2013-
2014

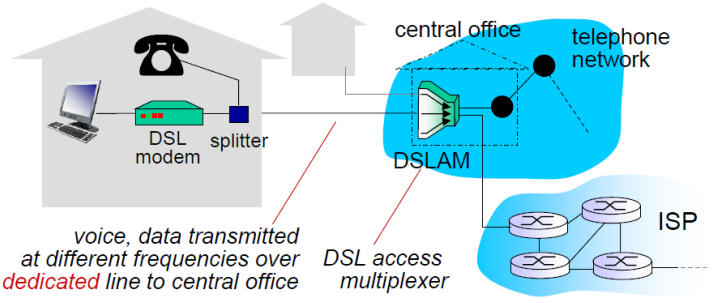
Computer Networking A Top-Down Approach

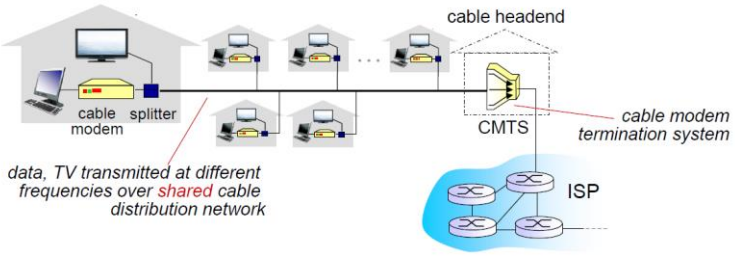
OPGELOSTE VRAGEN OVER DE CURSUS

GILLES CALLEBAUT

Chapter 1:
Computer Networks and the Internet

Bespreek de gelijkenissen en verschillen tussen DSL en Cable Network





Verschillen

De DSL modem maakt gebruik van de bestaande telefoonlijn om te communiceren met de DSLAM.
De DSL modem vormt de digitale data om naar high-frequency tones voor transmissie over de telefoonlijn.

De telefoonlijn is een dedicated line naar de CO.

Cable Internet acces maakt gebruik van de televisie kabel.

De Cable modems delen het HFC netwerk in 2 kanalen, een downstream en een upstream kanaal.

Cable internet access is een shared broadcast medium.
Omdat het upstream kanaal ook ge-shared is zal er gebruik moeten gemaakt worden van een distributed multiple acces protocol.

Gelijkenissen

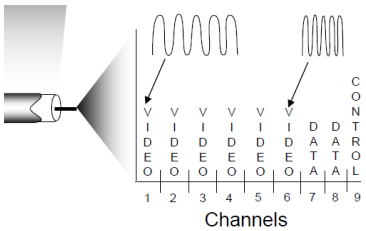
Downstream & upstream snelheden zijn verschillend,
Asymmetrische acces.

De analoge signalen worden terug in digitale formaten gezet bij de DSLAM.
De splitter scheidt de data (upstream & downstream) en de telefoon signalen en forward de data signalen naar de DSL modem.

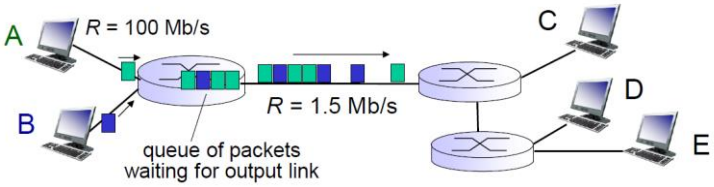
De DSLMA scheidt ook de data en de telefoon signalen en zend de data over het internet.

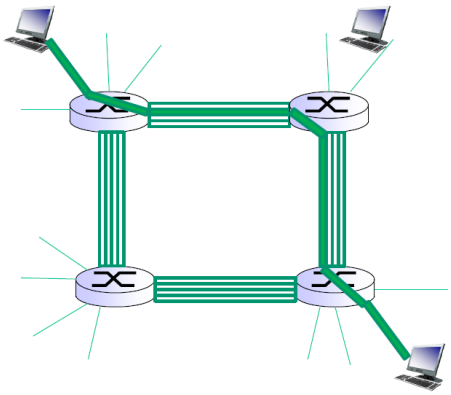
De analoge signalen worden terug in digitale formaten gezet bij de CMTS.

Beiden maken ze gebruik van FDM voor het opsplitsen in frequentie banden.



Bespreek de gelijkenissen en verschillen tussen Packet – en Circuit switching





Verschillen

Store-and-forward
Het volledige packet moet arriveren vooraleer deze naar de volgende link kan gestuurd worden.
Queuing and loss,
Als de pakketten sneller toekomen dan worden verstuurd naar de volgende link dan ontstaat er een queuing delay en kan dan uiteindelijk resulteren in pakket verlies. (output buffer loopt vol)

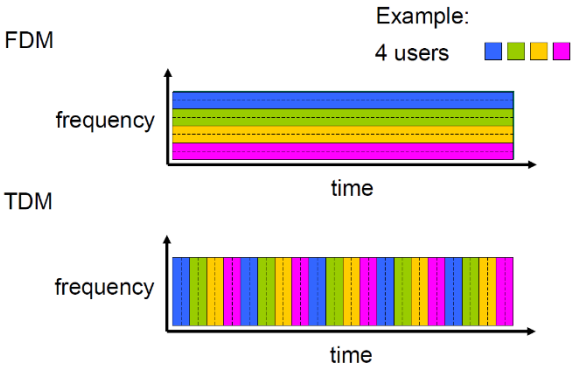
Een routing algoritme bepaalt het source-destination pad.
De router kijkt in zijn forwarding table om te zien welke pakketten naar welke ouput linken moeten worden gestuurd.

Beschikt over de volledige transmissiesnelheid van de link.

Connection set-up delay

Het netwerk stelt een dedicated end-to-end connectie op.
Zo reserveren ze een constante (gegarandeerde) transmissiesnelheid.

De link wordt opgedeeld in verschillende frequentie of tijd banden.



Bij FDM wordt de bandbreedte opgedeeld onder de gebruikers.
Bij TDM wordt tijd opgesplitst in frames en elke frame is opgedeeld in verschillende tijd slots.

Bij stille periodes zullen dus bandbreedtes en tijdslots leeg blijven.
En ze kunnen dus minder gebruikers aan.

