



Welcome to

# TCP/IP and Security in TCP/IP protocol suite

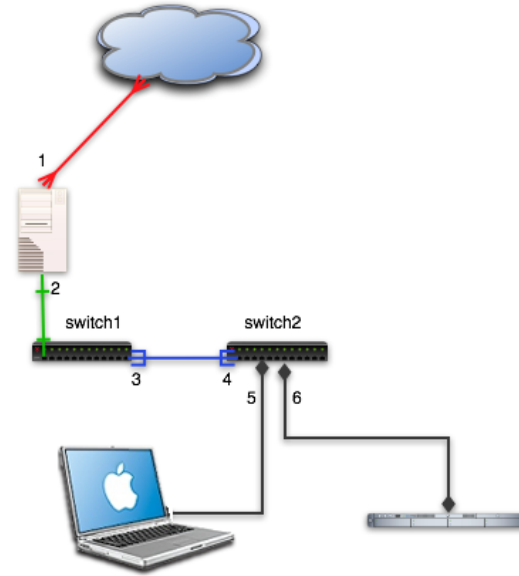
Communication and Network Security 2019

Henrik Lund Kramshøj [hlk@zencurity.com](mailto:hlk@zencurity.com)

Slides are available as PDF, [kramse@Github](mailto:kramse@Github)

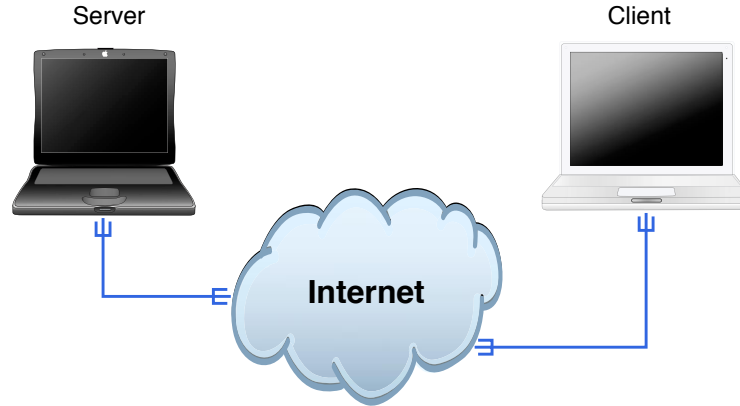
1-TCPIP-and-Security-in-TCPIP-protocol-suite.tex in the repo [security-courses](#)

# Course Network



Our network will be similar to regular networks, as found in enterprises  
We have an isolated network, allowing us to sniff and mess with hacking tools.

# Internet Today



Clients and servers, roots in the academic world

Protocols are old, some more than 20 years

Very little is encrypted, mostly HTTPS

# Internet er åbne standarder!



We reject kings, presidents, and voting.  
We believe in rough consensus and running code.  
– The IETF credo Dave Clark, 1992.

Request for comments - RFC - er en serie af dokumenter

RFC, BCP, FYI, informational

de første stammer tilbage fra 1969

Ændres ikke, men får status Obsoleted når der udkommer en nyere version af en standard

Standards track:

Proposed Standard → Draft Standard → Standard

Åbne standarder = åbenhed, ikke garanti for sikkerhed

# Hvad er Internet



Kommunikation mellem mennesker!

Baseret på TCP/IP

- best effort
- packet switching (IPv6 kalder det packets, ikke datagram)
- forbindelsesorienteret, *connection-oriented*
- forbindelsesløs, *connection-less*

RFC-1958:

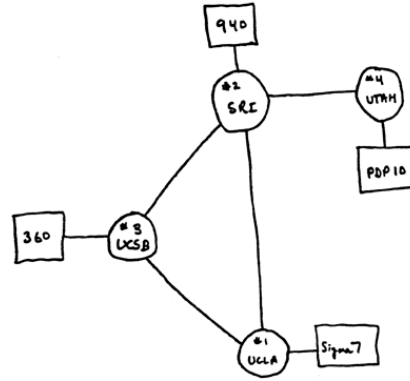
A good analogy for the development of the Internet is that of constantly renewing the individual streets and buildings of a city, rather than razing the city and rebuilding it. The architectural principles therefore aim to provide a framework for creating cooperation and standards, as a small "spanning set" of rules that generates a large, varied and evolving space of technology.

# IP netværk: Internettet historisk set



- 1961 L. Kleinrock, MIT packet-switching teori
- 1962 J. C. R. Licklider, MIT - notes
- 1964 Paul Baran: On Distributed Communications
- 1969 ARPANET startes 4 noder
- 1971 14 noder
- 1973 Arbejde med IP startes
- 1973 Email er ca. 75% af ARPANET trafik
- 1974 TCP/IP: Cerf/Kahn: A protocol for Packet Network Interconnection
- 1983 EUUG → DKUUG/DIKU forbindelse
- 1988 ca. 60.000 systemer på Internettet The Morris Worm rammer ca. 10%
- 2000 Maj I LOVE YOU ormen rammer
- 2002 Ialt ca. 130 millioner på Internet

# Internet historisk set - anno 1969



- Node 1: University of California Los Angeles
- Node 2: Stanford Research Institute
- Node 3: University of California Santa Barbara
- Node 4: University of Utah

# De tidlige notater om Internet



L. Kleinrock *Information Flow in Large Communication nets*, 1961

J.C.R. Licklider, MIT noter fra 1962 *On-Line Man Computer Communication*

Paul Baran, 1964 *On distributed Communications* 12-bind serie af rapporter

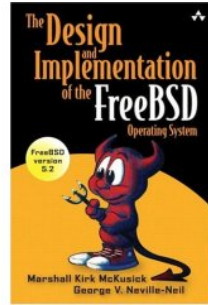
<http://www.rand.org/publications/RM/baran.list.html>

V. Cerf og R. Kahn, 1974 *A protocol for Packet Network Interconnection* IEEE Transactions on Communication, vol. COM-22, pp. 637-648, May 1974

De tidlige notater kan findes på nettet!

Læs evt. mere i mit speciale <http://www.inet6.dk/thesis.pdf>





UNIX kildeteksten var nem at få fat i for universiteter og mange andre

Bell Labs/AT&T var et telefonselskab - ikke et software hus

På Berkeley Universitetet blev der udviklet en del på UNIX og det har givet anledning til en hel gren kaldet BSD UNIX

BSD står for Berkeley Software Distribution

BSD UNIX har blandt andet resulteret i virtual memory management og en masse TCP/IP relaterede applikationer

## BSD licensen er pragmatisk



BSD licensen kræver ikke at man offentliggør sine ændringer, man kan altså bruge BSD kildetekst og stadig lave et kommercielt produkt!

GNU GPL bliver af nogle omtalt som en virus - der *inficerer* softwaren, og afledte projekter

# Hvad er Internet



80'erne IP/TCP starten af 80'erne

90'erne IP version 6 udarbejdes

- IPv6 ikke brugt i Europa og US
- IPv6 er ekstremt vigtigt i Asien
- historisk få adresser tildelt til 3.verdenslande
- Større Universiteter i USA har ofte større allokering end Kina!

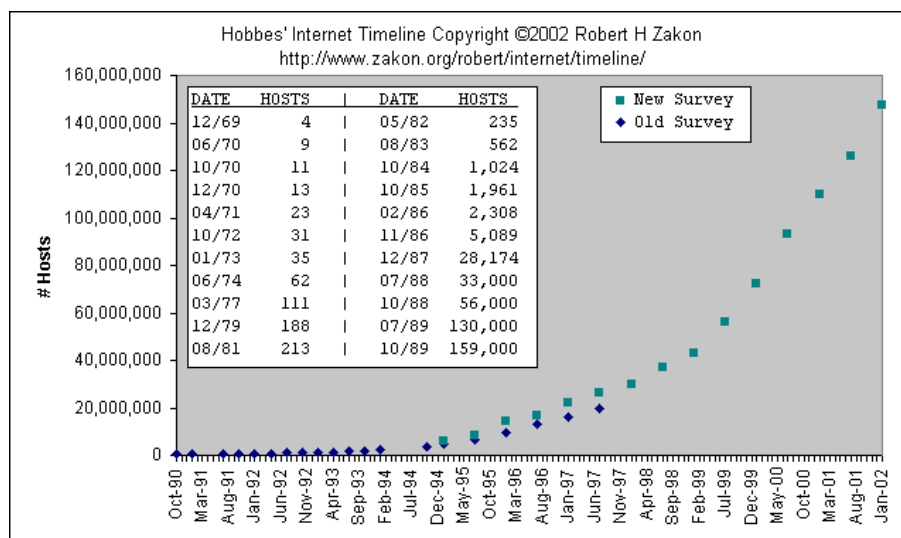
1991 WWW "opfindes" af Tim Berners-Lee hos CERN

E-mail var hovedparten af trafik - siden overtog web/http førstepladsen

# Hvad er Internet hosts



## Antallet af hosts på Internet



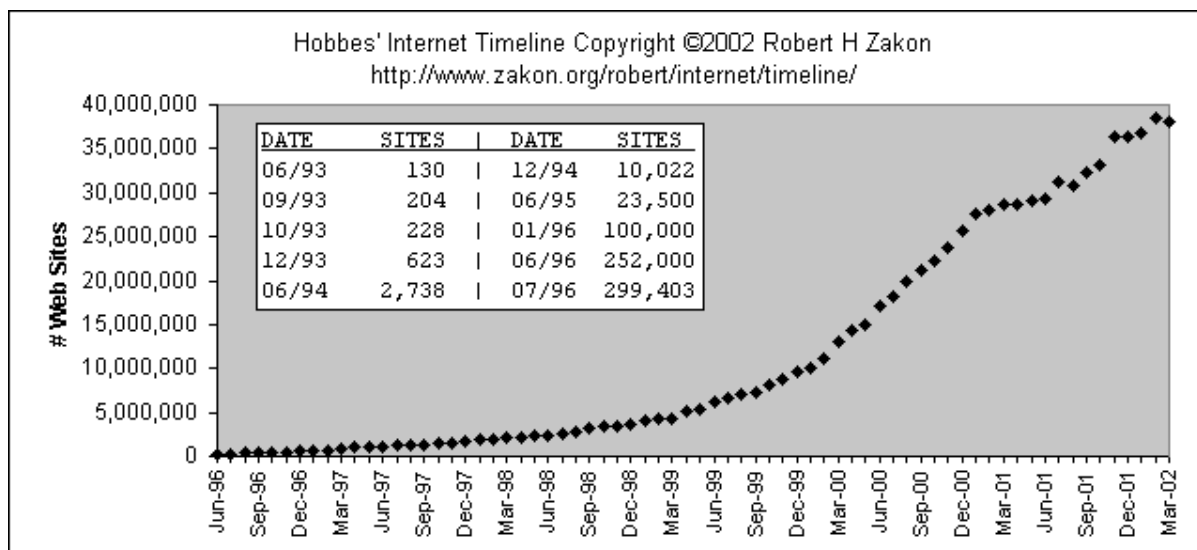
Kilde: Hobbes' Internet Timeline v5.6

<http://www.zakon.org/robert/internet/timeline/>

# Hvad er Internet World Wide Web



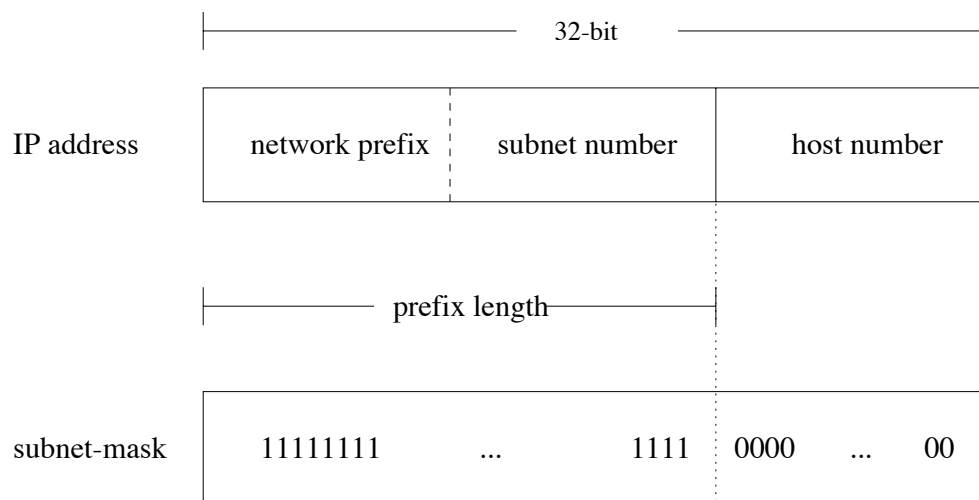
## Antallet af World Wide Web servere



Kilde: Hobbes' Internet Timeline v5.6

<http://www.zakon.org/robert/internet/timeline/>

# Fælles adresserum



Hvad kendetegner internet idag

Der er et fælles adresserum baseret på 32-bit adresser, example 10.0.0.1

# IPv4 adresser og skrivemåde



```
hlk@bigfoot:hlk$ ipconvert.pl 127.0.0.1
```

```
Adressen er: 127.0.0.1
```

```
Adressen er: 2130706433
```

```
hlk@bigfoot:hlk$ ping 2130706433
```

```
PING 2130706433 (127.0.0.1): 56 data bytes
```

```
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.135 ms
```

```
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.144 ms
```

IP-adresser skrives typisk som decimaltal adskilt af punktum

Kaldes **dot notation**: 10.1.2.3

Kan også skrive som oktal eller heksadecimale tal

# IP-adresser som bits



IP-adresse: 127.0.0.1

Heltal: 2130706433

Binary: 11111110000000000000000000000001

IP-adresser kan også konverteres til bits

Computeren regner binært, vi bruger dot-notationen





Tidligere benyttede man klasseinddelingen af IP-adresser: A, B, C, D og E

Desværre var denne opdeling ufleksibel:

- A-klasse kunne potentielt indeholde 16 millioner hosts
- B-klasse kunne potentielt indeholder omkring 65.000 hosts
- C-klasse kunne indeholde omkring 250 hosts

Derfor bad de fleste om adresser i B-klasser - så de var ved at løbe tør!

D-klasse benyttes til multicast

E-klasse er blot reserveret

Se evt. [http://en.wikipedia.org/wiki/Classful\\_network](http://en.wikipedia.org/wiki/Classful_network)

# CIDR Classless Inter-Domain Routing



Classfull routing		Classless routing (CIDR)	
4 Class C networks	Inherent subnet mask	Supernet	Subnet mask
192.0.08.0	255.255.255.0	192.0.08.0	255.255.252.0 (252d=11111100b)
192.0.09.0	255.255.255.0		
192.0.10.0	255.255.255.0		
192.0.11.0	255.255.255.0		
		Base network/prefix 192.0.8.0/	

Subnetmasker var oprindeligt indforstået

Dernæst var det noget man brugte til at opdele sit A, B eller C net med

Ved at tildele flere C-klasser kunne man spare de resterende B-klasser - men det betød en routing table explosion

Idag er subnetmaske en sammenhængende række 1-bit der angiver størrelse på nettet

10.0.0.0/24 betyder netværket 10.0.0.0 med subnetmaske 255.255.255.0

Nogle få steder kaldes det tillige supernet, supernetting

# RFC-1918 private netværk



Der findes et antal adresserum som alle må benytte frit:

- 10.0.0.0 - 10.255.255.255 (10/8 prefix)
- 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
- 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)

Address Allocation for Private Internets RFC-1918 adresserne!

NB: man må ikke sende pakker ud på internet med disse som afsender, giver ikke mening

# IPv4 adresser opsummering



- Altid 32-bit adresser
- Skrives typisk med 4 decimaltal dot notation 10.1.2.3
- Netværk angives med CIDR Classless Inter-Domain Routing RFC-1519
- CIDR notation 10.0.0.0/8 - fremfor 10.0.0.0 med subnet maske 255.0.0.0
- Specielle adresser
  - 127.0.0.1 localhost/loopback
  - 0.0.0.0 default route
- RFC-1918 angiver private adresser som alle kan bruge

# Stop - netværket idag



Bemærk hvilket netværk vi bruger idag

Primære server fiona har IP-adressen 10.0.45.36

Primære router luffe har IP-adressen 10.0.45.2 (og flere andre)

Sekundære router idag er Bianca som har IP-adressen 10.0.46.2 (og flere andre)

Hvis du kender til IP i forvejen så udforsk gerne på egen hånd netværket

Det er tilladt at logge ind på alle systemer, undtagen Henrik's laptop bigfoot :-)

**Det er forbudt at ændre IP-konfiguration og passwords**

Nu burde I kunne forbinde jer til netværket fysisk, check med ping 10.0.45.2

Det er nok at en PC i hver gruppe er på kursusnetværket

Pause for dem hvor det virker, mens vi ordner resten

# OSI og Internet modellerne



OSI Reference  
Model

Application
Presentation
Session
Transport
Network
Link
Physical

Internet protocol suite

Applications  HTTP, SMTP, FTP, SNMP,	NFS
	XDR
	RPC
TCP UDP	
IPv4	IPv6 ICMPv6 ICMP
ARP RARP	
MAC	
Ethernet token-ring ATM ...	



Der er mange muligheder med IP netværk, IP kræver meget lidt

Ofte benyttede idag er:

- Ethernet - varianter 10mbit, 100mbit, gigabit, 10 Gigabit findes, men er dyrt
- Wireless 802.11 teknologier
- ADSL/ATM teknologier til WAN forbindelser
- MPLS ligeledes til WAN forbindelser

Ethernet kan bruge kobberledninger eller fiber

WAN forbindelser er typisk fiber på grund af afstanden mellem routere

Tidligere benyttede inkluderer: X.25, modem, FDDI, ATM, Token-Ring

# Ethernet stik, kabler og dioder



Dioder viser typisk om der er link, hastighed samt aktivitet



# Trådløse teknologier



Et typisk 802.11 Access-Point (AP) der har Wireless og Ethernet stik/switch

# MAC adresser



00-03-93	(hex)	Apple Computer, Inc.
000393	(base 16)	Apple Computer, Inc.
		20650 Valley Green Dr.
		Cupertino CA 95014
		UNITED STATES

Netværksteknologierne benytter adresser på lag 2

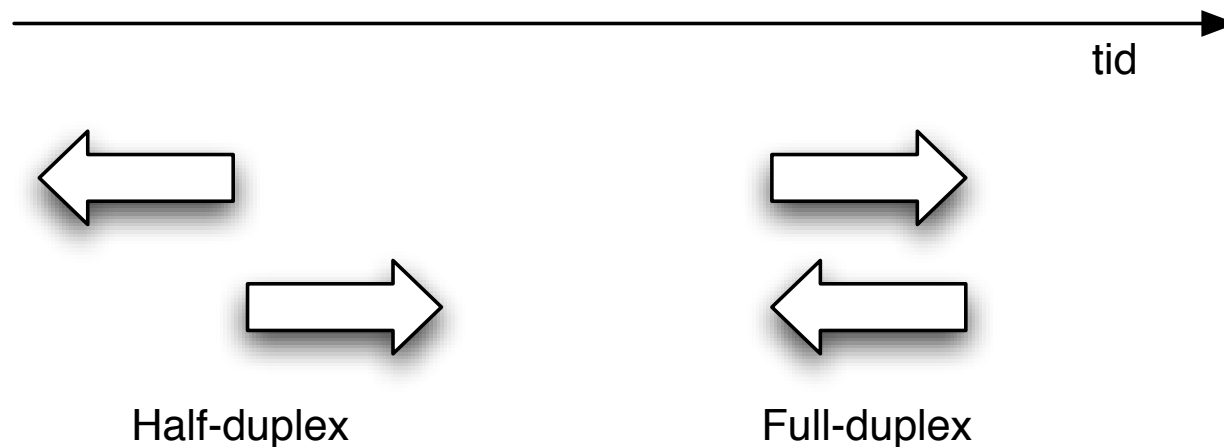
Typisk svarende til 48-bit MAC adresser som kendes fra Ethernet MAC-48/EUI-48

Første halvdel af adresserne er Organizationally Unique Identifier (OUI)

Ved hjælp af OUI kan man udlede hvilken producent der har produceret netkortet

<http://standards.ieee.org/regauth/oui/index.shtml>

# Half/full-duplex og speed



Hvad hastighed overføres data med?

