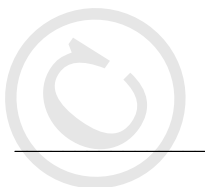


Student-notities
Linux/UNIX TCP/IP-netwerken
Voorbeeld-hoofdstuk

04. Address resolution

 **at computing**
The Linux/UNIXperts

Nijmegen



Student-notities

- De student-notities in dit voorbeeld-hoofdstuk zijn fragmenten uit het dictaat dat bij deze cursus wordt meegeleverd. □

Als node A een datagram wil sturen naar een node B (we nemen aan dat A en B zich op hetzelfde fysieke netwerk bevinden), vult de IP-laag van A in het datagram het IP-adres van B in. Op het data-link niveau echter zal, om het frame waarin het datagram verpakt wordt te kunnen versturen, het fysieke adres van B bekend moeten zijn. De Ethernet hardware bijvoorbeeld herkent geen IP-adressen, maar slechts Ethernet-adressen. De vraag rijst dus op welke manier A uit het IP-adres van B het fysieke adres van B kan bepalen.

In principe zou het een mogelijkheid zijn in elke node een tabel bij te houden waarin de afbeelding tussen IP-adres en fysiek adres voor alle nodes vastgelegd is. Deze organisatie heeft echter enkele voor de hand liggende nadelen vanwege het statische karakter ervan:

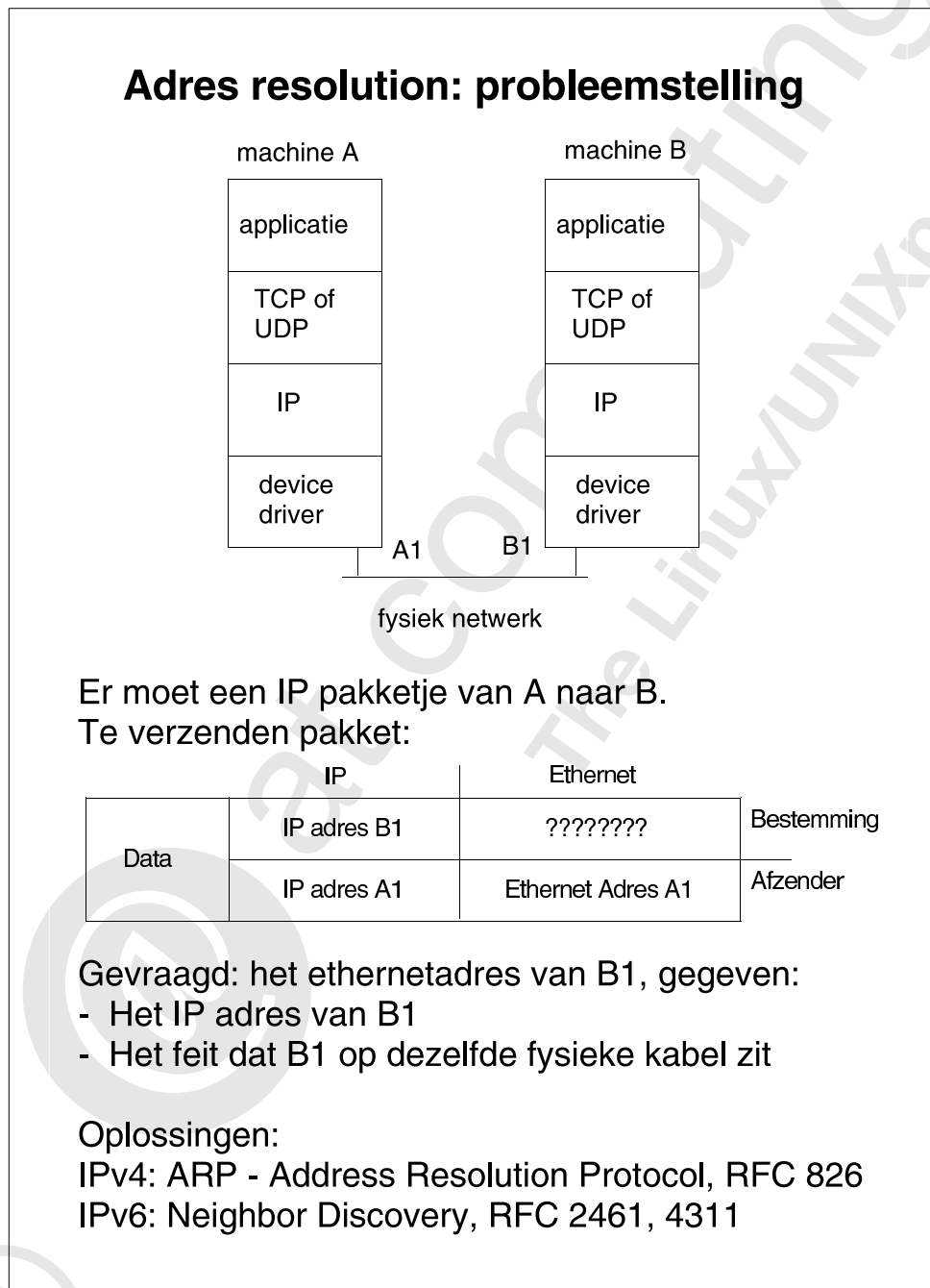
- a. Als in een node een kapotte Ethernet-controller vervangen wordt, kloppen de tabellen in de andere machines niet meer.
- b. Het bijhouden van één en dezelfde tabel in alle machines is vervelend en geeft gemakkelijk aanleiding tot fouten.

