

TLS

RFC2246: The TLS Protocol

What does it achieve?

- Confidentiality and integrity of the communication
- Server authentication
- Eventually: client authentication

What is does not do

- Protect the server
- Protect the client
- Protect stored data
- Provide any liability by itself
- See PKI story

The TLS Handshake Protocol

- Three sub-protocols used to allow peers:
 - To agree upon security parameters for the record layer
 - To authenticate themselves
 - To instantiate negotiated security parameters
 - To report error conditions to each other
- Change cipher spec protocol
- Alert protocol overview
- Handshake Protocol

Session

The Handshake Protocol is responsible for negotiating a session, which consists of the following items:

- session identifier: to identify an active or resumable session state.
- peer certificate: X509v3 [X509] certificate of the peer.
- compression method
- cipher spec: bulk data encryption algorithm and MAC algorithm (MD5/SHA)
- master secret: 48-byte secret shared between the client and server.
- is resumable: flag indicating whether the session can be used to initiate new connections.

Change cipher spec protocol

- to signal transitions in ciphering strategies
- Switch: pending state becomes current state

Alert protocol

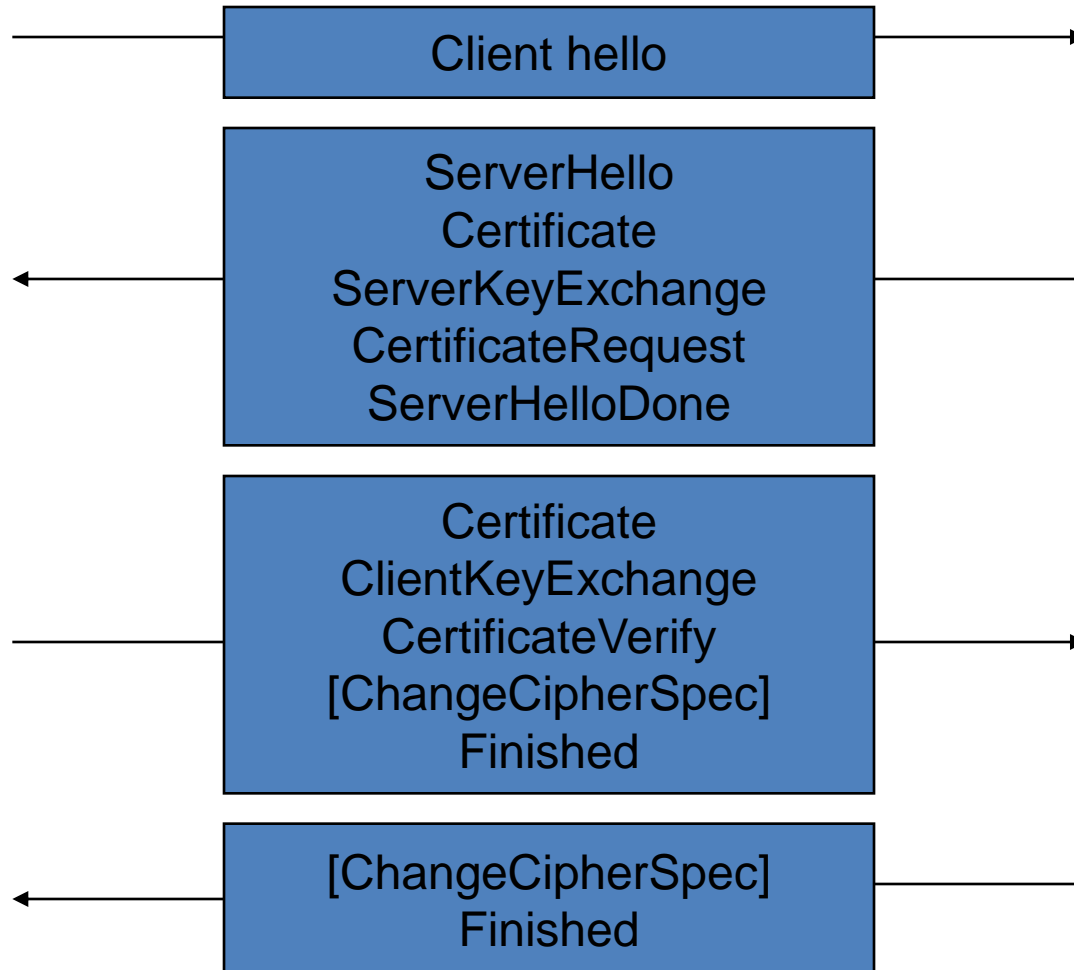
- Alert messages
 - the severity of the message
 - A description of the alert
- “fatal” messages: immediate termination of the connection
 - Closure alerts
 - Error alerts

