

3. Kuljetuskerros

3.1. Kuljetuspalvelu

■ 'End- to- end'

– prosessilta prosessille looginen yhteys

- portti

– verkkokerros koneelta koneelle

- IP-osoite

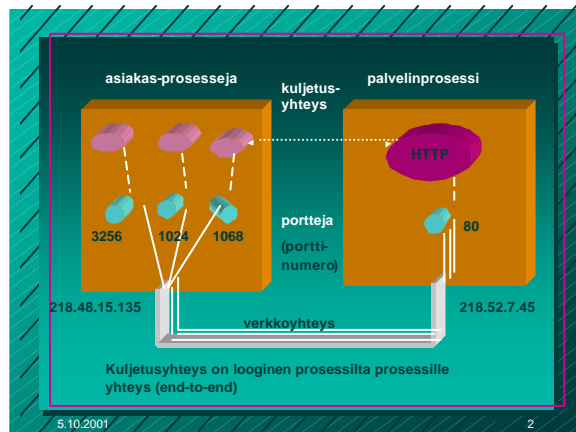
■ peittää verkkokerroksen puutteet

– jos verkkopalvelu ei ole riittävän hyvä, sitä voidaan parantaa kuljetuskerroksella

- kuljetuskerros huomaa verkkokerroksen kadottamat paketit ja pyytää niiden uudelleenlähetystä

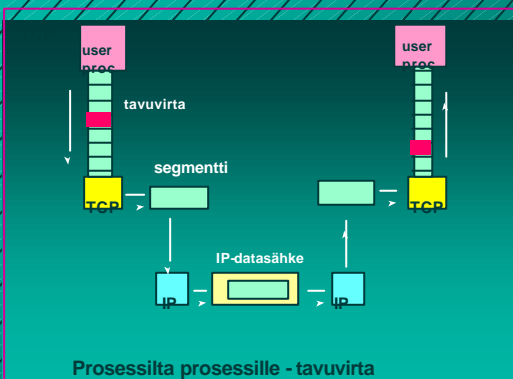
5.10.2001

1



5.10.2001

2



5.10.2001

3

kuljetuspalvelut parantavat verkkopalveluja

Sovelluksen näkemä palvelun laatu (Quality of Service, QoS)

kuljetuskerroksen palvelut

verkkokerroksen palvelut

kuljetuskerroksen palvelut

verkkokerroksen palvelut

5.10.2001

4

Sovelluksen vaatimuksia kuljetuspalvelulle:

- Virheetön, luotettava
 - järjestyksen säilyttävä
 - kaksoiskappaleet karsiva
 - mielivaltaisen pitkiä sanomia salliva
 - vuonvalvonnan mahdollistava
- Verkkokerros kuitenkin voi
- kadottaa sanomia
 - toimittaa sanomat epäjärjestyksessä
 - viivyttää sanomia satunnaisen pitkän ajan
 - luovuttaa useita kopioita samasta sanomasta
 - rajoittaa sanomien kokoa

5.10.2001

5

Internetin kuljetuskerros

■ UDP (User Datagram Protocol)

– yhteydetön, epäluotettava palvelu

■ TCP (Transmission Control Protocol)

– yhteydellinen, luotettava palvelu

- virhevalvonta

– havaitsee ja korjaa siirrossa syntyneet virheet

- vuonvalvonta

– ei ylikuormita vastaanottajaa

- ruuhkanvalvonta

– huolehtii ettei verkko pääse ruuhkautumaan

5.10.2001

6

Sovelluksien datavirtojen erottaminen

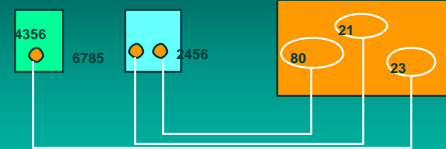
- IP-osoite
 - osoittaa koneen yksikäsitteisesti
- Sovellusprosessi tunnistetaan portinumerosta (16 bittiä =>0-65535)
 - jokaisessa lähetetyssä segmentissä on
 - lähetäjän porttinumero
 - vastaanottajan porttinumero
- Yleisillä palvelimilla omat varatut portinumerot (0-1023)
 - SMTP 25, HTTP 80, jne

