|  |
| --- |
| BAN CƠ YẾU CHÍNH PHỦ  **HỌC VIỆN KỸ THUẬT MẬT MÃ**  ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯ |
| **ĐỒ ÁN TỐT NGHIỆP**  **NGHIÊN CỨU GIẢI PHÁP XÁC THỰC VÀ BẢO MẬT CHO API SERVICE TRONG PHÁT TRIỂN PHẦN MỀM** |
| Ngành: An toàn thông tin  Mã số: 7.48.02.02  *Sinh viên thực hiện*:  **Đinh Xuân Dương**  Lớp: AT13i  *Giảng viên hướng dẫn* :  **ThS. Nguyễn Thị Hồng Hà**  Khoa An toàn thông tin – Học viện Kỹ thuật mật mã |
| Hà Nội, 2021 |

# Mở đầu

## Tính cấp thiết của việc xác thực và bảo mật cho API service

API cung cấp khả năng cung cấp khả năng truy xuất đến một tập các hàm hay dùng. Và từ đó có thể trao đổi dữ liệu giữa các ứng dụng. Nói một cách khác API là công cụ kết nối giữa các phần mềm.

Chính vì sự linh hoạt cũng như hiệu quả của API nên hiện nay hệ thống lớn đang sử dụng API để trao đổi thông tin giữa các phần mềm. Nên yêu đầu xác thực người dung cũng như bảo mật cho các API Service đang thực sự là cấp thiết.

## Mục tiêu thực hiện đồ án

* Hiểu được tính cấp thiết của API service
* Nắm được khái niệm, cơ chế hoạt động của một API service.
* Triền khai được một API service có đủ các tính chất của an toàn thông tin như:

+ Tính toàn vẹn

+ Tính bảo mật

+ Tính sẵn sang

+ Tính xác thực

+ Chống chối bỏ

- Khắc phục được một số nhược điểm của một số API service đang gặp phải

# Dự kiến các Chương, mục

1. Giới thiệu về API service
   1. Khái niệm về API service ? Tại sao API lại phổ biến ?

a. Khái niệm về API service:

- API là các phương thức, giao thức kết nối giữa các ứng dụng khách nhau. Nó là viết tắt của **Application Programming Interface (Giao diện lập trình ứng dụng).**

**- API cung cấp khả năng truy xuất hoặc thay đổi dữ liêu, tạo ra tương tác giữa các ứng dụng hiệu quả và tiện lợi hơn và cho phép các hệ thống có thể tận dụng năng lực, chức năng của nhau.**

**-** Một API có thể hoàn toàn tùy chỉnh, dành riêng cho một thành phần hoặc được thiết kế dựa trên tiêu chuẩn ngành để đảm bảo khả năng tương tác.

Thông qua việc ẩn thông tin , các API sẽ cho phép người dùng, ứng dụng sử dụng giao diện độc với việc thực hiện.

Ví dụ: Có 2 webstie A và B đều xử dụng API của Facebook để đăng nhập vào hệ thống. Nhưng quá trình này sẽ được web server thực ngầm, và giao diện có tùy chính với múc đích , mong muốn của chủ thể A và B.

- Theo công ty phần mềm Mona Software thì API có thể sử dụng cho web-based system, operating system, database system, computer hardware, software library. Tài liệu cho giao diện này thường được cung cấp để tạo thuận lợi cho việc sử dụng và triển khai. Hầu hết các công ty lớn đã xây dựng giao diện lập trình này cho khách hàng của họ hoặc để sử dụng nội bộ.

- Hầu như tất cả các môi trường xử lý phần cứng hiện đại đều cung cấp một số cách thức của [lớp trừu tượng phần cứng](https://en.wikipedia.org/wiki/Hardware_abstraction) (HAL) để trừu tượng hóa xung quanh các sản phẩm khác nhau với cùng một họ sản phẩm và có thể mở rộng trên nhiều họ sản phẩm, để dễ dàng di chuyển [phần mềm](https://en.wikipedia.org/wiki/Software_modernization) và [hiện đại hóa phần mềm](https://en.wikipedia.org/wiki/Software_modernization) .

