





Welcome to

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

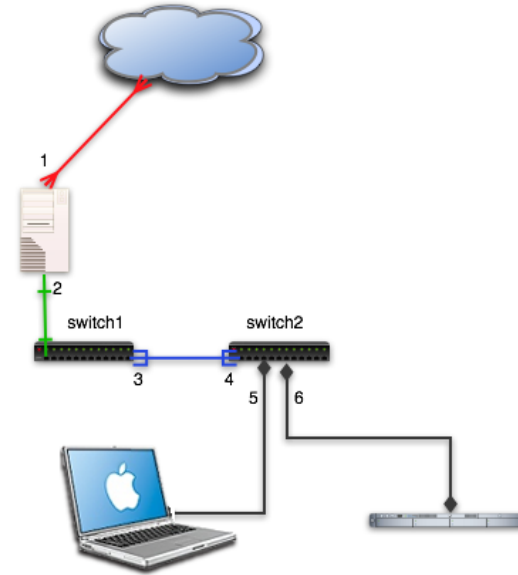
Communication and Network Security 2019

Henrik Lund Kramshøj hlk@zencurity.com @kramse  

Slides are available as PDF, kramse@Github

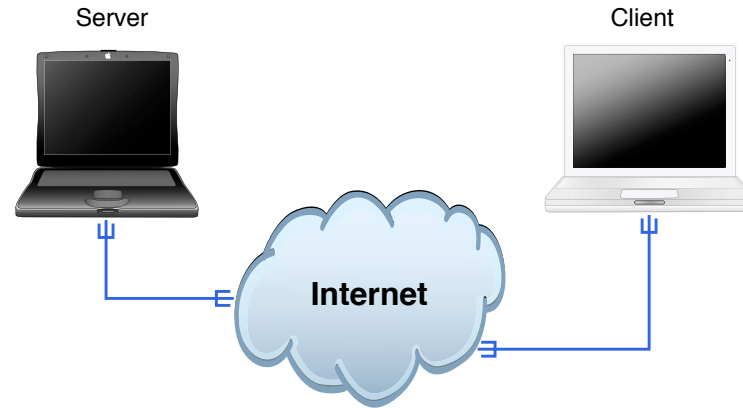
1-TCPIP-and-Security-in-TCPIP-protocol-suite.tex in the repo security-courses

# 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:

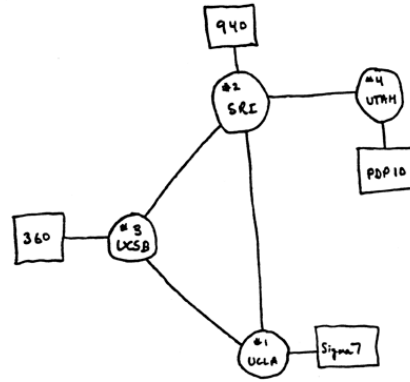
A good analogy for the development of the Internet is that of constantly renewing the individual streets and buildings of a city, rather than razing the city and rebuilding it. The architectural principles therefore aim to provide a framework for creating cooperation and standards, as a small "spanning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

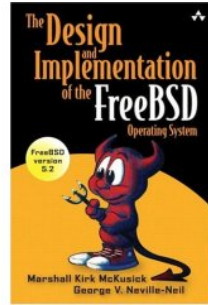
Paul Baran, 1964 *On distributed Communications* 12-bind serie af rapporter

<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!





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

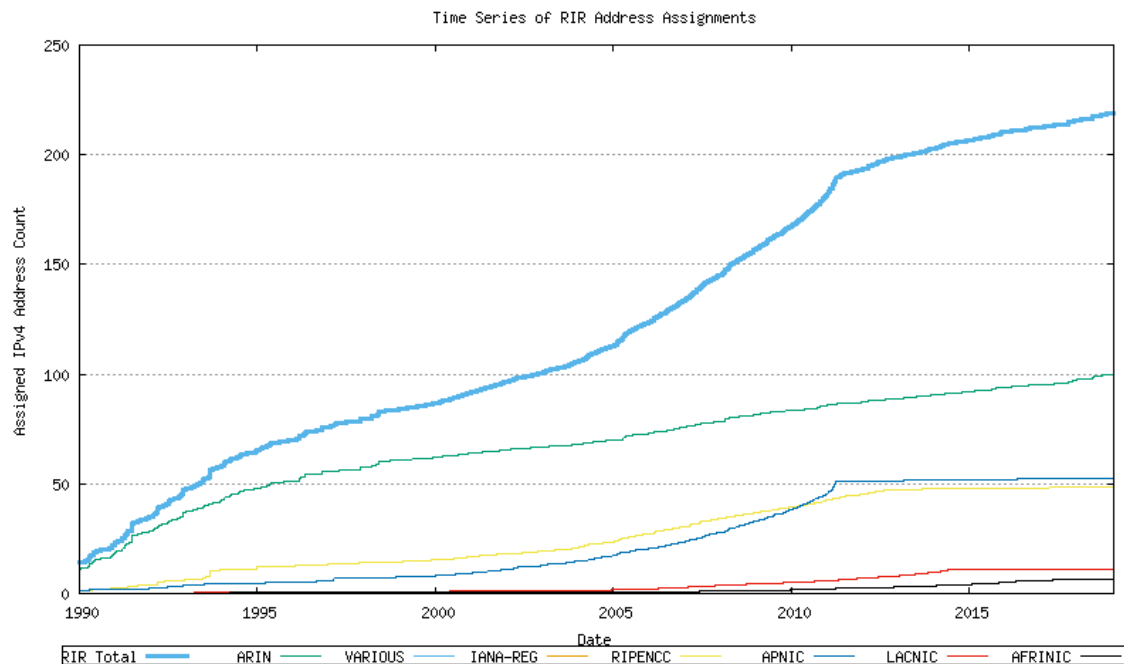
Bell Labs/AT&T var et telefonselskab - ikke et software hus

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

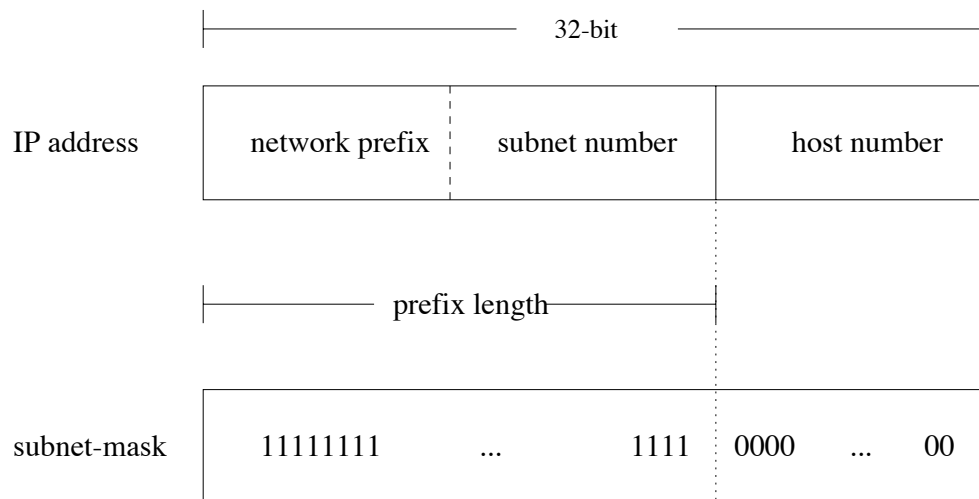
# Hvad er Internet hosts



Cumulative RIR address assignments, per RIR

Source: IPv4 Address Report - 28-Jan-2019 <http://www.potaroo.net/tools/ipv4/>

# 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)

**Stop saying C, say /24**

# CIDR Classless Inter-Domain Routing



Classful routing		Classless routing CIDR	
4 class C networks	Inherent subnet mask	Supernet	Subnet mask
192.0.8.0	255.255.255.0	192.0.8.0	255.255.252.0
192.0.9.0	255.255.255.0		252d=111111100b
192.0.10.0	255.255.255.0		
192.0.11.0	255.255.255.0	Base network/prefix	
		192.10.8.0/22	

Subnetmasker var oprindeligt indforstået

Man tildelte flere C-klasser - 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

# 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



# RFC-1918 Private Networks



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

The blocks 192.0.2.0/24 (TEST-NET-1), 198.51.100.0/24 (TEST-NET-2), and 203.0.113.0/24 (TEST-NET-3) are provided for use in documentation.