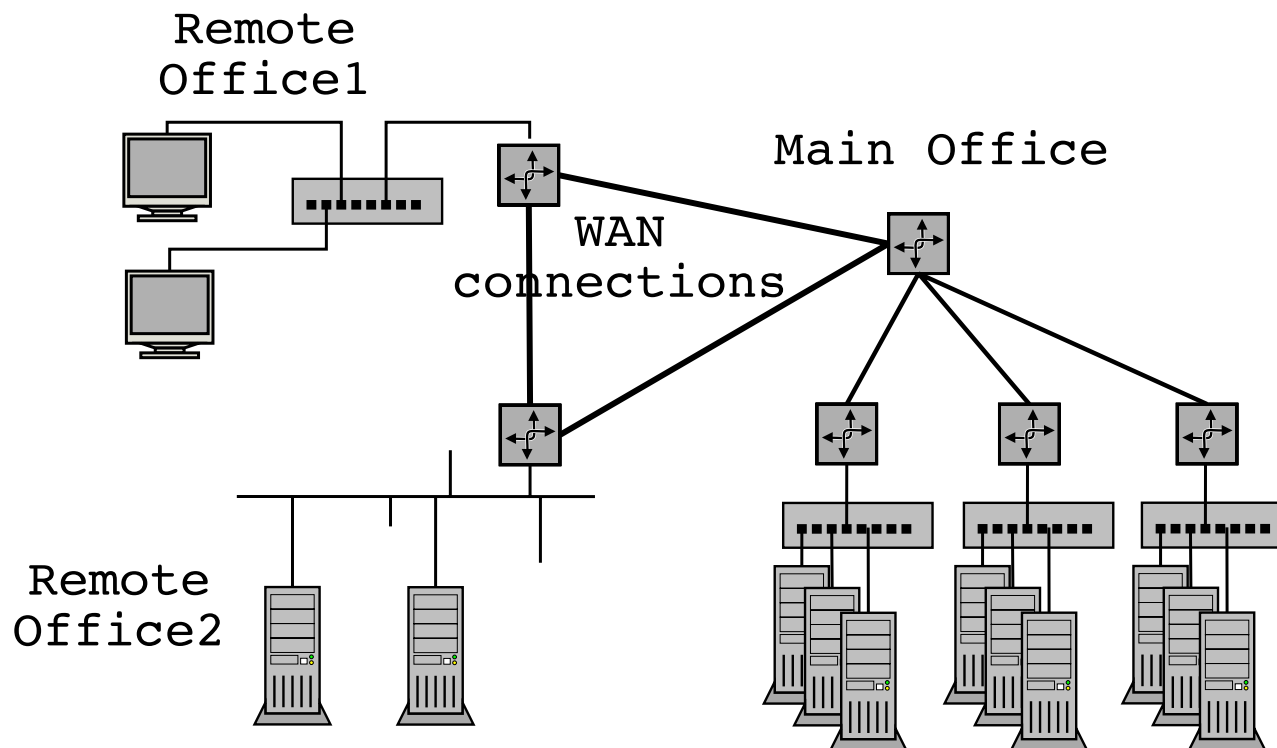
De fleste nyere Ethernet netkort kan køre i fuld-duplex

med full-duplex kan der både sendes og modtages data samtidigt

Ethernet kan benytte auto-negotiation - der ofte virker

Klart bedre i gigabitnetkort men pas på

# Broer og routere



Fysisk er der en begrænsning for hvor lange ledningerne må være

# Bridges



Ethernet er broadcast teknologi, hvor data sendes ud på et delt medie - Æteren

Broadcast giver en grænse for udbredningen vs hastighed

Ved hjælp af en bro kan man forbinde to netværkssegmenter på layer-2

Broen kopierer data mellem de to segmenter

Virker som en forstærker på signalet, men mere intelligent

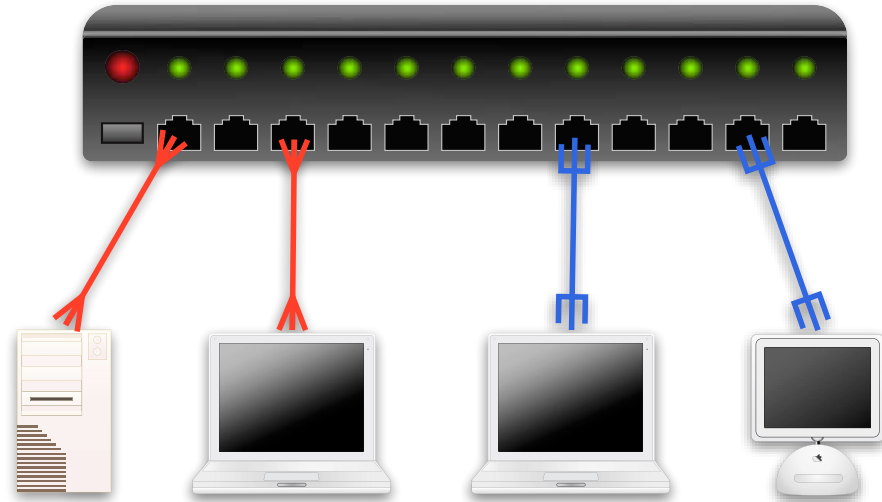
Den intelligente bro kender MAC adresserne på hver side

Broen kopierer kun hvis afsender og modtager er på hver sin side

Kilde: For mere information søg efter Aloha-net

<http://en.wikipedia.org/wiki/ALOHAnet>

## En switch

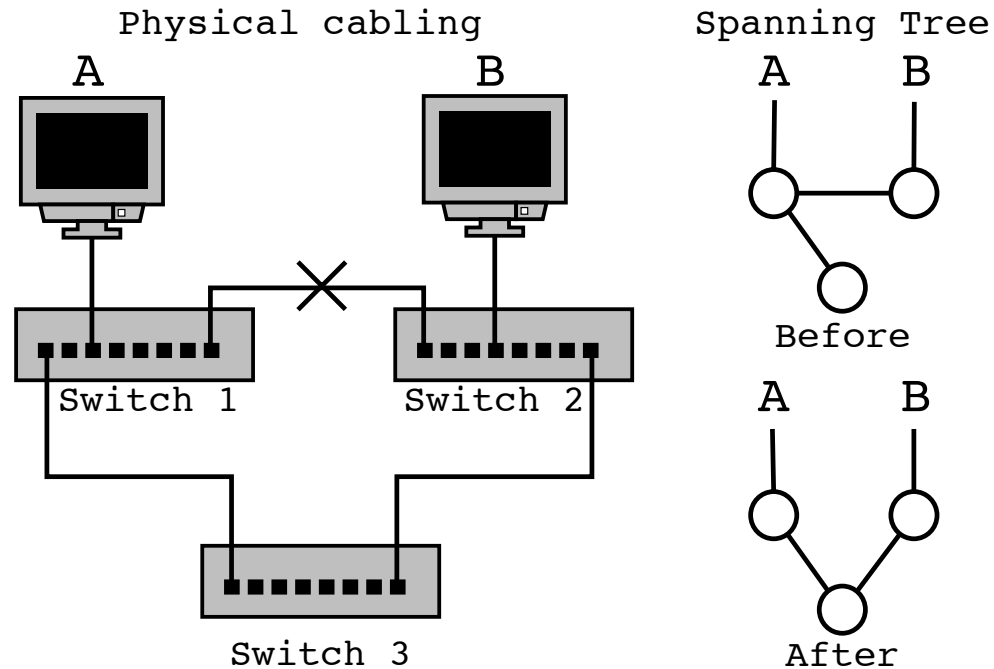


Ved at fortsætte udviklingen kunne man samle broer til en switch

En switch idag kan sende og modtage på flere porte samtidig, og med full-duplex

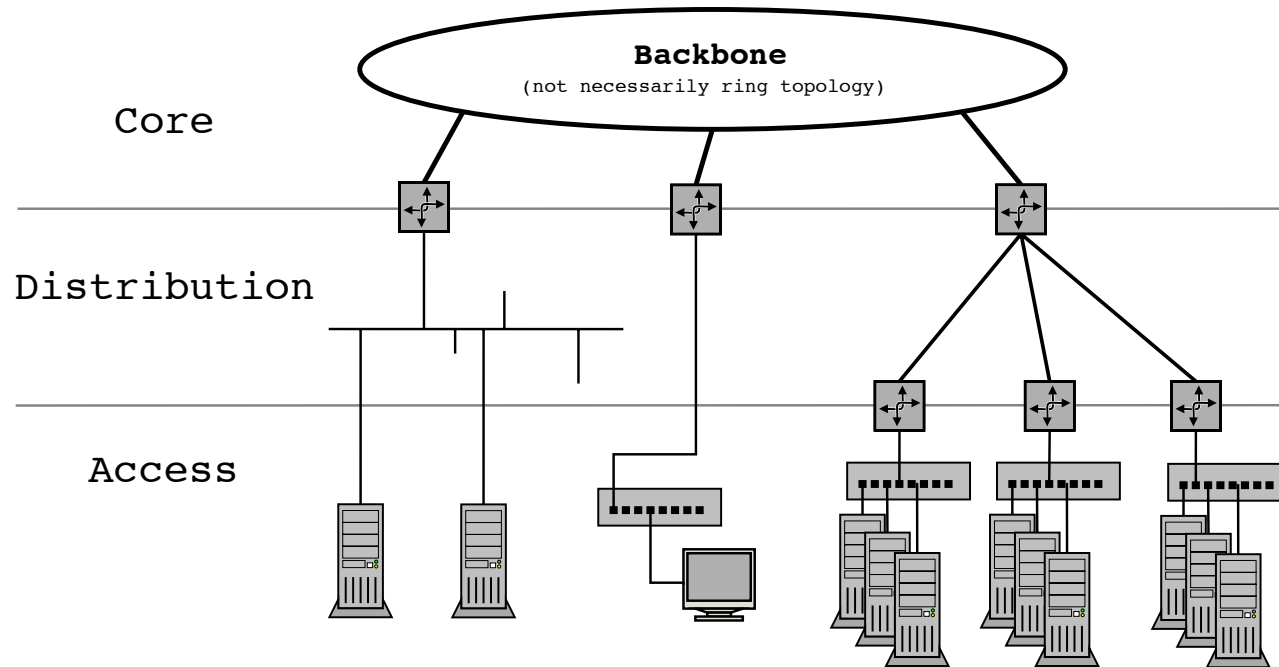
Bemærk performance begrænses af backplane i switchen

# Topologier og Spanning Tree Protocol



Se mere i bogen af Radia Perlman, *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols*

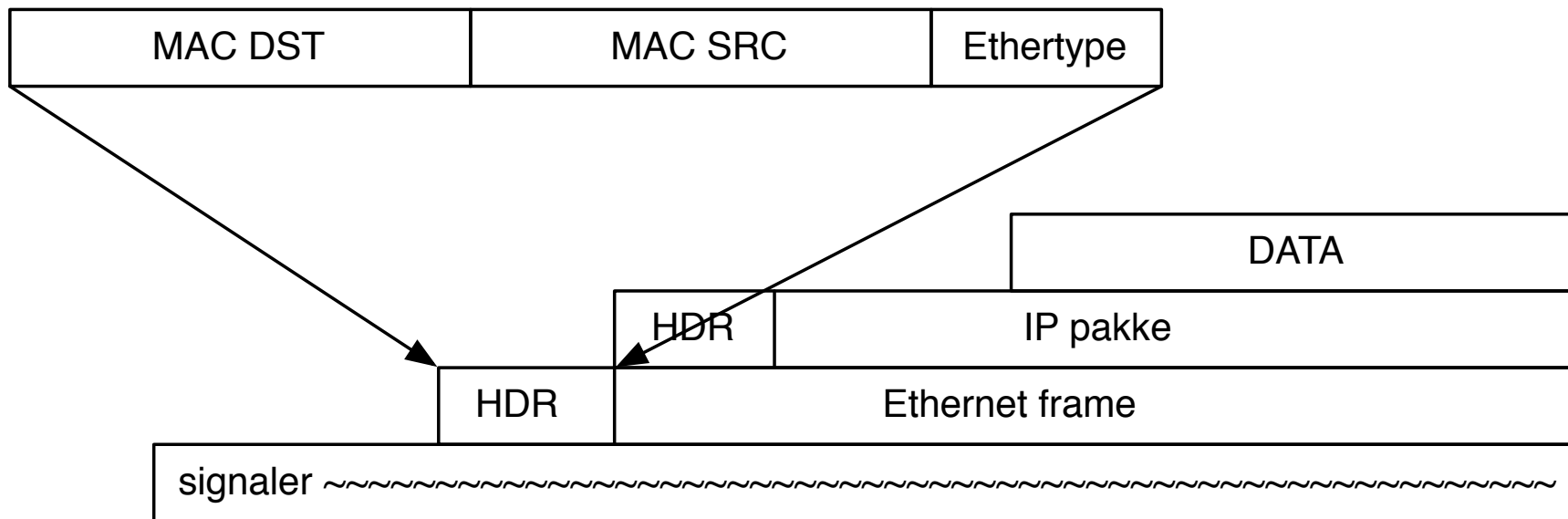
# Core, Distribution og Access net



Det er ikke altid man har præcis denne opdeling, men den er ofte brugt



# Pakker i en datastrøm

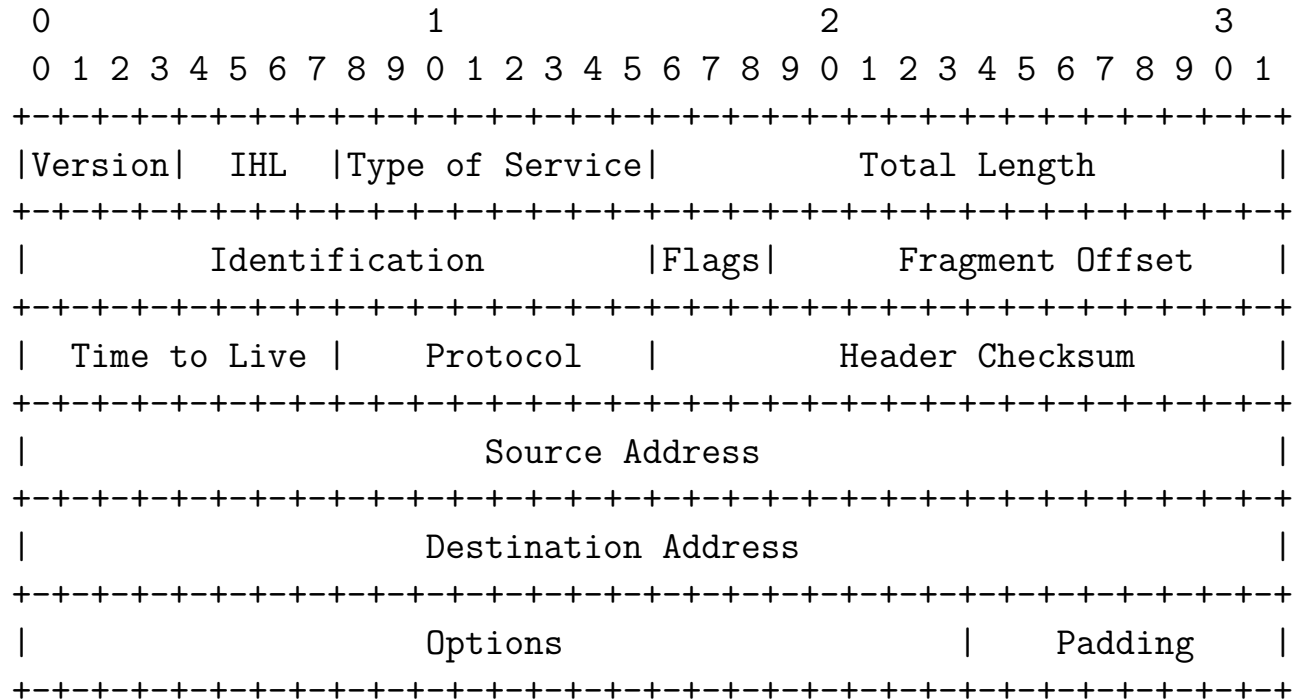


Ser vi data som en datastrøm er pakkerne blot et mønster lagt henover data

Netværksteknologien definerer start og slut på en frame

Fra et lavere niveau modtager vi en pakke, eksempelvis 1500-bytes fra Ethernet driver

# IPv4 pakken - header - RFC-791



Example Internet Datagram Header

# IP karakteristik



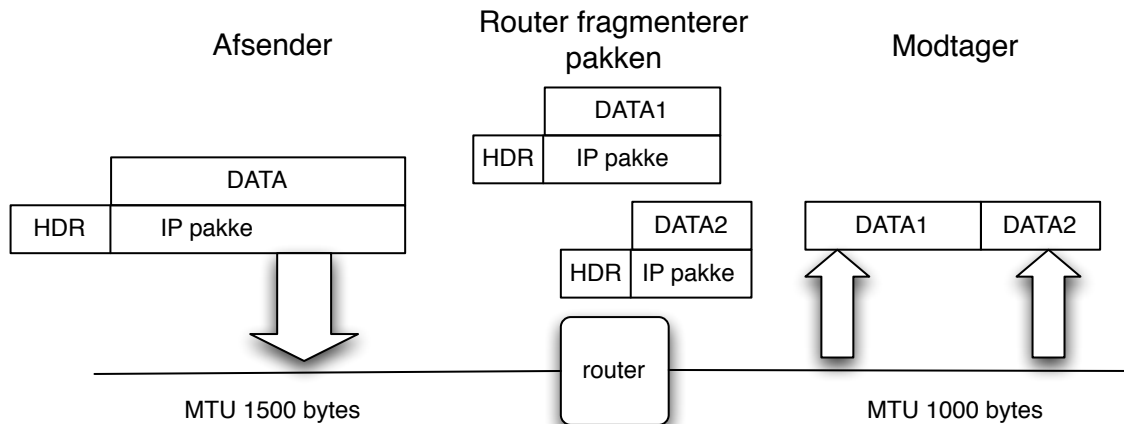
Fælles adresserum

Best effort - kommer en pakke fra er det fint, hvis ikke må højere lag klare det

Kræver ikke mange services fra underliggende teknologi *dumt netværk*

Defineret gennem åben standardiseringsprocess og RFC-dokumenter

# Fragmentering og PMTU



Hidtil har vi antaget at der blev brugt Ethernet med pakkestørrelse på 1500 bytes