- Khi các hệ thống trở nên phức tạp hơn, Các trung tâm API cũng cần quan tâm để bắt kịp được thời đại:

+ Trong những ngày đầu, API phần cứng thống trị. Một ví dụ điển hình là môi trường phần mềm [CP/M](https://en.wikipedia.org/wiki/CP/M) về cơ bản được bao bọc xung quanh tiêu chuẩn [bus S-100](https://en.wikipedia.org/wiki/S-100_bus) của năm 1974, khi bắt đầu kỷ nguyên [máy tính cá nhân](https://en.wikipedia.org/wiki/Personal_computer). Khi PC trở nên mạnh mẽ hơn, các hệ điều hành mạnh hơn bắt đầu cung cấp môi trường người dùng có cửa sổ để chạy [phần mềm ứng dụng](https://en.wikipedia.org/wiki/Application_software) phức tạp . Các ví dụ ban đầu là [Hệ điều hành Mac cổ điển](https://en.wikipedia.org/wiki/Classic_Mac_OS) được giới thiệu vào năm 1984 và [Microsoft Windows được](https://en.wikipedia.org/wiki/Microsoft_Windows) giới thiệu ở dạng thô sơ vào năm 1985. Điều hành hệ thống truyền thống trừu tượng trên các thiết bị (như ổ đĩa, máy in và modem). Dần dần các thiết bị của người dùng như nhạc cụ có giao diện [MIDI](https://en.wikipedia.org/wiki/MIDI) , [bộ điều khiển trò chơi từ](https://en.wikipedia.org/wiki/Game_controllers) những năm 1980 và thiết bị ngoại vi [USB từ](https://en.wikipedia.org/wiki/USB) những năm 1990 ngày càng bị tóm tắt bởi [trình điều khiển thiết bị](https://en.wikipedia.org/wiki/Device_driver) hệ điều hành , một lớp API quan trọng bên trong chính hệ điều hành.

*+* Về phía máy chủ, [triết lý Unix](https://en.wikipedia.org/wiki/Unix_philosophy) do [Ken Thompson](https://en.wikipedia.org/wiki/Ken_Thompson) tại [Bell Labs](https://en.wikipedia.org/wiki/Bell_Labs) bắt nguồn từ đầu những năm 1970 vẫn là một tiêu chuẩn phần mềm thống trị. Với sự xuất hiện của [Internet](https://en.wikipedia.org/wiki/Internet) công cộng vào những năm 1990, dựa trên [các](https://en.wikipedia.org/wiki/Unix)[giao thức mạng](https://en.wikipedia.org/wiki/Protocol_stack)[Unix](https://en.wikipedia.org/wiki/Unix) , [các API Web](https://en.wikipedia.org/wiki/Web_API) đã trở thành mô hình lập trình thống trị.

+ Năm 2005, Google bắt đầu triển khai [các ứng dụng web](https://en.wikipedia.org/wiki/Web_application) không đồng bộ, sau đó là năm 2007, [Steve Jobs](https://en.wikipedia.org/wiki/Steve_Jobs) đưa [HTML5](https://en.wikipedia.org/wiki/HTML5) và [AJAX trở](https://en.wikipedia.org/wiki/Ajax_(programming)) thành mô hình lập trình trung tâm của [iPhone](https://en.wikipedia.org/wiki/IPhone) trong khi điện thoại [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) của Google cũng áp dụng triết lý tương tự. Với hệ sinh thái [di động](https://en.wikipedia.org/wiki/Mobile_computing) bắt đầu chiếm ưu thế trong việc sử dụng hàng ngày, [JavaScript](https://en.wikipedia.org/wiki/JavaScript)ngày càng nổi bật trong bối cảnh HTML5 và các API phần mềm tập trung vào mạng hiện đã phổ biến cho hầu hết các lĩnh vực doanh nghiệp của con người. Trái ngược với thời đại của máy tính để bàn, nơi một ứng dụng dành cho máy tính để bàn có thể chủ yếu được mã hóa dựa trên một hệ điều hành của nhà cung cấp, ngay cả một ứng dụng di động tương đối đơn giản cũng thường được mã hóa trên cùng trên toàn bộ bộ API thành phần phần mềm có liên quan và chồng chéo, bao gồm những gì bây giờ thường được gọi là [ngăn xếp giải pháp](https://en.wikipedia.org/wiki/Solution_stack) .

