**Hà Nội, 7/2024**

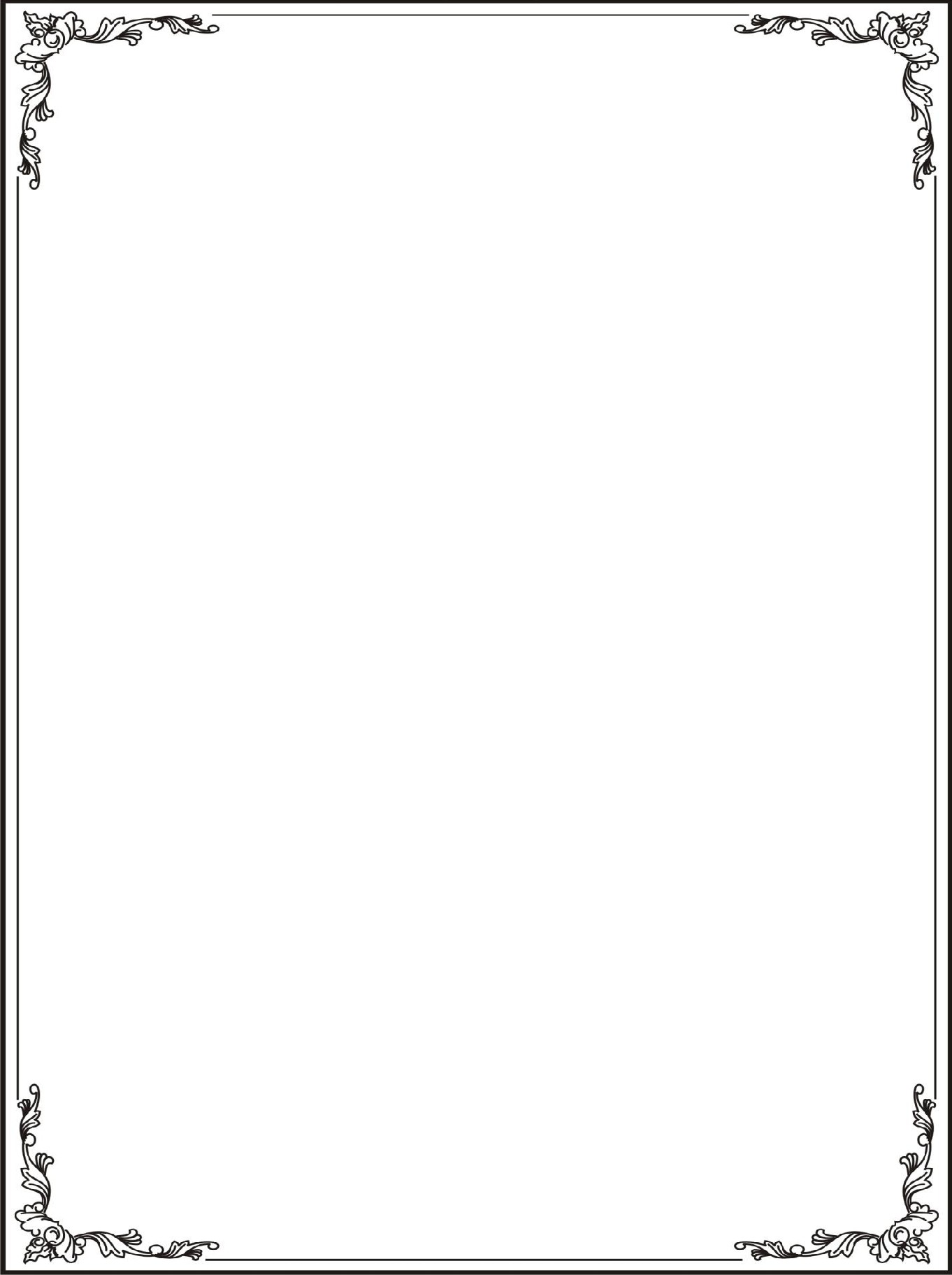
|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | PGS.TS. Đỗ Xuân Chợ |
| **Môn học:** | Mật mã học nâng cao |
| **Thành viên:** |  |
| Nguyễn Quang Tú | B19DCAT163 |
| Trần Việt Long | B19DCAT119 |
| Nguyễn Xuân Hoàng | B19DCAT079 |