Pakkestørrelsen kaldes MTU Maximum Transmission Unit

Skal der sendes mere data opdeles i pakker af denne størrelse, fra afsender

Men hvad hvis en router på vejen ikke bruger 1500 bytes, men kun 1000

# ICMP Internet Control Message Protocol



Kontrolprotokol og fejlmeldinger

Nogle af de mest almindelige beskedtyper

- echo
- netmask
- info

Bruges generelt til *signalering*

Defineret i RFC-792

**NB: nogle firewall-administratorer blokerer alt ICMP - det er forkert!**

# ICMP beskedtyper



## Type

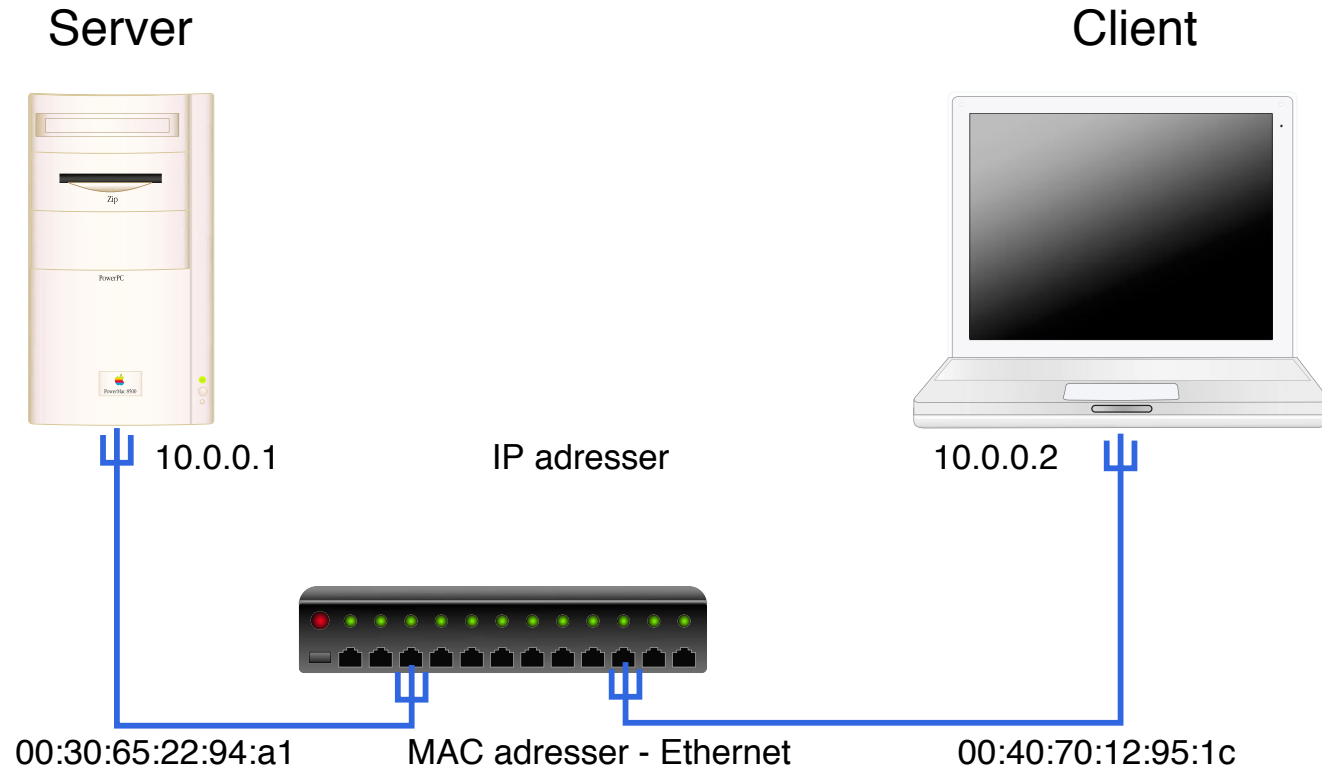
- 0 = net unreachable;
- 1 = host unreachable;
- 2 = protocol unreachable;
- 3 = port unreachable;
- 4 = fragmentation needed and DF set;
- 5 = source route failed.

Ved at fjerne ALT ICMP fra et net fjerner man nødvendig funktionalitet!

Tillad ICMP types:

- 3 Destination Unreachable
- 4 Source Quench Message
- 11 Time Exceeded
- 12 Parameter Problem Message

# Hvordan virker ARP?



## Hvordan virker ARP? - 2



**ping 10.0.0.2** udført på server medfører

ARP Address Resolution Protocol request/reply:

- ARP request i broadcast - Who has 10.0.0.2 Tell 10.0.0.1
- ARP reply (fra 10.0.0.2) 10.0.0.2 is at 00:40:70:12:95:1c

IP ICMP request/reply:

- Echo (ping) request fra 10.0.0.1 til 10.0.0.2
- Echo (ping) reply fra 10.0.0.2 til 10.0.0.1
- ...

ARP udføres altid på Ethernet før der kan sendes IP trafik  
(kan være RARP til udstyr der henter en adresse ved boot)



# ARP cache



```
hlk@bigfoot:hlk$ arp -an  
? (10.0.42.1) at 0:0:24:c8:b2:4c on en1 [ethernet]  
? (10.0.42.2) at 0:c0:b7:6c:19:b on en1 [ethernet]
```

ARP cache kan vises med kommandoen `arp -an`

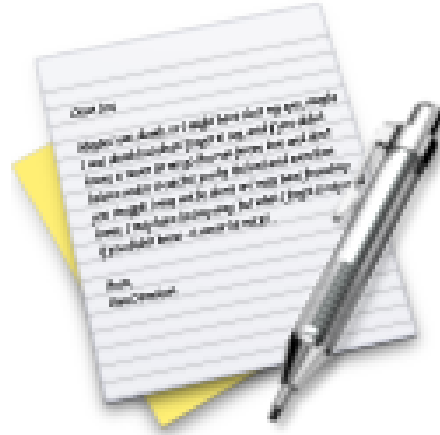
`-a` viser alle

`-n` viser kun adresserne, prøver ikke at slå navne op - typisk hurtigere

ARP cache er dynamisk og adresser fjernes automatisk efter 5-20 minutter hvis de ikke bruges mere

Læs mere med `man 4 arp`

# Exercise



Now lets do the exercise

## Wireshark and tcpdump

which is number 4 in the exercise PDF.

# TCP/IP basiskonfiguration



```
ifconfig en0 10.0.42.1 netmask 255.255.255.0  
route add default gw 10.0.42.1
```

konfiguration af interfaces og netværk på UNIX foregår med:

ifconfig, route og netstat

- ofte pakket ind i konfigurationsmenuer m.v.

fejlsøgning foregår typisk med ping og traceroute

På Microsoft Windows benyttes ikke ifconfig  
men kommandoerne ipconfig og ipv6

## Små forskelle



```
$ route add default 10.0.42.1
```

*uden gw keyword!*

```
$ route add default gw 10.0.42.1
```

*Linux kræver gw med*

**NB: UNIX varianter kan indbyrdes være forskellige!**

Very useful if you are trying to access a network without DHCP.

## Flere små forskelle



ping eller ping6

Nogle systemer vælger at ping kommandoen kan ping'e både IPv4 og Ipv6

Andre vælger at ping kun benyttes til IPv4, mens IPv6 ping kaldes for ping6

Læg også mærke til jargonen *at pinge*



Netværkskonfiguration på OpenBSD:

```
# cat /etc/hostname.sk0
inet 10.0.0.23 0xffffffff00 NONE
# cat /etc/mygate
10.0.0.1
# cat /etc/resolv.conf
domain security6.net
lookup file bind
nameserver 212.242.40.3
nameserver 212.242.40.51
```



Netværkskonfiguration på FreeBSD /etc/rc.conf:

```
# This file now contains just the overrides from /etc/defaults/rc.conf.  
hostname="freebsd.security6.net  
#ifconfig_vr0="DHCP"  
ifconfig_vr0="inet 10.20.30.75 netmask 255.255.255.0"  
router_enable="NO"  
defaultrouter="10.20.30.65"  
keyrate="fast"  
moused_enable="YES"  
ntptime_enable="NO"  
ntptime_flags="none"  
saver="blank"  
sshd_enable="YES"  
usbd_enable="YES"  
...
```

# GUI værktøjer - autoconfiguration



Built-in Ethernet

TCP/IP DNS WINS AppleTalk 802.1X Proxies Ethernet

Configure IPv4: Using DHCP

IPv4 Address: Renew DHCP Lease

Subnet Mask: DHCP Client ID: ( If required )

Router:

Configure IPv6: Automatically

Router:

IPv6 Address:

Prefix Length:

Advanced



# GUI værktøjer - manuel konfiguration



Built-in Ethernet

TCP/IP DNS WINS AppleTalk 802.1X Proxies Ethernet

Configure IPv4: Manually

IPv4 Address: 0.0.0.0

Subnet Mask:

Router:

Configure IPv6: Manually

Router:

IPv6 Address:

Prefix Length:

# ifconfig output



```
hlk@bigfoot:hlk$ ifconfig -a
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    inet 127.0.0.1 netmask 0xff000000
    inet6 ::1 prefixlen 128
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    ether 00:0a:95:db:c8:b0
    media: autoselect (none) status: inactive
    supported media: none autoselect 100baseTX <half-duplex> 100baseTX <full-duplex> 100baseTX <full-duplex,loopback> 1000baseT <full-duplex> 1000baseT <full-duplex,hw-loopback> 1000baseT <full-duplex,flow-control> 1000baseT <full-duplex,flow-control,hw-loopback>
en1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    ether 00:0d:93:86:7c:3f
    media: autoselect (<unknown type>) status: inactive
    supported media: autoselect
```

ifconfig output er næsten ens på tværs af UNIX

# Vigtigste protokoller



ARP Address Resolution Protocol

IP og ICMP Internet Control Message Protocol

UDP User Datagram Protocol

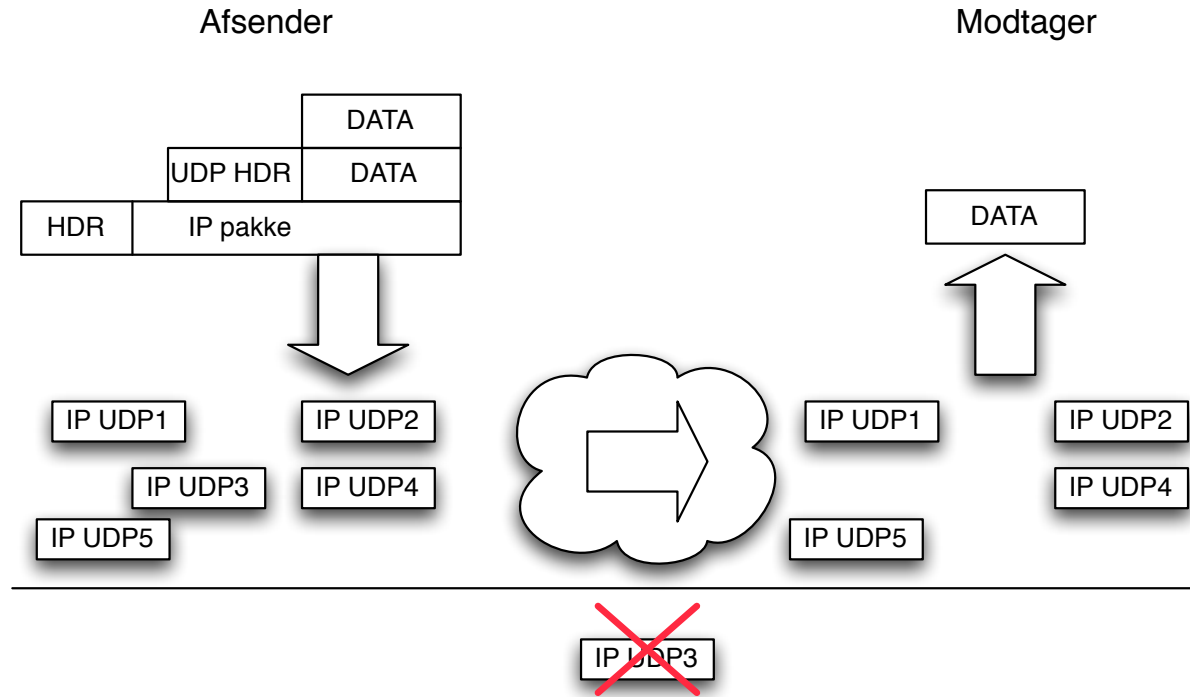
TCP Transmission Control Protocol

DHCP Dynamic Host Configuration Protocol

DNS Domain Name System

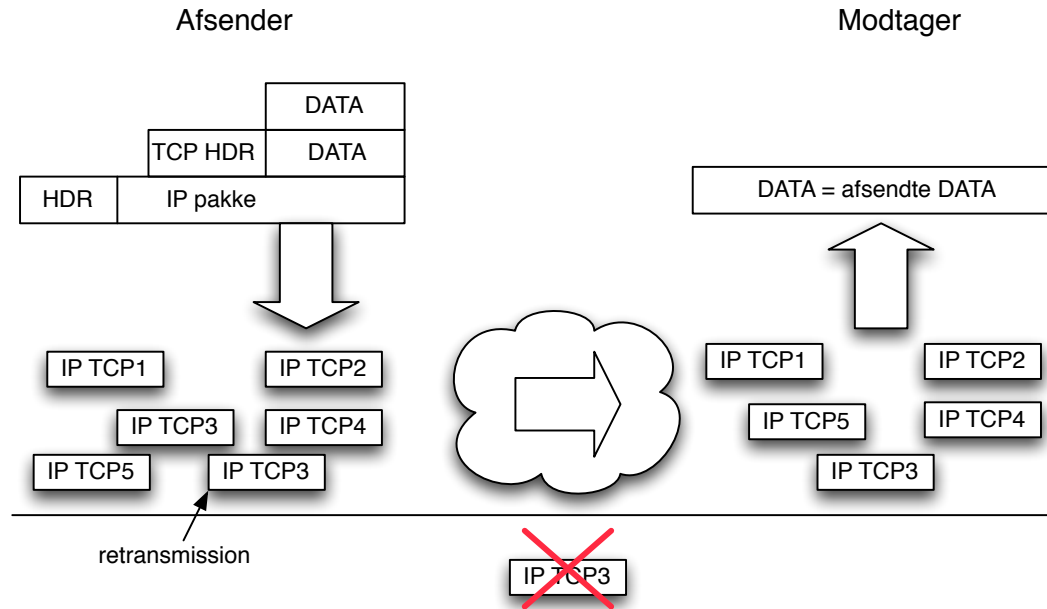
Ovenstående er omtrent minimumskrav for at komme på internet

# UDP User Datagram Protocol



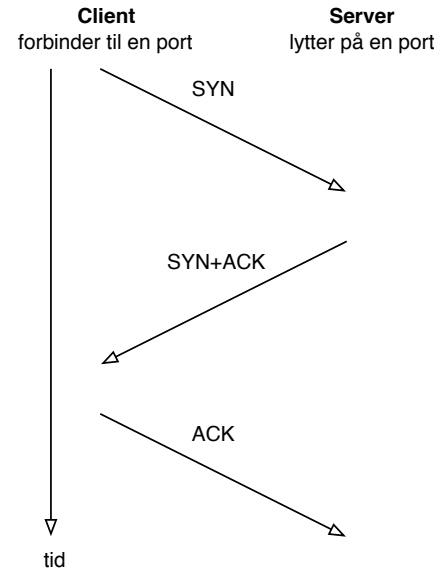
Forbindelsesløs RFC-768, *connection-less*

# TCP Transmission Control Protocol



Forbindelsesorienteret RFC-791 September 1981, *connection-oriented*  
Enten overføres data eller man får fejlmeddelelse

# TCP three way handshake



- **TCP SYN half-open** scans
- Tidligere loggede systemer kun når der var etableret en fuld TCP forbindelse - dette kan/kunne udnyttes til *stealth*-scans

# Exercise



Now lets do the exercise

## Capturing TCP Session packets

which is number **5** in the exercise PDF.

# Well-known port numbers



IANA vedligeholder en liste over magiske konstanter i IP

De har lister med hvilke protokoller har hvilke protokol ID m.v.

