Forelesning 10

Web-sikkerhet og SSH

Nye problemstillinger med WWW

- I motsetning til Text-TV etc, er internett en toveis-affære
- Stadig flere bedrifter bruker WWW som ansikt utad
- Selv om klient og tjener-programmer er enkle å installere og bruke, blir de stadig mer komplekse innvendig

Nye problemstillinger forts.

- En dårlig beskyttet Web-server kan være innfallsporten en inntrenger bruker for å få tilgang til et bedriftsnettverk
- Brukerterskelen for WWW er lav – en stadig mindre del av brukerne er i stand til å ta stilling til risikoen involvert i de forskjellige operasjoner

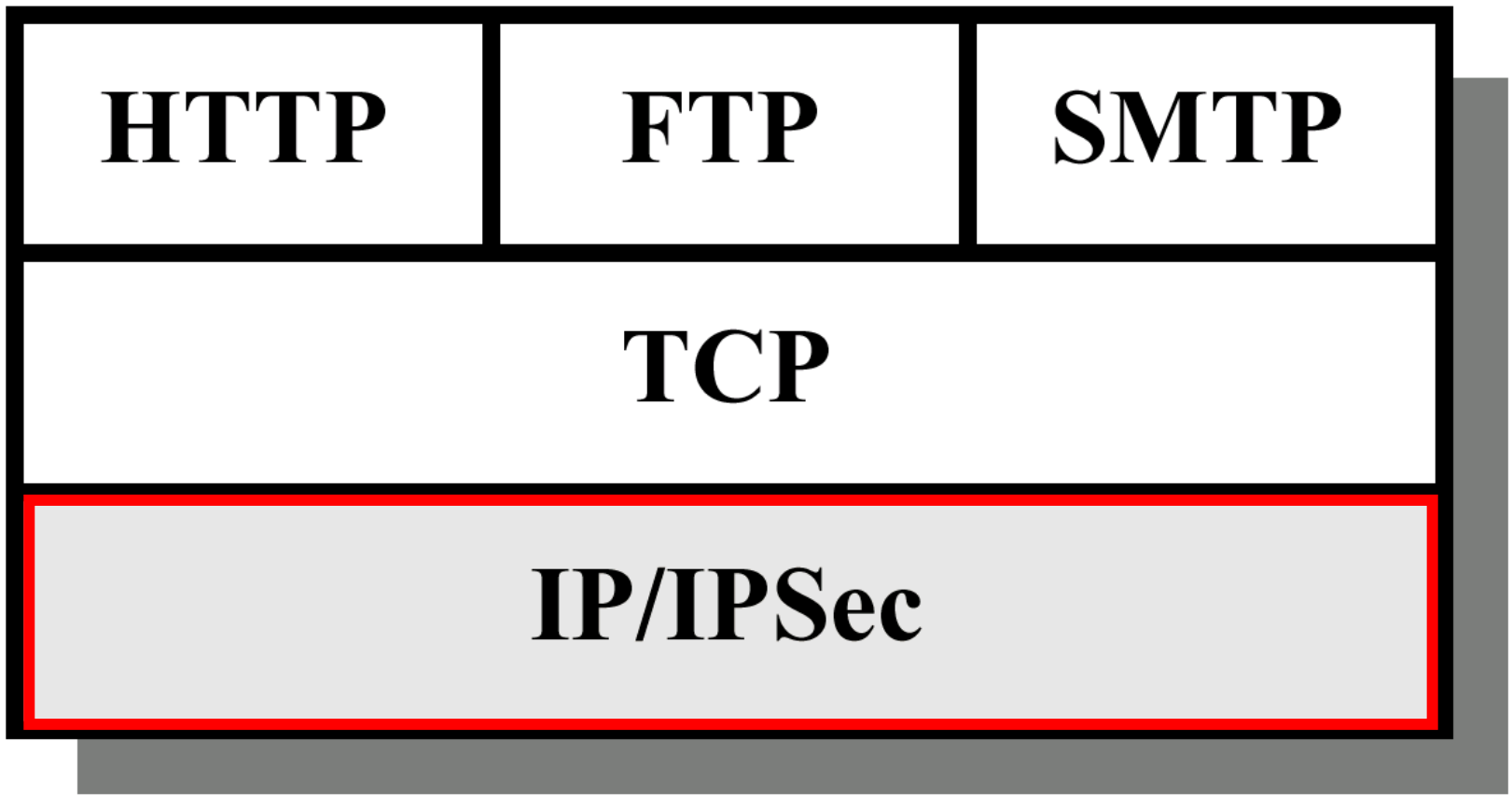
Trusler - repetisjon

- Integritet
  - Endring av brukerdata, endring av melding
- Konfidensialitet
  - Avlytting, datatyveri
- Tilgjengelighet
  - DoS
- Autentisering
  - Forfalskning av data, “impersonation”

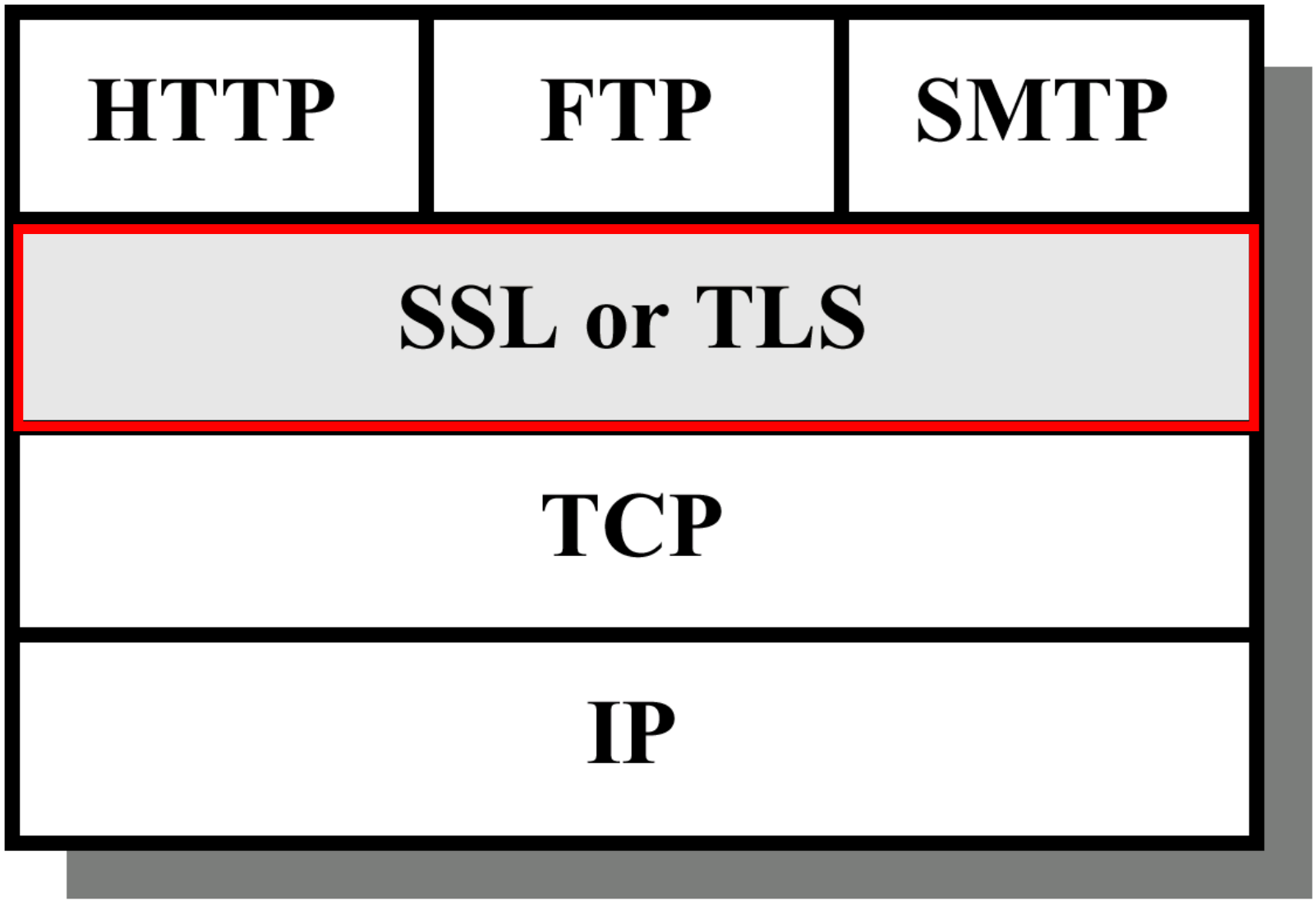
Tilnærminger til Web-sikkerhet

- Nettverkslaget
  - IPSec
  - Transparent for applikasjoner
- Transportlaget
  - SSL/TLS
  - Kan bakes inn i applikasjoner, evt. som "pseudo-lag" i protokollstakk
- Applikasjonslaget
  - Kerberos, S/MIME, PGP, SET