Gelijkenissen

Beiden ondervinden transmission en propagation delay

Welke vertragingen kunnen optreden bij een node

Processing delay
Tijd nodig om de pakket header te onderzoeken en te bepalen naar waar hij het moeten sturen.

Queuing delay
Het pakket moet wachten om 'getransmitteerd' te worden op de link.
Er kan ook sprake zijn van packet loss wanneer de snelheid aan toevoer aan nieuwe pakketten groter is dan transmissiesnelheid.

$$\frac{La}{R} \leq 1$$

Met a arriving packet speed (packet/sec).

Transmission delay
De tijd nodig om alle bits van het pakket op de link te zetten.

$$\frac{L}{R}$$

Propagation delay
De tijd nodig om te propageren van het begin van de link naar de andere router:

$$\frac{d}{s}$$

End-to-End delay

$$d_{nodal} = N(d_{proc} + d_{queu} + d_{trans} + d_{prop})$$

Throughput
De throughput van de connectie wordt bepaald door de bottleneck link, daar waar de throughput minimaal is.

Chaprter 2:
Application layer

Bespreek de gelijkenissen en verschillen tussen client-server en P2P architectuur van applicaties

Client-server architecture

Peer-to-Peer architecture

Verschillen

Always-on host: (dedicated) server
met een permanent IP adres
Clients die services vragen aan de server
Communicatie starten met server, kan geconnecteerd zijn met tussenpozen. Geen directe verbinding met elkaar.

De applicatie maakt gebruik van directe communicatie tussen paren van 'intermettently' geconnecteerde hostes, peers.

Self-scalability,
elke peer genereert zowel werk als service capaciteit aan het systeem door files de distribueren naar andere peers.
Cost effective, want er is geen nood aan hoge server infrastructuur en server bandwidth.

Problemen:
- Assymetrische bandwidth usage.
- Security
- Incentives (stimuli) om gebruikers te stimuleren om hun recources te verlenen aan the applicaties.

Gelijkenissen

De applicatie architectuur wordt ontworpen door de applicatie developer en zegt hoe de applicatie is gestructureerd op verscheidenen end-systems.
De process communicatie is bij beiden client process & server proces.

Hoe communiceren de applicaties met elkaar

Het zijn de processen[1] op verschillende hosts dat met elkaar communiceren door middel van het uitwisselen van **messages**.
Deze messages worden verstuurt en ontvangen door een socket[2].
De developer kan enkel op de transport laag kant de keuze van transport protocol maken.

(Meer over sockets bij de transportlaag)

[1] Een proces is programma dat draait op een end system.

[2] Een socket is een interface tussen de applicatie laag en de transport laag binnen een host.
Ook wel de API genaamd.

Bespreek HTTP en zijn verschillende soorten connecties

Een web pagina bestaat uit objecten. In de basis HTML file staan de referenties naar die objecten.
Deze kunnen worden aangesproken via een URL: hostname/object's_path_name

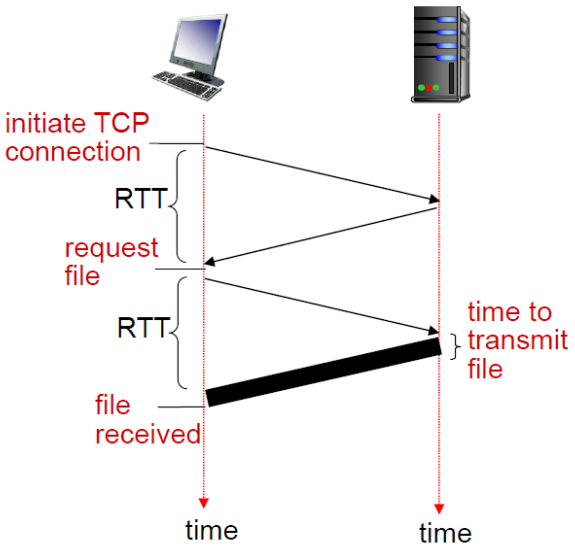
Web browsers implementeren de client side en web servers de server side van HTTP.

HTTP gebruikt TCP als onderliggend transport protocol.
De client start een TCP connectie met de server, creëren van socket met poort 80.
Server accepteert de TCP connectie.
HTTP messages worden uitgewisseld.
TCP connectie wordt afgesloten.

http is 'stateless' er wordt dus geen informatie over de clients bijgehouden.

HTTP connecties:

Non-persistent http	**Persistent http**
Hierbij zal de TCP connectie gesloten worden na dat de server een object heeft verzonden. Dus elke TCP connectie transporteert 1 request messages en 1 response message.	De server laat de TCP connectie open. De connectie wordt gesloten na een timeout interval.
Per object (2*RTT + file transmission time)	RTT (starten connective) + RTT (back-to-back requests) + file transmission time voor alle objecten



Bespreek P2P distribution

De verzameling van alle peers die deelnemen in het distribueren van een specifieke file noemt een torrent.
Zo'n file wordt opgesplitst in 256 KBytes, chunks genaamd , die de peers verzenden/ontvangen.

Een tracker houdt bij welke peers deelnemen in de torrent.
Als er een nieuwe peer bijkomt zal de tracker 'randomly' de IP adressen van een aantal peers verzenden naar de nieuwe peer. Deze peer zal nu een TCP connectie proberen opstarten met deze lijst van IP adressen.
De peer zal aan haar 'neighboring peers' vragen om een lijst door te geven van de chuncks die ze bezitten, en kan ze zo vragen aan de respectievelijke peers om de chuncks door te sturen die ze nog niet heeft, rarerst first.

De peer houdt bij wie haar de chuncks doorstuurt aan de hoogste snelheid (dit gebeurt om de 10 sec.), deze top 4 zullen dan ook chuncks ontvangen van haar, deze peers worden de unchoked peers genoemd.
Om de 30 seconden neemt ze 'random' een nieuwe neighbor en begint daarnaar chuncks te verzenden, deze peer wordt de optimistically unchoked peer genoemd. Zo kan deze peer misschien in haar top 4 terrecht komen.

De andere neighbor peers (choked) ontvangen geen chuncks van haar.

Dit mechanisme wordt ook wel gerefereerd als tit-for-tat.