5.10.2001

7

Asiakkaalle kuljetuskorros usein automaattisesti antaa käyttöön jonkin vapaan portinumeron yhteyden ajaksi

Palvelimilla kiinteät numerot

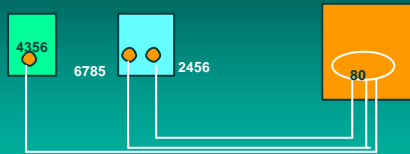


Kolme yhteyttä: 4356 <=> 23, 6785 <=> 21, 2456 <=> 80

5.10.2001

8

Tarvitaan sekä lähteen että kohteen portinumerot

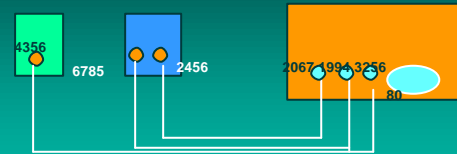


Kolme yhteyttä: 4356 <=> 80, 6785 <=> 80, 2456 <=> 80

5.10.2001

9

Palvelimessa yhteyksille uudet portinumerot, jotta portti 80 voi ottaa vastaan uusia yhteyspyyntöjä

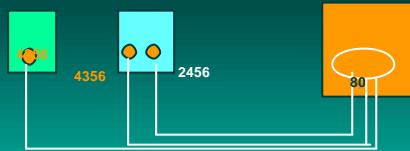


Kolme yhteyttä: 4356 <=> 2067, 6785 <=> 1994, 2456 <=> 3256

5.10.2001

10

Eri koneissa voidaan ottaa sama numero!



Kolme yhteyttä: 4356 <=> 80, 4356 <=> 80, 2456 <=> 80!

Kuljetusyhteydellä käytetään apuna myös IP-osoitetta:
=> koneilla eri IP-osoitteet, joten yhteydet pystytään erottamaan

5.10.2001

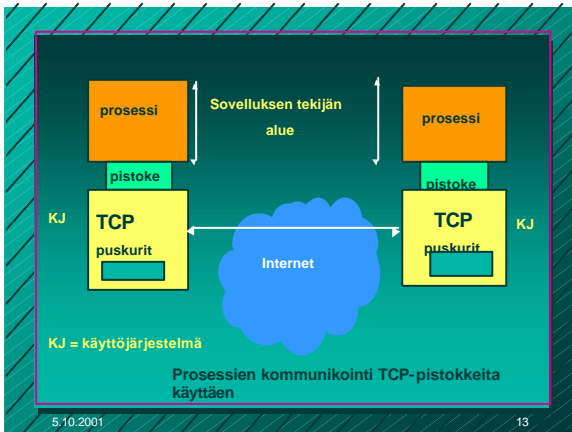
11

Pistokerajapinta (Socket interface)

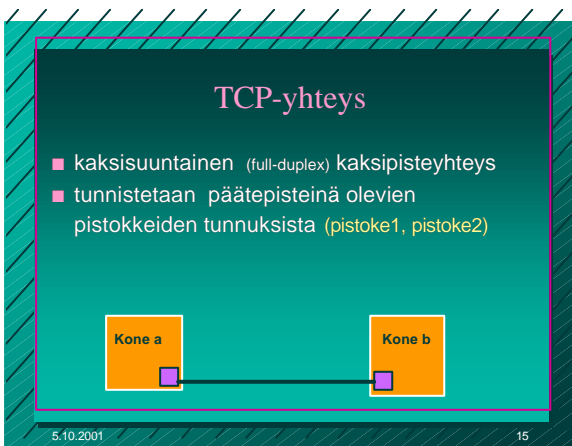
- Verkkopalvelun ja sitä käyttävän sovelluksen rajapinta
 - yleensä käyttöjärjestelmän tarjoama palvelu
 - pistokerajapinta alunperin Berkeley Unixin mukana, nyt lähes kaikissa käyttöjärjestelmissä
 - miten verkkoprotokollan tarjoamiin palveluihin päästään käsiksi sovelluksesta