169.254.0.0/16 has been ear-marked as the IP range to use for end node auto-configuration when a DHCP server may not be found

# 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:

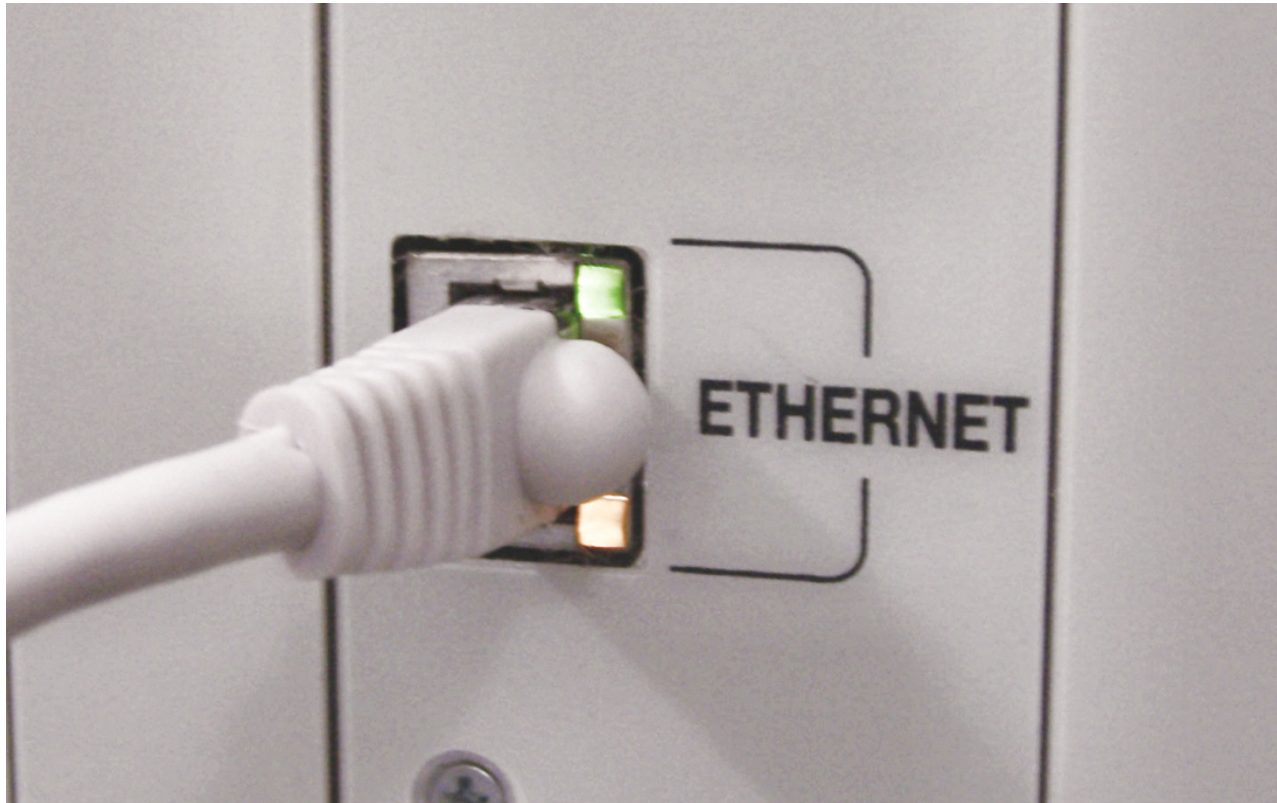
- Ethernet - varianter 10mbit, 100mbit, gigabit, 10G, 100G, 200G, 400G, ...
- Wireless 802.11 teknologier
- ADSL/ATM teknologier til WAN forbindelser
- MPLS ligeledes til WAN forbindelser

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

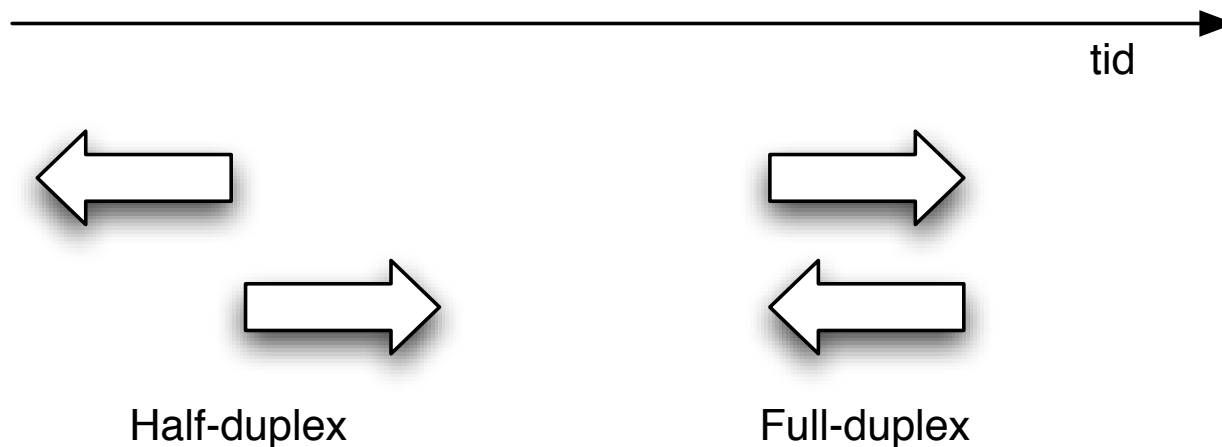
Typisk svarende til 48-bit MAC adresser som kendes fra Ethernet MAC-48/EUI-48

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?

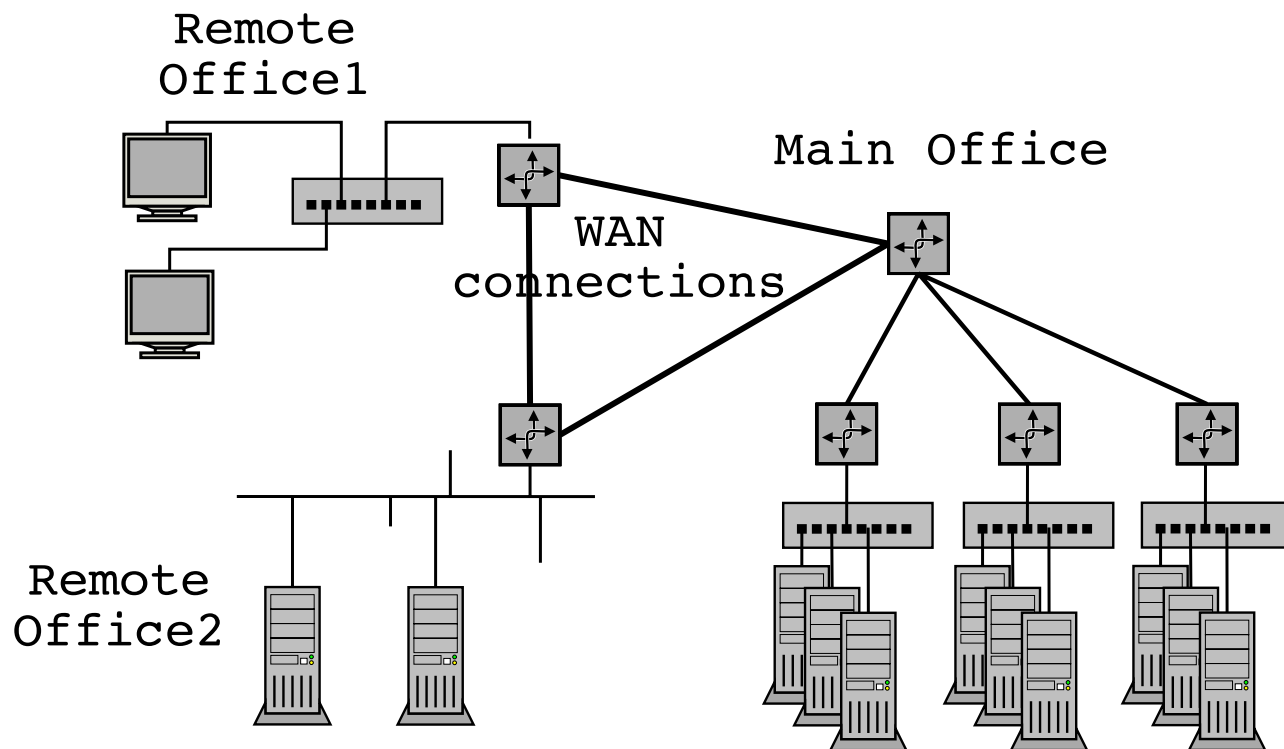
De fleste nyere Ethernet netkort kan køre i fuld-duplex

