TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

ĐHQG-HCM

---◆---

BÁO CÁO ĐỒ ÁN SEMINAR LẦN 2

– NHÓM 14 –

TOPIC 2 – TRUST NEGOTIATION

Sinh viên thực hiện: Nguyễn Trần Minh Tuấn – 20127420

Tô Đình Phương Nam – 20127569

Phan Trí Tài – 20127318

Phạm Hiếu Khải - 20127523

Lớp: CSC15003_20MMT-CNTThuc

I. Single Sign-On (SSO)

1. Giới thiệu

Single Sign-On, đúng như tên gọi, là cơ chế cho phép người dùng có thể truy cập nhiều trang web, ứng dụng mà chỉ cần đăng nhập một lần. Một khi đã được định danh ở một trang website A, thì cũng sẽ được định danh tương tự ở website B mà không cần lặp lại thao tác đăng nhập.

Trong bối cảnh người dùng ngày nay thường xuyên truy cập trực tiếp vào ứng dụng từ trình duyệt của họ, các tổ chức đang ưu tiên áp dụng những chiến lược quản lý quyền truy cập có thể cải thiện cả khả năng bảo mật và trải nghiệm người dùng.

2. Cơ chế hoạt động

Hệ thống nhận dạng liên kết (Federated Identity Glossary) là nơi tập trung và liên kết thông tin người dùng. Có 4 yếu tố nền tảng cấu thành nên hệ thống này:

- Xác thực (Authentication): kiểm tra thông tin đăng nhập và tiến hàng định danh người dùng.
- Phân quyền (Authorization) dựa trên thông tin định danh để kiểm tra quyền truy cập của user.
- Trao đổi thông tin người dùng (User attributes exchange): Mỗi hệ thống con sẽ cần và lưu trữ các thông tin khác nhau của người dùng, tuy nhiên sẽ có các thông tin bị lặp lại, ví dụ như tên, họ.... Do đó, cần có một nơi để tổng hợp lại các thông tin này, và trao đổi cho các hệ thống con.
- Quản lí người dùng (User management): admin có thể quản lí người dùng bằng các thao tác thêm, sửa, xóa... ở các hệ thống con.

<u>Quy trình SSO:</u>

- Khi người dùng đăng nhập vào một ứng dụng, ứng dụng sẽ tạo mã thông báo SSO và gửi yêu cầu xác thực đến dịch vụ SSO.
- Dịch vụ sẽ kiểm tra xem người dùng đã được xác thực trước đó trong hệ thống hay chưa. Nếu đã xác thực, dịch vụ sẽ gửi một phản hồi xác nhận xác thực đến ứng dụng để cấp quyền truy cập cho người dùng.
- Nếu người dùng không có thông tin chứng thực đã xác minh, dịch vụ SSO sẽ chuyển hướng người dùng đến hệ thống đăng nhập trung tâm và nhắc người dùng gửi tên người dùng và mật khẩu của họ.

- Sau khi gửi, dịch vụ xác mình thông tin chứng thực của người dùng và gửi phản hồi tích cực cho ứng dụng.
- Nếu không, người dùng sẽ nhận được thông báo lỗi và phải nhập lại thông tin chứng thực. Nhiều lần đăng nhập không thành công có thể dẫn đến việc dịch vụ chặn người dùng thử đăng nhập lại trong một khoảng thời gian cố định.

3. Ưu và nhược điểm

Ưu điểm

- Tiết kiệm thời gian cho người sử dụng trong việc đăng nhập vào nhiều dịch vụ được cung cấp trên các nền tảng khác nhau của hệ thống phân tán.
- Tăng cường khả năng bảo mật thông qua việc giúp người sử dụng không cần nhớ nhiều thông tin đăng nhập (định danh và mật khẩu).
- Giúp cho người quản trị hệ thống tiết kiệm thời gian trong việc tạo lập hay loại bỏ người dùng trên hệ thống, cũng như thay đổi quyền của một hay một nhóm người dùng nào đó.
- Tiết kiệm thời gian khi tái lập lại mật khẩu cho người dùng.
- Bảo mật các cấp độ của việc thoát hay truy xuất hệ thống.
- Người phát triển ứng dụng không cần thiết phải hiểu và thực hiện nhận dạng bảo mật trong ứng dụng của họ, điều họ cần làm là liên kết đến một máy chủ định danh đã được bảo đảm, việc này giúp những người dùng của họ có thể truy cập vào các dịch vụ khác cũng liên kết đến máy chủ đó như họ.
- Tạo nên sự đồng bộ giữa các dịch vụ và ứng dụng trong cùng một hệ thống thông tin phục vụ người dùng.