Chapter 3: Transport layer

Wat zijn de 2 internet transport-laag protocollen en vergelijk ze met elkaar

TCP	UDP
Verschillen	
Reliable, connection-oriented service Correct & in-order delivery Flow control Sequence numbers Congestion control	Unreliable, connectionless service
Gelijkenissen	
Integrity chechking door error-detection fields in de segment headers toe te voegen Process-to-Process data delivery Geen delay en bandwidth guarantee	

Connectionless transport: UDP

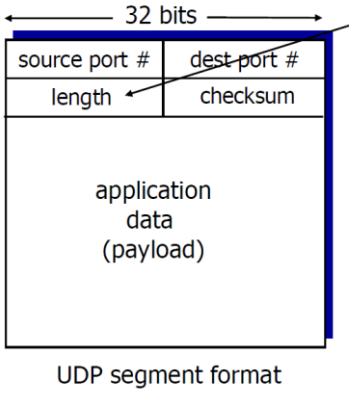
UDP voegt bijna niets toe aan IP, buiten het multiplexen/demultiplexen en lichte error checking. (best effort)

UDP is connectionless omdat er geen gebruik wordt gemaakt van handshaking tussen zender en ontvanger.

UDP wordt gebruikt door applicatie om volgende redenen:
- minimum sending rate en je kan verlies tolereren (want geen congestion control)
- geen delay door handshaking
- UDP houdt geen connectie open en houdt zich ook niet bezig met zijn parameters, zo kan de applicatie meer actieve clients bedienen.
- kleine packet header overhead (UDP header maar 8 bytes)

UDP segment header

De lengte is in bytes en drukt uit hoe groot het UDP segment is (incl. header).



UDP segment format

UDP checksum

Detecteren van errors bij getransmitteerde segementen.
Zender voert een checksum[3] uit en voegt deze toe aan de header.
Ontvanger voert een checksum uit en kijkt of deze dezelfde is dan in de header.

[3] De one's complement som van 2 16-bit integers

Bespreek multiplexing en demultiplexing

Demultiplexing is een proces waarbij de header info gebruikt wordt om de data van het transport-laag segment naar de correcte socket brengen.
Multiplexing is dan weer het verzamelen van data chunks van de source host van verschillende sockets, deze dan 'encapsulaten' met transport header informatie om segmenten te creëren en dat dan door te sturen naar de netwerk laag.

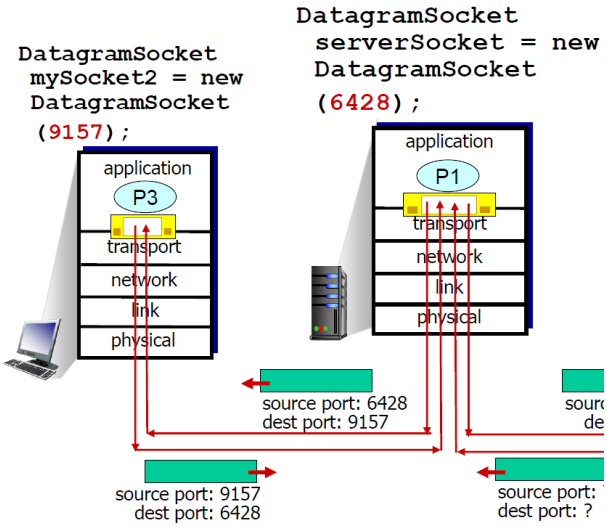
Demultiplexing

De host ontvangt een IP datagram via het IP adres en het poort nummer stuurt hij de segment door naar de bedoelde socket.

Connectionless demultiplexing (via UDP socket)

De host ontvangt een UDP segment, hij checkt de destination poort nummer en stuurt het UDP segment door naar de socket met dat nummer.
Merk op:

Zelfde destination port, maar verschillende source IP adressen/ source port nummer worden allemaal verstuurd naar dezelfde (destination) socket.

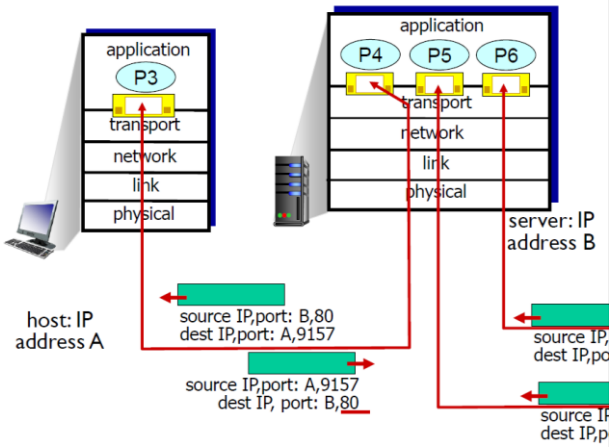


Connection-oriented demux (via TCP socket)

Een TCP socket is geidentificeerd door een 4-tuple:
(Source IP, source port#, destination IP, destination port#)

Al deze waarden worden in rekening gebracht om het segment door te verbinden naar zijn socket.

De server host kan meerdere simultane TCP sockets hebben.



Chapter 4:
The network layer

Bespreek Connection en Connectionless service

Deze services zijn host-to-host services geleverd door de netwerk laag voor de transport laag.
In tegenstelling tot de connection-oriented service van de transportlaag die geimplimnteerd zat aan het einde van het netwerk van de end systems, zal bij de netwerk laag de connection service geimplenteerd zitten in de routers van het netwerk alsook in de end systems.

Virtual circuits

VC setup:
De 'sending' transport laag contacteert de netwerk laag, die dan de VC opstelt.
De netwerk laag bepaalt het pad tussen verzender en ontvanger, en bepaalt ook de VC nummers voor elke link langs het pad.
De netwerk laag vult de forwarding tables aan.
De routers tussen de 2 end systems zijn dus volledig bewust van de VCs pakketen die erdoor lopen, dus de VC routers onthouden de connection state information.

Data transfer:
Pakketten kunnen worden verzonden. De VC nummer wordt gewijzigd van link tot link.