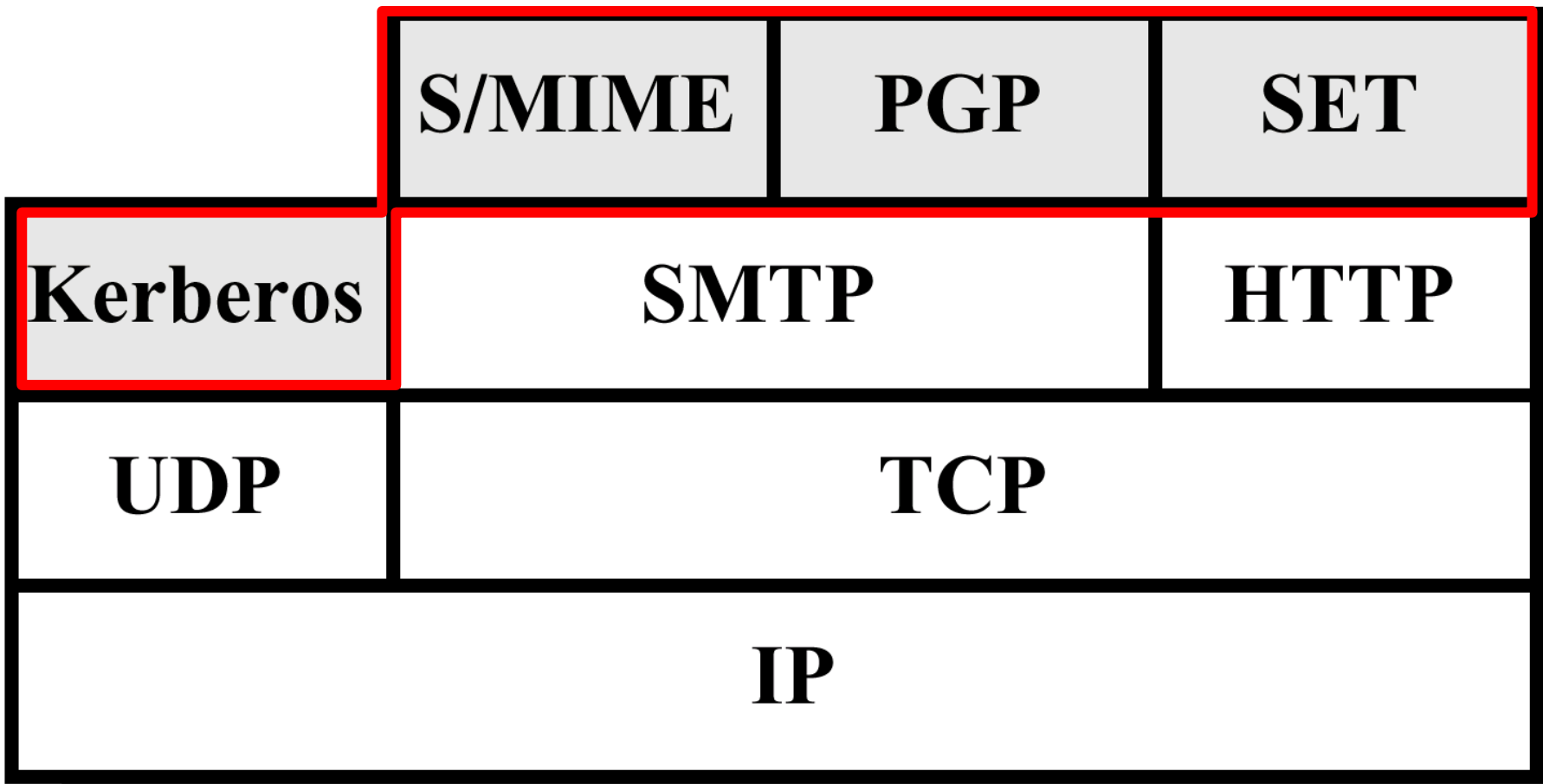
Nettverkslaget



Transportlaget



Applikasjonslaget



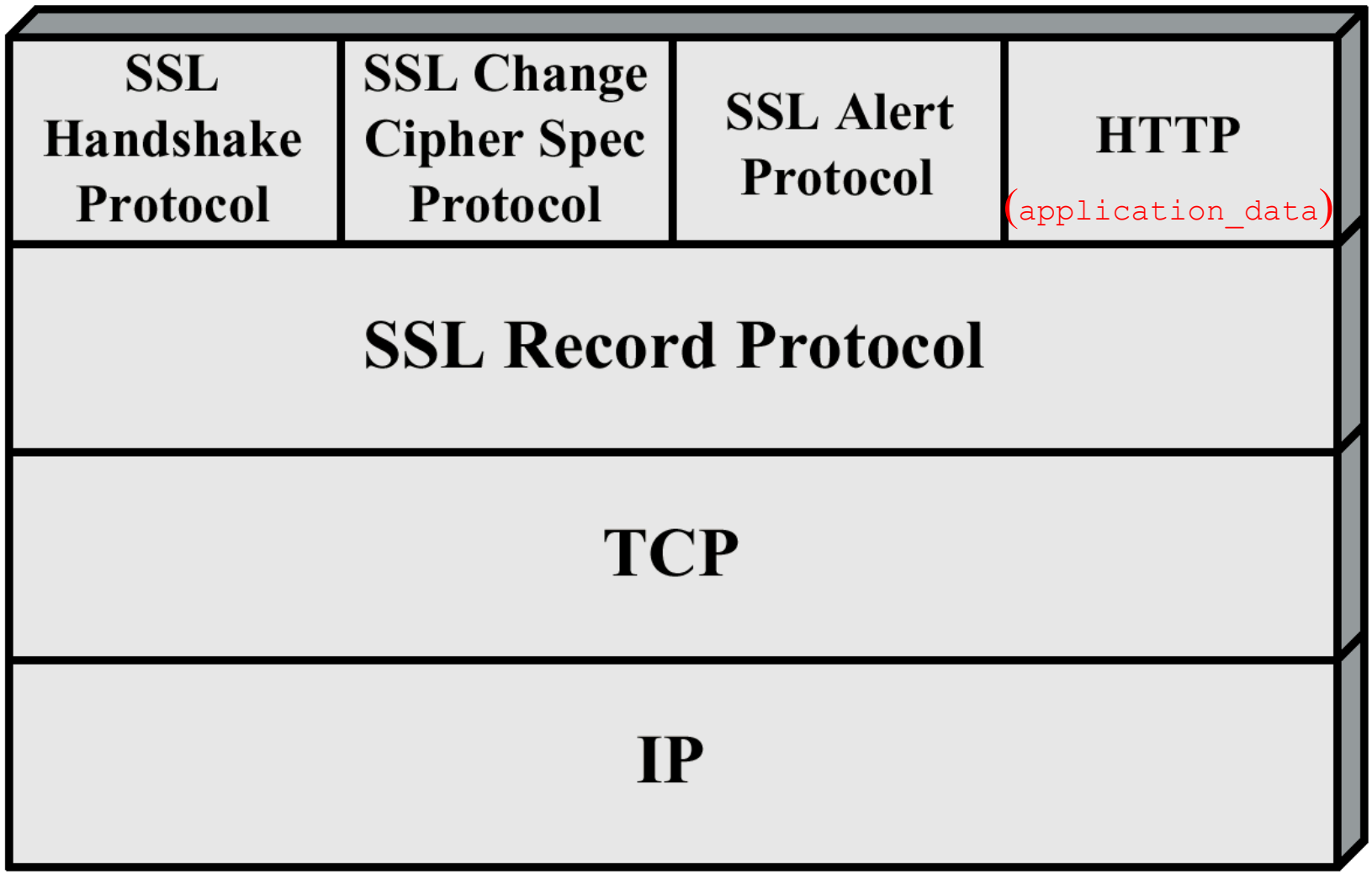
Secure Socket Layer

- Utviklet av Netscape
- Versjon 3.0 utviklet med bidrag fra åpne fagmiljøer, og publisert som Internet Draft
- Dannet TLS WG
- TLS 1.0 = “SSL 3.1” ≈ SSL 3.0

SSL arkitektur

- SSL bruker TCP for å lage en sikker og pålitelig ende-til-ende-tjeneste
- SSL er egentlig to protokollnivåer:
  - Nivå 1:
    SSL record protocol
  - Nivå 2:
    - Handshake
    - Change cipher spec
    - Alert

SSL protokollstakk



Sesjon og forbindelse

- SSL forbindelse
  - Forbindelse mellom likeverdige enheter
  - Transient
  - Hver forbindelse er assosiert med en sesjon (en sesjon kan ha flere forbindelser)
- SSL sesjon
  - Assosiasjon mellom klient og tjener
  - Opprettes av Handshake-protokollen
  - Definerer et sett sikkerhetsparametre

Sesjon forts.

- En sesjon består altså av en eller flere forbindelser
- Det er sesjonen som forhandler fram hvilke nøkler og algoritmer som skal brukes
- En forbindelse bruker de nøkler og algoritmer som til enhver tid gjelder for den tilhørende sesjonen!

Sesjonstilstand

- Sesjons-ID
- Motparts sertifikat (X.509)
- Kompresjonsmetode
- “Cipher spec”
  - Krypteringsalgoritme, hash-algoritme, etc
- Master secret – 48 byte nøkkel
- Is resumable – kan denne sesjonen ta initiativ til nye forbindelser?

Forbindelsestilstand

- Server/Client random – tilfeldig valgt for hver forbindelse (nonce)
- Server/Client write MAC secret – hemmelig nøkkel for MAC fra server/klient
- Server/Client write key – symmetrisk nøkkel for kryptering av data sendt fra server/klient
- IV – for CBC
- Sekvensnummer – hver side styrer sine

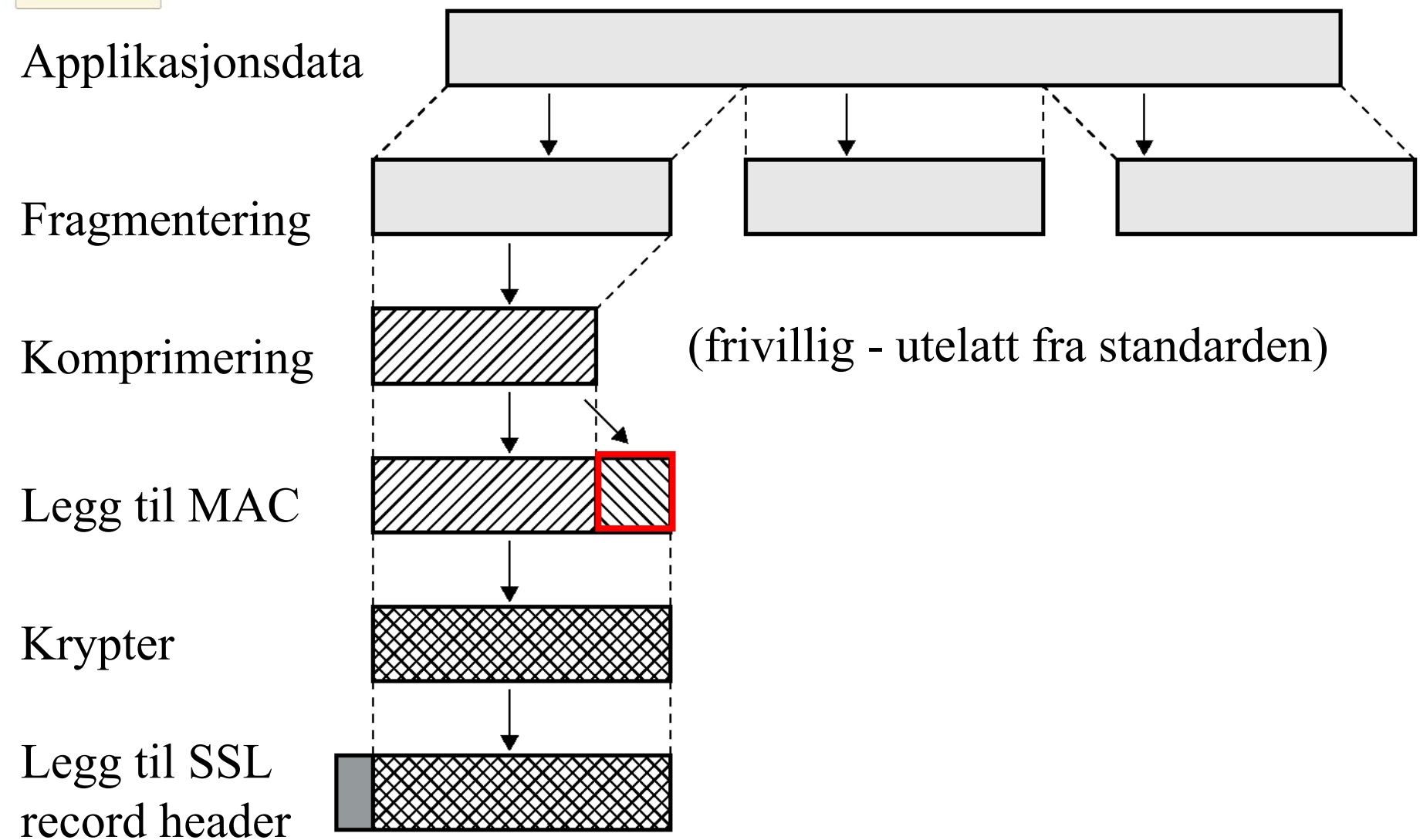
Kommende tilstand

- Ved å bruke handshake-protokollen, forhandler partene fram en ny sesjonstilstand
- Denne merkes som ”pending” inntil en `change_cipher_spec`-melding blir sendt

SSL Record Protocol

- Tilbyr **konfidensialitet** og **meldings-integritet** (MAC) til SSL-forbindelser
- Fragmenterer
- Komprimerer (eller lar være)
- Legger på MAC (variant av HMAC)
- Krypterer
- Legger på SSL Record header

SSL record protocol operasjoner



(frivillig - utelatt fra standarden)

Mulige krypteringsalgoritmer

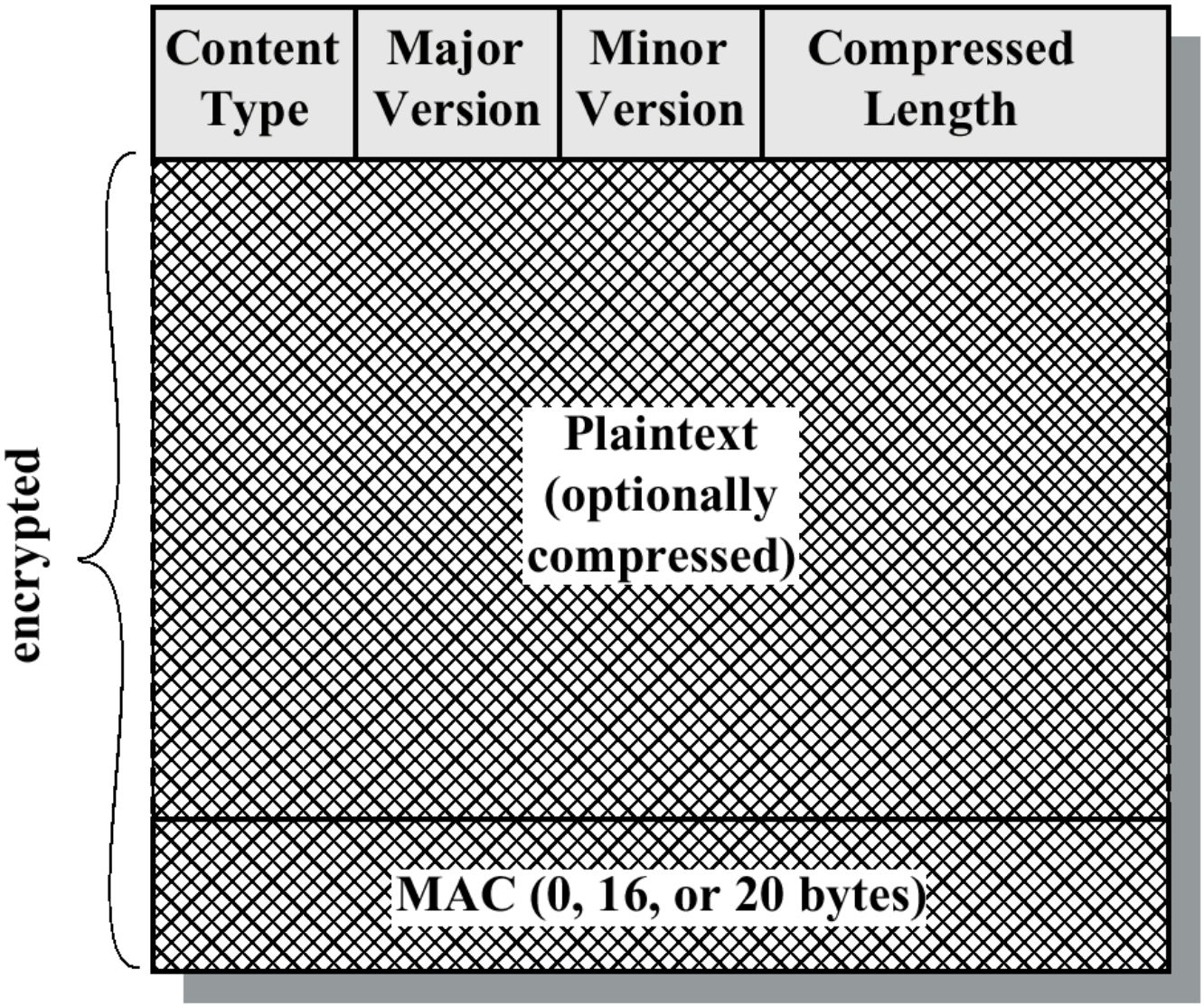
- Blokkchiffer
  - IDEA (128 bit)
  - RC2-40 (40 bit)
  - DES-40 (40 bit)
  - DES (56 bit)
  - 3DES (168 bit)
  - (Fortezza (80 bit))
- Flytchiffer
  - RC4-40 (40 bit)
  - RC4-128 (128 bit)