**BÁO CÁO BÀI TẬP LỚN**

**ỨNG DỤNG SSL/TLS TRONG TRUYỀN TẢI DỮ LIỆU**

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

**KHOA AN TOÀN THÔNG TIN**



**MỤC LỤC**

[I. Tìm hiểu giao thức SSL/TLS 3](#_Toc173226440)

[1. Khái niệm 3](#_Toc173226441)

[2. Cấu trúc giao thức 3](#_Toc173226442)

[3. Các loại mã hóa 4](#_Toc173226443)

[3.1. Mật mã bất đối xứng 4](#_Toc173226444)

[3.2. Mật mã đối xứng 5](#_Toc173226445)

[4. SSL Handshake 5](#_Toc173226446)

[5. SSL Record 6](#_Toc173226447)

[II. Demo 8](#_Toc173226448)

[1. Bảo mật FTP sử dụng FTPS (SSL/TLS) 8](#_Toc173226449)

[1.1. Sử dụng SSL/TLS 8](#_Toc173226450)

[1.2. Không sử dụng SSL/TLS 10](#_Toc173226451)

[2. Ứng dụng SSL/TLS trong HTTP 12](#_Toc173226452)

**DANH MỤC BẢNG BIỂU**

[Hình 1: Cấu trúc và giao thức SSL/TLS 3](#_Toc173226417)

[Hình 2: Mật mã bất đối xứng 5](#_Toc173226418)

[Hình 3: Mật mã đối xứng 5](#_Toc173226419)

[Hình 4: Quá trình SSL Handshake 6](#_Toc173226420)

[Hình 5: Các bước trong giao thức SSL Record 7](#_Toc173226421)

[Hình 6: File ftps.java 8](#_Toc173226422)

[Hình 7: Tạo keystore thành công 8](#_Toc173226423)

[Hình 8: Kết nối tới host 9](#_Toc173226424)

[Hình 9: Wireshark bắt gói tin 9](#_Toc173226425)

[Hình 10: Dữ liệu server gửi SSL/TLS cho client 9](#_Toc173226426)

[Hình 11: Dữ liệu đã được mã hóa 10](#_Toc173226427)

[Hình 12: File client đã tải xuống 10](#_Toc173226428)

[Hình 13: Phần màu xanh là server gửi file cho client 10](#_Toc173226429)

[Hình 14: File ftps.java (Bỏ phần SSL/TLS) 11](#_Toc173226430)

[Hình 15: Sử dụng Wireshark để bắt gói tin 11](#_Toc173226431)

[Hình 16: Dữ liệu đã không được mã hóa 12](#_Toc173226432)

[Hình 17: Client tải file và không được mã hóa 12](#_Toc173226433)

[Hình 18: Giao diện đăng nhập 13](#_Toc173226434)

[Hình 19: Đăng nhập thành công 13](#_Toc173226435)

[Hình 20: Dùng wireshark bắt gói tin chứa thông tin đăng nhập 14](#_Toc173226436)

[Hình 21: Dùng OpenSSL tạo khóa và chứng chỉ CA 15](#_Toc173226437)

[Hình 22: Thông tin chứng chỉ website 15](#_Toc173226438)

[Hình 23: Wireshark không còn bắt gói tin chứa bản rõ 16](#_Toc173226439)

