Netwerken

Paul Cobbaut

Netwerken

Paul Cobbaut

lt-1.0

Published Thu 01 Aug 2013 01:03:38 CEST

Abstract

This book is meant to be used in an instructor-led training. For self-study, the intent is to read this book next to a working Linux computer so you can immediately do every subject, practicing each command.

This book is aimed at novice Linux system administrators (and might be interesting and useful for home users that want to know a bit more about their Linux system). However, this book is not meant as an introduction to Linux desktop applications like text editors, browsers, mail clients, multimedia or office applications.

More information and free .pdf available at **http://linux-training.be** .

Feel free to contact the author:

- Paul Cobbaut: paul.cobbaut@gmail.com, http://www.linkedin.com/in/cobbaut

Contributors to the Linux Training project are:

- Serge van Ginderachter: serge@ginsys.eu, build scripts; infrastructure setup; minor stuff
- Hendrik De Vloed: hendrik.devloed@ugent.be, buildheader.pl script

We'd also like to thank our reviewers:

- Paul Cobbaut: paul.cobbaut@gmail.com, linux-training.be

Table of Contents

List of Tables

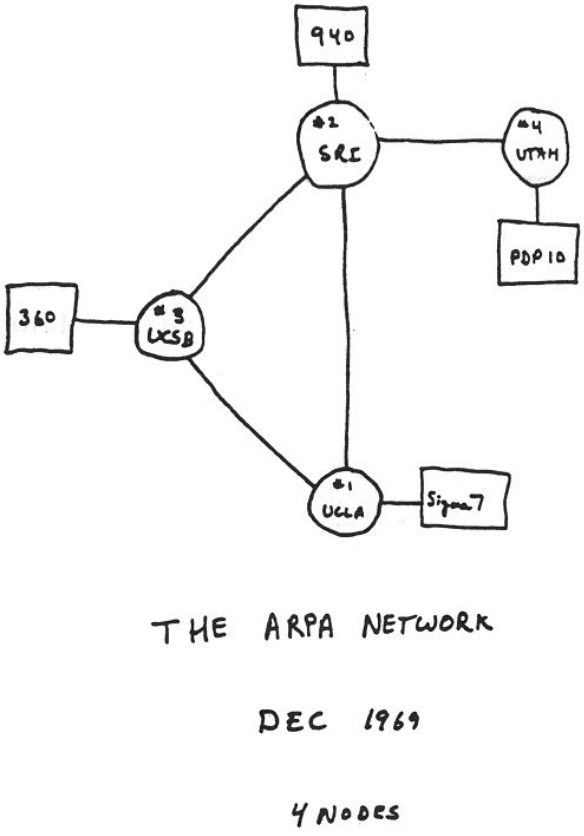
Chapter 1. geschiedenis van het internet

Table of Contents

Het internet is meer dan **de blauwe e** op de Windows desktop. Het **internet** bestaat sinds 1969 en omvat behalve het **World Wide Web** ook diensten zoals **smtp (e-mail)**, **nntp (usenet/nieuwsgroepen)** en de laatste jaren vooral **torrents** voor het downloaden van vele gigabytes en terabytes.

De geschiedenis van het internet is veel ruimer dan besproken in deze module.

1.1. 1969 arpanet



Het **arpanet** was het eerste netwerk van computers dat gebruik maakte van **packet switching** technologie. Eind 1969 waren vier computers verbonden op dit **internetwork**.

Het **arpanet** groeide aan tot honderden verbonden computers in 1983 (toen werd het opgesplitst in het publieke arpanet en het militaire **milnet**).

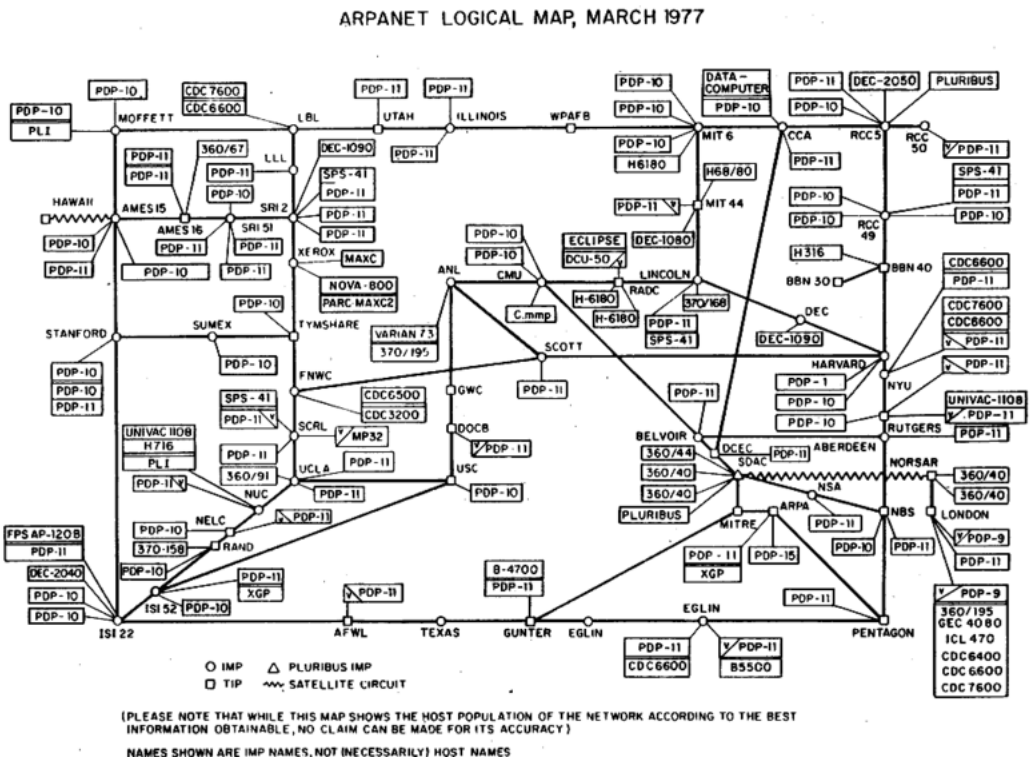
1.2. packet switching

Ter vervanging van vast toegewezen communicatiekanalen werd in de jaren zestig een nieuwe techniek ontwikkeld die alle gegevens die over een netwerk moeten getransporteerd worden opdeelt in kleine **packetjes**. Elk packetje bepaalt onafhankelijk van alle andere packetjes zijn route naar zijn bestemming. Dit laat toe dat meerdere computers tegelijkertijd gebruik maken van dezelfde (fysische) communicatiekanalen.

Beknopte uitleg op de Nederlandstalige wikipedia pagina, uitgebreide uitleg in het Engels.

```
http://nl.wikipedia.org/wiki/Packet_switching
http://en.wikipedia.org/wiki/Packet_switching
```

1.3. jaren zeventig



De jaren zeventig zagen ook de creatie van het **X.25** netwerk, **fidonet**, **compuserve**, **bulletin board systems** en **uucp**. Al deze netwerken zijn vandaag volledig opgegaan in het **internet** en **tcp/ip**.

1.4. tcp/ip

De jaren zeventig waren ook de start voor het **tcp/ip** protocol. Deze protocol stack zullen we later uitgebreid bespreken.

Vandaag gebruikt 99 procent van het internet **ipv4** (Internet Protocol versie 4). De opvolger **ipv6** is klaar, maar breekt nauwelijks door.

1.5. ncp en osi

Op 1 januari 1983, ook gekend als **flag day**, werd het **ncp** protocol op internet vervangen door tcp/ip. Van dan af zal tcp/ip, een protocol-stack waarvan de ontwikkeling begon in 1969, het internet domineren. Dit ondanks pogingen eind jaren 80 om het nieuwe **osi protocol** als standaard door te voeren.

Terzijde ook even vermelden dat in de jaren 80 zowat elke grote computerfirma begon met een eigen protocol ; IBM met **SNA**, Novell met **IPX/SPX**, Microsoft met **NetBEUI**, Apple met **Appletalk** en Banyan met **Vines**. Maar al deze protocols werden van de kaart geveegd door het oudere en open **tcp/ip**.

1.6. e-mail

Reeds in 1965 bestond er **e-mail** op mainframe computers. Het **@ teken** dateert uit 1971. In de jaren zeventig kon je e-mailen tussen verschillende netwerken zoals arpanet, vnet(ibm), bitnet, nsfnet en alle computers die uucp of ncp spraken.

Sinds 1982 bestaat e-mail als **SMTP (Simple Mail Transfer Protocol)**, gedefineerd oorspronkelijk in **rfc 821** (later meermaals uitgebreid).

```
http://nl.wikipedia.org/wiki/SMTP
http://en.wikipedia.org/wiki/SMTP
http://www.rfc-editor.org/rfc/rfc821.txt
http://www.rfc-editor.org/rfc/rfc5321.txt
```

1.7. rfc

Sinds 1969 maakt men voor elk internet protocol een **rfc (Request for Comments)**. Een **rfc** is een (ascii) tekstbestand en bepaalt de werking van alle protocols op het internet. Deze tekstbestanden zijn vrij te raadplegen en te implementeren. Enkele voorbeelden van protocols die bepaald zijn door een **rfc** zijn tcp, ip, ftp, udp, dns, smtp, ntp, dhcp en http.

Op **www.rfc-editor.org** vind je alle rfc's. Je kan ook rechtstreeks naar de tekstversie gaan als je de rfc nummer kent (zoals 2131 voor dhcp) via :

```
http://www.rfc-editor.org/rfc/rfc2131.txt
```

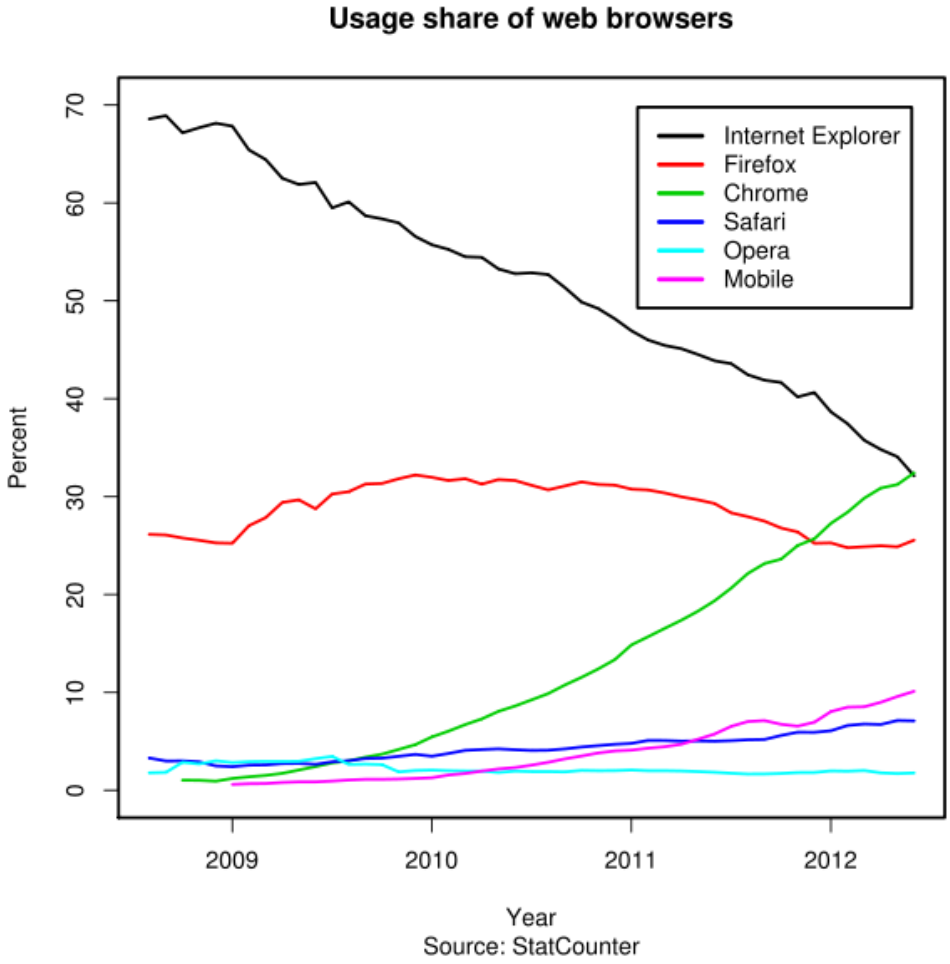
Elke **rfc** bevat o.a. een datum, opgepast voor humor als je **1 april** ziet.

1.8. www

Tim Berners Lee vond dat **gopher** en **ftp** niet volstonden, dus kwam hij in 1989 op de proppen met het **World Wide Web**. Toch was het nog wachten tot de release van eerst de **mosaic** browser en nadien **netscape navigator** alvorens dit **www** populair werd. In de jaren 1996-1997 surft 90 procent van de internetbevolking met **Netscape Navigator** op het world wide web.

In 1995 kwam **Spyglass, Inc** met de **internet explorer** browser op de markt. Deze browser werkte op **Windows 95** (zat in het **Plus pack**) en werd nadien gebundeld met **Windows 98**. Tussen 2001 en 2005 surft meer dan 90 procent van de internetbevolking met de **blauwe e** van **Internet Explorer 6** op het world wide web.

Na jaren van **Netscape** dominantie en jaren van **Internet Explorer 6** dominantie is de browser wereld nu verdeeld. De cijfers voor 2012 volgens **NetApplications**, **W3C** en **StatCounter** geven allemaal een verdeeld beeld.



(Grafiek CC Attribution 3.0 Unported Daniel Cardenas, Lifehacker)

Altijd opletten met statistieken!

1.9. piraterij

Vanaf eind jaren 90 brak de **piraterij** definitief door op het internet. Eerst populair via **napster**, een peer-to-peer programma om duizenden **mp3-tjes** te downloaden via een simpele muisklik. Groot verschil met het (jarenlang gedoogde) copieren van cassettebandjes was dat er geen kwaliteitsverlies was bij het nemen van een kopie van een kopie (van een kopie...).

Napster werd opgedoekt, maar er kwamen snel alternatieven zoals kazaa, emule, gnutella, limewire en in 2002 verscheen **suprnova.org**. Deze laatste werd immens populair.

Het (uiteindelijk) neerhalen van de suprnova.org website werd als een overwinning gezien voor Hollywood, ware het niet dat kort nadien **thepiratebay** verscheen.

Hieronder een foto (copyright dekaminski.se) van de allereerste piratebay server.

1.10. sociale netwerken

Sinds de opkomst van **myspace**, **orkut**, **linkedin** en zeker **facebook** is het al **web 2.0** wat de klok slaat. We noemen deze nieuwe trend van **vriendjes maken** in het Engels **social networking**, soms ongelukkig vertaald in het Nederlands als **sociale netwerken**.

De strijd om de gebruiker, die gewonnen leek door **Microsoft** met hun bijna monopolistische aandeel op de eindgebruikers' desktop, speelt zich nu vooral online af. **Google**, **Microsoft**, **Yahoo** en vele anderen proberen nu ook met **social networking** technieken het succes van **Facebook** te bereiken.

Nieuwe spelers op dit terrein zijn **twitter** en sinds dit jaar ook **Mega** (van Kim Dotcom) met plannen om ook email, chat, voice, video, mobile en vooral ook bitcoin te aanvaarden.

1.11. bitcoin

Sinds enkele jaren dwalen er **bitcoins** rond op het internet. Dit is een betaalmiddel dat vooral door **grijze websites** gerbuikt wordt, maar wel bezig is aan een serieuze opmars. Banken en regeringen vrezen nu al de grootte van facebook (1 miljard gebruikers) en Google/Blogger/Youtube (800 miljoen gebruikers). Wat als deze ook nog eens een betaalmiddel (en dus een economie) hebben die groter is dan het BNP van pakweg de G7 ?

1.12. Wie bestuurt het internet ?

1.12.1. IETF

De **Internet Engineering Task Force** is een organisatie die **rfc**'s kan goedkeuren als zijnde een **internet standard protocol**.

1.12.2. ICANN

De **Internet Corporation for Assigned Names and Numbers** is een Amerikaanse organisatie die beslist over domeinnamen (DNS) en controle heeft over **IANA**. De voorganger van **ICANN** was **InterNIC**.

1.12.3. RIR

Elke **Regional Internet Registry** krijgt ip-adressen van IANA en verdeelt die over een regio. Er zijn vijf van deze RIR in de wereld:

```
ARIN : USA + Canada
RIPE : Europa + USSR + Midden-Oosten
AFRINIC : Afrika
LACNIC : Latijns Amerika
APNIC : Oceanië + Azië (zonder USSR/Midden-Oosten)
```

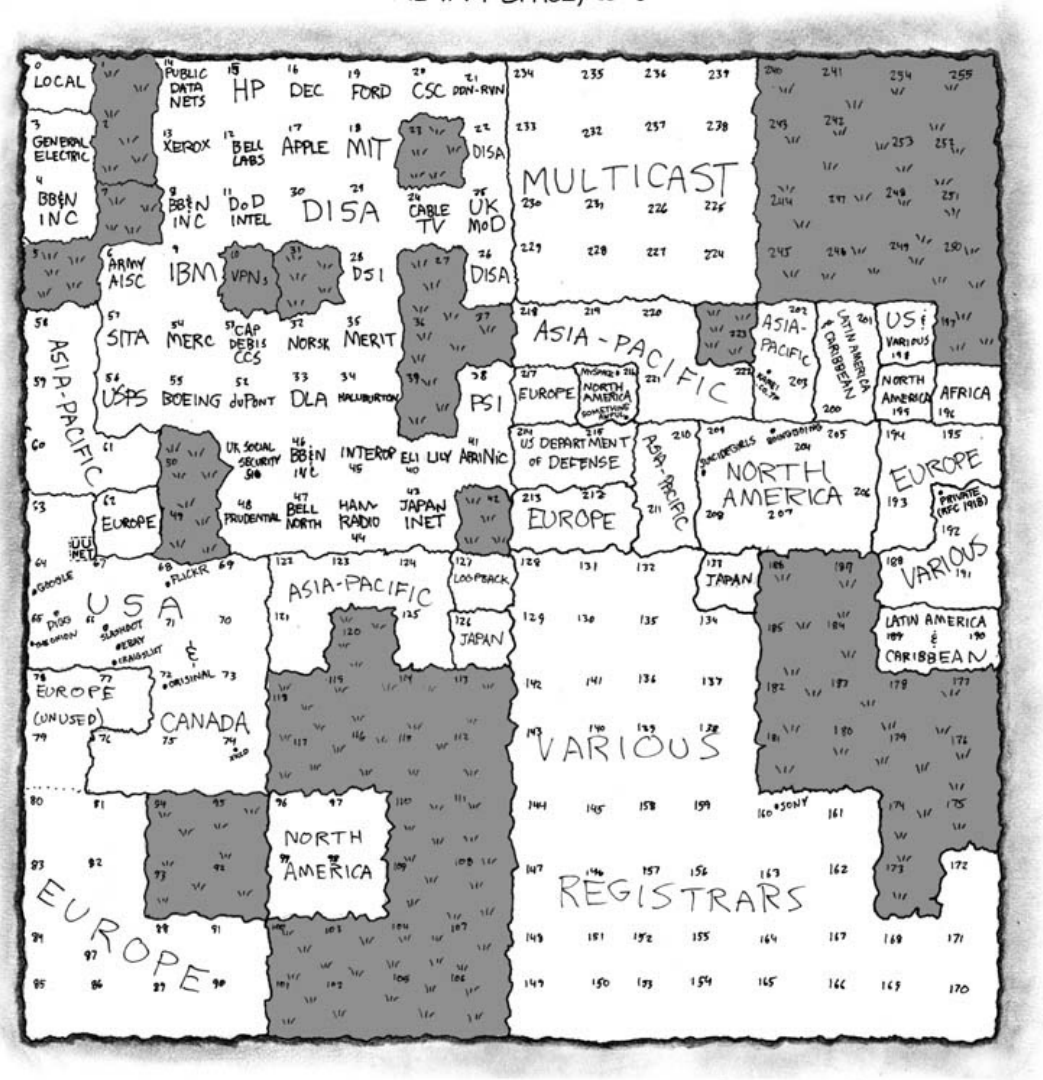
1.12.4. IANA

De **Internet Assigned Numbers Authority** beslist wereldwijd over wie welke ip-adressen krijgt (via de RIR's), beheert de DNS root servers en controleert MIME-types.

Hieronder een kaartje van de ip-adressen zoals ze verdeeld zijn. Dit is getekend door XKCD (een cartoonist).

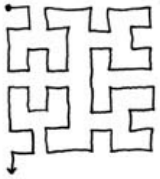
MAP OF THE INTERNET

THE IPv4 SPACE, 2006



THIS CHART SHOWS THE IP ADDRESS SPACE ON A PLANE USING A FRACTAL MAPPING WHICH PRESERVES GROUPING -- ANY CONSECUTIVE STRING OF IPs WILL TRANSLATE TO A SINGLE COMPACT, CONTIGUOUS REGION ON THE MAP. EACH OF THE 256 NUMBERED BLOCKS REPRESENTS ONE /8 SUBNET (CONTAINING ALL IPs THAT START WITH THAT NUMBER). THE UPPER LEFT SECTION SHOWS THE BLOCKS SOLD DIRECTLY TO CORPORATIONS AND GOVERNMENTS IN THE 1990's BEFORE THE RIRs TOOK OVER ALLOCATION.

0 1 14 15 16 19 →
3 2 13 12 17 18
4 7 8 11
5 6 9 10



= UNALLOCATED BLOCK

1.13. mensen

1.13.1. Vint Cerf

Vinton Gray Cerf werkte van 1976 tot 1982 bij DARPA aan de ontwikkeling van tcp/ip. Sinds 2005 werkt hij voor Google. Hij is medeoprichter van ICANN en de uitvinder van de EOF byte en de **ascii** tabel. Hij is ook auteur van heel wat **rfc**'s, onder andere rfc2468. Samen met **Kahn, Kleinrock en Roberts** vormt hij de vier vaders van het internet.

1.13.2. Jon Postel

Jon Postel is de man die in 1969 gestart was met de **rfc**'s. Hij beheerde ook tot zijn dood in 1998 de IANA.