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

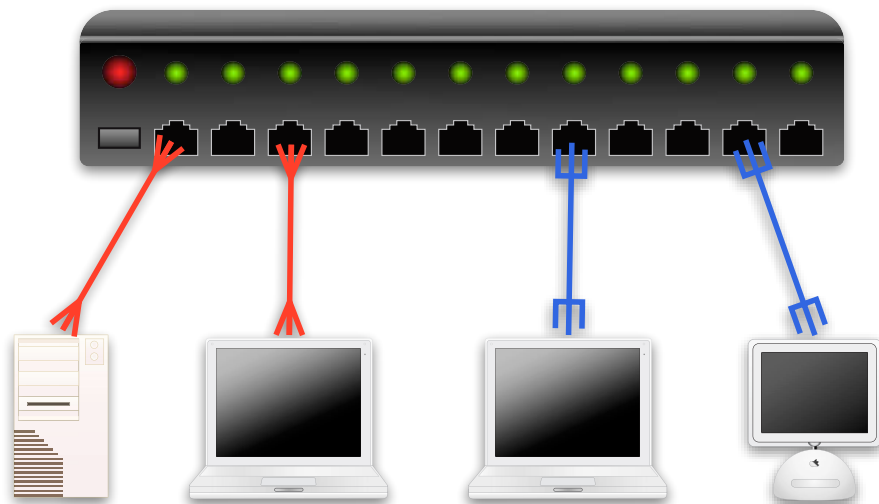
Den intelligente bro kender MAC adresserne på hver side

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

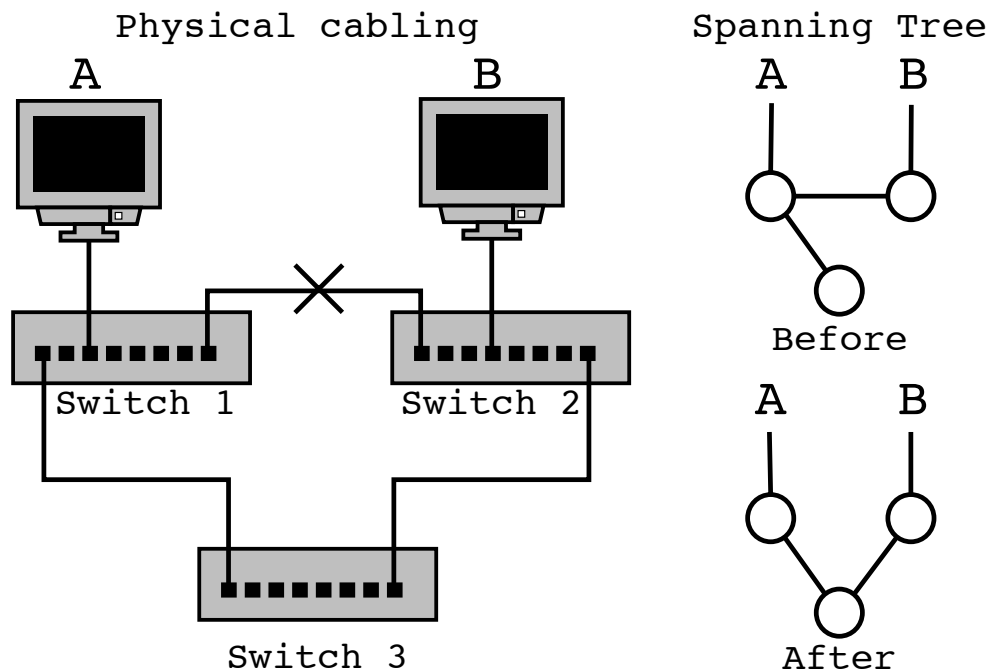


Ved at fortsætte udviklingen kunne man samle broer til en switch

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

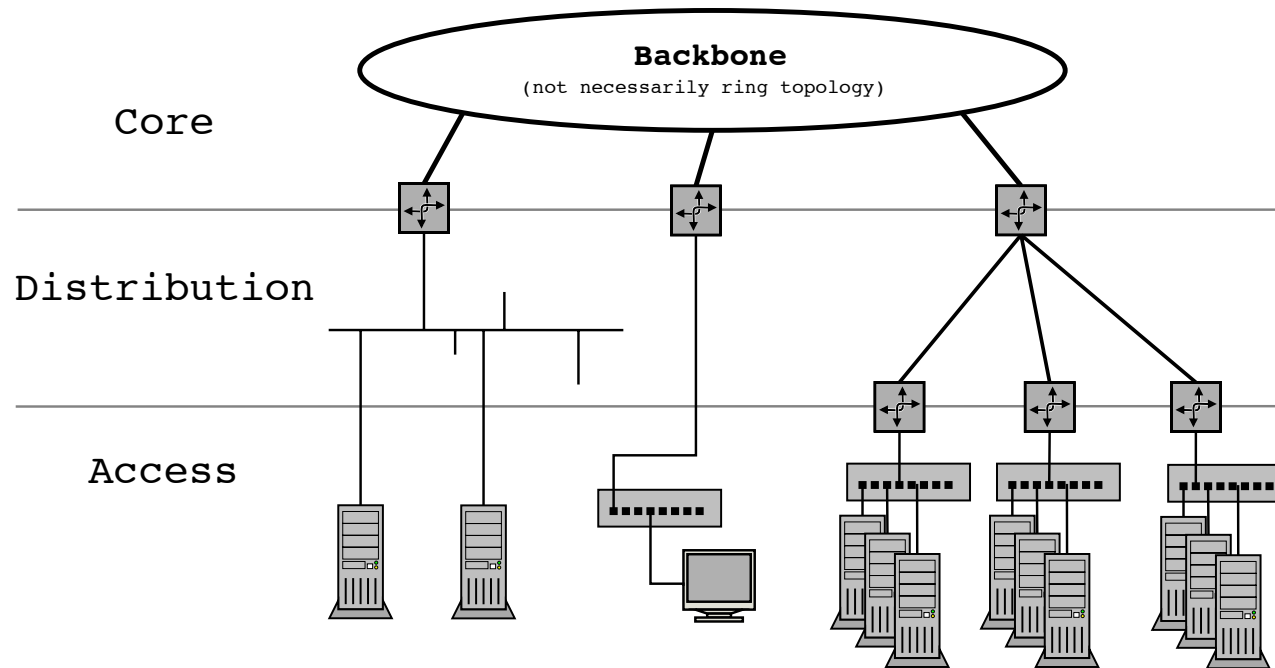
Bemærk performance begrænses af backplane i switchen

# Topologier og Spanning Tree Protocol



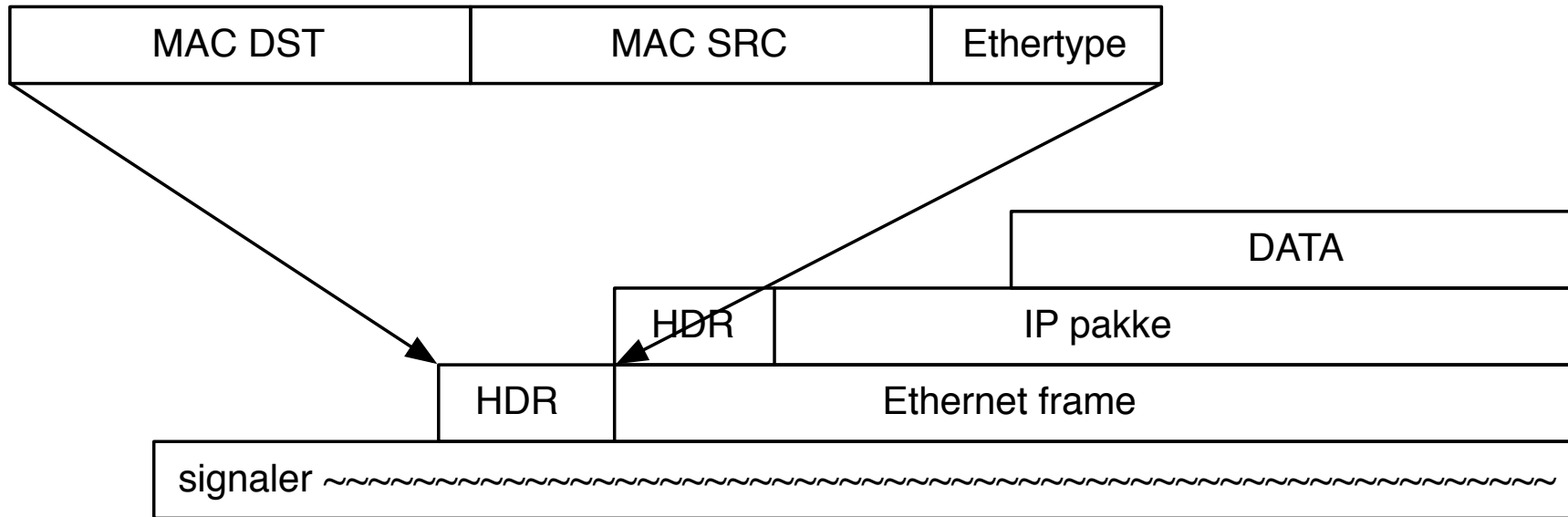
Se mere i bogen af Radia Perlman, *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols*

