**1. Giới thiệu chung về bài thực hành**

          Certificate Authority (CA) là một tổ chức chịu trách nhiệm cấp chứng chỉ số để xác minh danh tính trên internet. Trong khi các CA công cộng thường được sử dụng để xác thực danh tính của các trang web và dịch vụ cho công chúng, thì các CA riêng tư thường được dùng trong các nhóm kín và các dịch vụ nội bộ.

          Việc xây dựng một Certificate Authority (CA) riêng cho phép sinh viên cấu hình, kiểm tra và vận hành các chương trình yêu cầu kết nối được mã hóa giữa máy khách và máy chủ. Với một CA riêng, sinh viên có thể cấp chứng chỉ cho người dùng, máy chủ, hoặc các chương trình và dịch vụ cụ thể trong hệ thống của mình.

Một số ví dụ về các chương trình Linux sử dụng CA riêng bao gồm OpenVPN và Puppet. Sinh viên cũng có thể cấu hình máy chủ web sử dụng chứng chỉ do CA riêng cấp để phát triển hoặc kiểm thử, đảm bảo các môi trường giống như máy chủ sản xuất sử dụng TLS để mã hóa kết nối.

          Trong bài thực hành cấu hình CA này sinh viên sẽ được tự tay cấu hình CA riêng của riêng mình được sử dụng trong nội bộ. Sinh viên tự thực hiện cấu hình trên ubuntu, thực hiện tạo yêu cầu chứng chỉ. Qua bài thực hành sinh viên sẽ hiểu cách mà một CA server hoạt động, hiểu được cách tạo 1 yêu cầu và cách CA xác nhận yêu cầu đó. Điều đó cho sinh viên cái nhìn tốt hơn về việc CA server hoạt động và phát triển tốt hơn trong lĩnh vực an toàn thông tin

**2. Nội dung và hướng dẫn bài thực hành**

***- Mục đích***

          + Chúng ta sẽ tìm hiểu cách thiết lập CA server trên máy chủ Ubuntu cũng như cách tạo và ký chứng chỉ kiểm tra bằng CA mới

          + Chúng ta cũng sẽ tìm hiểu cách nhập chứng chỉ công khai của máy chủ CA vào kho chứng chỉ của hệ điều hành để có thể xác minh chuỗi tin cậy giữa CA và máy chủ hoặc người dùng từ xa.

***- Yêu cầu đối với sinh viên***

          + Biết sử dụng câu lệnh trên Linux

          + Biết sử dụng một số công cụ trên linux

**3. Nội dung thực hành**

          + Sinh viên tải bài lab bằng lệnh imodule:

[***https://github.com/ParadoxParadise/labattt-2025/raw/refs/heads/main/cryp-app-ca\_server/imodule.tar***](https://github.com/ParadoxParadise/labattt-2025/raw/refs/heads/main/cryp-app-ca_server/imodule.tar)

*(Chú ý: sinh viên sử dụng MÃ SINH VIÊN của mình để nhập thông tin người thực hiện bài lab khi có yêu cầu, để sử dụng khi chấm điểm)*

          Nội dung bài thực hành gồm 2 phần, phần 1 là cài đặt môi trường CA server (Certificate Authority Server) Trong bài này sẽ gọi là CA server. Khi bài lab khởi động thì có 2 container hiện lên là caserver và client

**PHẦN I: CẤU HÌNH VÀ CÀI ĐẶT**

***- TASK 1:***

          + Khởi động bài lab và nhập mã sinh viên của người làm bằng lệnh:

***labtainer - r cryp-app-ca\_server***

          + Trên caserver đã cài đặt đầy đủ các công cụ cần thực hiện như: ssh, openssl, easy-rsa

          + Chuẩn bị thư mục chứa PKI (Public key infrastructure) trên caserver bằng lệnh: ***mkdir easy-rsa***

          + Liên kết thư mục mới tạo easy-rsa với easy-rsa (đã cài đặt sẵn trong container)

***ln -s /usr/share/easy-rsa/\* easy-rsa/***

          + Hạn chế quyền truy cập vào thư mục PKI (easy-rsa), thực hiện cấp lại quyền với chmod