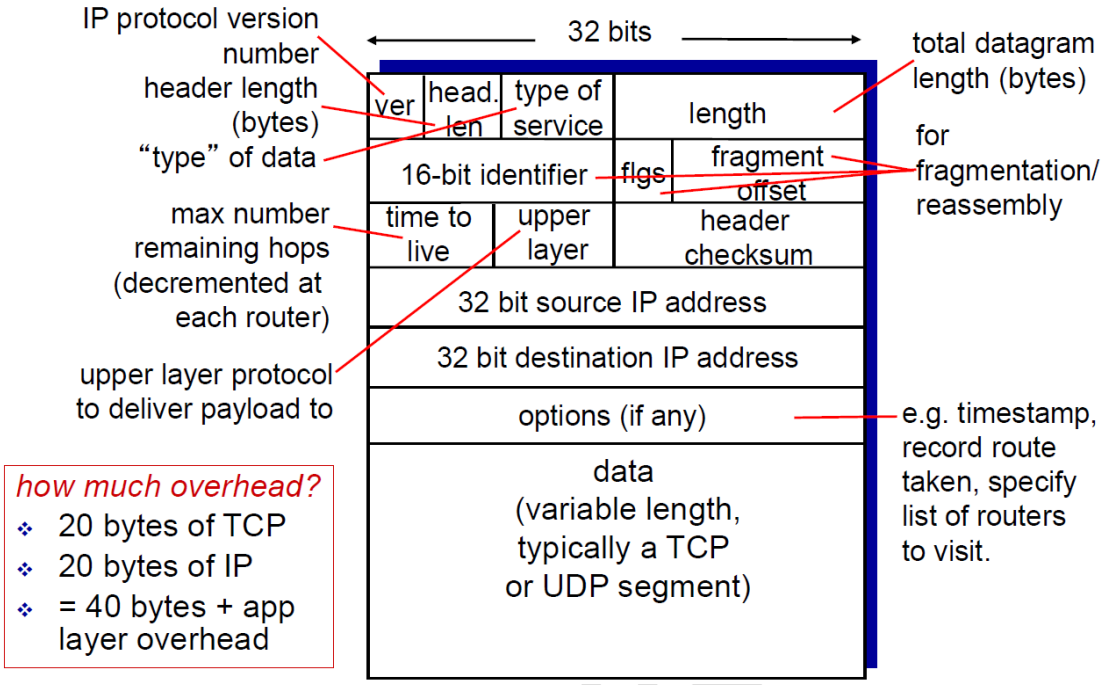
VC teardown:
De ontvanger wordt op de hoogte gesteld van de eindiging van de VC.
De forwarding tables worden geüpdatet dat de VC niet langer bestaat.

Datagram Networks

Geen call setup op de netwerk laag.
Datagrammen worden verstuurd op basis van hun destination host adres.
De routers behouden geen informaton over de end-to-end connections.

De pakketen worden ge-forwarded door middel van de longest prefix matching rule, om het pakket op de correcte link interface te zetten.

Bespreek het IP datagram formaat



how much overhead?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

Bespreek IP fragmentatie

MTU is de maximale grootte van een link-laag frame.
Vermits verschillende links verschillende link laag protocollen kan gebruiken zullen er verschillende MTUs zijn.
De IP datagrammen zullen dus worden opgesplitst en in nieuwe frames worden gestoken.

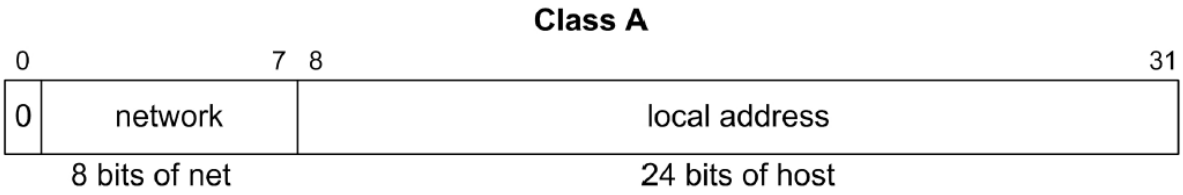
Om deze fragmenten opnieuw samen te steken werden er identification, flag, en fragmentation offsett velden toegevoegd aan de IP datagram header.

Het ID wordt onderzocht om te bekijken welke fragments samen horen.
De flag geeft aan of er nog een pakket achter moet moeten (1), laatste pakket (0).
De offset geeft aan op welke plaats men de data van het pakket moet invoegen (veelvoud van 8 bytes).

Wat is klassenadressering en wat is het probleem hierbij, bespreek ook een oplossing voor dit probleem

Klassenadressering

IP adressen worden voorgesteld als volgt, NET.HOST.



Netnummers:

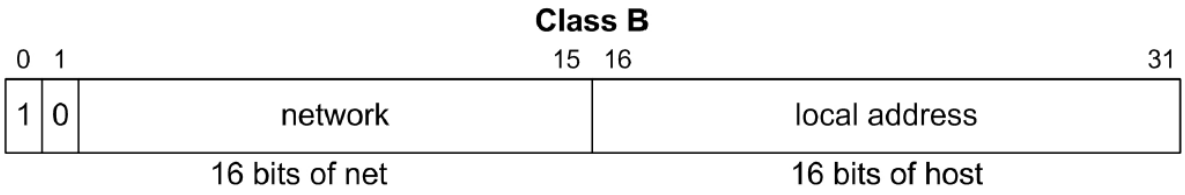
1.0.0.0 – 126.0.0.0

Waarbij 127.X.X.X het loopback adres is, pakketten worden lokaal verwerkt en behandeld als binnenkomende pakketten. Het netnummer 10.0.0.0 is gereserveerd voor privaat gebruik.

Hostnummers:

0.0.1 – 255.255.254

Waarbij 0.0.0 het netnummer aanduidt en 255.255.255 gereserveerd is voor broadcasting.

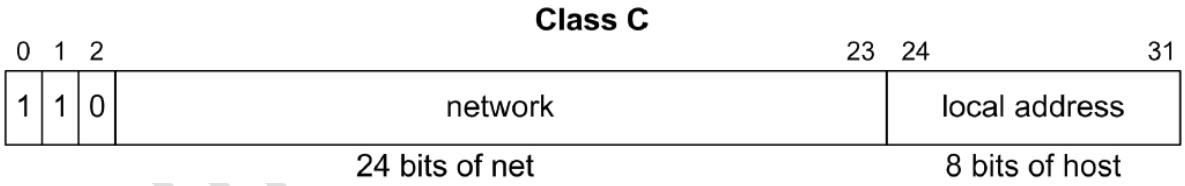


Netnummers:

128.0.0.0 – 191.255.0.0

Hostnummers:

0.1 - 255.254



Netnummers:

192.0.0.0 – 223.255.255.0

Hostnummers:

1 – 254

De routing tabellen dienen dus enkel netnummers te bevatten, enkel bij de laatste router zal men moeten kijken naar het host nummer.