Voor sommige fysieke netwerken, zoals het proNET-10 netwerk, is het mogelijk het fysieke netwerkadres te coderen in het node gedeelte van het IP-adres. In het geval van proNET-10 is dat mogelijk doordat fysieke proNET-10 adressen kleine getallen zijn. De adresvertaling van IP-adres naar fysiek adres wordt dan dus triviaal.

Voor andere fysieke netwerken is een dergelijke codering niet mogelijk. Ethernet-adressen bijvoorbeeld kosten 48 bits, en kunnen duidelijk niet opgenomen worden in 32-bits IPv4-adressen. Bij IPv6-adressen zou dit wel kunnen, maar als het al gedaan wordt, wordt daar toch geen gebruik van gemaakt bij het bepalen van het bijbehorende MAC-adres. Voor netwerken als Ethernet moet dus een andere oplossing gevonden worden voor het zogenaamde *adresresolutie probleem*.

IPv4 en IPv6 gebruiken voor het adresresolutie probleem verschillende technieken. IPv4 gebruikt ARP, IPv6 gebruikt ICMP-berichten. De gedachte achter beide technieken is echter min of meer gelijk: je stuurt een broadcast of multicast bericht de link op met daarin het IP-adres van de node waar je contact mee op wilt nemen. De node met dat IP-adres zal antwoorden met zijn hardware-adres. Om te voorkomen dat het sturen van het antwoord dezelfde problemen ondervindt, stuurt de verzender zijn eigen hardware-adres alvast mee.





Figuur 1.

Student-notities

Het ARP-protocol werkt globaal als volgt: als node A het IP-adres van node B kent, maar het fysieke adres van B niet, stuurt A een ARP-pakket verpakt als broadcast frame het fysieke netwerk op. Dit ARP-pakket bevat het IP-adres van B als data. Alle machines op het fysieke netwerk ontvangen dit broadcast frame, en bekijken het IP-adres dat onderdeel uitmaakt van het ARP-pakket. Slechts voor één machine (namelijk machine B) geldt dat dit adres gelijk is aan het eigen IP-adres. Deze machine stuurt een ARP-pakket terug met daarin het gevraagde fysieke adres.

Een aardige vraag is nu natuurlijk: hoe kan node B iets terugsturen naar node A? Immers, daar is het fysieke adres van machine A voor nodig. Het lijkt erop dat machine B dus ook ARP moet doen, en zo komen we niet verder. Om deze cirkel te doorbreken wordt als extra informatie bij het versturen van een ARP broadcast pakket het lokale IP-adres én het lokale fysieke adres opgenomen. Node B krijgt op deze manier dus direct het IP-adres en het fysieke adres van node A aangereikt.

ARP (IPv4)

ARP is altijd aanwezig bij gebruik van de combinatie IPv4 en Ethernet.

Werking:

- ARP zit op laag 3 (net als IP) en heeft een eigen type nummertje in de ethernet header
- ARP gebruikt dezelfde adressen als IP

	ARP	Ethernet	
Who has B1? Tell A1	IP adres B1	Broadcast	Bestemming
	IP adres A1	Ethernet Adres A1	Afzender

De ARP software in B pikt dit op herkent het IP adres en antwoordt:

	ARP	Ethernet	
B1(IP) is at. B1(ether)	IP adres A1	Ethernet Adres A1	Bestemming
	IP adres B1	Ethernet Adres B1	Afzender

De ARP software op A ontvangt dit en kan het oorspronkelijke pakket versturen

A en B cachen de antwoorden in de kernel.