5.10.2001

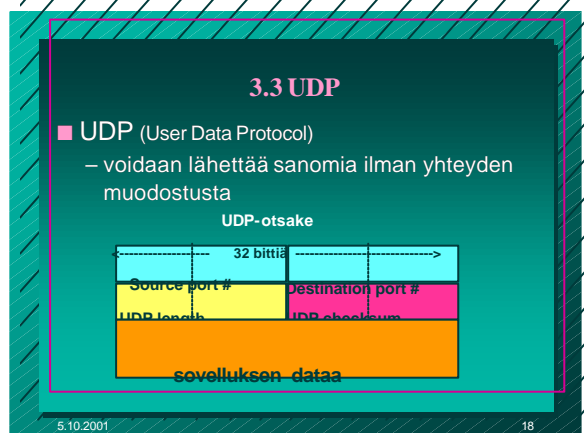
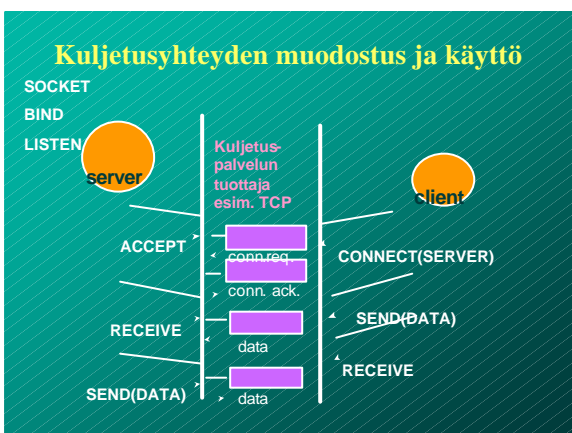
12



- **pistoke (socket)**
 - TCP-yhteyden päätepiste sovellukselle
 - lähettäjällä ja vastaanottajalla oma pistoke
 - pistokenumero 48 bittiä
 - koneen 32 bitin IP-osoite
 - 16 bitin porttinumero
- 5.10.2001 14



- ### TCP:n pistokeprimitiivit
- SOCKET luo uuden yhteyden päätepistepistoke
 - BIND anna pistokkeelle osoite
 - LISTEN halukas vastaanottamaan yhteyksiä
 - ACCEPT jää odottamaan yhteyksirytsiä
 - CONNECT yritä muodostaa yhteys
 - SEND lähetä dataa yhteyttä pitkin
 - RECEIVE vastaanota dataa yhteydeltä
 - CLOSE pura yhteys (symmetrinen)
- 5.10.2001 16



UDP-tarkistussumma

- Virheen havaitsemista varten otsakkeeseen liitetään tarkistussumma
 - kaikki segmentin 16 bitin sanat lasketaan yhteen ja summasta otetaan **yhden komplementti**
 - = muutetaan ykköset nolliksi ja nollat ykkösiksi
 - vastaanottaja laskee taas kaikkien segmentin sanojen (mukana myös tarkistussumma) summan
 - jos tulokseksi saadaan 16 ykköstä, niin ok!

5.10.2001

19

Esimerkki

- Lasketaan yhteen kolme 8 bitin mittaista sanaa:

- Lähettäjä vastaanottaja

```
1011 0100    1011 0100
0111 0101    1111 0101
1000 1101    1000 1101
=====
1011 0110    0100 1001
                    =====
```

0100 1001

Yhden komplementti

5.10.2001

20

- Miksi tarvitaan tarkistussumma?
 - Kaikki siirtoyhteyserrokset eivät suorita tarkistuksia
- UDP-tarkistussumma ei ole kovin tehokas havaitsemaan virheitä!
- Se ei myöskään yritä toipua virheistä!
 - Jotkut toteutukset voivat tuhota virheellisen segmentin
 - jotkut antavat se sovellukselle varoituksen kera

5.10.2001

21

UDP:n etuja:

- Yhteydetön
 - aikaa ei kulu yhteyden muodostamiseen ja purkamiseen
 - ei tarvita resursseja yhteyden tilatietojen ylläpitoon
- Otsake pienempi => pienempi yleisrasite => tehokkaampi
- Ruuhkanvalvonta ei säännöstele liikennettä

5.10.2001

22

Tehtäviä:

- Lähetetään 10 tavun viesti UDP:llä.
 - Miten kauan kestää lähettäminen, jos lähetyksenopeus on 56 kbps?
 - Miten suuri on etenemisviive, jos etäisyys lähettäjältä vastaanottajalle on 1000 km?
 - Miten suuri on UDP-otsakkeen aiheuttama yleisrasite (overhead)?