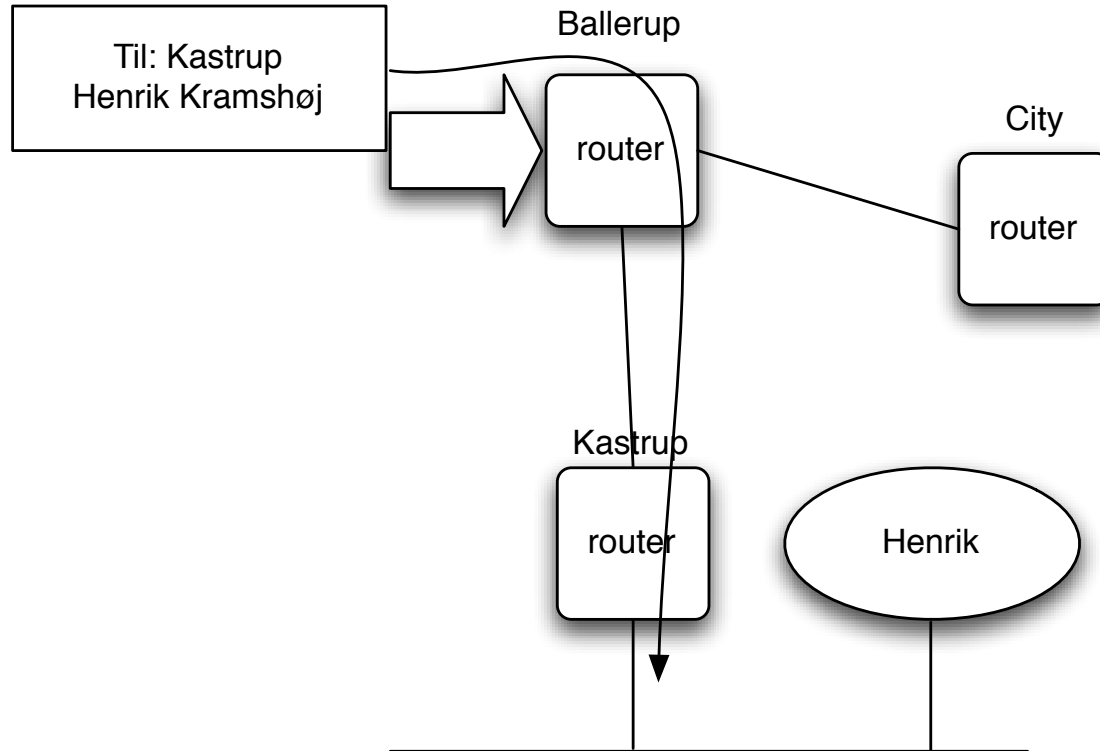
En liste af interesse er port numre, hvor et par eksempler er:

- Port 25 SMTP Simple Mail Transfer Protocol
- Port 53 DNS Domain Name System
- Port 80 HTTP Hyper Text Transfer Protocol over TLS/SSL
- Port 443 HTTP over TLS/SSL

Se flere på <http://www.iana.org>

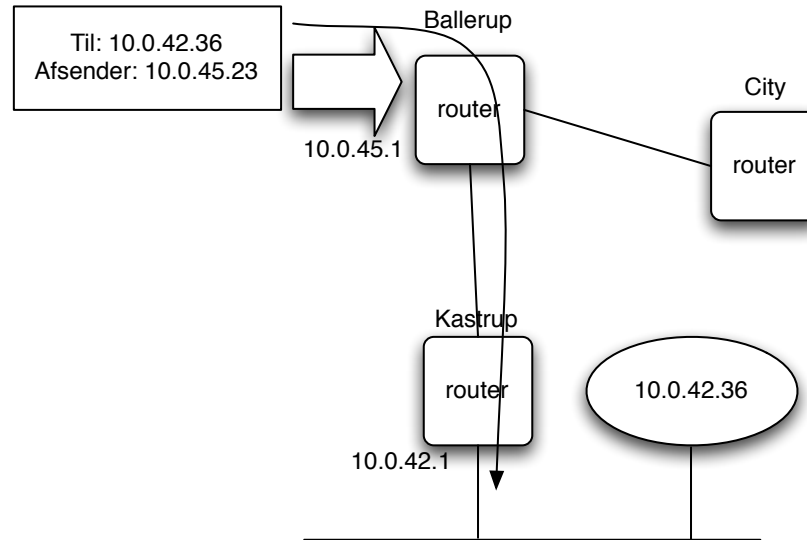


# Hierarkisk routing



Hvordan kommer pakkerne frem til modtageren

# IP default gateway

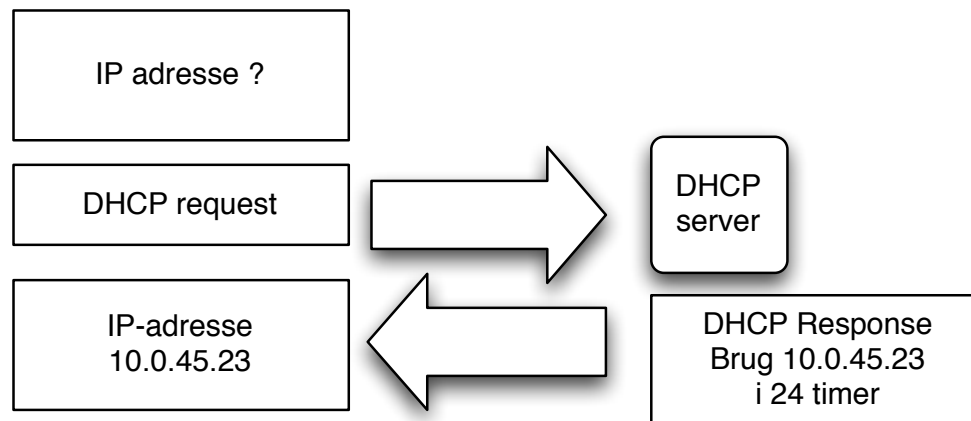


IP routing er nemt, longest match

En host kender typisk en default gateway i nærheden

En router har en eller flere upstream routere, få adresser den sender videre til

# DHCP Dynamic Host Configuration Protocol



Hvordan får man information om default gateway

Man sender et DHCP request og modtager et svar fra en DHCP server

Dynamisk konfiguration af klienter fra en centralt konfigureret server

Bruges til IP adresser og meget mere

# Routing



routing table - tabel over netværkskort og tilhørende adresser

default gateway - den adresse hvortil man sender *non-local* pakker  
kaldes også default route, gateway of last resort

routing styres enten manuelt - opdatering af route tabellen, eller konfiguration af adresser og subnet maske på netkort

eller automatisk ved brug af routing protocols - interne og eksterne route protokoller

de lidt ældre routing protokoller har ingen sikkerhedsmekanismer

**IP benytter longest match i routing tabeller!**

Den mest specifikke route gælder for forward af en pakke!

# Routing forståelse



```
$ netstat -rn
```

Routing tables

Internet:

Destination	Gateway	Flags	Refs	Use	Netif
default	10.0.0.1	UGSc	23	7	en0
10/24	link#4	UCS	1	0	en0
10.0.0.1	0:0:24:c1:58:ac	UHLW	24	18	en0
10.0.0.33	127.0.0.1	UHS	0	1	lo0
10.0.0.63	127.0.0.1	UHS	0	0	lo0
127	127.0.0.1	UCS	0	0	lo0
127.0.0.1	127.0.0.1	UH	4	7581	lo0
169.254	link#4	UCS	0	0	en0

Start med kun at se på Destination, Gateway og Netinterface

# whois systemet



IP adresserne administreres i dagligdagen af et antal Internet registries, hvor de største er:

- RIPE (Réseaux IP Européens) <http://ripe.net>
  - ARIN American Registry for Internet Numbers <http://www.arin.net>
  - Asia Pacific Network Information Center <http://www.apnic.net>
  - LACNIC (Regional Latin-American and Caribbean IP Address Registry) - Latin America and some Caribbean Islands
- disse fire kaldes for Regional Internet Registries (RIRs) i modsætning til Local Internet Registries (LIRs) og National Internet Registry (NIR)

## whois systemet-2



ansvaret for Internet IP adresser ligger hos ICANN The Internet Corporation for Assigned Names and Numbers

<http://www.icann.org>

NB: ICANN må ikke forveksles med IANA Internet Assigned Numbers Authority <http://www.iana.org/> som bestyrer portnumre m.v.

# Ping



ICMP - Internet Control Message Protocol

Benyttes til fejlbeskeder og til diagnosticering af forbindelser

ping programmet virker ved hjælp af ICMP ECHO request og forventer ICMP ECHO reply

```
$ ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=150 time=8.849 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=150 time=0.588 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=150 time=0.553 ms
```



# traceroute



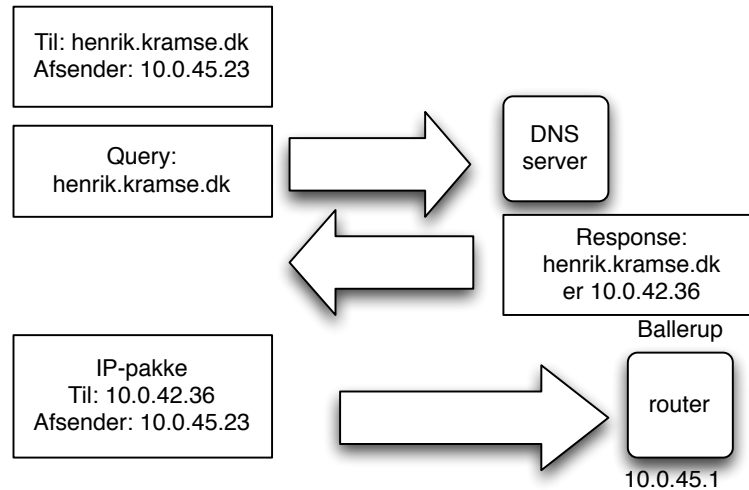
traceroute programmet virker ved hjælp af TTL

levetiden for en pakke tælles ned i hver router på vejen og ved at sætte denne lavt opnår man at pakken *timer ud* - besked fra hver router på vejen

default er UDP pakker, men på UNIX systemer er der ofte mulighed for at bruge ICMP

```
$ traceroute 217.157.20.129
traceroute to 217.157.20.129 (217.157.20.129),
30 hops max, 40 byte packets
 1  safri (10.0.0.11)  3.577 ms  0.565 ms  0.323 ms
 2  router (217.157.20.129)  1.481 ms  1.374 ms  1.261 ms
```

# Domain Name System



Gennem DHCP får man typisk også information om DNS servere

En DNS server kan slå navne, domæner og adresser op

Foregår via query og response med datatyper kaldet resource records

DNS er en distribueret database, så opslag kan resultere i flere opslag

# DNS systemet



navneopslag på Internet

tidligere brugte man en **hosts** fil

hosts fil bruges stadig lokalt til serveren - IP-adresser

UNIX: /etc/hosts

Windows c:\windows\system32\drivers\etc\hosts

Eksempel: www.security6.net har adressen 217.157.20.131

skrives i database filer, zone filer

ns1	IN	A	217.157.20.130
	IN	AAAA	2001:618:433::1
www	IN	A	217.157.20.131
	IN	AAAA	2001:618:433::14

# Mere end navneopslag



består af resource records med en type:

- adresser A-records
- IPv6 adresser AAAA-records
- autoritative navneservere NS-records
- post, mail-exchanger MX-records
- flere andre: md , mf , cname , soa , mb , mg , mr , null , wks , ptr , hinfo , minfo , mx ....

IN	MX	10	mail.security6.net.
IN	MX	20	mail2.security6.net.

# Basal DNS opsætning på klienter



`/etc/resolv.conf`

NB: denne fil kan hedde noget andet på UNIX varianter!

eksempelvis `/etc/netsvc.conf`

typisk indhold er domænenavn og IP-adresser for navneservere

```
domain security6.net
```

```
nameserver 212.242.40.3
```

```
nameserver 212.242.40.51
```

# DNS root servere



Root-servere - 13 stk geografisk distribueret fordelt på Internet

I.ROOT-SERVERS.NET.	3600000 A	192.36.148.17
E.ROOT-SERVERS.NET.	3600000 A	192.203.230.10
D.ROOT-SERVERS.NET.	3600000 A	128.8.10.90
A.ROOT-SERVERS.NET.	3600000 A	198.41.0.4
H.ROOT-SERVERS.NET.	3600000 A	128.63.2.53
C.ROOT-SERVERS.NET.	3600000 A	192.33.4.12
G.ROOT-SERVERS.NET.	3600000 A	192.112.36.4
F.ROOT-SERVERS.NET.	3600000 A	192.5.5.241
B.ROOT-SERVERS.NET.	3600000 A	128.9.0.107
J.ROOT-SERVERS.NET.	3600000 A	198.41.0.10
K.ROOT-SERVERS.NET.	3600000 A	193.0.14.129
L.ROOT-SERVERS.NET.	3600000 A	198.32.64.12
M.ROOT-SERVERS.NET.	3600000 A	202.12.27.33

# DK-hostmaster



bestyrer .dk TLD - top level domain

man registrerer ikke .dk-domæner hos DK-hostmaster, men hos en registrator

Et domæne bør have flere navneservere og flere postservere

autoritativ navneserver - ved autoritativt om IP-adresse for maskine.domæne.dk findes

ikke-autoritativ - har på vegne af en klient slået en adresse op

Det anbefales at overveje en service som <http://www.gratisdns.dk> der har 5 navneservere distribueret over stor geografisk afstand - en udenfor Danmark

# Navngivning af servere



Hvordan skal vi kunne huske og administrere servere?

Det er ikke nemt at navngive hverken brugere eller servere!

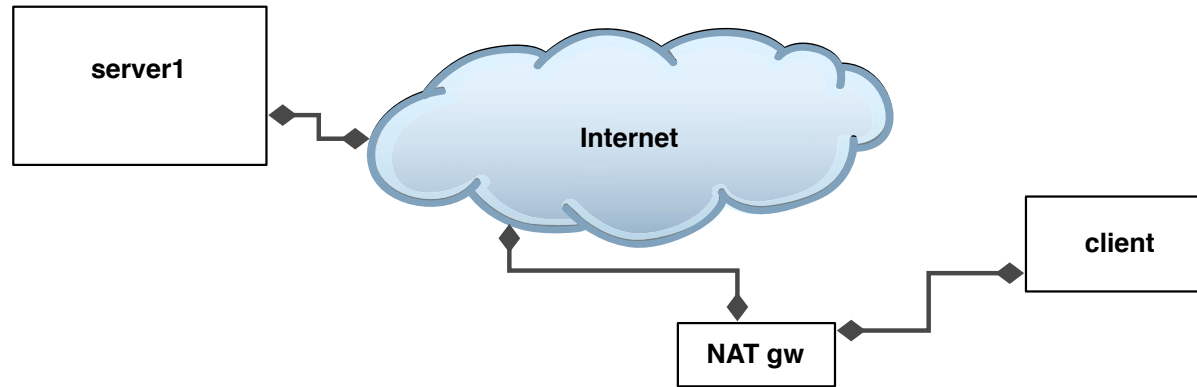
Selvom det lyder smart med A01S13, som forkortelse af Afdeling 01's Server nr 13, er det umuligt at huske

... men måske nødvendigt i de største netværk

- Windows serveren er domænecontroller - skal hedde:
- Linux server som er terminalserver - skal hedde:
- PC-system med NetBSD skal måske være vores ene server - skal hedde: ?
- PC-system 1 med en Linux server - skal hedde:
- PC-system 2 med en Linux server - skal hedde:

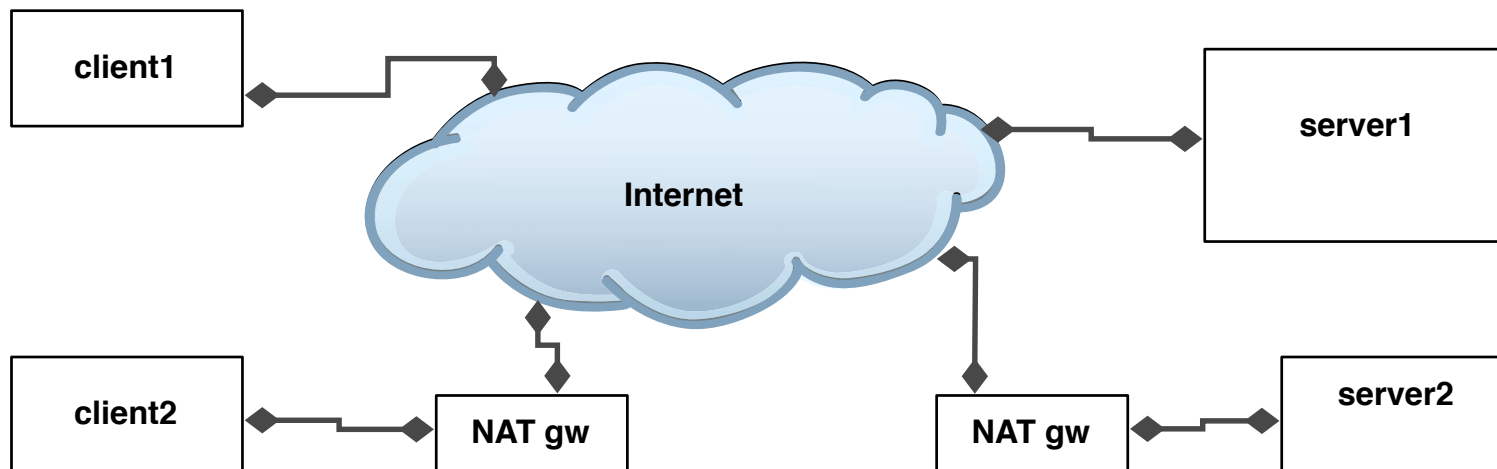


# NAT Network Address Translation



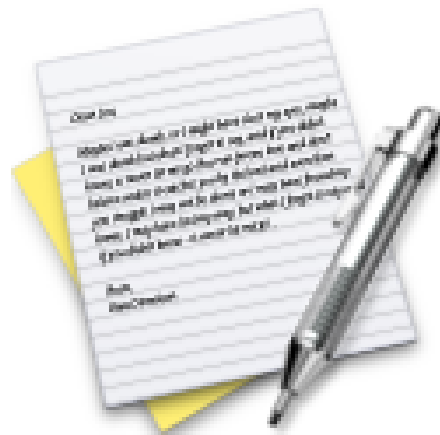
- NAT bruges til at forbinde et privat net (RFC-1918 adresser) med internet
- NAT gateway udskifter afsender adressen med sin egen
- En quick and dirty fix der vil forfølge os for resten af vores liv
- Lægger state i netværket - ødelægger fate sharing

# NAT is BAD



- NAT ødelægger end-to-end transparency!
- Problemer med servere bagved NAT
- "løser" problemet "godt nok"(tm) for mange
- Men idag ser vi multilevel NAT! - eeeeeewwwwww!
- Se RFC-2775 Internet Transparency for mere om dette emne

# Exercise

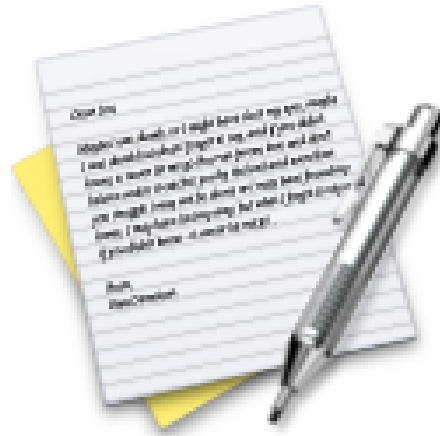


Now lets do the exercise

## Opslag i whois databaser

which is number **6** in the exercise PDF.

# Exercise

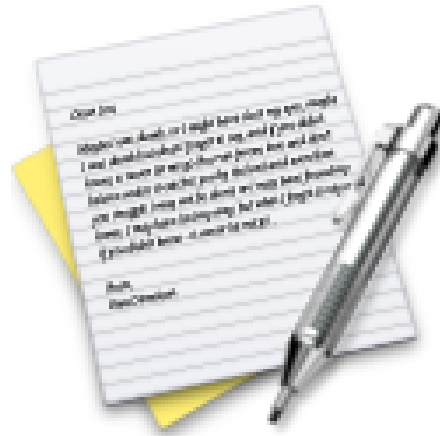


Now lets do the exercise

## ping og traceroute

which is number **7** in the exercise PDF.