➢ Flere kommer nok!

Record protocol format

- Header
  - Content type
  - Major version (av SSL, e.g. 3)
  - Minor version (av SSL, e.g. 0)
  - Compressed length (lengde av payload)
- Payload
- MAC

Record protocol format



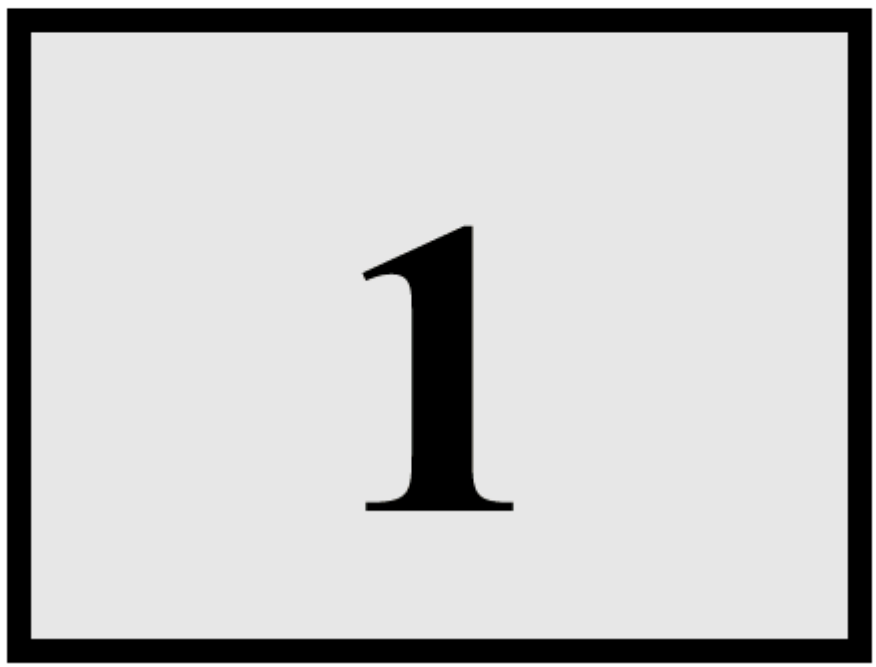
Record Protocol nyttelast

(Dvs.: De forskjellige Content Types)

- Change Cipher Spec Protocol
  ”nå går vi for det vi ble enige om”
- Alert Protocol
  ”her er det noe galt!”
- Handshake Protocol
  ”la oss bli enige om noen få ting”
- Application Data
  ”det vi egentlig var interessert i å gjøre”

Change Cipher Spec

1 byte



Alert

1 byte 1 byte

Level	Alert

Alert forts.

- To nivåer
  - warning (1)
  - fatal (2)
- Hvis en fatal alert mottas, termineres forbindelsen umiddelbart
- Andre forbindelser i sesjonen kan fortsette, men nye forbindelser tillates ikke (må i så fall opprette ny sesjon)

Fatal alerts

- unexpected_message
- bad_record_mac
  - mottatt MAC stemmer ikke med generert
- decompression_failure
- handshake_failure
  - partene klarte ikke å bli enige
- illegal_parameter

Warnings

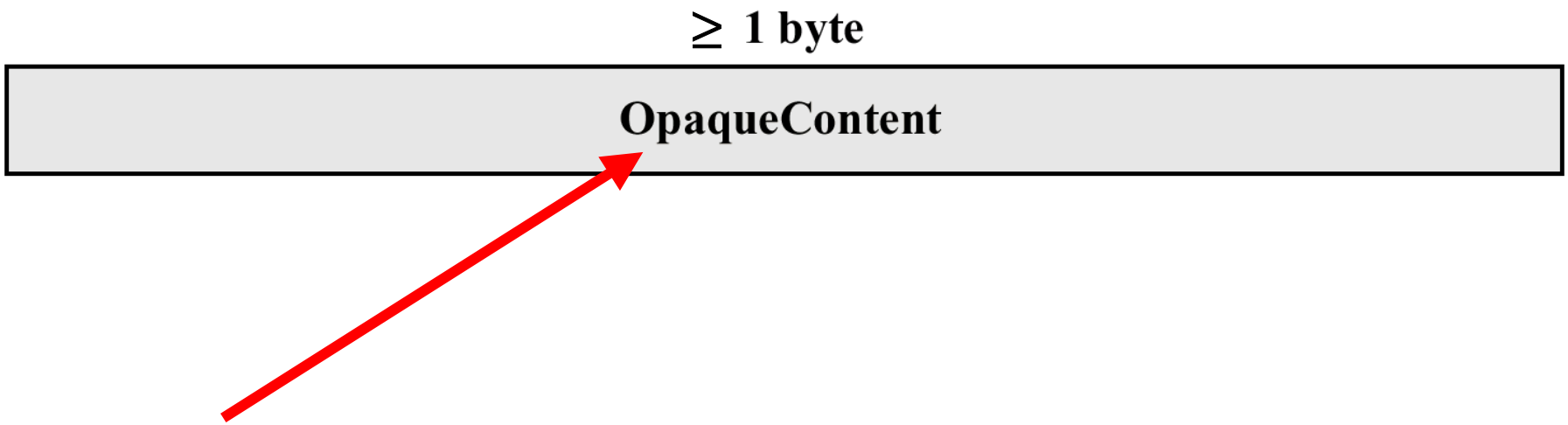
- close_notify
- no_certificate
- bad_certificate
- unsupported_certificate
- certificate_revoked
- certificate_expired
- certificate_unknown
  - ”noe annet er galt med sertifikatet”

Handshake

1 byte	3 bytes	≥ 0 bytes
Type	Length	Content

Annen (applikasjons-)protokoll

f.eks. HTTP

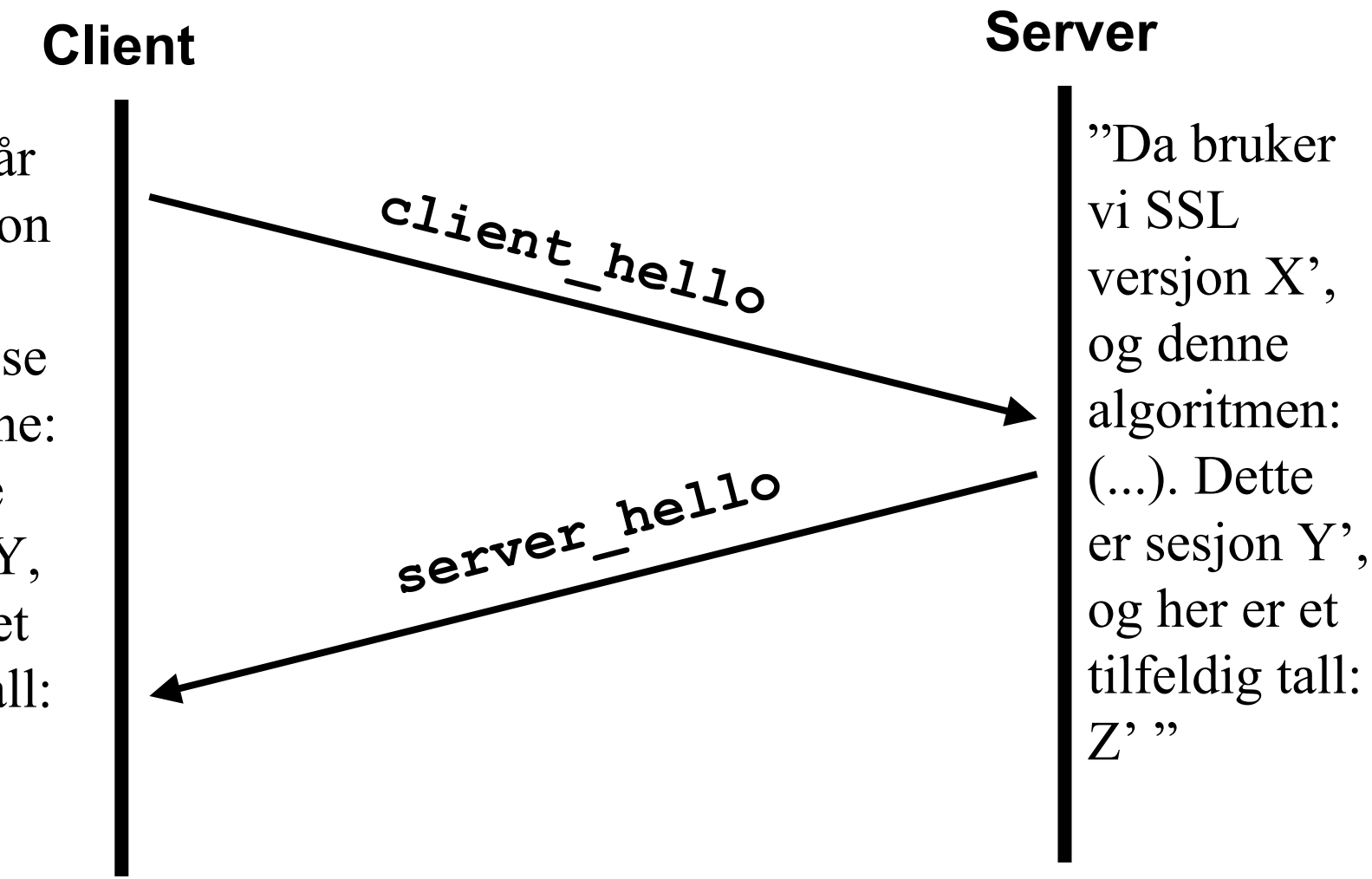


Dette verken ønsker vi eller har vi behov for å vite noe om!