Heel wat beschikbare IP adressen zijn nooit toegekend en gaan dus verloren.
Oplossingen voor dit probleem zijn de klassenloze adressering, CIDR en IPv6.

Klassenloze adressering

Subnetten

De netwerkbeheerder gaat het lokaal netwerk (klasse X adresruimte) in subnetten onderverdelen.

Het onderscheidt tussen de verschillende 2^n subnetten worden gemaakt door de n eerste bits van het hostnummer te gebruiken, zodat je de subnetten kan onderscheiden.

Voorbeeld: Een klasse C netwerk met 4 subnetten
de eerste 2 nummers van het 'hostnummer' vormen het subnetnummer. Van de overblijvende 6 bits kunnen er $2^6 - 2 = 62$ hosts aangesloten worden.

203.15.65.00 000000
203.15.65.01 000000
203.15.65.10 000000
203.15.65.11 000000

Voor de buitenwereld zitten we nog altijd met klassen, CIDR daarentegen werkt niet met klassen.

CIDR

$u.v.w.x/nn$ met nn het aantal bits die gereserveerd zijn voor het netnummer.
Meestal maakt men gebruik van een netmask, dit zijn 32 bits waarbij het aantal enen weergeeft welke bits bestaan uit het netnummer.

Voorbeeld:

IP adres:	203.15.165.69
Netmask:	255.255.255.192

Het netnummer is dus in dit geval 203.15.165.64 en het hostnummer is 5.

Hoe verkrijgt een host een IP adres

Het IP adres kan een vast IP adres zijn die hard gecodeerd is door de system admin in een file.

IP adressen worden echter meestal automatisch toegekend.
Dit gebeurt door DHCP, de netwerk admin kan DHCP configureren dat een host altijd hetzelfde IP adres ontvangt of een verschillend en tijdelijk IP adres ontvangt elke keer dat de host het netwerk betreed.
Door deze automatische functie wordt DHCP vaak gerefereerd als een plug-and-play protocol.

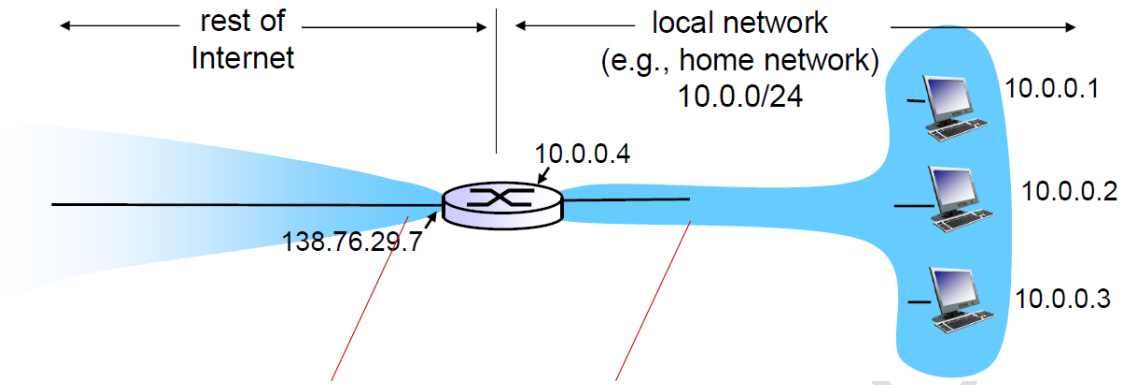
DHCP is een client-server protocol.

Het proces:

1. DHCP server discovery
   DHCP discover msg, UDP pakket naar poort 67, in een IP datagram met destination IP 255.255.255.255 en source IP adres is 0.0.0.0
2. DHCP server offer(s)
   DHCP offer msg, een broadcast met daarin het transactie ID van de discover msg, het IP adres voor de client, network mask en de lease time
3. DHCP request
   DHCP request msg, echoing back the config. parameters.
4. DHCP ACK
   DHCP ACK msg, conformeren van de gevraagde parameters.

Als de client het DHCP ACK pakket heeft ontvangen dan kan de client het DHCP-verkregen IP adres gebruiken voor de lease duur.

Wat is de noodzaak van een NAT en de werking ervan
En welk probleem wordt daardoor ondervonden



Network Address Translation zorgt ervoor dat verschillende computers op een LAN gebruik kunnen maken van dezelfde internetverbinding en daarmee hetzelfde internet adres, zonder hun interne netwerk-adres te hoeven publiceren. De aangesloten computers mogen dan een adres hebben in een van de gereserveerde address-ranges die IPv4 ter beschikking stelt voor privé-netwerken, wat op het internet niet is toegestaan.

Dit heeft een aantal voordelen:

- Computers op het lokale net zijn niet bereikbaar vanaf het internet, aangezien het interne adres niet bekend wordt gemaakt.
- Doordat meerdere computers een openbaar IP-adres delen, wordt de schaarste aan IP-adressen enigszins getemperd.

De router verkrijgt zijn adres van de ISP-DHCP server, en de router fungeert als DHCP server voor de computers binnen het NAT-DHCP-router-controlled home netwerk.

De datagrammen met source of destination adres in het netwerk hebben allemaal een 10.0.0/24 adres.

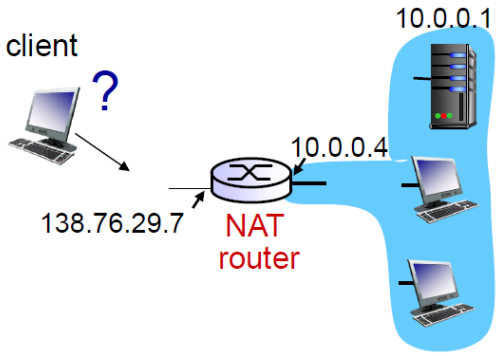
Wat doet de NAT

Het maakt gebruikt van een NAT translation table, waar men zowel IP adressen als poort nummers opslaat.
Zo verandert de NAT de uitgaande datagrammen met een NAT IP adres en een nieuwe poort nummer.
En deze gegevens worden ibjgehouden in de NAT translation tabel.

