Welcome to

Encrypting the Network Layer

Communication and Network Security 2019

Henrik Lund Kramshøj hlk@zencurity.com

Slides are available as PDF, kramse@Github

4-Encrypting-the-Network-Layer.tex in the repo security-courses

kryptering, OpenPGP

kryptering er den eneste måde at sikre:

- fortrolighed
- autenticitet

kryptering består af:

- Algoritmer - eksempelvis RSA
- *protokoller* - måden de bruges på
- programmer - eksempelvis PGP

fejl eller sårbarheder i en af komponenterne kan formindske sikkerheden

PGP = mail sikkerhed, se eksempelvis Enigmail plugin til Mozilla Thunderbird

PGP/GPG verifikation af integriteten

Pretty Good Privacy PGP

Gnu Privacy Guard GPG

Begge understøtter OpenPGP - fra IETF RFC-2440

Når man har hentet og verificeret en nøgle kan man fremover nemt checke integriteten af software pakker

```
hlk@bigfoot:postfix$ gpg --verify  postfix-2.1.5.tar.gz.sig
gpg: Signature made Wed Sep 15 17:36:03 2004 CEST using RSA key ID D5327CB9
gpg: Good signature from "wietse venema <wietse@porcupine.org>"
gpg:                 aka "wietse venema <wietse@wzv.win.tue.nl>"
```

Make and install programs from source

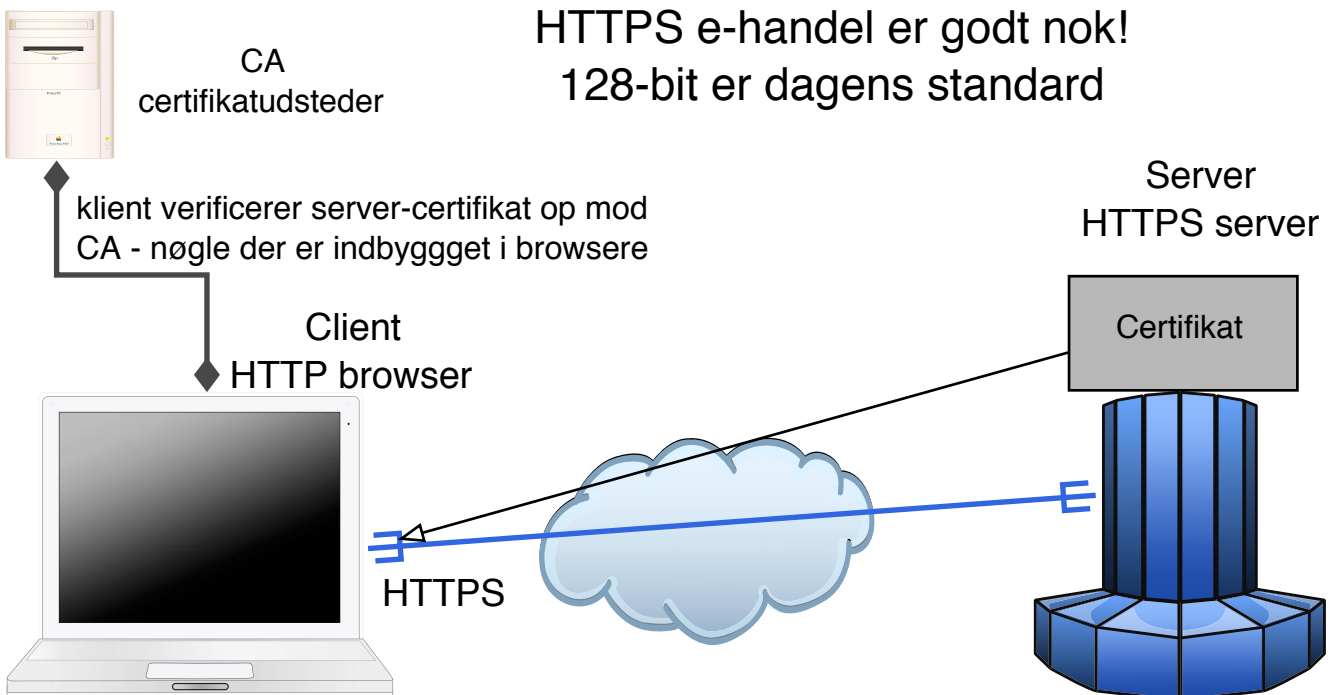
Mange open source programmer kommer som en tar-fil

De fleste C programmer benytter sig så af følgende kommando

- konfigurer softwaren - undersøg hvilket operativsystem det er
- byg software ved hjælp af en Makefile - kompilerer og linker
- installer software - ofte i `/usr/local/bin`

```
./configure;make;make install
```

SSL og TLS

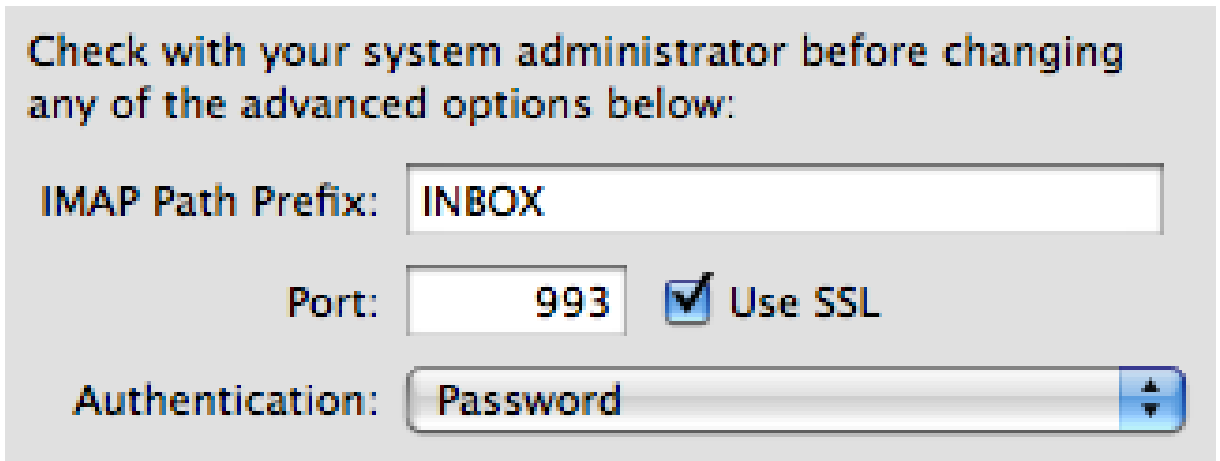


Oprindeligt udviklet af Netscape Communications Inc.

Secure Sockets Layer SSL er idag blevet adopteret af IETF og kaldes derfor også for Transport Layer Security TLS TLS er baseret på SSL Version 3.0

RFC-2246 The TLS Protocol Version 1.0 fra Januar 1999

SSL/TLS udgaver af protokoller



Mange protokoller findes i udgaver hvor der benyttes SSL

HTTPS vs HTTP

IMAPS, POP3S, osv.

Bemærk: nogle protokoller benytter to porte IMAP 143/tcp vs IMAPS 993/tcp

Andre benytter den samme port men en kommando som starter:

SMTP STARTTLS RFC-3207

Secure Shell - SSH og SCP

Hvad er Secure Shell SSH?

Oprindeligt udviklet af Tatu YlÃ¶nen i Finland, se `http://www.ssh.com`

SSH afløser en række protokoller som er usikre:

- Telnet til terminal adgang

- r* programmerne, rsh, rcp, rlogin, ...
- FTP med brugerid/password

SSH - de nye kommandoer er

kommandoerne er:

- ssh - Secure Shell
- scp - Secure Copy
- sftp - secure FTP

Husk: SSH er både navnet på protokollerne - version 1 og 2 samt programmet `ssh` til at logge ind på andre systemer

SSH tillader også port-forward, tunnel til usikre protokoller, eksempelvis X protokollen til UNIX grafiske vinduer

NB: Man bør idag bruge SSH protokol version 2!

SSH nøgler

I praksis benytter man nøgler fremfor kodeord

I kan lave jeres egne SSH nøgler med programmerne i Putty

Hvilken del skal jeg have for at kunne give jer adgang til en server?

Hvordan får jeg smartest denne nøgle?

Installation af SSH nøgle

Vi bruger login med password på kurset, men for fuldstændighedens skyld beskrives her hvordan nøgle installeres:

- først skal der genereres et nøglepar **id_dsa og id_dsa.pub**
- Den offentlige del, filen id_dsa.pub, kopieres til serveren
- Der logges ind på serveren
- Der udføres følgende kommandoer:

```
$ cd  skift til dit hjemmekatalog
$ mkdir .ssh  lav et katalog til ssh-nøgler
$ cat id_dsa.pub >> .ssh/authorized_keys  kopierer nøglen
$ chmod -R go-rwx .ssh  skift rettigheder på nøglen
```

OpenSSH konfiguration

Sådan anbefaler jeg at konfigurere OpenSSH SSHD

Det gøres i filen `sshd_config` typisk `/etc/ssh/sshd_config`

```
Port 22780
Protocol 2

PermitRootLogin no
PubkeyAuthentication yes
AuthorizedKeysFile      .ssh/authorized_keys
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no

#X11Forwarding no
#X11DisplayOffset 10
#X11UseLocalhost yes
```

Det er en smagssag om man vil tillade *X11 forwarding*

IPsec

Sikkerhed i netværket

RFC-2401 Security Architecture for the Internet Protocol

RFC-2402 IP Authentication Header (AH)

RFC-2406 IP Encapsulating Security Payload (ESP)

RFC-2409 The Internet Key Exchange (IKE) - dynamisk keying

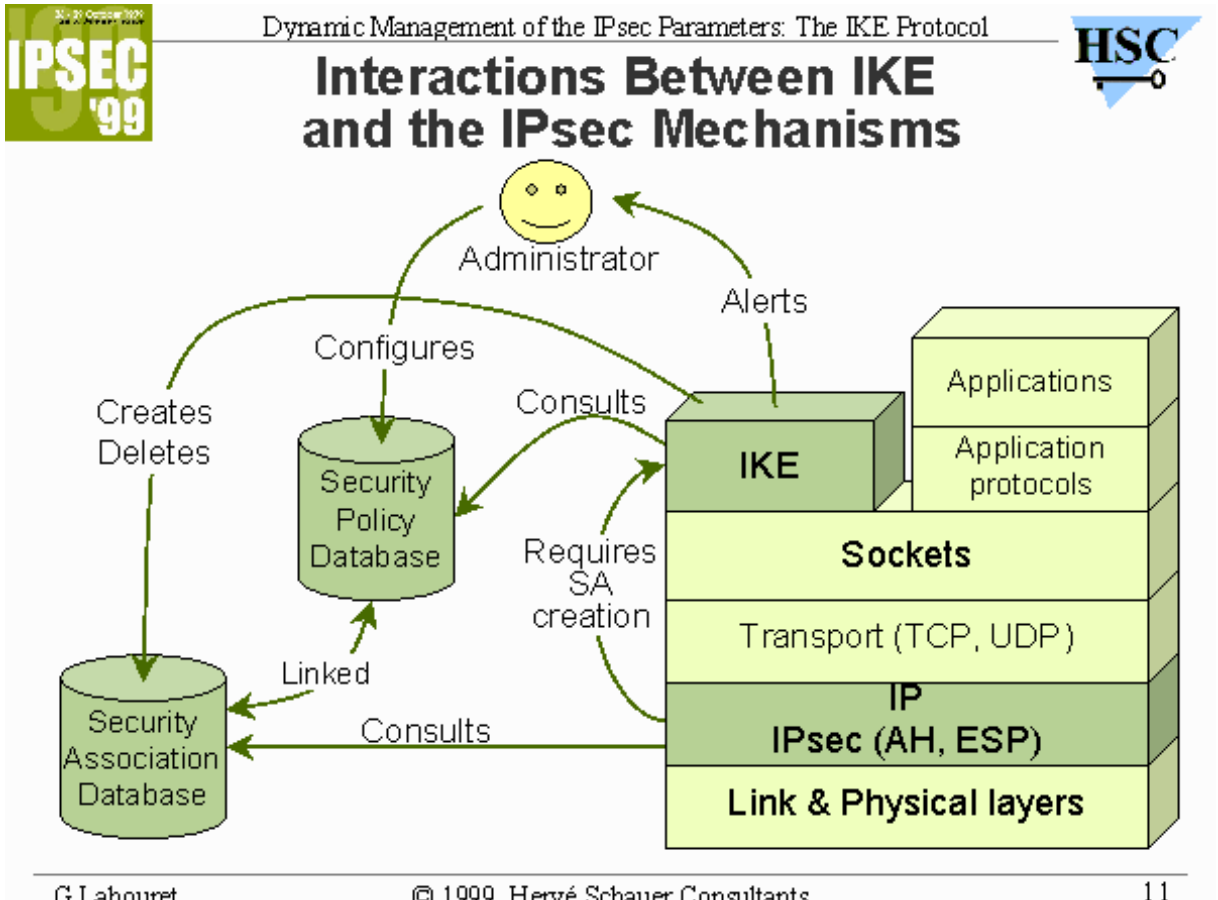
Både til IPv4 og IPv6

MANDATORY i IPv6! - et krav hvis man implementerer fuld IPv6 support

god præsentation på `http://www.hsc.fr/presentations/ike/`

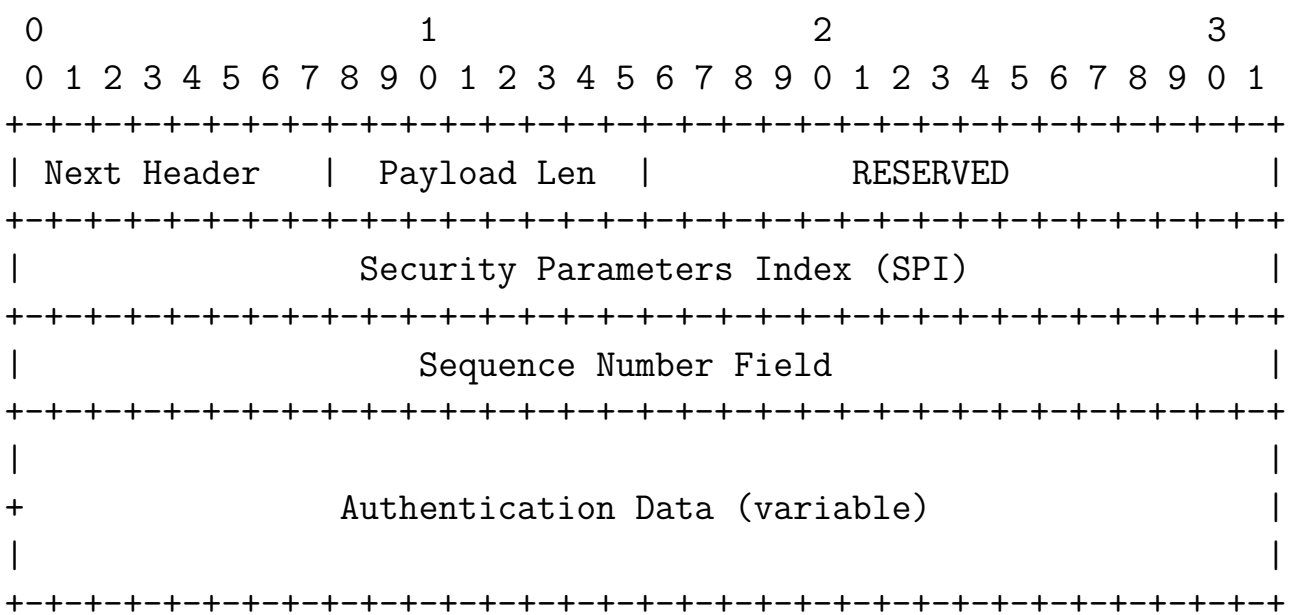
Der findes IKEscan til at scanne efter IKE porte/implementationer
`http://www.nta-monitor.com/ike-scan/index.htm`

IPsec er ikke simpelt!



Kilde: `http://www.hsc.fr/presentations/ike/`

RFC-2402 IP AH



RFC-2402 IP AH

Indpakning - pakkerne før og efter Authentication Header:

```
                BEFORE APPLYING AH
             ----------------------------
      IPv4   |orig IP hdr  |     |      |
             |(any options)| TCP | Data |
             ----------------------------

                 AFTER APPLYING AH
             ---------------------------------
      IPv4   |orig IP hdr  |    |     |      |
             |(any options)| AH | TCP | Data |
             ---------------------------------
             |<------- authenticated ------->|
                  except for mutable fields
```

RFC-2406 IP ESP

Pakkerne før og efter:

```
              BEFORE APPLYING ESP
            ---------------------------------------
      IPv6  |              | ext hdrs |     |      |
            | orig IP hdr  |if present| TCP | Data |
            ---------------------------------------


              AFTER APPLYING ESP
            -------------------------------------------------------
      IPv6  | orig |hop-by-hop,dest*,|   |dest|   |     | ESP   | ESP|
            |IP hdr|routing,fragment.|ESP|opt*|TCP|Data|Trailer|Auth|
            -------------------------------------------------------
                                    |<---- encrypted ---->|
                                  |<---- authenticated ---->|
```

ipsec konfigurationsfiler

Der er følgende filer tilgængelige

- konfigurationsfiler i NetBSD/FreeBSD/Mac OS X format - med `setkey` kommandoen
- konfigurationsfil til OpenBSD server - med `ipsecadm` kommandoen

IPsec setup

Client: Mac OS X/NetBSD/FreeBSD - samme syntaks
`rc.ipsec.client`

Server: OpenBSD - bruger ipsecadm kommando
`rc.ipsec.server`

Øvelse til læseren: lav samme i Cisco IOS

Det vil ofte være relevant at se på IOS og IPsec i laboratoriet

Dette setup når vi ikke at demonstrere

rc.ipsec.client - client setup - adresser

```
#!/bin/sh
# /etc/rc.ipsec.client - IPsec client configuration
# built from http://rt.fm/~jcs/ipsec_wep.phtml
# FreeBSD/NetBSD syntaks! - used on Mac OS X
# IPv4
SECSERVER=10.0.42.1
SECCLIENT=10.0.42.53
# IPv6
#SECSERVER=2001:618:433:101::1
#SECCLIENT=2001:618:433:101::153
ESPKEY=`cat ipsec.esp.key`
AHKEY=`cat ipsec.ah.key`

# Flush IPsec SAs in case we get called more than once
setkey -F
```

```
setkey -F -P
```

rc.ipsec.client - client setup - SAs

```
# Establish Security Associations
# 1000 is from the server to the client
# 1001 is from the client to the server
setkey -c <<EOF

add $SECSERVER $SECCLIENT esp 0x1000 \
-m tunnel -E blowfish-cbc 0x$ESPKEY  -A hmac-sha1 0x$AHKEY;

add $SECCLIENT $SECSERVER esp 0x1001 \
-m tunnel -E blowfish-cbc 0x$ESPKEY -A hmac-sha1 0x$AHKEY;

spdadd $SECCLIENT $SECSERVER any -P out \
ipsec esp/tunnel/$SECCLIENT-$SECSERVER/default;

spdadd $SECSERVER $SECCLIENT any -P in \
```

```
ipsec esp/tunnel/$SECSERVER-$SECCLIENT/default;
EOF
```

rc.ipsec.server - server setup - adresser

```
#!/bin/sh
#
# Henrik Lund Kramshøj
# /etc/rc.ipsec - IPsec server configuration
# built from http://rt.fm/~jcs/ipsec_wep.phtml
# OpenBSD syntaks!
SECSERVER=10.0.42.1
SECCLIENT=10.0.42.53
#SECSERVER6=2001:618:433:101::1
#SECCLIENT6=2001:618:433:101::153

ESPKEY=`cat ipsec.esp.key`
AHKEY=`cat ipsec.ah.key`

# Flush IPsec SAs in case we get called more than once
```

```
ipsecadm flush
```

rc.ipsec.server - server setup - SAs

```
# Establish Security Associations
#
# 1000 is from the server to the client
ipsecadm new esp -spi 1000 -src $SECSERVER -dst $SECCLIENT \
-forcetunnel -enc blf -key $ESPKEY \
-auth sha1 -authkey $AHKEY

# 1001 is from the client to the server
ipsecadm new esp -spi 1001 -src $SECCLIENT -dst $SECSERVER \
-forcetunnel -enc blf -key $ESPKEY \
-auth sha1 -authkey $AHKEY
```

rc.ipsec.server - server setup - flows

```
# Create flows
#
# Data going from the outside to the client
ipsecadm flow -out -src $SECSERVER -dst $SECCLIENT -proto esp \
-addr 0.0.0.0 0.0.0.0 $SECCLIENT 255.255.255.255 -dontacq
# IPv6
#ipsecadm flow -out -src $SECSERVER -dst $SECCLIENT -proto esp \
#-addr :: :: $SECCLIENT ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff -dontacq

# Data going from the client to the outside
ipsecadm flow -in -src $SECSERVER -dst $SECCLIENT -proto esp \
-addr $SECCLIENT 255.255.255.255 0.0.0.0 0.0.0.0 -dontacq
# IPv6
#ipsecadm flow -in -src $SECSERVER -dst $SECCLIENT -proto esp \
#-addr :: :: $SECCLIENT ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff -dontacq
```

