# The OAuth 2.0 Authorization Framework

Trừu tượng hóa:

The OAuth 2.0 authorization framework cho phép application bên thứ 3 có thể lấy được truy cập giới hạn tới một HTTP service, hoặc là dựa trên danh nghĩa chính chủ sở hữu tài nguyên và HTTP service hoặc dựa trên chính họ.

# Nội dung của OAuth 2.0

## Các khái niệm

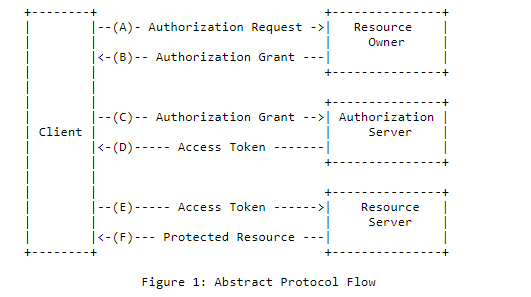
### Role

Trong thiết kế của OAuth 2.0 thì nó đã định nghĩa với 4 roles. Khái niệm role ở đây là role trong thiết kế của giao thức này, không nên nhầm lẫn giữa khai niệm role của giao thức này với các khai niệm role trong các context khác (như trong identity framework của AspNetCore có khái niệm Role để phần quyền user)

1. Resource owner – Thường hiểu là end-user, người mà có thể trao cho ta các thông tin để xác thực và từ các thông tin đó sau khi xác thực thành công sẽ được cấp quyền để truy cập vào tài nguyên
2. Resource server – Nó là một server, nơi mà làm host lưu giữ các tài nguyên được bảo vệ
3. Client – Hiểu đơn giản thì client là các application mà ta dùng nó dựa trên danh nghĩa của end-user để truy cập vào tài nguyên.
4. Authorization server – Server này là nơi mà phát cho client một loại mã mà có thể truy cập vào các resource server, resource server sẽ gọi lên authorization server để kiểm tra xem chứng chỉ đó có chính xác hay k, nếu đúng nó sẽ dựa trên quyền của chứng chỉ để cấp tài nguyên hoặc reject

### Protocol Flow

Ta có thiết kế trừu tượng của protocol flow như sau



1. - Client (application) yêu cầu Resource owner (end-user) cấp quyền
2. - Resource owner trao cho client Authorization Grant. Đây là một loại chứng chỉ mà resuorce owner cấp cho client
3. - Client dùng Authorization Grant này để yêu cầu Authorization Server cấp cho nó Access Token (một loại chứng chỉ khác có chức năng truy cập resource server)
4. - Authorization Server cấp cho Client một Access Token để có thể dùng vào việc truy cập trên các resource server
5. - Client dùng Access Token để lấy tài nguyên trên Resource Server
6. - Resource Server cấp tài nguyên cho client

Nhìn như trên thì ta có thể phân tách flow của giao thức thành 3 bước

B1 - Lấy chứng chỉ của end-user mà ở đây là Authorization Grant

B2 – Lấy chứng chỉ truy cập tài nguyên, ở đây là Access Token. Bằng cách dùng chứng chỉ của end-user để trao đổi lấy chứng chỉ truy cập tài nguyên

B3 – Dùng chứng chỉ truy cập tài nguyên như một cái chìa khóa để có thể đi vào lấy tài nguyên trên các Resource Servers

Việc Implementation của flow này dựa trên cách thức mà một end-user trao cho client chứng chỉ như thế nào, bởi vì việc trao chứng chỉ rất là nhạy cảm vì có thể bị đánh cắp. Có 4 cách thức để một end-user trao chứng chỉ cho client với độ an toàn ưu nhược điểm khác nhau.

### Các cách thức end-user cấp chứng chỉ cho client (grant type)

Một authorization grant là một loại chứng chỉ đại diện cho quyền truy cập của resource owner. Nó được client sử dụng để trao đổi lấy access token ở B2.

1. Authorization Code

Với cách thức cấp chứng chỉ này, end-user không cấp trực tiếp cho client mà sẽ gián tiếp thông qua authorization server. Tức là khi client yêu cầu cấp chứng chỉ thì end-user sẽ được chuyển hướng tới trên authorization server, ở đây end-user nhập các chứng chỉ trực tiếp cho authorization server vì trên này sẽ bảo mật an toàn hơn. Sau đó server sẽ trả về một đoạn code gọi là Authorization code nó hiểu như là chứng chỉ của user nhưng dưới dạng code. Và client dùng đoạn code này để trao đổi access token với authorization server.

1. Implicit
2. Password credentials
3. Client credentials

### Access Token

### Refresh Token

### TLS Version

Transport Layer Security

### HTTP Redirections

### Interoperability

### Notational Conventions

## Client Registration

Với giao thức này, the client cần đăng ký với authorization server. Việc đăng kí này không yêu cầu tương tác trực tiếp giữa client và authorization server. Nó có thể dựa trên các cách khác như là thiết lập sự tin tưởng và lấy các properties được yêu cầu tới client.

Khi mà đăng kí một client cần:

* chỉ định kiểu của client
* cung cấp URI redirections
* Bao gồm bất kì thông tìn nào được yêu cầu khác từ server như logo, website,...