***chmod 700 /home/ubuntu/easy-rsa***

          + Khởi tạo PKI bên trong thư mục đã tạo:

***cd easy-rsa***

***./easyrsa init-pki***

          + Kiểm tra output trả về có dạng:

*Output:*

*init-pki complete; you may now create a CA or requests.*

*Your newly created PKI dir is: /home/ubuntu/easy-rsa/pki*

→  Sau khi hoàn thành quá trình trên một thư mục chứa tất cả các tệp cần thết để tạo CA

***- TASK 2:***

          + Sau khi tạo xong PKI ở TASK 1, ở TASK 2 này chúng ta sẽ tạo Certificate Authority (CA)

          + Di chuyển vào thư mục easy-rsa và thực hiện tạo file “vars”

***cd easy-rsa***

***nano vars (thực hiện copy nội dung dưới đây vào file vars)***

        ```

set\_var EASYRSA\_REQ\_COUNTRY    "NV"

set\_var EASYRSA\_REQ\_PROVINCE   "Hanoi"

set\_var EASYRSA\_REQ\_CITY       "Hanoi City"

set\_var EASYRSA\_REQ\_ORG        "none"

set\_var EASYRSA\_REQ\_EMAIL      "admin@example.com"

set\_var EASYRSA\_REQ\_OU         "Community"

set\_var EASYRSA\_ALGO           "ec"

set\_var EASYRSA\_DIGEST         "sha512"

      ```

*(Chú ý: Ctrl + X, sau đó Y và Enter để lưu lại file)*

          + Sau khi lưu File, bây giờ chúng ta sẽ xây dựng CA

          + Để tạo khóa công khai (public key) và khóa riêng tự (private key) gốc cho CA, thực hiện chạy lệnh

***./easyrsa build-ca***

***(thực hiện lệnh này trong thư mục ban đầu easy-rsa)***

*Output đầu ra:*

          + Bạn sẽ thấy 1 dòng về version của openssl

          + Bạn sẽ phải cập nhật cụm mật khẩu (passphrase) cho cặp khóa của mình (tron thực tế mật khẩu đảm bảo phải mạnh và an toàn, tuy nhiên trong quá trình thử nghiệm đặt mật khẩu dễ để bạn có thể nhớ được, VD: 123, abc, …)

          + Bạn sẽ phải nhập Common Name (CN) cho CA (Là tên của máy này trong ngữ cảnh của CA, có thể nhập chuỗi ký tự bất kỳ để bỏ quả hoặc nhấn Enter để chấp nhận mặc định)

*Output:*

*. . .*

*Enter New CA Key Passphrase:*

*Re-Enter New CA Key Passphrase:*

*. . .*

*Common Name (eg: your user, host, or server name) [Easy-RSA CA]:*

*CA creation complete and you may now import and sign cert requests.*

*Your new CA certificate file for publishing is at:*

*/home/sammy/easy-rsa/pki/ca.crt*

          - Sau khi hoàn thành quá trình này chúng ta có 2 tệp quan trọng là

*+ easy-rsa/pki/ca.crt*: Tệp này là chứng chỉ công khải của CA, Người dùng, máy khách, máy chủ dùng tệp này để xác minh rằng họ là một phần của cùng một trang web tin cậy. Tránh giả mạo Man-in-the-middle attack

*+ easy-rsa/pki/private/ca.key*: Là khóa riêng của CA sử dụng để ký chứng chỉ cho máy chủ và máy khách. Nếu kẻ tấn công có quyền truy cập CA của bạn và có tệp ca.key, bạn cần phải hủy CA của mình. Lý tưởng nhất là tệp ca.key chỉ có trên máy chủ CA và máy chủ này được giữ ngoại tuyến khi không ký các yêu cầu chứng chỉ như một biện pháp bảo mật mạnh

→ Sau khi thành công tạo CA, lúc này CA đã sẵn sàng được sử dụng để ký các yêu cầu chứng chỉ cũng như thu hồi, trong bài lab này chúng ta chỉ đề cập quá trình ký.