# Exercise

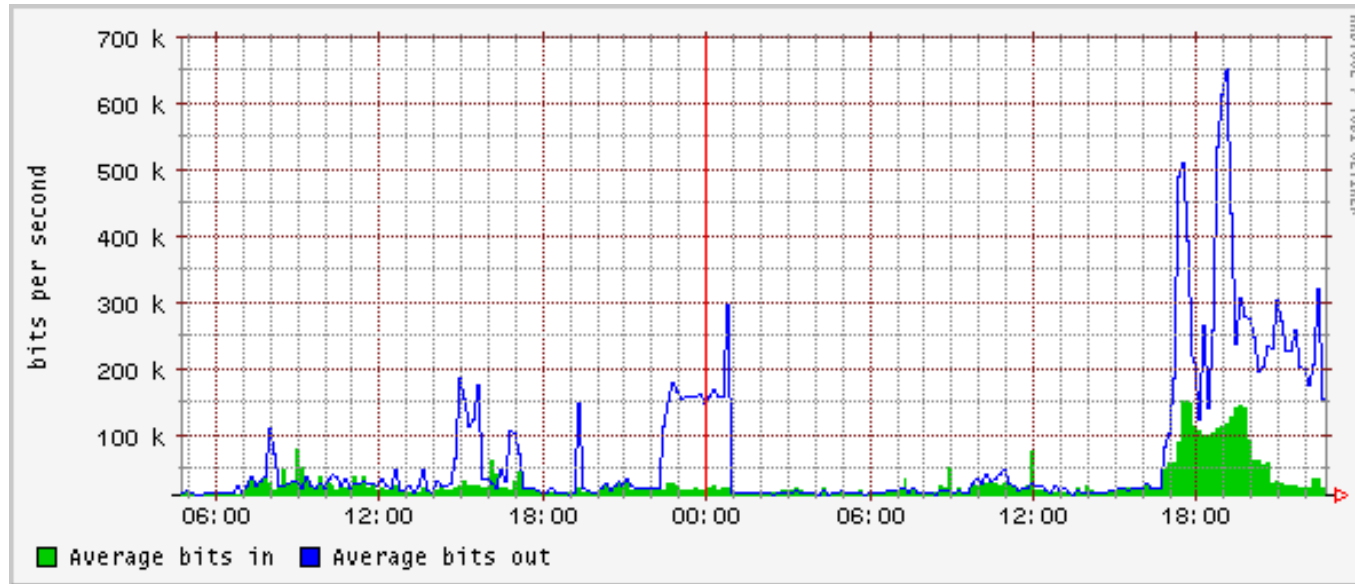


Now lets do the exercise

## DNS og navneopslag

which is number **8** in the exercise PDF.

## Dag 2 IPv6, Management, diagnosticering



# IPv4 Adresserummet er ved at løbe ud



Adresserummet er ved at løbe ud! faktum!

32-bit - der ikke kan udnyttes fuldt ud

Tidligere brugte man begreberne A,B og C klasser af IP-adresser

- 10.0.0.0 - 10.255.255.255 (10/8 prefix)
- 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
- 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)

Address Allocation for Private Internets RFC-1918 adresserne!

Husk at idag benyttes Classless Inter-Domain Routing CIDR

[http://en.wikipedia.org/wiki/Classless\\_Inter-Domain\\_Routing](http://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing)

Notation: 192.168.1.0/24

det sædvanlige hjemmenet med subnet maske 255.255.255.0

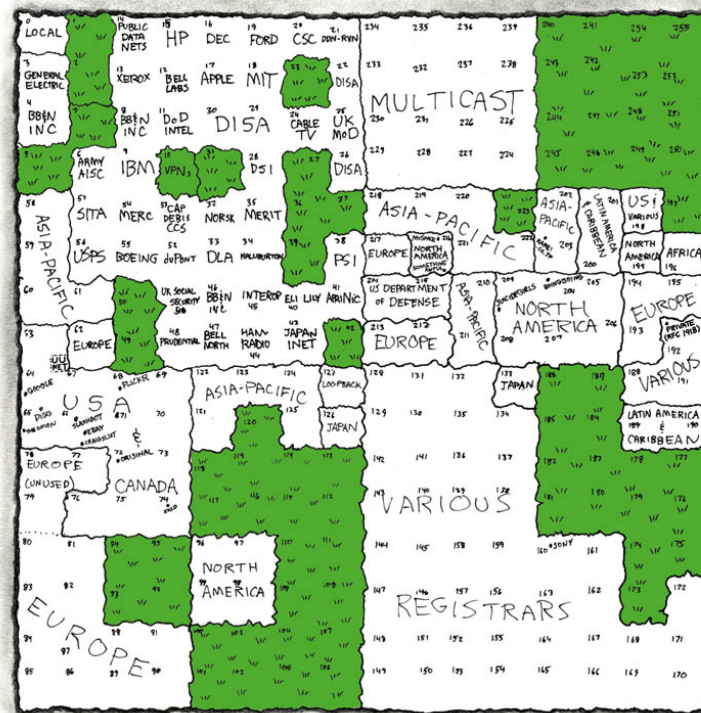
# Status idag





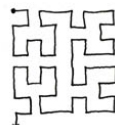


# MAP OF THE INTERNET THE IPV4 SPACE, 2006



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

## Tidslinie for IPv6 (forkortet)



- 1990 Vancouver IETF meeting det estimeres at klasse B vil løbe ud ca. marts 1994
- 1990 ultimo initiativer til at finde en afløser for IPv4
- 1995 januar RFC-1752 Recommendation for the IP NG Protocol
- 1995 september RFC-1883, RFC-1884, RFC-1885, RFC-1886 1. generation
- 1998 10. august "core"IPv6 dokumenter bliver Draft Standard
- Kilde: RFC-2460, RFC-2461, RFC-2463, RFC-1981 - m.fl.

# IPv6: Internet redesigned? - nej!



Målet var at bevare de gode egenskaber

- basalt set Internet i gamle dage
- back to basics!
- fate sharing
- kommunikationen afhænger ikke af state i netværket
- end-to-end transparency

Idag er Internet blevet en nødvendighed for mange!

**IP er en forretningskritisk ressource**

IPv6 basis i RFC-1752 The Recommendation for the IP Next Generation Protocol

# KAME - en IPv6 reference implementation



<http://www.kame.net>

- Er idag at betragte som en reference implementation
  - i stil med BSD fra Berkeley var det
- KAME har været på forkant med implementation af draft dokumenter
- KAME er inkluderet i OpenBSD, NetBSD, FreeBSD og BSD/OS - har været det siden version 2.7, 1.5, 4.0 og 4.2
- Projektet er afsluttet, men nye projekter fortsætter i WIDE regi <http://www.wide.ad.jp/>

- Der er udkommet to bøger som i detaljer gennemgår IPv6 protokollerne i KAME



# Hvordan bruger man IPv6



[www.inet6.dk](http://www.inet6.dk)

[hik@inet6.dk](mailto:hik@inet6.dk)

DNS AAAA record tilføjes

www	IN A	91.102.91.17
	IN AAAA	2001:16d8:ff00:12f::2
mail	IN A	91.102.91.17
	IN AAAA	2001:16d8:ff00:12f::2

# IPv6 adresser og skrivemåde



subnet prefix	interface identifier
---------------	----------------------

2001:16d8:ff00:012f:0000:0000:0000:0002

2001:16d8:ff00:12f::2

- 128-bit adresser, subnet prefix næsten altid 64-bit
- skrives i grupper af 4 hexcifre ad gangen adskilt af kolon :
- foranstillede 0 i en gruppe kan udelades, en række 0 kan erstattes med ::
- dvs 0:0:0:0:0:0:0:0 er det samme som  
0000:0000:0000:0000:0000:0000:0000:0000
- Dvs min webservers IPv6 adresse kan skrives som: 2001:16d8:ff00:12f::2
- Specielle adresser: ::1 localhost/loopback og :: default route

- Læs mere i RFC-3513





# IPv6 addresser - prefix notation



CIDR Classless Inter-Domain Routing RFC-1519

Aggregatable Global Unicast

2001::/16 RIR subTLA space

- 2001:200::/23 APNIC
- 2001:400::/23 ARIN
- 2001:600::/23 RIPE

2002::/16 6to4 prefix

3ffe::/16 6bone allocation

link-local unicast addresses

fe80::/10 genereres ud fra MAC adresserne EUI-64

# IPv6 adresser - multicast



Unicast - identificerer ét interface pakker sendes til en modtager

Multicast - identificerer flere interfaces pakker sendes til flere modtagere

Anycast - indentificerer en "gruppe" en pakke sendes til et vilkårligt interface med denne adresse typisk det nærmeste

Broadcast? er væk, udeladt, finito, gone!

Husk også at site-local er deprecated, se RFC-3879

# IPv6 pakken - header - RFC-2460



- Simplere - fixed size - 40 bytes
- Sjældent brugte felter (fra v4) udeladt (kun 6 vs 10 i IPv4)
- Ingen checksum!
- Adresser 128-bit
- 64-bit aligned, alle 6 felter med indenfor første 64

Mindre kompleksitet for routere på vejen medfører mulighed for flere pakker på en given router

# IPv6 pakken - header - RFC-2460

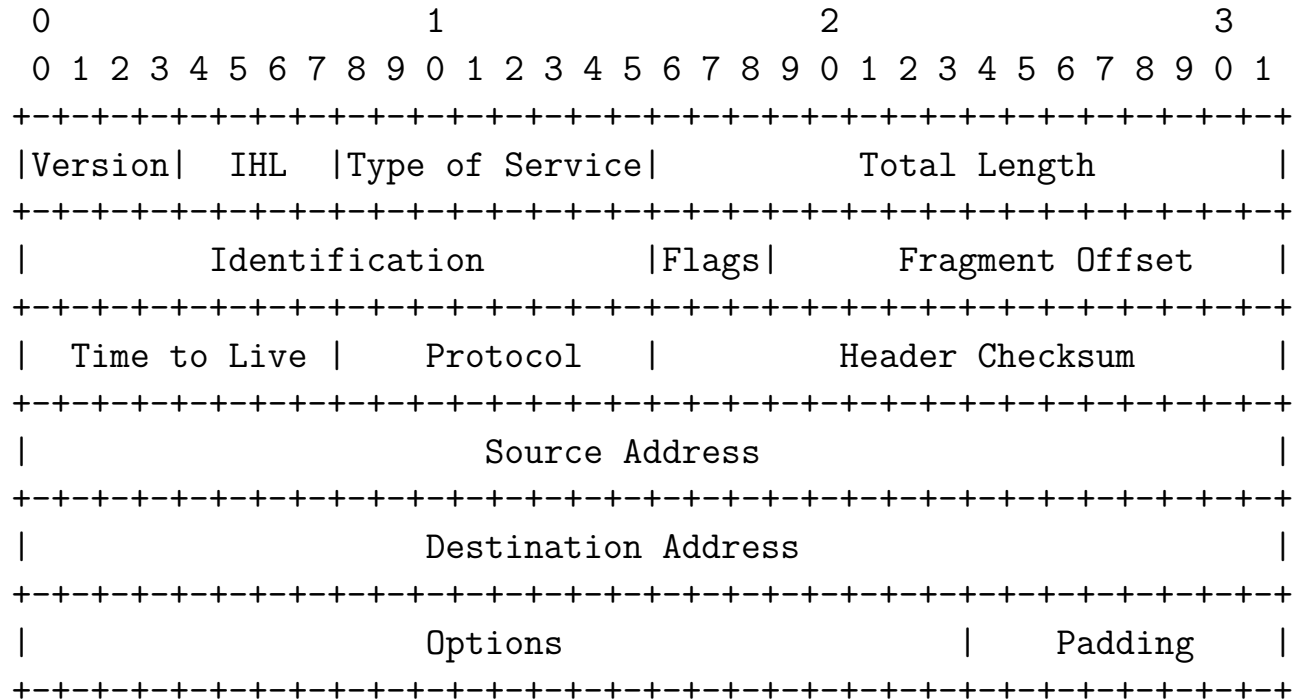


```
+++++
|Version| Traffic Class |          Flow Label          |
+++++
|          Payload Length          | Next Header | Hop Limit |
+++++
|
+
|
+          Source Address          +
|
+
|
+++++
|
+
|
+          Destination Address      +
|
+
|
```

+++++



# IPv4 pakken - header - RFC-791



Example Internet Datagram Header

# IPv6 pakken - extension headers RFC-2460



Fuld IPv6 implementation indeholder:

- Hop-by-Hop Options
- Routing (Type 0)
- Fragment - fragmentering KUN i end-points!
- Destination Options
- Authentication
- Encapsulating Security Payload

Ja, IPsec er en del af IPv6!

# Placering af extension headers



+-----+-----+-----		
IPv6 header	Routing header	TCP header + data
Next Header =	Next Header =	
Routing	TCP	
+-----+-----+-----		

+-----+-----+-----+-----			
IPv6 header	Routing header	Fragment header	fragment of TCP
			header + data
Next Header =	Next Header =	Next Header =	
Routing	Fragment	TCP	
+-----+-----+-----+-----			



# IPv6 configuration - kom igang



Router bagved NAT skal blot kunne forwarder protokoltype 0x41

Cisco 677: `set nat entry add 10.1.2.3 0 41`

Teredo - the Shipworm er også en mulighed og benyttes aktivt på Windows Vista idag

Officiel IPv4 adresse kan bruges med 6to4 til at lave prefix og router

DNS nameserver anbefales!! tænk på om den skal svare IPv6 AAAA record OG svare over IPv6 sockets - er måske ikke nødvendigt

IPv6-only netværk er sikkert sjældne indtil videre men det er muligt at lave dem nu

Jeg bruger <http://www.sixxs.net> som har vejledninger til diverse operativsystemer

# IPv6 configuration - klienter



```
$ ping6 ::1
PING6(56=40+8+8 bytes) ::1 --> ::1
16 bytes from ::1, icmp_seq=0 hlim=64 time=0.254 ms
16 bytes from ::1, icmp_seq=1 hlim=64 time=0.23 ms
^C
--- ::1 ping6 statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.230/0.242/0.254 ms
```

Microsoft Windows XP `ipv6 install` fra kommandolinien eller brug kontrolpanelet

`ipv6` giver mulighed for at konfigurere tunnel svarer omtrent til 'ifconfig' på Unix

Migrering er vigtigt i IPv6! Hvis I aktiverer IPv6 nu på en router, vil I pludselig have IPv6 på alle klienter ;-)

Se evt. appendix F Enabling IPv6 functionality i <http://inet6.dk/thesis.pdf>

# ifconfig med ipv6 - Unix



Næsten ingen forskel på de sædvanlige kommandoer ifconfig, netstat,

```
root# ifconfig en1 inet6 2001:1448:81:beef::1
root# ifconfig en1
en1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    inet6 fe80::230:65ff:fe17:94d1 prefixlen 64 scopeid 0x5
    inet6 2001:1448:81:beef::1 prefixlen 64
    inet 169.254.32.125 netmask 0xffff0000 broadcast 169.254.255.255
    ether 00:30:65:17:94:d1
    media: autoselect status: active
    supported media: autoselect
```

Fjernes igen med:

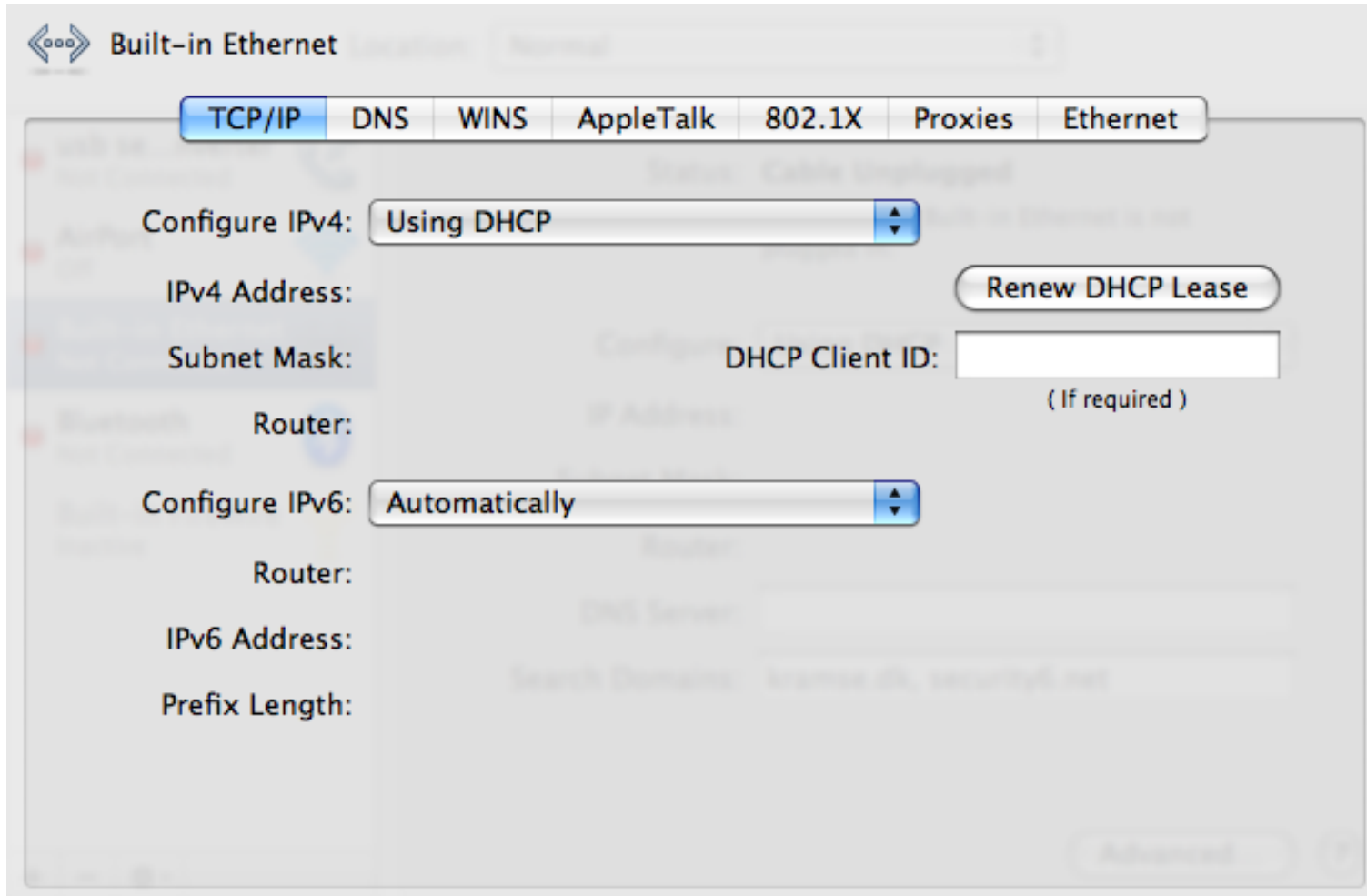
```
ifconfig en1 inet6 -alias 2001:1448:81:beef::1
```

Prøv også:

```
ifconfig en1 inet6
```

# GUI værktøjer - autoconfiguration





De fleste moderne operativsystemer er efterhånden opdateret med menuer til IPv6



# GUI værktøjer - manuel konfiguration



Built-in Ethernet

TCP/IP DNS WINS AppleTalk 802.1X Proxies Ethernet

Configure IPv4: Manually

IPv4 Address: 0.0.0.0

Subnet Mask:

Router:

Configure IPv6: Manually

Router:

IPv6 Address:

Prefix Length:

Advanced





Bemærk hvorledes subnetmaske nu blot er en prefix length



# ping til IPv6 adresser



```
root# ping6 ::1
PING6(56=40+8+8 bytes) ::1 --> ::1
16 bytes from ::1, icmp_seq=0 hlim=64 time=0.312 ms
16 bytes from ::1, icmp_seq=1 hlim=64 time=0.319 ms
^C
--- localhost ping6 statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.312/0.316/0.319 ms
```

Nogle operativsystemer kalder kommandoen ping6, andre bruger blot ping

# ping6 til global unicast adresse



```
root# ping6 2001:1448:81:beef:20a:95ff:fef5:34df
PING6(56=40+8+8 bytes) 2001:1448:81:beef::1 --> 2001:1448:81:beef:20a:95ff:fef5:34df
16 bytes from 2001:1448:81:beef:20a:95ff:fef5:34df, icmp_seq=0 hlim=64 time=10.639 ms
16 bytes from 2001:1448:81:beef:20a:95ff:fef5:34df, icmp_seq=1 hlim=64 time=1.615 ms
16 bytes from 2001:1448:81:beef:20a:95ff:fef5:34df, icmp_seq=2 hlim=64 time=2.074 ms
^C
--- 2001:1448:81:beef:20a:95ff:fef5:34df ping6 statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 1.615/4.776/10.639 ms
```

# ping6 til specielle adresser



```
root# ping6 -I en1 ff02::1
PING6(56=40+8+8 bytes) fe80::230:65ff:fe17:94d1 --> ff02::1
16 bytes from fe80::230:65ff:fe17:94d1, icmp_seq=0 hlim=64 time=0.483 ms
16 bytes from fe80::20a:95ff:fe5:34df, icmp_seq=0 hlim=64 time=982.932 ms
16 bytes from fe80::230:65ff:fe17:94d1, icmp_seq=1 hlim=64 time=0.582 ms
16 bytes from fe80::20a:95ff:fe5:34df, icmp_seq=1 hlim=64 time=9.6 ms
16 bytes from fe80::230:65ff:fe17:94d1, icmp_seq=2 hlim=64 time=0.489 ms
16 bytes from fe80::20a:95ff:fe5:34df, icmp_seq=2 hlim=64 time=7.636 ms
^C
--- ff02::1 ping6 statistics ---
4 packets transmitted, 4 packets received, +4 duplicates, 0% packet loss
round-trip min/avg/max = 0.483/126.236/982.932 ms
```

- ff02::1 ipv6-allnodes
- ff02::2 ipv6-allrouters
- ff02::3 ipv6-allhosts

## Stop - tid til leg



Der findes et trådløst netværk med IPv6

Join med en laptop og prøv at pinge lidt

1. Virker `ping6 ::1` eller `ping ::1`, fortsæt
2. Virker kommando svarende til: `ping6 -I en1 ff02::1`
  - burde vise flere maskiner
3. Kig på dine egne adresser med: `ipv6` (Windows) eller `ifconfig` (Unix)
4. Prøv at trace i netværket

Hvordan fik I IPv6 adresser?