Jon Postel is ook de man achter de uitspraak "Wees conservatief in wat je zegt, en liberaal in wat je aanvaardt".

```
http://tools.ietf.org/html/rfc2468
http://www.postel.org/postel.html
```

1.13.3. Tim Berners Lee

Sir Tim Berners Lee is de uitvinder van het **www** (samen met de onbekende Belg **Robert Cailliau**).

```
http://en.wikipedia.org/wiki/Tim_Berners_Lee
http://en.wikipedia.org/wiki/Robert_Cailliau
```

1.13.4. Al Gore

Ondanks de misplaatste grappen rond zijn uitspraak "I invented the internet" tijdens de presidentsverkiezingen van 2000 hoort **Al Gore** toch thuis in dit rijtje omdat hij tijdens de jaren zeventig, tachtig en begentig de grote drijfveer was achter de commercialisatie van het internet.

Chapter 2. terminologie

Table of Contents

Termen zoals **bit**, **byte**, **broadcast** en meer worden uitgelegd in dit hoofdstuk.

2.1. bit

Een **bit** is de kleinste eenheid van informatie in de informaticawereld. Een **bit** kan twee waarden aannemen : nul of een. Hieronder een tabel met alle mogelijk waarden van een **bit** en de manier waarop deze worden voorgesteld.

Table 2.1. de waarden van een bit

waarde	voorstelling
nul	0
een	1

2.2. byte

Een **byte** is acht **bits** wordt wel eens gezegd. Maar zo eenvoudig is het niet. Toen **Werner Buchholz** de term in 1956 voor het eerst gebruikte bedoelde hij letterlijk "een reeks bits die de computer kan verwerken" (of bijten). Heden is een **byte** altijd gelijk aan **acht bits**, maar in de jaren zeventig waren er computers met **bytes** van 4 tot 36 bits (de PDP 10 uit 1966 had 36-bit bytes).

Een **byte** wordt ook wel een **octet** genoemd. De volgende tabel toont termen die wel eens gebruikt worden in relatie tot een byte.

Table 2.2. de byte en zijn verwanten

naam	aantal bits
nibble	4
byte	8
word (x86)	16
word (meestal)	8, 16 of 32
word (andere)	9, 12, 18, 24, 36, 39, 40, 48 of 60
dword (x86)	32
long word (68k)	32

De term **dword** staat voor **double word** en is op de **Intel** architectuur gelijk aan **32 bits**. Dit ondanks dat de fysische **word**-grootte nu **32 bits** of zelfs **64 bits** is.

Het **long word** was eind jaren 80 **32 bits** lang op de **Apple** en **Commodore Amiga** computers met **68k** processor.

Wat deze cursus betreft is een **byte** steeds **8 bits**. Verder spreken we niet over een **word** of zijn varianten, maar zullen we steeds het aantal **bits** aanduiden.

2.3. kilobyte

2.3.1. 1000 of 1024

Hoeveel bytes is een **kilobyte** ? Sinds de beginjaren van de informatica was het antwoord hierop eenduidig **1024 bytes**, toch als we praten over harde schijven en computer geheugen. (In de telecommunicatie en netwerkterminologie was een kilo gelijk aan 1000. Zo is de kilo in **kilobits per seconde** gelijk aan 1000.)

Maar eind jaren negentig beginnen enkele makers van harde schijven veelvouden van 1000 te gebruiken i.p.v. 1024. Als voorbeeld drie harde schijven in mijn desktop computer. De eerste is een 60GB Seagate, de tweede een 80GB Maxtor en de derde een 200GB Western Digital. In veelvouden van 1000 geeft dit:

```
ST: 60.0 GB, 60022480896 bytes
Maxtor: 81.9 GB, 81964302336 bytes
WDC: 200.0 GB, 200049647616 bytes
```

In veelvouden van 1024 kom je op respectievelijk 56GB, 76GB en 186GB.

De reden om 1000 te gebruiken i.p.v. 1024 is simpele marketing: hoe groter de harde schijven, hoe groter het verschil!

Table 2.3. veelvouden van 1000 of 1024

1000	1024	verschil
1000	1024	2,4 procent
1000000	1048576	4,9 procent
1000000000	1073741824	7,4 procent
1000000000000	1099511627776	10 procent

2.3.2. kibibyte

Er kwam een einde aan de verwarring (not!) toen de **IEC** in 1999 (met update in 2000 en tweede update in 2005) begon met een standaard die in 2008 samen met de **IEEE**, de **ISO** en het **SI** het volgende besliste: kilo = 1000 en kibi = 1024, behalve voor RAM geheugen en cache geheugen (en SD-kaartjes....).

In de praktijk wordt nog steeds gesproken over **1024 bytes** in elke **kilobyte**, tenzij je een harde schijf koopt.

2.3.3. megabytes en meer

Een **megabyte** is dus 1000 maal 1000 bytes, een **gigabyte** is 1000 **megabyte**, en een **terabyte** is 1000 **gigabyte**. Toch als we de nieuwe standaard volgen. In de prakrijk wordt nog wel eens 1024 gebruikt i.p.v. 1000.

Het rijtje gaat verder zoals je in deze tabellen kan zien.

Table 2.4. megabytes en meer

aantal bytes	juiste term
1000	kilobyte
1000 kilo	megabyte
1000 mega	gigabyte
1000 giga	terabyte
1000 tera	petabyte
1000 peta	exabyte
1000 exa	zettabyte
1000 zetta	yottabyte

Table 2.5. mebibytes en meer

aantal bytes	juiste term
1024	kibibyte
1024 kibi	mebibyte
1024 mebi	gibibyte
1024 gibi	tebibyte
1024 tebi	pebibyte
1024 pebi	exbibyte
1024 exbi	zebibyte
1024 zebi	yobibyte

Wikipedia heeft een uitgebreid overzicht van de kilo-1000 versus kilo-1024 geschiedenis en de geboorte van **kibibytes** en **zebibytes**.

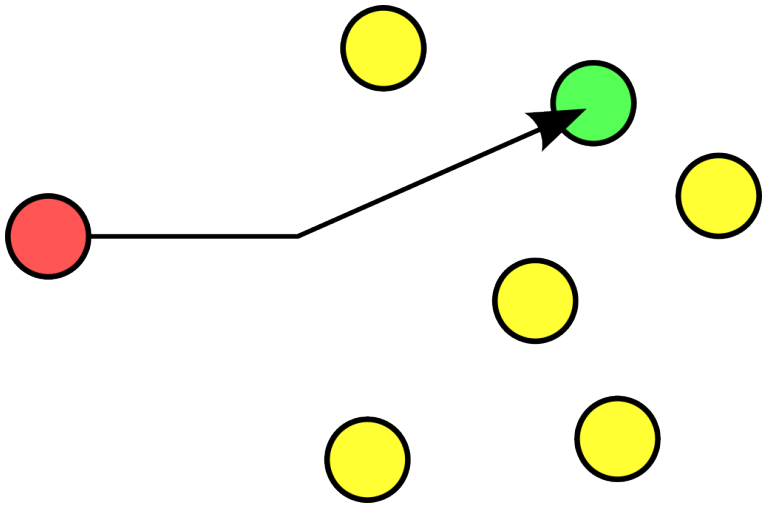
`http://en.wikipedia.org/wiki/Binary_prefix`

2.4. zender en ontvangers

Alle vier deze termen worden regelmatig gebruikt in datacommunicatie en netwerken om aan te duiden welk bereik een zender heeft. (Met dank aan een anonieme weldoener om deze tekeningen gratis en onvoorwaardelijk in de **public domain** los te laten.)

2.4.1. unicast

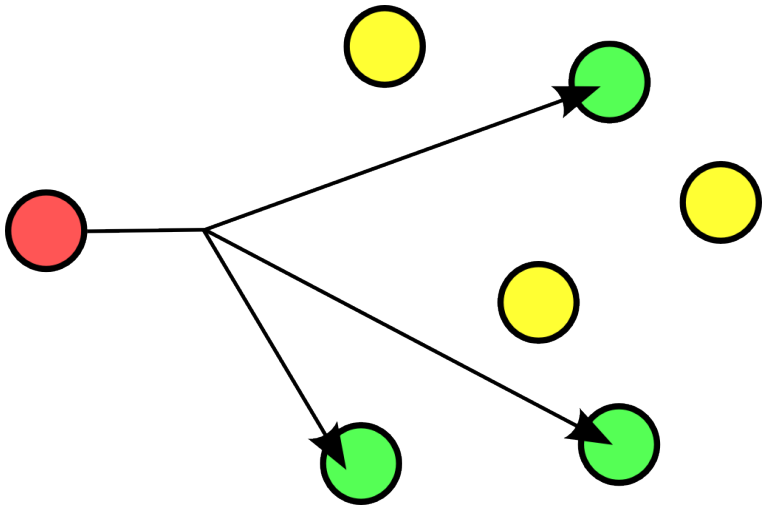
Gaat een signaal naar een eenduidige unieke bestemming, dan is dit een **unicast**. Heel wat datacommunicatie bestaat uit **unicast** berichten tussen twee computers.



Voorbeelden van een unicast zijn frames op laag 2 naar een specifiek mac-adres en packets op laag 3 naar een ip-adres (dat niet het broadcast adres is).

2.4.2. multicast

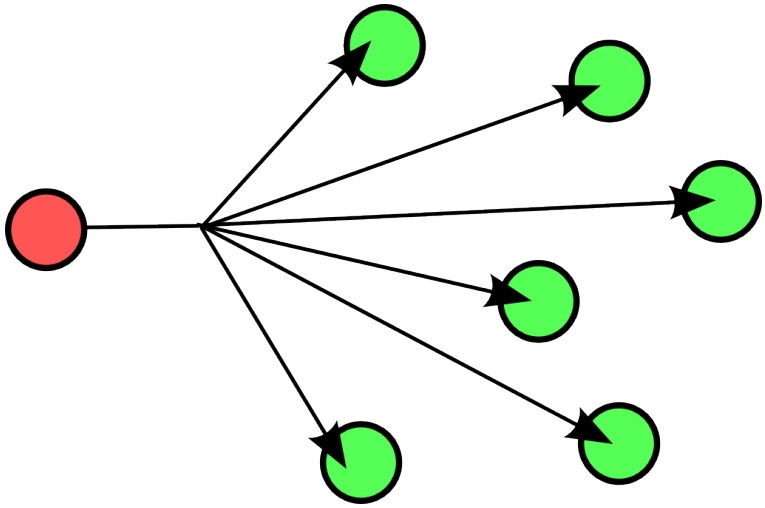
Is het signaal bedoeld voor leden van een welbepaalde groep, dan spreken we over een **multicast**.



Voorbeelden van multicast zijn Realplayer (voor .sdp bestanden) en RIPv2 (zie later bij routers).

2.4.3. broadcast

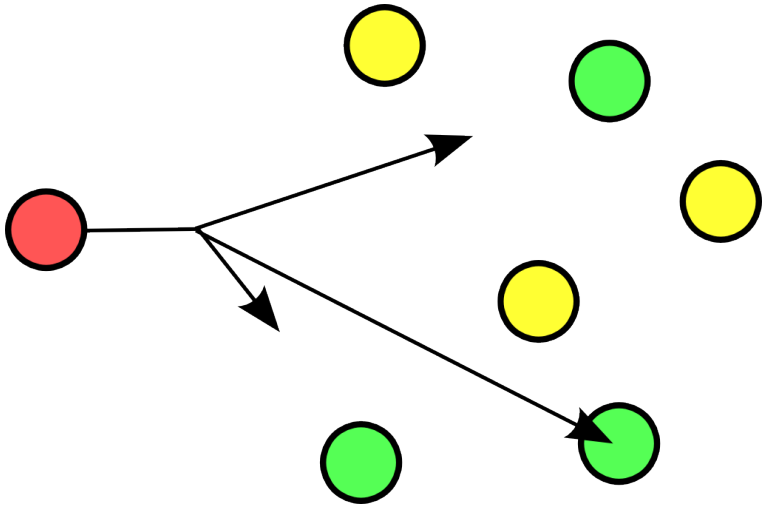
Wanneer een signaal gestuurd wordt naar iedereen heet dit **broadcast**. Let wel op dat je duidelijk bepaalt wie 'iedereen' is. Als we de klas even beschouwen als 'iedereen', dan is een bericht dat de hele klas bereikt een **broadcast** in de klas. Als we echter alle klassen in de school beschouwen als 'iedereen', dan is een bericht dat de hele klas bereikt, maar niet de hele school, geen **broadcast**.



Typisch voorbeeld hier is de BBC (British Broadcasting Corporation) die uitzendt over de hele wereld. In datacommunicatie is broadcast meestal beperkt tot de LAN.

2.4.4. anycast

Nieuw tegenwoordig, en in de praktijk gebruikt door de **root name servers** van internet is de **anycast**. Een **anycast** is een signaal naar de dichtstbijzijnde van een groep.

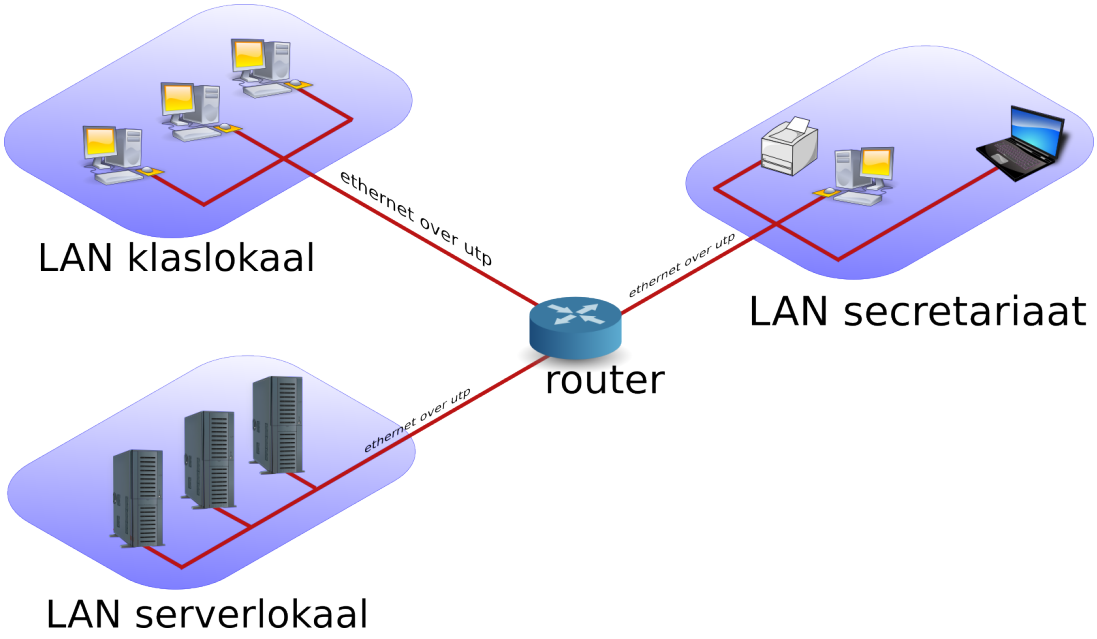


2.5. lan-wan-man

2.5.1. lan

Een **lan** (Local Area Network) is een lokaal netwerk. Dit kan een klaslokaal zijn, of een verdieping, of zelfs een gebouw. Men spreekt over een **lan** omdat de computers dicht bij elkaar staan. Heden wordt een **lan** meestal bepaald doordat de computers verbonden zijn met **ethernet**.

Een **lan** kan bestaan uit meerdere kleine **lan**'s. In deze tekening zie je een voorbeeld van een netwerk van een school, bestaande uit drie **lan**'s.



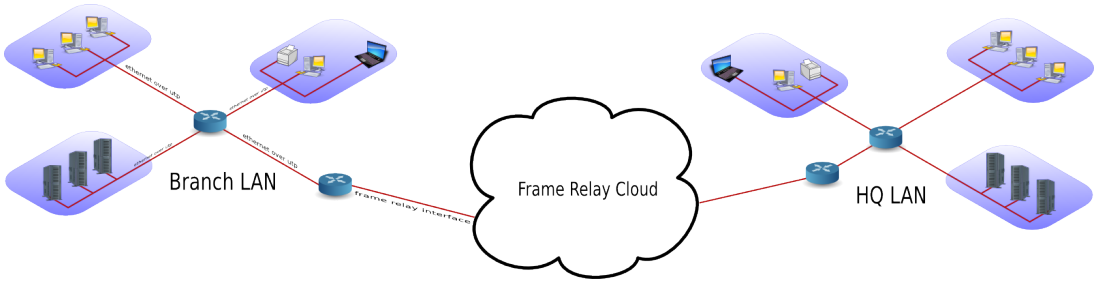
2.5.2. man

Een **man** (Metropolitan Area Network) is een term die soms gebruikt wordt voor een netwerk dat meerdere gebouwen in dezelfde agglomeratie omvat.

2.5.3. wan

Een **wan** (Wide Area Network) is een netwerk waar de computers ver uit elkaar staan. Deze computers zijn dikwijls verbonden via **gehuurde lijnen**. Een **wan** is dus het omgekeerde van een **lan**. Een **wan** gebruikt geen **ethernet** maar andere protocols zoals **FDDI**, **Frame Relay**, **ATM** of **X.25** om computers te verbinden die enkele of enkele duizenden kilometers van elkaar verwijderd zijn.

Hieronder een voorstelling van een bijkantoor dat via **Frame Relay** verbonden is met het hoofdkwartier.



De term **wan** wordt ook gebruikt voor netwerken die een groot oppervlak bestrijken, zoals het **internet**, ook al gebruiken deze netwerken **tcp/ip**.

Cisco is vooral gekend voor **wan** technologie, zij maken **routers** die onder andere **lan** netwerken verbinden via **wan** protocols.

2.5.4. pan-can-wpan

Heel soms wordt er gesproken over een **pan** (Personal Area Network) of een **can** (Campus Area Network). Een draadloze **pan** is een **wpan**.

2.6. topologie van een netwerk

Meer info en mooie tekeningen op Wikipedia:

```
http://nl.wikipedia.org/wiki/Netwerktopologie
http://en.wikipedia.org/wiki/Network_topology
```

2.6.1. punt tot punt

De eenvoudigste topologie van een netwerk is een **point-to-point** verbinding tussen twee toestellen. Bijvoorbeeld een telefoonverbinding op het oude telefoonnetwerk (**circuit switched**). Deze verbinding is meestal dynamisch, maar kan ook permanent zijn, zoals de rode telefoon.

We spreken ook van een **punt-tot-punt** verbinding als je twee computers verbindt met een **seriële** of een **parallelle** kabel, of als je de netwerkkaarten rechtstreeks verbindt. Ook een klassieke **modem** verbinding tussen twee computers is een **punt-tot-punt** verbinding.

Niet te verwarren met **peer-to-peer** waar een applicatie zorgt voor overzicht tussen de **peers**.

2.6.2. bus

In een **bus** netwerk is elke computer verbonden met dezelfde kabel (meestal een coax kabel). Aan de twee eindes van de **bus** zit dan een **terminator**. Met **T-stukjes** kan je dan computers bij op de kabel plaatsen. Er kan slechts één computer tegelijk gebruik maken van de **bus**. Als twee computers tegelijk zenden, dan botsen de signalen tegen elkaar. Deze botsing heet een **collision**.

2.6.3. ster

Van zodra we een centraal toestel hebben zoals een **hub**, **switch** of **MAU** spreken we van een **ster** netwerk. Elke computer heeft een eigen kabel naar dit centrale toestel.

In theorie kan je de **point-to-point** topologie beschouwen als een ster met één arm.

2.6.4. ring

Een **ring** topologie bestaat uit toestellen die elk met twee andere toestellen verbonden zijn, en wel zo dat ze een ring vormen. Gegevens gaan van de ene **node** naar de volgende, en zo de hele ring rond. Een probleem met één verbinding kan de hele ring platleggen.

Typisch voorbeeld hier is een **Token Ring** netwerk. Deze technologie van IBM bestond fysisch uit een **ster** topologie met centrale **MAU** (Multistation Access Unit)

en speciale **token ring** kabels naar elke computer. Logisch gezien vormden de computers een ring. Telkens was er één computer die het **token** had, deze computer kon zenden op de **ring**.

2.6.5. tree

Een netwerk kan ook hierarchisch georganiseerd zijn, zoals een boomstructuur. We spreken dan van een **tree** met een **root node** aan de top van de boomstructuur. Type voorbeeld is de logische structuur van **DNS** servers op internet.

2.6.6. mesh

Elke computer in een **mesh** kan met vele (of alle) andere computers verbonden zijn.

Een **mesh** netwerk bestaat dus uit computers die allen dienst kunnen doen als **router**.

2.6.7. hybride

Elke combinatie van bovenstaande architecturen die niet onder één van deze valt, noemt men een **hybride** netwerk.

2.7. telefoon en data

2.7.1. pstn

In tegenstelling tot het internet is het originele **pstn** (Public Switched Telephone Network) een **circuit-switched** netwerk. Elk telefoongesprek eiste een nagenoeg fysieke verbinding tussen de twee sprekers.

Dit oude analoge telefoonnetwerk, ook wel **pots** (Plain Old Telephone Service) genoemd, werd gebouwd met simpele **twisted pair** koperdraden om de menselijke stem over te brengen. Dit netwerk draagt golven tussen 300 en 3400 **Hertz**, wat overeenkomt met een **bandbreedte** van 3100 Hertz.

Omdat dit netwerk (geluids)golven rechtstreeks als communicatie gebruikt, noemt men dit een analoog netwerk.

2.7.2. twisted pair ?

Deze oude **twisted pair** is een paar koperdraden waar af en toe een draai in zit. Niet te verwarren met moderne **stp** of **utp** kabels die heel wat meer **twists** bevatten.

2.7.3. modem

Een **modem** is een toestel dat een digitaal apparaat, bijvoorbeeld een computer, verbindt met het analoge telefoonnetwerk. Aan de andere kant staat eveneens een modem die het geluid op het telefoonnetwerk weer omzet in (digitale) bytes. Op die manier kunnen twee computers verbonden worden.

Tegenwoordig zijn er ook **modems** die niet met geluid werken, zoals **adsl-modems** en **kabelmodems**. Later volgt er een les die meer uitlegt geeft over allerhande **modems**.