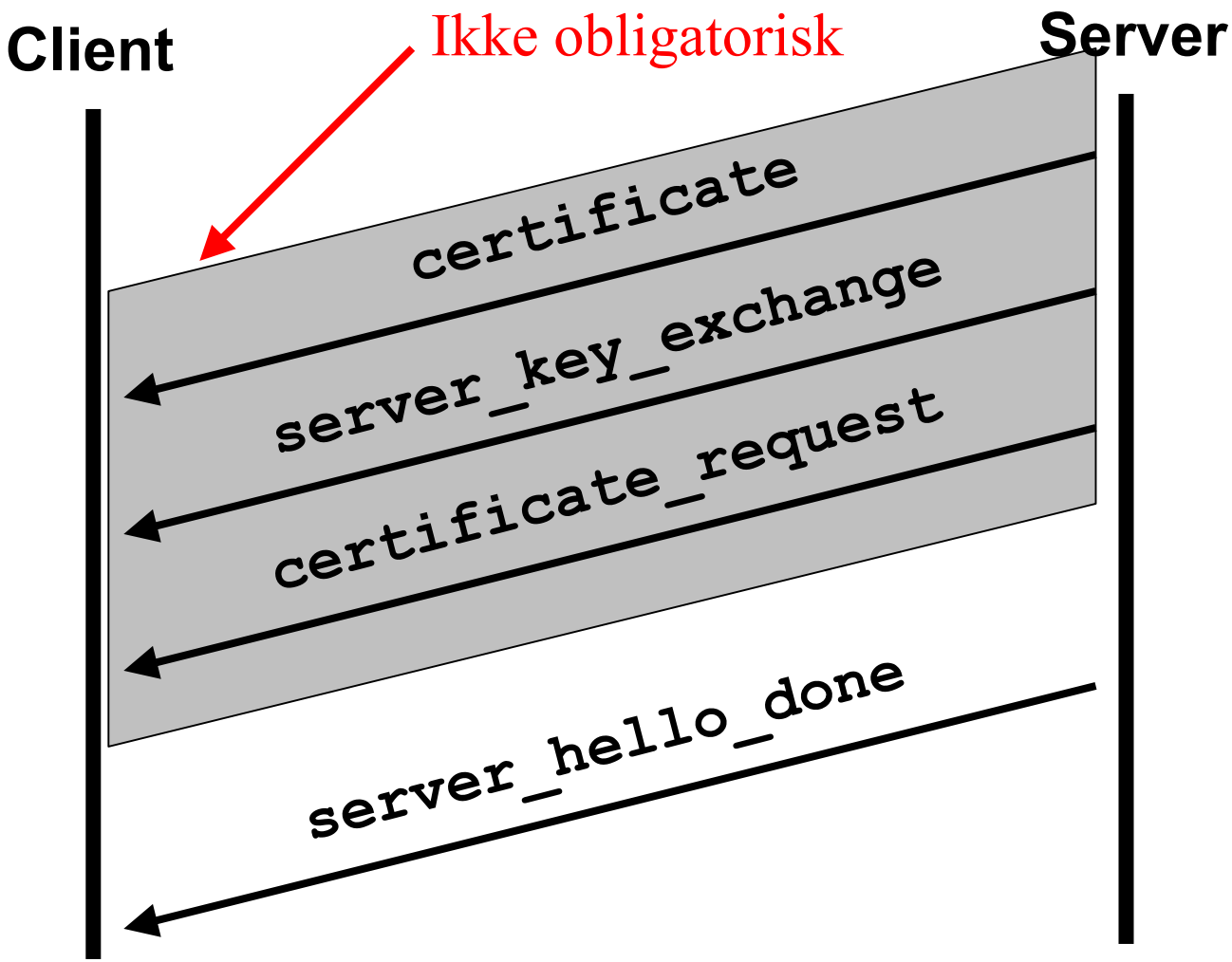
Faser i handshake-protokollen

1. Hvilke sikkerhetsmekanismer støtter partene?
2. Autentisering av server og nøkkelutveksling
3. Autentisering av klient og nøkkelutveksling
4. Avslutning

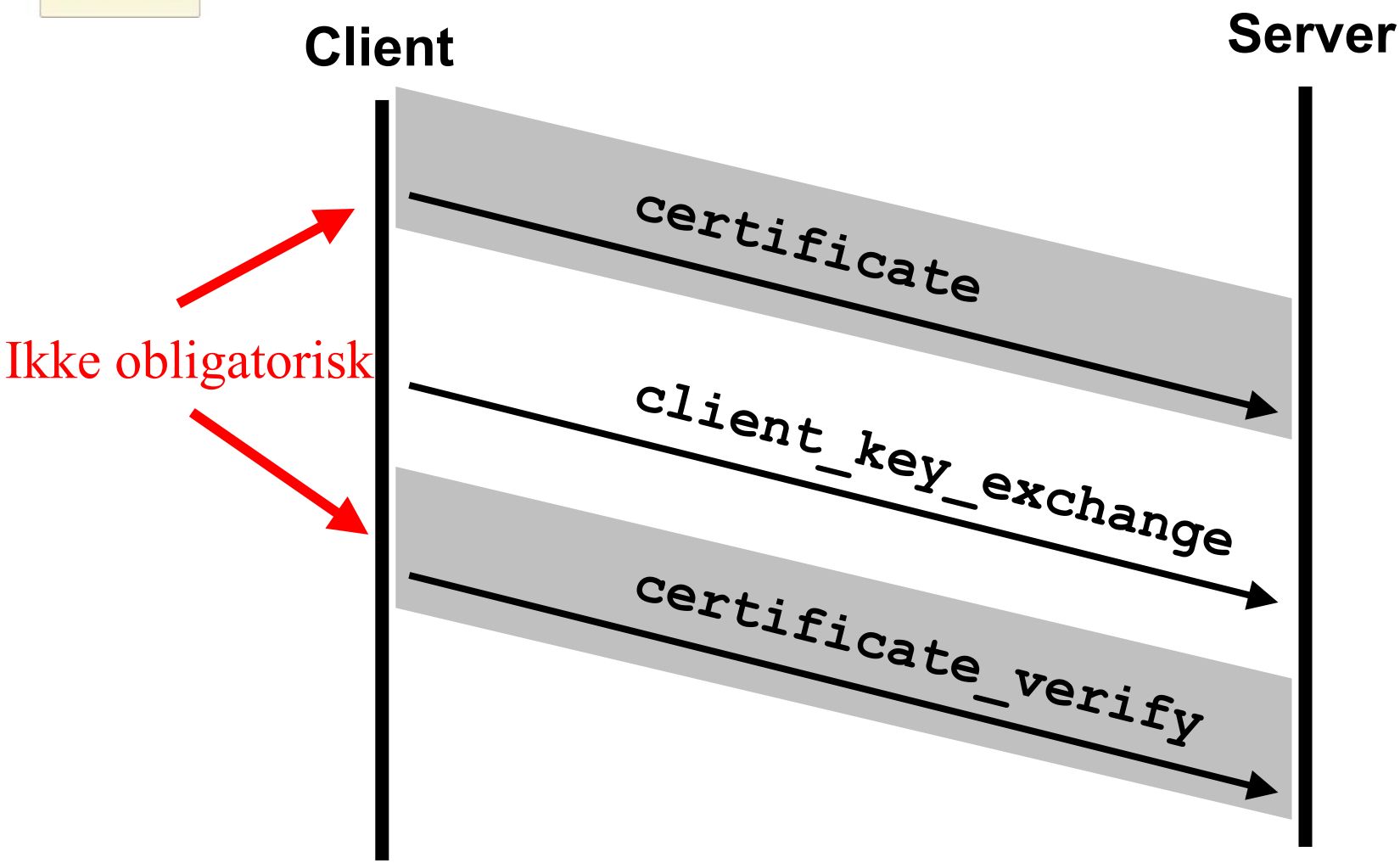
Handshake protocol - fase 1



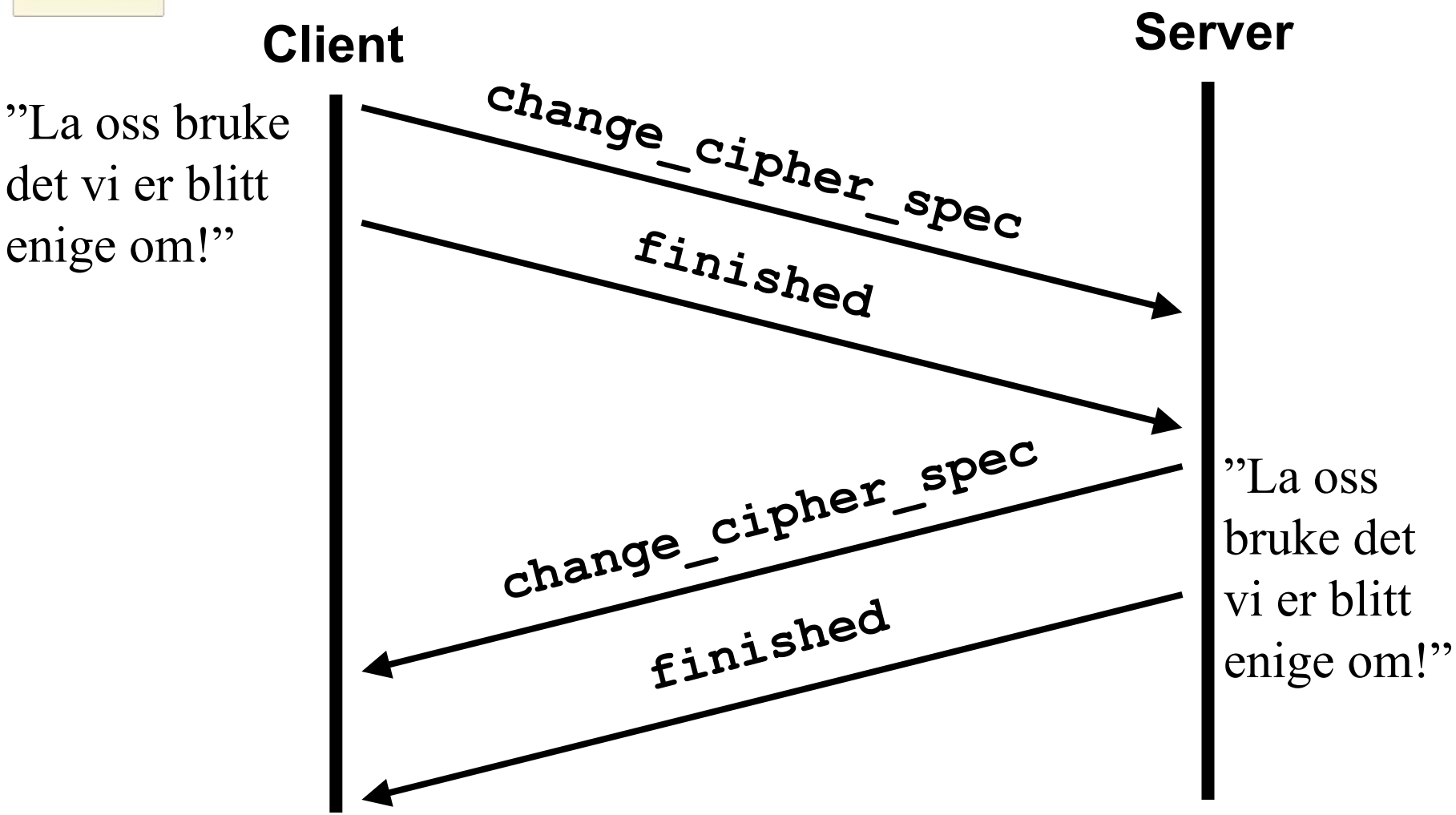
Handshake protocol - fase 2



Handshake protocol - fase 3



Handshake protocol - fase 4



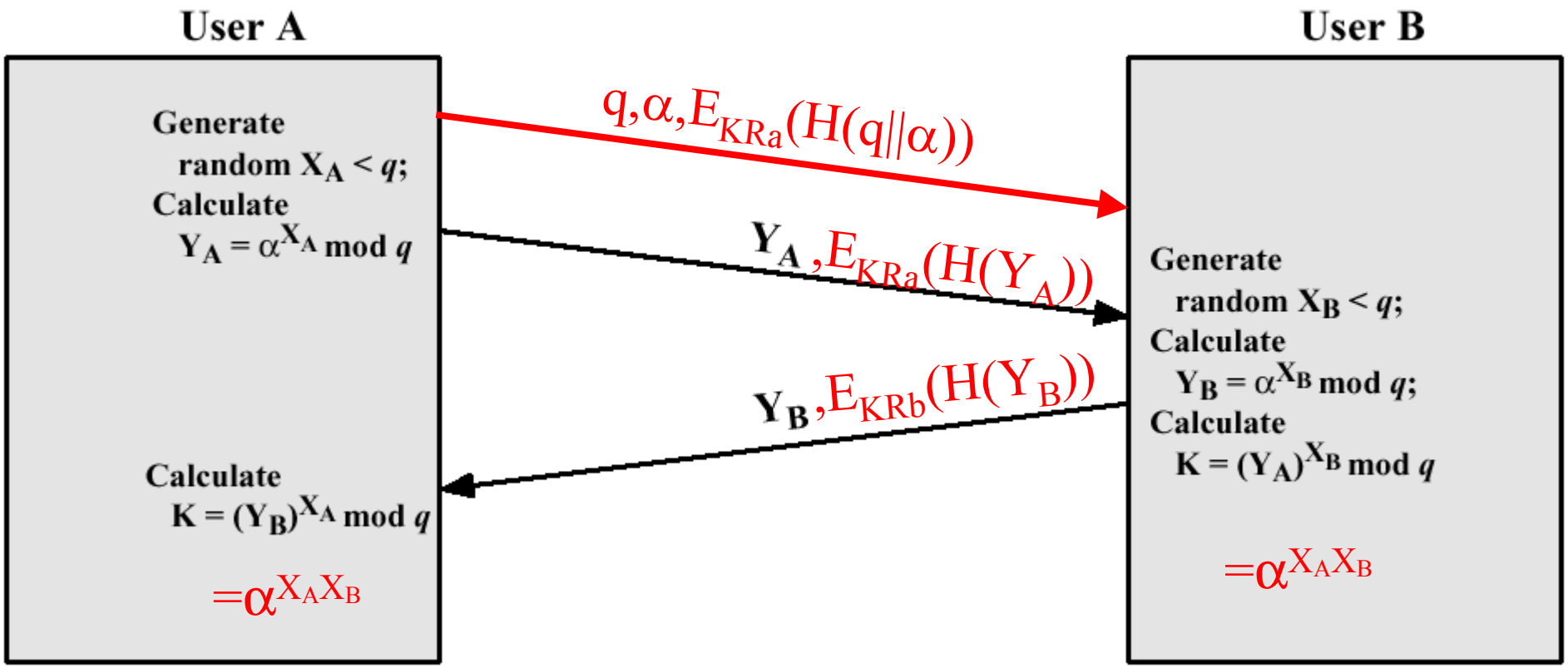
Cipher Suite

- En kombinasjon av kryptografiske algoritmer som støttes av en klient
- Handshake-meldingen client_hello inneholder en liste slike ”cipher suites” i foretrukket rekkefølge
- Hver cipher suite består av
  - Nøkkelfordelingsalgoritme
  - CipherSpec