Handshake Protocol overview

Steps:

- Exchange “hello” messages to agree on algorithms
 - exchange random values
 - check for session resumption.
- Exchange cryptographic parameters
- Client and server agree on a premaster secret.
- Exchange certificates and cryptographic information to allow the client and server to authenticate themselves.
- Generate a master secret from the premaster secret and exchanged random values.
- Provide security parameters to the record layer.
- Allow the client and server to verify that their peer has calculated the same security parameters and that the handshake occurred without tampering by an attacker.

TLS messages



Starting TLS

- Client: client hello
- Server: server hello
- Info:
 - Protocol Version
 - Session ID
 - Cipher Suite
 - Compression Method
 - ClientHello.random
 - ServerHello.random

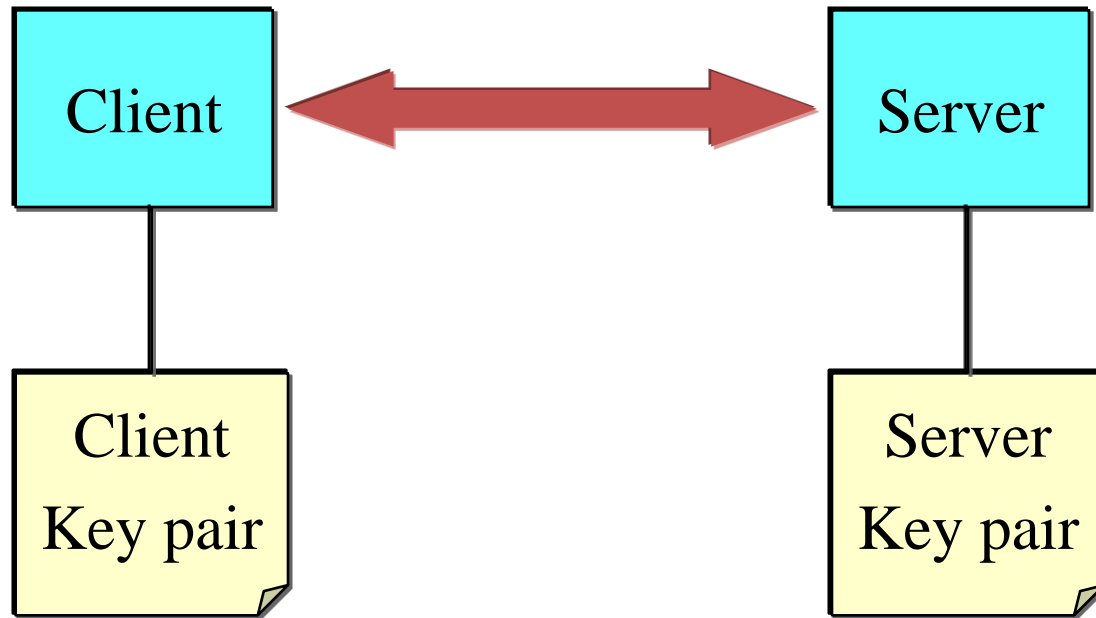
TLS key exchange

- Up to four messages:
 - the server certificate
 - the server key exchange
 - the client certificate
 - The client key exchange.