2.7.4. ppp

Het **Point-to-Point Protocol** zit in **laag 2** van het **OSI Model** en kan twee computers met elkaar verbinden via het telefoonnetwerk of via een seriële kabel. **ppp** werd veel gebruikt voor **dial up** toegang tot internet.

Het **ppp** protocol is ontworpen om behalve met **ip** ook te werken met **ipx/spx** en **appletalk**. Meer info op Wikipedia of **rfc-editor.org**.

`http://www.rfc-editor.org/rfc/rfc1661.txt`

Voor **ppp** was er **slip** om **ip** packetjes in te pakken zodat ze over een seriële kabel of modem konden verstuurd worden. De meeste **Unix** computers ondersteunen dit nog steeds.

`http://www.rfc-editor.org/rfc/rfc1055.txt`

2.7.5. isdn

Het **isdn** of **Integrated Services Digital Network** is recenter dan **pots** en is digitaal. Oorspronkelijk was **isdn** enkel te verkrijgen in stedelijke gebieden, de dag van vandaag is het (voor thuisgebruikers) overbodig gemaakt door **(a)dsl** en internet via de **kabel**.

Een isdn **bearer(B)** kanaal heeft een bandbreedte van **64kb/s**, en wordt met de letter B aangegeven. De meest courante isdn connecties werden genoteerd als **2B+D**, wat betekent dat er twee **bearer** kanalen van **64kb/s** voorzien zijn, samen met een **16kB/s data(D)**. Deze **16kB/s** link wordt ook **bri** of **Basic Rate Interface** genoemd.

Table 2.6. isdn in kb en kB

isdn type	bits/s	Bytes/s	kb/s	kB/s
bearer B	64000	8000	64	8
2B	128000	16000	128	16
D kanaal	16000	2000	16	2
2B+D	144000	18000	144	18

2.7.6. kb of kB ?

Noteer even dat **kb** hier de afkorting is van **kilobits** en **kB** staat voor **kiloBytes**. In beide gevallen staat de **k** of **kilo** voor 1000, dat was vroeger al zo, en is volgens de nieuwe regels nog steeds zo.

Als je **b(bits)** en **B(bytes)** verwarring wil vermijden, gebruik dan steeds **kilobits** of **kilobytes** i.p.v. van **kb** of **kB**.

2.7.7. T1

Een **T1** is een **23B+D isdn** verbinding in de Verenigde Staten. En ook al spreekt men van een **T1** in de rest van de wereld, eigenlijk gebruikt men dan een **E1** ofte **30B+D**. Een **T1/E1** noemt men ook een **PRI (Primary Rate Interface)**.

2.7.8. T3

Een **T3** lijn is een **gehuurde lijn** met een bandbreedte van **44.736 Mbit** per seconde. Dit type verbinding wordt meestal gebruikt tussen providers en telefoonmaatschappijen onderling.

2.8. oefening

Als oefening rekenen we even de totale bandbreedte van deze verbindingen.

Hoe lang duurt het om een bestand van 1 kilobyte, 1 megabyte, 1 gigabyte en 1 gibibyte te kopiëren langs een 9.6 kilobits/s modem, een 64 kilobits/s isdn modem, een T1, een E1, een 10Mbit LAN, een 100Mbit LAN en een 1gigabit LAN ?

Table 2.7. weinig data over netwerken

medium	1 KB	1 MB	1 GB	1 GiB
9.6 modem	0,83"	13'53"	9d15u28'	10d8u33'
isdn modem	0,13"	2'5"	34u43'20"	37u16'58"
T1		5,43"	1u30'35"	1u37'16"
E1		4,17"	1u09'27"	1u14'34"
10Mbit LAN		0,8"	13'20"	14'19"
100Mbit LAN			1'20"	1'25"
1Gbit LAN			8"	8,59"

Idem voor een terabyte, een exabyte en een zettabyte (veelvouden van duizend dus).

Table 2.8. veel data over netwerken

medium	1 TB	1 EB	1 ZB
9.6 modem	26 jaar 155 dagen	26,4 miljoen jaar	26,4 miljard jaar
isdn modem	3jaar 351 dagen	3,9 miljoen jaar	3,9 miljard jaar
T1	62 dagen 21 uur	172 duizend jaar	172 miljoen jaar
E1	48 dagen 5 uur	132 duizend jaar	132 miljoen jaar
10Mbit LAN	9d6u13'	25 duizend jaar	25 miljoen jaar
100Mbit LAN	22u13'20"	2536 jaar	2,536 miljoen jaar
1Gbit LAN	2u13'20"	238j8m5d	253 duizend jaar

Chapter 3. netwerklagen

Table of Contents

Elke cursus over netwerken bespreekt het **OSI-model** en het **DoD-model**, deze cursus is daarop geen uitzondering.

3.1. OSI model

In 1977 richtte de **ISO** (International Organization for Standardization) het **OSI** (Open Systems Interconnection) op. Het **OSI** ontwierp een zeven-lagen model om de werking van computernetwerken te beschrijven. De zeven lagen zijn (van 7 naar 1) Application, Presentation, Session, Transport, Network, Data Link en Physical.

Er is heden geen populair protocol dat de zeven lagen ook effectief correct implementeert. In de praktijk worden vooral de lagen twee en drie vernoemd om toestellen duidelijk te onderscheiden.

Table 3.1. OSI model

laag	protocols	opmerking
7. Application	dns http dhcp smtp putty	applicatieprotocols
6. Presentation	mime aes 3des ascii tls ssl	dataformaten, compressie, versleuteling
5. Session	nfs rpc smb OS sshv2	sessie dialoog naamgeving
4. Transport	tcp udp	fragmentatie
3. Network	ip icmp ipsec rip	ip-adres, router
2. Data Link	ethernet atm ppp stp	mac-adres, bridge, switch
1. Physical	nic wlan glasvezel fddi adsl	kabels, hub, netwerkkaart

We bespreken elke laag (layer) even apart.

3.1.1. layer 7: application

De **application layer** beschrijft applicaties. Op het internet vandaag betekent dat protocols zoals dns, http, ntp, smtp, snmp, ssh, ftp, X, pop3, imap, irc, telnet en tftp. Dit is de laag die de gebruiker 'ziet' in de vorm van toepassingen.

3.1.2. layer 6: presentation

De **presentation layer** beschrijft data-formaten. De presentatielaag zorgt bij de ontvanger dat de data wordt samengesteld in een formaat dat de applicatie begrijpt.

Het **MIME** protocol (rfc2046) dat toelaat om bestandsformaten (die meekomen met e-mail of websites) te identificeren hoort ook in deze laag. Dankzij **MIME** komt je applicatie te weten of de **attachment** een **jpeg** foto of een **mp3** geluidje is.

3.1.3. layer 5: session

De **session layer** beschrijft sessies tussen **hosts**. Als je een website bezoekt, dan heeft jouw computer een **sessie** met de webserver.

Een **host** op internet kan omschreven worden als elk toestel dat een **ip-adres** heeft, maar zelf geen **router** is.

3.1.4. layer 4: transport

De **transport layer** beschrijft het knip- en plakwerk om data te segmenteren in segmenten (of **datagrams**). Typisch vandaag vind je hier de **tcp** en **udp** protocollen.

Deze laag zorgt dat elk stukje data genummerd is, zodat de zender de in stukjes geknipte data mooi terug kan opbouwen.

3.1.5. layer 3: network

Met de **network layer** komen we in een boeiende materie. Het **IP** protocol nestelt zich in deze laag en zal in de cursus uitgebreid besproken worden. We zullen ook **IpSec**, **icmp** en **igmp** kort bespreken en demonstreren.

Deze laag is verantwoordelijk voor de adressering van individuele packetjes, en zorgt ervoor dat packetjes van de bron tot aan de bestemming komen (bijvoorbeeld van je laptop naar een webserver).

3.1.6. layer 2: data link

Ook de **data link layer** gaat regelmatig onze aandacht opeisen. Deze laag zorgt voor de communicatie tussen computers (of nodes) die rechtstreeks met elkaar verbonden zijn (zoals de computers in deze klas, of nodes aan beide uiteinden van een T1 leased line).

We bespreken in de les later het **arp** protocol van deze laag. We zullen zien dat **arp** de verbinding is tussen laag 3 (ip) en laag 2(mac).

In deze laag vinden we ook het **mac-adres** van onze **ethernet** netwerkkaart. Het **ethernet** protocol zelf zit ook in deze laag, en zorgt voor de link naar laag 1.

Ook **isdn**, **ppp**, **slip**, **fddi**, **frame relay** en **ATM** vinden we hier terug.

3.1.7. layer 1: physical

De **physical layer** beschrijft stekkers en elektromagnetische signalen van een fysieke verbinding tussen nodes.

In deze laag vinden we bijvoorbeeld **ethernet**, **wi-fi**, **twisted pair** en **modems**, adsl, isdn, T1, gsm, bluetooth, firewire, usb en irda.

Sommige protocols komen voor in meer dan 1 laag, omdat ze functionaliteiten hebben die op meerdere lagen van toepassing zijn. Zo beschrijft onder andere het **ATM** protocol zowel laag 1 als laag 2.

3.2. OSI-model versus DoD

Het zeven-lagen model van het **OSI** is verdrongen door het vier-lagen model van **tcp/ip**. Deze protocol stack is vooral in de jaren zeventig ontwikkeld door **DARPA**(Defense Advanced Research Projects Agency), een onderdeel van het Amerikaanse Department of Defense (DoD). Vandaar dat het **tcp/ip** model ook wel het **DoD model** wordt genoemd.

De vier lagen van dit model zijn : Application (laag 5-6-7 osi), Transport (ongeveer laag 4), Internet (bovenkant osi laag 3) en Link Layer (onderkant osi laag 3 en ook osi laag 2 en 1). Op het internet zijn heel wat grafiekjes te vinden die beide lagen naast elkaar zetten, sommige doen dit correcter dan anderen.

Hieronder geen vergelijking, wel een beknopt overzicht van enkele typische tcp/ip protocols naast hun desbetreffende laag in het tcp/ip model.

Layer	Protocols
Application	dns http dhcp smtp ssh ftp
Host-to-Host(Transport)	tcp udp
Internet	ip icmp igmp ipsec rip ospf bgp
Network Access(Link)	tokenring ethernet ppp slip stp atm

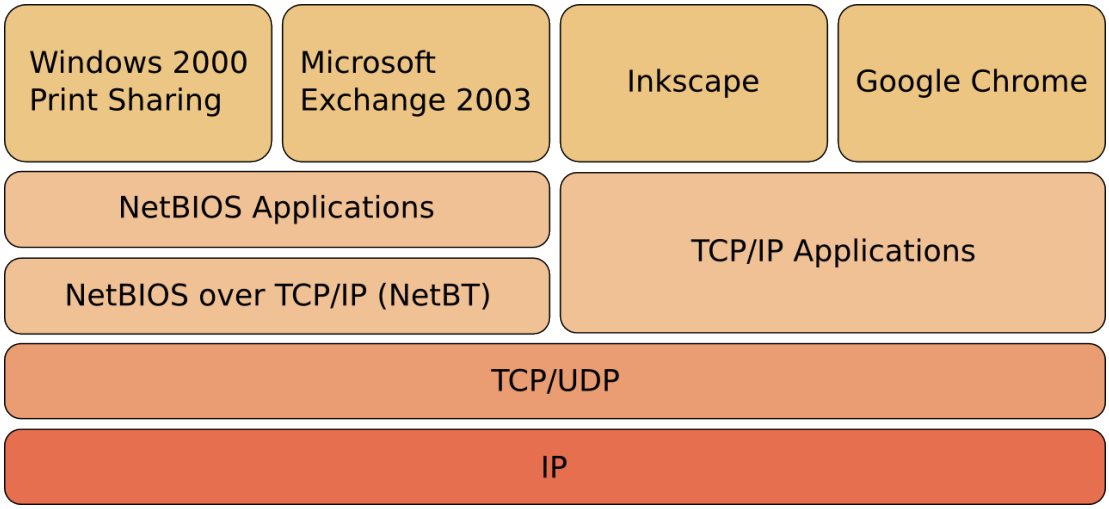
3.3. NetBIOS versus DoD

NetBIOS(sessie), **NetBEUI**(namen) en **NetBIOS frames protocol**(netwerk en transport) zijn een collectie van protocols die vooral tussen 1990 en 2008 door Microsoft producten (o.a. alle Windows File/Print sharing, Exchange Server 2003, ...) werden gebruikt. Heel wat organisaties hebben vandaag nog steeds NetBIOS applicaties op hun netwerk.

Zonder al te veel in detail te gaan, toch even vermelden dat NetBIOS geen 'routable' protocol is, je kan er dus geen erg grote netwerken mee bouwen omdat heel wat zaken enkel via **broadcast** werken.

Applicaties die vandaag nog NetBIOS gebruiken worden opgevangen door **NetBT** wat languit als **NetBIOS over TCP/IP** wordt uitgesproken.

Hieronder een schema dat dit duidelijk maakt, we kunnen later eventueel uitgebreider terugkomen op **NetBT** en daaraan gerelateerde services zoals **WINS** en **node types**.



In zo een netwerk volstond het om een computer een unieke naam te geven op het netwerk om die computer te verbinden met dat netwerk. Geen enkele computer kon dezelfde naam hebben als een andere.

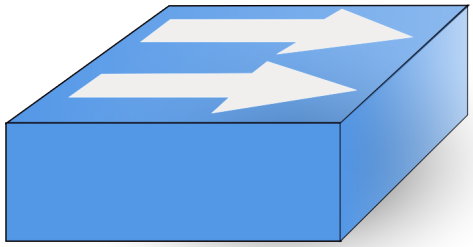
3.4. toestellen en lagen

In een netwerk vinden we heel wat toestellen (devices). We zetten ze even laag per laag op een rijtje.

3.4.1. physical

Toestellen in laag 1 zijn in principe onzichtbaar op ons netwerk. We kunnen ze niet allemaal eenvoudig met software detecteren. We vinden hier onder andere de **netwerkkaart**, de **repeater**, de **hub** en de **modem**.

Hieronder de voorstelling van een **hub** in een netwerkdiagram.



De **netwerkkaart** of **nic (Network Interface Card)** of ook **network adapter** is tegenwoordig een vast onderdeel van elke computer. De netwerkkaart ontvangt elektrische signalen en zet deze om in bytes (en omgekeerd). De netwerkkaart controleert ook of een packetje van het netwerk wel voor haar is of niet. Indien niet, dan gooit de netwerkkaart dit packetje weg (**drop**).

De meeste netwerkkaarten vandaag zijn **ethernet** kaarten met een 48-bit uniek **mac-adres**. Enkele jaren terug waren ook **Token Ring** kaarten courant in gebruik.

Een **repeater** is een versterker. Elektrische signalen verzwakken, dus als de netwerkkabel te lang is, dan kan een repeater het signaal versterken.

Een **hub** wordt ook wel een **multiport repeater** genoemd. Net zoals een **repeater** zal ook een hub het elektrische signaal versterken. Daar waar een repeater tussen twee kabels zit, zit een hub tussen meerdere kabels, en versterkt hij het signaal naar alle verbonden kabels.

Hubs kunnen **actief** of **passief** zijn. Een passieve hub versterkt het elektrische signaal, en logischerwijs ook de ruis op het kanaal. Te veel passieve hubs achter elkaar zorgt voor een te grote versterking van de ruis, waardoor het signaal verloren kan gaan.

Een **actieve hub** is slimmer dan een **passieve hub**. In plaats van het signaal gewoon te versterken, leest een actieve hub de bits van een signaal, en stuurt deze bits naar alle verbonden kanalen. Op deze manier treedt er geen ruisversterking op.

Een **modem** (**modulator/demodulator**) is een toestel dat (in het geval van een **pots-modem**) bits omzet in geluid, en omgekeerd aan de andere kant van de lijn dit geluid weer omzet in dezelfde bits.

Een **kabel modem** zoals die van Telenet of Brutele vertaalt bits naar frequenties die oorspronkelijk bedoeld waren om TV-signalen door te sturen. Ook **ADSL** modems gebruiken een frequentie die buiten de menselijke stem liggen.

3.4.2. data link

In deze laag vinden we toestellen zoals een **bridge** en een **switch**. In dat laatste geval uiteraard een **layer 2 switch**.

Hieronder de voorstelling van een **switch** in een netwerkdiagram.

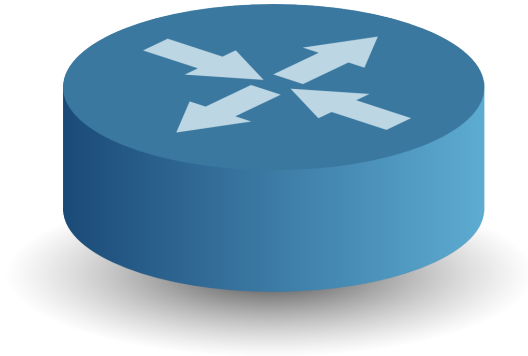
Een **bridge** en **switch** zijn beide slimmer dan een hub of repeater, in de zin dat deze laatste niet weten wat een mac-adres is. Een bridge en een switch interpreteren het mac-adres om te weten of (in het geval van de bridge) en waar (in het geval van de switch) ze het packetje doorsturen.

Een echte bridge is zeldzaam vandaag (want een switch kan alles wat een bridge kan).

3.4.3. network

De **router** en de **layer 3 switch** zijn typische layer 3 toestellen. Toestellen in deze laag weten wat een ip-adres is, en kunnen uiteraard ook alles wat **layer 2 devices** kunnen.

Hieronder de voorstelling van een **router** in een netwerkdiagram.



Een echte router verbindt netwerken (bijvoorbeeld ethernet met T1, of ADSL met WiFi), maar de term wordt ook gebruikt voor ethernet-ethernet toestellen.

Een layer 3 switch kent bijvoorbeeld **icmp** en wordt ook gebruikt om sniffen van het netwerk te verhinderen.

3.4.4. hogere lagen

Een **gateway** is een toestel dat protocols vertaalt, bijvoorbeeld **NetBIOS** naar **tcp/ip**.

3.5. lagen oefening

0. Kies een netwerk voor de volgende vragen : thuis, werk, klas

1. Maak een tekening (papier of pc) van je netwerk voor laag 1.

2. Maak een tekening (papier of pc) van je netwerk voor laag 2.

3. Maak een tekening (papier of pc) van je netwerk voor laag 3 (wanneer je een website bezoekt).

4. Maak een tekening (papier of pc) van je netwerk voor laag 7 (wanneer je een website bezoekt).

Chapter 4. sniffer

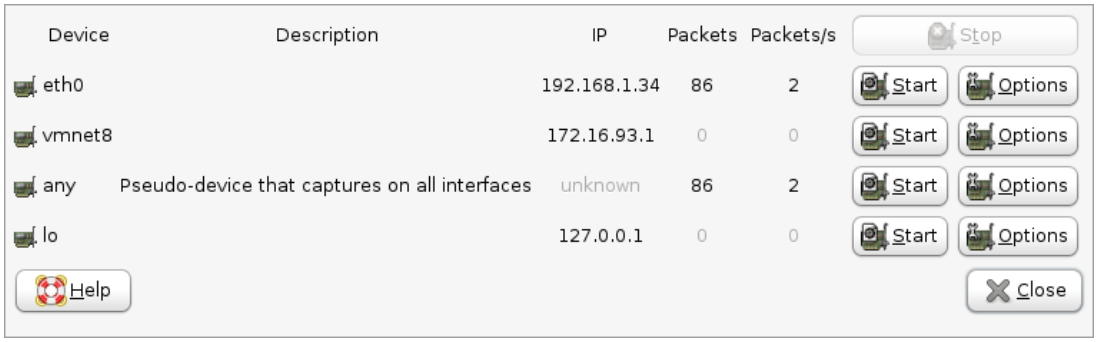
Table of Contents

Op het netwerk, op de bekabeling, reizen vele packetjes (hopelijk) naar hun bestemming. Als je een **sniffer** gebruikt, dan kan je al deze packetjes zien (en bewaren). Een **sniffer** is dus een software die toelaat dat je alles ziet wat er gebeurt op het netwerk.

De gekendste en meest gebruikte **sniffer** heet **wireshark** (vroeger **ethereal**). We zullen deze **sniffer** in de klas gebruiken om meer inzicht te krijgen in een netwerk.

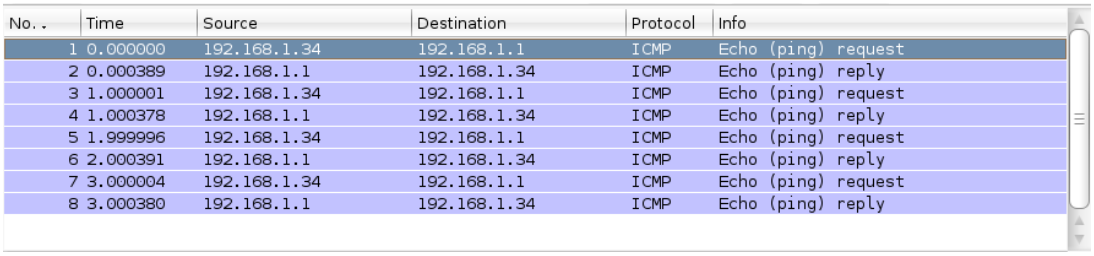
4.1. interface selectie

Als je de eerste keer **wireshark** start, dan moet je een interface selecteren.



4.2. begin te snuffelen

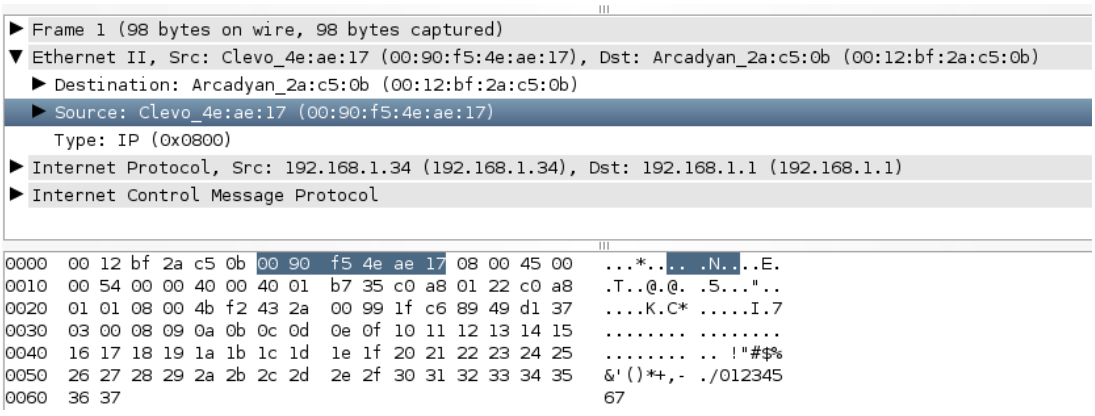
In dit voorbeeld hebben we een **ping** besnuffeld tussen twee computers. Het bovenste paneel van **wireshark** toont je dat het **icmp** protocol wordt herkend.