Khuyết điểm

- Đòi hỏi cơ sở hạ tầng của toàn bộ hệ thống phải bảo đảm.
- Do nhiều domain cùng sử dụng chung cơ sở dữ liệu người dùng nên việc xác thực khi người dùng đăng ký với hệ thống phải chặt chẽ, nếu không sẽ rất dễ vi phạm việc đảm bảo an ninh cho hệ thống.
- Cần có cơ chế xác thực đảm bảo khi truyền các thông tin định danh người dùng giữa người sử dụng với các máy chủ dịch vụ khác nhau.
- Chi phí để triển khai hệ thống SSO là rất tốn kém, cả về phần mềm, phần cứng lẫn nguồn nhân lực, cần phải có sự tính toán cẩn thận trước khi triển khai.

4. Các loại giao thức

SAML

SAML, hoặc Ngôn ngữ đánh dấu xác nhận bảo mật, là một giao thức hoặc tập hợp các quy tắc mà các ứng dụng sử dụng để trao đổi thông tin xác thực với dịch vụ SSO. SAML sử dụng XML, một ngôn ngữ đánh dấu thân thiện với trình duyệt, để trao đổi dữ liệu nhận dạng người dùng. Các dịch vụ SSO dựa trên SAML có tính bảo mật và tính linh hoạt tốt hơn, vì các ứng dụng không cần lưu trữ thông tin chứng thực của người dùng trên hệ thống của chúng.

OAuth

OAuth, hay Xác thực mở, là một tiêu chuẩn mở cho phép các ứng dụng truy cập một cách bảo mật vào thông tin người dùng từ các trang web khác mà không cần cung cấp mật khẩu. Thay vì yêu cầu mật khẩu của người dùng, các ứng dụng sử dụng OAuth để được người dùng cho phép truy cập vào dữ liệu được bảo vệ bằng mật khẩu. OAuth xác lập tín nhiệm giữa các ứng dụng thông qua API, cho phép ứng dụng gửi và phản hồi các yêu cầu xác thực trong một khuôn khổ đã thiết lập.

OIDC

OpenID là một phương pháp sử dụng một tập hợp thông tin chứng thực của người dùng để truy cập nhiều trang web. Nó cho phép nhà cung cấp dịch vụ đảm nhận vai trò xác thực thông tin chứng thực của người dùng. Thay vì chuyển mã thông báo xác thực cho bên thứ ba cung cấp danh tính, các ứng dụng web sử dụng OIDC để yêu cầu thông tin bổ sung và xác minh tính xác thực của người dùng.

Kerberos

Kerberos là một hệ thống xác thực dựa trên phiếu cho phép hai hoặc nhiều bên xác minh lẫn nhau danh tính trên mạng của họ. Hệ thống này sử dụng mật mã bảo mật để ngăn chặn truy cập trái phép vào thông tin nhận dạng được truyền giữa máy chủ, máy khách và Trung tâm phân phối khóa.

5. Tầm quan trọng của SSO

SSO là một phương tiện đặc biệt quan trọng để tăng tính bảo mật và hiệu quả của các hệ thống và ứng dụng máy tính.

- Tiết kiệm thời gian: Người dùng không cần phải đăng nhập nhiều lần khi truy cập nhiều ứng dụng khác nhau.
- Tăng cường bảo mật mật khẩu: SSO giúp hạn chế việc quên mật khẩu và khuyến khích người dùng tạo một mật khẩu mạnh có thể được sử dụng cho nhiều trang web.
- Tăng cường an ninh: Với SSO, thông tin đăng nhập chỉ được nhập một lần và được xác thực bởi một hệ thống quản lý chung. Điều này giúp giảm thiểu nguy cơ bị hack mật khẩu hoặc tấn công
- Tăng hiệu quả làm việc: Người dùng dễ dàng truy cập vào các ứng dụng khác nhau chỉ bằng việc đăng nhập một lần duy nhất. Điều này giúp tăng hệ thống hạ tầng và giảm thời gian giải quyết vấn đề kỹ thuật liên quan đến việc quản lý tài khoản của từng ứng dụng.