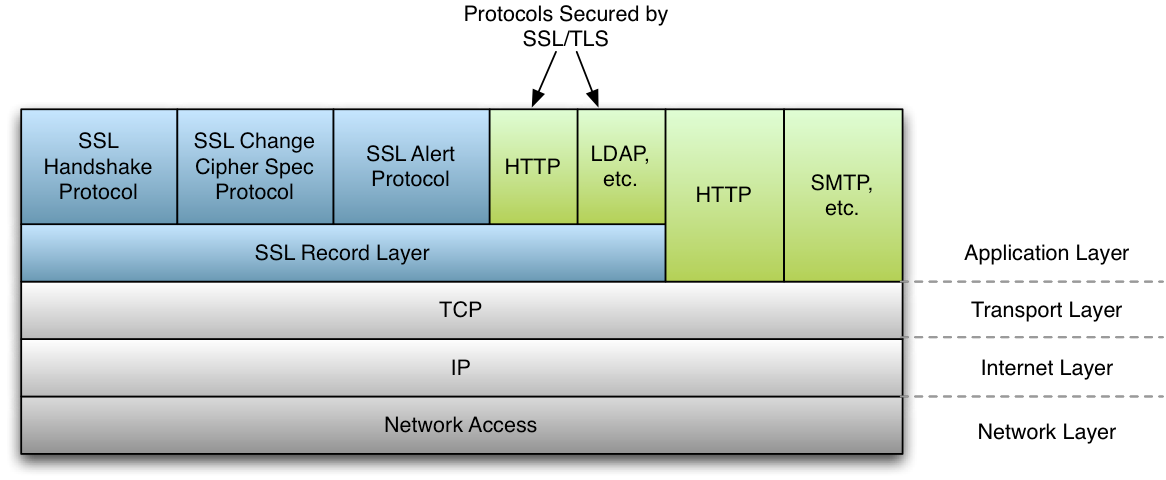
# I. Tìm hiểu giao thức SSL/TLS

## 1. Khái niệm

* SSL (*Secure Sockets Layer*) và TLS (*Transport Layer Security*) là giao thức bảo mật được sử dụng để tạo kết nối an toàn và mã hóa dữ liệu truyền đi trên Internet. Nhưng do SSL không còn được phát triển nữa và đang được thay thế hoàn toàn bởi TLS.
* SSL/TLS đã kết hợp những yếu tố sau để thiết lập một giao dịch an toàn:
* **Xác thực:** Đảm bảo tính xác thực của trang mà bạn sẽ làm việc ở đầu kia của kết nối. Các trang web cũng cần phải kiểm tra tính xác thực của người dùng.
* **Mã hóa:** Đảm bảo thông tin không thể bị truy cập bởi đối tượng thứ ba. Để loại trừ việc nghe trộm những thông tin “nhạy cảm” khi được lan truyền qua Internet, dữ liệu phải được mã hóa để không thể bị đọc được bởi những người khác.
* **Toàn vẹn dữ liệu:** Đảm bảo thông tin không bị sai lệch và nó phải thể hiện chính xác thông tin gốc gửi đến.

## 2. Cấu trúc giao thức

* Cấu trúc của giao thức SSL được minh họa trong hình 1



Hình 1: Cấu trúc và giao thức SSL/TLS

* SSL được xếp lớp lên trên một dịch vụ vận chuyển định hướng nối kết và đáng tin cậy, chẳng hạn như được cung cấp bởi TCP. Nó có thể cung cấp các dịch vụ bảo mật cho các giao thức ứng dụng tùy ý dựa vào TCP chứ không chỉ HTTP.
* Một ưu điểm chính của giao thức bảo mật lớp vân chuyển (Transport Layer) nói chung và giao thức SSL nói riêng là chúng độc lập với ứng dụng theo nghĩa là chúng có thể được sử dụng để bảo vệ bất kỳ giao thức ứng dụng được xếp lớp lên trên TCP một cách trong suốt.
* Một số giao thức ứng dụng điển hình bao gồm HTTP, FTP, Telnet, IMAP, IRC và POP3.
* Tóm lại, giao thức SSL cung cấp sự bảo mật truyền thống vốn có ba đặc tính cơ bản:
  + Các bên giao tiếp (Client và Server) có thể xác thực nhau bằng cách sử dụng mật mã khóa chung.
  + Tính bí mật của lưu lượng dữ liệu đươc bảo vệ vì kết nối được mã hóa trong suốt sau khi một sự thiết lập quan hệ ban đầu và sự thương lượng khóa session đã xảy ra.
  + Tính xác thực và tính toàn vẹn của lưu lượng dữ liệu cũng được bảo vệ vì các thông báo được xác thực và kiểm tra tính toàn vẹn một cách trong suốt bằng cách sử dụng MAC.
* Dưới đây là bảng các cổng được gán cho giao thức ứng dụng chạy trên SSL/TLS

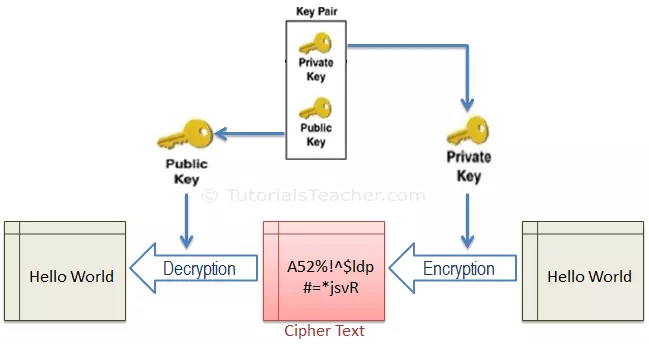
|  |  |
| --- | --- |
| **Từ khóa** | **Cổng** |
| HTTPS | 443 |
| SMTPS | 465 |
| FTPS | 990 |
| TENETS | 992 |
| POP3S | 995 |
| IMAPS | 995 |