5.10.2001

23

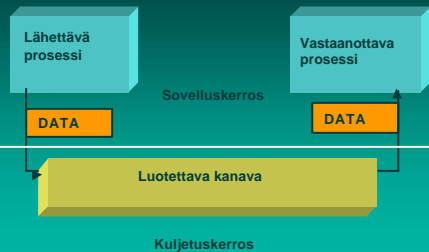
UDP:n käyttö

- Vaikka UDP on epäluotettava, se sopii monien sovellusten tarpeisiin:
 - Remote file server (NFS)
 - multimedia
 - Internet-puhelin
 - verkon hallinta (SNMP)
 - reititys (RIP)
 - nimipalvelu (DNS)
- Miksi nämä sovellukset suosivat UDP:tä?

5.10.2001

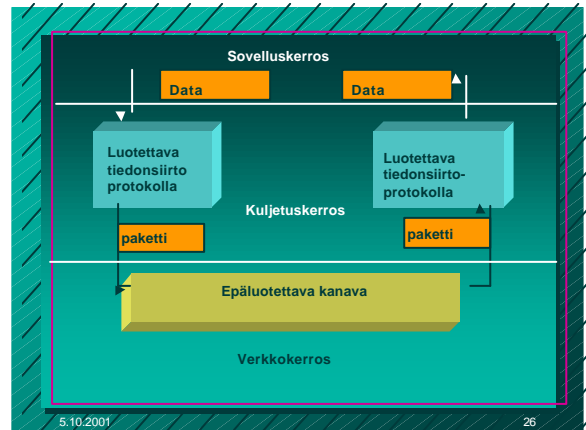
24

3.4 Luotettava tiedonsiirto



5.10.2001

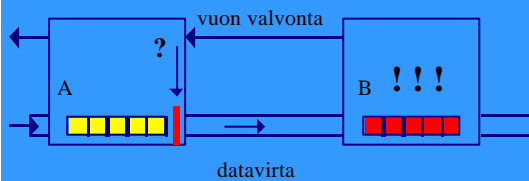
25



5.10.2001

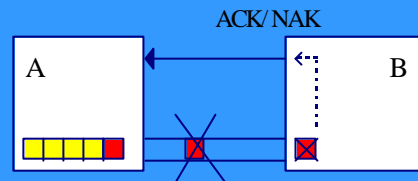
26

Vuon valvonta



- X-ON / X-OFF : GO! | STOP!

Kohinainen kanava



- sanoma vääristyy => virhetarkistus
- sanoma katoaa => ajastin ja uudelleenlähetyks
- duplikaattien havaitseminen

Yksinkertainen Stop and wait -protokolla

- Oletus
 - virheetön siirto => ei huolta virheistä, mutta vuonvalvontaa tarvitaan
- lähettäjä
 - lähettää sanoman
 - odottaa lupaa lähettää seuraava sanoma
- vastaanottaja
 - käsittelee sanoman
 - lähettää tiedon (=antaa luvan) lähettäjälle

5.10.2001

29

Entä jos virheitä?

- Sanomissa virheitä tai sanomat voivat puuttua kokonaan
- Myös kuittaukset voivat kadota
- Tarvitaan
 - virheen havaitseminen ja korjaaminen
 - tarkistussumma
 - kuittaus
 - uudelleenlähetyks
 - sanomien numerointi
 - uudelleenlähetyksajastin

5.10.2001

30

Monimutkaisempi stop and wait -protokolla

■ ajastin lähettäjälle

- jos kuittausta ei kuulu, sanoma lähetetään automaattisesti uudelleen
- kuittaus: ACK = 'ok, lähetä seuraava'
- uudelleenlähetykset synnyttää kaksoiskappaleita!