### Client types (các loại client)

OAuth định nghĩa 2 kiểu của client dựa trên độ an toàn của nó là confirential và public

web application – là loại application confidential chạy trên web server. Resource owner truy cập client thông qua một HTML UI được render trong user-agent trên thiết bị. Tất cả các chứng chỉ của client cũng như access token được phát thì lưu trên web server nên bảo mật cao, cái mà người dùng thấy chỉ là các phần ta cho phép render ra mà thôi.

user-agent-based application – là loại application public với code được download từ webserver và code được execute trên user-agent (như là web-browser). Ta lưu bất cứ thông tin nào trên thằng này thì đều bị show ra hết, source code xem được hết.

native application – là một public application được cài trên thiết bị và chạy trên thiết bị, và thông tin có thể bị truy cập bởi resource owner.

### Client Identifier

authorization server phát cho client đã đăng kí một id. thằng này thì bị phơi ra nên cấm sử dụng một mình nó khi xác thực client.

### Client Authentication

Với confidential client thì xác thực là cần thiết. Còn với public client thì thằng này chỉ là optional mà thôi, tuyệt đối không dựa trên public client authentation cho việc định danh một client

The client tuyệt đối không sử dụng nhiều hơn một method authentication trong cùng một request.

### Client Password

Phần này nói về cách thức sử dụng để request lên xác thực ntn.

### Unregistered Clients

## Protocol Endpoints

Có 2 endpoints trên authorization server

* Authorization endpoint
* Token endpoint

Có 1 client endpoint

* Redirection endpoint – được sử dụng bởi server để trả về phản hồi chứa các chứng chửn authorization tới client thông qua user-agent

## Authorization Endpoint

Đây là nơi mà authorization server tương tác với resource owner để lấy được authorization grant (có thể sử dụng usernam và password login, session cookies)

### Response Type

Với authorization endpoint thì nó được sử dụng trong 2 kiểu authorization grant là code và implicit. Nên client cần chỉ định kiểu mong muốn tới authorization server để nó còn biết mà trả về

### Redirection Endpoint

Nơi mà authorization server sẽ điều hướng user-agent quay trở lại kèm với authorization grant sau khi lấy được thông tin end-user xong

### Endpoint Request Confidentiality

### Registration Requirements

The authorization server bắt buộc yêu cầu các client sau đây phải đăng kí endpoint redirection của họ:

* public clients
* Confidential clients sử dụng implicit grant type

The authorization server sẽ yêu cầu tất cả clients đăng kí redirection endpoint để sử dụng authorization endpoint.

Nếu không có endpoint thì authorization server không xác thực được client và không có URI để chuyển hướng lại client.

### Dynamic Configuration

Nếu có multiple redirection được đăng kí, nếu chỉ một phần của redirection được đăng kí hoặc nếu không có redirection nào được đăng kí, the client bắt buộc phải thêm redirection URI cùng với authorization request bằng việc sử dụng “redirect\_uri” parameter.

Khi một redirection URI được thêm theo kèm authorization request, authorization server bắt buộc phải so sánh và match giá trị đã nhận với ít nhất một thằng trong các redirections đã đăng kí.

### Invalid Endpoint

### Endpoint Content

The redirection request tới the client’s endpoint thường đưa đến kết quả trong một HTML document, được xử lý bởi user-agent. Nếu HTML response được phục vụ trực tiếp, bất kì một script nào được thêm vào HTML document sẽ execute với full access tới redirection URI và credentials nó chứa.

## Token Endpoint

### Client Authentication

Confidential clients hoặc các clients khác đã cấp client credentials bắt buộc phải xác thực với server khi request tới token endpoint. Client authentication được sử dụng cho các mục đích sau:

* Thực thi sự ràng buộc của refresh tokens và authorization codes tới client đã phát.
* Phục hồi cho client bị vô hiệu hóa hoặc thay đổi chứng chỉ của nó, trống lại những kẻ đánh cắp refresh tokens

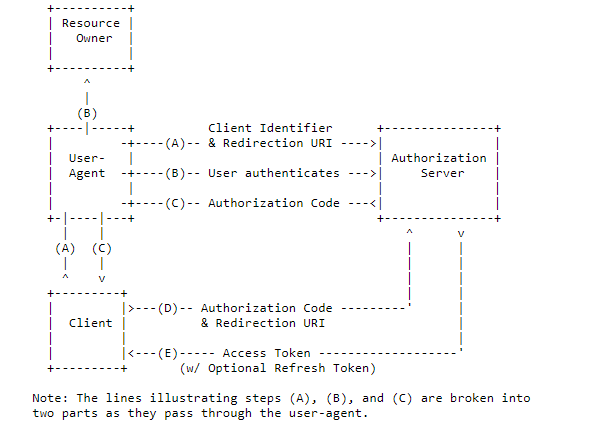
### Access Token Scope

## Obtaining Authorization

Để request một access token, the client cần lấy được authorization từ resource owner.

### Authorization Code Grant

Sử dụng cho confidential client, không dùng cho public client



### Authorization Request



### Authorization Response

Sau khi được end-user cấp quyền, authorization code được phát tới client bằng việc thêm vào các paramenters tới the query component của the redirection URI.

* Code – REQUIRED : The authorization code phải bắt buộc có thời gian hết hạn ngắn, dài nhất là 10’. Nghiêm cấm client sử dụng authorization nhiều hơn 1 lần. Nếu bị sử dụng nhiều hơn 1 lần, authorization server sẽ thu hồi lại toàn bộ token đã phát trên authorization code này
* state – REQUIRED nếu the “state” có trong request