Nøkkelfordelingsmetoder

- RSA
- Fixed Diffie-Hellman
  - Offentlige parametre i sertifikat
- Ephemeral Diffie-Hellman
  - Offentlige parametre utveksles, signert
- Anonymous Diffie-Hellman
- (Fortezza)

Diffie-Hellman repitisjon



Server Key Exchange melding

- Behøves ikke hvis man bruker
  - Fixed Diffie-Hellman
  - RSA (sign/encrypt public key)

} **men husk sertifikatet!**

- Må brukes for
  - Anonymous Diffie-Hellman
  - Ephemeral Diffie-Hellman
  - RSA (sign only)
  - Fortezza

SSL signering

- Når data signeres, inkluderes både server og klients nonce-verdier i hashen:
  hash(ClientHello.random||
  ServerHello.random||{parametre})
- Hashen signeres deretter med senderens private nøkkel
- Dette beskytter mot replay!

Certificate Request

- Hvis serveren ikke bruker anonym D-H, kan den be om et sertifikat fra klienten
- Sertifikat-typer:
  - RSA signature only/fix. D-H/eph. D-H
  - DSS signature only/fix. D-H/eph. D-H
  - (Fortezza)

Client Key Exchange melding

- RSA
  - Klienten generer en tilfeldig 48-byte ”pre-master secret” og krypterer med servers offentlige nøkkel
- Ephemeral/Anonymous D-H
  - De offentlige parametrene sendes
- Fixed D-H
  - Tomt innhold!

SSL finish

- Har blitt enige om 48 bytes master_secret
- change_cipher_spec aktiverer de forhandlede parametrene
- finish verifiserer at autentisering og key exchange gikk ok
- Parameteren til finish er to hash-verdier som avhenger av alle meldingene sendt hittil, pluss master_secret

Master Secret Creation

- RSA
  - Pre_master_secret generert av klient og mottatt av server
- D-H
  - Pre_master_secret er nøkkelen man kommer fram til ($\alpha^{X_A X_B}$)
- For enkelhets skyld: pre_master_secret = p_m_s

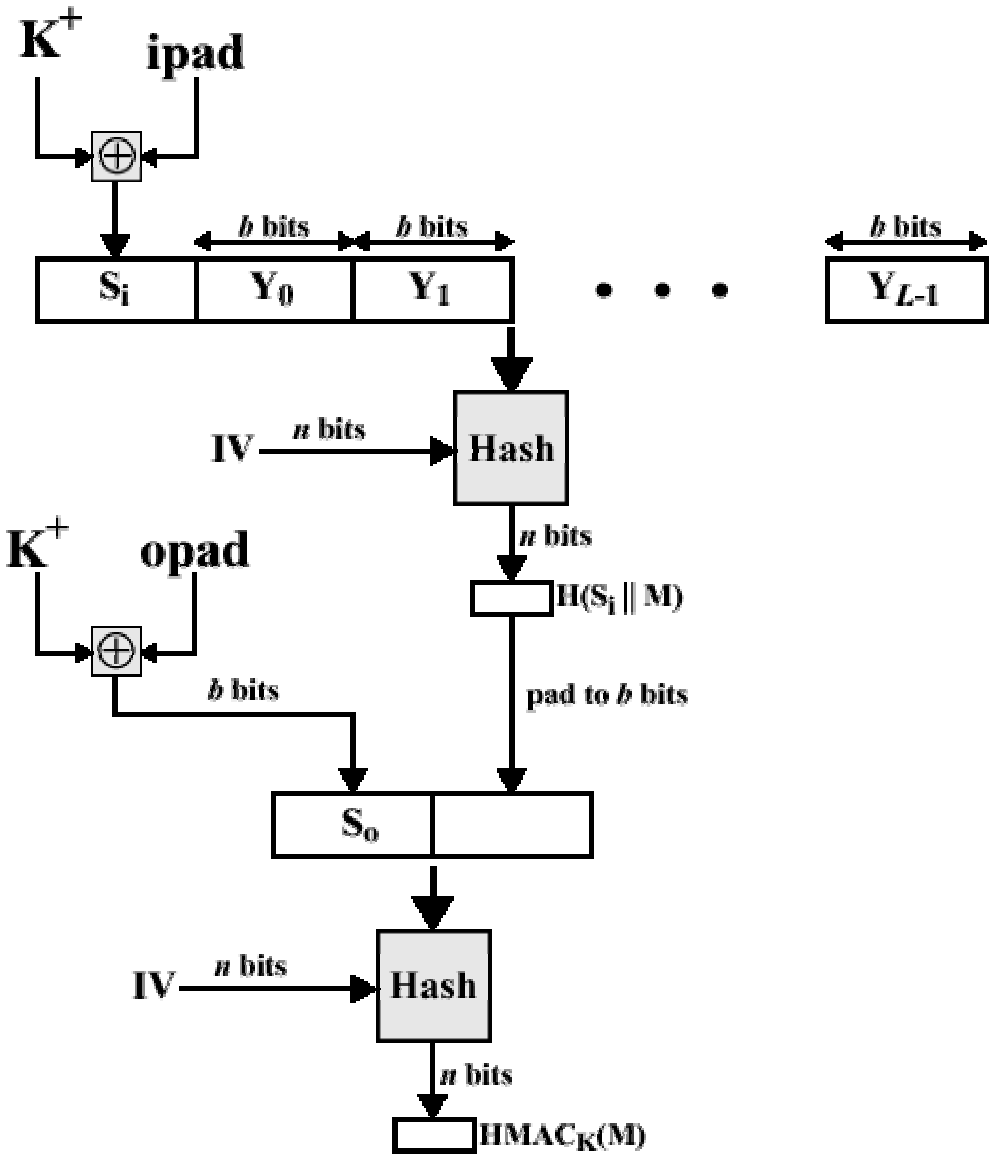
Master Secret forts.

- master_secret=
  MD5(p_m_s || SHA('A'||p_m_s||
  ClientHello.rand||ServerHello.rand)) ||
  MD5(p_m_s || SHA('BB'||p_m_s||
  ClientHello.rand||ServerHello.rand)) ||
  MD5(p_m_s || SHA('CCC'||p_m_s||
  ClientHello.rand||ServerHello.rand))

Generering av nøkkelmateriale

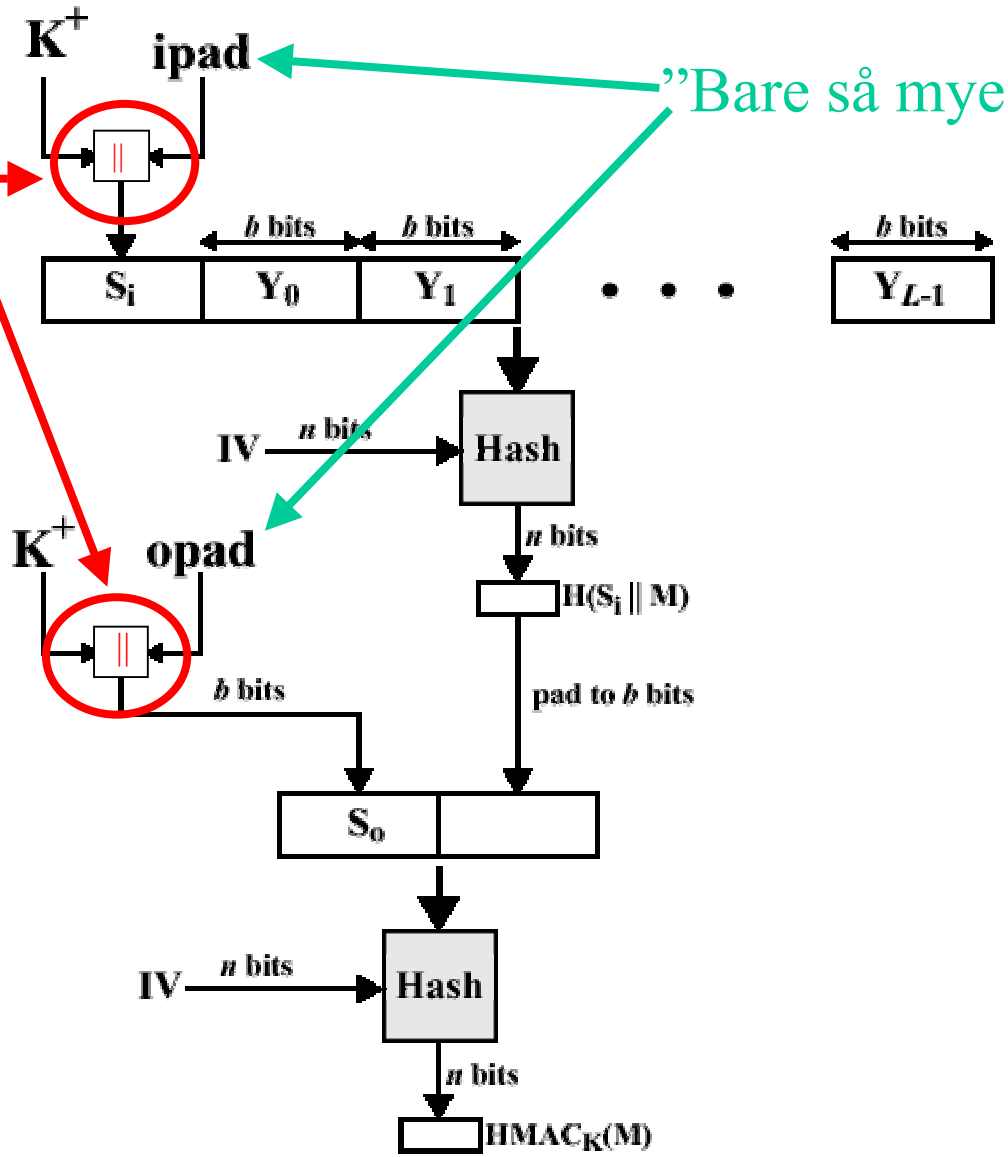
- På en tilsvarende måte brukes så master_secret for å generere nøkkelmateriale for sesjonsnøkler
- Disse funksjonene er pseudo-random-funksjoner med ”random”-verdiene som ”salt”