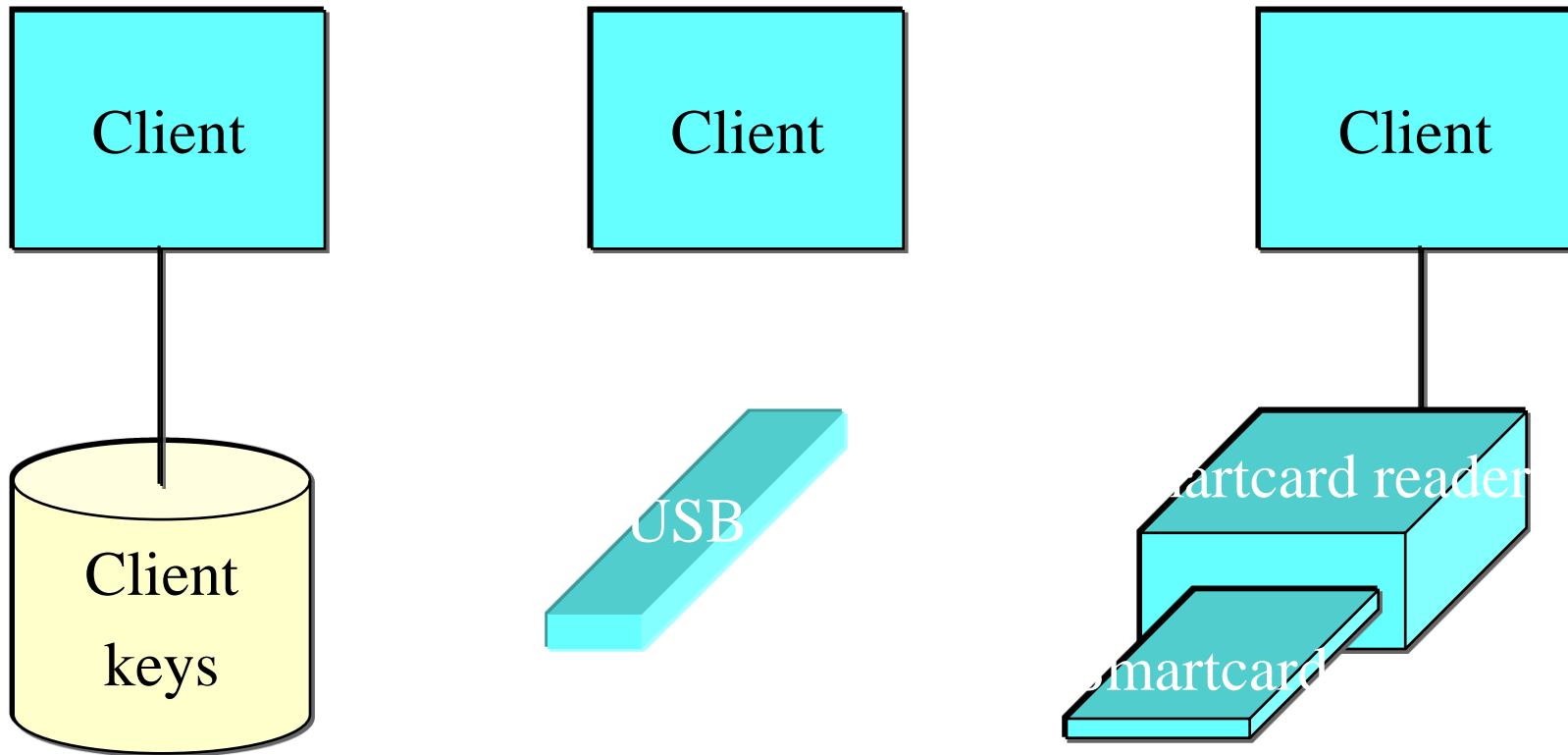
End TLS negotiation

- Server: “hello done” message
- Client: “change cipher spec” message

Client-server



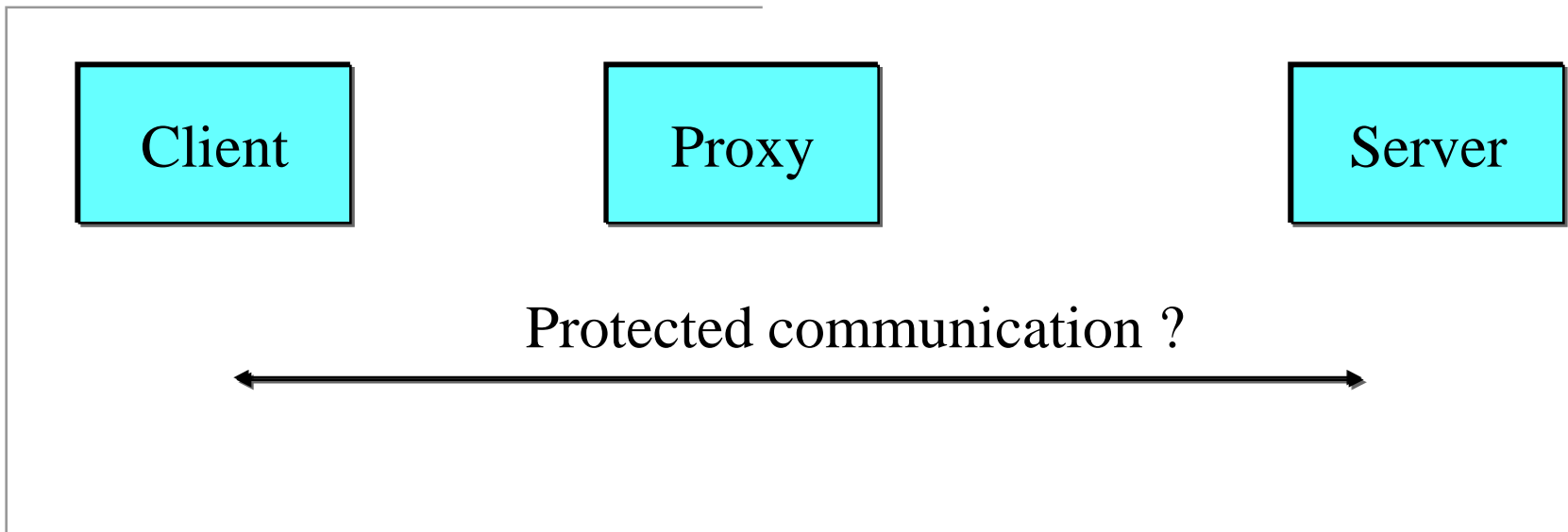
Client: key store



