



# Encrypted Message Transport and Certificate based Relaying in Postfix

Patrick Koetter

Linuxforum 2005  
Kopenhagen



# TOC

- TLS Primer
- Certificates
- Server-side TLS
  - Basic setup
  - Session caching
  - Enforcing TLS
  - Protecting SASL
- Client-side TLS
  - Basic setup
  - Session caching
  - Protecting SASL
  - Selective TLS
- Certificate based relaying
  - Server setup
  - Client setup



## Why talk about TLS in Postfix now?

- TLS as defined in [RFC 3207](#) will become part of the regular Postfix source tree in version 2.2
- The code has been rewritten completely
- Some things have changed
- Some things have been added

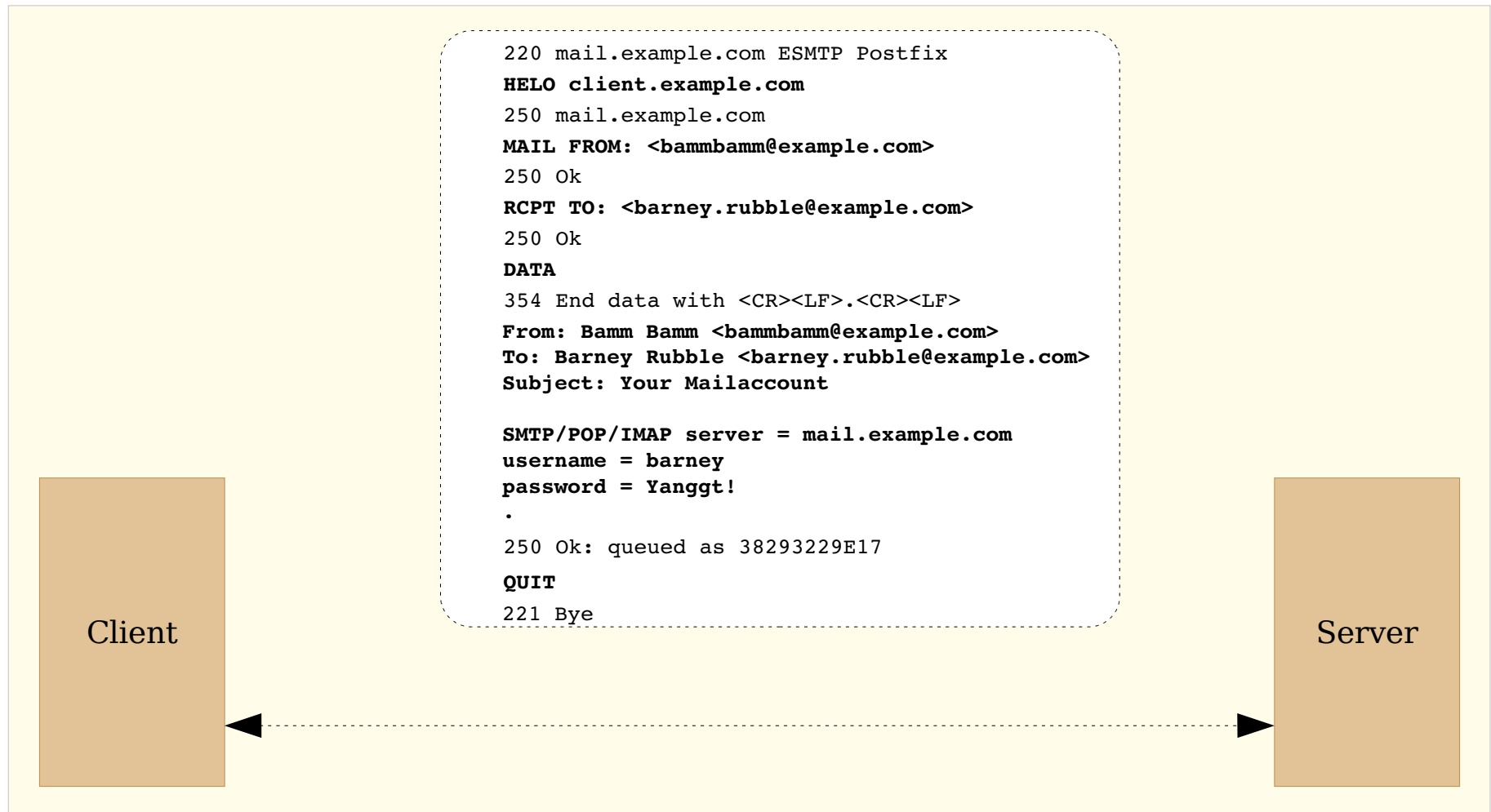


# What is Transport Layer Security?

- TLS is a method to encrypt the Transport Layer between to hosts.
- TLS is the successor of SSL.
- It gives you
  - Privacy
  - Integrity
  - Authenticity
  - Access Control