6. Social Login

Social login cũng là một dạng của single sign-on (SSO), cho phép người dùng đăng nhập vào một ứng dụng hoặc trang web bằng các tài khoản mạng xã hội của họ, thay vì phải tạo tài khoản mới hoặc đăng nhập bằng thông tin đăng nhập truyền thống như email và mật khẩu.

Ưu điểm:

- Giảm số lượng tên đăng nhập (username) và mật khẩu (password) mà người dùng cần phải ghi nhớ.
- Giảm thiểu sự cản trở của các bước đăng nhập và đăng ký truyền thống, giúp tăng tính tiện lợi cho người dùng.
- Social login giúp giảm thiểu sự phân tán thông tin người dùng. Thay vì cung cấp thông tin đăng nhập khác nhau cho từng dịch vụ, người dùng chỉ cần cung cấp thông tin đăng nhập vào tài khoản mạng xã hội của họ và các dịch vụ ứng dụng sẽ lấy thông tin từ đó.

Nhược điểm:

- Phát sinh thêm các chi phí phát triển khi thông qua server bên thứ ba
- Phải phụ thuộc vào bên cung cấp dịch vụ SSO

- Các ứng dụng và trang web phải được tích hợp với các nhà cung cấp dịch vụ mạng xã hội khác nhau để hỗ trợ Social Login.
- Việc phụ thuộc vào một số nền tảng mạng xã hội lớn có thể làm cho việc đăng nhập trở nên khó khăn hơn nếu nhà cung cấp dịch vụ đó gặp sự cố.

II. ĐỒ ÁN

1. Oauth 2.0

- Các vai trò:

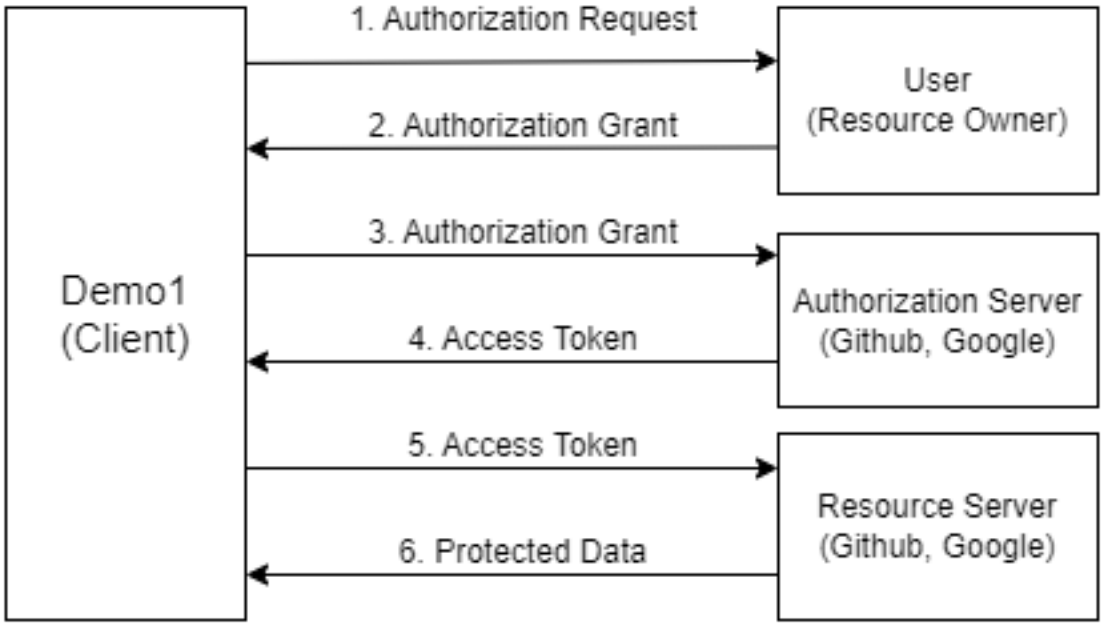
    + **Resource Owner**: Người sở hữu tài nguyên, là người dùng cuối cùng muốn cho phép ứng dụng của bên thứ ba truy cập vào tài nguyên của họ

    + **Client**: Ứng dụng của bên thứ ba muốn truy cập vào tài nguyên của người dùng. Để sử dụng OAuth, Client phải đăng ký với nhà cung cấp OAuth và nhận được Client ID và Client Secret.