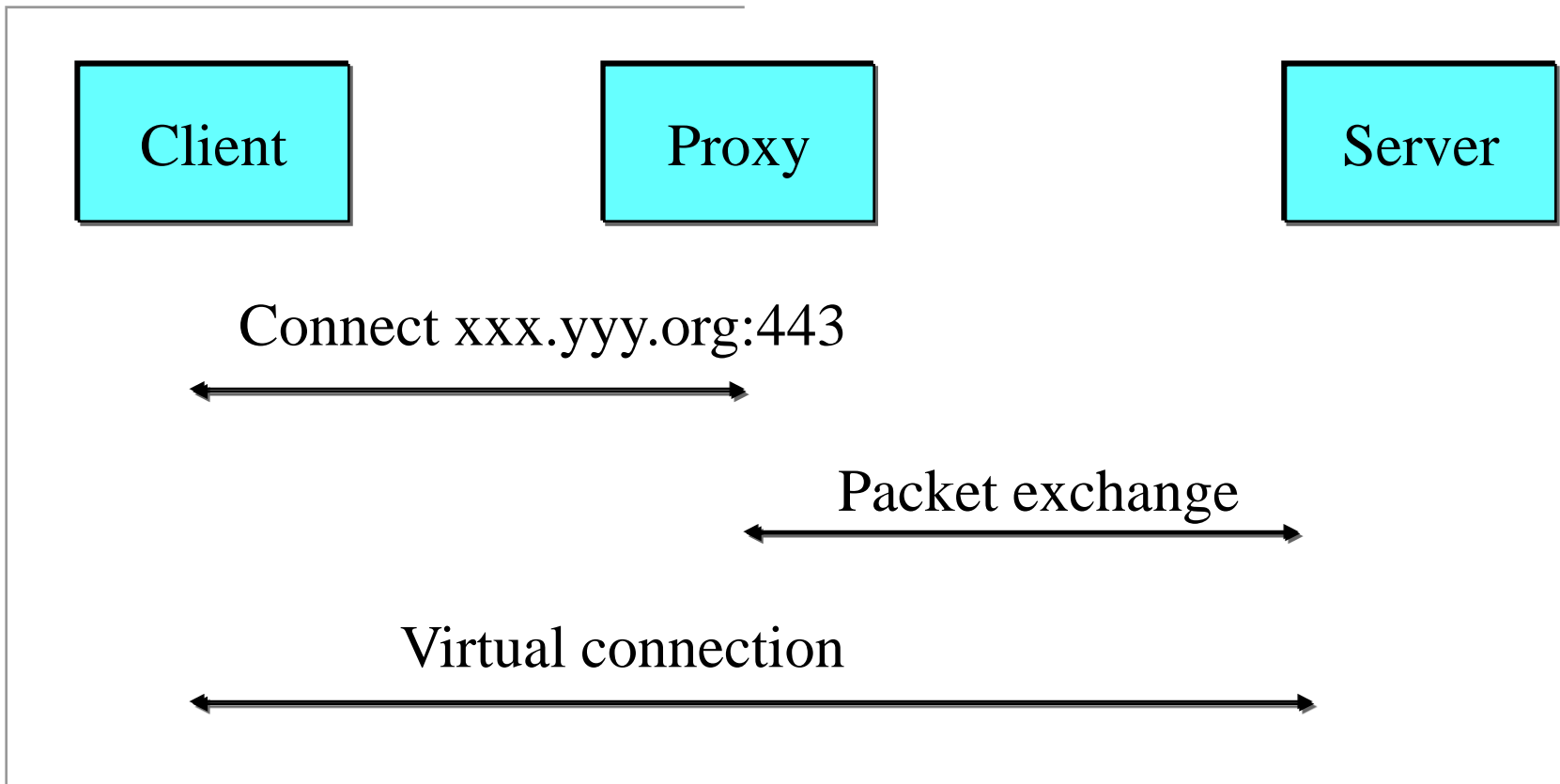
Client – store

- High-level interfaces:
 - PKCS#11 interface
 - Microsoft: CSP
- Smartcard APIs
- Disk storage:
 - Key encryption (password derived key)
 - Wallets