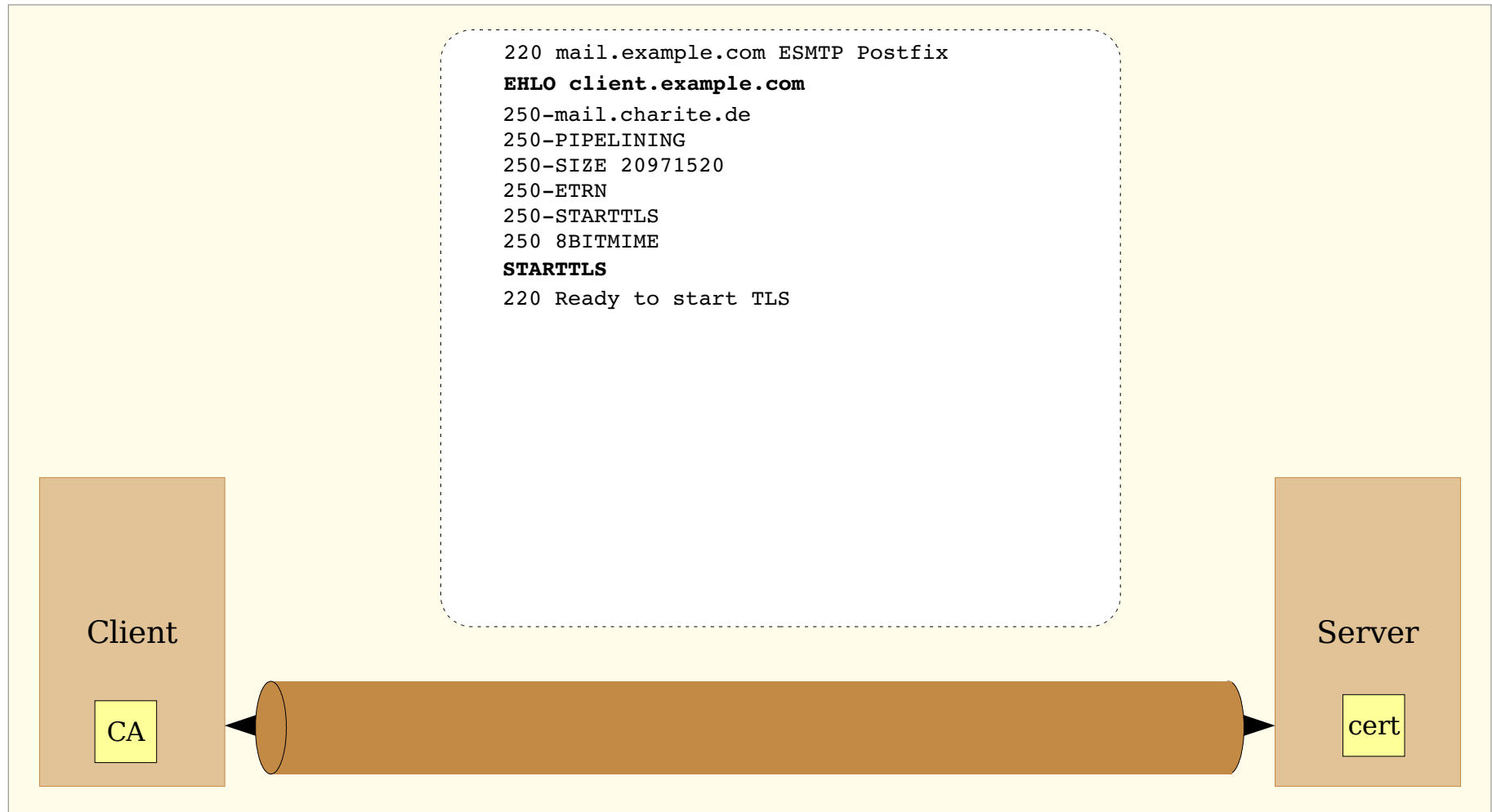


# Unencrypted SMTP





# TLS encrypted ESMTP





# Technical Security aspects of unencrypted and TLS encrypted SMTP

	Unencrypted SMTP	Encrypted SMTP
Privacy		✓
Integrity		✓
Authenticity		✓
Availability		
Controlled Access	✓*	✓

\* IP based rules or SMTP AUTH rules



# Common Misunderstandings of TLS

- **TLS only protects the transport between two hosts**
  - If the message needs to be transported further it could be transported without TLS.
  - If the message gets bounced it could take a different route without TLS.
- **TLS only protects the transport, but not the storage**
  - The moment the message is written to the mail queue it is unencrypted.
    - > Encrypt data with PGP or S-MIME
  - The moment the message is written to the mailbox it is unencrypted.
    - > Encrypt data with PGP or S-MIME



# Postfix TLS functionalities

- Server-side Transport Layer Security  
The Postfix smtpd server offers TLS to receive mail from clients.
- Client-side Transport Layer Security  
The Postfix smtp client uses TLS to send mail.
- Certificate Based Relaying
  - Postfix relays mail for a remote client because a rule based on the client certificate permits this.
  - The smtp client uses its certificate to acquire relay permission from a mail relay.



# Certificates - the basis for TLS

- The Certification Authority (CA) signs the certificate request of a certificate owner. The CA guarantees the **authenticity** of the certificate.
- Official CA vs. Private CA
  - **Official Use**  
Buy an official certificate if your business demands it.
  - **Private Use**  
Create your own CA if you run a private network only.
  - **Mixed Use**  
Be aware that others might refuse to transport mail to or accept mail from your server if they can't verify your (private) certificates validity.