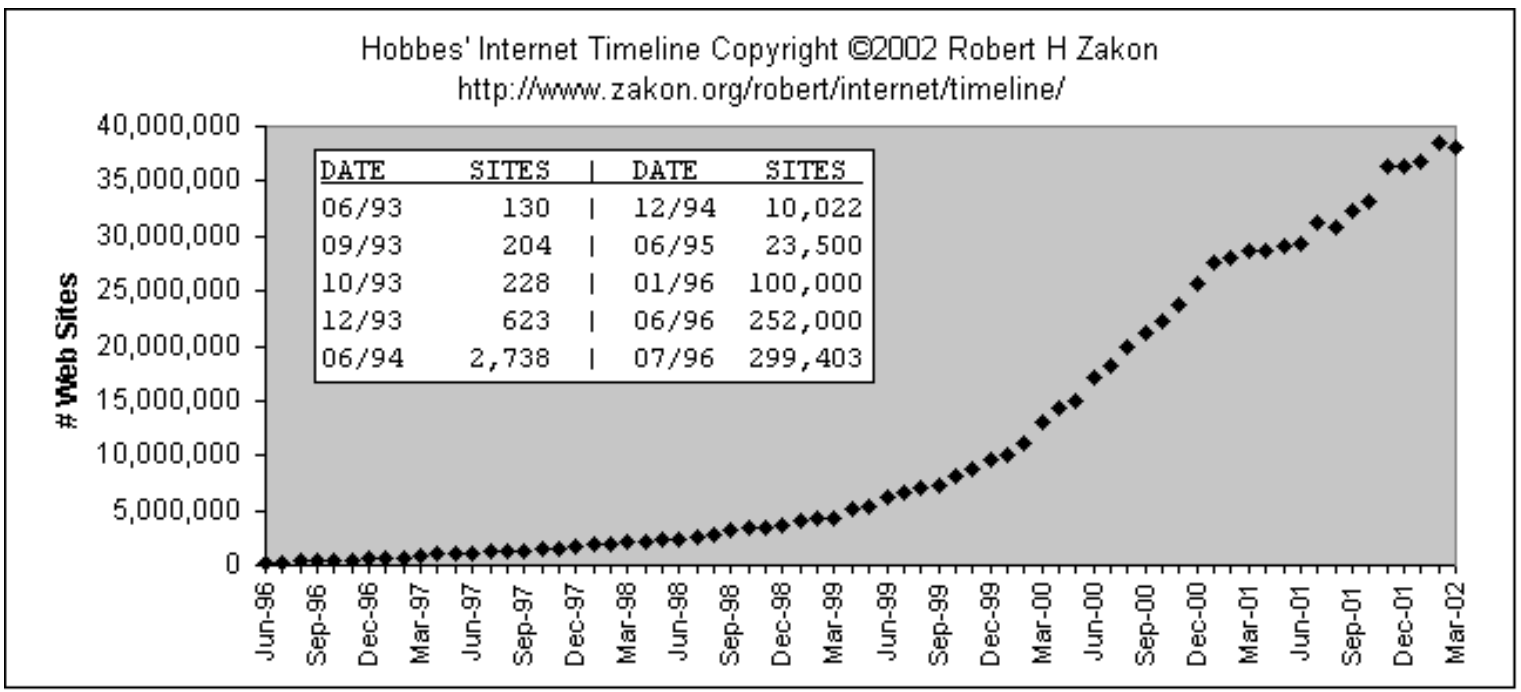
World Wide Web fødes

Tim Berners-Lee opfinder WWW 1989 og den første webbrowser og server i 1990 mens han

arbejder for CERN

Kilde: `http://www.w3.org/People/Berners-Lee/`

World Wide Web udviklingen



DATE	SITES	DATE	SITES
06/93	130	12/94	10,022
09/93	204	06/95	23,500
10/93	228	01/96	100,000
12/93	623	06/96	252,000
06/94	2,738	07/96	299,403

Udviklingen på world wide web bliver en stor kommerciel success

Kilde: Hobbes Internet time-line
`http://www.zakon.org/robert/internet/timeline/`

Nogle HTTP og webrelaterede RFC'er

1945 Hypertext Transfer Protocol – HTTP/1.0. T. Berners-Lee, R. Fielding, H. Frystyk. May 1996.

2068 Hypertext Transfer Protocol – HTTP/1.1. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee. January 1997. (Obsoleted by RFC2616)

2069 An Extension to HTTP : Digest Access Authentication. J. Franks, P. Hallam-Baker, J. Hostetler, P. Leach, A. Luotonen, E. Sink, L. Stewart. January 1997. (Obsoleted by RFC2617)

2145 Use and Interpretation of HTTP Version Numbers. J. C. Mogul, R. Fielding, J. Gettys, H. Frystyk. May 1997.

2518 HTTP Extensions for Distributed Authoring – WEBDAV. Y. Goland, E. Whitehead, A. Faizi, S. Carter, D. Jensen. February 1999.

2616 Hypertext Transfer Protocol – HTTP/1.1. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee. June 1999. (Obsoletes RFC2068) (Updated by RFC2817)

2818 HTTP Over TLS. E. Rescorla. May 2000.

HTTP er basalt set en sessionsløs protokol bestående at individuelle HTTP forespørgsler via TCP forbindelser

Infokager og state management

2109 HTTP State Management Mechanism. D. Kristol, L. Montulli. February 1997. (Format: TXT=43469 bytes) (Obsoleted by RFC2965) (Status: PROPOSED STANDARD)

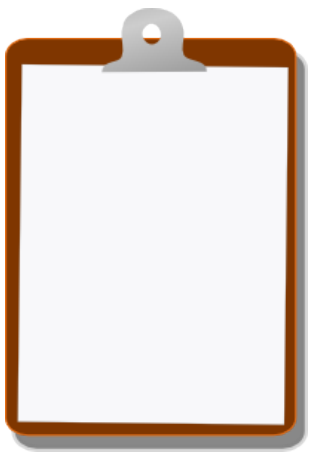
2965 HTTP State Management Mechanism. D. Kristol, L. Montulli. October 2000. (Format: TXT=56176 bytes) (Obsoletes RFC2109) (Status: PROPOSED STANDARD)

> 1. ABSTRACT This document specifies a way to create a stateful session with HTTP requests and responses. It describes two new headers, Cookie and Set-Cookie, which carry state information between participating origin servers and user agents. The method described here differs from Netscape's Cookie proposal, but it can interoperate with HTTP/1.0 user agents that use Netscape's method. (See the HISTORICAL section.)

(Citatet er fra RFC-2109)

Transport Layer Security

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