No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.34	192.168.1.1	ICMP	Echo (ping) request
2	0.000389	192.168.1.1	192.168.1.34	ICMP	Echo (ping) reply
3	1.000001	192.168.1.34	192.168.1.1	ICMP	Echo (ping) request
4	1.000378	192.168.1.1	192.168.1.34	ICMP	Echo (ping) reply
5	1.999996	192.168.1.34	192.168.1.1	ICMP	Echo (ping) request
6	2.000391	192.168.1.1	192.168.1.34	ICMP	Echo (ping) reply
7	3.000004	192.168.1.34	192.168.1.1	ICMP	Echo (ping) request
8	3.000380	192.168.1.1	192.168.1.34	ICMP	Echo (ping) reply

4.3. in de packetjes kijken

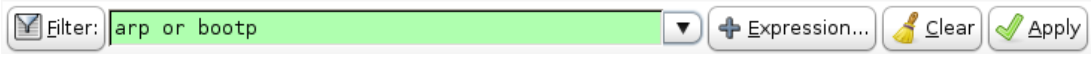
In het middelste paneel kan je de verschillende tcp/ip lagen open klikken. Als je in het middelste paneel een lijn selecteert, dan zie je de positie van deze data in het packetje in het onderste paneel.



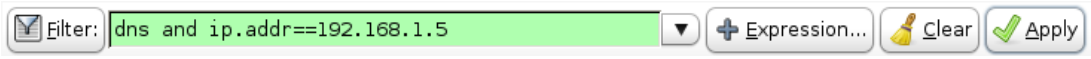
4.4. filters

Soms verlies je de weg in de vele packetjes. Een eenvoudigde oplossing is om **filters** te gebruiken, zo zie je enkel de packetjes die je interesseren. Als je **arp** intikt en dan op **apply** klikt, dan zie je enkel **arp** packetjes.

Je kan twee protocols combineren met een logische **of** door het Engelse **or** ertussen te zetten. Het voorbeeld hieronder toont hoe je **arp** en **bootp** (of **dhcp**) packetjes kan bekijken.



En hier een voorbeeld van **dns** verkeer voor een welbepaald **ip-adres**.



!!!! Vraag toelating alvorens op een bedrijfsnetwerk een sniffer te gebruiken !!!!

4.5. oefenen met de sniffer

1. Installeer wireshark op de computer (google voor wireshark).

2. Open via "Start - cmd" een **command prompt** op Windows, of open een **shell** op Unix. Laat deze open staan, we zullen er later commando's in uitvoeren.

3. Noteer het ip-adres van jouw computer en van je buurman. Gebruik hiervoor op Windows het commando **ipconfig** in een command prompt. Op Unix, Linux, MacOSX kan je **ifconfig** gebruiken.

4. Start wireshark, en start in wireshark een capture.

5. Open een website en controleer dat je vanalles ziet in wireshark (roep anders de leraar). Zo weet je zeker dat wireshark werkt. Je moet nog niet begrijpen wat je allemaal ziet, we bespreken dat later stap voor stap.

6. Werk even samen met je buurman voor deze vraag. Start op beide computers een nieuwe capture. Deze oefening is makkelijker als je geen browser (internet explorer of firefox) hebt open staan (want dit zorgt soms voor veel packetjes in wireshark). Gebruik het commando **ping ip-adres-buurman** op 1 van de computers (niet op beide!), naar de andere computer.

7. Welke packetjes heb je in wireshark gezien door het uitvoeren van die ping ? Normaal gezien moet er twee **arp** packetjes zijn voor de ping (icmp packetjes).

8. Doe een ping naar een onbestaand ip-adres (bv 192.168.0.109). Wat zie je nu in wireshark ?

9. Je kan in wireshark een filter instellen om niet te veel packetjes te zien, bijvoorbeeld **arp or icmp**. Je kan elke lijn in wireshark selecteren en dan onderaan meer details zien. Kan je het ip-adres of het mac-adres vinden in de gesnuffelde packetjes ?

10. Wat zie je als je het commando **arp** (of arp -a) uitvoert ?

11. Doe een ping naar google.com. Wat zie je nu in de uitvoer van **arp** ?

12. Open nu een eenvoudige website waar je een account hebt (niet facebook/hotmail/gmail, maar een simpel hobby-ding of leerplatform ofzo). Zorg dat je uitgelogd bent. Start een nieuwe capture en log dan in op die website. Stop de capture en bestudeer de boeiende informatie die je nu hebt zoals :

Naam van de website :

Besturingssysteem van de webserver :

Besturingssysteem van jouw pc :

MAC adres van jouw pc :

Webserver software :

Browser op jouw pc :

ip-adres van de webserver :

ip-adres van jouw pc :

De poort gebruikt om de webserver te bereiken :

De poort gebruikt om terug naar je client te gaan :

Jou gebruikersnaam :

Jou paswoord :

13. Waarom zie je het MAC-adres van de webserver niet ?

14. Welk ander MAC-adres zie je dan wel ?

Iedereen die met een sniffer tussen jouw pc en de webserver zit kan dezelfde informatie zien. De mensen die de webserver beheren, beschikken ook over deze informatie...

15. Probeer het paswoord te sniffen van een gmail of hotmail account. Waarom lukt dit niet ?

16. Kopieer een tekstbestand van de ene computer naar de andere. Kan je dit bestand ook lezen via de sniffer ?

17. Kan je ook tekstbestanden lezen die andere computers in de klas naar elkaar sturen ? Waarom wel of waarom niet ?

Chapter 5. voorbeeld netwerklagen

Table of Contents

Dit hoofdstuk beschrijft zeer beknopt de weg die onze data bewandelt als we in een **browser** zoals **firefox** een webpagina van de VRT (van teletekst) bekijken. Je kan dit hoofdstuk bekijken als een beknopt overzicht van de hele cursus, alles wat we in de cursus bekijken is wel ergens in dit verhaal te plaatsen.

5.1. onze data op reis

5.1.1. laag 7: applicatie

De reis begint in de bovenste laag. Dat is laag 7, de applicatielaag. We gebruiken als voorbeeld de applicatie genaamd **firefox** waarop pagina 525 van vrt teletekst open staat. Uiteraard bevindt de applicatie **firefox** zich in de bovenste laag.

We klikten op het knopje **toon** om de pagina te verversen. Klikken op dat knopje stuurt een **http request** naar de vrt teletekst **webserver**. Het **layer 7 protocol** waar webservers en web clients (aka browsers) mee werken is **http (hyper text transfer protocol)**. Onze data zakt naar laag 6.

5.1.2. lagen 6 en 5: presentatie en sessie

Deze lagen zijn niet echt van toepassing omdat we met **tcp/ip** werken en niet met een 7-lagig osi protocol. Je zou kunnen zeggen dat de **browser** in laag 6 een **html** document ontvangt van de **webserver**. Dit **html** document kan bestaan uit **ascii** of **unicode** karakters.

5.1.3. laag 4: transport

In deze laag zien we hoe de **tcp** van **tcp/ip** een **sessie** opzet.

Zoals eerder al geschreven vindt er in deze laag wat knip- en plakwerk plaats. Alleen is het nu net een geval waar er niet moet geknipt worden. We vragen een **webpagina** aan een **webserver**, en deze vraag past perfect in een packetje, dus er moet niet geknipt worden. Het protocol aan het werk hier is **tcp (Transfer Control Protocol)**.

Maar er gebeurt toch nog iets meer dan enkel de vraag sturen van **firefox** naar de vrt teletekst **webserver**. Het **tcp** protocol gaat eerst een **tcp sessie** opzetten met de vrt webserver. We kunnen dit controleren door een **sniffer** op te starten.

Protocol	Info
TCP	34614 > http [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=10991739 TSER=0 WS=6
TCP	http > 34614 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1452 TSV=1835689667 TSER=10991739 WS=0
TCP	34614 > http [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSV=10991742 TSER=1835689667
HTTP	GET /tt_een.php?p=511&g=0&s=1 HTTP/1.1

Je ziet drie **tcp packetjes** voordat het **http** packetje (wat ook een tcp packetje is) wordt verstuurd. **tcp** zet een sessie op door eerst een packetje te sturen met de vraag om een sessie op te zetten (SYN), hierop komt een antwoord van de VRT webserver (SYN,ACK) en als laatste wordt er vanuit mijn laptop nog een (ACK) gestuurd.

In het screenshot hierboven zie je een **http GET** van pagina 511 i.p.v. pagina 525, maar dat heeft geen belang voor de theorie.

5.1.4. laag 3: netwerk

Onze vraag voor een webpagina moet terecht komen bij de **webserver** van de vrt. Deze computer staat niet in onze klas (en ook niet bij mij thuis). Het **ip** protocol zorgt voor de bezorging van het packetje ter plaatse in de Reyerslaan (waar deze vrt webserver staat).

Om op de correcte plaats te geraken, is er een **ip-adres** nodig. Het ip-adres van de vrt webserver is **193.191.175.137**. We vinden dit terug in elke packetje dat we naar de webserver van de vrt sturen, in de ruimte voorzien voor **destination IP-address**. Als bron ip-adres wordt telkens het ip-adres van mijn laptop thuis gebruikt, in dit geval **192.168.1.34**.

Onze sniffer vertaalt de **hexadecimale code** die de computer gebruikt naar menselijk leesbare **decimale cijfers**.

Source	Destination	Protocol	Info
192.168.1.34	193.191.175.137	TCP	34614 > htt
193.191.175.137	192.168.1.34	TCP	http > 3461
192.168.1.34	193.191.175.137	TCP	34614 > htt
192.168.1.34	193.191.175.137	HTTP	GET /tt_een

Hoe kan onze computer het **ip-adres** van de webserver van vrt teletekst kennen ? Daarvoor doet **tcp/ip** een beroep op **dns**. Indien je voor het eerst sinds een tijdje naar de website van teletekst gaat op **url http://teletekst.een.be**, zal er een **dns-query** gaan van onze computer naar onze lokale **DNS server**.

DNS (Domain Name System) (zit niet in laag 3!) wordt later uitgebreid besproken, maar het zou kunnen dat je het volgende ziet in de sniffer.

```
192.168.1.34    192.168.1.1     DNS     Standard query A teletekst.een.be
192.168.1.1     192.168.1.34    DNS     Standard query response CNAME tt.vrt.be A 193.191.175.137
```

5.1.5. laag 2: data link

De **OSI data link layer** is een deel van de **DoD link layer** van **tcp/ip**. In deze laag maken we gebruik van het **MAC** adres van een netwerkkaart. Een **mac adres** is een fysisch adres dat ingebrand is in de netwerkkaart.

Voordat ons eerste packetje kan vertrekken, moet het **mac-adres** van de lokale bestemming gevonden worden. De webserver van vrt teletekst staat niet bij mij thuis,

dus de lokale bestemming is mijn **router** (aka de adsl modem). Deze heeft **ip-adres** 192.168.1.1.

Mijn laptop zal een **arp** broadcast uitvoeren om het mac-adres te vinden van de computer met ip-adres 192.168.1.1. Mijn **router** zal hierop antwoorden. In een sniffer ziet dit er als volgt uit.

```
ARP      Who has 192.168.1.1?  Tell 192.168.1.34
ARP      192.168.1.1 is at 00:02:cf:aa:68:f0
```

5.1.6. laag 1: physical

De netwerkkaart is gelukkig als het packetje volledig is, en smijt het letterlijk op mijn lokaal netwerk. Fysisch is dit een broadcast, alle computers in het lokale netwerk ontvangen dit packetje, maar enkel de enige correcte bestemmeling zal het packetje ook aanvaarden. Die correcte bestemming is mijn lokale **router**.

In de OSI wereld is dit een aparte laag, in de DoD wereld is dit een onderdeel van de **DoD link layer**.

5.2. onze data onderweg

Onze **http request** voor een teletekst pagina is nu onderweg van centrum Antwerpen naar de Reyerslaan in Brussel. Het packetje springt van router tot router, totdat het op het netwerk van de **webserver** van de vrt zit.

Je kan deze weg (de routers) bekijken door een **traceroute** te doen. Hieronder zie je de uitvoer van het **traceroute** commando op mijn laptop.

```
traceroute to 193.191.175.137 (193.191.175.137), 30 hops max, 60 byte packets
1  illyria (192.168.1.1)  0.812 ms  1.285 ms  1.764 ms
2  213.219.148.1.adsl.dyn.edpnet.net (213.219.148.1) 26.453 ms 26.833 ms 27.216 ms
3  ndl2-rb01.edpnet.be (213.219.132.237)  27.655 ms  28.035 ms  28.412 ms
4  adslgwbe.edpnet.net (212.71.1.78)  28.924 ms  29.305 ms  29.798 ms
5  10ge.cr1.brueve.belnet.net (194.53.172.65)  30.165 ms  30.543 ms  30.888 ms
6  10ge.ar1.brucam.belnet.net (193.191.16.193)  31.991 ms  31.530 ms  31.960 ms
7  vrt.customer.brussels.belnet.net (193.191.4.189) 32.046 ms 9.004 ms 24.205 ms
8  * * *
9  * * *
```

Wat je ziet, is dat de eerste router **illyria** heet en **192.168.1.1** als ip-adres heeft. Dit is mijn lokale **adsl modem** met ingebouwde router functie. Op de tweede lijn zie je de andere kant van mijn adsl modem, de kant van mijn **internet service provider (ISP)** genaamd **edpnet**.

Daarna passeren we enkele routers van edpnet, om vervolgens te reizen naar **belnet**. Het Belgische **belnet** is een snel backbone netwerk waar o.a. universiteiten en openbare instellingen op aangesloten zijn.

De laatste router die we zien heet **vrt.customer.brussels.belnet.net**. Uit deze naam kan je afleiden dat de VRT een klant is van **belnet** in Brussel.

Nadien zien we enkel nog sterretjes, dat is omdat de **systeembeheerder** van de VRT beslist heeft om geen **traceroute** door zijn **firewall** te laten. Onze **http request** mag gelukkig wel door.

5.3. bestemming bereikt

Als alles goed gaat, dan bereikt onze data (of onze **http request**) zijn bestemming. De laatste **router** gooit onze data op het lokale netwerk van de **webserver** van de VRT.

De netwerkkaart van deze server converteert de elektrische signalen in bytes, en controleert of de eerste zes bytes wel overeenstemmen met haar **MAC adres**. Dat is gelukkig juist, dus is het de beurt aan **IP**.

Het **ip protocol** controleert of het **destination ip address** wel correct is, en dat is (gelukkig) weeral juist.

Verdere **metadata** in het packetje vertelt deze computer dat het een **tcp packet** is, met als bestemming de **http-server**.

De **http server (of webserver)** zelf zoekt dan de gevraagde webpagina, en stuurt deze webpagina als antwoord op onze vraag.

5.4. er komt antwoord

De webserver heeft de bewuste **webpagina** klaar en wil een antwoord sturen naar mijn laptop. Deze computer kent mijn **ip-adres** omdat dit meegeleverd is in het packetje als **source ip-adres**.

We zakken weer van de **applicatie-laag** (ja ook een http-server is een applicatie) naar beneden.

Tussen haakjes, de **presentatie** van deze data aan de browser gebeurt met een **mime type**. Email berichten (en websites) bestaan al lang niet meer enkel uit **ascii** karakters, maar bevatten ook andere karakters, foto's, geluid en film.

MIME (Multipurpose Internet Mail Extensions) defineert een mechanisme om deze inhoud te versturen over e-mail (of zoals in ons voorbeeld over **http**). Je zou **mime** in **osi layer 6** kunnen zetten.

De **tcp-sessie** die mijn laptop heeft opgezet, die is er nog steeds. Het antwoord kan dus onmiddellijk door de **transport-laag** naar **ip**, of toch bijna. De gevraagde webpagina is immers te groot om in 1 packetje door te sturen, dus moet er geknipt en geplakt worden door **tcp**. Dit kan je zien in de sniffer als een opeenvolging van **tcp segment** packetjes, gevolgd door een **tcp ack**nowledgement.

193.191.175.137	192.168.1.34	TCP	[TCP segment of a reassembled PDU]
193.191.175.137	192.168.1.34	TCP	[TCP segment of a reassembled PDU]
192.168.1.34	193.191.175.137	TCP	34614 > http [ACK] Seq=1253 Ack=3174 Win=12736 Len=0 TSV=10991757 TSER=1835689727
193.191.175.137	192.168.1.34	TCP	[TCP segment of a reassembled PDU]
193.191.175.137	192.168.1.34	TCP	[TCP segment of a reassembled PDU]
192.168.1.34	193.191.175.137	TCP	34614 > http [ACK] Seq=1253 Ack=6054 Win=18496 Len=0 TSV=10991759 TSER=1835689727
193.191.175.137	192.168.1.34	TCP	[TCP segment of a reassembled PDU]
193.191.175.137	192.168.1.34	HTTP	HTTP/1.1 200 OK (text/html)
192.168.1.34	193.191.175.137	TCP	34614 > http [ACK] Seq=1253 Ack=8753 Win=24320 Len=0 TSV=10991760 TSER=1835689737

Het **ip** protocol zal zorgen voor de terugweg van Brussel naar Antwerpen, eventueel via andere **routers**.

Eens het antwoord terug op ons netwerk is, gaat de data van onder naar boven door de lagen, totdat onze **browser** de webpagina kan tonen.

5.4.1. vragen

1. Hoe weet de computer van de VRT dat het packetje naar de http-server moet ?

2. Hoe weet mijn laptop voor welke applicatie het packetje is ?

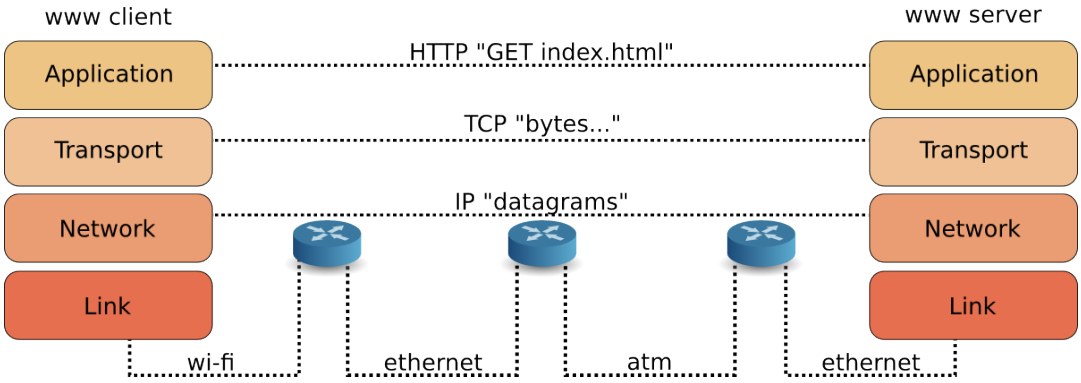
Chapter 6. enkele protocols

Table of Contents

Bij deze een korte verduidelijking over **ping**, **arp**, **tcp**, **udp** en over het tekenen van pakketjes op het bord en de juiste positie van de bytes in een pakketje.

6.1. internet lagen

We weten ondertussen dat het internet werkt met **vier lagen** die hun oorsprong vinden in de ontwikkeling van **Unix** en **tcp/ip** sinds 1969, en die perfect gedocumenteerd zijn in de **rfc's**. De tekening hieronder verduidelijkt de functionaliteit per laag.



Op de bovenste laag vinden we een applicatieprotocol (in dit geval http). De web client applicatie (bijvoorbeeld Firefox of Chrome) wil communiceren met de web server software (bijvoorbeeld apache). De bovenste laag van de client communiceert dus met de bovenste laag van de server. De web client geeft een **http** instructie **get index.html** aan de web server.

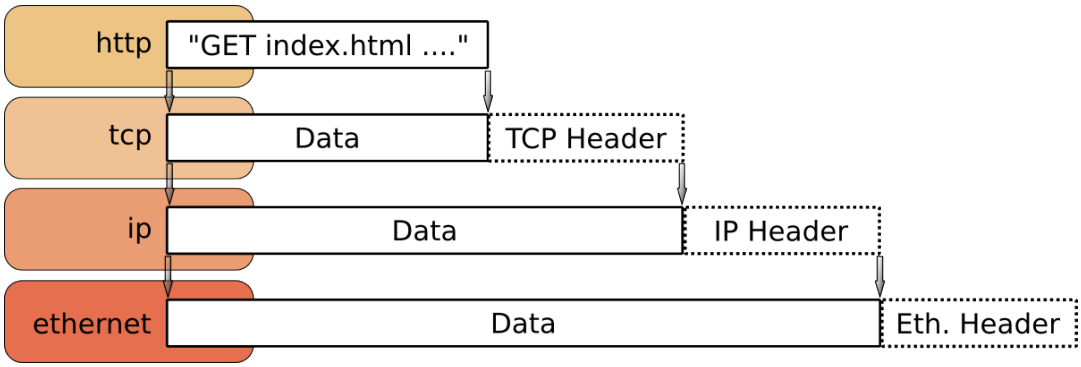
Op de **transportlaag** vinden we tcp, en die ziet een **reeks bytes** komen van de applicatielaag, maar heeft geen kennis over de betekenis van deze bytes. **tcp** wil deze stroom van bytes zonder fout afleveren bij **tcp** aan de andere kant, ook hier hebben we dus een communicatie tussen overeenkomstige lagen van beide computers. De opdracht van **tcp** is om deze stroom aan bytes zonder fout over te brengen naar de andere kant (de server).