## 3. Các loại mã hóa

* Về cơ bản, SSL hoạt động với 2 loại mật mã sau:
  + Mật mã bất đối xứng (Mã khóa công khai)
  + Mật mã đối xứng

### 3.1. Mật mã bất đối xứng

* Mật mã bất đối xứng sử dụng một cặp khóa liên quan đến toán học để mã hóa và giải mã dữ liệu. Trong một cặp khóa, một khóa được chia sẻ với bất kỳ ai quan tâm đến giao tiếp. Nó gọi là Public Key và khóa khác trong cặp khóa được giữ bí mật và được gọi là Private Key.
* Trong mật mã không đối xứng, dữ liệu có thể được ký bằng khóa riêng, chỉ có thể giải mã bằng khó công khai liên quan trong một cặp.

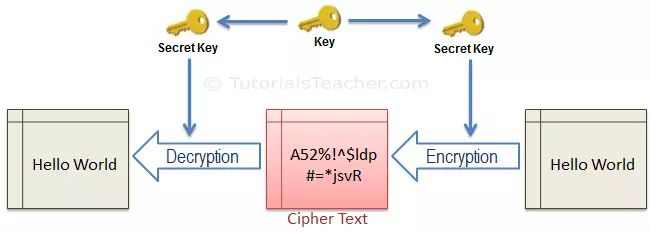


Hình 2: Mật mã bất đối xứng

* SSL sử dụng mật mã không đối xứng để bắt đầu giao tiếp gọi là SSL handshake. Các thuật toán mã hóa khóa bất đối xứng được sử dụng phổ biến nhất bao gồm RSA, DSA, đường cong Elliptic.

### 3.2. Mật mã đối xứng

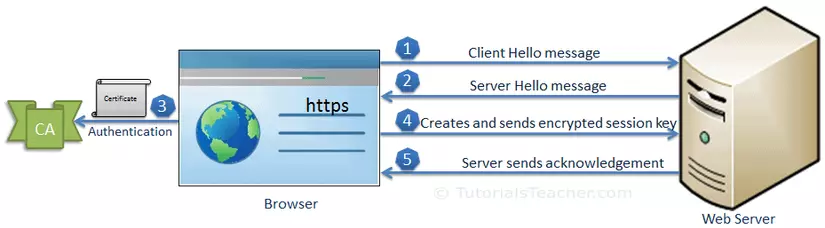
* Trong mật mã đối xứng, chỉ có một khóa mã hóa và giải mã dữ liệu. Cả người gửi và người nhận đều phải có khóa này mà chỉ họ mới biết.
* SSL sử dụng mật mã đối xứng bằng cách sử dụng khóa phiên sau khi quá trình bắt tay ban đầu được thực hiện. Các thuật toán đối xứng được sử dụng rộng rãi nhất là AES-128, AES-192 và AES-256.



Hình 3: Mật mã đối xứng

## 4. SSL Handshake

* SSL Handshake là một mật mã không đối xứng cho phép trình duyệt xác minh máy chủ web, lấy khóa công khai và thiết lập kết nối an toàn trước khi bắt đầu truyền dữ liệu thực tế.



Hình 4: Quá trình SSL Handshake

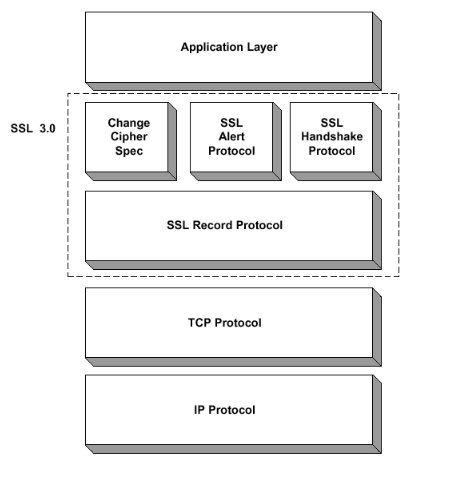
* Quá trình SSL Handshake được thể hiện như sau:

1. Máy khách gửi thông báo “Client Hello”. Điều này bao gồm số phiên bản SSL của máy khách, cài đặt mật mã, dữ liệu theo phiên cụ thể mà máy chủ cần giao tiếp với khách hàng qua bằng SSL.
2. Máy chủ phản hồi bằng thông báo “Server Hello”. Điều này bao gồm số phiên bản SSL của máy chủ, cài đặt mật mã, dữ liệu theo phiên cụ thể, chứng chỉ SSL có khóa công khai và thông tin khác mà máy khách cần để giao tiếp với máy chủ qua SSL.
3. Máy khách xác minh chứng chỉ SSL của máy chủ từ CA (Certificate Authority) và xác thực máy chủ. Nếu xác thực không thành công, thì máy khách từ chối kết nối SSL và ném một ngoại lệ. Nếu xác thực thành công, chuyển sang bước 4.
4. Máy khách tạo một session key, mã hóa nó bằng khóa công khai của máy chủ và gửi đến máy chủ. Nếu máy chủ đã yêu cầu xác thực máy khách thì máy khách sẽ gửi chứng chỉ của chính mình đến máy chủ.
5. Máy chủ giải mã khóa phiên bằng khóa riêng của nó và gửi xác nhận đến máy khách được mã hóa bằng khóa phiên.

* Do đó, khi kết thúc quá trình bắt tay SSL, cả máy khách và máy chủ đều có khóa phiên hợp lệ mà họ sẽ sử dụng để mã hóa hoặc giải mã dữ liệu thực tế. Khóa công khai và khóa cá nhân sẽ không được sử dụng nữa sau đó.

## 5. SSL Record

* Giao thức SSL Record nhận dữ liệu từ các giao thức con SSL lớp cao hơn và xử lý việc phân đoạn, nén, xác thực và mã hóa dữ liệu.



Hình 5: Các bước trong giao thức SSL Record

* Mỗi bản ghi SSL chứa các trường thông tin sau đây: Loại nôi dung, số phiên bản giao thức, tải trọng dữ liệu, MAC.
* Một số giao thức con SSL được xếp lớp trên SSL Record. Mỗi giao thức con có thể tham chiếu đến các loại thông báo cụ thể vốn được gửi bằng cách sử dụng giao thức SSL Record. Thông số kỹ thuật của SSL 3.0 xác định 3 giao thức SSL sau đây:
* **Alert Protocol:** Được dùng để truyền cảnh báo với liên kết bên kia như không thể thiết lập các thông số bảo mật được đưa ra từ lựa chọn có sẵn, certificate nhận được không hợp lệ, hoặc certificate đã hết hạn đăng ký…
* **Handshake Protocol:** Giao thức giúp client và server trao đổi các thông tin để thiết lập kết nối SSL
* **Change CipherSpec Protocol:** Sinh ra trạng thái tiếp theo để gắn vào trạng thái hiện tại, và ở trạng thái hiện tại cập nhật lại bộ mã hóa để sử dụng trên kết nối này

# II. Demo

## 1. Bảo mật FTP sử dụng FTPS (SSL/TLS)

### 1.1. Sử dụng SSL/TLS

* Phần demo sẽ sử dụng SSL/TLS với file code ftps.java

A screen shot of a computer program

Description automatically generated

Hình 6: File ftps.java

* Code sử dụng thư viện: org.apache.ftpserver:ftpserver-core:1.2.0
* Tạo keystore để sử dụng SSL/TLS: Dùng keytool của jdk

*keytool -genkey -alias ftps -keyalg RSA -keysize 2048 -keystore keystore.jks -dname "CN=Nguyen Xuan Hoang, OU=PTIT, O=DEMO, L=Nguyen trai, S=Hanoi, C=VN" -storepass password -keypass password*

A black background with white text

Description automatically generated

Hình 7: Tạo keystore thành công

A screenshot of a computer

Description automatically generated

Hình 8: Kết nối tới host

A screenshot of a computer

Description automatically generated

Hình 9: Wireshark bắt gói tin

A screenshot of a computer program

Description automatically generated

Hình 10: Dữ liệu server gửi SSL/TLS cho client

* Ở bên phía client, dữ liệu sẽ được nhận từ bên server và nó đã được mã hóa

A screen shot of a computer

Description automatically generated

Hình 11: Dữ liệu đã được mã hóa

A screenshot of a computer code

Description automatically generated

Hình 12: File client đã tải xuống

A screenshot of a computer

Description automatically generated

Hình 13: Phần màu xanh là server gửi file cho client

### 1.2. Không sử dụng SSL/TLS

* Vẫn sử dụng file ftps.java nhưng lược bỏ phần thiết lập SSL/TLS

A screen shot of a computer program

Description automatically generated

Hình 14: File ftps.java (Bỏ phần SSL/TLS)