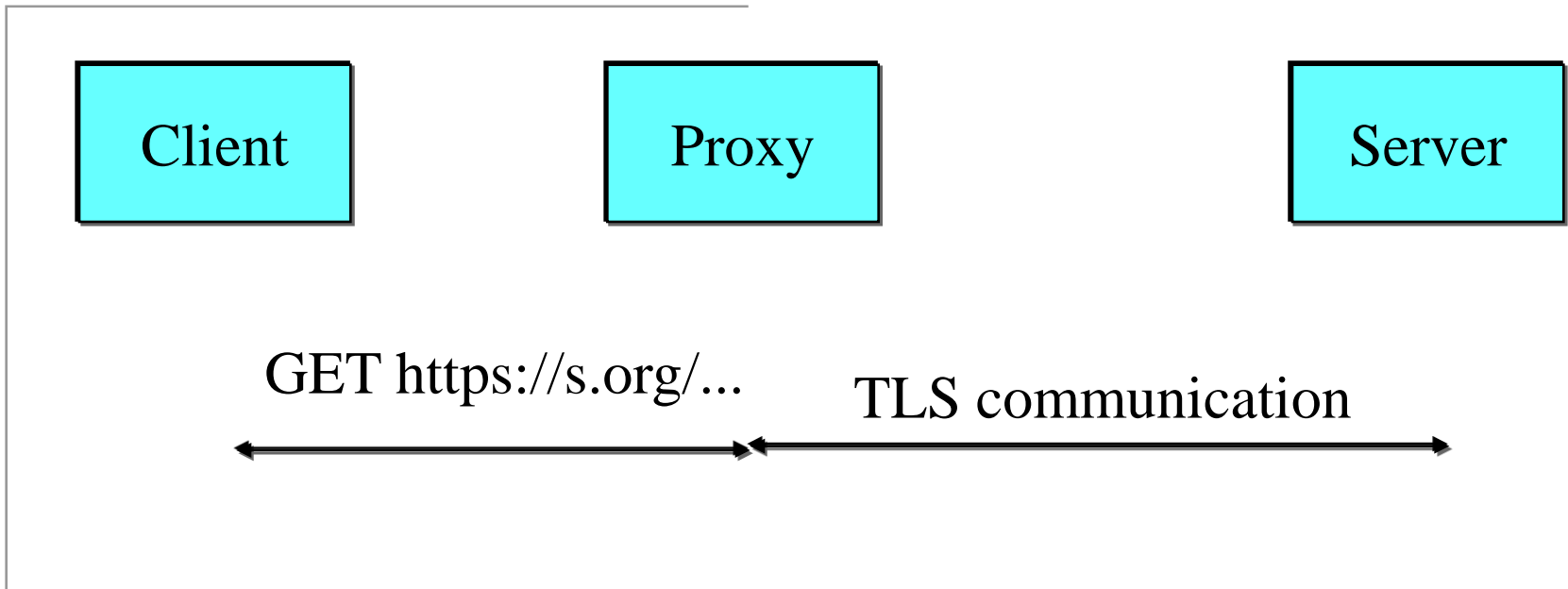
Client: via proxy