Op de **netwerklaag** moet **ip** een degelijke inspanning leveren om de **data** die van de transportlaag komt op het juiste adres te bezorgen. Het is mogelijk dat **ip** datagrams **laat vallen** omdat een router niet kan volgen bijvoorbeeld. **ip** geeft dus geen garantie dat packetjes de bestemming bereiken en wordt daardoor **onbetrouwbaar** genoemd.

De **link** laag onderaan is verantwoordelijk voor de communicatie van link naar link (van hop naar hop), en kan bestaan uit een opeenvolging van verschillende protocols. Je kan **traceroute** gebruiken om het aantal **hops** te zien tussen je eigen computer en een bestemming op internet.

6.2. encapsulation

Elk protocol voegt een **header** toe en beschouwt al wat van de laag erboven komt als **data**. Afgebeeld ziet dat er ongeveer zo uit (niet op ware grootte).



In de applicatielaag zorgt **http** voor een instructie, gebruik makende van **ascii** bijvoorbeeld, die zinvol is voor de applicatielaag aan de andere kant. Via een standaard interface geeft **http** zijn instructie door aan **tcp**.

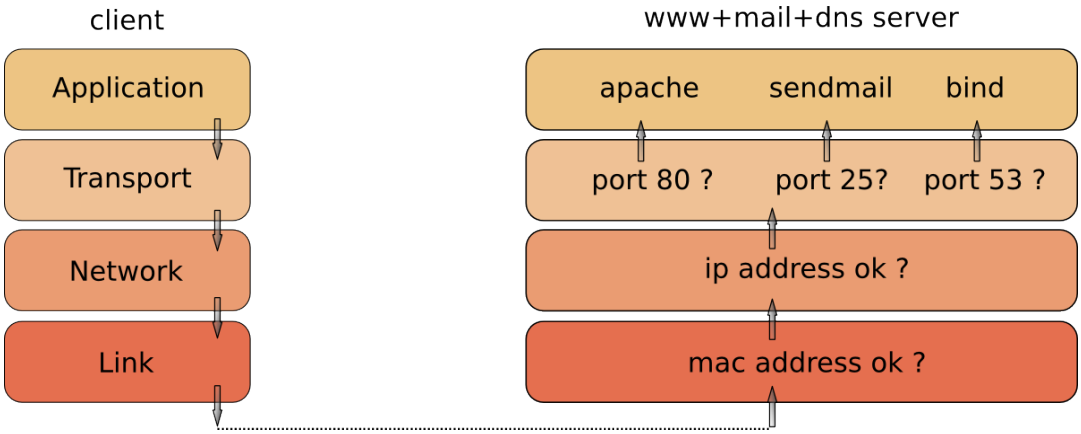
Maar **tcp** ziet enkel een stroom aan bytes, zonder interpretatie te geven aan deze bytes. Wel gaat **tcp** een header toevoegen om er voor te zorgen dat deze reeks **bytes** volledig en zonder fout (en in dezelfde volgorde) aankomt bij de transportlaag aan de andere kant.

Het hele **segment** van de transportlaag komt aan bij **ip** in de netwerklaag en wordt aldaar wederom als **data** beschouwd. Zonder enige betekenis te geven aan deze data gaat **ip** proberen om deze op de bestemming te brengen (door er een **ip header** aan toe te voegen en het vervolgens via weer een standaard interface door te geven aan de linklaag).

Aangekomen in de onderste laag beschouwt **ethernet** het hele **ip datagram** als **data** en voegt er vervolgens een **ethernet header** aan toe zodat het **frame** tot aan de volgende link geraakt.

6.3. port demultiplexing

Hoe raakt het packetje bij de juiste applicatie ?

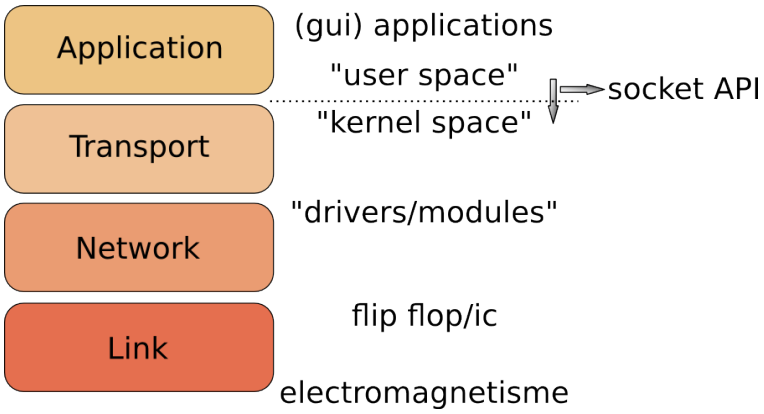


Een **frame** dat aankomt in een computer, wordt gecontroleerd op het juiste **link layer address** (mac adres in ethernet etc). Daarna wordt het resterende gedeelte gecontroleerd op **ip adres**. Indien het **destination ip address** anders is dan het ip adres van deze computer, dan wordt dit packetje weggegooid.

Raakt het alsnog door de controle, dan wordt gekeken naar de **destination port** en aan de hand van dit poortnummer wordt beslist aan welke applicatie de bytes moeten gegeven worden.

6.4. waar zitten de lagen ?

Soms vragen studenten zich af waar deze lagen te vinden zijn in de computer, daarvoor hebben we deze tekening.



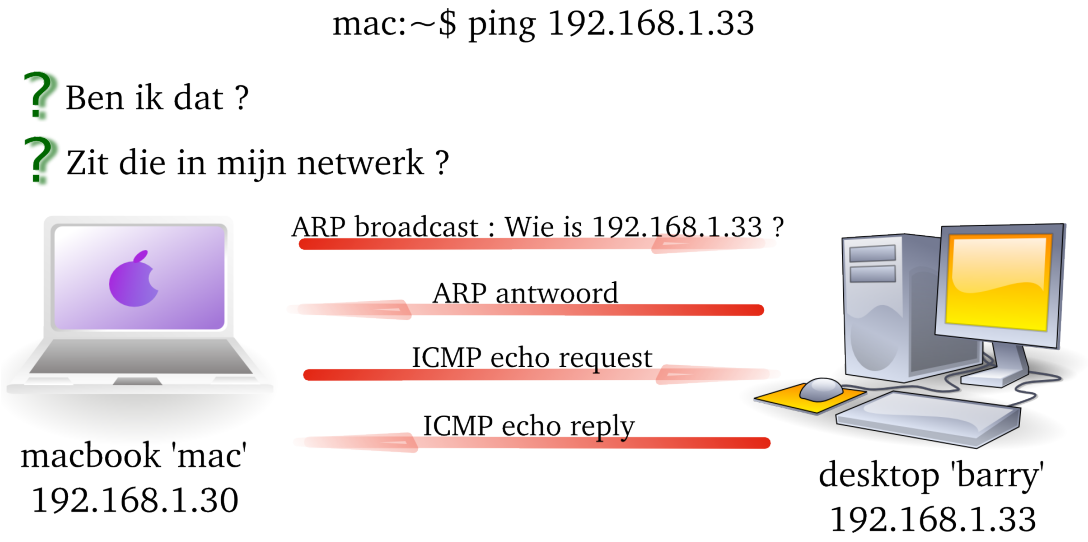
De bovenste laag zit in wat **user space** wordt genoemd. Dat is het gedeelte van het operating system en het geheugen waar de applicaties voor eindgebruikers zitten. Programmeurs gebruiken een **socket** om iets te doen op het netwerk, en kunnen deze aanmaken via aan **api** (Application Programming Interface).

De transport en de netwerklaag zijn ook geschreven in software, maar zitten in **kernel space**. Op Unix/Linux computers maken de protocols **tcp** en **ip** deel uit van de **kernel** (als module), terwijl dit in Microsoft systemen ook als **driver** kan geprogrammeerd zijn.

Als we bij de linklaag komen, dan vinden we ons frame als bytes in **ic's** op de netwerkkaart. Een **ic** is een integrated circuit (of chip). De bits van de bytes zelf zitten opgeslagen in wat men een **flip flop** noemt. Eens op de utp kabel zijn onze bytes volledig omgezet in electromagnetische straling.

6.5. ping en arp

Het **arp** protocol is een **layer 2** protocol dat een link legt tussen het **ip-adres** en het **mac-adres** van een computer. We hebben in de klas deze werking aangetoond met behulp van het **ping** commando en de **wireshark** sniffer. We hebben ook het **arp** commando uitgevoerd om de **arp-cache-table** te bekijken.



6.6. tcp en udp

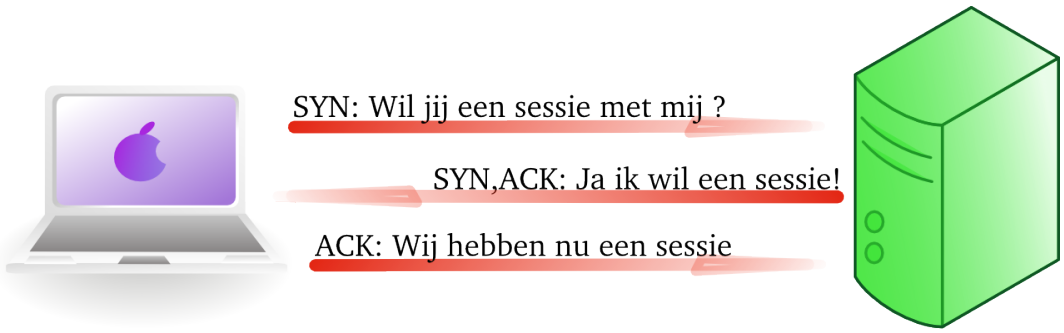
Behalve **tcp** vinden we in **layer 4** ook nog **udp**. U hebt genoteerd dat alvorens een **http request** verstuurd wordt, er eerst een **tcp handshake** plaatsvindt, maar dat er voor de **dns query** helemaal geen handshake was. Dat komt omdat **http** gebruikt maakt van **tcp** in **layer 4**, terwijl **dns** hier werkt bovenop **udp**.

6.6.1. tcp

tcp staat voor **Transmission Control Protocol**. Het is een protocol dat een **connectie** opzet alvorens data te versturen. Als je zekerheid wil dat je packetjes aankomen, dan is **tcp** het aangewezen protocol. In de **tcp header** zit heel wat overhead. Als een **tcp packetje** niet aankomt, dan wordt het nogmaals verstuurd.

De huidige **tcp** standaard werd vastgelegd in **rfc 793** van 1981.

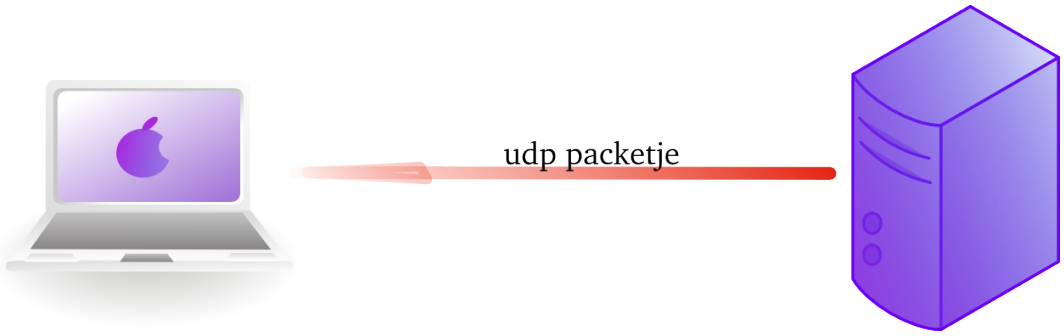
De **tcp triple handshake** gaat steeds vooraf aan de **tcp** connectie. Dit wil zeggen dat er minstens vier packetjes over het netwerk gaan.



6.6.2. udp

In tegenstelling tot **tcp** is er bij **udp** geen connectie, en ook geen handshake. Men noemt **udp** een **connectionless** protocol.

udp heeft minder overhead dan **tcp** en is bijgevolg sneller voor het doorsturen van data. Maar **udp** doet geen controle of een packetje ook wel degelijk aankomt. Als een **udp** packetje zijn bestemming niet bereikt, dan ben je het kwijt.



6.6.3. oefening

Bedenk zelf wanneer je udp zou gebruiken, en wanneer tcp.

e-mail van je manager ?

website van een klant ?

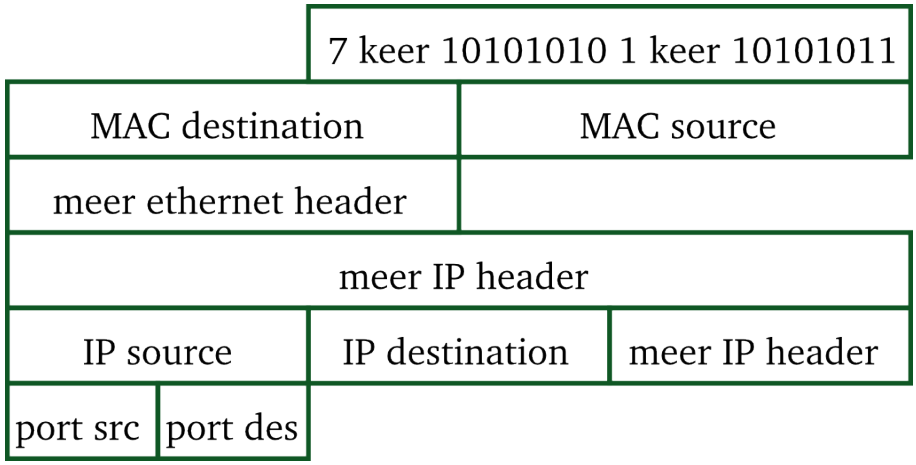
live radio uitzending van Clijsters tegen Henin ?

dns ?

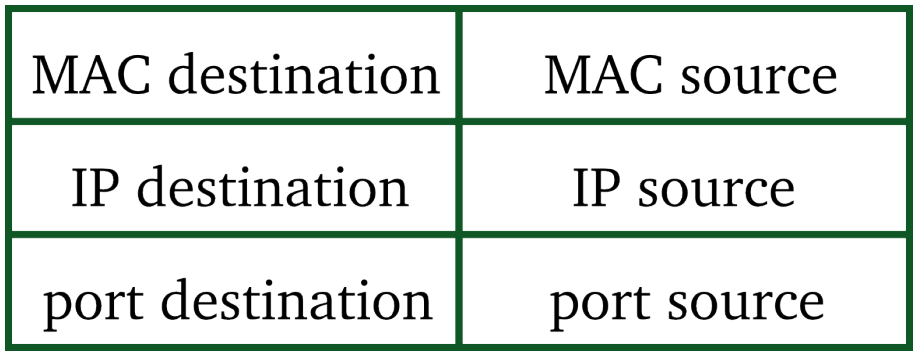
6.7. packetjes

Op het bord heb ik regelmatig 'packetjes' getekend. Technisch spreken we van een **ethernet frame** en van een **ip datagram**. Het exacte formaat (per bit of byte) van deze packetjes behoort niet tot de leerstof. Ik veronderstel hier enkel dat jullie weten dat een ethernet netwerkkaart de eerste zes bytes, zijnde het bestemmings-mac-adres vergelijkt met haar eigen mac-adres.

Hieronder een **ethernet frame** met de bytes op ware grootte en enige info op de juiste positie.



Omdat we toch iets moeten kennen van **mac-adressen**, **ip-adressen** en **poorten** om protocols en toestellen te begrijpen, gebruiken we onderstaande vereenvoudigde voorstelling van een **packetje** op het netwerk.



Chapter 7. weergave getallen

Table of Contents

Werken met netwerken in de informatica wil zeggen dat U regelmatig getallen gaat tegenkomen die niet in het decimale talstelsel geschreven zijn. Dit hoofdstuk leert u tellen in het binaire, het octale en het hexadecimale talstelsel.

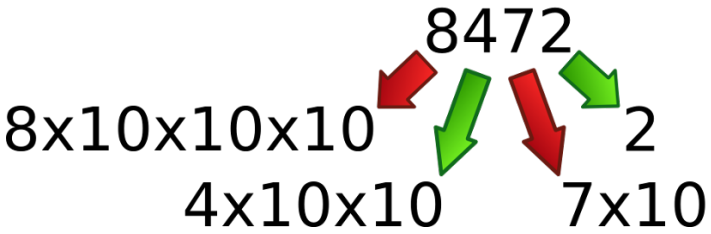
7.1. decimaal of tientallig stelsel

In het **decimaal** stelsel zijn er **tien** cijfers waarmee we kunnen werken, dat zijn in opklimmende volgorde 0, 1, 2, 3, 4, 5, 6, 7, 8 en 9. Met deze cijfers kunnen we van nul tot negen tellen.

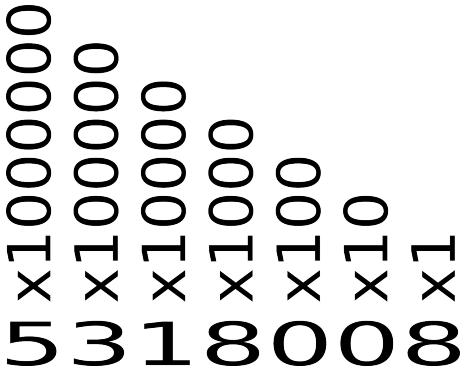
Als we nog verder willen tellen, dan moeten we twee cijfers combineren. Het eerste cijfer heeft dan een grotere waarde dan het tweede. De waarde van het eerste cijfer is het **tienvoud** van zijn nominale waarde.

Willen we nog verder, dan zetten we nog een extra cijfer voor de bestaande twee, en vermenigvuldigen deze waarde met **tien en nog eens tien**. Deze methode laat toe om oneindig ver te tellen.

Hieronder de decimale getallen 42, 33 en 8472 en de waarden van elk individueel cijfer. We vermenigvuldigen telkens met **tien** omdat we in het **tien**tallig stelsel aan het werken zijn.



Hieronder ziet U het getal vijfhonderdeenendertig miljoen achtduizend en acht, en een verkorte berekening van de waarde van elke cijfer.



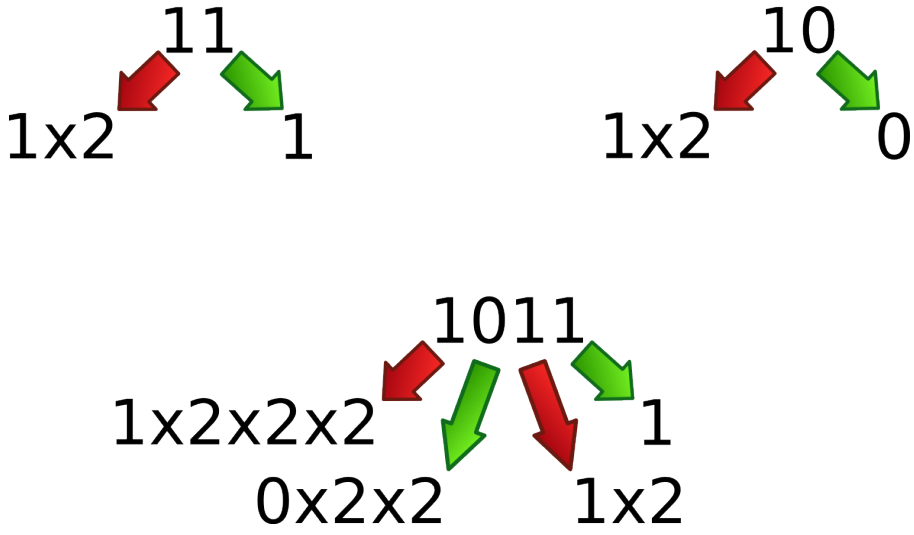
7.2. binair of tweetallig stelsel

In het **binair** stelsel zijn er **twee** cijfers waarmee we kunnen werken, dat zijn in opklimmende volgorde 0 en 1. Met deze cijfers kunnen we van nul tot een tellen.

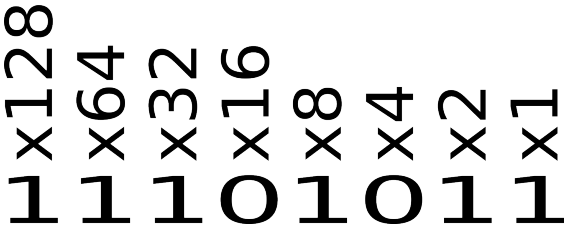
Als we nog verder willen tellen, dan moeten we twee cijfers combineren. Het eerste cijfer heeft dan een grotere waarde dan het tweede. De waarde van het eerste cijfer is het **tweevoud** van zijn nominale waarde.

Willen we nog verder, dan zetten we nog een extra cijfer voor de bestaande twee, en vermenigvuldigen deze waarde met **twee en nog eens twee**. Deze methode laat toe om oneindig ver te tellen.

Hieronder de binaire getallen 11, 10 en 1011 en de waarden van elk individueel cijfer. We vermenigvuldigen telkens met **twee** omdat we in het **twee**tallig stelsel aan het werken zijn.

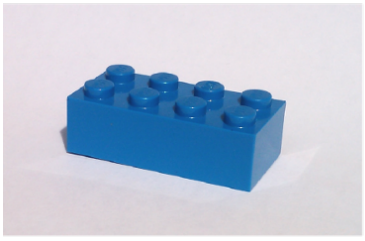
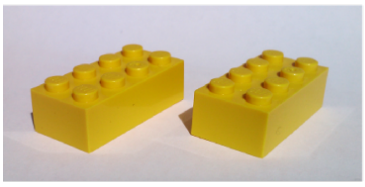
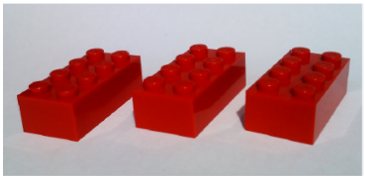
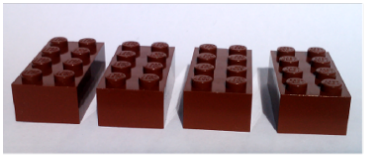
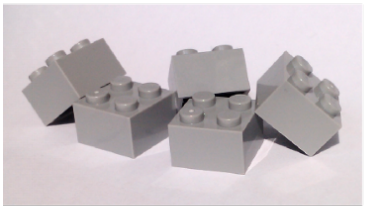
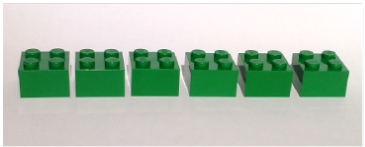


Hieronder ziet U het getal 11101011, en een verkorte berekening van de waarde van elke cijfer.