# Core, Distribution og Access net



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

# Pakker i en datastrøm

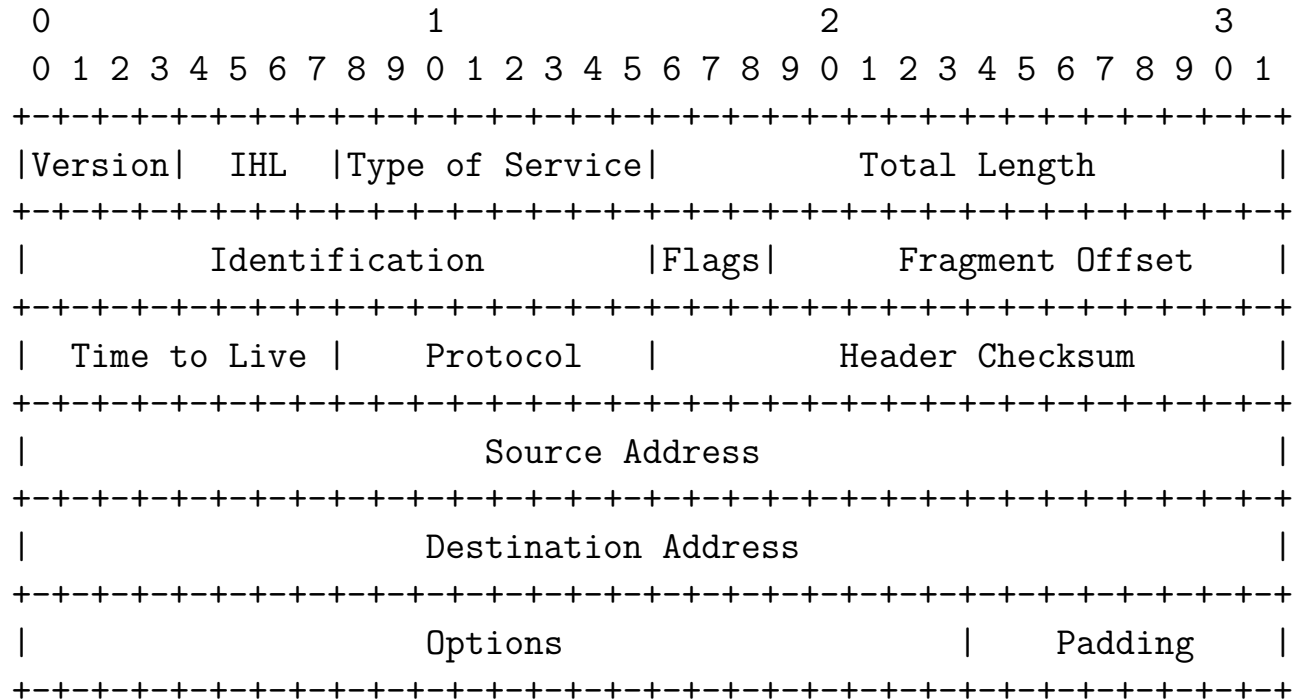


Ser vi data som en datastrøm er pakkerne blot et mønster lagt henover data

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



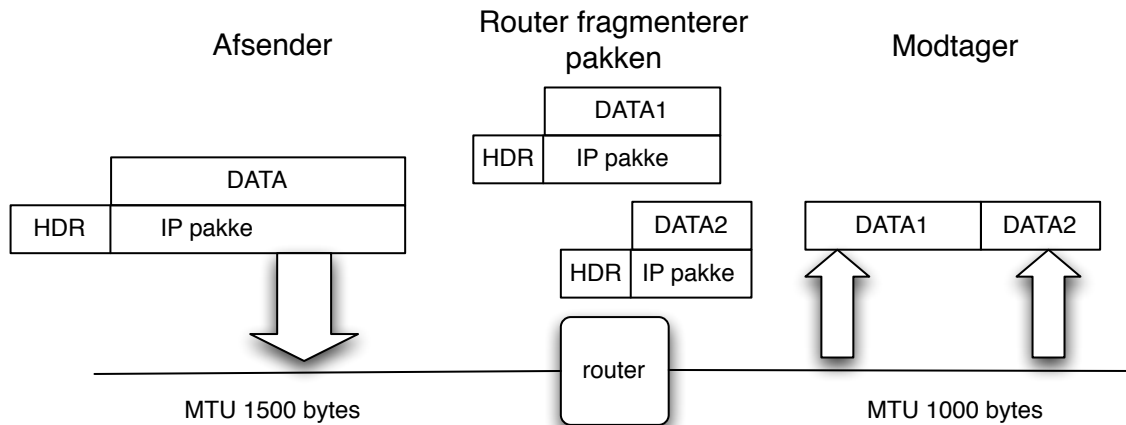
Fælles adresserum

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

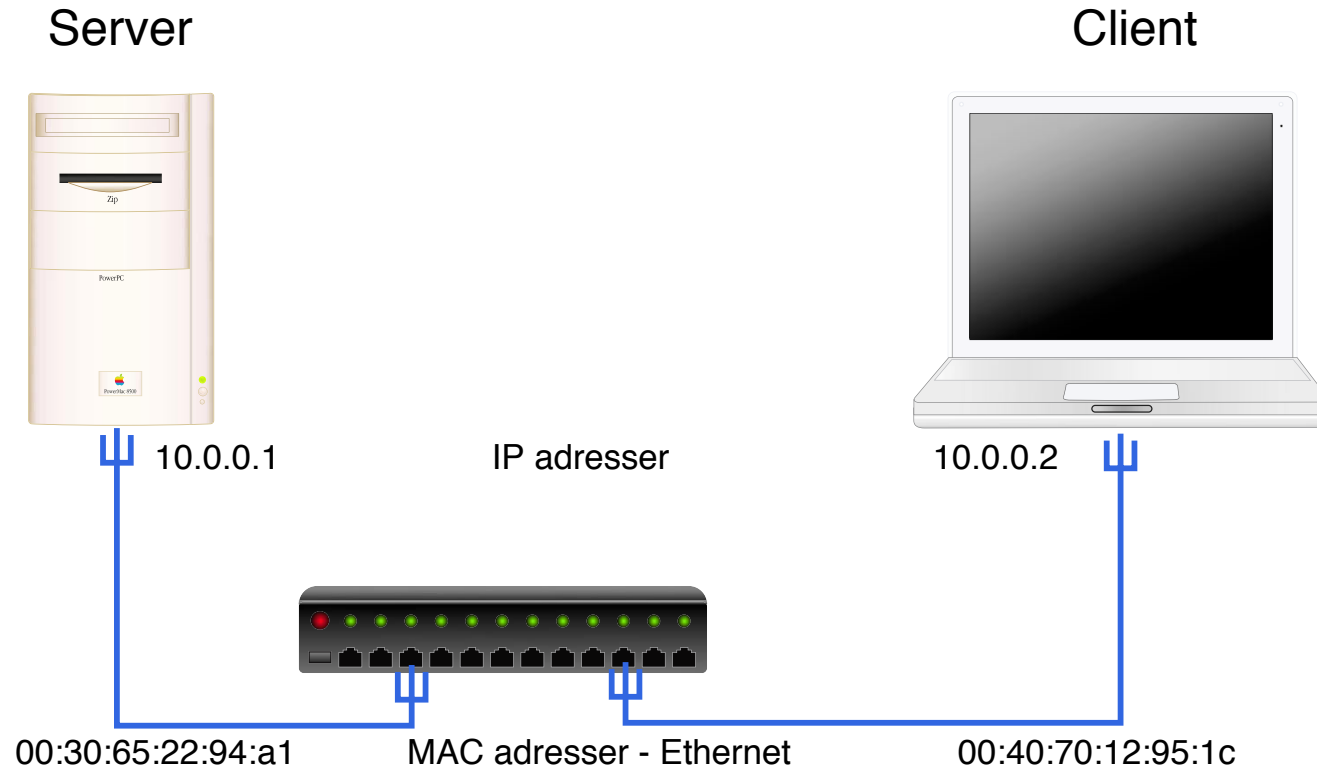
- 0 = net unreachable;
- 1 = host unreachable;
- 2 = protocol unreachable;
- 3 = port unreachable;
- 4 = fragmentation needed and DF set;
- 5 = source route failed.