■ Sanomanumerointi

- jotta vastaanottaja tunnistaa kaksoiskappaleet
- Miten paljon numeroita tarvitaan?
 - » Numero vie tilaa sanomassa!

Stop and wait -protokollan suorituskyky

■ Esim. satelliittiyhteydellä

- 50 kbps, kiertoviive ~520 ms, sanoma 1000 bittiä
- kanavan käyttöaste < 4%

■ => lähetetään useita sanomia ja sitten vasta odotetaan kuittauksia

- ideaali: lähetykset liukuhihnalla (pipeline)
- lähetykset ja kuittaukset limittyvät
- ei mitään odottelua
- lähetyiskanava koko ajan käytössä
- suorituskyky kasvaa

Liukuvan ikkunan protokolla

(Sliding Window)

■ Lähetyksikkuna

- ikkunan koko
 - montako sanomaa saa korkeintaan olla kuittaamatta
 - järkevä koko riippuu yhteyden tyypistä ja vastaanottajan kapasiteetista
- sisältö = mitkä sanomat saa lähettää
 - sanomalla järjestysnumero
 - rajallinen, N bittiä => 2^N arvoa
 - numerot käytettävä järjestyksessä

5.10.2001

33

■ Lähettäjä joutuu odottamaan vasta, kun kaikki ikkunan sanomat on lähetetty

- eli numerot käytetty

■ Kun kuittaus saapuu => ikkuna liikuu

- seuraavat numerot tulevat luvallisiksi

■ eli

- lähettäjä: tietyllä hetkellä sallittujen numeroiden joukko = lähettäjän ikkuna
- mitkä sanomat saa lähettää "etukäteen" odottamatta kuittausta

5.10.2001

34

■ Vastaanottajan ikkuna

- kullakin hetkellä sallittujen numeroiden joukko
- mitä sanomia suostuu vastaanottamaan

- kuittaus muuttaa myös vastaanottajan ikkunan

■ ikkuna pysäyttää sanomien lähetyksen

- seuraava sanomanumero ei ole lähetyksikkunassa

■ ikkuna estää sanoman vastaanoton

- saadun sanoman numero ei ole vastaanottoikkunassa

Kun ikkunan koko on 1

■ Aina vain yksi sanoma kuittaamattomana

- => One Bit Sliding Window -protokolla
- ~ stop and wait -protokolla

■ sanomanumerot 0 ja 1 riittävät

■ ACK-sanoma identifioi viimeksi vastaanotetun virheettömän sanoman

- jotta kuittausduplikaatti ei voi kuitata väärää sanomaa

- ACK ilmoittaa joko

- » seuraavaksi odotetun sanoman numeron
- » viimeksi vastaanotetun sanoman numeron

5.10.2001

36

■ entä kun tapahtuu virhe?

- kaksi eri tapaa hoitaa
- **toisto virheestä lähtien (go back n)** (tai paluu n:ään)
- **valikoiva toisto (selective repeat)**

5.10.2001

37

Toisto virheestä eli Paluu n:ään ("Go back n")

- virheellisen sanoman havaittuaan
 - vastaanottaja hylkää kaikkia sen jälkeiset sanomat
 - eikä lähetä niistä kuittauksia
- kun lähettäjä ei saa kuittauksia,
 - sen lähetyksikuna 'täyttyy'
 - eikä se voi enää lähettää
- lähettäjän ajastimet laukeavat aikanaan ja
 - **virheellinen sanoma**
 - **sekä kaikki sen jälkeen lähetetyt sanomat lähetetään uudelleen**
- tehoton, jos paljon virheitä ja iso ikkuna

Valikoiva toisto

- vastaanottaja hyväksyy kaikki kelvolliset sanomat
 - se kuittaa sanomat
 - puskuroi ne ja toimittaa eteenpäin oikeassa järjestyksessä
 - » tarvitaan puskuritilaa
- lähettäjä ei saa kuittausta virheellisestä sanomasta
 - ajastin laukeaa ja sanoma lähetetään uudelleen
 - **lähettää uudelleen vain virheellisen sanoman**
 - ikkuna liukuu nytkin tasaisesti
 - » yksi puuttuva kuittaus voi pysäyttää lähetyksen