7.3. decimaal en binair naast elkaar

We zetten het decimale en binaire talstelsel even naast elkaar.

	decimaal	binair
	= 0 =	0
	= 1 =	1
	= 2 =	10
	= 3 =	11
	= 4 =	100
	= 5 =	101
	= 6 =	110
	= 7 =	111

7.4. hexadecimaal of zestientallig stelsel

In het **hexadecimaal** stelsel zijn er **zestien** cijfers waarmee we kunnen werken, dat zijn in opklimmende volgorde 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E en F. Met deze cijfers kunnen we van nul tot F tellen.

Als we nog verder willen tellen, dan moeten we twee cijfers combineren. Het eerste cijfer heeft dan een grotere waarde dan het tweede. De waarde van het eerste cijfer is het **zestienvoud** van zijn nominale waarde.

Willen we nog verder, dan zetten we nog een extra cijfer voor de bestaande twee, en vermenigvuldigen deze waarde met **zestien en nog eens zestien**.

7.5. octaal of achttallig stelsel

In het **octaal** stelsel zijn er **acht** cijfers waarmee we kunnen werken, dat zijn in opklimmende volgorde 0, 1, 2, 3, 4, 5, 6 en 7. Met deze cijfers kunnen we van nul tot zeven tellen.

Als we nog verder willen tellen, dan moeten we twee cijfers combineren. Het eerste cijfer heeft dan een grotere waarde dan het tweede. De waarde van het eerste cijfer is het **achtvoud** van zijn nominale waarde.

Willen we nog verder, dan zetten we nog een extra cijfer voor de bestaande twee, en vermenigvuldigen deze waarde met **acht en nog eens acht**.

Table 7.1. vier talstelsels naast elkaar

decimaal	binair	hexadecimaal	octaal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17
16	10000	10	20
17	10001	11	21
18	10010	12	22
19	10011	13	23
20	10100	14	24
21	10101	15	25
22	10110	16	26
23	10111	17	27
24	11000	18	30
25	11001	19	31
26	11010	1A	32
27	11011	1B	33
28	11100	1C	34
29	11101	1D	35
30	11110	1E	36
31	11111	1F	37
32	100000	20	40

7.6. oefening talstelsels

1. Vul de ontbrekende waarden in. (Octaal is niet belangrijk.)

Table 7.2. oefening talstelsels

decimaal	binair	hexadecimaal	octaal
5			
	100		
		FF	
192			300
168			
		0A	
33			
	101010		

2. Schrijf de cijfers C0 A8 01 01 in decimale vorm.

3. Hoe schrijf je 255.255.255.0 in hexadecimale vorm ?

4. En nu een moeilijke vraag : Wanneer is 6 maal 9 gelijk aan 42 ?

7.7. veel voorkomende getallen

Het layer 2 broadcast mac adres bestaat uit 48 eentjes, maar wordt altijd hexadecimaal voorgesteld:

```
FF:FF:FF:FF:FF:FF
```

Standaard **classful** subnet masks (we bespreken dit later in detail) komen soms decimaal, soms hexadecimaal voor in de uitvoer van systeembeheer programma's:

```
decimaal       hexadecimaal
255.0.0.0      FF000000
255.255.0.0    FFFF0000
255.255.255.0 FFFFFF00
```

Alle data en alle applicaties op de computer bestaan uit bits. Als je in een tekstbestand het getal **42** wil bewaren, dan worden de volgende bits (in een volledige byte) bewaard:

```
00101010
```

7.8. machten van twee

De **map van het internet** van pagina 16 bestaat uit exact 256 vakjes, in een byte kunnen 8 bits, **ip-adressen** bestaan uit 32 bits, het aantal mogelijke waarden die je in een **byte** kan steken is 256, heel wat encryptie werkt heden met sleutels van 128-bits, niet toevallig allemaal machten van het getal twee!

Machten van twee zijn eenvoudiger te schrijven in het binaire of het hexadecimale stelsel.

Table 7.3. machten van twee

decimaal	binair	hexadecimaal
2	10	2
4	100	4
8	1000	8
16	10000	10
32	100000	20
64	1000000	40
128	10000000	80
256	100000000	100

Ook grotere machten van twee worden gebruikt. Zo is de grootte van een sector op harde schijven (voorlopig) 512 bytes, is het aantal **tcp** en **udp** poorten gelijk aan 65.536, is het aantal kleuren dat een truecolor videokaart kan tonen 16777256, is de 640KB geheugen in oude computers niet toevallig 512KB + 128KB en is een kilobyte (of kibibyte) exact 1024 bytes. De maximum grootte van variabelen in

programmeertalen is ook een macht van twee (65536 of 4294967296), schermen hebben wel eens een 1024x786 of 1280x1024 resolutie (768 = 512 + 256, 1280 = 1024 + 256) en Microsoft Excel kan maximum 65536 of 1048576 kolommen bevatten.

Table 7.4. grote machten van twee

decimaal	hexadecimaal
256	100
512	200
1024	400
2048	800
4096	1000
65536	10000
16777216	1000000
4294967296	100000000

Chapter 8. ip-adressen

Table of Contents

Jullie hebben geluk. In de jaren 90 waren zowel IPX/SPX, NetBEUI, Appletalk als IBM SNA verplichte kost naast tcp/ip. Vandaag spreken alle computers tcp/ip, we hebben dus ruim de tijd om ip-adressen in detail te bespreken.

8.1. oefening

0. Sluit je browser, tijdens de oefening ben je even het internet kwijt.

1. Gebruik **ipconfig** op Microsoft of **ifconfig** op alle andere besturingssystemen om je eigen **ip-adres** te vinden. Noteer het hier.

2. Zoek een partner en noteer zijn/haar **ip-adres**.

3. Test met **ping** dat jullie elkaar kunnen bereiken. Dit zou moeten werken.

```
ping ip-adres-partner
```

4a. Zoek met **netstat -nr** of **route** of **route print** de default router (default gateway) en test of je deze kan pingen. Dit zou moeten werken.

4b. Zoek met **ipconfig /all** op Windows of **cat /etc/resolv.conf** op Linux het ip-adres van de nameserver (dns server) en test dat je ook deze kan pingen. Dit zou moeten werken.

5. Zoek de **grafische interface** om de eigenschappen van je ip-configuratie aan te passen, en noteer de informatie de je daar vindt. De kans is groot dat hier 'automatisch' of 'dhcp' staat.

6. Gebruik deze grafische interface en de informatie uit vraag 1 om het **eerste cijfer** van je ip-adres aan te passen. Verander dit eerste cijfer in 10. Heb je nog steeds verbinding met je buurman ? (Dit zou moeten werken als jullie allebei de aanpassing gedaan hebben.)

7. Test met **ping** dat jullie nog steeds de router en dns-server kunnen bereiken. Dit zou niet meer mogen werken.

8. Lukt dit ook als je het eerste cijfer verandert in 8 ? Idem als voorheen zou je wel met elkaar verbinding moeten hebben, maar niet met de dns-server of router.

9. Lukt dit ook als je het eerste cijfer verandert in 127 ? Dit zou niet mogen werken.

10. Verander de eerste twee cijfers van je ip-adres in **169.254**, maar behoud de laatste twee. Lukt dit ?

11. Verander je ip-adres in 192.168.1.259. Waarom lukt dit niet ?

12. Zet je ip-configuratie terug zoals ze stond voor de oefening. Test dat je terug verbinding hebt met internet door te surfen naar linux-training.be .

8.2. ip-adressen

Een **ip-adres** is een 32-bit getal dat als logisch adres aan een computer wordt gegeven. Een **ip-adres** is de unieke identificatie van een **host** op laag 3.

Theoretisch hebben we 4294967296 ip-adressen (256*256*256*256).

In decimale octetjes schrijven we dat van 0.0.0.0 tot 255.255.255.255, in hexadecimale bytes is dat van 00:00:00:00 tot FF:FF:FF:FF en binair is dat van 00000000.00000000.00000000.00000000 tot 11111111.11111111.11111111.11111111.

8.3. private en publieke adressen

Om een geldig **ip-adres** te hebben op het **internet**, moet je dit aankopen bij je lokale **RIR (Regional Internet Registry)**. Voor Europa is dit **RIPE NCC (Réseaux IP Européens Network Coordination Centre)** in Amsterdam.

In de meeste gevallen is het de **isp (Internet Service Provider)** die deze aankoop doet, eindgebruikers merken daar weinig van.

Niet alle **ip-adressen** zijn geldig op internet. Sommige reeksen zijn gereserveerd voor speciale doeleinden.

8.3.1. private adressen

Er zijn drie reeksen gereserveerd voor **privé**-gebruik. Deze ip-adressen zijn niet te gebruiken op internet (ze werken niet). Technisch kan je zeggen dat routers op internet alle packetjes voor deze adresssen automatisch verwerpen.

De drie reeksen zijn:

Table 8.1. private ip-adressen

eerste ip-adres	laatste ip-adres	aantal adressen in de reeks
10.0.0.0	10.255.255.255	16777216 (+16 miljoen)
172.16.0.0	172.31.255.255	1048576 (+1 miljoen)
192.168.0.0	192.168.255.255	65536 (+65 duizend)

De volledige definitie van deze reeksen vindt u in **rfc 1918**.

`http://www.rfc-editor.org/rfc/rfc1918.txt`

8.3.2. loopback adressen

Elke **tcp/ip stack** heeft een virtuele **loopback interface**. Dit is een ip-adres dat gebruikt wordt om naar de computer zelf te wijzen (en om te testen of je **tcp/ip** stack werkt).

Meestal is dit **127.0.0.1**, vaak wijst de hele reeks echter naar ditzelfde adres.

```
van 127.0.0.0 tot 127.255.255.255
```

Meestal wordt **localhost** vertaald naar **127.0.0.1**.

8.3.3. zeroconf adressen

Een **zeroconf** adres wordt ook wel een **link local** adres genoemd. Het is een ip-adres dat begint met **169.254** en dan twee willekeurig gekozen bytes. Microsoft noemt dit **APIPA (Automatic Private IP Adddressing)**.

Een host die automatisch een ip-adres zou moeten krijgen van **dhcp**, maar dit niet krijgt, kan zichzelf van een **zeroconf** adres voorzien. De host zal op het lokale netwerk (de **local link**) eerst testen dat geen enkele andere host dit adres gebruikt, alvorens het aan zichzelf toe te wijzen.

```
van 169.254.0.0 tot 169.254.255.255
```

Als een computer in de praktijk in een bedrijf een **zeroconf** adres heeft, dan wil dit meestal zeggen dat de netwerkbekabeling van de computer onderbroken is. Soms duidt zo'n adres op een probleem met de **dhcp server**.

8.3.4. oefening ip-adressen

Duid voor de volgende ip-adressen aan waarvoor ze dienen (privaat, publiek, loopback of zeroconf)

Table 8.2. oefening gereserveerde ip-adressen

ip-adres	gereserveerd doel
192.168.244.16	privaat
195.238.2.21	
169.254.33.42	
127.0.0.1	
9.33.42.12	
10.11.12.13	
171.191.42.33	
172.18.33.42	

8.4. ip-adres klassen

IP-adressen zijn onderverdeeld in klassen. Een adres van **klasse A** begint binair met als meest significante bit een 0. Een **klasse B** adres begint binair met 10, **klasse C** met 110, **klasse D** met 1110 en tenslotte **klasse E** met 1111.

De eerste byte decimaal omgerekend, komen we tot de volgende indeling van ip-adressen:

Table 8.3. ip address classes

class	first bit(s)	first decimal byte
A	0	0-127
B	10	128-191
C	110	192-223
D	1110	224-239
E	1111	240-255

Alle ip-adressen die beginnen met een getal lager dan 127 zijn dus klasse A adressen (onthoud dat 127 zelf gereserveerd is voor de loopback adapter).

8.5. oefening ip-adres klassen

Duid voor de volgende ip-adressen steeds de correcte klasse aan:

Table 8.4. oefening classful ip addressing

ip-adres	klasse ?
192.168.42.33	
9.101.12.01	A
188.33.42.33	
9.42.12.33	
230.19.4.42	
11.19.6.200	
191.192.193.194	
134.0.0.42	

8.6. default subnet masks

Klasse A, B en C ip-adressen hebben een standaard **subnet mask**. Klasse A heeft een subnet mask waar de eerste 8 bits gelijk zijn aan 1 (en de rest aan 0). Bij klasse B zijn de eerste 16 bits een 1, en bij klasse C de eerste 24 bits.

Dit geeft ons de volgende tabel:

Table 8.5. default subnet mask

klasse	aantal bits op 1	default subnet mask
A	8	255.0.0.0
B	16	255.255.0.0
C	24	255.255.255.0

8.7. oefening default subnet masks

Schrijf voor de volgende ip-adressen steeds de default subnet mask op:

Table 8.6. oefening default subnet masks

ip-adres	mask ?
192.168.42.33	
9.101.12.01	255.0.0.0
188.33.42.33	
9.42.12.33	
230.19.4.42	
11.19.6.200	
191.192.193.194	
134.0.0.42	

8.8. network id en host id

Bij het uitvoeren van **ifconfig** op een Unix/Linux en **ipconfig** op een MS Windows computer merkt je dat je steeds een **ip-adres** en en **subnet mask** krijgt. De combinatie van die twee bepaalt in welk netwerk een computer zich bevindt.

Als het subnet mask gelijk is aan 255.0.0.0, dan vormt de eerste byte van het ip-adres aangevuld met nullen het **network id**. Bij 255.255.0.0 als subnet mask zijn er twee bytes (aangevuld met nullen) die het **network id** vormen. Bij 255.255.255.0 zijn er drie bytes (en een nul) die het **network id** vormen.

De rest van het ip-adres is dan het **host id**. Het **network id** bepaalt het netwerk waarin een computer zich bevindt, het **host id** is uniek voor een host binnen het netwerk.

Een overzichtje met voorbeelden:

Table 8.7. network id en host id

ip-adres	default subnet mask	network id	host id
192.168.1.42	255.255.255.0	192.168.1.0	42
192.168.1.33	255.255.255.0	192.168.1.0	33
192.168.12.1	255.255.255.0	192.168.12.0	1
172.16.12.1	255.255.0.0	172.16.0.0	12.1
172.16.33.42	255.255.0.0	172.16.0.0	33.42
10.3.0.4	255.0.0.0	10.0.0.0	3.0.4
10.33.0.42	255.0.0.0	10.0.0.0	33.0.42

8.9. oefening network id en host id

1. Noteer de **network id** en **host id** voor de volgende ip-adressen.

```
192.168.42.42
```

```
9.8.7.6
```

```
42.42.42.42
```

```
169.254.42.1
```

```
191.42.17.18
```

```
193.42.17.18
```

8.10. lokale computer of niet ?

Wat is het nut van het kennen van een **network id**? Wel het **network id** bepaalt of een computer lokaal in je netwerk staat of niet.

Indien je wil communiceren met een andere computer, dan heb je zijn **ip-adres** nodig. Als die computer dan op hetzelfde netwerk zit, dan doet jouw computer een **arp** om het **MAC adres** van de andere computer te vinden. Als die computer echter op een ander netwerk zet, dan stuurt jouw computer het packetje naar de **router**.

Alvorens over te gaan tot een **arp** (voor die andere computer of voor de router) zal jouw computer het **network id** van jouw computer vergelijken met dat van de andere computer. Indien gelijk, dan betreft het een computer op hetzelfde netwerk, indien verschillend, dan staat die computer achter de **router**.

Een overzichtje met voorbeelden (gebruik standaard subnet mask):

Table 8.8. local or remote computer?

computer A	computer B	network id A	network id B	lokaal ?
192.168.1.42	192.168.1.33	192.168.1.0	192.168.1.0	ja
192.168.1.33	192.168.12.1	192.168.1.0	192.168.12.0	nee
10.3.0.4	10.33.0.42	10.0.0.0	10.0.0.0	ja

8.11. oefening lokale computer of niet?

1. Staan de volgende computers in hetzelfde netwerk ?

```
192.168.1.42 en 192.168.1.33


10.105.42.42 en 10.105.42.33


10.105.42.42 en 10.99.42.33


11.16.42.42 en 12.16.42.33


169.254.18.42 en 169.254.33.42


191.168.42.42 en 191.168.33.33


9.1.2.3 en 9.123.234.42
```

8.12. subnet notatie

We kunnen de notatie van **ip-adres/subnet mask** afkorten als we voor de subnet mask enkel het aantal bits vernoemen dat op 1 staat. Zo wordt 255.0.0.0 gelijk aan /8, wordt 255.255.0.0 gelijk aan /16 en 255.255.255.0 gelijk aan /24.

Afgekort kunnen we dus 192.168.1.42/24 schrijven i.p.v. 192.168.1.42 met subnet mask 255.255.255.0 .

Table 8.9. cidr notatie

klasse	default subnet mask	notatie
A	255.0.0.0	/8
B	255.255.0.0	/16
C	255.255.255.0	/24

8.13. computers in een netwerk tellen

Hoeveel computers kan je zetten in een netwerk? Houd rekening met de **network id** die zelf al een ip-adres gebruikt. Elk netwerk heeft ook een **broadcast** adres (alle decimale delen van het host id zijn dan 255).

Bijvoorbeeld:

Table 8.10. aantal computers in een subnet

network	network id	broadcast ip	max aantal hosts
192.168.1.0/24	192.168.1.0	192.168.1.255	256 - 2 = 254
192.168.15.0/24	192.168.15.0	192.168.15.255	256 - 2 = 254
172.16.0.0/16	172.16.0.0	172.16.255.255	256*256 - 2 = 65534
10.0.0.0/8	10.0.0.0	10.255.255.255	256*256*256 - 2 = 16777214

Chapter 9. 4 miljard ip-adressen

Table of Contents

9.1. te weinig ip-adressen ?

Is vier miljard dan niet genoeg ? We verspillen enorm veel ip-adressen door de verkoop van klasse A en klasse B aan organisaties die eigenlijk veel minder adressen nodig hebben.

Wat doe je als je 300 adressen nodig hebt ? Of 2000 ? Je kan kiezen voor een klasse B range, maar dan verspil je meer dan 90 procent. Je kan ook zeggen dat 8 klasse C adressen voldoen voor 2000 computers, maar dan vergroot je de **routing tables** weer.

9.2. ip-adressen verdelen

Klasse A adressen kunnen dus een dikke 16 miljoen computers bevatten, **klasse B** een goeie 65 duizend en **klasse C** iets meer dan 200.

Stel dat jouw organisatie 3000 ip-adressen nodig heeft, dan moet je een klasse B gebruiken. Maar dat wil zeggen dat je meer dan 62 duizend ip-adressen **verspilt**. Onthou ook dat heel wat klasse A ranges begin jaren 90 in hun geheel verkocht zijn.

In de jaren 70-80 was dit een goed systeem, er waren immers maar enkele duizenden computers op dit **internetwerk**. Maar de laatste jaren zijn er meer dan een miljard computers verbonden met internet. Met een totaal van vier miljard (256*256*256*256) ip-adressen kunnen we ons niet langer veroorloven om ip-adressen te verspillen.

9.3. probleem voor de routing tables

Je zou kunnen argumenteren op het vorige dat je ook twaalf **klasse C** kan gebruiken voor een netwerk met 3000 computers. En dat is correct, maar heeft wel tot gevolg dat de **routing tables** in de internet routers er twaalf routes bij krijgen i.p.v. slechts eentje.

Alle grote netwerken opbouwen met klasse C ip ranges is dus ook geen oplosssing. (routing tables bespreken we later)

9.4. Is nat een oplossing ?

Een andere verzachtende techniek voor het naderende tekort aan ip-adressen is **nat**. Een **nat** toestel kan meerdere **private ip-adressen** (en dus meerdere computers) met een (of enkele) publieke adressen verbinden met het internet.

Maar **nat** heeft dan weer het nadeel dat niet alle applicaties kunnen werken achter een **nat**. Bijvoorbeeld **ipsec** (want de poort-informatie is versleuteld), **sip** (Voice over IP), **ftp**, **dns zone transfers** (we bespreken dit later), **dhcp**, **snmp** en sommige multiplayer spelletjes op Microsofts xbox werken niet over **nat**.

Chapter 10. van subnet naar supernet

Table of Contents

Stel je eens voor dat computers **binair** rekenen i.p.v. decimaal. We bekijken de gevolgen hiervan.

10.1. binaire subnets

Computers rekenen met **bits** en **bytes**, mensen rekenen decimaal. Onze computers zien een **ip-adres** als een 32-bit getal, hetzelfde geldt voor de **subnet mask**. De drie subnet masks die we tot nu toe kennen zijn decimaal en binair voorgesteld in de volgende tabel:

Table 10.1. binary classful subnets

class	subnet	binary subnet
A	255.0.0.0	11111111.00000000.00000000.00000000
B	255.255.0.0	11111111.11111111.00000000.00000000
C	255.255.255.0	11111111.11111111.11111111.00000000

Voor de volledigheid volgt ook nog eens het aantal computers dat in een **classful** netwerk past.

Table 10.2. max computers classful subnets

class	binary subnet	computers
A	11111111.00000000.00000000.00000000	256*256*256 - 2
B	11111111.11111111.00000000.00000000	256*256 - 2
C	11111111.11111111.11111111.00000000	256 - 2

10.2. supernetting

Zoals je ziet, begint de subnet mask binair steeds met eentjes en eindigt steeds met nulletjes. Wat als we nu een eentje meer of minder zetten ?

Onze verkorte notatie laat toe om dit snel te noteren. We schrijven simpelweg 172.16.0.0/17 i.p.v. 172.16.0.0/16 . Maar wat zijn de gevolgen hiervan ?

Ten eerste is een eentje meer hetzelfde als het halveren van het aantal computers in het netwerk. Want we vergroten het **network id** en verkleinen het aantal mogelijke **host id's**.

Laten we ons voorbeeld beginnen met het netwerk in de klas zoals het er nu uitziet:

Table 10.3. /24 netwerk in de klas

network id	subnet	binair subnet	aantal computers
192.168.0.0	/24	11111111.11111111.11111111.00000000	256 - 2 = 254

En kijk wat er gebeurt als we de **subnet mask** veranderen van 24 **bits** naar 25 **bits** die op 1 staan.