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`

# 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

# Wireshark - grafisk pakkesniffer




[Get Acquainted](#) [Get Help](#) [Develop](#) [Sharkfest '15](#) [Our Sponsor](#) [WinPcap](#)

We're having a conference! You're invited!




## 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](#)

### 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.](#)

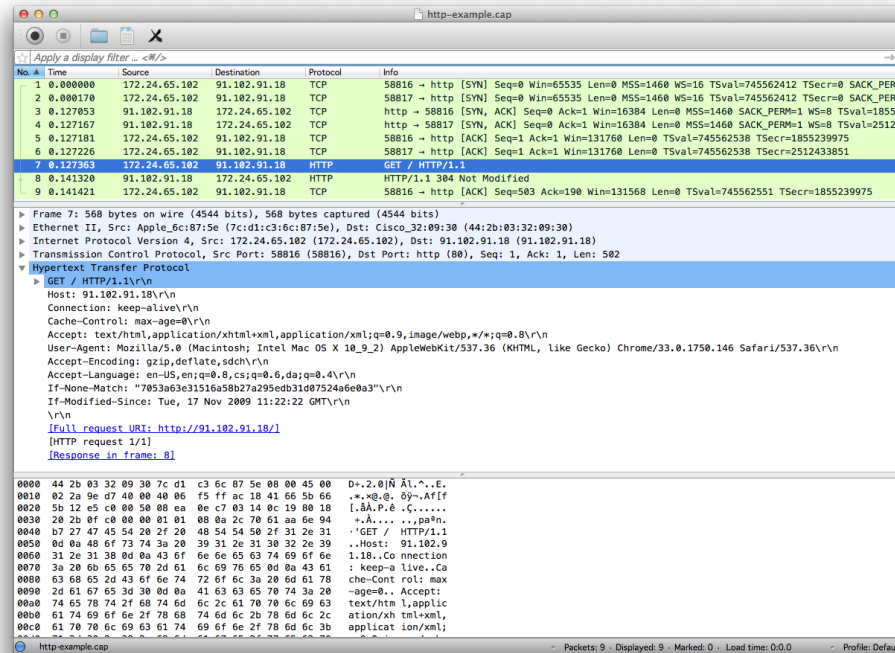
**802.11 Packet Capture**

- 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](#)

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

# Brug af Wireshark



Man starter med Capture - Options



# Brug af Wireshark



Secure Sockets Layer

- ▼ TLSv1.2 Record Layer: Handshake Protocol: Client Hello
  - Content Type: Handshake (22)
  - Version: TLS 1.0 (0x0301)
  - Length: 198
  - ▼ Handshake Protocol: Client Hello
    - Handshake Type: Client Hello (1)
    - Length: 194
    - Version: TLS 1.2 (0x0303)
    - ▶ Random
    - Session ID Length: 0
    - Cipher Suites Length: 32
    - ▶ Cipher Suites (16 suites)
    - Compression Methods Length: 1
    - ▶ Compression Methods (1 method)
    - Extensions Length: 121
    - ▶ Extension: Unknown 56026
    - ▶ Extension: renegotiation\_info
    - ▼ Extension: server\_name
      - Type: server\_name (0x0000)
      - Length: 16
      - ▼ Server Name Indication extension
        - Server Name list length: 14
        - Server Name Type: host\_name (0)
        - Server Name length: 11
        - Server Name: twitter.com
      - ▶ Extension: Extended Master Secret

0050	a4 1d 52 8f 2c 18 99 91 54 68 0a 77 0d 95 73 64	..R.,... Th.w..sd
0060	7d 00 00 20 5a 5a c0 2b c0 2f c0 2c c0 30 cc a9	}.. ZZ.+ ./.,.0..
0070	cc a8 cc 14 cc 13 c0 13 c0 14 00 9c 00 9d 00 2f	...../
0080	00 35 00 0a 01 00 00 79 da da 00 00 ff 01 00 01	.5.....y .....
0090	00 00 00 00 10 00 0e 00 00 0b 74 77 69 74 74 65	..... .twitter
00a0	72 2e 63 6d 00 17 00 00 00 23 00 00 00 0d 00	r.com... .#.....
00b0	14 00 12 04 03 08 04 04 01 05 03 08 05 05 01 08	.....

Læg også mærke til filtermulighederne

# Hardware IPv4 checksum offloading



IPv4 checksum skal beregnes hvergang man modtager en pakke

IPv4 checksum skal beregnes hvergang man sender en pakke

Lad en ASIC gøre arbejdet!

De fleste servernetkort tilbyder at foretage denne beregning på IPv4

IPv6 benytter ikke header checksum, det er unødvendigt

**NB:** kan resultere i at værktøjer siger checksum er forkert!

# Exercise



Now lets do the exercise

## Wireshark and Tcpdump 15 min

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 `tracert`

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.



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 zencurity.com
lookup file bind
nameserver 91.239.100.100
```

# 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 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
```

Linux vil gerne have man bruger ip address til at vise IP

# Vigtigste protokoller



ARP Address Resolution Protocol

IP og ICMP Internet Control Message Protocol

UDP User Datagram Protocol

TCP Transmission Control Protocol

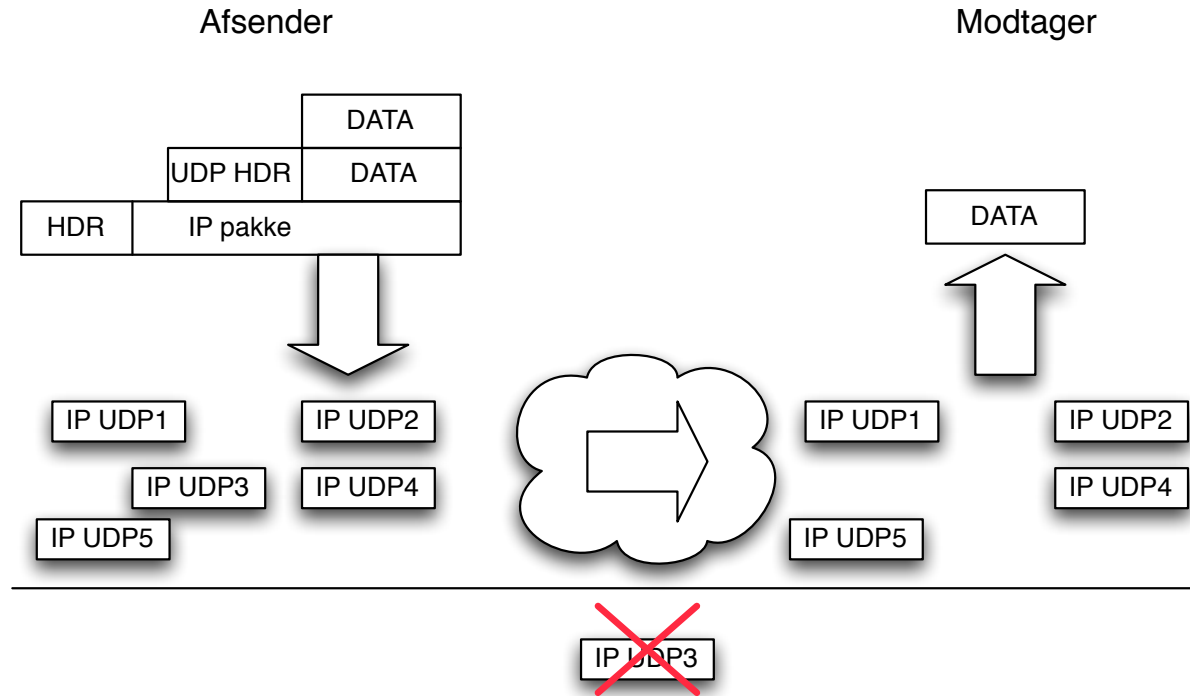
DHCP Dynamic Host Configuration Protocol