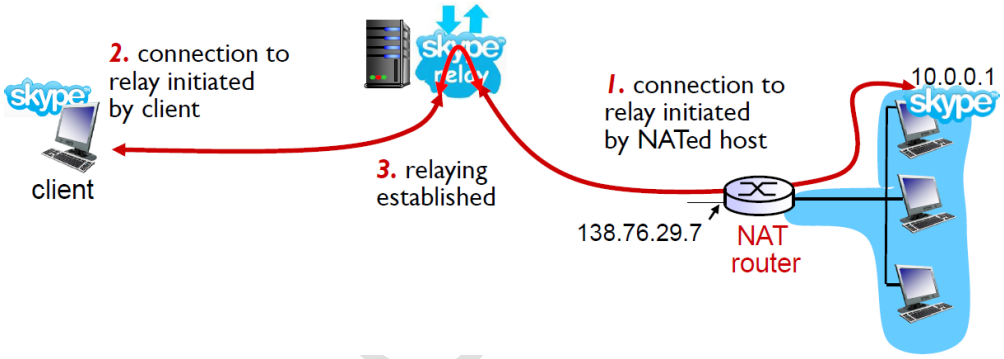
Inkomende datagrammen worden terug verandert naar hun oorspronkelijke IP adres en poort nummer, die opgeslagen zaten in de NAT table.

NAT traversal probleem

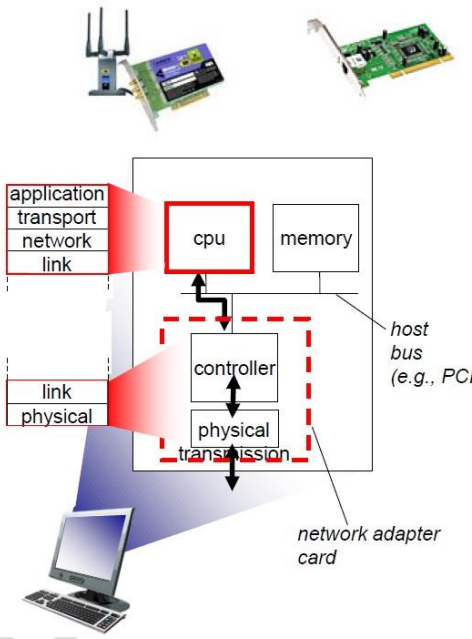
Wat als een client van buitenaf een computer wilt benaderen in het NAT-DHCP-router-controlled home netwerk?



1. Statisch de NAT configureren
   Een inkomende connectie request wordt dan automatisch verstuurd naar een IP adres met hetzelfde poort nummer.
2. Universal Plug and Play:
   IGD Protocol zorgt ervoor dat een NATed host haar publieke IP adres en publiek poot nummer te weten kan komen.
   Het protocol kan een NAP mapping toevoegen/verwijderen.
   En zo kan een externe host een communicatie sessie straten met een NATed host (UDP of TCP).
3. Relaying, connection reversal:
   De Nated client maakt connectie met een relay
   Een externe client connecteert met diezelde relay
   De externe client vraagt via de relay aan de NATed host of hij een TCP verbinding wil opstarten, en zo kan men dan berichten sturen naar elkaar.



Chapter 5:
The link layer



Waar is de link laag geïmplementeerd

Grotendeels is de link laag geïmplementeerd in de netwerk adapter, ook wel network interface card (NIC) genaamd.

Aan de verzendende kant:
De controller neemt een datagram uit het geheugen en maakt een link-layer frame, met daarin error checking bits, rdt, flow control, etc.
Aan de ontvangende kant:
De controller ontvangt de frame en haalt het netwerk laag datagram eruit, deze zoekt ook voor errors, rdt , flow control, etc.

De software componenten van de link laag implementeren een higher-level functionaliteit.

Waarom zijn multiple acces protocollen nodig en bespreek ze

Wanneer we te maken hebben met een broadcast link kunnen er collisies optreden.
Het is dus nodig om protocollen te gebruiken die de transmissies van de noden regelt.

De gewenste karakteristieken

1. Als er 1 node actief is, zenden aan een snelheid R
2. Als er M nodes actief zijn, zenden aan een snelheid R/M
3. Volledig gedecentraliseerd:
   Geen speciale node die de transmissie coördineert
   Geen synchronisatie van clocks, slots
4. Simpel, en dus goedkoop om te implementeren

Onderverdeling

Channel partitioning

Het kanaal verdelen in kleiner stukken, en exclusief gebruik van die stukjes voor een welpbepaalde node.

Random acces

Kaneel niet op gesplitst, collisies mogen voorkomen, die ze dan proberen 'recoveren'.

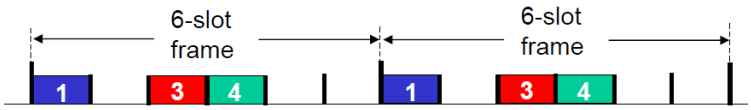
Taking turns

Nodes gaan om de beurt, maar nodes die meer moeten zenden krijgen een langere beurt.

Channel partitioning

TDMA

Tijd wordt verdeeld in **time frames** die dan verder opgedeeld zijn in **time slots**.
Elke tijd slot is dan toegewezen aan één van de nodes.
De lengte van zo'n slot is typisch te grote van een pakket transmissie tijd).



Voordelen

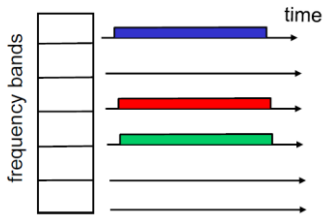
- Geen collisies
- Eerlijk, iedereen R/N bps tijdens elke frame (time)

Nadelen

- De node moet altijd wachten voor te verzenden.
- Een node verzend altijd tegen gemiddelde R/N ook al is enkel die node actief.

FDMA

R bps kanaal wordt verdeeld onder verschillende frequenctiebanden (R/N).