A screenshot of a computer

Description automatically generated

Hình 15: Sử dụng Wireshark để bắt gói tin

A screenshot of a computer program

Description automatically generated

Hình 16: Dữ liệu đã không được mã hóa

A screenshot of a computer

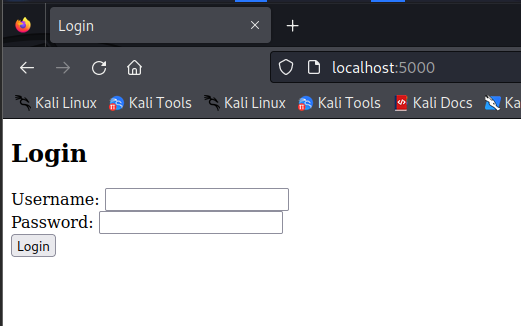
Description automatically generated

Hình 17: Client tải file và không được mã hóa

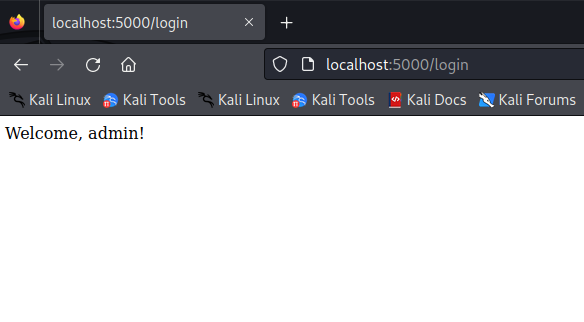
* Có thể tải Demo qua Github: <https://github.com/hoangsvn/DemoMMNC>