Standard HMAC



HMAC i SSLv3

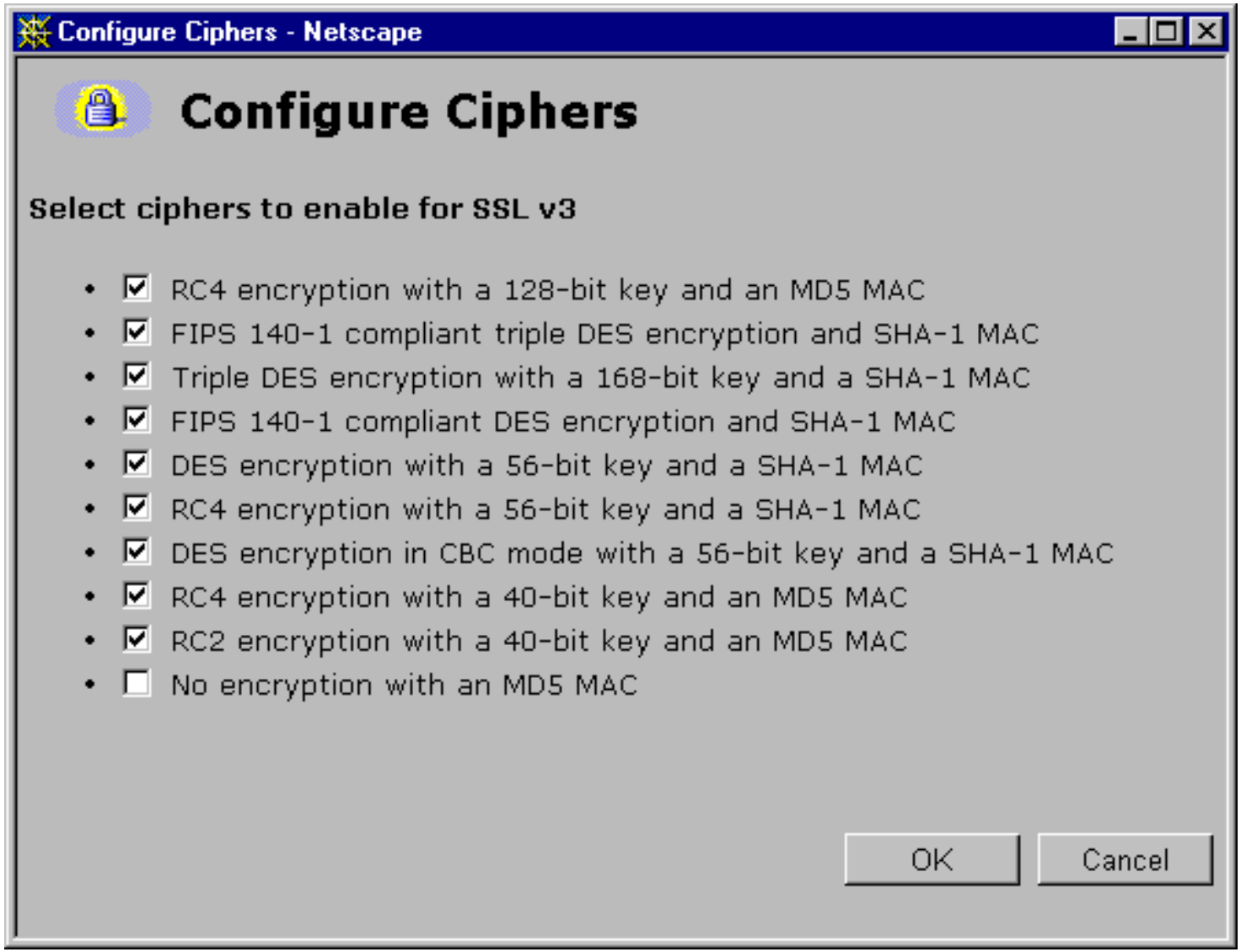
Konkatenering



Nettbank og Netscape

- Fellesdata driver mange bankers nettbank
- Benytter et sertifikat utstedt av Verisign
- Krypteringsalgoritme: RC4 128 bit

Hvilke algoritmer kan du bruke?



Et nettbank-sertifikat

View A Certificate - Netscape

This Certificate belongs to:
nettbank.fellesdata.no
Terms of use at
www.verisign.com/rpa (c)00
NettBank
Fellesdata AS
Oslo, Oslo, NO

This Certificate was issued by:
www.verisign.com/CPS Incorp.by Ref.
LIABILITY LTD.(c)97 VeriSign
VeriSign International Server CA -
Class 3
VeriSign, Inc.
VeriSign Trust Network

Serial Number: 29:94:52:0C:60:43:69:42:99:8D:F7:ED:7F:7C:B1:CC

This Certificate is valid from Fri Jul 28, 2000 to Fri Aug 24, 2001

Certificate Fingerprint:
7B:78:46:0B:57:C7:A8:B8:BD:A8:99:83:CA:D5:4A:64

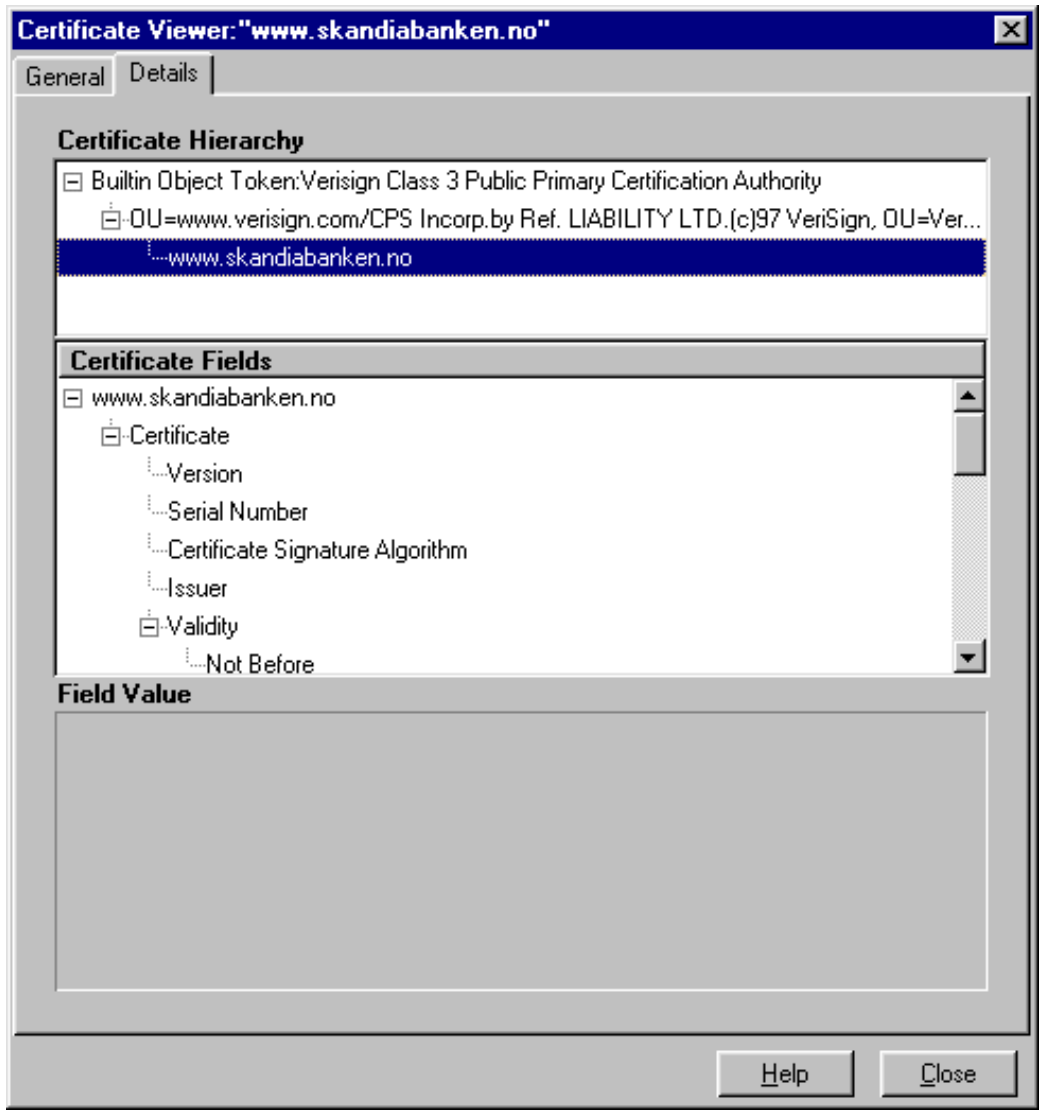
OK

Transport Layer Security (TLS)

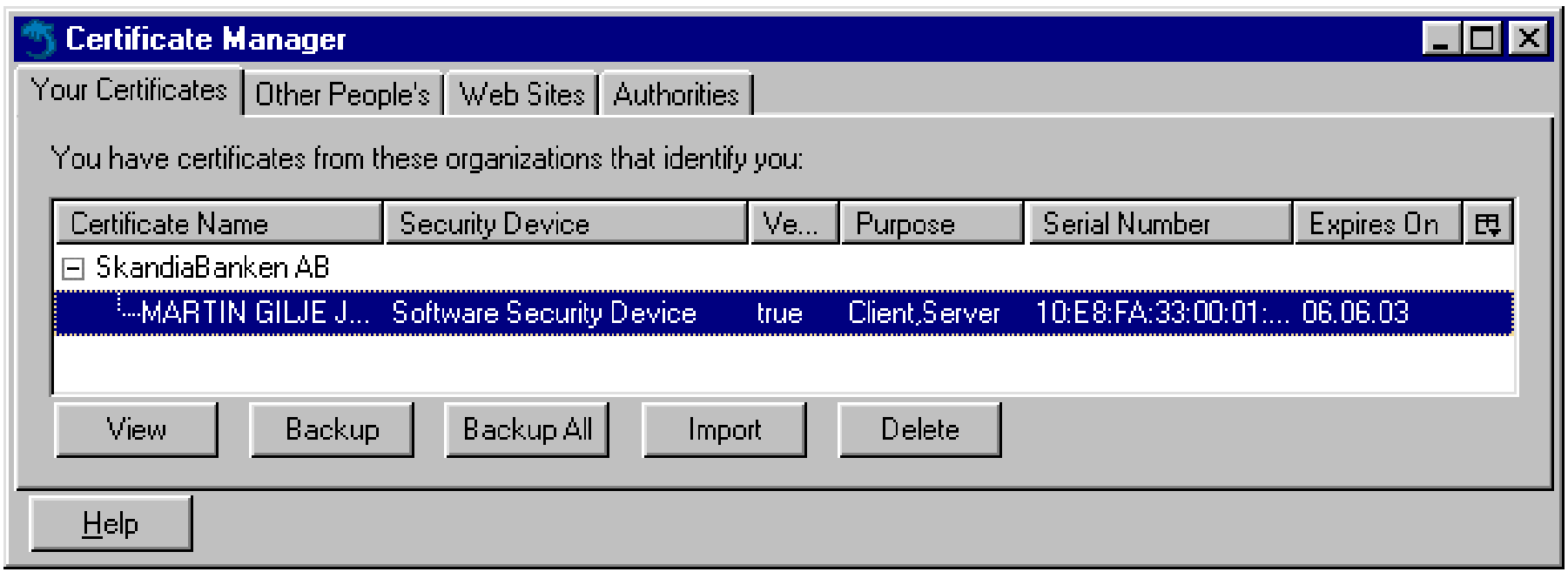
- Arvtageren til SSL
- TLS har en RFC (det har ikke SSL)