Voordelen

- Geen collisies
- Eerlijk, iedereen R/N bps tijdens elke frame (time)

Nadelen

- Een node verzend altijd tegen gemiddelde R/N ook al is enkel die node actief.

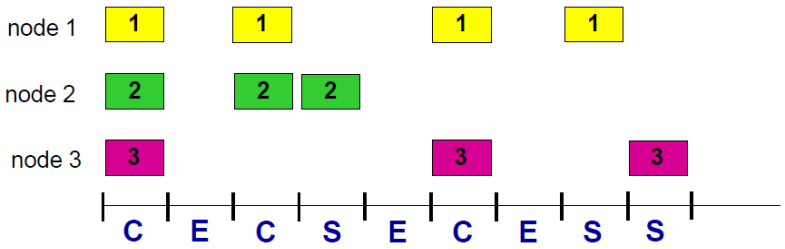
Random Acces Protocols

De node verstuurt het pakket aan de volledige snelheid R, als er een collisie is dan wacht de node een 'randomly' hoeveelheid van tijd voor hij het pakket opnieuw probeert te versturen.
Random acces MAC protocol specifieert hoe een collisie te detecteren en hoe ervan te 'recoveren'.

Slotted ALOHA

We veronderstellen:

- Alle frames L bits lang
- Tijd is verdeelt in slots van L/R seconden
- Nodes starten enkel te verzenden in het begin van een slot
- De nodes weten wanneer een slot begint (synchronisatie)
- Als er een collisie plaats vindt, detecteren alle noden dit voor het einde van de slot



Operatie

- Als een node een frame verzendt
  - Geen collisie: node kan een nieuwe frame verzenden in het volgende slot
  - Collisie: node verzendt opnieuw de frame in volgende sloten tot succes

Voordelen

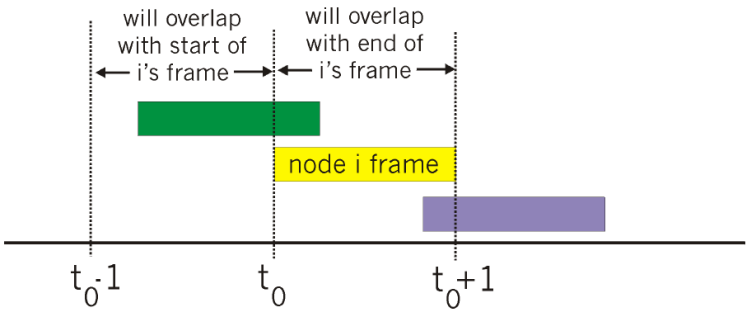
- Enkele actieve node kan verzenden aan R bps
- Heel gedecentraliseerd buiten de synchronisatie van de slots
- simpel

Nadelen

- Collisies en dus verspilde slots
- Lege slots
- syncronisatie van clocks
- Node kunnen misschien de collisie detecteren voor de slot op zijn einde loopt
- Beste efficiëntie: 37%

Pure (unslotted) ALOHA

Geen gebruik maken van slots, gewoon frame verzenden vanaf het arriveert.
De collisies zullen dus sneller voorkomen, de beste efficiëntie is hier nu 18%.



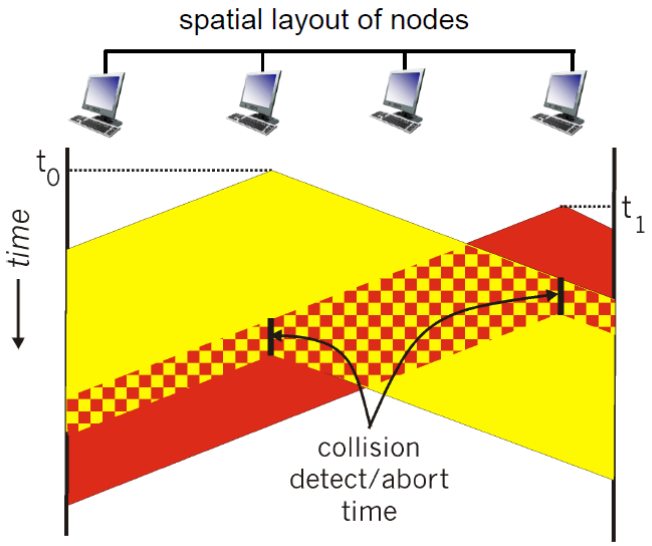
CSMA

Luisteren alvorens de frames te sturen, als het kanaal leeg is verstuur dan u pakket anders wachten.

Botsingen kunnen nog altijd voorkomen vermits we met propagation delays zitten tussen de nodes, en daardoor zal het kanaal de volledige transmissietijd verspilt zijn.

CSMA/CD

De botsingen worden in een korte tijd gedetecteerd, de transmissie wordt stop gezet zodat het kanaal niet nog meer wordt verspilt.



Algorithm:

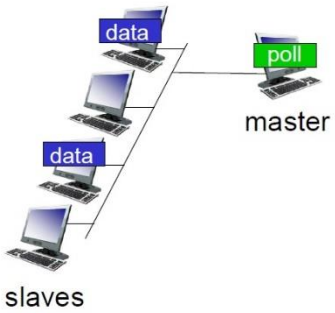
1. NIC ontvangt datagram van netwerk laag en vormt een frame
2. Als het kanaal vrij is, frame verzenden.
   Als het kanaal bezig is, wacht tot het vrij is om te verzenden
3. Als de NIC de volledige frame verzend zonder een botsing te detecteren, goed!
4. Als de NIC een andere transmissie detecteert zal ze haar transmissie stoppen en een jam signal zenden.
5. Na de stopzetting, NIC volgt binary (exponential) backoff:
   a. Achter de n^{de} botsing zal de NIC een K kiezen 'at random' tussen $\{0,1,\dots,2^n-1\}$ en wacht K*512 bit tijden om dan terug te keren naar stap 2.
   b. Dus hoe meer botsingen hoe langer de 'backoff interval'

Beste efficiëntie

Taking turns MAC protocol

Polling protocol

De Master node 'invites' de slaven nodes om om beurt te sturen.



Voordelen

- Geen collisies
- Geen lege sloten

Nadelen

- Polling delay
- Als de master node fails, dan zal heel het kanaal in-operatief worden.