# Create a Certification Authority

1. Create the CA directory structure and the CA key and certificate

```
# /usr/share/ssl/misc/CA.pl -newca
```

2. Convert the CA Cert for Windows users:

```
# openssl x509 \  
-in /usr/local/ssl/misc/demoCA/cacert.pem \  
-out /usr/local/ssl/misc/demoCA/cacert.der \  
-outform DER
```



# Key and Certificate Requirements

- Whenever Postfix starts and stops it must access the key  
—> The key and the certificate request **must not** be password protected.
- The **identity** of a TLS server is verified by comparing the CN value with the hostname.  
—> The CN in the certificate **must** match the hostname
- A TLS client does not have to match its hostname with the CN value in the certificate.
- A TLS server serving multiple domains must have all hostnames in the certificate or clients will refuse to work.  
—> Use “Subject Alternatives” in the certificate.



## Creating the certificate key request

Use the `openssl` command to create a key (including the certificate request) being valid for 365 days:

```
# openssl req -new -nodes -keyout key.pem \  
-out key.pem -days 365
```



# Signing the Certificate Request

- **Official Certification Authority**  
Follow the procedure described by the Certification Authority you've chosen.
- **Private Certification Authority**  
Use the `openssl` command to sign the certificate request with your CA certificate:  

```
# openssl ca -policy policy_anything -out cert.pem \  
-infiles key.pem
```



# Preparing Postfix for Certificates and Key

- **Separate private from global certificates**

- Store Postfix' key and certificate separately accessible for Postfix only.
- Store the CA certificate that signed Postfix' certificate in a central CA root store shareable with other applications (unless you want to limit that too).