b. Lịch sử phát triển của API service:

- Các API có nguồn gốc từ nhiệm vụ [độc lập phần cứng](https://en.wikipedia.org/wiki/Hardware_independent) , bắt nguồn từ hình thức cụ thể cho cộng đồng lập trình hình thành xung quanh [Univac 1108](https://en.wikipedia.org/wiki/UNIVAC_1100/2200_series#1108) , được giới thiệu lần đầu tiên vào năm 1964.

- Ý tưởng về API cũ hơn nhiều so với thuật ngữ này. Các nhà khoa học máy tính người Anh [Wilkes](https://en.wikipedia.org/wiki/Maurice_Wilkes) và [Wheeler](https://en.wikipedia.org/wiki/David_Wheeler_(computer_scientist)) đã làm việc trên các thư viện phần mềm mô-đun vào những năm 1940 cho máy tính [EDSAC](https://en.wikipedia.org/wiki/EDSAC) . [Joshua Bloch](https://en.wikipedia.org/wiki/Joshua_Bloch) tuyên bố rằng Wilkes và Wheeler "phát minh ra" API một cách tiềm ẩn, bởi vì nó là một khái niệm được khám phá nhiều hơn là được phát minh.

- Ý nghĩa của thuật ngữ API đã mở rộng trong lịch sử của nó. Đầu tiên, nó mô tả một giao diện chỉ dành cho các chương trình hướng tới người dùng cuối, được gọi là [cácapplication programs (chương trình ứng dụng](https://en.wikipedia.org/wiki/Application_program)) . Nguồn gốc này vẫn được phản ánh trong tên gọi "giao diện lập trình ứng dụng." Ngày nay, thuật ngữ API rộng hơn, bao gồm cả [phần mềm tiện ích](https://en.wikipedia.org/wiki/Utility_software) và thậm chí cả giao diện phần cứng.

- Thuật ngữ " **Application Programming Interface**" lần đầu tiên được ghi lại trong một bài báo có tên Cấu trúc dữ liệu và kỹ thuật [đồ họa máy tính](https://en.wikipedia.org/wiki/Computer_graphics) từ xa được trình bày tại một hội nghị [AFIPS](https://en.wikipedia.org/wiki/AFIPS) vào năm 1968. Các tác giả của bài báo này sử dụng thuật ngữ để mô tả sự tương tác của một ứng dụng - một chương trình đồ họa trong trường hợp này - với phần còn lại của hệ thống máy tính. Một giao diện ứng dụng nhất quán (bao gồm các lệnh gọi chương trình con [Fortran](https://en.wikipedia.org/wiki/Fortran) ) nhằm mục đích giải phóng lập trình viên khỏi việc xử lý các đặc điểm riêng của thiết bị hiển thị đồ họa và cung cấp sự [độc lập về phần cứng](https://en.wikipedia.org/wiki/Hardware_independence) nếu máy tính hoặc màn hình được thay thế.