...og mens vi snakker om banker

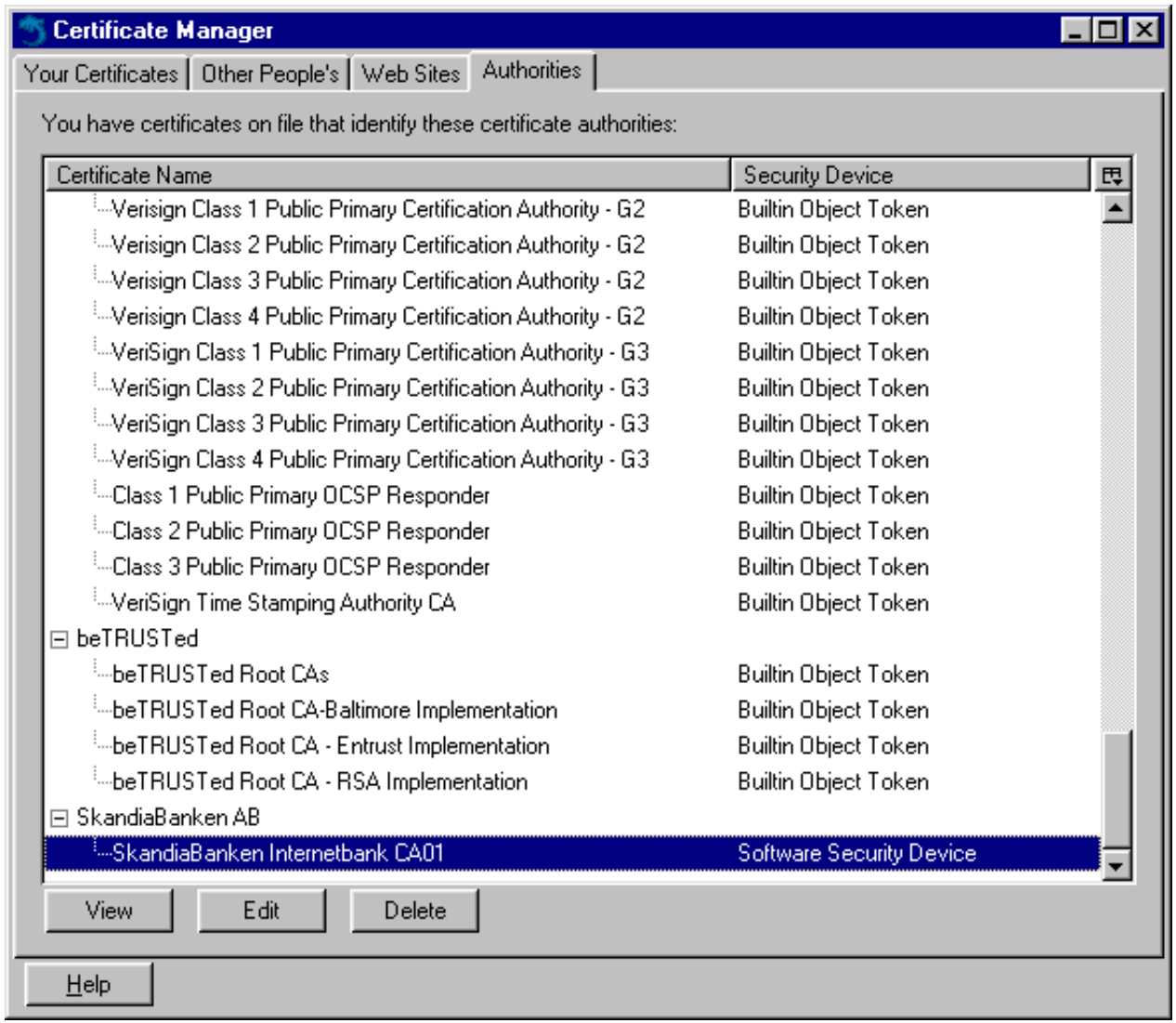
Mer detaljert



Klient-sertifikat



...og til slutt

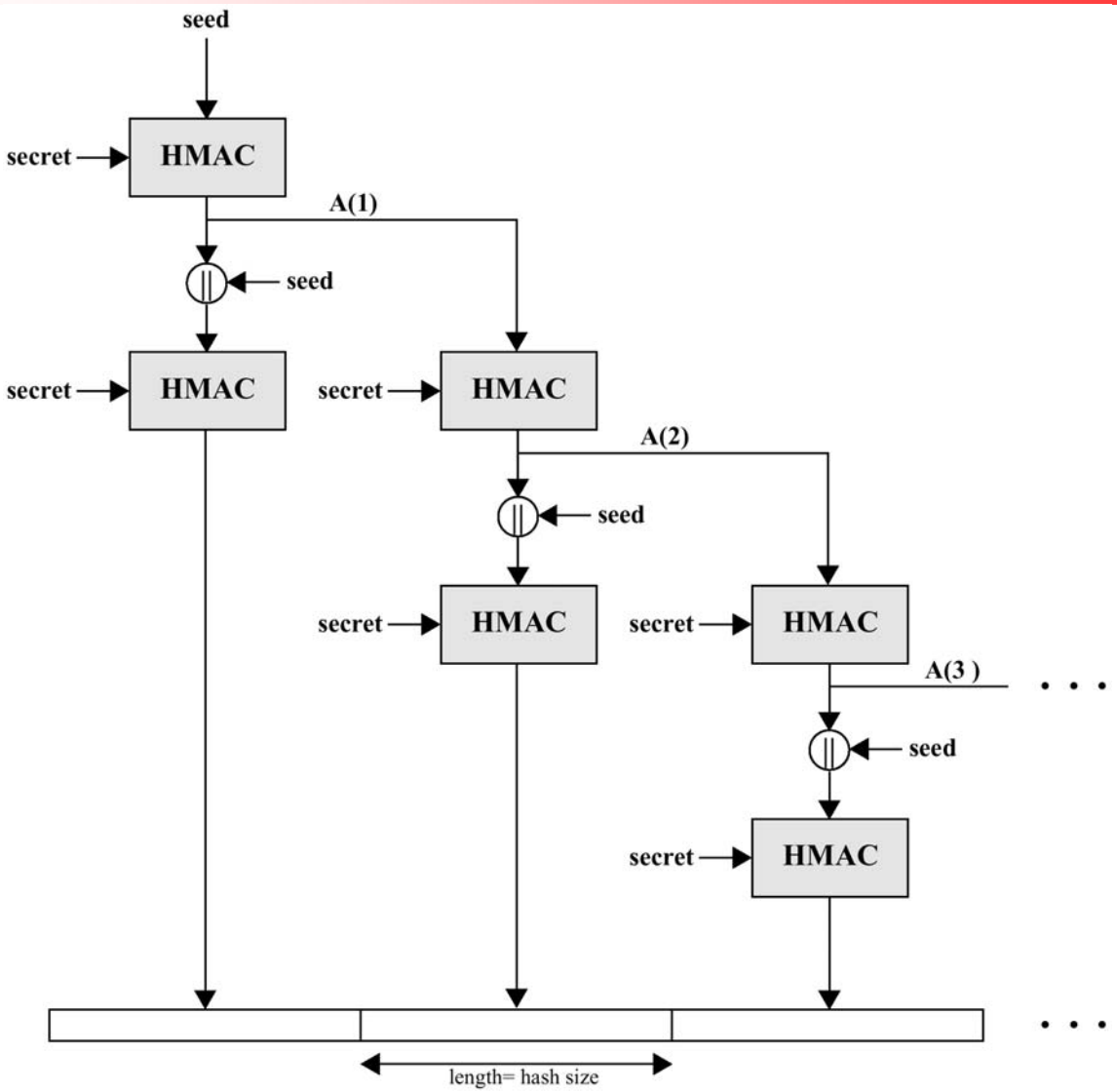

Certificate Manager
Your Certificates
Other People's
Web Sites
Authorities
You have certificates on file that identify these certificate authorities:
Certificate Name
Security Device
Verisign Class 1 Public Primary Certification Authority - G2
Builtin Object Token
Verisign Class 2 Public Primary Certification Authority - G2
Builtin Object Token
Verisign Class 3 Public Primary Certification Authority - G2
Builtin Object Token
Verisign Class 4 Public Primary Certification Authority - G2
Builtin Object Token
VeriSign Class 1 Public Primary Certification Authority - G3
Builtin Object Token
VeriSign Class 2 Public Primary Certification Authority - G3
Builtin Object Token
VeriSign Class 3 Public Primary Certification Authority - G3
Builtin Object Token
VeriSign Class 4 Public Primary Certification Authority - G3
Builtin Object Token
Class 1 Public Primary OCSP Responder
Builtin Object Token
Class 2 Public Primary OCSP Responder
Builtin Object Token
Class 3 Public Primary OCSP Responder
Builtin Object Token
VeriSign Time Stamping Authority CA
Builtin Object Token
beTRUSTed
beTRUSTed Root CAs
Builtin Object Token
beTRUSTed Root CA-Baltimore Implementation
Builtin Object Token
beTRUSTed Root CA - Entrust Implementation
Builtin Object Token
beTRUSTed Root CA - RSA Implementation
Builtin Object Token
SkandiaBanken AB
SkandiaBanken Internetbank CA01
Software Security Device
View
Edit
Delete
Help

Forskjell mellom SSL og TLS

- TLS bruker standard HMAC
- TLS har en egen pseudorandom funksjon for å generere nøkkelmateriell (P_(hash))
- TLS har definert en del nye ”alert codes”
- En del andre mindre forskjeller

TLS P_(hash)-funksjon

Enten P_MD5 eller P_SHA-1, avhengig av hvilken hash HMAC brukes med.



SET

Secure Electronic Transaction (SET)

- Rammeverk for å beskytte kredittkort-handel på Internett
- Utviklet etter initiativ fra MasterCard og Visa
- Bidrag fra IBM, Microsoft, Netscape, etc.

Hva tilbyr SET?

- Sikker kommunikasjonskanal mellom alle parter involvert i en transaksjon
- Oppnår tiltro (”trust”) ved hjelp av X.509-sertifikater
- Bidrar til personvern, ettersom informasjon bare er tilgjengelig for de deltakerne som har behov for den, og bare *når* de har behovet

SET krav

- Konfidensialitet for betalings- og bestillings-informasjon
- Integritet for alle sendte data
- Verifikasjon av at gitt kortbruker er en gyldig bruker av en konto
- Verifikasjon av at en butikk kan motta kredittkortbetaling via en avtale med en finansinstitusjon

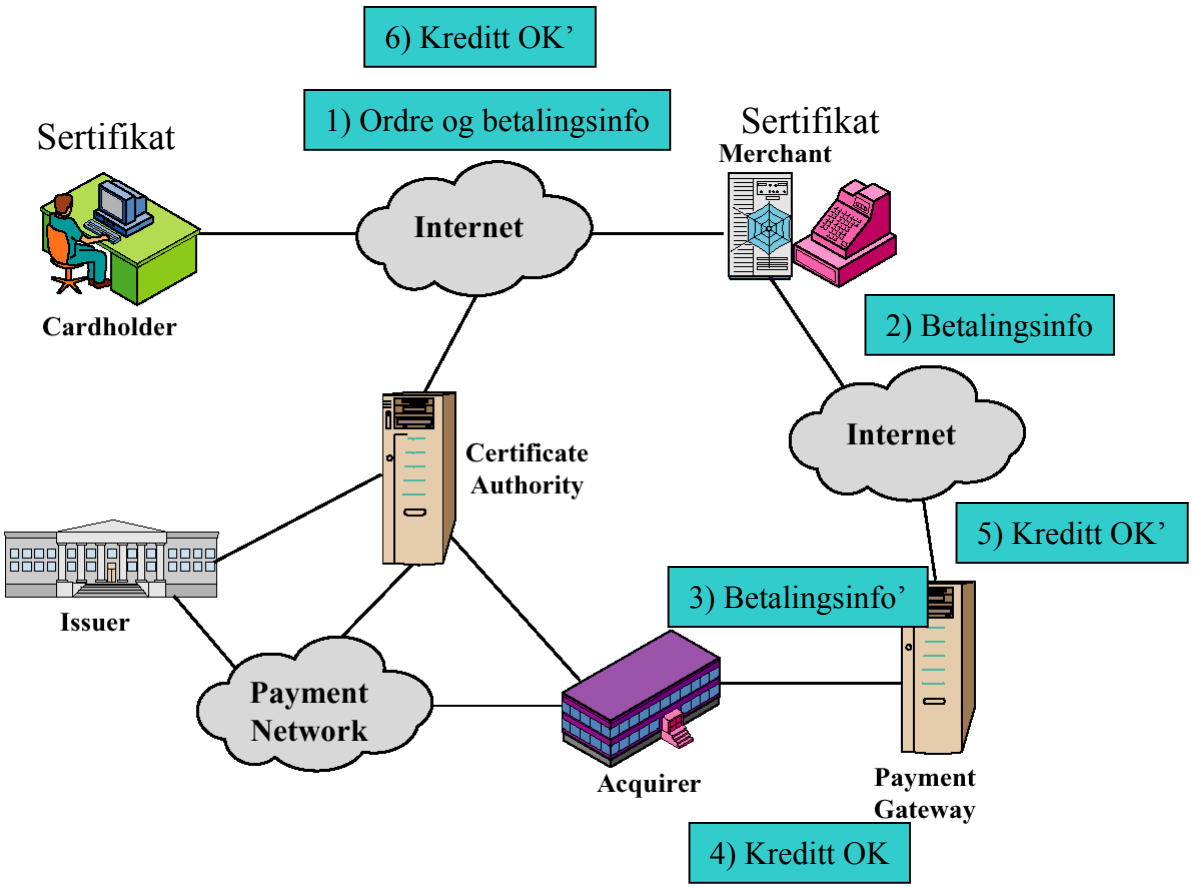
SET krav forts.