Commando: **arp -a [n]**

Als A geen antwoord krijgt:

ICMP Destination Unreachable

Figuur 2.

Student-notities

Gezien het feit dat ARP gebruik maakt van broadcasting op het fysieke netwerk vergt het ARP-protocol een fysiek netwerk dat een broadcast faciliteit kent. Daar een broadcast een dure operatie is (elke machine op het netwerk moet het pakket verwerken) moeten ARP broadcast pakketten zoveel mogelijk vermeden worden. Temeer daar alle machines behalve één het pakket negeren. Er is een optimalisatie in het ARP-protocol die hiervoor bedoeld is: elke node houdt een *ARP-cache* bij, waarin recentelijk ontdekte paren

(IP-adres, fysiek adres)

opgeslagen staan. Wanneer een eerder ontdekt paar opnieuw nodig is, staat het nog in de cache en kan een broadcast achterwege blijven. Uiteraard mag niet te lang op oude informatie vertrouwd worden (omdat anders het gevaar van verouderde paren dreigt). Daarom moeten entries in de cache een beperkte levensduur hebben.

Merk overigens op dat niet alleen de machine die antwoord geeft op een ARP broadcast pakket het fysieke adres van de zender in de ARP-cache kan opslaan: alle machines kunnen er hun voordeel mee doen.

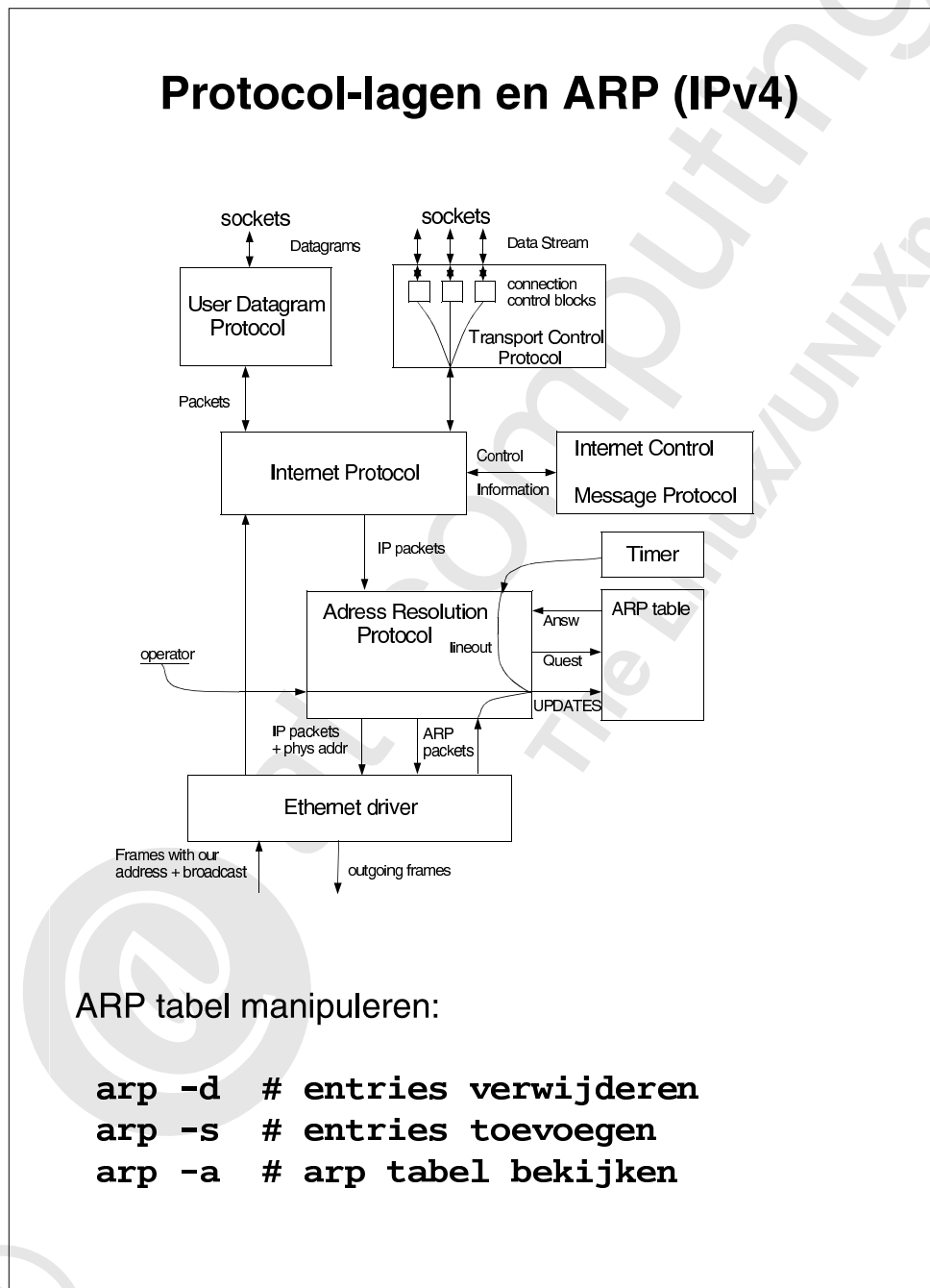
Met het commando `arp` kan de ARP cache geïnspecteerd en gemanipuleerd worden. Het ARP-protocol gaat niet van een bepaald fysiek netwerk uit, maar de meeste ARP implementaties en het `arp` commando gaan uit van Ethernet als fysiek medium.