*-* API còn được giới thiệu với các lĩnh vực [cơ sở dữ liệu](https://en.wikipedia.org/wiki/Database) bởi [C. J. Date](https://en.wikipedia.org/wiki/Christopher_J._Date) ( là một tác giả, giảng viên, nhà nghiên cứu chuyên về lý thuyết cơ sở dữ liệu quan hệ) trong một bài báo cáo năm 1970 được gọi tên là các [quan hệ](https://en.wikipedia.org/wiki/Relational_database) và [mạng](https://en.wikipedia.org/wiki/Network_model) Phương pháp tiếp cận: So sánh các Application Programming Interface . [[8]](https://en.wikipedia.org/wiki/API#cite_note-8) API trở thành một phần của [khung ANSI / SPARC](https://en.wikipedia.org/wiki/ANSI-SPARC_Architecture) cho [các hệ thống quản lý cơ sở dữ liệu](https://en.wikipedia.org/wiki/Database_management_system) . Khung này xử lý giao diện lập trình ứng dụng riêng biệt với các giao diện khác, chẳng hạn như giao diện truy vấn. Các chuyên gia cơ sở dữ liệu trong những năm 1970 đã quan sát thấy những giao diện khác nhau này có thể được kết hợp với nhau; một giao diện ứng dụng đủ phong phú cũng có thể hỗ trợ các giao diện khác.

- Quan sát này dẫn đến các API hỗ trợ tất cả các kiểu lập trình, không chỉ lập trình ứng dụng. Đến năm 1990, API được định nghĩa đơn giản là "một tập hợp các dịch vụ có sẵn cho một lập trình viên để thực hiện các tác vụ nhất định" bởi nhà công nghệ [Carl Malamud](https://en.wikipedia.org/wiki/Carl_Malamud) .

- Khái niệm về API đã được mở rộng trở lại và được sử dụng rộng dãi nhờ [các API web](https://en.wikipedia.org/wiki/Web_API) . API web hiện là nghĩa phổ biến nhất của thuật ngữ API. Khi được sử dụng theo cách này, thuật ngữ API có một số trùng lặp về ý nghĩa với thuật ngữ **communication protocal** và **remote procedure call** .

c. Tại sao API trở lên phổ biến ?

* 1. Hiểu rõ về chuẩn REST và RESTful API.
  2. Các đặc trưng cơ bản của API service
  3. Phân lại các API :
     1. Phân lại API theo phân ngành nhỏ trong công nghệ thông tin.
     2. Phân loại API theo quyền hạn truy cập
  4. Ưu điểm và nhược điểm của API
  5. Các khó khăn trong trong quá trình xây dựng một API service

Kết luận Chương 1

1. Xác thực người dùng và bảo mật cho API service

Chương này trình bày...

*2.1 Xác thực người dùng trong API service*

* + 1. Authentication và Authorization
    2. HTTP Authentication
       1. Basic Authentication
       2. Bearer Authentication
    3. API Keys
       1. Khái niệm API keys
       2. Nguyên lý hoạt động
       3. Ưu điểm nhược điểm tại sao nên sử dụng API keys
    4. OAuth (2.0)
       1. Khái niệm
       2. Nguyên lý hoạt động
       3. Ưu điểm nhược điểm
    5. Phương pháp kết nối OpenID
  1. Bảo mật trong API serive
     1. Request injection
     2. Spam request
     3. Chặn outside request

Kết luận Chương 2

1. Thực nghiệm xây dựng Web API theo chuẩn RESTful và an toàn
   1. Giới thiệu các công nghệ sử dụng trong phần thực nghiệm
   2. Mô hình , lược đồ
   3. Thực nghiệm
   4. Kết quả, ý nghĩa

# Tài liệu tham khảo để làm đề cương

[] 4 Most used REST API Authentication Methods

https://blog.restcase.com/4-most-used-rest-api-authentication-methods/

[2] What is REST

https://restfulapi.net/

[] Khái niệm API theo Wikipedia  
<https://en.wikipedia.org/wiki/API>

[4] https://backendless.com/what-is-api-as-a-service/#:~:text=API%20(Application%20Programming%20Interface)%20services,change%20the%20data%20within%20it.