- **Protect the key**

The key is not password protected. You must limit key access to user `root`. Postfix will read the key as `root` before it drops privileges and runs as user `postfix`.



## CA root store: Best Practice

A CA root store contains all the CA certificates your application needs to verify certificates, but neither OpenSSL nor Postfix provide a collection of CA certificates.

- Use the `ca-bundle.crt` file that comes with the Apache Webserver.

```
$ locate ca-bundle.crt  
/usr/share/ssl/certs/ca-bundle.crt
```

- Add the CA certificate of the CA that signed Postfix' certificate to the root store. If your CA is part of a CA chain add all CA certificates from the CA chain.

```
# cat /usr/local/ssl/misc/demoCA/cacert.pem >> \  
    /usr/share/ssl/certs/ca-bundle.crt
```



# Basic server-side TLS configuration

## Basic TLS server parameters

```
## TLS Server configuration
smtpd_use_tls = yes
smtpd_tls_loglevel = 2
smtpd_tls_received_header = yes
tls_random_source = dev:/dev/urandom
smtpd_tls_key_file = /etc/postfix/certs/key.pem
smtpd_tls_cert_file = /etc/postfix/certs/cert.pem
```

## CA certs in one file:

```
smtpd_tls_CAfile = /usr/share/ssl/certs/ca-bundle.crt
```

## CA certs in separate files:

```
smtpd_tls_CApath = /usr/share/ssl/certs/
```

Use `c_rehash` to create an index if you use separate CA files.



# Caching Postfix TLS server sessions

- Cryptography puts load on the CPU.
- When smtpd processes terminate the session keys gets lost.
- Postfix can maintain an out of process session key database to lessen the burden
  - Expired keys must be deleted from the database
  - The database must be rebuilt when Postfix is restarted

```
smtpd_tls_session_cache_database =  
    btree:/etc/postfix/smtpd_scache
```

```
smtpd_tls_session_cache_timeout = 3600s
```



# Managing Postfix TLS sessions

## **tlsmgr**

tlsmgr is an additional daemon to manage TLS specific jobs

- Assist in creating random numbers on systems that can't do that themselves
- Clear expired session keys from the session key database
- Rebuild the session key database after Postfix has been restarted



# Enforcing server-side TLS

## Private networks only!

“A publicly-referenced SMTP server MUST NOT require use of the STARTTLS extension in order to deliver mail locally. This rule prevents the STARTTLS extension from damaging the interoperability of the Internet's SMTP infrastructure.” (RFC 2487)

```
## TLS Server configuration
...
smtpd_enforce_tls = yes
...
```



# Protecting Postfix server SMTP AUTH

Most SMTP servers offer plaintext mechanisms. Clients submit username and password encoded, but unencrypted.

TLS can protect the plaintext authentication.

- **General approach**

Offer SMTP AUTH only when TLS is used.

```
smtpd_tls_auth_only = yes
```

- **Specific approach**

Offer plaintext SMTP AUTH only when TLS is used.

```
smtpd_sasl_security_options = noanonymous, noplaintext  
smtpd_sasl_tls_security_options = noanonymous
```



# Basic client-side TLS configuration

## Basic TLS client parameters

```
smtp_use_tls = yes  
smtp_tls_loglevel = 2
```

## CA certs in one file:

```
smtp_tls_CAfile = /usr/share/ssl/certs/ca-bundle.crt
```

## CA certs in separate files:

```
smtp_tls_CApath = /usr/share/ssl/certs/
```

Use `c_rehash` to create an index if you use separate CA files.



# Caching Postfix TLS client sessions

Cryptography puts load on the CPU.