Table 10.4. /25 netwerk in de klas

network id	subnet	binair subnet	aantal computers
192.168.0.0	/25	11111111.11111111.11111111.10000000	128 - 2 = 126

De reeks ip-adressen die behoren tot 192.168.0.0/25 begint met 192.168.0.1 en eindigt met 192.168.0.126, de network id is 192.168.0.0 en het **broadcast** adres is 192.168.0.127.

Als we dit binair voorstellen wordt het duidelijk:

Table 10.5. /25 binair bekijken

omschrijving	binair	decimaal
network id	11000000.10101000.00000000.00000000	192.168.0.0
subnet mask	11111111.11111111.11111111.10000000	255.255.255.128
eerste ip	11000000.10101000.00000000.00000001	192.168.0.1
tweede ip	11000000.10101000.00000000.00000010	192.168.0.2
derde ip	11000000.10101000.00000000.00000011	192.168.0.3
vierde ip	11000000.10101000.00000000.00000100	192.168.0.4
vijfde ip	11000000.10101000.00000000.00000101	192.168.0.5
...	...	...
voorlaatste ip	11000000.10101000.00000000.01111101	192.168.0.125
laatste ip	11000000.10101000.00000000.01111110	192.168.0.126
broadcast ip	11000000.10101000.00000000.01111111	192.168.0.127

We voegen nog een **bit** toe aan de **subnet mask**, dan komen we aan /26 (255.255.255.192). De tabel ziet er dan als volgt uit:

Table 10.6. /26 binair bekijken

omschrijving	binair	decimaal
network id	11000000.10101000.00000000.00000000	192.168.0.0
subnet mask	11111111.11111111.11111111.11000000	255.255.255.192
eerste ip	11000000.10101000.00000000.00000001	192.168.0.1
tweede ip	11000000.10101000.00000000.00000010	192.168.0.2
derde ip	11000000.10101000.00000000.00000011	192.168.0.3
...	...	...
voorlaatste ip	11000000.10101000.00000000.00111101	192.168.0.61
laatste ip	11000000.10101000.00000000.00111110	192.168.0.62
broadcast ip	11000000.10101000.00000000.00111111	192.168.0.63

10.3. binaire subnets decimaal voorstellen

We weten al dat 255 **decimaal** gelijk is aan 11111111 **binair** en dat 0 decimaal gelijk is aan 00000000 binair (in byte-vorm). Een binair **subnet mask** begint steeds met eentjes, en eindigt met nulletjes. De volgende tabel kan dus handig zijn bij het decimaal neerschrijven van een binair subnet mask.

Table 10.7. decimale waarde binaire subnet bytes

binair subnet mask	decimaal getal
11111111	255
11111110	254
11111100	252
11111000	248
11110000	240
11100000	224
11000000	192
10000000	128
00000000	0

10.4. 32 binaire subnet masks

Bij supernetting zijn er theoretisch 32 **binaire subnet masks** i.p.v. de drie **classful** (255.255.255.0, 255.255.0.0, 255.0.0.0). We zetten er 31 op een rijtje.

Table 10.8. 31 binaire subnets

aantal bits op 1	decimale waarde
1	128.0.0.0
2	192.0.0.0
3	224.0.0.0
4	240.0.0.0
5	248.0.0.0
6	252.0.0.0
7	254.0.0.0
8	255.0.0.0
9	255.128.0.0
10	255.192.0.0
11	255.224.0.0
12	255.240.0.0
13	255.248.0.0
14	255.252.0.0
15	255.254.0.0
16	255.255.0.0
17	255.255.128.0
18	255.255.192.0
19	255.255.224.0
20	255.255.240.0
21	255.255.248.0
22	255.255.252.0
23	255.255.254.0
24	255.255.255.0
25	255.255.255.128
26	255.255.255.192
27	255.255.255.224
28	255.255.255.240
29	255.255.255.248
30	255.255.255.252
31	255.255.255.254

10.5. aantal computers

We kunnen de tabel uitbreiden met een kolom die het aantal computers telt dat we in deze **binaire** netwerkjes kunnen plaatsen.

De simpelste formule om het aantal computers in een **subnet** te berekenen is twee verheffen tot de macht 'het aantal bits op 0 in de subnet' min twee. Bij 16 bits op 1 wordt dat dus 2 tot de zestiende min twee. In de tabel gebruiken we 256 i.p.v. twee tot de achtste.

Table 10.9. aantal computers in binaire subnets

aantal bits op 1	berekening	aantal computers
8	256*256*256-2	16777214
9	256*256*128-2	8388606
10	256*256*64-2	4194302
11	256*256*32-2	2097150
12	256*256*16-2	1048574
13	256*256*8-2	524286
14	256*256*4-2	262142
15	256*256*2-2	131070
16	256*256-2	65534
17	256*128-2	32766
18	256*64-2	16382
19	256*32-2	8190
20	256*16-2	4094
21	256*8-2	2046
22	256*4-2	1022
23	256*2-2	510
24	256-2	254
25	128-2	126
26	64-2	62
27	32-2	30
28	16-2	14
29	8-2	6
30	4-2	2

Zie ook eens op **http://www.rfc-editor.org/rfc/rfc1878.txt**.

10.6. network id en host id vinden

Als we een **ip-adres** krijgen, kunnen we dan het **network id** en het **host id** vinden ?

We zullen beginnen met een simpel classful voorbeeld: **192.168.1.5/24**.

```
ip-adres    : 192.168.  1.5
subnet mask : 255.255.255.0
network id  : 192.168.  1.0
host id     :             5
```

We kunnen dit ook binair bekijken:

```
ip-adres    : 11000000.10101000.00000001.00000101
subnet mask : 11111111.11111111.11111111.00000000
network id  : 11000000.10101000.00000001.00000000
host id     : 00000000.00000000.00000000.00000101
```

Nog een tweede voorbeeld: **192.168.199.233/24**.

```
ip-adres    : 192.168.199.233
subnet mask : 255.255.255.0
network id  : 192.168.199.0
host id     :             233
```

We kunnen dit tweede voorbeeld ook binair bekijken.

```
ip-adres    : 11000000.10101000.11000111.11101001
subnet mask : 11111111.11111111.11111111.00000000
network id  : 11000000.10101000.11000111.00000000
host id     : 00000000.00000000.00000000.11101001
```

Als derde voorbeeld zullen we hetzelfde ip-adres nemen als in het tweede, maar met een supernet: **192.168.199.233/22**. We bekijken het eerst binair, want dit is het eenvoudigste.

```
ip-adres    : 11000000.10101000.11000111.11101001
subnet mask : 11111111.11111111.11111100.00000000
network id  : 11000000.10101000.11000100.00000000
host id     : 00000000.00000000.00000011.11101001
```

Decimaal wordt dat dan:

```
ip-adres    : 192.168.199.233
subnet mask : 255.255.252.0
network id  : 192.168.196.0
host id     :           3.233
```

10.7. voorbeeldoefening binaire subnets

De vraag "Zitten de volgende computers in hetzelfde netwerk?" was met classful subnets niet zo moeilijk. Deze vraag komt bijna letterlijk terug op het examen, maar dan met binaire subnet masks.

Vul de volgende tabel aan voor **192.168.234.234/17**.

Table 10.10. oefening 192.168.234.234/17

	binair	**decimaal**
ip address	11000000.10101000.11101010.11101010	192.168.234.234
subnet mask	11111111.11111111.10000000.00000000	255.255.128.0
network id		
eerste ip		
laatste ip		
broadcast ip		
aantal ip's		

Hieronder vind je de oplossing van bovenstaande oefening.

Table 10.11. oplossing 192.168.234.234/17

	binair	**decimaal**
ip address	11000000.10101000.11101010.11101010	192.168.234.234
subnet mask	11111111.11111111.10000000.00000000	255.255.128.0
network id	11000000.10101000.10000000.00000000	192.168.128.0
eerste ip	11000000.10101000.10000000.00000001	192.168.128.1
laatste ip	11000000.10101000.11111111.11111110	192.168.255.254
broadcast ip	11000000.10101000.11111111.11111111	192.168.255.255
aantal ip's	van 0000000.00000001 tot 1111111.11111110	128*256-2=32766

Het bovenstaande netwerk bevat exact de helft van alle ip-adressen in de 192.168.0.0/16 reeks. Kan je de tabel ook invullen voor de andere helft ?

Table 10.12. andere helft van 192.168.234.234/17

	binair	**decimaal**
ip address		
subnet mask	11111111.11111111.10000000.00000000	255.255.128.0
network id		
eerste ip		
laatste ip		
broadcast ip		
aantal ip's		

Hieronder de oplossing van de andere helft.

Table 10.13. oplossing andere helft 192.168.234.234/17

	binair	**decimaal**
ip address	11000000.10101000.**0**1101010.11101010	192.168.106.234
subnet mask	11111111.11111111.10000000.00000000	255.255.128.0
network id	11000000.10101000.**0**0000000.00000000	192.168.0.0
eerste ip	11000000.10101000.**0**0000000.00000001	192.168.0.1
laatste ip	11000000.10101000.**0**1111111.11111110	192.168.127.254
broadcast ip	11000000.10101000.**0**1111111.11111111	192.168.127.255
aantal ip's	van 0000000.00000001 tot 1111111.11111110	128*256-2=32766

10.8. oefeningen binaire subnets

Probeer nu dezelfde oefening voor:

```
168.186.240.192/11
192.168.248.234/17
168.190.248.199/27
```

Kan je de network id, subnet mask, eerste ip, laatste ip, broadcast ip en aantal ip's geven voor de subnets van die drie ip-adressen ? Zonder naar de oplossing hieronder te kijken ?

Hieronder eerst drie lege tabellen om te oefenen, dan de oplossing.

Table 10.14. lege tabel 168.186.240.192/11

	binair	decimaal
ip address		
subnet mask		
network id		
eerste ip		
laatste ip		
broadcast ip		
aantal ip's		

Table 10.15. lege tabel 192.168.248.234/17

	binair	decimaal
ip address		
subnet mask		
network id		
eerste ip		
laatste ip		
broadcast ip		
aantal ip's		

Table 10.16. lege tabel 168.190.248.199/27

	binair	decimaal
ip address		
subnet mask		
network id		
eerste ip		
laatste ip		
broadcast ip		
aantal ip's		

Table 10.17. oplossing 168.186.240.192/11

	binair	decimaal
ip address	10101000.10111010.11110000.11000000	168.186.240.192
subnet mask	**11111111.111**00000.00000000.00000000	255.224.0.0
network id	**10101000.101**00000.00000000.00000000	168.160.0.0
eerste ip	**10101000.101**00000.00000000.0000000**1**	168.160.0.1
laatste ip	**10101000.101**11111.11111111.11111110	168.191.255.254
broadcast ip	**10101000.101**11111.11111111.11111111	168.191.255.255
aantal ip's	van 00000.00000000.00000001 tot 11111.11111111.11111110	32*256*256-2

Table 10.18. oplossing 192.168.248.234/17

	binair	decimaal
ip address	11000000.10101000.11111000.11101010	192.168.248.234
subnet mask	**11111111.11111111.1**0000000.00000000	255.255.128.0
network id	**11000000.10101000.1**0000000.00000000	192.168.128.0
eerste ip	**11000000.10101000.1**0000000.0000000**1**	192.168.128.1
laatste ip	**11000000.10101000.1**1111111.11111110	192.168.255.254
broadcast ip	**11000000.10101000.1**1111111.11111111	192.168.255.255
aantal ip's	van 0000000.00000001 tot 1111111.11111110	128*256-2

Table 10.19. oplossing 168.190.248.199/27

	binair	decimaal
ip address	10101000.10111110.11111000.11000111	168.190.248.199
subnet mask	**11111111.11111111.11111111.111**00000	255.255.255.224
network id	**10101000.10111110.11111000.110**00000	168.190.248.192
eerste ip	**10101000.10111110.11111000.110**00001	168.190.248.193
laatste ip	**10101000.10111110.11111000.110**11110	168.190.248.222
broadcast ip	**10101000.10111110.11111000.110**11111	168.190.248.223
aantal ip's	van 00001 tot 11110	32-2

10.9. zelfde of ander netwerk ?

Zitten de computers 192.168.117.5/18 en 192.168.34.18/18 in hetzelfde netwerk ?

10.10. subnetworks

Je beheert een departement met 200 computers, verdeeld over vier verdiepingen in eenzelfde gebouw, met ongeveer vijftig computers per verdieping. Je krijgt van je netwerkbeheerder **192.168.5.0/24** en moet deze reeks verdelen over de vier verdiepen. Hoe doe je dat ?

Je begint met uit te rekenen hoeveel computers er in die reeks kunnen:

```
192.168.5.0/24 --> 256 - 2 = 254 computers
```

Je weet ook uit de tabel hierboven dat een **/26 subnet mask** volstaat voor elke individuele verdieping. Je verdeelt jouw /24 netwerk in vier /26 netwerken.

We bekijken dit eerst binair:

```
192.168.5.0 == 11000000.10101000.00000101.00000000
/24 mask    == 11111111.11111111.11111111.00000000
/26 mask    == 11111111.11111111.11111111.11000000
```

We moeten dus twee **bits** toevoegen aan de /24 **netword id** om een /26 network id te maken. Twee bits kunnen exact vier mogelijke waarden hebben: 00, 01, 10 of 11. We komen dus tot de volgende vier nieuwe /26 network id's.

```
/26 network id 1 == 11000000.10101000.00000101.00000000
/26 network id 2 == 11000000.10101000.00000101.01000000
/26 network id 3 == 11000000.10101000.00000101.10000000
/26 network id 4 == 11000000.10101000.00000101.11000000
```

Hieronder een tabel van eerste en laatste ip van de vier /26 netwerken. Tel bij de laatste ip eentje bij voor de broadcast ip.

Table 10.20. echt supernetten

verdieping	eerste ip	laatste ip
1	192.168.5.1	192.168.5.62
2	192.168.5.65	192.168.5.126
3	192.168.5.129	192.168.5.190
4	192.168.5.193	192.168.5.254

Tussen haakjes, nu zijn we echt aan het **supernetten**. Want in de routers van buitenaf staat er enkel 192.168.5.0/24 als bestemming, terwijl er intern eigenlijk vier /26 netwerkjes zijn.

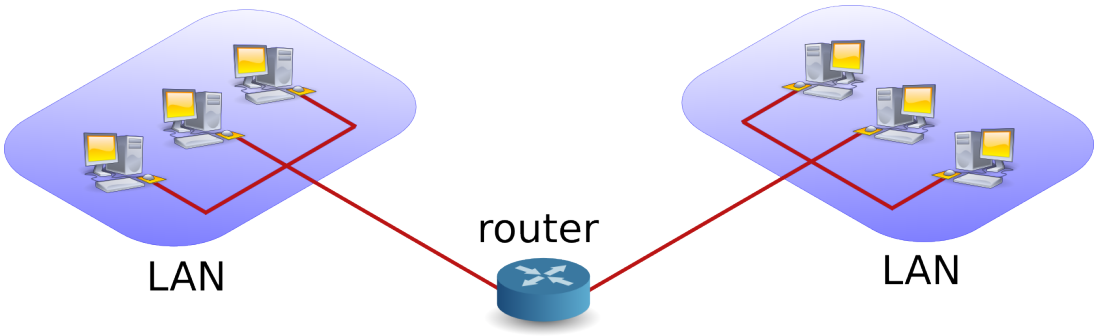
Chapter 11. inleiding tot routers

Table of Contents

Routers zijn fundamentele onderdelen van een netwerk en van het internet. Dit hoofdstuk is slechts een inleiding.

11.1. router tussen twee netwerken

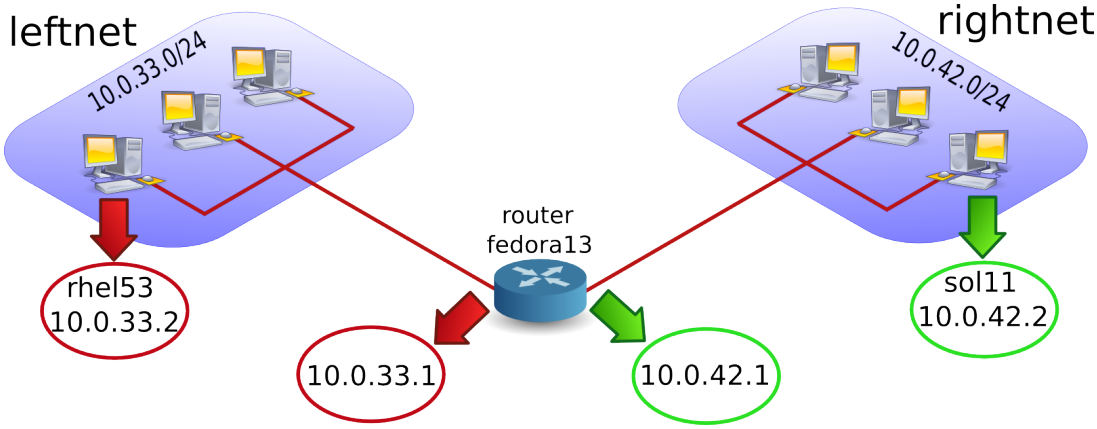
Een **router** verbindt twee netwerken.



Om iets te laten zien over **routers** heb je dus een **router** nodig en minstens twee computers, eentje aan elke kant van de **router**. Hier volgen twee voorbeelden, gedaan met vrije software in **Virtualbox**.

11.1.1. de opstelling

We beginnen met twee netwerken te bouwen, eentje links in de range **10.0.33.0/24** en eentje rechts in de range **10.0.42.0/24**. Het linkse netwerk wordt **leftnet** genoemd, het andere **rightnet**.



In het linkse netwerk zetten we een Red Hat Enterprise Linux (rhel53) server met ip-adres 10.0.33.2/24. Linux noemt de lokale netwerkkaart **eth0**. Het volgende commando configureert het juiste ip-adres en subnet mask:

```
ifconfig eth0 10.0.33.2 netmask 255.255.255.0
```

Het mac-adres (of hardware address) van deze netwerkkaart is 08:00:27:F6:67:53.

In het rechtse netwerk zetten we een Oracle Solaris 11 (sol11) server met ip-adres 10.0.42.2/24. Solaris noemt de lokale netwerkkaart **e1000g0**. Het volgende commando configureert het juiste ip-adres en subnet mask:

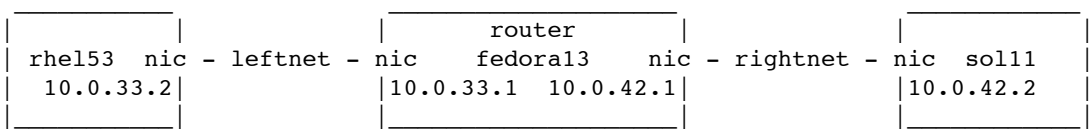
```
ifconfig e1000g0 10.0.42.2 netmask 255.255.255.0
```

Het mac-adres (of ethernet address) van deze netwerkkaart is 8:0:27:5a:16:76.

Tussen de twee netwerken staat een Fedora 13 (fed13) server die dienst doet als router. Deze router heeft twee netwerkkaarten, eentje verbonden met **leftnet** en eentje verbonden met **rightnet**. Het commando om beide netwerkkaarten te configureren is:

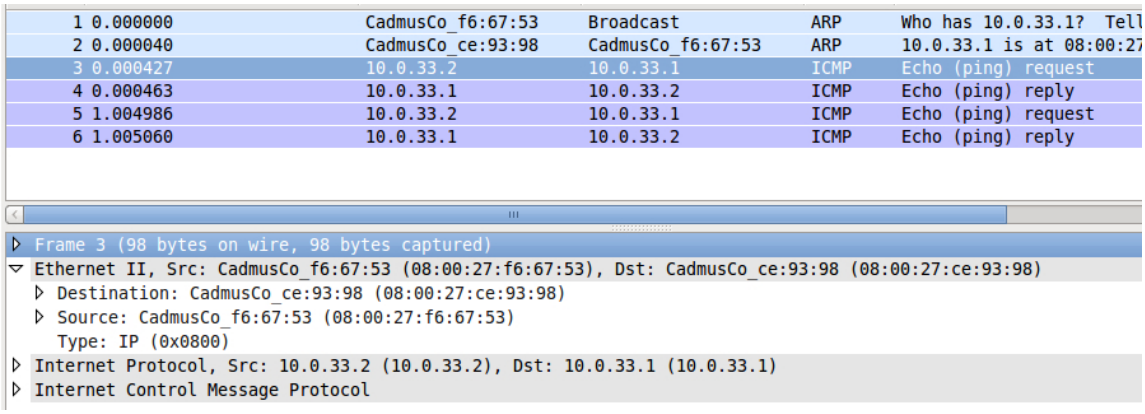
```
ifconfig eth0 10.0.33.1 netmask 255.255.255.0
ifconfig eth1 10.0.42.1 netmask 255.255.255.0
```

Het mac-adres van **eth0** is hier 08:00:27:CE:93:98 en van **eth1** is het 08:00:27:E8:BE:68.



11.1.2. ping op leftnet

We controleren even dat **wireshark** goed werkt door een **ping** te doen van de **rhel53** naar 10.0.33.1. We testen hiermee dat de verbinding tussen de links rhel53 computer en de fedora13 router werkt. Dit is noodzakelijk opdat de computer de router kan gebruiken. Een router moet in hetzelfde netwerk staan als de computer.



Dit is een lokaal netwerk dus we zien twee gekende **arp** packetjes. De ping wordt met de correcte mac-adressen op het netwerk gezet.

11.1.3. standaard gateway instellen

We configureren nu de computers uit **leftnet** en **rightnet** om elk de **router** te gebruiken. Elke computer ziet een andere kant van de router.

Op **sol11** geven we 10.0.42.1 op als adres van de router:

```
route add default 10.0.42.1
```

Op **rhel53** geven we 10.0.33.1 op als adres van de router:

```
route add default gw 10.0.33.1
```

Tussen haakjes, op **MS Windows** kan je dit door via Start - Instellingen - Controle Paneel - Netwerkbeheer - Eigenschappen - tcp/ip - Eigenschappen ergens de **standaard gateway** in te vullen en de laatste twee geopende vensters te sluiten.

11.1.4. ping over de router?

Als we echter een ping proberen van het ene naar het andere netwerk, dan krijgen we geen antwoord want de netwerken zijn gescheiden en de router staat nog niet aan.