    + **Authorization Server**: Máy chủ xác thực quản lý quy trình xác thực và ủy quyền cho ứng dụng của bên thứ ba để truy cập vào tài nguyên của người dùng.

    + **Resource Server**: Dịch vụ trực tuyến cung cấp tài nguyên mà ứng dụng của bên thứ ba muốn truy cập.

- Sơ đồ:



+ Client yêu cầu ủy quyền (Authorization Request) từ Authorization Server, cung cấp client ID và secret làm nhận dạng. Đồng thời, cũng cung cấp các scope và endpoint URI (redirect URI) để gửi Access Token hoặc Authorization Code.

+ Authorization Server xác thực client và xác minh rằng scope yêu cầu được cho phép.

+ Resource owner tương tác với Authorization Serve để cấp quyền truy cập.

+ Authorization Server chuyển hướng trở lại client bằng Authorization Code hoặc Access Token, tùy thuộc vào loại. Refresh Token cũng có thể được return.

+ Với Access Token, client yêu cầu quyền truy cập vào tài nguyên từ máy Resource server.

2. Mã giả:

a. Đăng ký ứng dụng với OAuth2 Provider

SET Homepage URL = http://localhost:8080

SET Github Authorization callback URL = http://localhost:8080/login/oauth2/code/github

SET Google Authorization callback URL = http://localhost:8080/login/oauth2/code/google

b. Xác định các thông tin xác thực từ OAuth2 Provider (config application.yml)

SET Github ClientId = 5efbe86270da095f4e71

SET Github ClientSecret = b0f02576d089ee52d8fc6e1c9ffc0f1a4bd1193e

SET Google ClientId = …

SET Google ClientSecret = …

c. Front-end

- Link xác thực đuợc đăng ký với OAuth2 Provider:

```
<div> With GitHub: <a href="/oauth2/authorization/github">click here</a> </div>
<div> With Google: <a href="/oauth2/authorization/google">click here</a> </div>
```

- Hiển thị trang web khi chưa xác thực và đã xác thực

Tên đăng nhập:

```
$.get("/user", function(data) {
    $("#user").html(data.name);
    $(".unauthenticated").hide();
    $(".authenticated").show();
});
```

d. Cấu hình Oauth2 Client

- Lấy thuộc tính login (name) từ Github (Google):

```
@RequestMapping("/user")
public Map<String, Object> user(@AuthenticationPrincipal OAuth2User principal) {
    if(principal.getAttribute("email") == null) {
        return Collections.singletonMap("name", principal.getAttribute("login"));
    }
    return Collections.singletonMap("name", principal.getAttribute("name"));
}
```

- Sử dụng configure(HttpSecurity http) để cấu hình các quy tắc bảo mật trong ứng dụng

```
@Override
protected void configure(HttpSecurity http) throws Exception {
    http
        .authorizeRequests(a -> a .antMatchers("/", "/error", "/webjars/**
        .exceptionHandling(e -> e.authenticationEntryPoint(new HttpStatusE
        .csrf(c -> c.csrfTokenRepository(CookieCsrfTokenRepository.withHtt
        .logout(l -> l.logoutSuccessUrl("/").permitAll())
        .oauth2Login();
}
```

III. TIẾN ĐỘ BÁO CÁO CHI TIẾT

Thời gian	Nội dung
22/2 - 28/2	Chọn chủ đề Viết báo cáo seminar 1
1/3 – 10/3	Tìm hiểu về Single Sign-On Đọc tài liệu của thầy gửi Phân chia công việc
11/3 – 17/3	Tìm hiểu *SAML, OAuth, OIDC, Kerberos* Chọn OAuth 2.0 để demo
18/3 – 24/3	Tìm hiểu ngôn ngữ Java
25/3 – 31/3	Tìm hiểu Spring boot Viết báo cáo seminar 2 Code demo
1/4 – 7/4	Hệ thống lại nội dung Fix bug
8/4 – 14/4	Thiết kế power point
15/4 – 21/4	Tập dượt thuyết trình Viết báo cáo seminar 3