# router advertisement daemon



```
/etc/rtadvd.conf:
```

```
en0:
```

```
    :addrs#1:addr="2001:1448:81:b00f::":prefixlen#64:
```

```
en1:
```

```
    :addrs#1:addr="2001:1448:81:beef::":prefixlen#64:
```

```
root# /usr/sbin/rtadvd -Df en0 en1
```

```
root# sysctl -w net.inet6.ip6.forwarding=1
```

```
net.inet6.ip6.forwarding: 0 -> 1
```

Stateless autoconfiguration er en stor ting i IPv6

Kommandoen starter den i debug-mode og i forgrunden

- normalt vil man starte den fra et script

Typisk skal forwarding aktiveres, som vist med BSD sysctl kommando

# IPv6 og andre services



```
root# netstat -an | grep -i listen
```

tcp46	0	0	*.80	*.*	LISTEN
tcp4	0	0	*.6000	*.*	LISTEN
tcp4	0	0	127.0.0.1.631	*.*	LISTEN
tcp4	0	0	*.25	*.*	LISTEN
tcp4	0	0	*.20123	*.*	LISTEN
tcp46	0	0	*.20123	*.*	LISTEN
tcp4	0	0	127.0.0.1.1033	*.*	LISTEN

ovenstående er udført på Mac OS X

# IPv6 output fra kommandoer - inet6 family



```
root# netstat -an -f inet6
```

Active Internet connections (including servers)

Proto	Recv	Send	Local	Foreign	(state)
tcp46	0	0	*.80	*.*	LISTEN
tcp46	0	0	*.22780	*.*	LISTEN
udp6	0	0	*.5353	*.*	
udp6	0	0	*.5353	*.*	
udp6	0	0	*.514	*.*	
icm6	0	0	*.*	*.*	
icm6	0	0	*.*	*.*	
icm6	0	0	*.*	*.*	

ovenstående er udført på Mac OS X og tilrettet lidt



# IPv6 er default for mange services



```
root# telnet localhost 80
```

```
Trying ::1...
```

```
Connected to localhost.
```

```
Escape character is '^['.
```

```
GET / HTTP/1.0
```

```
HTTP/1.1 200 OK
```

```
Date: Thu, 19 Feb 2004 09:22:34 GMT
```

```
Server: Apache/2.0.43 (Unix)
```

```
Content-Location: index.html.en
```

```
Vary: negotiate,accept-language,accept-charset
```

```
...
```

# IPv6 er default i OpenSSH



```
hlk$ ssh -v localhost -p 20123
OpenSSH_3.6.1p1+CAN-2003-0693, SSH protocols 1.5/2.0, OpenSSL 0x0090702f
debug1: Reading configuration data /Users/hlk/.ssh/config
debug1: Applying options for *
debug1: Reading configuration data /etc/ssh_config
debug1: Rhosts Authentication disabled, originating port will not be trusted.
debug1: Connecting to localhost [::1] port 20123.
debug1: Connection established.
debug1: identity file /Users/hlk/.ssh/id_rsa type -1
debug1: identity file /Users/hlk/.ssh/id_dsa type 2
debug1: Remote protocol version 2.0, remote software version OpenSSH_3.6.1p1+CAN-2003-0693
debug1: match: OpenSSH_3.6.1p1+CAN-2003-0693 pat OpenSSH*
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_3.6.1p1+CAN-2003-0693
```

# Apache access log



```
root# tail -f access_log
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /images/IPv6ready.png
HTTP/1.1" 304 0
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /images/valid-html401.png
HTTP/1.1" 304 0
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /images/snowflake1.png
HTTP/1.1" 304 0
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /~h1k/security6.net/images/logo-1.png
HTTP/1.1" 304 0
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:35 +0100]
"GET / HTTP/1.1" 200 1456
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:35 +0100]
"GET /apache_pb.gif HTTP/1.1" 200 2326
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:36 +0100]
"GET /favicon.ico HTTP/1.1" 404 209
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:36 +0100]
"GET /favicon.ico HTTP/1.1" 404 209
```

Apache konfigureres nemt til at lytte på IPv6

# Apache HTTPD server



Mange bruger HTTPD fra Apache projektet  
<http://httpd.apache.org> - netcraft siger omkring 70%  
konfigureres gennem httpd.conf

```
Listen 0.0.0.0:80
Listen [::]:80
...
Allow from 127.0.0.1
Allow from 2001:1448:81:0f:2d:9ff:f86:3f
Allow from 217.157.20.133
```

# OpenBSD fast IPv6 adresse



Netværkskonfiguration på OpenBSD - flere filer:

```
# cat /etc/hostname.sk0
inet 10.0.0.23 0xffffffff00 NONE
inet6 2001:1448:81:30::2
# cat /etc/mygate
10.0.0.1
# grep 2001 /etc/rc.local
route add -inet6 default 2001:1448:81:30::1
# cat /etc/resolv.conf
domain security6.net
lookup file bind
nameserver 212.242.40.3
nameserver 212.242.40.51
nameserver 2001:1448:81:30::10
```

# Basal DNS opsætning



```
domain security6.net
nameserver 212.242.40.3
nameserver 212.242.40.51
nameserver 2001:1448:81:30::2
```

/etc/resolv.conf angiver navneservere og søgedomæner  
typisk indhold er domænenavn og IP-adresser for navneservere  
Filen opdateres også automatisk på DHCP klienter

**Husk at man godt kan slå AAAA records op over IPv4**

NB: denne fil kan hedde noget andet på UNIX varianter!

eksempelvis /etc/netsvc.conf

# DNS systemet



Navneopslag på Internet - centralt for IPv6

Tidligere brugte man en **hosts** fil

hosts fil bruges stadig lokalt til serveren - IP-adresser

UNIX: /etc/hosts

Windows c:\windows\system32\drivers\etc\hosts

Eksempel: www.security6.net har adressen 217.157.20.131

skrives i database filer, zone filer

ns1	IN	A	217.157.20.130
	IN	AAAA	2001:618:433::1
www	IN	A	217.157.20.131
	IN	AAAA	2001:618:433::14

# Mere end navneopslag



består af resource records med en type:

- adresser A-records
- IPv6 adresser AAAA-records
- autoritative navneservere NS-records
- post, mail-exchanger MX-records
- flere andre: md , mf , cname , soa , mb , mg , mr , null , wks , ptr , hinfo , minfo , mx ....

IN	MX	10	mail.security6.net.
IN	MX	20	mail2.security6.net.



# BIND DNS server



Berkeley Internet Name Daemon server

Mange bruger BIND fra Internet Systems Consortium - altså Open Source

konfigureres gennem `named.conf`

det anbefales at bruge BIND version 9

- *DNS and BIND*, Paul Albitz & Cricket Liu, O'Reilly, 4th edition Maj 2001
- *DNS and BIND cookbook*, Cricket Liu, O'Reilly, 4th edition Oktober 2002

Kilde: <http://www.isc.org>

# BIND konfiguration - et udgangspunkt



```
acl internals { 127.0.0.1; ::1; 10.0.0.0/24; };
options {
    // the random device depends on the OS !
    random-device "/dev/random"; directory "/namedb";
    listen-on-v6 any;
    port 53; version "Dont know"; allow-query { any; };
};
view "internal" {
    match-clients { internals; }; recursion yes;
    zone "." {
        type hint; file "root.cache"; };
    // localhost forward lookup
    zone "localhost." {
        type master; file "internal/db.localhost"; };
    // localhost reverse lookup from IPv4 address
    zone "0.0.127.in-addr.arpa" {
        type master; file "internal/db.127.0.0"; notify no; };
};
```

...  
}



# Routing forståelse - IPv6



```
$ netstat -f inet6 -rn
```

Routing tables

Internet6:

Destination	Gateway	Flags	Netif
default	fe80::200:24ff:fec1:58ac	UGc	en0
::1	::1	UH	lo0
2001:1448:81:cf0f::/64	link#4	UC	en0
2001:1448:81:cf0f::1	0:0:24:c1:58:ac	UHLW	en0
fe80::/64	fe80::1	Uc	lo0
fe80::1	link#1	UHL	lo0
fe80::/64	link#4	UC	en0
fe80::20d:93ff:fe28:2812	0:d:93:28:28:12	UHL	lo0
fe80::/64	link#5	UC	en1
fe80::20d:93ff:fe86:7c3f	0:d:93:86:7c:3f	UHL	lo0
ff01::/32	::1	U	lo0
ff02::/32	::1	UC	lo0

ff02::/32  
ff02::/32

link#4  
link#5

UC  
UC

en0  
en1



# IPv6 neighbor discovery protocol (NDP)



OSI	IPv4	IPv6
Network	IP / ICMP	IPv6 / ICMPv6
Link	ARP	
Physical	Physical	Physical

ARP er væk

NDP erstatter og udvider ARP, Sammenlign arp -an med ndp -an

Til dels erstatter ICMPv6 således DHCP i IPv6, DHCPv6 findes dog

**NB: bemærk at dette har stor betydning for firewallregler!**



# ARP vs NDP



```
hlk@bigfoot:basic-ipv6-new$ arp -an
? (10.0.42.1) at 0:0:24:c8:b2:4c on en1 [ethernet]
? (10.0.42.2) at 0:c0:b7:6c:19:b on en1 [ethernet]
hlk@bigfoot:basic-ipv6-new$ ndp -an
```

Neighbor	Linklayer Address	Netif	Expire	St	Flgs	Prbs
::1	(incomplete)	lo0	permanent	R		
2001:16d8:ffd2:cf0f:21c:b3ff:fec4:e1b6	0:1c:b3:c4:e1:b6	en1	permanent	R		
fe80::1%lo0	(incomplete)	lo0	permanent	R		
fe80::200:24ff:fec8:b24c%en1	<b>0:0:24:c8:b2:4c</b>	en1	8h54m51s	S	R	
fe80::21c:b3ff:fec4:e1b6%en1	0:1c:b3:c4:e1:b6	en1	permanent	R		



# Fremtiden er nu



Det er sagt mange gange at nu skal vi igang med IPv6

Der er sket store ændringer fra starten af 2007 til nu

Hvor det i starten af 2007 var status quo er flere begyndt at presse på

Selv på version2.dk omtales det <http://www.version2.dk/artikel/6147>

*Seks DNS-rodservere tænder for IPv6 ICANN har nu aktiveret IPv6 på seks af internettets 13 rodservere. Med det rigtige udstyr kan man nu køre helt uden om IPv4.*

# Hvorfor implementere IPv6 i jeres netværk?



## Addresserummet

- end-to-end transparency
- nemmere administration

## Autoconfiguration

- stateless autoconfiguration
- automatisk routerconfiguration! (router renumbering)

## Performance

- simplere format
- kortere behandlingstid i routere

## Fleksibilitet - generelt

## Sikkerhed

- IPsec er et krav!
- Afsender IP-adressen ændres ikke igennem NAT!

# Hvorfor migrere til IPv6?



IPv4 er mere end 25 år gammel - fra 1981 til idag

Idag har folk ønsker/krav til kommunikationen

- båndbredde
- latency
- Quality-of-service
- sikkerhed

Meget af dette er, eller kan, implementeres med IPv4 - men det bliver lappeløsninger

NB: IPv6 er designet til at løse SPECIFIKKE problemer

# The Internet has done this before!



Because all hosts can not be converted to TCP simultaneously, and some will implement only IP/TCP, it will be necessary to provide temporarily for communication between NCP-only hosts and TCP-only hosts. To do this certain hosts which implement both NCP and IP/TCP will be designated as relay hosts. These relay hosts will support Telnet, FTP, and Mail services on both NCP and TCP. These relay services will be provided beginning in November 1981, and will be fully in place in January 1982.

Initially there will be many NCP-only hosts and a few TCP-only hosts, and the load on the relay hosts will be relatively light. As time goes by, and the conversion progresses, there will be more TCP capable hosts, and fewer NCP-only hosts, plus new TCP-only hosts. But, presumably most hosts that are now NCP-only will implement IP/TCP in addition to their NCP and become "dual protocol" hosts. So, while the load on the relay hosts will rise, it will not be a substantial portion of the total traffic.

NCP/TCP Transition Plan November 1981 RFC-801

## Er IPv6 klar? - Korte svar - ja



Det bruges idag aktivt, især i dele af verden der ikke har store dele af v4 adresserummet

Kernen af IPv6 er stabil

IPv6 er inkluderet i mange operativsystemer idag

AIX, Solaris, BSD'erne, Linux, Mac OS X og Windows XP Cisco IOS, Juniper Networks

Juniper har haft hardware support for IPv6 i mange år!

IPv6 TCP/IP stakke til indlejrede systemer er klar

prøv at lave `ping6 ::1` på jeres maskiner - det er IPv6

Se listen over IPv6 implementationer på <http://playground.sun.com/ipv6/ipng-implementations.html>

# IPv6 bruges idag



Listen over brugere vokser konstant

Store nye netværk designes alle med IPv6 en liste kan eksempelvis ses på adressen: <http://www.ipv6.ac.uk/gtpv6/eu.html>

Andre links kan vise statistik for internet og IPv4/IPv6

<http://www.bgpexpert.com/addrspace2007.php>

<https://wiki.caida.org/wiki/iic/bin/view/Main/WebHome>

<http://bgp.he.net/ipv6-progress-report.cgi>

Se også: <http://www.eu.ipv6tf.org/>

## 5 dårlige argumenter for ikke at bruge IPv6 nu



Det er ikke færdigt

- IPv4 har ALDRIG været færdigt ;-)

Ikke nødvendigt

- man kan stikke hovedet i busken

NAT løser alle problemer og er meget sikkert ...

- NAT er en lappeløsning

Udskiftning af HELE infrastrukturen er for dyrt

- man opgraderer/udskifter jævnligt udstyr

Vent til det er færdigt!

- man mister muligheden for at påvirke resultatet!

# Exercise



Now lets do the exercise

??

which is number ?? in the exercise PDF.



# Exercise



Now lets do the exercise

??

which is number ?? in the exercise PDF.

## Færdig med IPv6



I resten af kurset vil vi ikke betragte IPv6 eller IPv4 som noget specielt

Vi vil indimellem bruge det ene, indimellem det andet

Alle subnets er konfigureret ens på IPv4 og IPv6

Subnets som i IPv4 hedder prefix.45 vil således i IPv6 hedde noget med prefix:45:

At have ens routing på IPv4 og IPv6 vil typisk IKKE være tilfældet i praksis

Man kan jo lige så godt forbedre netværket mens man går over til IPv6 :-)

# Nu skal vi til management og diagnosticering



Always check the spark plugs!

Når man skal spore fejl i netværk er det essentielt at starte fra bunden:

- Er der link?
- Er der IP-adresse?
- Er der route?
- Modtager systemet pakker
- Er der en returvej fra systemet! Den her kan snyde mange!
- Lytter serveren på den port man vil forbinde til, UDP/TCP

Hvis der ikke er link vil man aldrig få svar fra databasen/webserveren/postserveren

# Udtræk af netværkskonfigurationen



De vigtigste kommandoer til udtræk af netværkskonfigurationen:

- cat - til at vise tekstfiler
- ifconfig - interface configuration
- netstat - network statistics
- lsof - list open files

# Basale testværktøjer TCP - Telnet og OpenSSL



Telnet blev tidligere brugt til login og er en klartekst forbindelse over TCP

Telnet kan bruges til at teste forbindelsen til mange ældre serverprotokoller som benytter ASCII kommandoer

- `telnet mail.kramse.dk 25` laver en forbindelse til port 25/tcp
- `telnet www.kramse.dk 80` laver en forbindelse til port 80/tcp

Til krypterede forbindelser anbefales det at teste med openssl

- `openssl s_client -host www.kramse.dk -port 443`  
laver en forbindelse til port 443/tcp med SSL
- `openssl s_client -host mail.kramse.dk -port 993`  
laver en forbindelse til port 993/tcp med SSL

Med OpenSSL i client-mode kan services tilgås med samme tekstkommandoer som med telnet

# Basale testværktøjer UDP



UDP er lidt drilsk, for de fleste services er ikke *ASCII protokoller*

Der findes dog en række testprogrammer, a la ping

- nsping - name server ping
- dhcping - dhcp server ping
- ...

Derudover kan man bruge de sædvanlige programmer som host til navneopslag osv.