***- TASK 3:***

          + Sau khi hoàn thành TASK 2, CA giờ đây đã sẵn sàng hoạt động như một nguồn đáng tin cậy gốc cho bất kỳ hệ thống nào mà chúng ta muốn cấu hình. Bất kỳ người dùng hay máy chủ nào muốn xác minh chúng ta cần phải copy file “ca.crt” vào nơi lưu chứng chỉ ở hệ điều hành của người đó

          + Có nhiều cách để dùng chứng chỉ CA này, trong bài lab này sử dụng hệ thống Linux (Phân phối ubuntu hoặc debian) là máy chủ thứ 2 (Được gọi là máy khách)

          + Để nhập chứng chỉ công khai trên máy chủ caserver chúng ta thực hiện

***cat ~/easy-rsa/pki/ca.crt***

*Output:*

*-----BEGIN CERTIFICATE-----*

*MIIDSzCCAjOgAwIBAgIUcR9Crsv3FBEujrPZnZnU4nSb5TMwDQYJKoZIhvcNAQEL*

*BQAwFjEUMBIGA1UEAwwLRWFzeS1SU0EgQ0EwHhcNMjAwMzE4MDMxNjI2WhcNMzAw*

*. . .*

*. . .*

*-----END CERTIFICATE-----*

*(Chú ý: Copy toàn bộ thông tin trả về cả —BEGIN— và —END—)*

          + Trên máy khách thực hiện ghi vào file ca.crt bằng lệnh nano

***nano /tmp/ca.crt***

          + Copy thông tin đã copy trước đó vào file này, Ctrl +X, Y và Enter để hoàn thành

          + Bây giờ chúng ta đã có bản sao của File ca.crt trên hệ thống máy khách, bây giờ là lúc nhập chứng chỉ vào kho lưu chứng chỉ của hệ điều hành (Hệ điều hành ở đây là linux)

***sudo cp /tmp/ca.crt /usr/local/share/ca-certificates/***

***sudo update-ca-certificates***

→ Bây giờ máy khách sẽ tin cậy bất kỳ chứng chỉ nào được máy chủ CA ký

**PHẦN II: TẠO YÊU CẦU KÝ CHỨNG CHỈ**

***- TASK 4:***

          + Khi đã hoàn thành phần 1 thì lúc này chúng ta có một CA hoạt động và được cấu hình đầy đủ

          + Ở phần II này chúng ta sẽ đề cập tới tạo và ký yêu cầu chứng chỉ

          + Certificate Signing Request (CSR) được gọi với tên là yêu cầu ký chứng chỉ gồm 3 phần chính.

* + Khóa chung (Public key)
  + Thông tin nhận dạng về hệ thống yêu cầu (Identifying information about the requesting system)
  + Chữ ký chính của yêu cầu (Signature of the request itself)

          + Private key sẽ được giữ bí mật và sẽ được sử dụng để mã hóa thông tin mà bất kỳ ai có Public Certificate đã ký đều có thể giải mã

          + Thực hiện chạy trên máy khách, sử dụng openssl

          + Đầu tiên, chúng ta cần hoàn thành để tạo CSR là tạo Private Key. để tạo private key, ta tạo thư mục practice-csr rồi tạo khóa bên trong. Đặt tên cho máy chủ

***mkdir practice-csr***

***cd practice-csr***

***openssl genrsa -out catest-server.key***

*Output:*

*Generating RSA private key, 2048 bit long modulus (2 primes)*

*. . .*

*. . .*

*e is 65537 (0x010001)*

***- TASK 5:***

          + Bây giờ chúng ta đã có Private key, chúng ta tiếp tục tạo CSR tương ứng dùng openssl. Một số thông tin được đề cập bao gồm Country, State, and City. Có thể để trống 1 trường nhưng lưu ý là CA thực sự tốt nhất nếu có các giá trị chính xác cho vị trí Organization

***openssl req -new -key catest-server.key -out catest-server.req***

*Output:*

. . .