DNS Domain Name System

Ovenstående er omtrent minimumskrav for at komme på internet

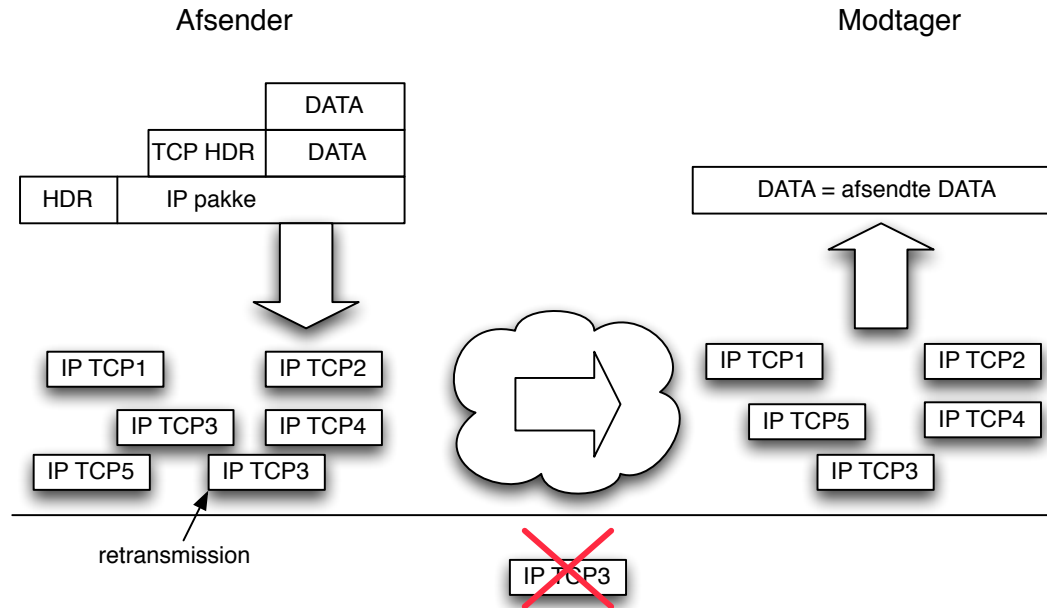


# UDP User Datagram Protocol



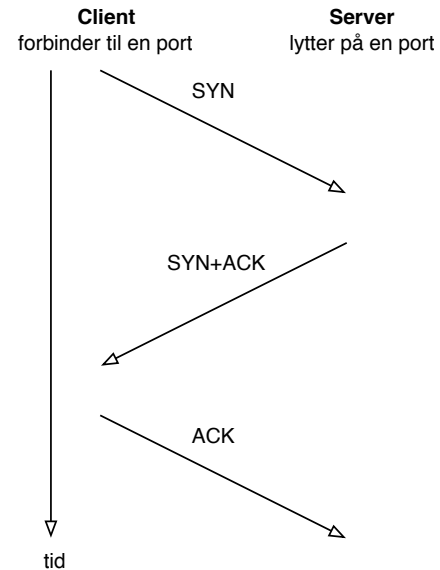
Forbindelsesløs RFC-768, *connection-less*

# TCP Transmission Control Protocol



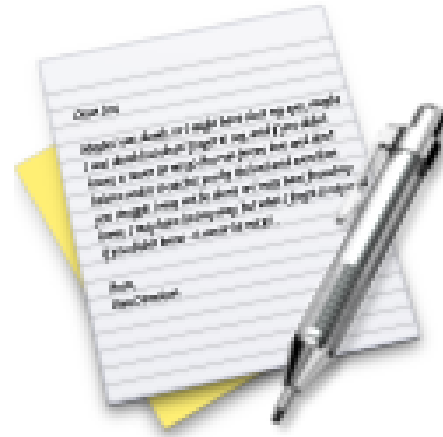
Forbindelsesorienteret RFC-791 September 1981, *connection-oriented*  
Enten overføres data eller man får fejlmeddelelse

# 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 10 min

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

# Well-known port numbers



IANA vedligeholder en liste over magiske konstanter i IP

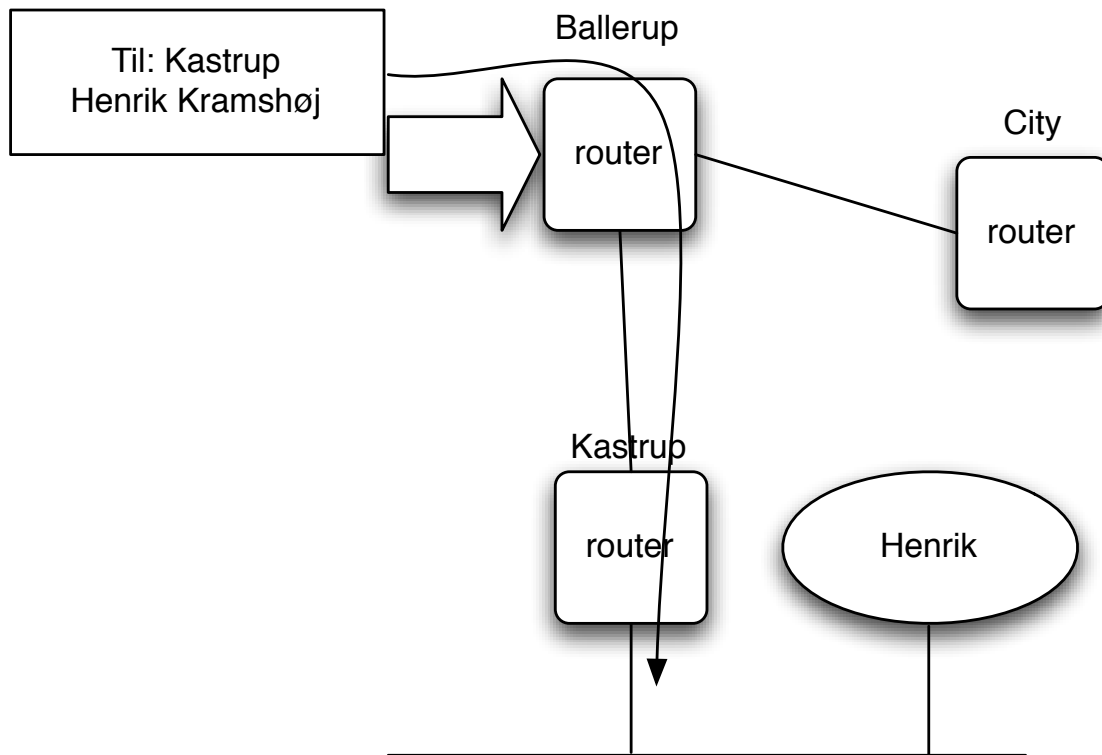
De har lister med hvilke protokoller har hvilke protokol ID m.v.