# IP netværkstuning



IP har eksisteret mange år

Vi har udskiftet langsomme forbindelser med hurtige forbindelser

Vi har udskiftet langsomme MHz maskiner med Quad-core GHz maskiner

IP var tidligere meget konservativt, for ikke at overbelaste modtageren

Billedet er en HP arbejdsstation med 19" skærm og en 60MHz HP PA-RISC processor





# Anbefalet netværkstuning - hvad skal tunes



Der er visse indstillinger som tidligere var standard, de bør idag slås fra  
En del er allerede tunet i nyere versioner af IP-stakkene, men check lige

Ideer til ting som skal slås fra:

- broadcast ICMP, undgå smurfing
- Source routing, kan måske omgå firewalls og filtre

Ideer til ting som skal slås til/ændres:

- Bufferstørrelser - hvorfor have en buffer på 65535 bytes på en maskine med 32GB ram?
- Nye funktioner som RFC-1323 TCP Extensions for High Performance

Det anbefales at finde leverandørens vejledning til hvad der kan tunes

# Netværkskonfiguration med sysctl



```
# tuning
net.inet.tcp.recvspace=65535
net.inet.tcp.sendspace=65535
net.inet.udp.recvspace=65535
net.inet.udp.sendspace=32768
# postgresql tuning
kern.seminfo.semmni=256
kern.seminfo.semmns=2048
kern.shminfo.shmmax=50331648
```

På mange UNIX varianter findes et specielt tuningsprogram, sysctl

Findes blandt andet på alle BSD'erne: FreeBSD, OpenBSD, NetBSD og Darwin/OSX

Ændringerne skrives ind i filen /etc/sysctl.conf

På Linux erstatter det til dels konfiguration med echo

```
echo 1 > /proc/net/ip/forwarding
```

På AIX benyttes kommandoen `network options no`



# Tuning



Hvad er flaskehalsen for programmet?

I/O bundet - en enkelt disk eller flere

CPU bundet - regnekraften

Netværket - 10Mbit half-duplex adapter

Memory - begynder systemet at *swappe* eller *thrash*

brug top og andre statistikprogrammer til at se disse data

# Måling af througput



Når der skal tunes er det altid nødvendigt med en baseline

Man kan ikke begynde at tune ud fra subjektive målinger

*Det kører langsomt, Svartiden er for høj*

Målinger der giver præcise tal er nødvendige, før og efter målinger!

Der findes et antal værktøjer til, blandt andet lperf

# Målinger med Iperf



```
hlk@fluffy:hlk$ iperf -s
```

```
-----  
Server listening on TCP port 5001
```

```
TCP window size: 64.0 KByte (default)  
-----
```

```
[ 4] local 10.0.42.23 port 5001 connected with 10.0.42.67 port 51148
```

```
[ 4]  0.0-10.2 sec  6.95 MBytes  5.71 Mbits/sec
```

```
[ 4] local 10.0.42.23 port 5001 connected with 10.0.42.67 port 51149
```

```
[ 4]  0.0-10.2 sec  7.02 MBytes  5.76 Mbits/sec
```

Ovenstående er set fra server, client kaldes med `iperf -c fluffy`

## Stop - vi prøver i fællesskab lperf



Vi prøver lige lperf sammen

hvis alle prøver samtidig giver det stor variation i resultaterne

# Apache benchmark og andre programmer



```
hlk@bigfoot:hlk$ ab -n 100 http://www.kramse.dk/  
This is ApacheBench, Version 2.0.41-dev <$Revision: 1.121.2.12 $> apache-2.0  
Copyright (c) 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Copyright (c) 2006 The Apache Software Foundation, http://www.apache.org/
```

```
Benchmarking www.kramse.dk (be patient)...
```

```
...
```

Der findes specialiserede værktøjer til mange protokoller

Eksempelvis følger der et apache benchmark med Apache HTTPD serveren

Mange andre værktøjer til at simulere flere samtidige brugere



# Apache Benchmark output - 1



```
Server Software:      Apache
Server Hostname:      www.kramse.dk
Server Port:          80

Document Path:        /
Document Length:       7547 bytes

Concurrency Level:     1
Time taken for tests:  13.84924 seconds
Complete requests:     100
Failed requests:        0
Write errors:           0
Total transferred:     778900 bytes
HTML transferred:      754700 bytes
Requests per second:   7.64 #/sec (mean)
Time per request:      130.849 ms (mean)
Time per request:      130.849 ms (mean, across all concurrent requests)
Transfer rate:         58.08 Kbytes/sec received
```

# Apache Benchmark output - 3



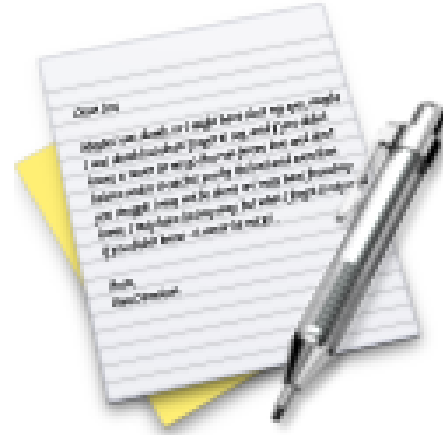
## Connection Times (ms)

	min	mean+/-sd	median	max
Connect:	22	24 4.0	24	58
Processing:	96	105 33.0	99	421
Waiting:	63	71 32.7	65	386
Total:	119	130 33.5	124	446

## Percentage of the requests served within a certain time (ms)

50%	124
66%	126
75%	128
80%	130
90%	143
95%	153
98%	189
99%	446
100%	446 (longest request)

# Exercise

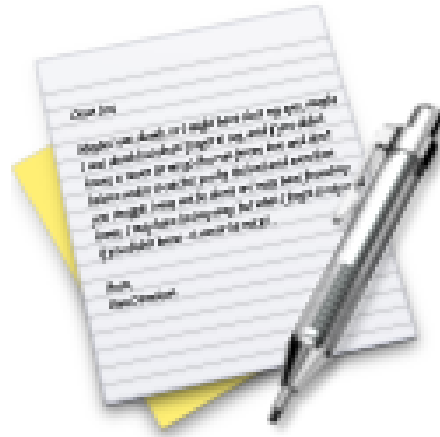


Now lets do the exercise

??

which is number ?? in the exercise PDF.

# Exercise

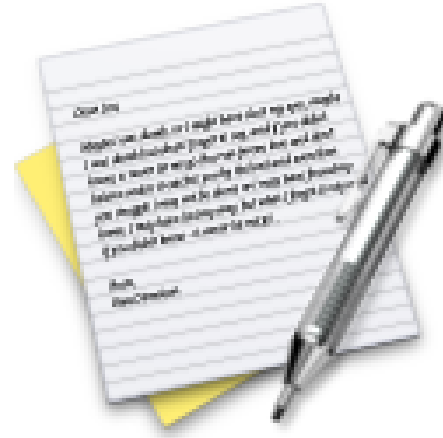


Now lets do the exercise

??

which is number ?? in the exercise PDF.

# Exercise



Now lets do the exercise

??

which is number ?? in the exercise PDF.

## Antal pakker per sekund



Til tider er det ikke båndbredden som sådan man vil måle

Specielt for routere er det vigtigt at de kan behandle mange pakker per sekund, pps

Til dette kan man lege med det indbyggede Ping program i flooding mode

Når programmet kaldes (som systemadministrator) med `ping -f server` vil den sende ping pakker så hurtigt som netkortet tillader

Programmer der kan teste pakker per sekund kaldes generelt for blaster tools

# traceroute



traceroute programmet virker ved hjælp af TTL

levetiden for en pakke tælles ned i hver router på vejen og ved at sætte denne lavt opnår man at pakken *timer ud* - besked fra hver router på vejen

default er UDP pakker, men på UNIX systemer er der ofte mulighed for at bruge ICMP

```
$ traceroute 217.157.20.129
```

```
traceroute to 217.157.20.129 (217.157.20.129),
```

```
30 hops max, 40 byte packets
```

```
1  safri (10.0.0.11)  3.577 ms  0.565 ms  0.323 ms
```

```
2  router (217.157.20.129)  1.481 ms  1.374 ms  1.261 ms
```

# traceroute - med UDP



```
# tcpdump -i en0 host 217.157.20.129 or host 10.0.0.11
tcpdump: listening on en0
23:23:30.426342 10.0.0.200.33849 > router.33435: udp 12 [ttl 1]
23:23:30.426742 safri > 10.0.0.200: icmp: time exceeded in-transit
23:23:30.436069 10.0.0.200.33849 > router.33436: udp 12 [ttl 1]
23:23:30.436357 safri > 10.0.0.200: icmp: time exceeded in-transit
23:23:30.437117 10.0.0.200.33849 > router.33437: udp 12 [ttl 1]
23:23:30.437383 safri > 10.0.0.200: icmp: time exceeded in-transit
23:23:30.437574 10.0.0.200.33849 > router.33438: udp 12
23:23:30.438946 router > 10.0.0.200: icmp: router udp port 33438 unreachable
23:23:30.451319 10.0.0.200.33849 > router.33439: udp 12
23:23:30.452569 router > 10.0.0.200: icmp: router udp port 33439 unreachable
23:23:30.452813 10.0.0.200.33849 > router.33440: udp 12
23:23:30.454023 router > 10.0.0.200: icmp: router udp port 33440 unreachable
23:23:31.379102 10.0.0.200.49214 > safri.domain: 6646+ PTR?

200.0.0.10.in-addr.arpa. (41)
23:23:31.380410 safri.domain > 10.0.0.200.49214: 6646 NXDomain* 0/1/0 (93)
14 packets received by filter
0 packets dropped by kernel
```



# Værdien af traceroute



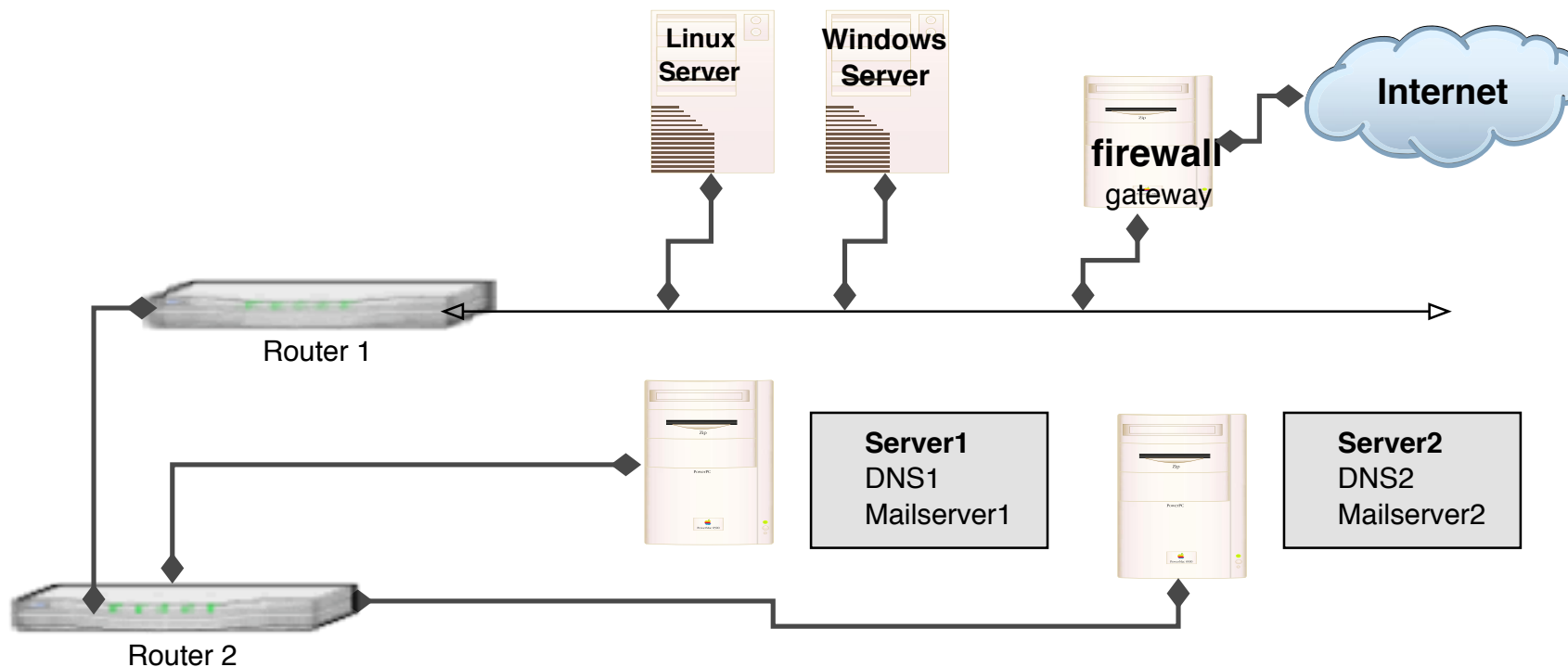
Diagnosticering af netværksproblemer - formålet med traceroute

Indblik i netværkets opbygning!

Svar fra hosts - en modtaget pakke fremfor et *sort hul*

Traceroute er ikke et angreb - det er også vigtigt at kunne genkende normal trafik!

# Network mapping



Ved brug af traceroute og tilsvarende programmer kan man ofte udlede topologien i det netværk

man undersøger



## Flere traceprogrammer



mtr My traceroute - grafisk <http://www.bitwizard.nl/mtr/>

lft - *layer four trace* benytter TCP SYN og FIN prober

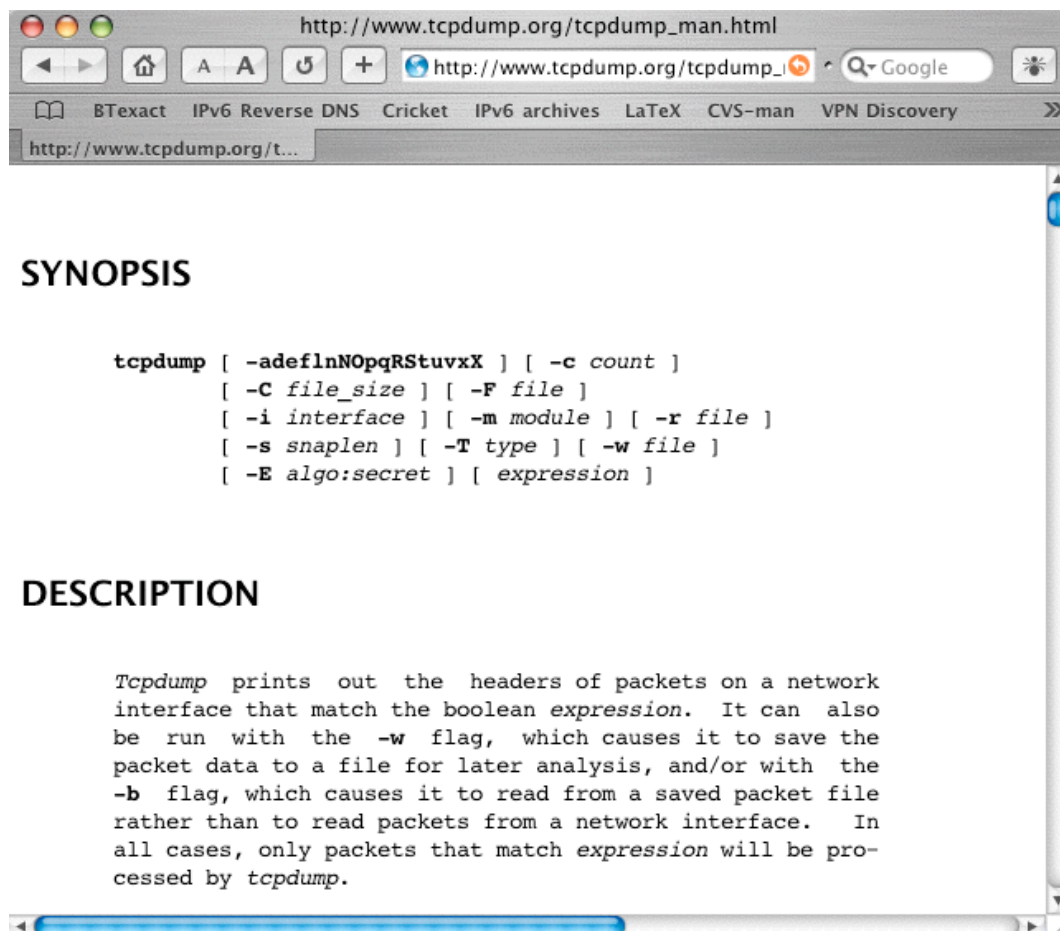
trace ved hjælp af TCP og andre protokoller findes

paratrace - *Parasitic Traceroute via Established TCP Flows and IPID Hopcount*

Der findes webservices hvor man kan trace fra, eksempelvis: <http://www.samspade.org>

# TCPDUMP - protokolanalyse pakkesniffer





<http://www.tcpdump.org> - både til Windows og UNIX

# tcpdump - normal brug



- tekstmode
- kan gemme netværkspakker i filer
- kan læse netværkspakker fra filer
- er de-facto standarden for at gemme netværksdata i filer

```
[root@otto hlk]# tcpdump -i en0
tcpdump: listening on en0
13:29:39.947037 fe80::210:a7ff:fe0b:8a5c > ff02::1: icmp6: router advertisement
13:29:40.442920 10.0.0.200.49165 > dns1.cybercity.dk.domain: 1189+[|domain]
13:29:40.487150 dns1.cybercity.dk.domain > 10.0.0.200.49165: 1189 NXDomain*+[|domain]
13:29:40.514494 10.0.0.200.49165 > dns1.cybercity.dk.domain: 24765+[|domain]
13:29:40.563788 dns1.cybercity.dk.domain > 10.0.0.200.49165: 24765 NXDomain*+[|domain]
13:29:40.602892 10.0.0.200.49165 > dns1.cybercity.dk.domain: 36485+[|domain]
13:29:40.648288 dns1.cybercity.dk.domain > 10.0.0.200.49165: 36485 NXDomain*+[|domain]
13:29:40.650596 10.0.0.200.49165 > dns1.cybercity.dk.domain: 4101+[|domain]
13:29:40.694868 dns1.cybercity.dk.domain > 10.0.0.200.49165: 4101 NXDomain*+[|domain]
13:29:40.805160 10.0.0.200 > mail: icmp: echo request
13:29:40.805670 mail > 10.0.0.200: icmp: echo reply
...
```

# TCPDUMP syntaks - udtryk



filtre til husbehov

- type - host, net og port
- src pakker med afsender IP eller afsender port
- dst pakker med modtager IP eller modtager port
- host - afsender eller modtager
- proto - protokol: ether, fddi, tr, ip, ip6, arp, rarp, decnet, tcp og udp

IP adresser kan angives som dotted-decimal eller navne

porte kan angives med numre eller navne

komplekse udtryk opbygges med logisk and, or, not



## tcpdump udtryk eksempler



Host 10.1.2.3

Alle pakker hvor afsender eller modtager er 10.1.2.3

host 10.2.3.4 and not host 10.3.4.5

Alle pakker til/fra 10.2.3.4 undtagen dem til/fra 10.3.4.5


- meget praktisk hvis man er logget ind på 10.2.3.4 via netværk fra 10.3.4.5

host foo and not port ftp and not port ftp-data

trafik til/fra maskine *foo* undtagen hvis det er FTP trafik


# Wireshark - grafisk pakkesniffer






Get Acquainted ▾Get Help ▾Develop ▾

Sharkfest '15Our SponsorWinPcap




Download

Get Started Now



Learn

Knowledge is Power



Enhance

With Riverbed Technology

News And Events


Join us at SHARKFEST '15!