## 2. Ứng dụng SSL/TLS trong HTTP

* Tiến hành đăng nhập vào website chưa có có chứng chỉ CA

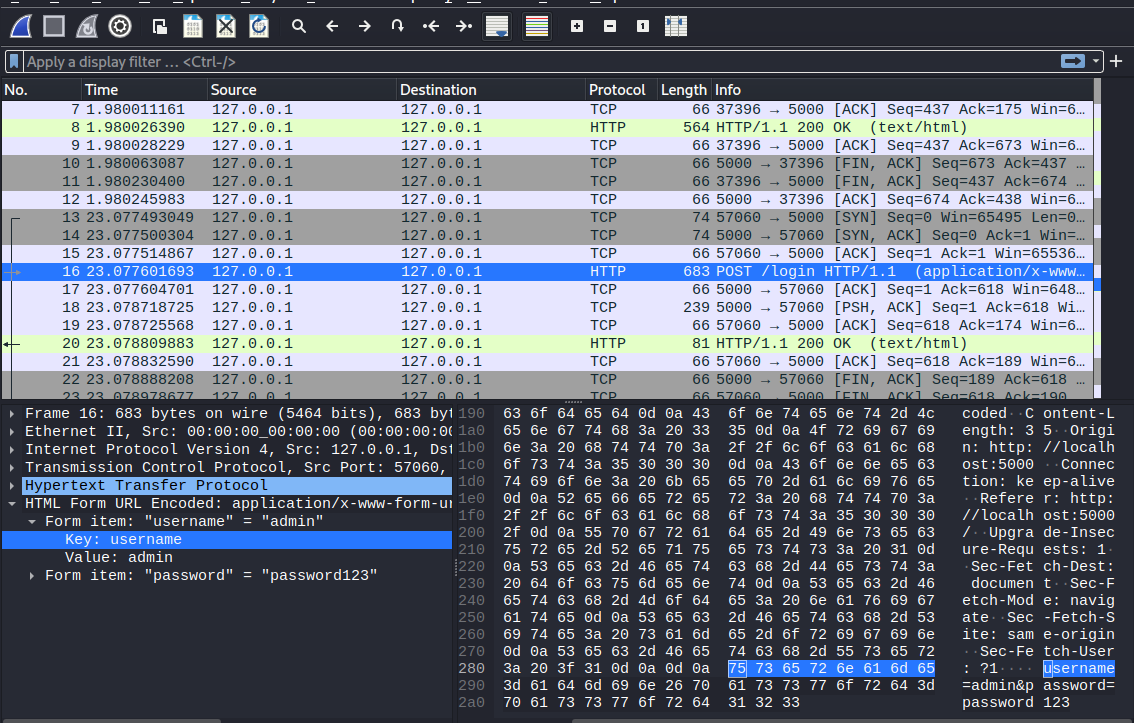


Hình 18: Giao diện đăng nhập



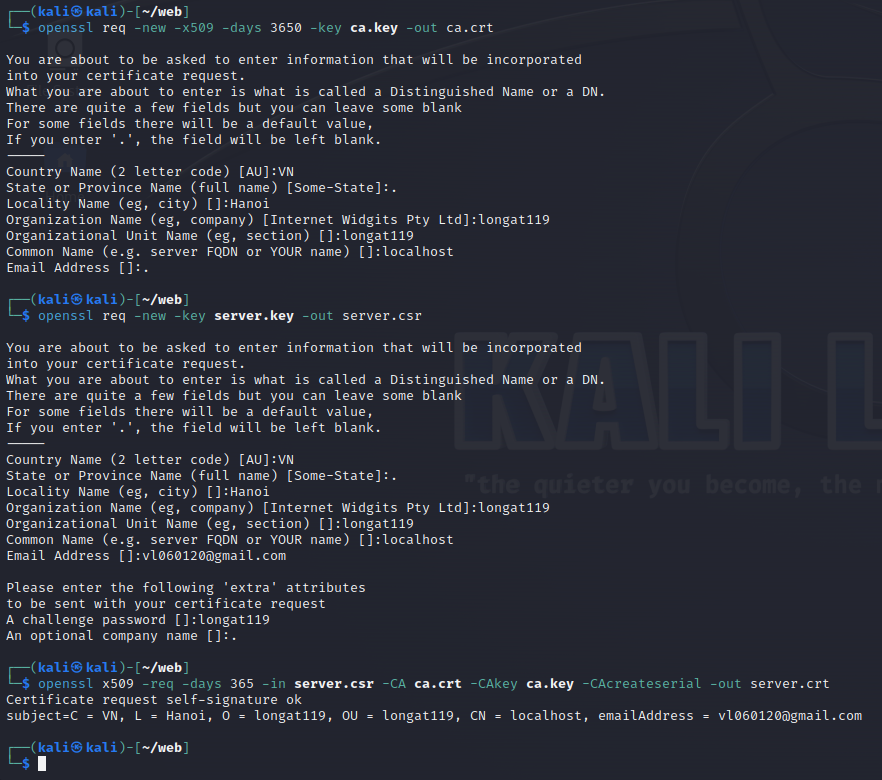
Hình 19: Đăng nhập thành công

* Cùng lúc đó mở Wireshark và bắt gói tin



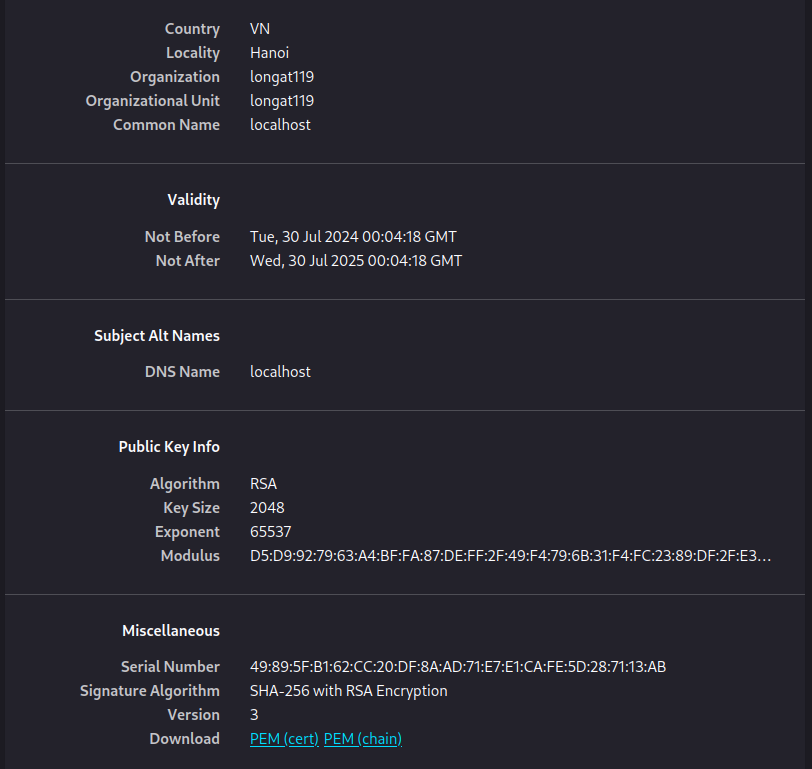
Hình 20: Dùng wireshark bắt gói tin chứa thông tin đăng nhập