<code>arp -a</code>	Toon alle entries in ARP cache.
<code>arp host</code>	Toon entry van host in ARP cache.
<code>arp -d host</code>	Verwijder entry van host in arp cache.
<code>arp -s host ether [temp] [pub]</code>	Voeg entry (host, ether) toe aan ARP cache.

Over het algemeen behoeft de ARP cache niet met de hand gemanipuleerd te worden; daar dient het ARP-protocol immers voor. Als een bepaalde correcte entry met het commando `arp -d` uit de cache verwijderd wordt, kan dat geen kwaad omdat het ARP-protocol die entry weer toe zal voegen op het moment dat de betreffende informatie nodig is.

Het argument `temp` bij het commando `arp -s` zorgt ervoor dat de opgegeven entry niet het eeuwige leven heeft: na een bepaalde tijd (in de orde van 5 minuten) wordt de entry weer uitgeoetst, om te vermijden dat teveel op oude informatie vertrouwd wordt.

Het argument `pub` bij het commando `arp -s host ether` zorgt ervoor dat de lokale machine als een *ARP server* voor host zal optreden: als er een ARP request voor host komt, zal de lokale machine antwoorden, terwijl dat normaal gesproken alleen door host zélf gedaan wordt.



Figuur 3.

Student-notities

In IPv6 is het adres resolution probleem met behulp van ICMP-berichten geïmplementeerd. Het hart van dit mechanisme wordt gevormd door het "Neighbor Solicitation" en het "Neighbor Advertisement" ICMP-bericht.

De beide ICMP-berichten hebben meer dan één functie. Eén van de functies is dezelfde als die van ARP: het koppelen van het IP-adres aan het link-layer adres. Een tweede functie is het controleren van de verbinding tussen twee nodes op dezelfde link (buren).



Adres resolution in IPv6

- Niet via ARP maar met ICMP!
- Neighbor Solicitation (NS) van A naar B
 - Type = 135
- bevat source link layer adres in ICMP-optie, vraagt om target link layer adres
- ook Target Adres nodig i.v.m. multicast (net als bij ARP ivm broadcast)

Figuur 4.

Student-notities

Voor het bepalen van het link-layer adres van een buur stuurt de node een neighbor solicitation naar het solicited host multicast-adres. In een optie wordt het eigen link-layer adres meegestuurd.



at computing
The Linux/UNIXperts

Solicited host multicast (IPv6)

Adres waarvan we het link-layer adres zoeken:

FE80::800:2007:9757

Bij ARP: broadcast

Bij IPv6 Neighbor Solicitation naar Solicited Node Multicast:

FF02::1:FF07:9757

Multicast Link scope Laatste drie bytes van IPv6-adres

Het IPv6-destination adres is dus *niet* zijn eigen adres!

Dat staat in Target Address en wordt in de Neighbor Advertisement geretourneerd.

De kans dat meer dan één host deze multicast "ziet" is gering.

(RFC 3513)

Figuur 5.

Student-notities

Het antwoord (een Neighbor Advertisement) ziet er bijna hetzelfde uit. De Neighbor Advertisement bevat als optie ook het link-layer adres. Dit dekt de ARP-achtige functionaliteit af.