SHARKFEST '15 will be held from June 22 – 25 at the Computer History Museum in Mountain View, CA.

Learn More ▸

Troubleshooting with Wireshark

By Laura Chappell  
Foreword by Gerald Combs  
Edited by Jim Aragon




This book focuses on the tips and techniques used to identify

Wireshark Blog


Cool New Stuff

Dec 17 | By Evan Huus




Wireshark 1.12 Officially Released!

Jul 31 | By Evan Huus



To Infinity and Beyond! Capturing Forever with Tshark

Jul 8 | By Evan Huus



More Blog Entries ▸

Enhance Wireshark

Riverbed is Wireshark's primary sponsor and provides our funding. They also make great products.


802.11 Packet Capture

WLAN packet capture and transmission

Full 802.11 a/b/g/n support

View management, control and data frames

Multi-channel aggregation (with multiple adapters)



Learn More ▸

Buy Now ▸

170

<http://www.wireshark.org>  
både til Windows og UNIX, tidligere kendt som Ethereal



# Programhygiejne!



**Download, installer - kør!** - farligt!

Sådan gøres det:

- download program OG signaturfil/MD5
- verificer signatur eller MD5
- installer
- brug programmet
- hold programmet opdateret!

Se eksempelvis teksten på hjemmesiden:

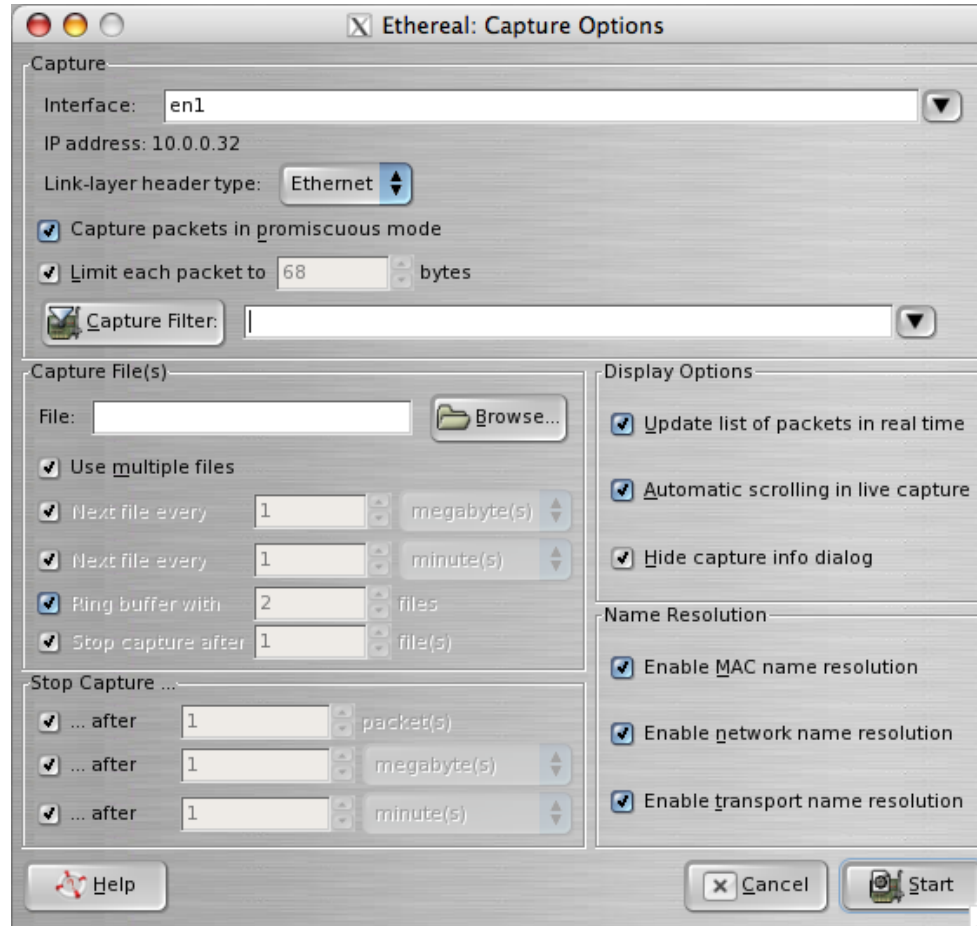
*Wireshark 0.99.2 has been released. Several security-related vulnerabilities have been fixed and several new features have been added.*

NB: ikke alle programmer har signaturer :(

MD5 er en envejs hash algoritme - mere om det senere

# Brug af Wireshark





Man starter med Capture - Options

# Brug af Wireshark





(Untitled) - Ethereal

File Edit View Go Capture Analyze Statistics Help

561 6.760947 10.0.0.32 sunny.kramse.dk TCP 54021 > imap [ACK] Seq=426 Ack=10773 Win=6535 Len=0

562 6.763144 sunny.kramse.dk 10.0.0.32 TLS Continuation Data, [Unreassembled Packet]

563 6.820037 10.0.0.32 sunny.kramse.dk TCP 54021 > imap [ACK] Seq=426 Ack=11106 Win=6535 Len=0

564 6.919635 10.0.0.32 sunny.kramse.dk TCP 54023 > imap [SYN] Seq=0 Ack=0 Win=65535 Len=0

565 6.921708 sunny.kramse.dk 10.0.0.32 TCP imap > 54023 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0

566 6.921794 10.0.0.32 sunny.kramse.dk TCP 54023 > imap [ACK] Seq=1 Ack=1 Win=65535 Len=0

567 6.922614 10.0.0.32 sunny.kramse.dk TLS Client Hello

▶ Frame 563 (66 bytes on wire, 66 bytes captured)

▶ Ethernet II, Src: AppleCom\_86:7c:3f (00:0d:93:86:7c:3f), Dst: Olicom\_c3:57:d8 (00:00:24:c3:57:d8)

▶ Internet Protocol, Src: 10.0.0.32 (10.0.0.32), Dst: sunny.kramse.dk (217.157.20.131)

▶ Transmission Control Protocol, Src Port: 54021 (54021), Dst Port: imap (993), Seq: 426, Ack: 11106, Len: 0

0000 00 00 24 c3 57 d8 00 0d 93 86 7c 3f 08 00 45 00 ..\$.W... ..|?..E.

0010 00 34 7e 8b 40 00 40 06 c3 f8 0a 00 00 20 d9 9d .4~. @. @. .... ..

0020 14 83 d3 05 03 e1 cd 31 c9 ea 0d 7b a2 bf 80 10 .....1 ...{....

0030 ff ff 32 0d 00 00 01 01 08 0a 62 e0 c3 42 bb e3 ..2..... ..b..B..

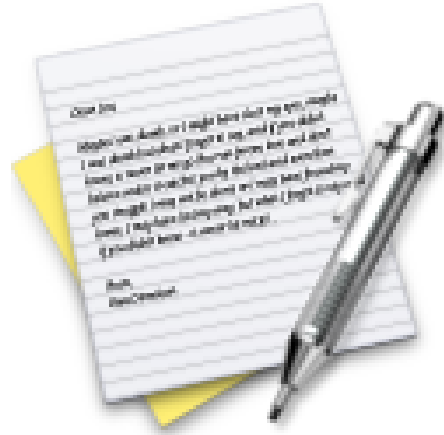
Filter: Expression... Clear Apply File: "/var/tmp/ether0ARkxt..."



Læg mærke til filtermulighederne



# Exercise



Now lets do the exercise

??

which is number ?? in the exercise PDF.



syslog er system loggen på UNIX og den er effektiv

- man kan definere hvad man vil se og hvor man vil have det dirigeret hen
- man kan samle det i en fil eller opdele alt efter programmer og andre kriterier
- man kan ligeledes bruge named pipes - dvs filer i filsystemet som tunneller fra chroot'ed services til syslog i det centrale system!
- man kan nemt sende data til andre systemer

Hvis man vil lave en centraliseret løsning er følgende link vigtigt:

Tina Bird, Counterpane

<http://loganalysis.org>

# syslogd.conf eksempel



```
*.err;kern.debug;auth.notice;authpriv.none;mail.crit    /dev/console
*.notice;auth,authpriv,cron,ftp,kern,lpr,mail,user.none /var/log/messages
kern.debug;user.info;syslog.info                        /var/log/messages
auth.info                                                /var/log/authlog
authpriv.debug                                           /var/log/secure
...
# Uncomment to log to a central host named "loghost".
#*.notice;auth,authpriv,cron,ftp,kern,lpr,mail,user.none @loghost
#kern.debug,user.info,syslog.info                        @loghost
#auth.info,authpriv.debug,daemon.info                   @loghost
```

## Andre syslogs syslog-ng



der findes andre syslog systemer eksempelvis syslog-ng  
konfigureres gennem `/etc/syslog-ng/syslog-ng.conf`  
Eksempel på indholdet af filen kunne være:

```
options {  
    long_hostnames(off);  
    sync(0);  
    stats(43200);  
};  
  
source src  unix-stream("/dev/log"); internal(); pipe("/proc/kmsg"); ;  
destination messages  file("/var/log/messages"); ;  
destination console_all  file("/dev/console"); ;  
log  source(src); destination(messages); ;  
log  source(src); destination(console_all); ;
```

# Exercise



Now lets do the exercise

??

which is number ?? in the exercise PDF.

# Logfiler og computer forensics



Logfiler er en nødvendighed for at have et transaktionsspor

Logfiler er desuden nødvendige for at fejlfinde

Det kan være relevant at sammenholde logfiler fra:

- routere
- firewalls
- intrusion detection systemer
- adgangskontrolsystemer
- ...

Husk - tiden er vigtig! Network Time Protocol (NTP) anbefales

Husk at logfilerne typisk kan slettes af en angriber - hvis denne får kontrol med systemet

# Simple Network Management Protocol



SNMP er en protokol der supporteres af de fleste professionelle netværksenheder, såsom switche, routere

hosts - skal slås til men følger som regel med

SNMP bruges til:

- *network management*
- statistik
- rapportering af fejl - SNMP traps

**sikkerheden baseres på community strings der sendes som klartekst ...**

det er nemmere at brute-force en community string end en brugerid/kodeord kombination





## Simple Network Management Protocol

sikkerheden afhænger alene af en Community string SNMPv2

typisk er den nem at gætte:

- public - default til at aflæse statistik
- private - default når man skal ændre på enheden, skrive
- cisco
- ...

Der findes lister og ordbøger på nettet over kendte default communities

## Systemer med SNMP



kan være svært at finde ... det er UDP 161

Hvis man finder en så prøv at bruge **snmpwalk** programmet - det kan vise alle tilgængelige SNMP oplysninger fra den pågældende host

det kan være en af måderne at identificere uautoriserede WLAN Access Points på - sweep efter port 161/UDP

snmpwalk er et af de mest brugte programmer til at hente snmp oplysninger - i forbindelse med hackning og penetrationstest

## snmpwalk



Typisk brug er:

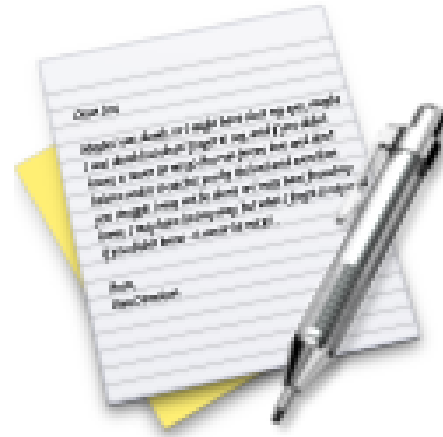
```
snmpwalk -v 1 -c secret switch1
```

```
snmpwalk -v 2c -c secret switch1
```

Eventuelt bruges snmpget og snmpset

Ovenstående er en del af Net-SNMP pakken, <http://net-snmp.sourceforge.net/>

# Exercise



Now lets do the exercise

??

which is number ?? in the exercise PDF.

# brute force



hvad betyder bruteforcing?  
afprøvning af alle mulighederne

Hydra v2.5 (c) 2003 by van Hauser / THC <vh@thc.org>

Syntax: hydra [[[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]]

[-o FILE] [-t TASKS] [-g TASKS] [-T SERVERS] [-M FILE] [-w TIME]

[-f] [-e ns] [-s PORT] [-S] [-vV] server service [OPT]

Options:

- S connect via SSL
- s PORT if the service is on a different default port, define it here
- l LOGIN or -L FILE login with LOGIN name, or load several logins from FILE
- p PASS or -P FILE try password PASS, or load several passwords from FILE
- e ns additional checks, "n" for null password, "s" try login as pass
- C FILE colon separated "login:pass" format, instead of -L/-P option
- M FILE file containing server list (parallizes attacks, see -T)
- o FILE write found login/password pairs to FILE instead of stdout

...

# Eksempler på SNMP og management



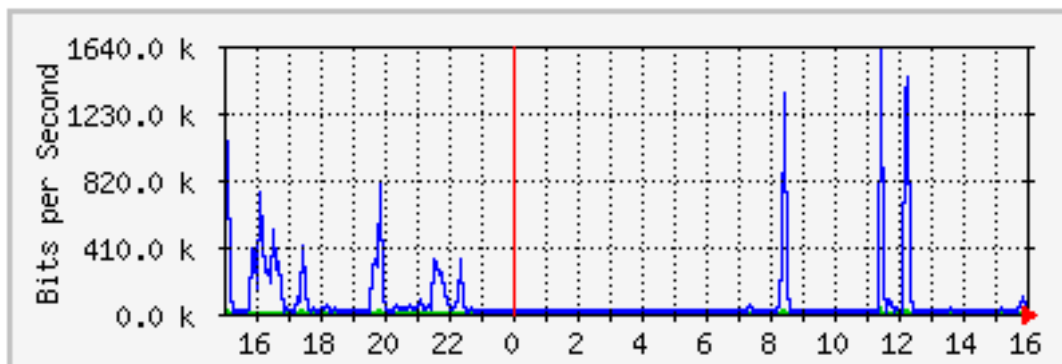
Ofte foregår administration af netværksenheder via HTTP, Telnet eller SSH

- små dumme enheder er idag ofte web-enabled
- bedre enheder giver både HTTP og kommandolinieadgang
- de bedste giver mulighed for SSH, fremfor Telnet

# Tobi Oetiker's MRTG The Multi Router Traffic Grapher



## 'Daglig' graf (5 minuts Middel)



	Max	Middel	Nu
Ind	35.5 kb/s (0.0%)	2392.0 b/s (0.0%)	5280.0 b/s (0.0%)
Ud	1604.6 kb/s (1.6%)	57.6 kb/s (0.1%)	51.4 kb/s (0.1%)

Monitorering af SNMP enheder og grafer

Inkluderer en nem configmaker og benytter idag RRDTool til data

Hjemmesiden: <http://oss.oetiker.ch/mrtg/>

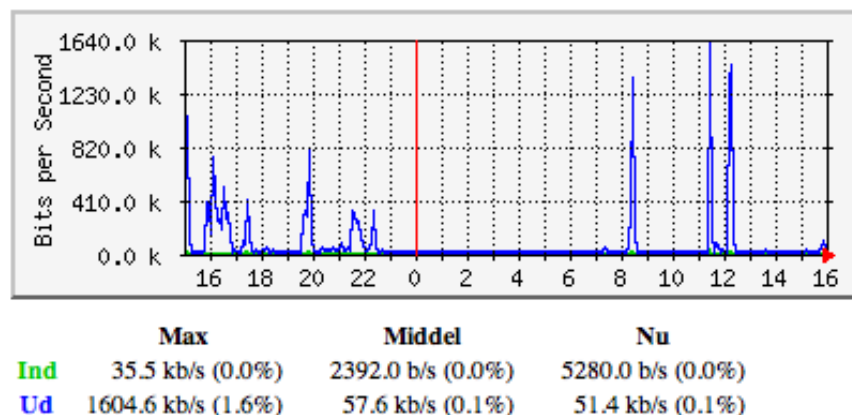




# RRDTool Round Robin Database Tool



**`Daglig' graf (5 minuts Middel)**



Round Robin Database Tool er en måde at gemme data på

Med RRDTool kan man derefter få lavet grafer

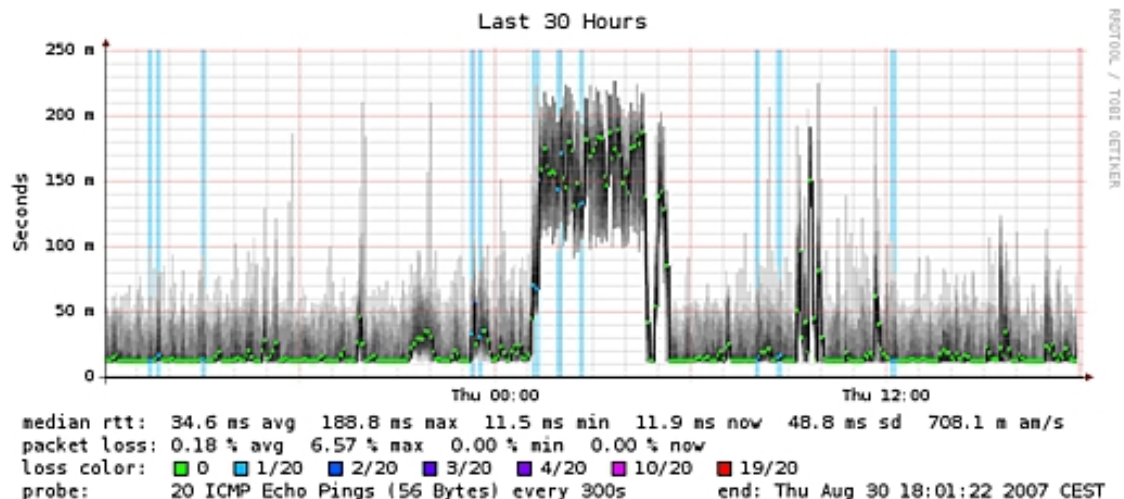
Typisk bruger man et andet værktøj som benytter RRDTool til data

<http://oss.oetiker.ch/rrdtool/doc/index.en.html>

Kan bruges til temperaturmålinger og alt muligt andet



# Smokeping



Måling af latency for netværksservice

Understøtter et stort antal prober: ICMP, DNS, HTTP, LDAP, SMTP, ...

Min SmokePing server <http://pumba.kramse.dk/smokeping/>

Hjemmesiden for SmokePing <http://oss.oetiker.ch/smokeping/>  
Lavet af Tobias Oetiker og Niko Tyni



# Nagios



Overvågningsværktøj der giver godt overblik

- Monitoring af diverse services (SMTP, POP3, HTTP, NNTP, PING, etc.)
- Monitoring af host resources (processor load, disk and memory usage, running processes, log files, etc.)
- Monitoring af andre ressourcer som temperatur
- Simpel plugin design som gør det nemt at udvide
- Kan sende e-mail, SMS m.v.

Benyttes mange steder

Hjemmesiden for Nagios <http://www.nagios.org/>

## Stop - overvågningsværktøjer



Brug lidt tid på at se på vores netværk

Valgfrit om I vil se på Administrationsinterface på switche, SNMP indstillinger eksempelvis

Eller Nagios og SmokePing på mine servere