-----

Country Name (2 letter code) [XX]:VN

State or Province Name (full name) []:Ha Noi

Locality Name (eg, city) [Default City]:Ha Noi City

Organization Name (eg, company) [Default Company Ltd]:none

Organizational Unit Name (eg, section) []:none

Common Name (eg, your name or your server's hostname) []:catest-server

Email Address []:

Please enter the following 'extra' attributes

to be sent with your certificate request

A challenge password []:

An optional company name []:

          + Xác minh nội dung CSR đã tạo, có thể xem lại với lệnh

***openssl req -in catest-server.req -noout -subject***

          + Sau khi đã hài lòng với thông tin đã tạo, copy File catest-server.req vào máy chủ CA bằng lệnh scp, Mật khẩu của container lưu trữ CA server: ubuntu

***scp catest-server.req*** [***ubuntu@10.0.0.199:/tmp/catest-server.req***](mailto:ubuntu@10.0.0.199:/tmp/catest-server.req)

***(MK là ubuntu)***

          + Trong bước này ta đã tạo CSR cho máy chủ có tên là catest-server. Tuy nhiên, trong thực tế yêu cầu đến từ thứ gì đó như máy chủ web chạy thử hoặc phát triển cần TLS, hoặc máy chủ OpenVPN

          + Trong Task tiếp theo chúng ta sẽ tiến hành ký Yêu cầu chứng chỉ bằng Private key

***- TASK 6:***

          + Trong bước này chúng ta sẽ thực hiện ký CSR

          + Trong thư mục PKI ở caserver sử dụng lệnh

***cd easy-rsa***

***./easyrsa import-req /tmp/catest-server.req catest-server***

*Output:*

. . .

The request has been successfully imported with a short name of: catest-server

You may now use this name to perform signing operations on this request.

          + Bây giờ có thể ký yêu cầu bằng cách chạy lệnh

***./easyrsa sign-req server catest-server***

*(vì đang thực hiện với máy chủ ảo nên chọn option là server)*

          + Ở đầu ra, chúng ta sẽ được xác nhận rằng yêu cầu đến từ một nguồn đáng tin cậy. Nhấn **‘YES’**sau đó Enter để xác nhận điều này

*You are about to sign the following certificate.*

*Please check over the details shown below for accuracy. Note that this request*

*has not been cryptographically verified. Please be sure it came from a trusted*

*source or that you have verified the request checksum with the sender.*

*Request subject, to be signed as a server certificate for 3650 days:*

*subject=*

*commonName                = catest-server*

*Type the word 'yes' to continue, or any other input to abort.*

*Confirm request details: yes*

*. . .*

*Certificate created at: /home/ubuntu/easy-rsa/pki/issued/catest-server.crt*

          + Ta sẽ nhập mật khẩu vào thời điểm này, mật khẩu đã được tạo ở TASK 2

          + Khi hoàn thành, chúng ta đã ký CSR catest-server.req bằng private key của CA server trong /home/ubuntu/easy-rsa/pki/private/ca.key. File catest-server.crt thu được chứa public encryption key của máy khách cũng như chữ ký mới của CA server. Mục đích của chữ ký là thông báo cho bất kỳ ai tin cậy CA rằng họ cũng có thể tin cậy chứng chỉ của catest-server

          + Gửi lại nó với máy khách

***scp ubuntu@10.0.0.199:/home/ubuntu/easy-rsa/pki/issued/catest-server.crt /tmp***

***scp ubuntu@10.0.0.199:/home/ubuntu/easy-rsa/pki/ca.crt /tmp***

***Chú ý:***

          - Kết thúc lab: Trên terminal khởi động lab, sinh viên sử dụng lệnh:

***stoplab***

          - Khi bài lab kết thúc, một tệp lưu kết quả được tạo và lưu vào một vị trí được hiển thị bên dưới stoplab. Sinh viên cần nộp file .lab để chấm điểm.

          - Để kiểm tra kết quả khi trong khi làm bài thực hành sử dụng lệnh:

***checkwork <tên bài thực hành>***

          - Sinh viên cần nộp file .lab để chấm điểm.

          - Kiểm tra kết quả trong quá trình làm bài:

***checkwork <tên bài lab>***

          - Khởi động lại bài lab: Trong quá trình làm bài sinh viên cần thực hiện lại bài lab, dùng câu lệnh:

***labtainer -r cryp-app-ca\_server***