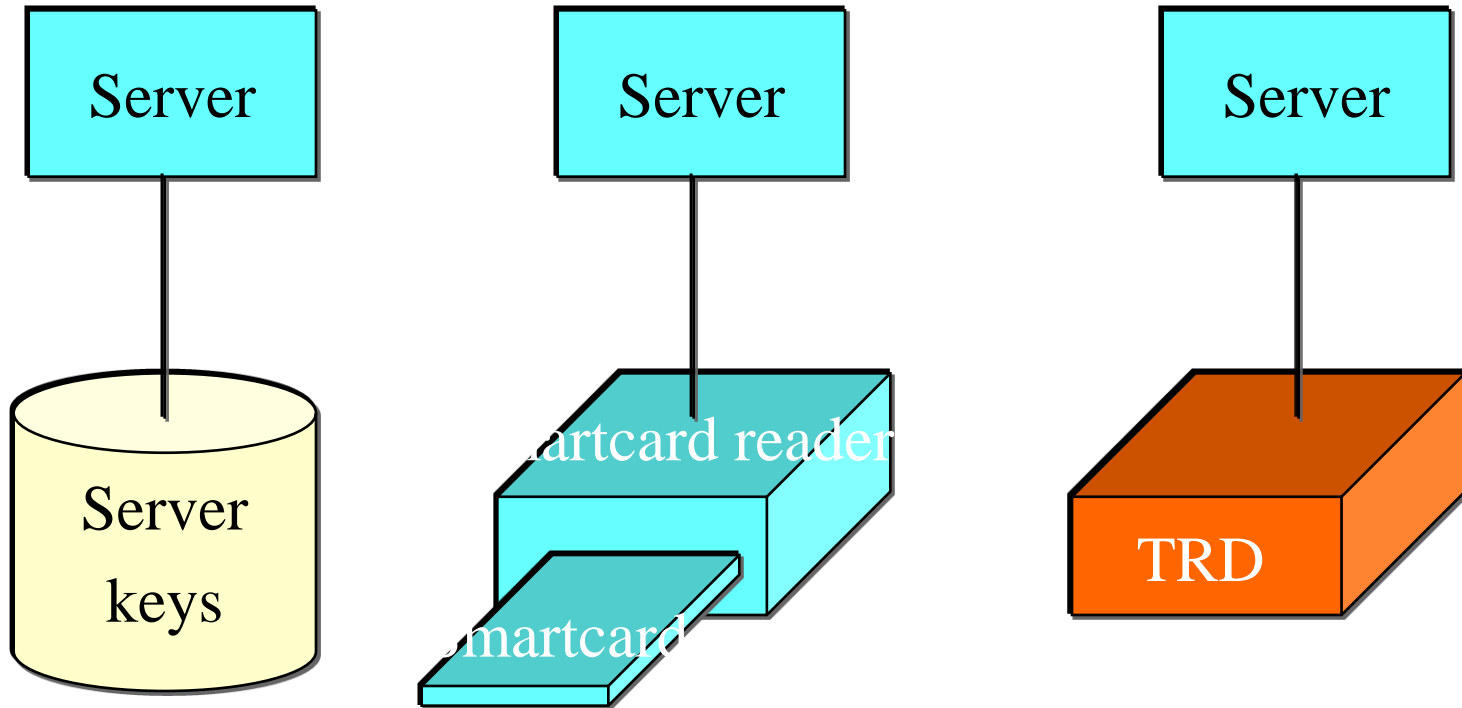
Client: via proxy (1)