Omdat we geen idee kunnen hebben van het MAC-adres van de ontvanger (we weten immers niet of, en zo ja door wie het adres in gebruik is) moeten we gebruik maken van een multicast. Niet naar het all nodes multicast-adres (want dan had men net zo goed broadcasts in IPv6 kunnen laten zitten), maar naar een specifiek "Solicited Host" multicast-adres. Een solicited host multicast wordt gevormd uit het prefix `FF02:0:0:0:0:1:FF00::/104` en de laatste drie bytes van het IPv6-adres van de ontvanger. De kans dat een andere node deze zelfde laatste drie bytes als adres heeft is erg klein. Alleen als de laatste drie bytes van het token gelijk zijn, wordt er ten onrechte een node geïnterrupteerd. Die node kan dan aan het target adres in de neighbor solicitation zien of het bericht daadwerkelijk voor hem bedoeld is.

Neighbor Advertisement (IPv6)

De Neighbor Advertisement is het antwoord op een Neighbor Solicitation (NS):

- Neighbor Advertisement (NA)
 - Type = 136
 - R, S, O bitjes
 - Verder identiek aan NS
- NA bevat target link layer adres in optie
- Het Target Adres geeft aan waar de vraag over ging
- S-bit: dit is een antwoord op een NS
- O-bit: gebruikt bij IPv6-mobility
- R-bit: de verzender is een router

Figuur 6.

Student-notities

Stel je hebt een router met twee interfaces. Op één van de links geldt een bepaalde reeks adressen. De andere link echter gebruikt een *gedeelte* van die reeks.

De techniek werkt als volgt:

Stel machine A op de ene link wil een pakket versturen naar machine E op de andere link. Dan zal host A een ARP-verzoek doen om het fysieke adres van E op te vragen. Dit ARP-verzoek zal machine E echter niet bereiken: ARP werkt immers via een broadcast op de lokale link. Een oplossing is om router D te laten antwoorden voor E: de router D reageert op het ARP-verzoek van A door *zijn eigen* fysieke adres te geven. Machine A meent antwoord gekregen te hebben van E en stuurt het pakket naar E, althans dat denkt hij: in feite wordt (op data-link niveau) het pakket naar D gestuurd. Deze zorgt er vervolgens voor dat het bij E aankomt.

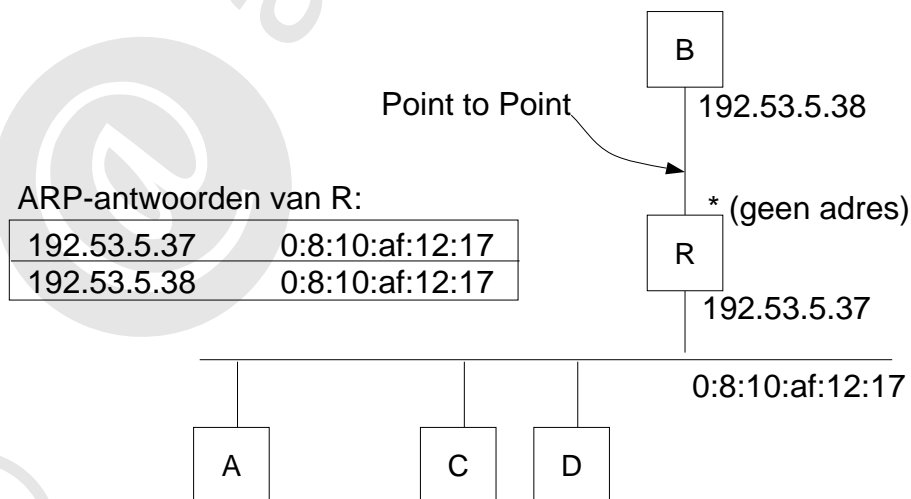
Wat er dus eigenlijk gebeurt bij proxy-ARP is dat een router ARP-verzoeken die bestemd zijn voor een host die niet op de betreffende link zit, afhandelt alsof het één van zijn eigen IP-adressen betreft.

Overigens moet deze techniek dan wel door de router ondersteund worden. De overige nodes merken er (bijna) niets van.