When smtp processes terminate the session keys gets lost.

Postfix can maintain an out of process session key database to lessen the burden

- Expired keys must be deleted from the database

The database must be rebuilt when Postfix is restarted

```
smtp_tls_session_cache_database =  
    btree:/etc/postfix/smtp_scache  
smtp_tls_session_cache_timeout = 3600s
```



# Controlling TLS in Postfix smtp client

Find out who offers TLS and limit whom Postfix smtp client uses TLS with:

```
smtp_tls_note_starttls_offer = yes
smtp_tls_per_site = hash:/etc/postfix/tls_per_site
```

There are four rules to apply:

```
# /etc/postfix/tls_per_site
dom.ain                NONE
host.dom.ain           MAY
important.host         MUST
some.host.dom.ain     MUST_NOPEERMATCH
```

The map will always override `main.cf` settings. If you turned off TLS, it will use TLS for those hosts found in the map. Vice versa, if you turned TLS on in `main.cf` and the host cannot be found in the policy map, it will still use TLS.



# Protecting Postfix client SMTP AUTH

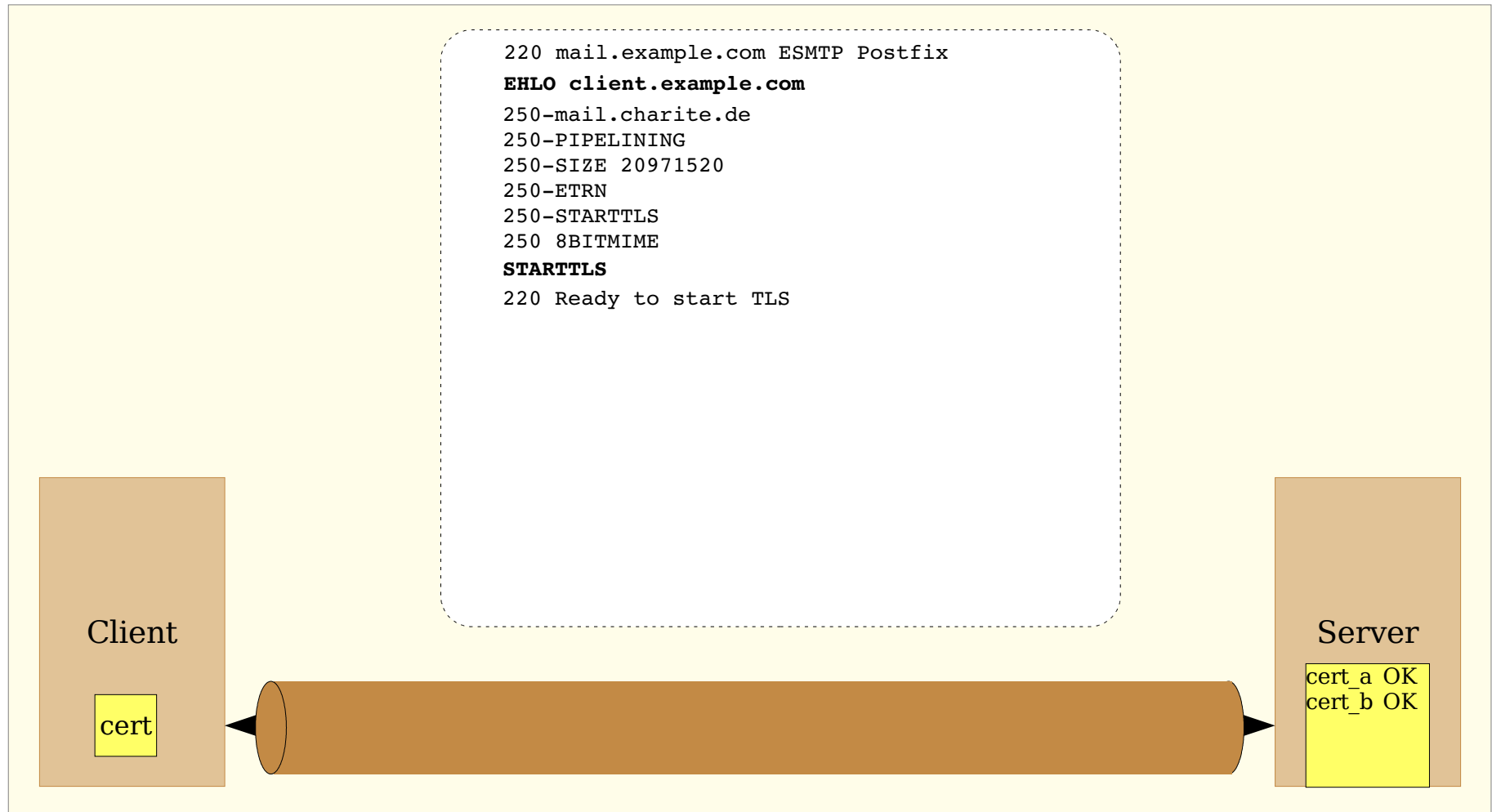
Most SMTP servers offer plaintext mechanisms. Postfix smtp client will submit username and password encoded, but unencrypted if plaintext mechanisms are being used.