Source	Destination	Protocol	Info
CadmusCo_f6:67:53	Broadcast	ARP	Who has 10.0.33.1?
CadmusCo_ce:93:98	CadmusCo_f6:67:53	ARP	10.0.33.1 is at 08:0
10.0.33.2	10.0.42.2	ICMP	Echo (ping) request
10.0.33.2	10.0.42.2	ICMP	Echo (ping) request
10.0.33.2	10.0.42.2	ICMP	Echo (ping) request
10.0.33.2	10.0.42.2	ICMP	Echo (ping) request
10.0.33.2	10.0.42.2	ICMP	Echo (ping) request
10.0.33.2	10.0.42.2	ICMP	Echo (ping) request

Het is niet omdat een computer twee netwerkkaarten heeft, dat die automatisch router gaat spelen.

11.1.5. routerfunctie aanzetten

We zetten de router aan door 1 bit aan te passen in Linux, die zal dan automatisch alle packetjes routen over al zijn netwerkkaarten. Dit is handig als je over slechts één router beschikt die al je netwerken verbindt. We zullen later zien dat er meer werk is om twee routers te laten samenwerken.

11.1.6. ping over de router!

Het packetje dat vertrekt bij 10.0.42.2 komt eerst (via **rightnet**) aan bij 10.0.42.1. Dat zie je in de screenshot van **wireshark** hieronder.

No. .	Time	Source	Destination	Protocol	Info
1	0.000000	10.0.42.2	10.0.33.2	ICMP	Echo (ping) request
2	0.004927	10.0.33.2	10.0.42.2	ICMP	Echo (ping) reply

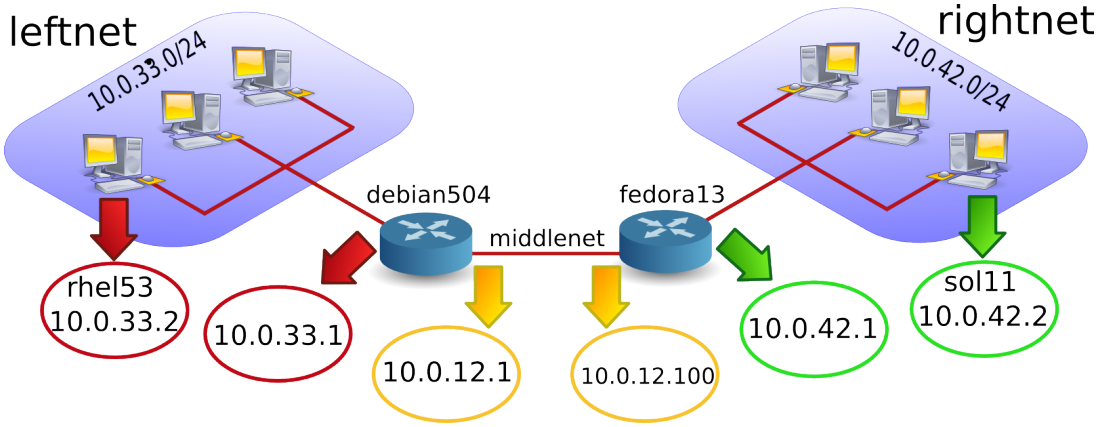
```
▷ Frame 1 (98 bytes on wire, 98 bytes captured)
▷ Ethernet II, Src: CadmusCo_5a:16:76 (08:00:27:5a:16:76), Dst: CadmusCo_e8:be:68 (08:00:27:e8:be:68)
▷ Internet Protocol, Src: 10.0.42.2 (10.0.42.2), Dst: 10.0.33.2 (10.0.33.2)
▷ Internet Control Message Protocol
```

Vervolgens gaat de router het destination-mac-adres aanpassen naar het mac-adres van 10.0.33.2, en terzelfdertijd het source-mac-adres vervangen door dat van zijn eigen 10.0.33.1 interface. Zo komt het packetje (via **leftnet**) aan bij de bestemming.

We hebben nu een werkende router tussen onze twee netwerken.

11.2. twee routers met elkaar verbinden

11.2.1. opstelling met vier machines



We voegen een **Debian 5.04** server als router toe aan onze opstelling en verbinden beide routers via **middlenet** (en de nieuwe reeks 10.0.12.0/24). Er staan nu twee routers tussen de 10.0.42.0/24 en 10.0.33.0/24 netwerken.

```
                                                                    r
l n    |       router        |              |       router        |   i n
e e -  nic     deb504     nic - middlenet - nic    fedora13     nic - g e
f t    |10.0.33.1  10.0.12.1|              |10.0.12.100 10.0.42.1|   h t
t      |_____________________|              |_____________________|   t
```

De configuratie van de twee gewone computers moet niet aangepast worden, zij behouden (wat ip-adres betreft) dezelfde **default gateway**.

11.2.2. rip

Het **routing interface protocol** laat toe dat routers hun eigen **routing table** opsturen naar nabije routers.

Hieronder kan je zien hoe de fedore13 router via 10.0.12.100 aan de andere router vertelt dat hij direct (met een metric van 1) verbonden is met het 10.0.42.0/24 netwerk.

No. .	Time	Source	Destination	Protocol	Info
13	293.487416	10.0.12.100	10.0.12.1	RIPv2	Response

```
▷ Frame 13 (66 bytes on wire, 66 bytes captured)
▷ Ethernet II, Src: CadmusCo_ce:93:98 (08:00:27:ce:93:98), Dst: CadmusCo_5d:1f:6c (08:00:27:5d:1f:6c)
▷ Internet Protocol, Src: 10.0.12.100 (10.0.12.100), Dst: 10.0.12.1 (10.0.12.1)
▷ User Datagram Protocol, Src Port: router (520), Dst Port: router (520)
▽ Routing Information Protocol
    Command: Response (2)
    Version: RIPv2 (2)
    Routing Domain: 0
  ▷ IP Address: 10.0.42.0, Metric: 1
```

De **metric** in de **routing table** kan je zien als een **kost**. Een router zal steeds proberen om packetjes te leveren via de goedkoopste route.

11.3. lokale routing tables

Elke computer met een ip-adres heeft intern een **routing table**. Op Windows kan je deze zien met **route print**, op Unix met **netstat**.

Onderstaande tabel is simpel omdat er maar 1 netwerkkaart in de computer een ip-adres heeft. Alle trafiek voor dat netwerk, gaat via die netwerkkaart (genaamd eth2). Alle andere trafiek wordt via die netwerkkaart naar de router (192.168.1.1) gestuurd.

```
[root@fed13 ~]# netstat -nr
Kernel IP routing table
Destination   Gateway       Genmask         Flags   MSS Window  irtt Iface
192.168.1.0   0.0.0.0       255.255.255.0   U         0 0          0 eth2
0.0.0.0       192.168.1.1   0.0.0.0         UG        0 0          0 eth2
```

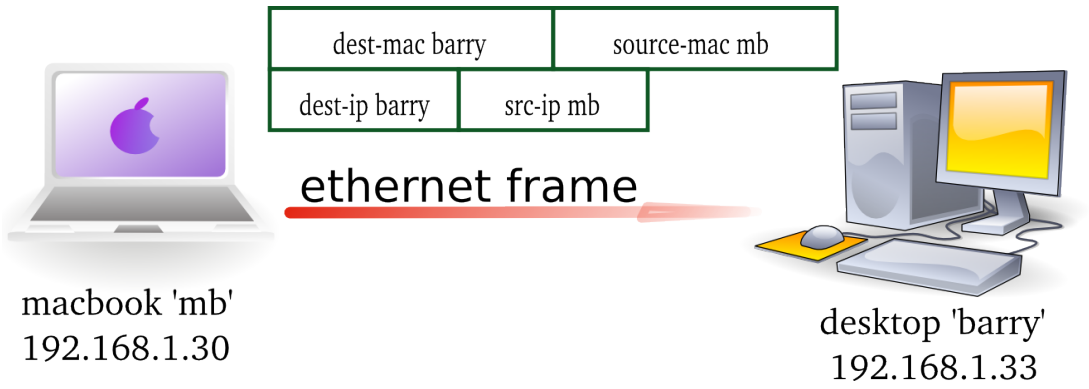
Onderstaande tabel is uitgebreider omdat deze computer nu drie netwerkkaarten heeft met elk een ip-adres in een ander netwerk. Trafiek dat niet voor een van deze netwerken is, wordt nog steeds naar 192.168.1.1 gestuurd via eth2.

```
[root@fed13 ~]# netstat -nr
Kernel IP routing table
Destination   Gateway       Genmask         Flags   MSS Window  irtt Iface
10.0.33.0     0.0.0.0       255.255.255.0   U         0 0          0 eth0
192.168.1.0   0.0.0.0       255.255.255.0   U         0 0          0 eth2
10.0.42.0     0.0.0.0       255.255.255.0   U         0 0          0 eth1
0.0.0.0       192.168.1.1   0.0.0.0         UG        0 0          0 eth2
```

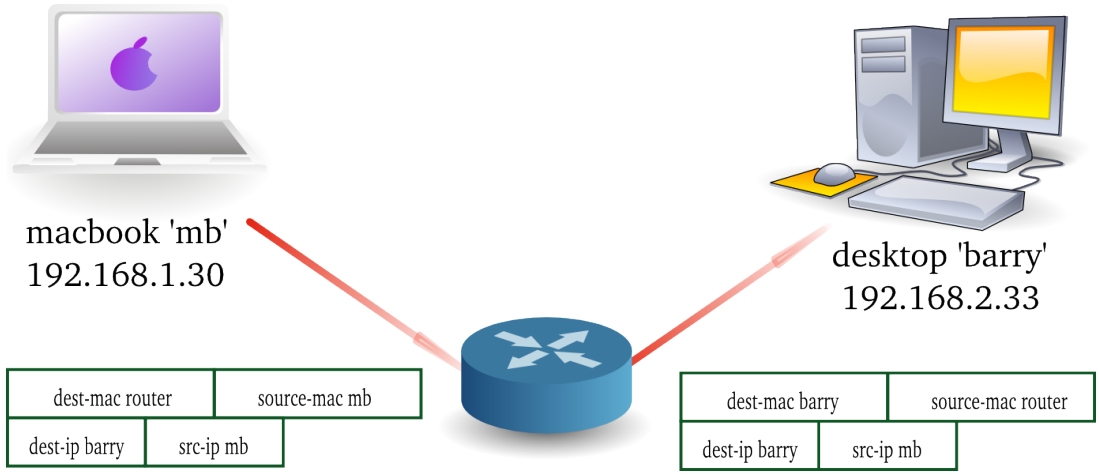
11.4. nat

We hebben op het bord getekend hoe **nat** samenwerkt met **pat** om private adressen om te zetten in publieke, en omgekeerd.

Hieronder een tekening van een packetje in een netwerk zonder router.



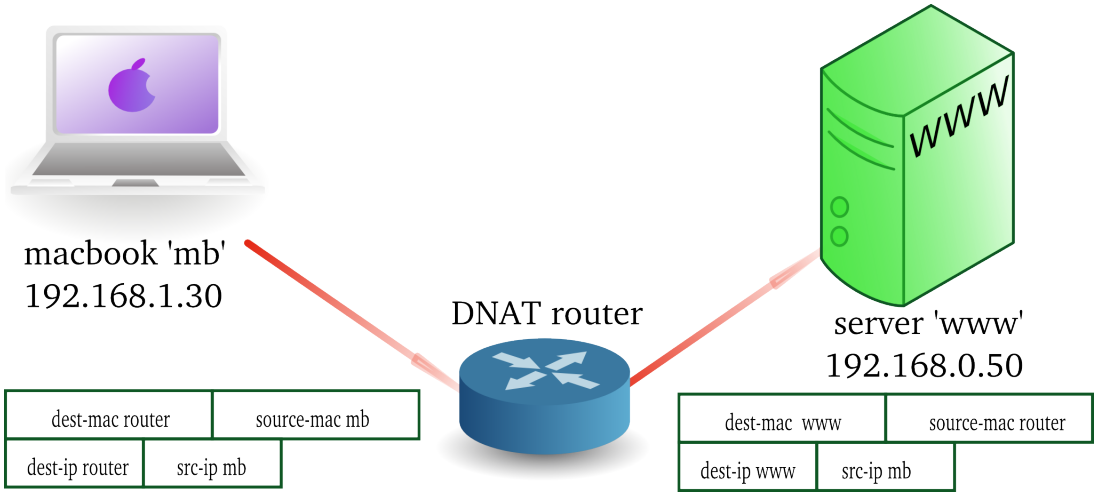
Hieronder een tekening van een packetje dat door een eenvoudige router gaat. Zoals je ziet, worden de mac-adressen aangepast.



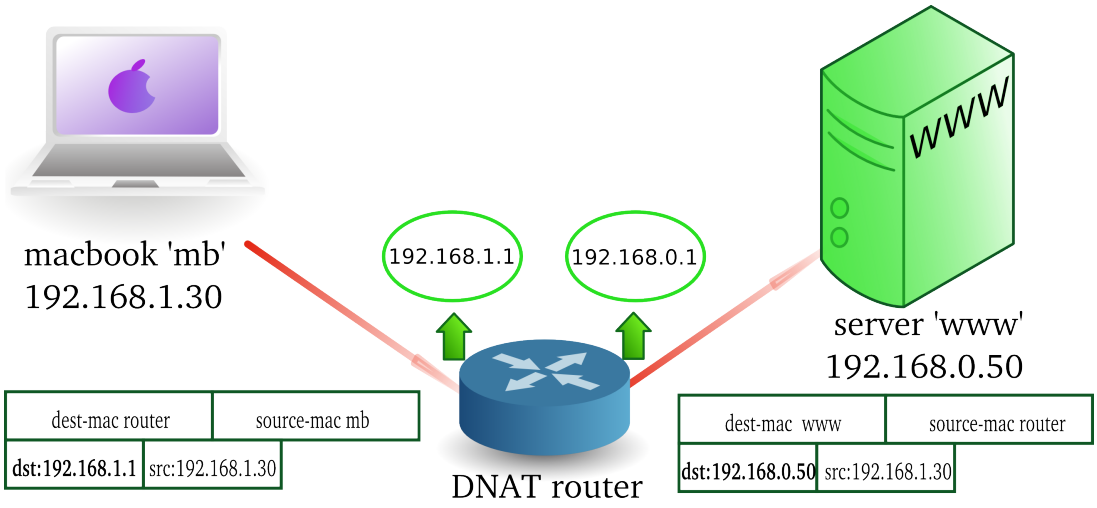
11.5. dnat

We spreken van **destination nat** wanneer de router het **bestemmings**-ip-adres vertaalt naar een ander ip-adres.

We beginnen met een simpel voorbeeld intern in een organisatie. Dit is een tekening van een packetje dat door een **dnat router** gaat. Zoals je ziet, wordt naast de mac-adressen, nu ook het **bestemmings**-ip-adres aangepast. De 'mb' laptop ziet dus het ip-adres van de 'router' als bestemming, terwijl de 'www' server achter de router staat.



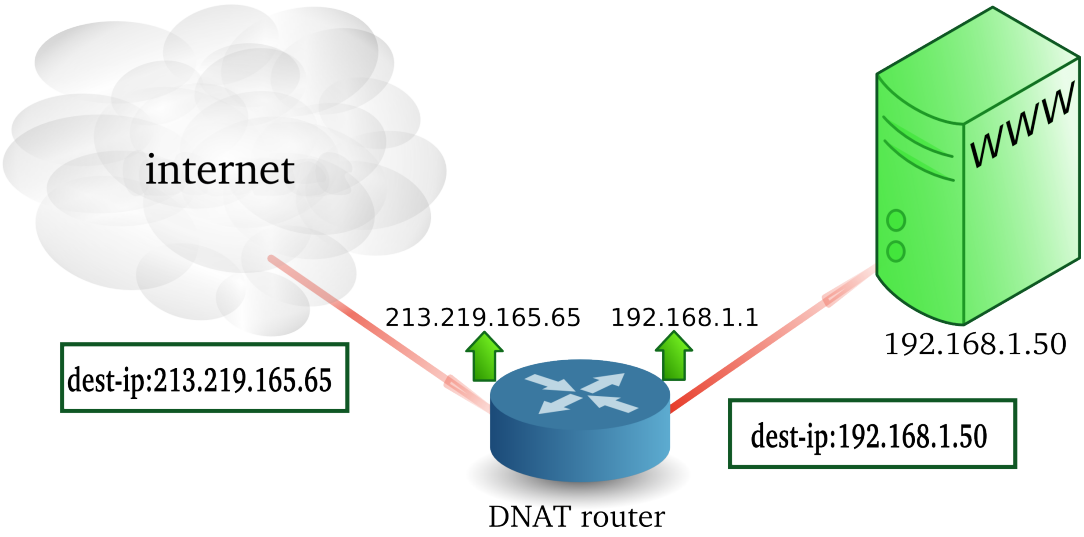
Hieronder hetzelfde voorbeeld, hetzelfde scenario, maar met ip-adressen ingevuld. Het **bestemmings**-ip-adres staat vet gedrukt omdat dit de kern is van een **dnat** router.



11.6. dnat en internet

dnat wordt meestal gebruikt om een publiek adres te vertalen naar een intern en privaat adres. Het is uiteraard goed mogelijk dat een server met een publiek ip-adres rechtstreeks met internet verbonden is, maar het is ook mogelijk om dit publieke adres toe te kennen aan een **dnat router** (die eventueel ook firewall is) en de eigenlijke server in een private ip-range te zetten.

Deze tekening toont dat op het hele internet het publieke adres bekend is, terwijl eens achter de **dnat router** een privaat adres wordt gebruikt.



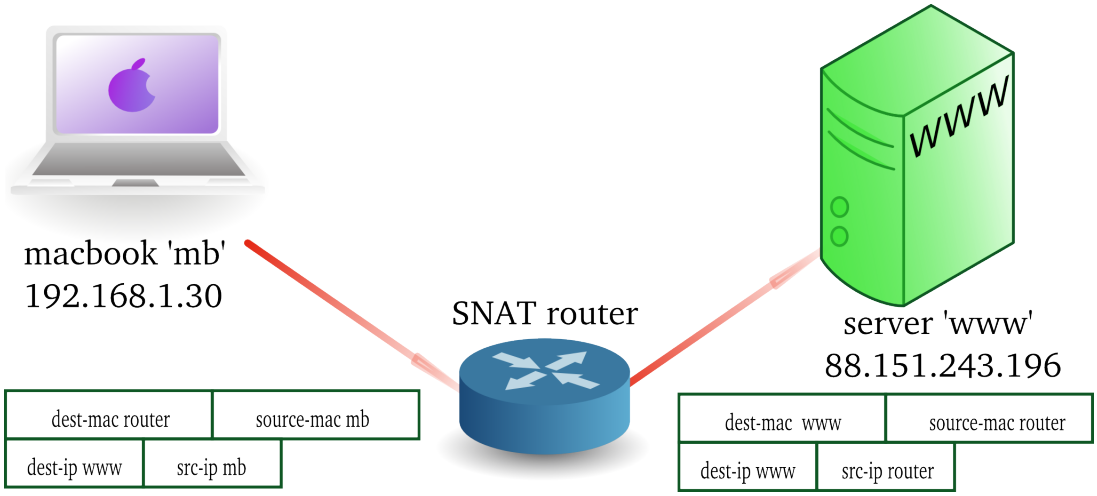
11.7. port forwarding en pat

We hebben beide in de klas besproken.

11.8. snat

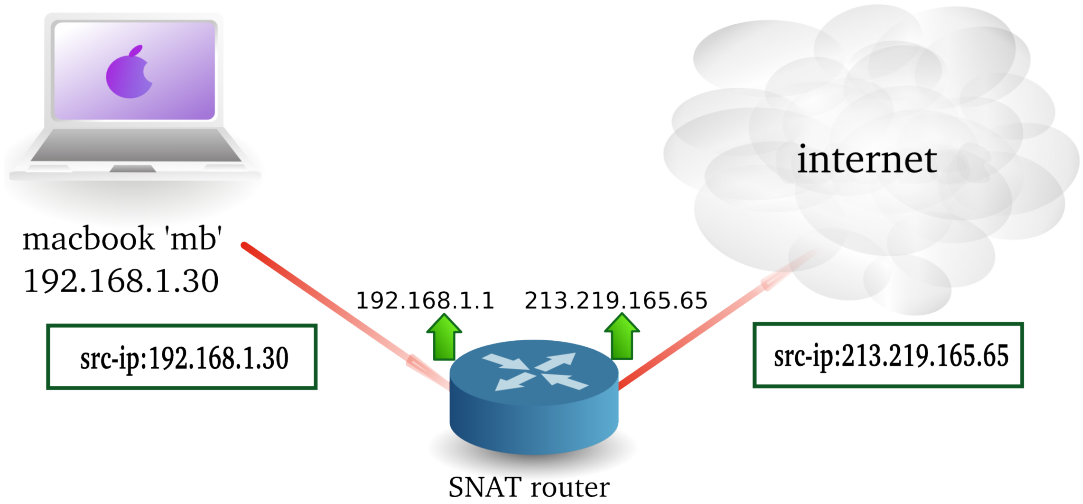
We spreken van **source nat** wanneer de router het **bron**-ip-adres aanpast.

Hieronder een tekening van een packetje dat door een **snat router** gaat. Zoals je ziet, wordt behalve de mac-adressen, nu ook het **bron**-ip-adres aangepast. De 'www' server ziet dus het ip-adres van de 'router' als bron.



11.9. snat en internet

snat wordt meestal gebruikt om onze interne private adressen om te zetten in een publiek adres dat op internet gezien wordt. Onze private adressen worden verborgen voor het internet.



We spreken van **masquerading** wanneer **snat** wordt toegepast op een dynamisch publiek ip-adres. Dit is het geval bij de meeste **adsl** verbindingen in Vlaanderen.

Chapter 12. License

GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles

are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either

commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   * A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

* B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
* C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
* D. Preserve all the copyright notices of the Document.
* E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
* F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
* G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
* H. Include an unaltered copy of this License.
* I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
* J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
* K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
* L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
* M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
* N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
* O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of,

you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies

that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

Index

Symbols

A

B

C

D

E

F

G

R

S

T

U

V

W

X

Z