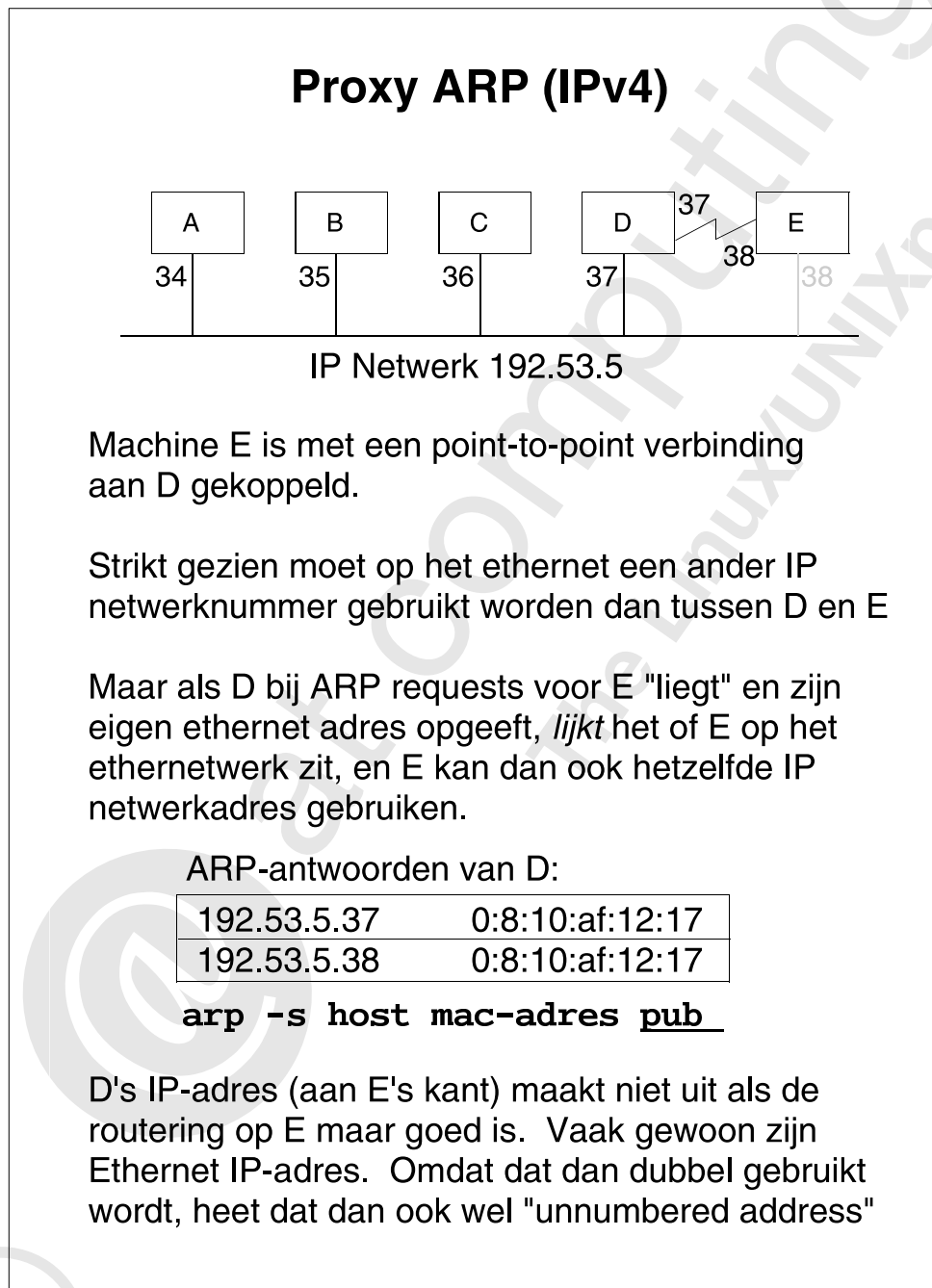
Deze techniek vindt vaak ook toepassing bij het gebruik van "Point-to-Point" protocollen.

Eén van de kanten kan zonder eigen IP-adres geconfigureerd worden, deze kant "leent" dan zijn IP-adres van een ander interface. De host achter het Point-to-Point netwerk gebruikt voor zijn kant een IP-adres dat eigenlijk bij het netwerk van dat andere interface hoort (unnumbered interface).

In een plaatje:



Als we de ARP-cache van node D bekijken dan zal achter de entry voor 192.53.5.38 vaak het woordje `pub` staan; het geeft aan dat de host ook voor deze host antwoorden zal op ARP-verzoeken (de eigen adressen zien we soms niet als we in de ARP-cache kijken).



Figuur 7.