- Bruk av ”best mulig” sikkerhets-mekanismer og prosedyrer
- Protokollen skal være uavhengig av eventuelle sikkerhetsmekanismer i underliggende protokoller
- Protokollen skal være uavhengig av hardware, OS og nettleser

SET egenskaper

- Konfidensialitet av informasjon
- Integritet til data
- Autentisering av kortbrukere
- Autentisering av butikker

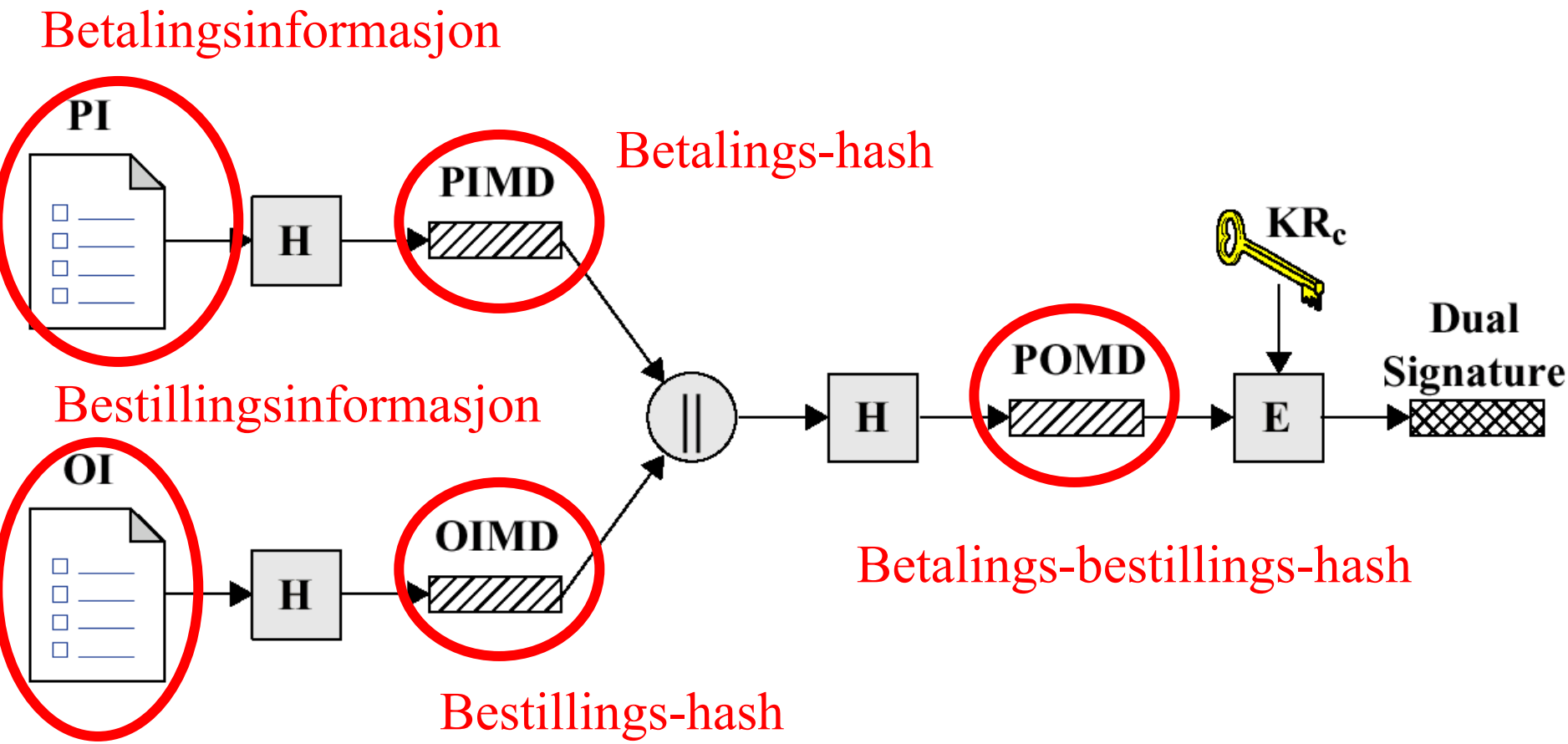
Deltakere i SET



Dobbel signatur

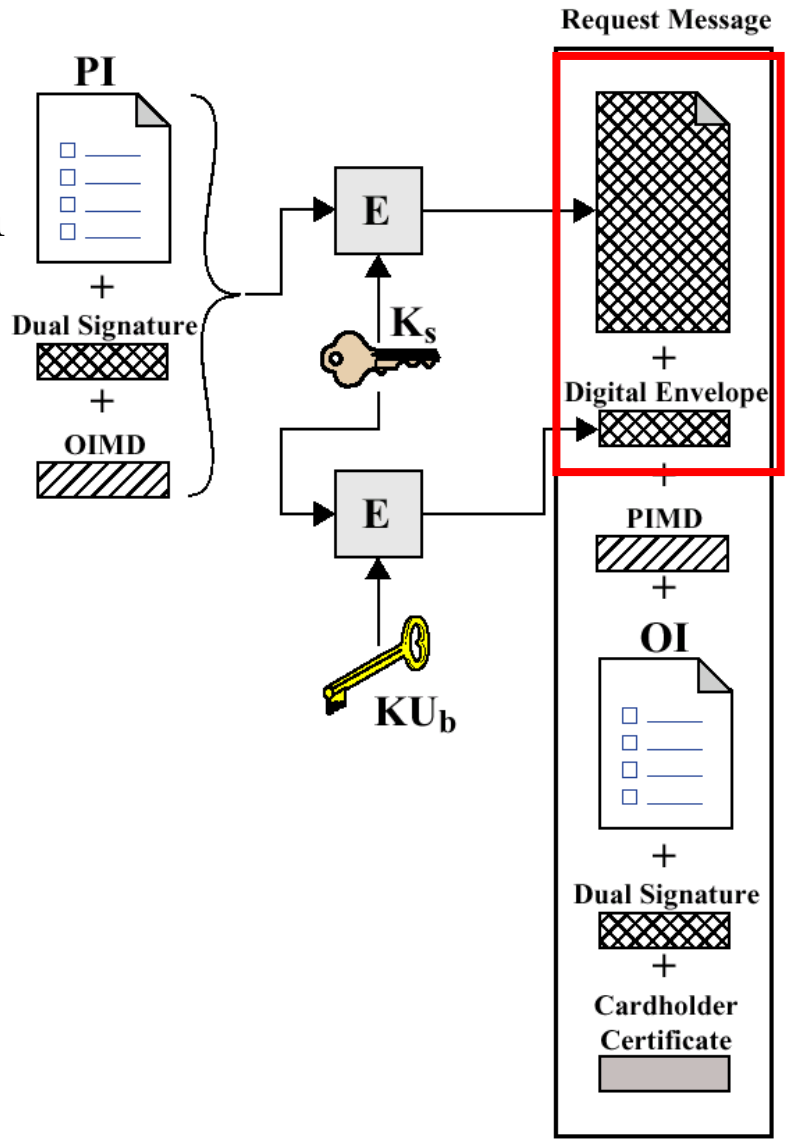
- For å hindre at en butikk skal kunne få betaling for noe annet enn det du har bestilt, lages en signatur basert på både bestillings-informasjon og betalings-informasjon
- Butikken får ikke tilgang til betalings-informasjonen, men en *hash* av denne

Generering av dobbel signatur



Bestilling

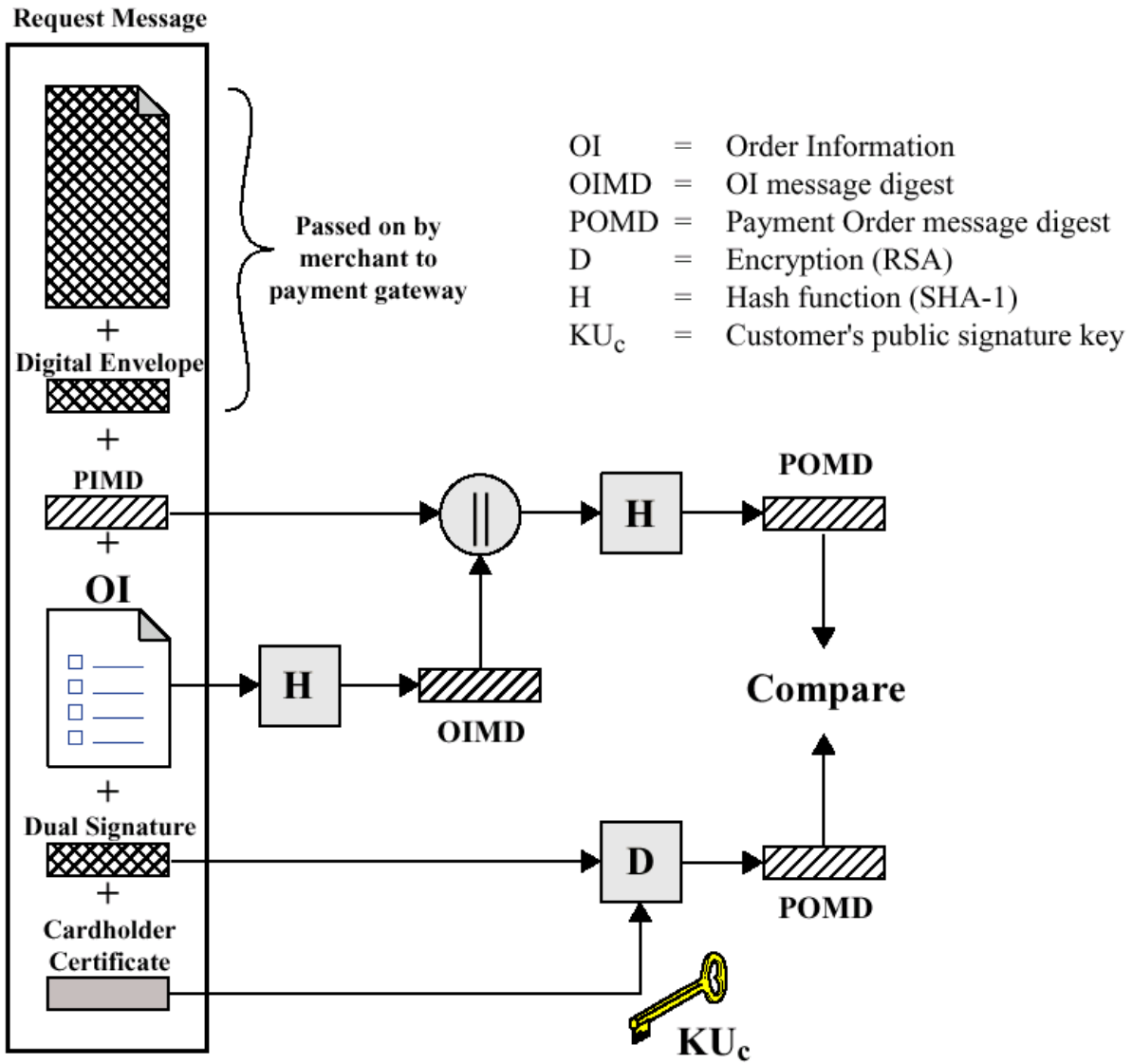
Betalings-
informasjon



Sendes videre
til betalings-
formidler av
butikken

Bestillings-
informasjon

Verifikasjon av bestilling



Praktisk bruk av SET

- Brukes praktisk talt ikke!
- Kortselskapene har ikke vært flinke nok til å ”selge” konseptet
- Hvem skal betale?
- Et genialt konsept er verdiløst hvis det ikke markedsføres skikkelig!

SSH

Secure Shell - SSH

- SSH er en protokoll for sikker fjern-innlogging og andre sikre tjenester over et usikkert nettverk
- SSH består av
  - Transportlagsprotokoll
  - Brukerautentiseringsprotokoll
  - Forbindelsesprotokoll

SSH Transport

- SSH Transport er en sikker ”lavnivå” (lower layer) protokoll
- Tilbyr konfidensialtitet, autentisering av maskin (server) og integritet
- Bruker en protokoll som ligner SSL for å forhandle fram krypteringsalgoritmer, nøkkelutvekslingsalgoritmer, etc.

Grunnleggende SSH

- En server må ha et offentlig-nøkkel-sertifikat
- Klienten og serveren kommer gjennom nøkkelforhandling fram til en felles (symmetrisk) sesjonsnøkkel
- Resterende kommunikasjon krypteres med sesjonsnøkkelen
- Brukeren på klienten kan autentisere seg enten ved passord, eller bruk av et eget sertifikat

Dagen website

- http://www.ssh.com

  Protokoll-utkast, nedlasting (gratis for ”academic” bruk!)
- Firmaet SSH leverer også (bl.a.) en IPSec-implementasjon