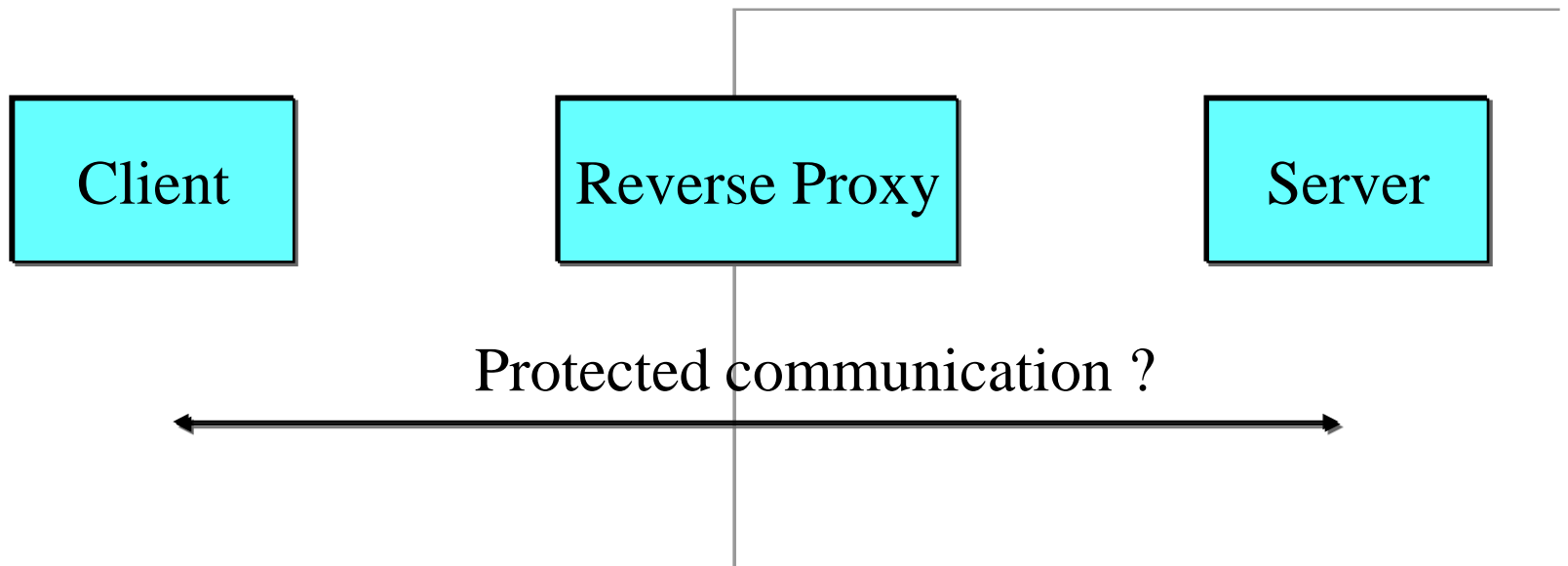
Client: via proxy



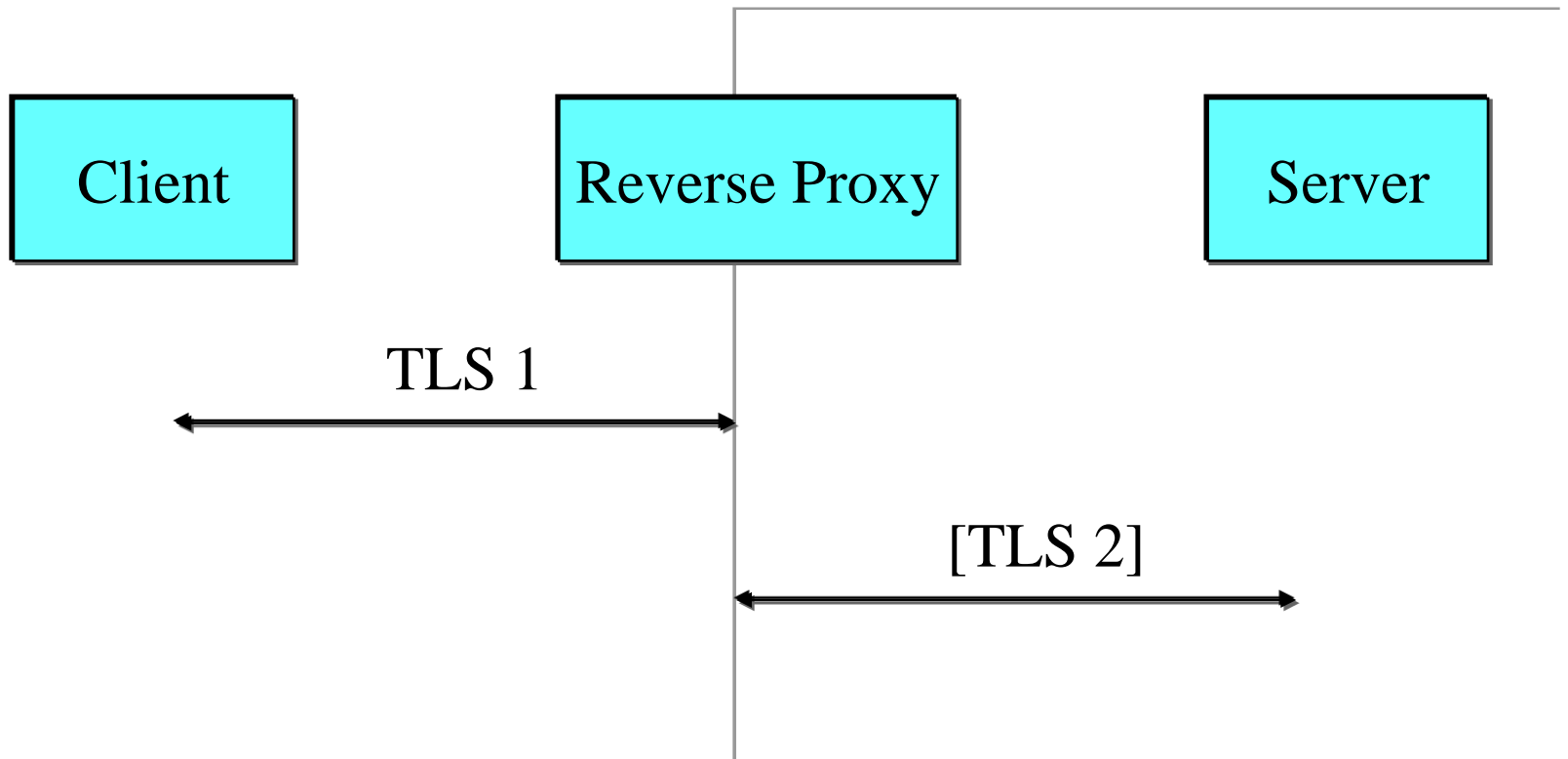
Server: key store



Server: via reverse proxy



Server: via reverse proxy



openssl

- Public implementation
- Hooks into Apache (and others)
- All common functionalities
- Vendors of TRD link into it

PKI

Public Key Infrastructure

Fantastic technology...

- Public/private key technology is a magnificent solution for
 - Key exchange
 - Digital signatures
- But key management is a nightmare, much like with symmetric keys

Critical element

- To encrypt or someone, use his public key
- To check signature, use his public key
- CRITICAL: link between public key and entity, otherwise no use
- Need to get the public key
- Need to trust the public key
- Intruder can fake it all...

Certificates: why

- Certificates are an important concept to make PK manageable
- Replace individual links and trust by trust in CA
 - CA signs statements on individual key pairs
 - Need only to trust the CA

Certificates: what

- Signed statement by CA about link between public key and entity
- Assumed:
 - Trust that CA checked possession of private key by the entity
 - Trust in CA itself: the CA can fake anything

PKI: main elements

- Issuing certificates
- Trust models
- Certificate publishing
- Certificate suspension and revocation

CA operation

- CA issues certificates (signs pieces of data)
- CA publishes certificates (to allow customers to find certificates, without prior contact)
- CA publishes revocation lists (LDAP server)
- CA uses Registration Authoritie(s) (RA) to establish link between key and entity

RA – CA