Postfix smtp client can refuse to use plaintext mechanisms if TLS is not used:

```
smtp_sasl_security_options = noanonymous, noplaintext  
smtp_sasl_tls_security_options = noanonymous
```



# Certificate Based Relaying





# Server-side Access Control

Postfix has three restrictions to control certificate based relaying:

- **permit\_tls\_clientcerts**  
Allow relaying if client certificate passes verification.
- **permit\_tls\_all\_clientcerts**  
Allow relaying if client certificate stems from a trusted Certification Authority. Use this with caution!
- **check\_ccert\_access type:table**  
Select an access(5) policy for each client.



# Access Control with `permit_tls_clientcerts`

Postfix smtpd server must ask for client certificates because the default is not to tell:

```
smtpd_tls_ask_ccert = yes
```

Create a map to hold the MD5 Fingerprints.

```
00:8B:02:30:9D:18:F4:81:5D:2F:48:E4:5B:17:82:A7    client_1
18:F4:81:5D:2F:82:A7:48:E4:5B:17:00:8B:02:30:9D    client_2
...
```

Configure Postfix to use the map.

```
relay_clientcerts = hash:/etc/postfix/relay_clientcerts
```

Permit relaying for TLS clients in the map.

```
smtpd_recipient_restrictions =
    ...
    permit_tls_clientcerts
    ...
```



## NEW: Configuring the server side with check\_ccert\_access

Postfix smtpd server must ask for client certificates because the default is not to tell:

```
smtpd_tls_ask_ccert = yes
```

Create a map to hold the MD5 Fingerprints.

```
00:8B:02:30:9D:18:F4:81:5D:2F:48:E4:5B:17:82:A7 OK
18:F4:81:5D:2F:82:A7:48:E4:5B:17:00:8B:02:30:9D some_restriction
...
```

Permit relaying for TLS clients in the map.

```
smtpd_recipient_restrictions =
    ...
    check_ccert_access hash:/etc/postfix/client_cert_access
    ...
```



## Configuring the client-side

- The client must send a certificate.
- The key must not require a password.
- Set restrictive permissions for the key.
- Postfix smtp client must know where to find key and certificate.

```
smtp_tls_cert_file = /etc/postfix/certs/cert.pem  
smtp_tls_key_file = /etc/postfix/certs/key.pem
```



# Resources

- Postfix Website  
<http://www.postfix.org/>
- OpenSSL  
<http://www.openssl.org/>
- The Book of Postfix  
<http://www.postfix-book.com/>



about:speaker

Patrick Ben Koetter

[patrick.koetter@state-of-mind.de](mailto:patrick.koetter@state-of-mind.de)

<http://postfix.state-of-mind.de>