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

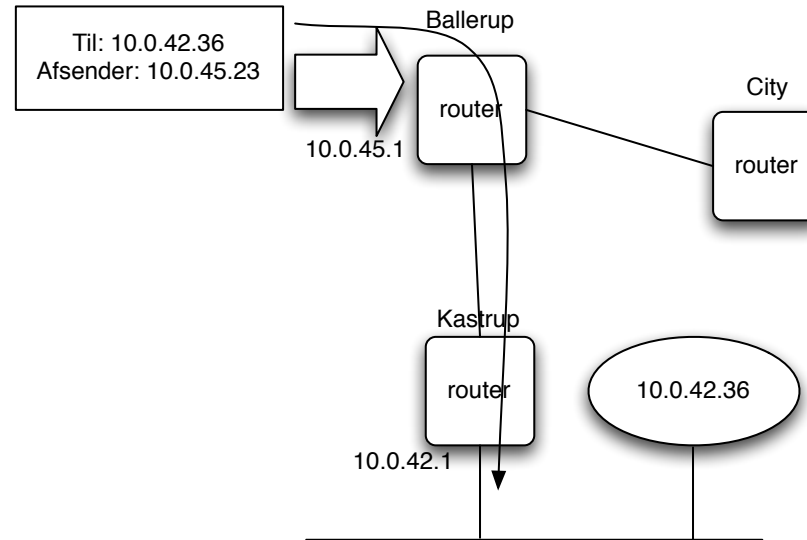
Se flere på <http://www.iana.org>

# Hierarkisk routing



Hvordan kommer pakkerne frem til modtageren

# IP default gateway

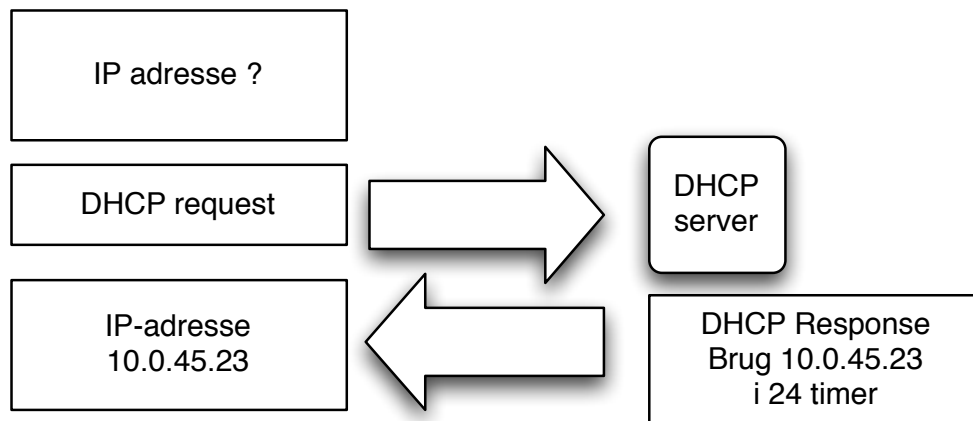


IP routing er nemt, longest match

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 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



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



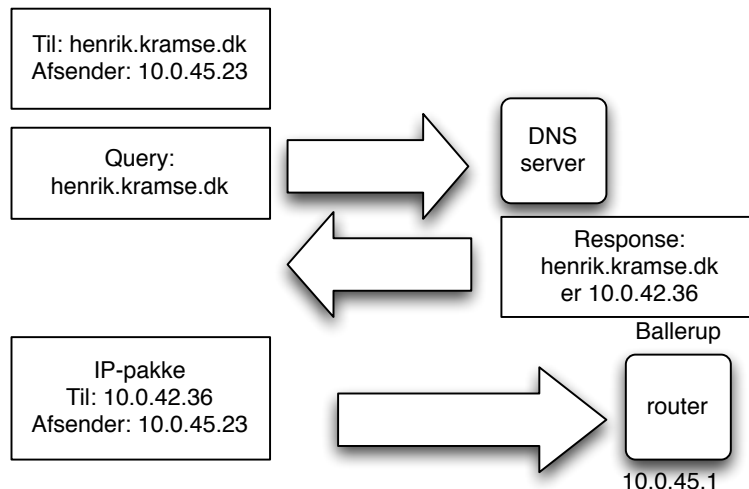
traceroute programmet virker ved hjælp af TTL

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 185.129.60.129
traceroute to 185.129.60.129 (185.129.60.129),
30 hops max, 40 byte packets
 1  safri (10.0.0.11)  3.577 ms  0.565 ms  0.323 ms
 2  router (185.129.60.129)  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.zencurity.com har adressen 185.129.60.130

skrives i database filer, zone filer

ns1	IN	A	185.129.60.130
	IN	AAAA	2a06:d380:0:3065::53
www	IN	A	185.129.60.130
	IN	AAAA	2a06:d380:0:3065::80



# 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



As of 2019-01-29, the root server system consists of 933 instances operated by the 12 independent root server operators.

<http://root-servers.org/>



bestyrer .dk TLD - top level domain

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

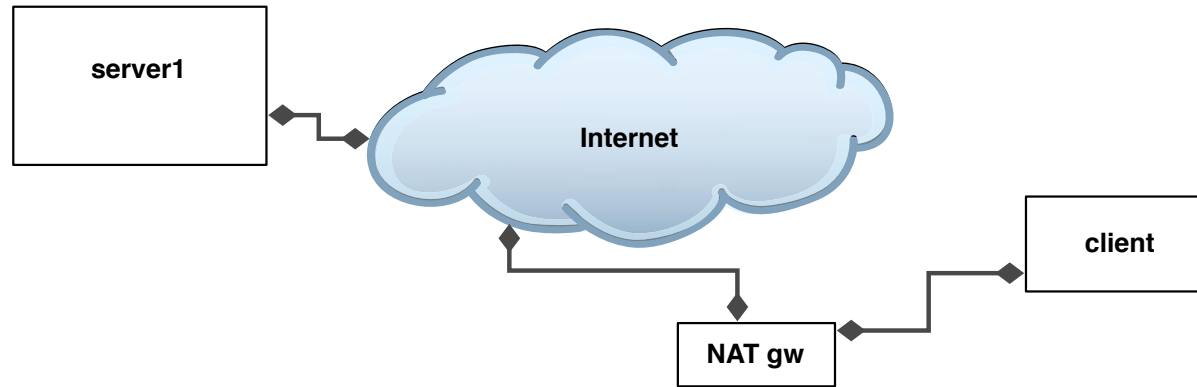
Et domæne bør have flere navneservere og flere postservere

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

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

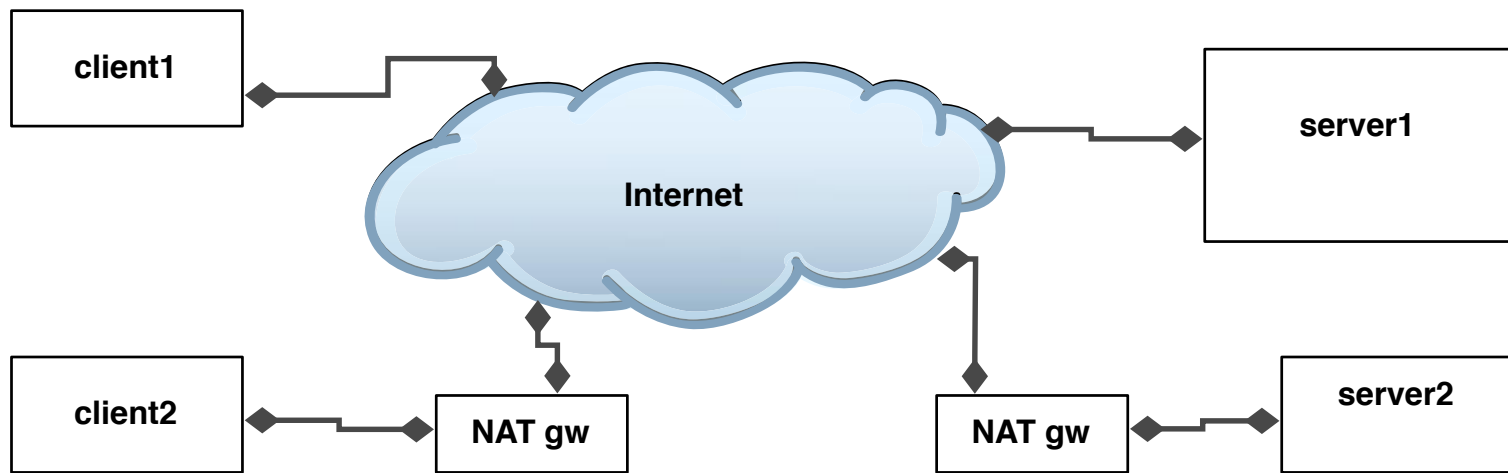
Det anbefales at overveje en service som <http://www.gratisdns.dk> der har 5 navneservere distribueret over stor geografisk afstand - en udenfor Danmark

# NAT Network Address Translation



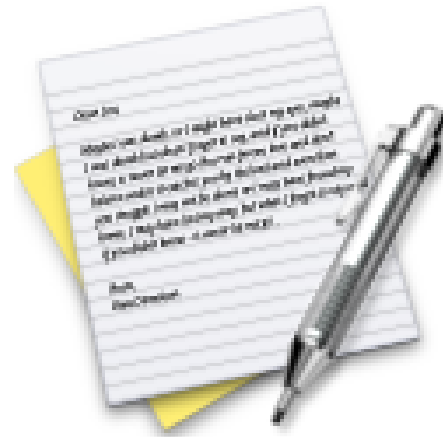
- NAT bruges til at forbinde et privat net (RFC-1918 adresser) med internet
- NAT gateway udskifter afsender adressen med sin egen
- En quick and dirty fix der vil forfølge os for resten af vores liv
- Lægger state i netværket - ødelægger fate sharing