- CA has certificate practice statement (CPS): rules of operation
- CA assign RA and gives guidelines
- RA may have multiple Local RA (LRA)
- (L)RA establishes and registers entity/public key link

RA checking

- Physical verification
- Paper trail checks
- Round trip checks
 - Send mail with code: check address (access)
 - Send email with code: check email address (access)
- Key possession checking
 - Challenge response: generate challenge, request must be encrypted with matching private key

Trust models

- Top level CAs
- Subordinate CAs
 - People certificates
 - Level 1, level 2, level 3
 - Server certificates (SSL)
 - Code signing certificates (activeX, java jars)
 - VPN certificates
 - CRL/OCSP signing

Certificate publishing:search

- Web based search & import mechanisms

Certificate suspension and revocation

- LDAP
 - Globalsign:
 - Host : `directory.globalsign.net`
 - port LDAP : 389
 - port LDAPS : 636
 - Base DN : `dc=globalsign, dc=net`
 - CRL: `ou=Class 1 CA`
 - Attribute: `certificateRevocationList`

eID – Belgium

- Electronic identity card
- Cards: company a
- Certificates: company b
- Important role: 'rijksregister'

Information

- Name, first name(s), birthdate, gender, hash of photo
- RRN, Nationality
- Card number, validity, issuing municipality
- Keys, certificates
 - Authentication
 - Signing

CA

- Belgian root CA
 - Card CA
 - Citizen CA
 - Authentication
 - Signing
 - Government CA
 - Code sign
 - Server sign
 - RRN

Revocation

- CRLs
- Delta CRLs
 - Making CRL practical
 - Alternative for OCSP

Operations

- RA: Rijksregister
- LRA: municipality
- Card production
- Certificate production
- CRL and delta CRL production