* Có thể thấy rằng gói tin truyền đi mà không được mã hóa khiến thông tin đăng nhập được truyền dưới dạng bản rõ.
* Tiến hành tạo khóa riêng và chứng chỉ CA cho website bằng OpenSSL



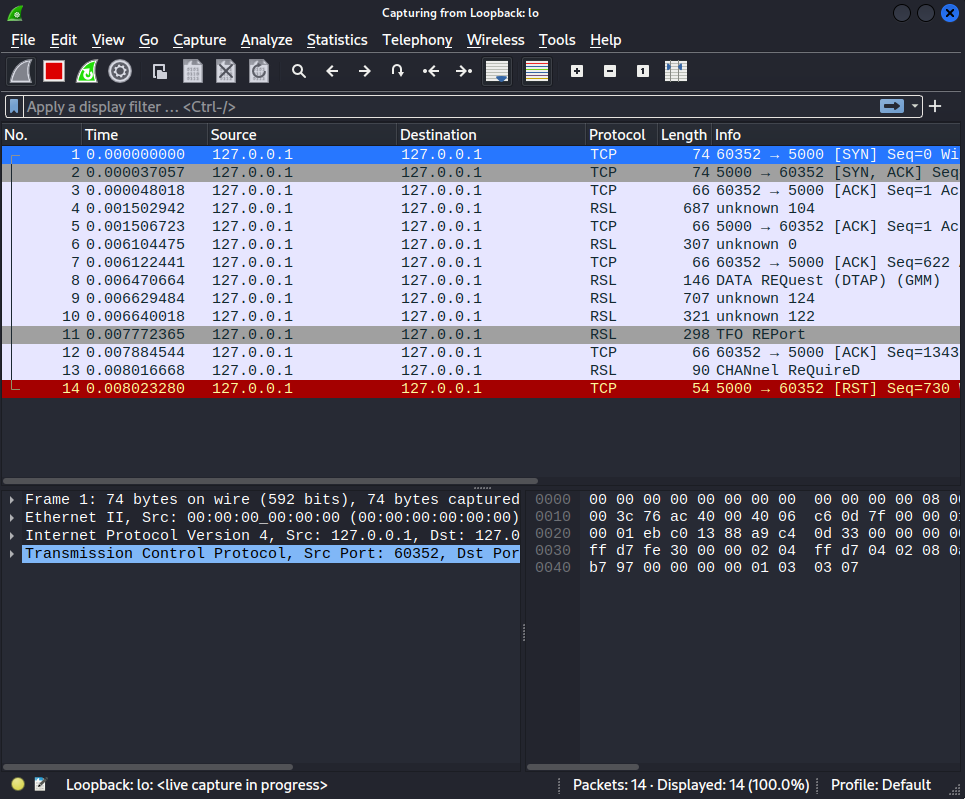
Hình 21: Dùng OpenSSL tạo khóa và chứng chỉ CA

* Sau khi tạo kháo và chứng chỉ, cập nhật chứng chỉ cho website.



Hình 22: Thông tin chứng chỉ website

* Tiến hành đăng nhập lại vào website và dùng wireshark bắt gói tin



Hình 23: Wireshark không còn bắt gói tin chứa bản rõ

**TÀI LIỆU THAM KHẢO**

[1]. <https://luanvan.net.vn/luan-van/de-tai-tim-hieu-giao-thuc-ssl-tls-cach-tan-cong-va-phong-chong-32911/>

[2]. Đỗ Xuân Chợ, Bài giảng Mật mã học nâng cao, Học viện Công nghệ Bưu chính Viễn thông, 2017