Token passing

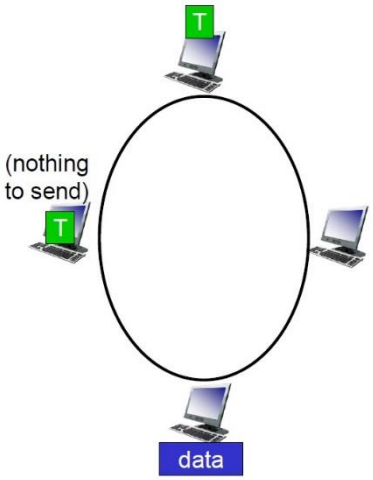
Een special-purpose frame, tokten, wordt uitgewisseld tussen verschillende nodes.

Voordelen

- Geen collisies
- Geen lege sloten
- Gedecntraliseerd
- Heel efficiënt

Nadelen

- Als een node de token vergeet vrij te geven, dan moet er een recovery procedure opgestart worden.
- Als de token fails, dan zal heel het kanaal in-operatief worden.



DOCSIS:

Link laag protocol voor Cable Internet Acces

DOCSIS specifieert de kabel data netwerk architectuur en zijn protocollen.

Downstream: FDM

Upstream: TDM

Tijd intervallen die nog eens onderverdeeld zijn in mini-time slots.

Via een MAP msg op de downstream kanaal, die verstuurd werd door de CMTS, worden de cable modems verwittigd in welke mini-time slots ze mogen verzenden.

De CMTS weet wanneer modems data wil verzenden door het gebruik van mini-slot-request frames.
Deze worden verzonden in een 'random acces manner' dus hier kan er wel botsingen optreden.
Wanneer een collisie is waargenomen (doordat de modem geen respond krijgt) zal de modem binary exponential backoff gebruiken om opnieuw de mini-slot-request frame te sturen.

Wat is een MAC adres en wat is zijn functie

Een MAC adres is een 48 bit hexadecimaal (met elke nummer 4 bits) adres die 'gebrand' is in de NIC ROM, maar kan ook software matig ge-set worden.

De functie van het MAC/LAN/physical/Ethernet adres is om een frame van de ene interface naar de ander fysisch geconnecteerde interface te krijgen. Dus als een adapter ene frame krijgt met hetzelfde MAC adres zal hij het datagram eruit halen, anders 'discards' hij de frame gewoon.

Wat is ARP en waarvoor wordt het gebruikt

Hoe kunnen we het MAC adres van een interface te weten komen als we enkel zijn IP adres weten?
Daarvoor wordt ARP gebruikt, om een IP adres 'om te zetten' naar een MAC adres.

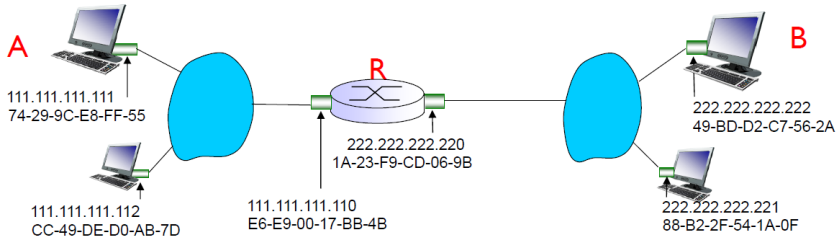
Elke node heeft een ARP table, met daarin IP/MAC adres mappings voor sommige LAN nodes.

<IP address; MAC address; TTL>

Zelfde LAN

Broadcast ARP query pakket versturen, met B's IP adres. B ontvant het ARP pakket en stuurt een reply naar A met zijn MAC adres (Unicast).
A caches IP-to-MAC adres in zijn ARP table.
ARP is plug-and-play, nodes creëren zelf hun ATP tables.

Andere LAN



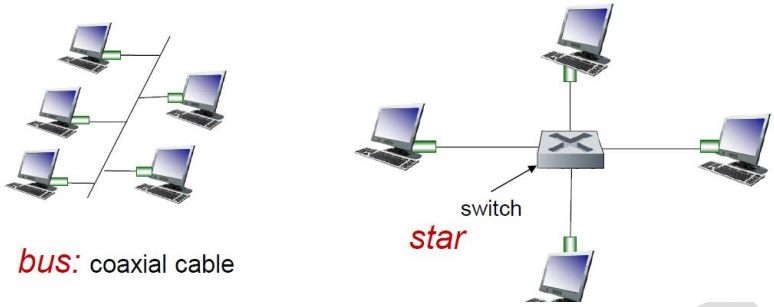
A:
IP datagram met IP source A en destination IP adres B
Link layer frame met R's MAC adres als destination, en A's MAC adres als source.

R :
Frame doorgegeven naar de netwerk laag. De router zal nu de juiste interface kiezen via zijn forwarding table.
Creëren van link-layer frame met B's MAC adres (via ARP) als destination adres.

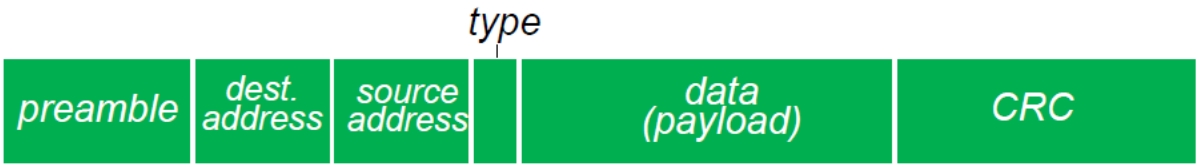
Wat is Ethernet, hoe ziet zijn frame structuur en standaards eruit

Ethernet was de eerste 'widley deployed high-speed' LAN en was bovendien simpel, goedkoop en snel.

Physical topology



Frame structure



Preamble:
7 bytes met het patroon 10101010 gevolgd door één byte met een patroon 10101011.
Dit gebeurt zodat de ontvanger kan synchroniseren met de zenders clock snelheid.

Type:
Het type van het hoger liggende protocol. (meestal IP)

CRC:
Error gedetecteerd is frame droppen.

Unreliable, connectionless

Geen handshaking tussen NICs, ontvangen NIC stuurt geen acks of nacks.
Data in gedropped frames enkelgerecoverd als TCP gebruikt werd.
Ethernets's MAC protocol is unslotted CSMA/CD met binary backoff.

Standards