# 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

## Whois databases 15 min

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

# Exercise



Now lets do the exercise

## Using ping and traceroute 10 min

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



# Exercise



Now lets do the exercise

## DNS and Name Lookups 10 min

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

# Short IPv6 introduction



IPv4 running out

32-bit - der ikke kan udnyttes fuldt ud

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)

IPv6 was developed in the 1990s

## 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.zencurity.com

hlk@zencurity.com

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

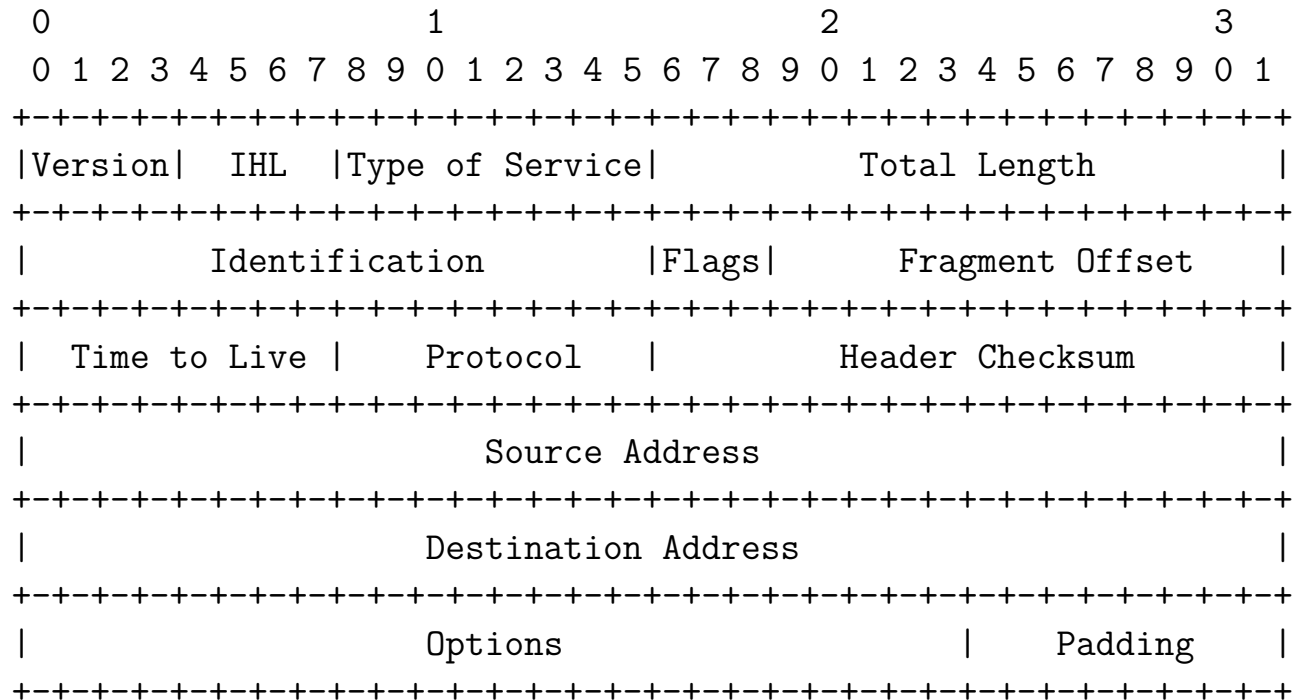
Mindre kompleksitet for routere på vejen medfører mulighed for flere pakker på en given router

# 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) - deprecated
- 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	
+-----+-----+-----+-----			

# 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
```

# 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:fe5:34df
PING6(56=40+8+8 bytes) 2001:1448:81:beef::1 --> 2001:1448:81:beef:20a:95ff:fe5:34df
16 bytes from 2001:1448:81:beef:20a:95ff:fe5:34df, icmp_seq=0 hlim=64 time=10.639 ms
16 bytes from 2001:1448:81:beef:20a:95ff:fe5:34df, icmp_seq=1 hlim=64 time=1.615 ms
16 bytes from 2001:1448:81:beef:20a:95ff:fe5:34df, icmp_seq=2 hlim=64 time=2.074 ms
^C
--- 2001:1448:81:beef:20a:95ff:fe5:34df 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
```



# 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	link#4	UC	en0
ff02::/32	link#5	UC	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		

# 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!

## 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 :-)

# Security problems in the TCP/IP Suite



The title of a nice paper, and the rest of today

The paper “Security Problems in the TCP/IP Protocol Suite” was originally published in Computer Communication Review, Vol. 19, No. 2, in April, 1989

Problems described in the original:

- sequence number spoofing
- routing attacks,
- source address spoofing
- authentication attacks

# TCP sequence number prediction



TCP SEQUENCE NUMBER PREDICTION One of the more fascinating security holes was first described by Morris [7] . Briefly, he used TCP sequence number prediction to construct a TCP packet sequence without ever receiving any responses from the server. This allowed him to spoof a trusted host on a local network.

tidligere baserede man login/adgange på source IP adresser, address based authentication

Er ikke en pålidelig autentifikationsmekanisme

Mest kendt er nok Shimomura der blev hacket på den måde,  
måske af Kevin D Mitnick eller en kompagnon

I praksis vil det være svært at udføre på moderne operativsystemer

Se evt. <http://www.takedown.com/> (filmen er ikke så god ;-))

Det er naturligvis fint med filtre så man kun kan tilgå services FRA bestemte IP

# Routing attacks



Problems described in the original from 1989:

- IP Source routing attacks - angiv en rute for pakkerne  
Knap så brugbar idag
- Routing Information Protocol Attacks  
The Routing Information Protocol [15] (RIP) - denne bruges ikke mere, outdated
- BGPv4 som bruges idag har kæmpe problemer, kludetæppe af kludges

Vi kommer til at snakke om <https://github.com/tomac/yersinia>



# Solutions to TCP/IP security problems



## Solutions:

- Use RANDOM TCP sequence numbers, Win/Mac/Linux - DO, but IoT?
- Filtering, ingress / egress:  
"reject external packets that claim to be from the local net"
- Routers and routing protocols must be more skeptical  
Routing filter everywhere, auth på OSPF/BGP etc.

Has been recommended for some years, but not done in all organisations

BGP routing Resource Public Key Infrastructure RPKI

# DNS problems



The Domain Name System (DNS) [32][33] provides for a distributed database mapping host names to IP addresses. An intruder who interferes with the proper operation of the DNS can mount a variety of attacks, including denial of service and password collection. There are a number of vulnerabilities.

We have a lot of the same problems in DNS today

Plus some more caused by middle-boxes, NAT, DNS size, DNS inspection

- DNS must allow both UDP and TCP port 53
- Your DNS servers must have updated software, see DNS flag day  
<https://dnsflagday.net/> after which kludges will be REMOVED!

# SNMP problems



5.5 Simple Network Management Protocol The Simple Network Management Protocol (SNMP) [37] has recently been defined to aid in network management. Clearly, access to such a resource must be heavily protected. The RFC states this, but also allows for a null authentication service; this is a bad idea. Even a “read-only” mode is dangerous; it may expose the target host to netstat-type attacks if the particular Management Information Base (MIB) [38] used includes sequence numbers. (T

True, and we will talk more about SNMP later in this course.

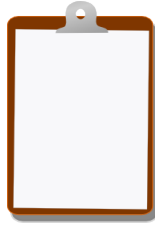


6.1 Vulnerability of the Local Network Some local-area networks, notably the Ethernet networks, are extremely vulnerable to eavesdropping and host-spoofing. If such networks are used, physical access must be strictly controlled. It is also unwise to trust any hosts on such networks if any machine on the network is accessible to untrusted personnel, unless authentication servers are used. If the local network uses the Address Resolution Protocol (ARP) [42] more subtle forms of host-spoofing are possible. In particular, it becomes trivial to intercept, modify, and forward packets, rather than just taking over the host's role or simply spying on all traffic.

Today we can send VXLAN spoofed packets across the internet layer 3 and inject ARP behind firewalls, in some cloud infrastructure cases ...

A Look Back at “Security Problems in the TCP/IP Protocol Suite” about  $1989 + 15 \text{ years} = 2004$

## For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books

Most days have about 100 pages or less, but one day has 4 chapters to read!

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools