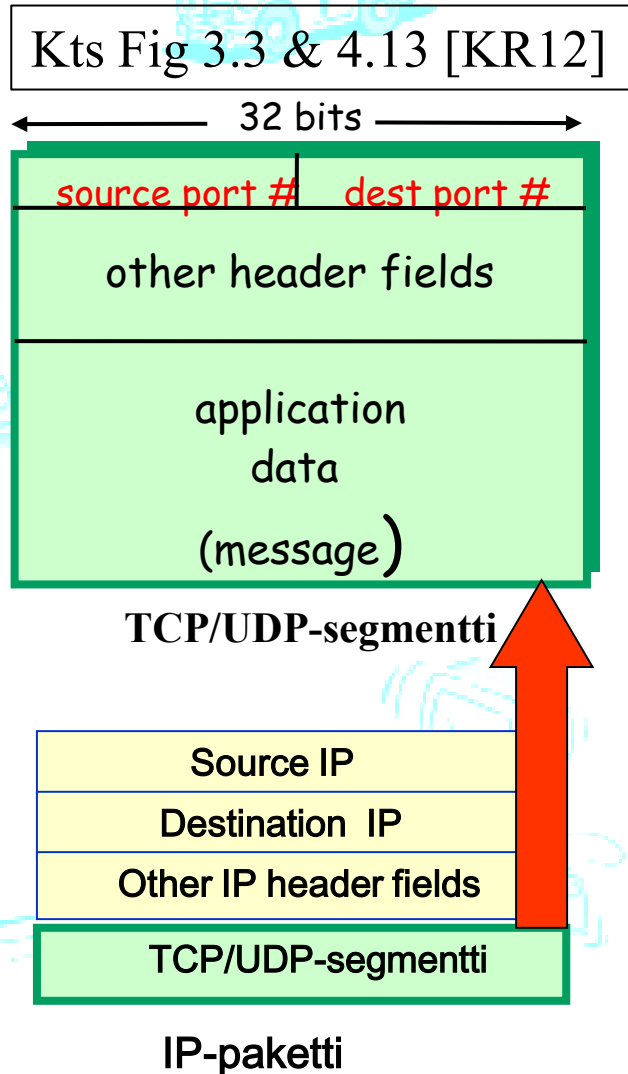
Kts Fig 3.3 & 4.13 [KR12]



Mikä kone / Mikä prosessi?

Vastaanotto (palvelija)

- Verkkokerros
 - Vastaanottaa IP-paketin
 - Poistaa verkkokerroksen otsaketiedot
 - Luovuttaa paketissa olleen segmentin kuljetuskerrokselle
- Kuljetuskerros
 - Poistaa kuljetuskerroksen otsaketiedot
 - Kokoo yhteenkuuluvat segmentit sanomiksi (tavuvirraksi)
 - Ohjaa sanoman (tavuvirran) oikealle prosessille (eli oikeaan pistokkeseen) porttinumeron avulla
 - TCP: huolehtii myös luotettavuudesta
 - UDP: tarjoaa pelkän välityspalvelun



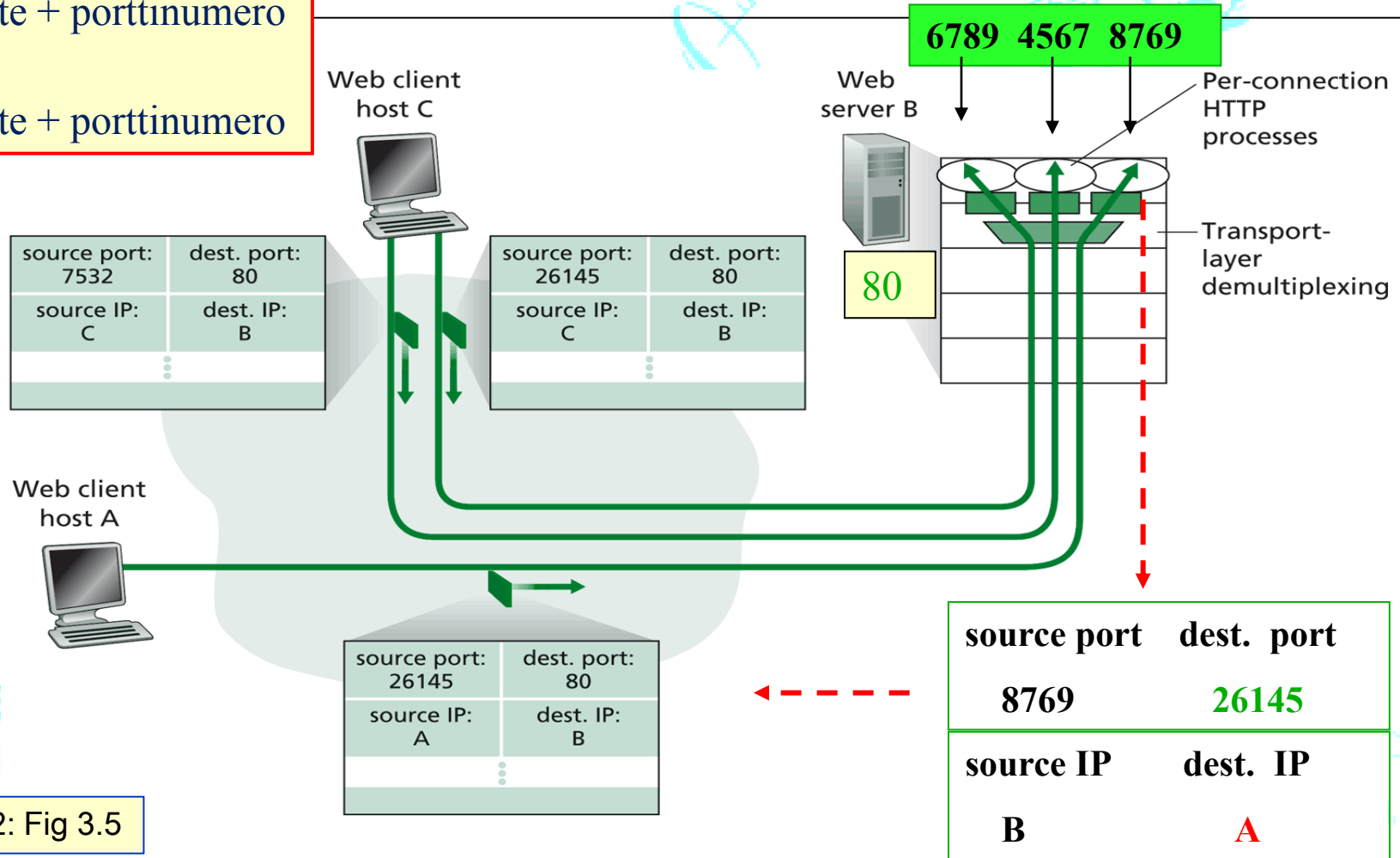
Kaksi www-asiakasta ja palvelija

TCP-yhteys:

koneosoite + porttinumero

UDP:

koneosoite + porttinumero



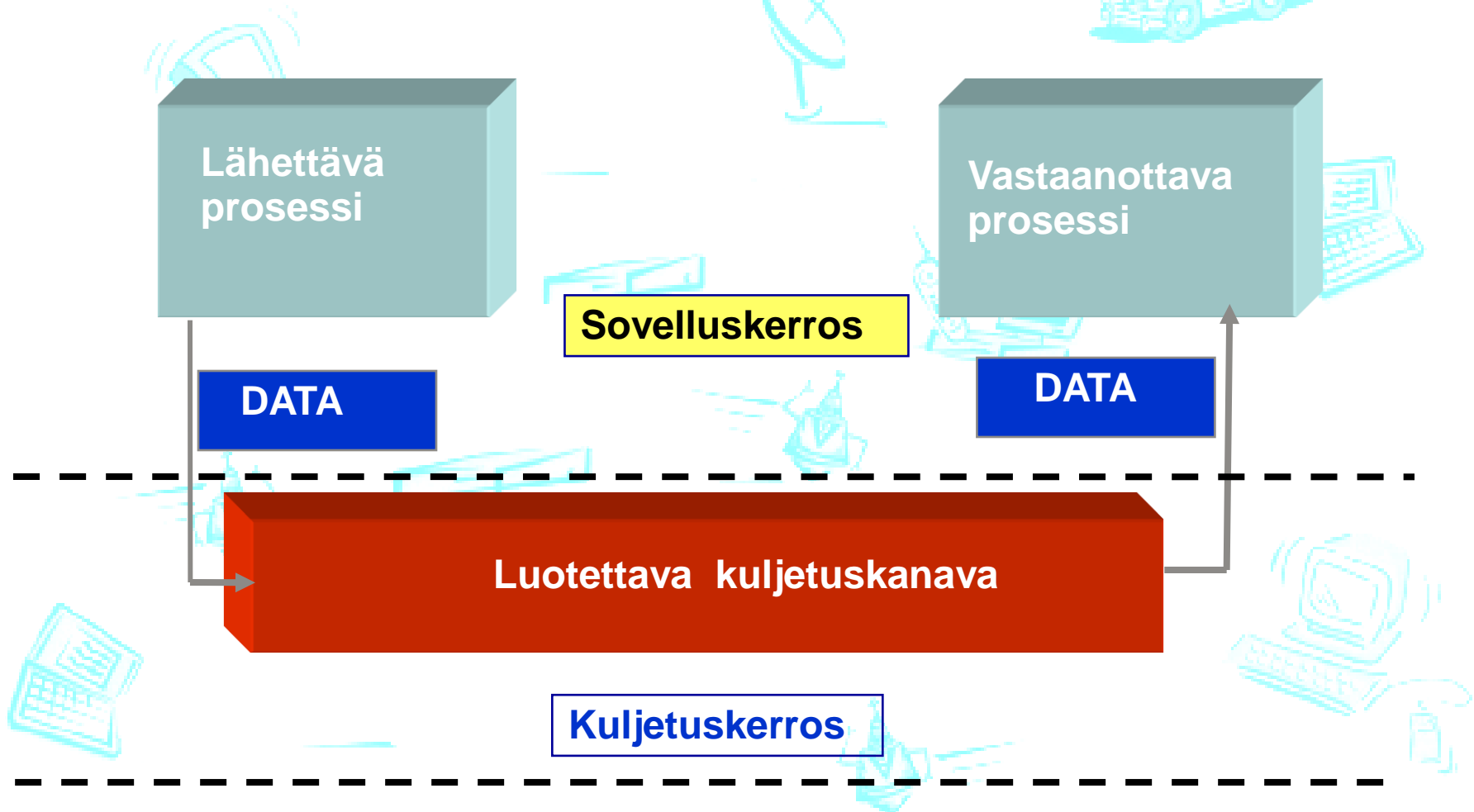
KR12: Fig 3.5

Figure 3.5 ♦ Two clients, using the same destination port number (80) to communicate with the same Web server application

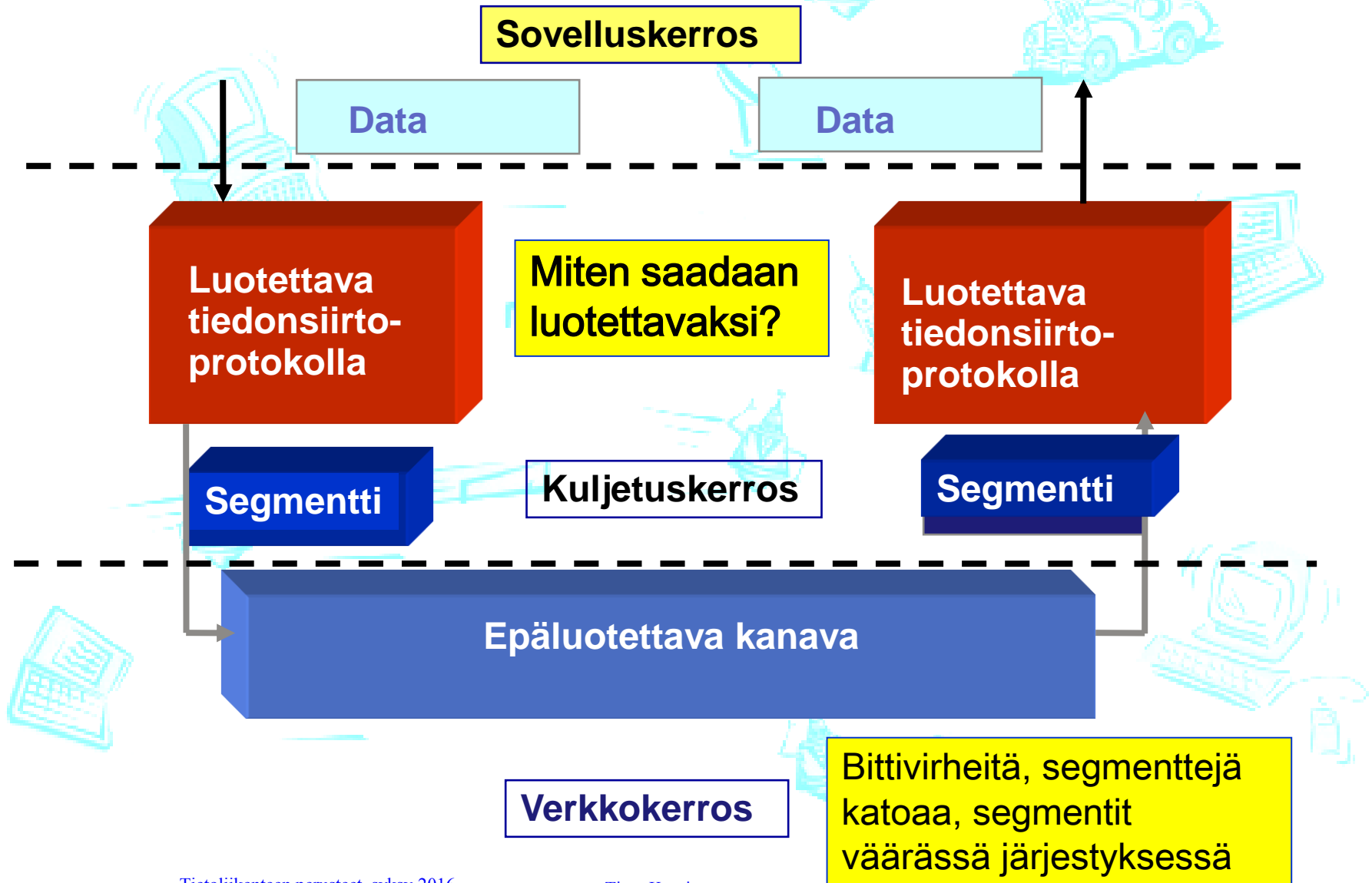


LUOTETTAVAN KULJETUS- PALVELUN PERIAATTEET

Luotettava tiedonsiirto?



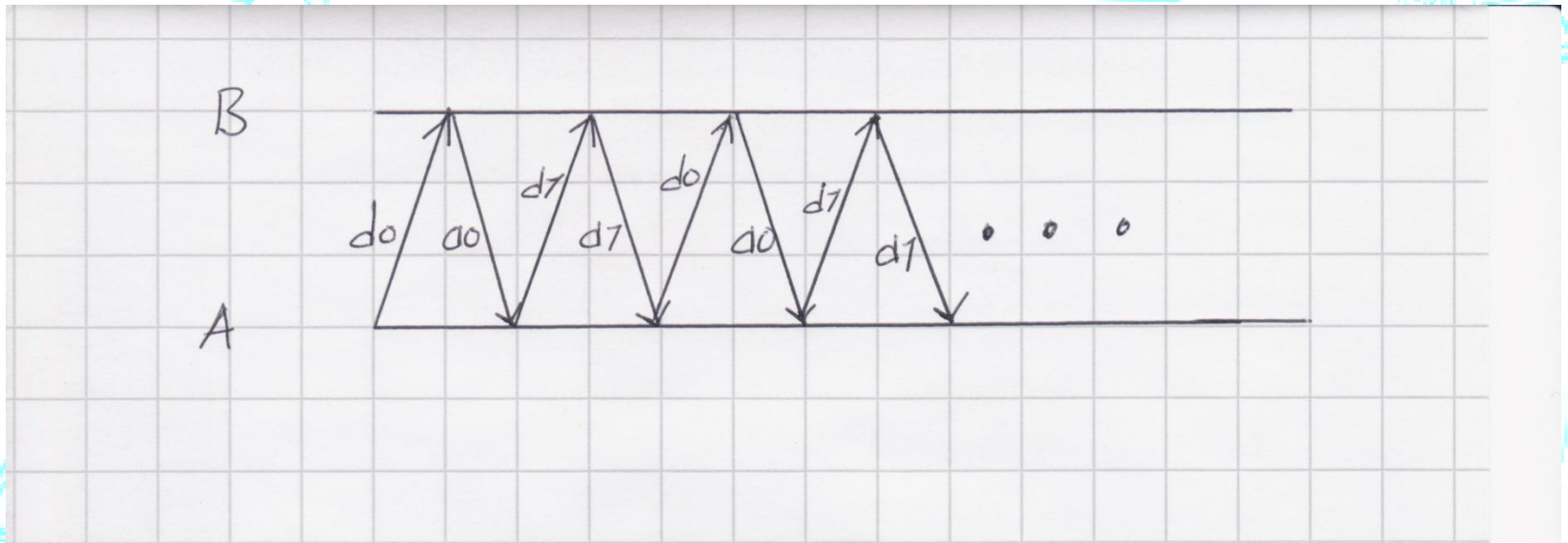
Luotettava tiedonsiirto?



Yksinkertainen siirtoprotokolla

- Aloitamme luotettavan tiedonsiirron tarkastelemalla yksinkertaista protokollaa, ns. vaihtuvan bitin protokollaa eli AB-protokollaa.
- Siinä jokaisen sanoman jälkeen odotetaan kuittausta, jonka jälkeen vasta voidaan lähettää.
- Sanomat ja kuittaukset on numeroitu 0:lla tai 1:llä.
- Seuraava kuvio näyttää, miten protokolla toimii virheettömän kanavan yhteydessä.
- d:t ovat datasanomia, a:t kuittauksia, A lähettäjä, B vastaanottaja.

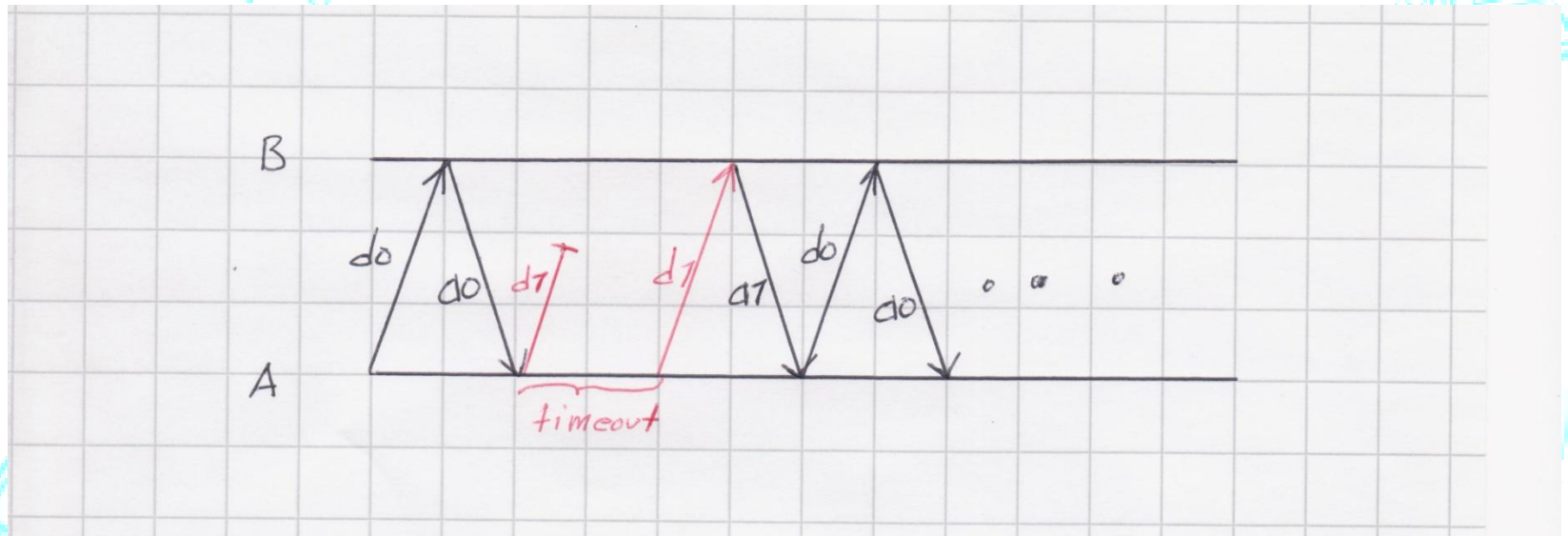
AB-protokolla, virheetön kanava



AB-protokolla, kadottava kanava

- Pakettien numerointi tulee käyttöön, jos kanava kadottaa tai vääristää sanomia.
- Lisäksi tarvitaan myös ajastin lähettäjän puolella.
- Seuraava kuvio näyttää, miten protokolla käsittelee tilanteen, kun datapaketti katoaa.
- Tässä tilanteessa ei tarvita vielä pakettien numerointia, vaan pelkkä ajastin riittää.

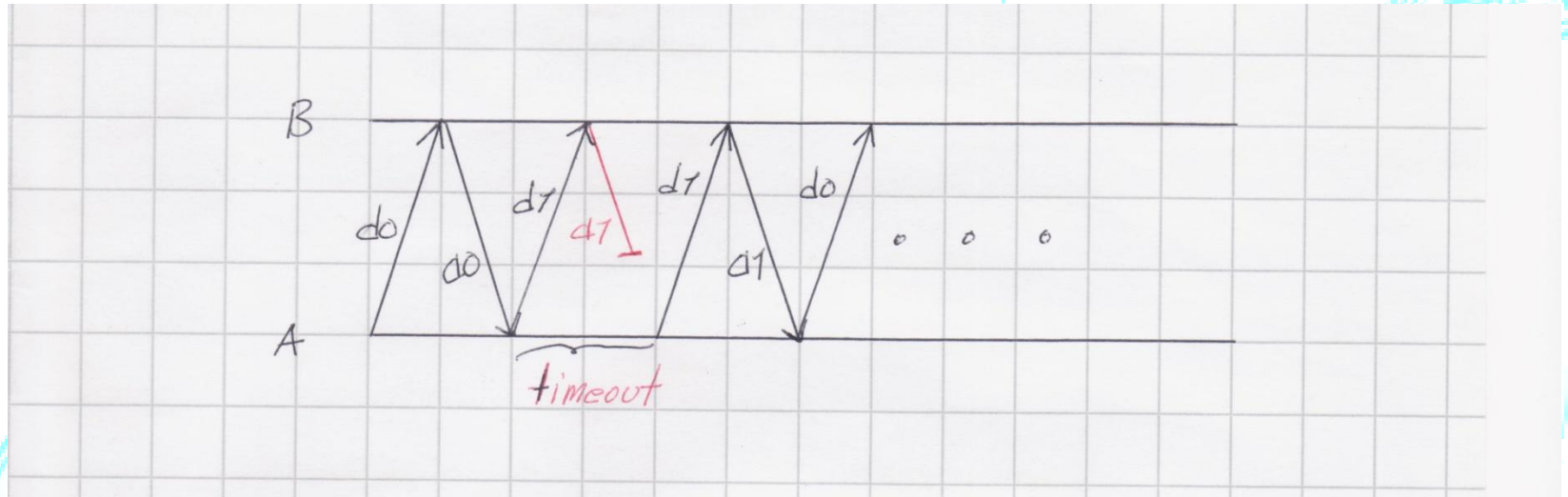
Sanoma katoaa



Kuittaus katoaa

- Seuraavassa skenaariossa kuittaus katoaa.
- Tällöin ajastin laukeaa lähettäjän puolella ja samoin kuin edellisessä skenaariossa lähettäjä lähettää edellisen datapaketin uudestaan.
- Nyt vastaanottaja huomaa, että sama datapaketti tulee toisen kerran, eikä lähetä sitä edelleen ylemmälle kerrokselle. Tässä siis numerointi tulee käyttöön.
- Sen sijaan vastaanottaja lähettää vain kuittauksen uudestaan.

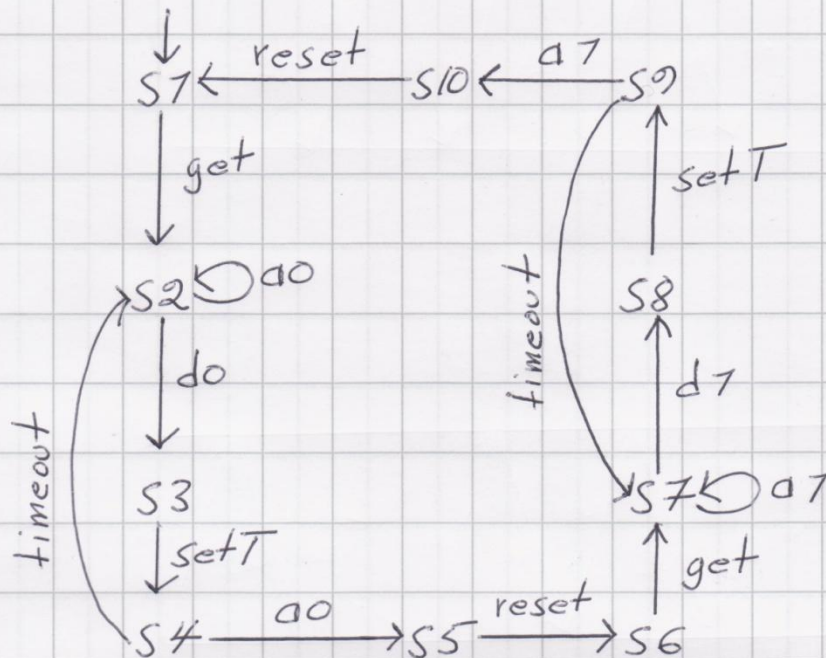
Kuittaus katoaa



Tila-automaattiesitys

- Tällaisia protokollia on helppo kuvata automaateilla.
- Automaatti koostuu tiloista ja tilojen välisistä siirtymistä.
- Protokollien tapauksessa siirtymiin liittyy lähetettävä sanoma.
- Tila määrää, mikä sanoma lähetetään tai otetaan vastaan.
- Seuraava kuvio näyttää AB-protokollan lähettäjän automaattina.

AB-protokollan lähettäjän tila- automaatti



Tila-automaatti, jatkoa

- Kuviossa get-sanoma tarkoittaa, seuraava datasanoma haetaan ylemmältä kerrokselta.
- Ajastinta ei ole määritelty erikseen, vaan se on kiinteästi osa lähettäjää. Huomattakoon, että tässä spesifioinnissa ajastin voi laueta turhan aikaisin, kun kuittaus on vielä matkalla.
- Tälläkin ajastimella protokolla toipuu sekä virheistä että turhan nopeasta ajastuksesta.

rdt3.0: Tehokkuus?

- Esim: 1 Gbps linkki, 15 ms päästä-päähän etenemisviive eli $RTT = 30$ ms, 1 KB:n paketti

$$T_{\text{transmit}} = \frac{L \text{ (packet length in bits)}}{R \text{ (transmission rate, bps)}} = \frac{8\text{kb/pkt}}{10^{**9} \text{ b/sec}} = 8 \text{ microsec}$$

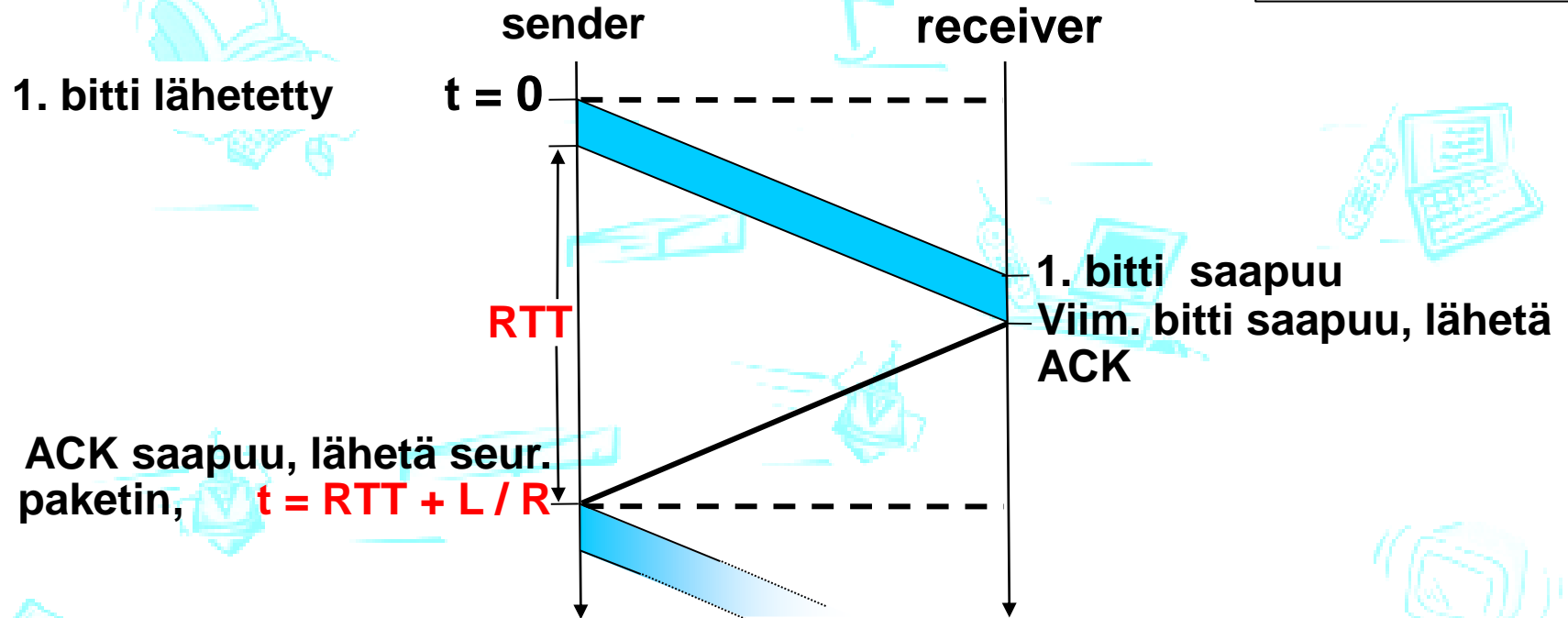
$$U_{\text{sender}} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

Käyttöaste (utilization): se osa kokonaisajasta, jolloin lähettäjä lähettää (RTT+L/R on kokonaisaika lähetyksessä kun odotetaan ACK)

- 1KB paketti 30 ms:n välein -> **33kB/s nopeus 1 Gbps linkillä.**
- Stop-and-wait-protokolla rajoittaa, ei linkin kyky siirtää dataa (linkin siirtonopeus)

rdt3.0: stop-and-wait tehokkuus

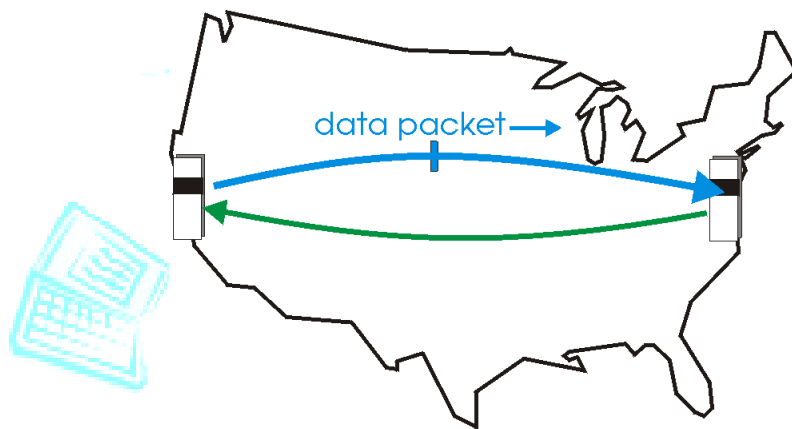
Fig 3.18a [KR12]



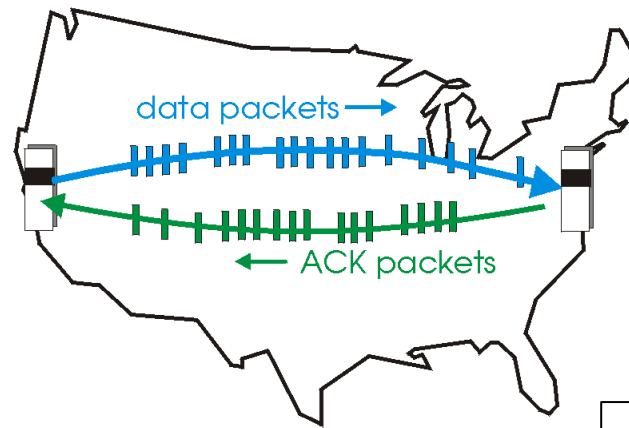
$$U_{\text{sender}} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

rdtX.X: Liukuhinnaprotokollat

- Lähettäjä saa lähettää useita paketteja, vaikka ei ole saanut kuittauksia edeltäviin
 - Numerointi (0,1) ei enää riitä, lisää numeroita tarvitaan
 - Tarvitaan puskurointia molemmissa päissä



(a) a stop-and-wait protocol in operation

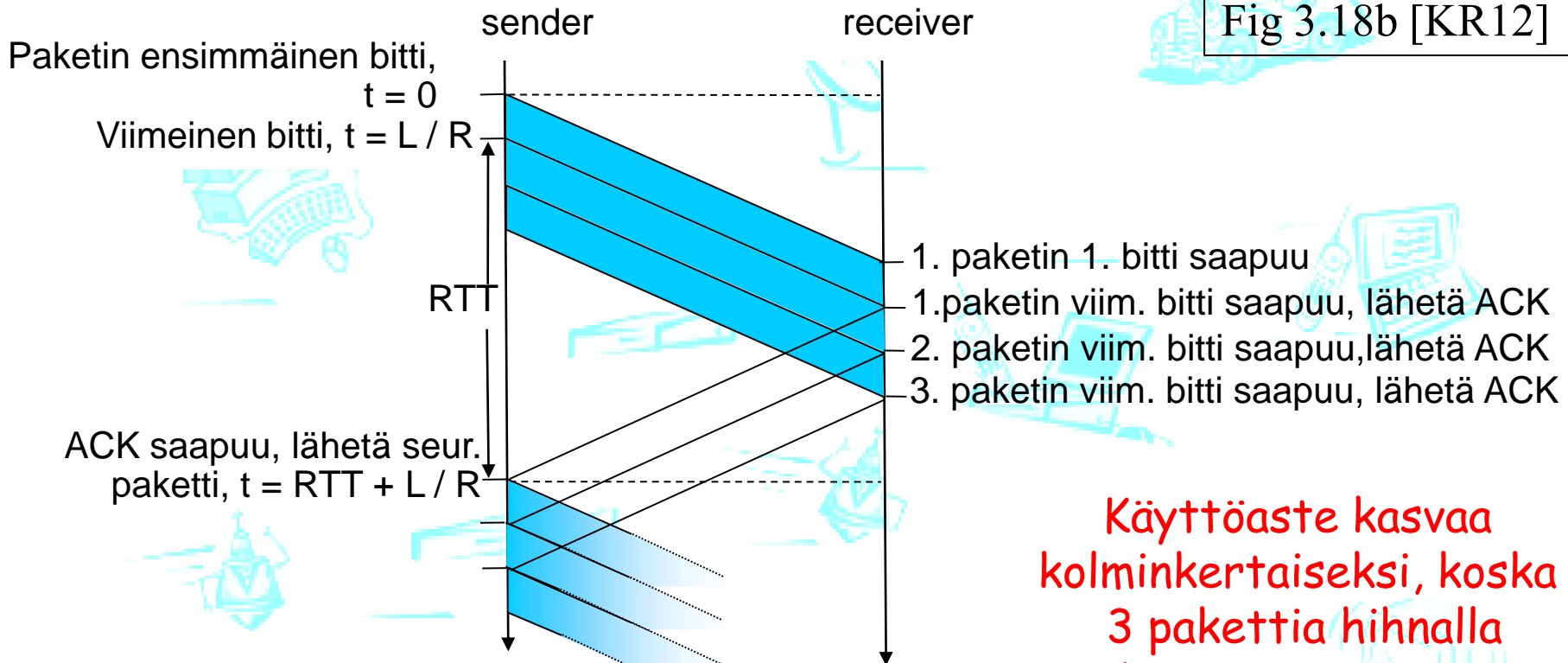


(b) a pipelined protocol in operation

Fig 3.17 [KR12]

Liukuhihnoitus: käyttöasteen kasvattaminen

Fig 3.18b [KR12]



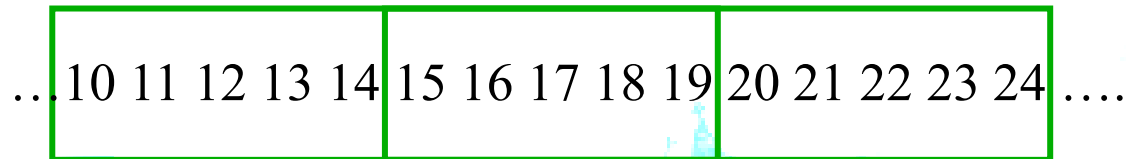
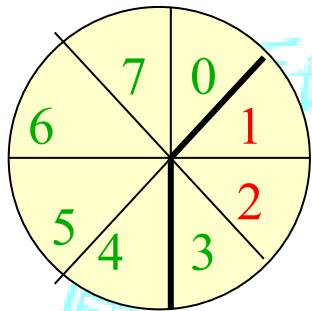
Käyttöaste kasvaa kolminkertaiseksi, koska 3 pakettia hihnalla

$$U_{\text{sender}} = \frac{3 * L / R}{RTT + L / R} = \frac{.024}{30.008} = 0.0008$$

ilman liukuhihnaa: 0.00027

Liukuva ikkuna (sliding window)

- Säätelee pakettien lähettämistä ja vastaanottoa, kertoo
 - millä pakettinumerolla on lähetetty/vastaanotettu,
 - mistä saatu/lähetetty kuittaukset ja
 - millä numeroilla voi vielä lähettää/vastaanottaa paketteja
- Ikkunan koko riippuu yhteyden tyypistä ja puskurien koosta



Liukuva ikkuna

- **Lähetysikkuna** (sender window)
 - Ikkunan koko = montako pakettia saa olla kuittaamatta
 - Mitkä pakettinumerot on käytetty, mutta kuittaamatta
 - Mitä pakettinumeroita voi vielä käyttää
- Lähettäjän on odotettava, jos kaikki ikkunan numerot on käytetty
 - Kun kuittaus saapuu, ikkuna liukuu
 - Seuraavat numerot tulevat luvallisiksi
- **Vastaanottoikkuna** (receiver window)
 - Mitkä pakettinumerot otettu vastaan, mutta kuittaamatta
 - Mitä pakettinumeroita lähettäjä saa vielä käyttää eli mitkä hyväksytään
- Jos saadussa paketissa on ikkunan viimeinen numero
 - Ikkuna pysäyttää pakettien lähetyksen vastapäätä
 - Ikkuna estää uusien pakettien vastaanoton
- Paketin kuittaus liu'uttaa myös vastaanottajan ikkunaa
 - Hyväksytään uusia pakettinumeroita

Kun ikkunan koko on 1

- Vain yksi paketti kuittaamattomana
 - One Bit Sliding Window –protokolla
 - = AB –protokolla
- Pakettinumerot 0 ja 1 riittävät
- ACK ilmoittaa
 - Joko seuraavaksi odotetun paketin numeron (esim.TCP)
 - tai viimeksi vastaanotetun virheettömän paketin numeron
- ACK sisältää paketin numeron
 - Kuittausduplikaatti ei voi kuitata väärää paketteja

Virhetilanteen käsittely

- Entä, kun huomataan virhe?
 - Monta muuta pakettia jo matkalla!
- Pakettiin ei tule kuittausta
 - Paketti katosi tai virheellinen
 - Kuittaus katosi tai virheellinen
 - => Ajastin laukeaa aikanaan
- "Go-Back-N" (paluu N:ään)
 - Paketit uudelleenlähetetään virheellisestä lähtien
- Selective Repeat (Valikoiva toisto)
 - Lähetetään vain virheelliset paketit

Go-Back-N

- Vastaanottaja hyväksyy paketit vain järjestyksessä
 - Kuittaa järjestyksessä tulleen virheettömän paketin
 - Hylkää kaikki puuttuvan tai virheellisen paketin jälkeiset paketit eikä lähetä niistä kuittauksia
- Kun lähettäjä ei saa pakettiin kuittausta
 - Lähetysikkuna täyttyy ja estää uusien pakettien lähettämisen
 - Lähettäjän ajastimet laukeavat
 - Lähettäjä lähettää uudestaan kaikki viimeisen kuittauksen jälkeiset paketit
 - Näiden kuittaukset siirtävät taas lähetysikkunaa
- **Tehoton, jos paljon virheitä ja iso ikkuna**

Go-Back-N

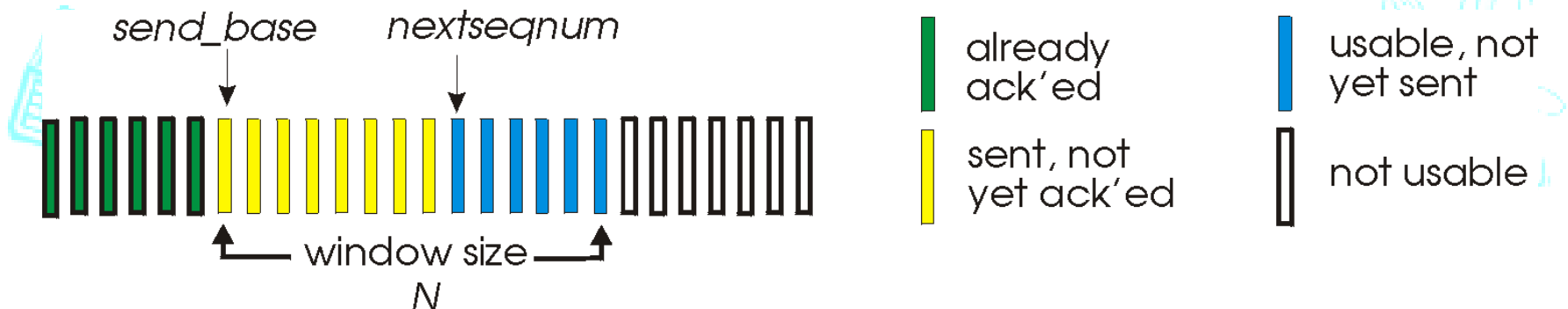
Fig 3.19 [KR12]

- **Kumulatiivinen ACK**

- Lähetä ACK, jossa korkein järjestyksessä saadun kelvollisen paketin numero
- Tämä kuittaa kaikki pienemmällä numerolla lähetetyt paketit

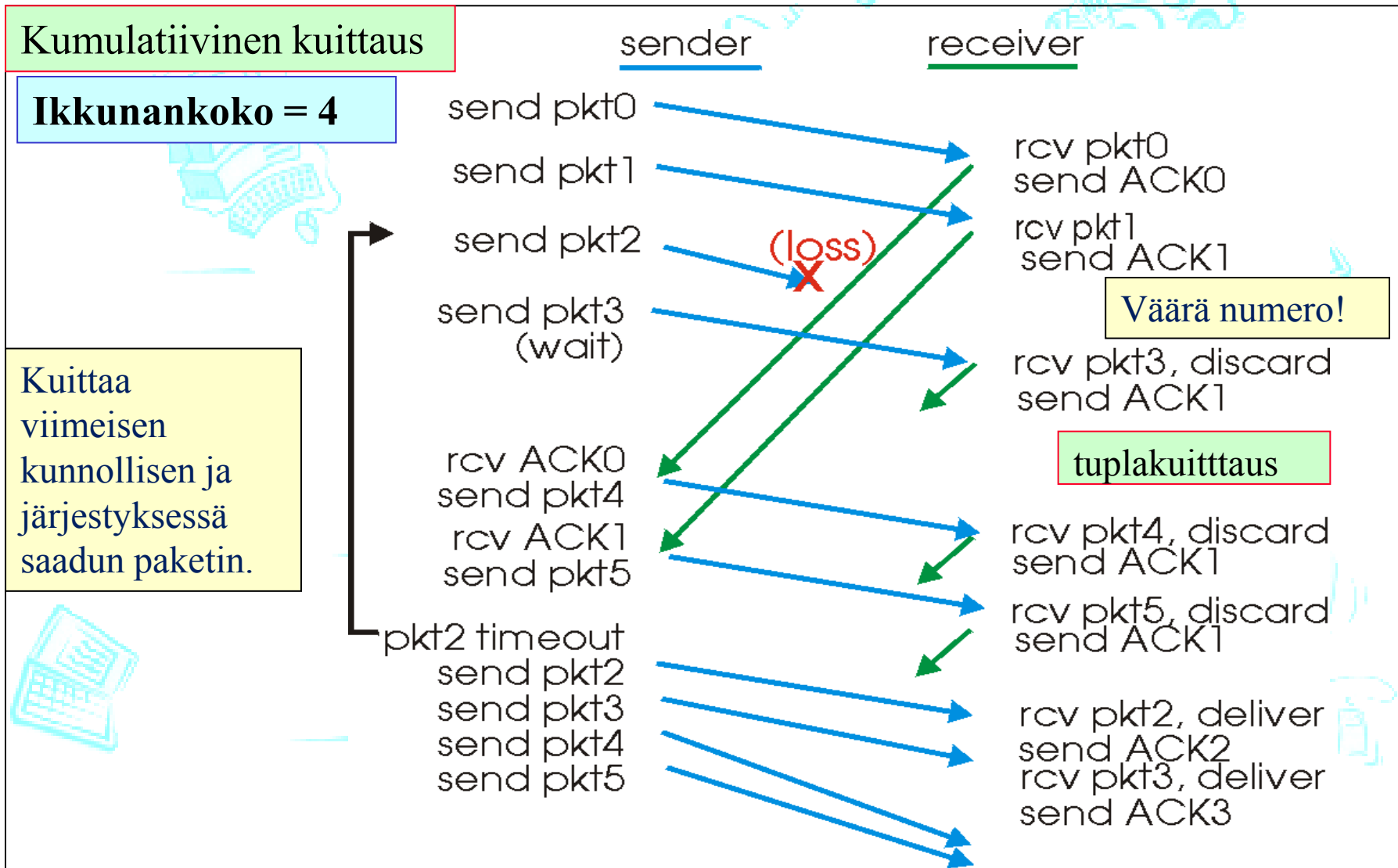
- Jos välistä puuttuu paketti

- Lähetä uudestaan ACK, jossa korkein järjestyksessä saadun paketin numero (~ NAK)
- **Tuplakuittaus** (duplicate ACK)
- Parannus: => nopeampi reagointi puuttuvaan



Go-Back-N: Esimerkki

Fig 3.22 [KR12]

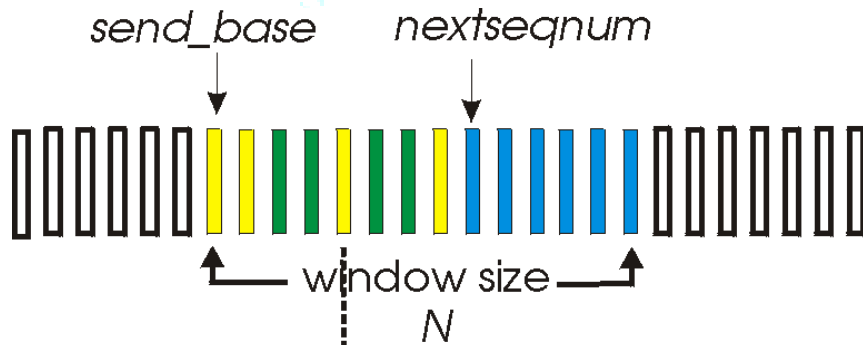


Valikoiva toisto (Selective Repeat)

- Valikoiva uudelleenlähetyks
 - Lähetä uudelleen vain virheellinen /puuttuva paketti
- **Kuittaus jokaiselle kelvolliselle paketille**
- **Paketit sovellukselle oikeassa järjestyksessä**
 - Vastaanottajalla oltava puskuritilaa pakettien järjestämiseen
- Jos lähettäjä ei saa kuittausta paketista
 - Lähetyksikkunan täytyminen pysäyttää lähettämisen
 - Aikanaan ajastin laukeaa ja aiheuttaa uudelleenlähetyksen
 - Jokaisella paketilla on oma ajastin
- Ikkuna liikuu nytkin tasaisesti
 - Yksi puuttuva kuittaus voi pysäyttää lähetyksen
 - **Kun puuttuva paketti saatu, ikkuna liikuu kaikkien kuitattujen yli**

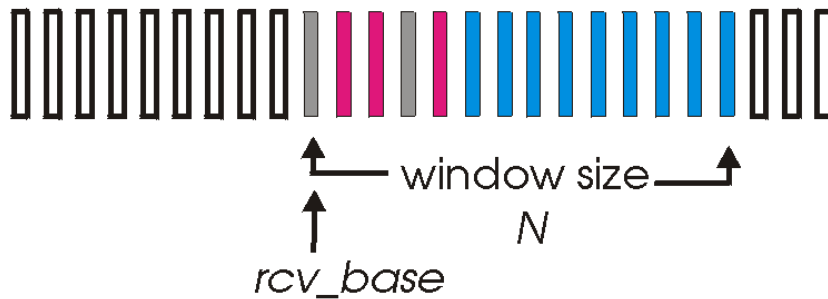
Valikoiva toisto

Fig 3.23 [KR12]



- already ack'ed
- sent, not yet ack'ed
- usable, not yet sent
- not usable

(a) sender view of sequence numbers



- out of order (buffered) but already ack'ed
- Expected, not yet received
- acceptable (within window)
- not usable

(b) receiver view of sequence numbers

Esimerkki: Valikoiva toisto

Jokainen sanoma
kuitattava erikseen
Paketin 2 katoaminen
estää ikkunaa liikkumasta

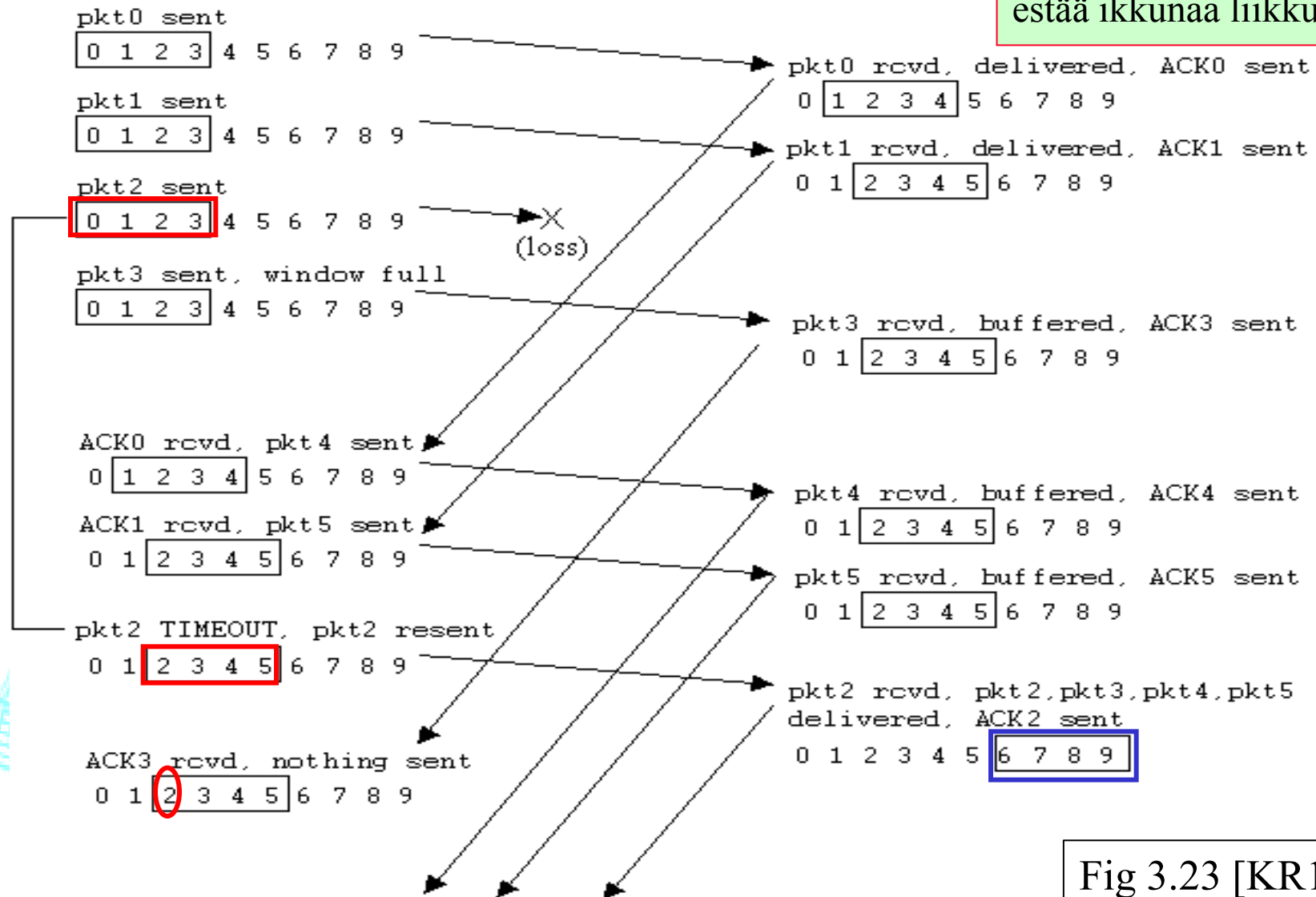


Fig 3.23 [KR12]

Ikkunankoko

- Pakettinumeroille varatun kentän koko vaikuttaa myös ikkunankokoon
 - Yleensä jokin kakkosen potenssi
 - Kentän koko k bittiä \Rightarrow käytössä 2^k pakettinumeroa
- Kun paketit numeroidaan $0, 1, \dots, n$, niin ikkunan koko saa olla korkeintaan:

$n+1$ kpl

MIKSI NÄIN?

Harjoitus-
tehtävinä!

Go-Back- n : n (k bit sekvenssinumeroilla $< 2^{k-1}$)

Valikoiva toisto: $(n+1)/2$ (k bit sekv.num. $< 2^{k-1}$)

Vastaanottajan pitää tietää onko kyse uusista paketeista vai uudelleenlähetyksistä

Yhteenveto menetelmistä

- Tarkistussumma
- Ajastin
- Järjestysnumero
 - Uudelleenlähetys vai uusi paketti
- Kuittaukset
 - Positiiviset ACK, tuplakuittaukset
 - Negatiiviset NAK
- Ikkunat, liukuhihnoitus

Kts Table 3.1 [KR12]

ACK vai NAK

- ACK ja NAK signaaleja voidaan yhdistellä
- TCP perustuu vain ACKiin
- TCP: Kumulatiivinen ACK
 - Tupla ACK $X = \text{NAK } X+1$
- Selektiivinen ACK
 - Parempi toteuttaa oma NAK signaali, tehokkaampi
 - Ei välttämättä tarvita NAKia
- Pelkkä NAK