معماری احراز هویت در نرم افزار های بزرگ چیست؟

امروزه یافتن یک برنامه کاربردی جدی که در آن نیازهای امنیتی و کنترل دسترسی وجود نداشته باشد، اگر ناممکن نباشد کاری بسیار سخت خواهد بود. در دنیای اینترنت و درجایی که تمامی ارتباطات، تبادلات بانکی، تماسها و... درحال الکترونیکی شدن هستند، اثبات وجود خود در اینترنت، اهمیت بسیاری پیدا میکند و با گسترش فناوری اطلاعات و ارتباطات، امنیت به یکی از مهمترین مباحث در فرآیند طراحی و مدیریت سازمانها تبدیل شده است و در تمام زیرسیستمها و سرویسهای درون‌سازمانی برای استفاده از خدمات اختصاصی باید به طریقی هویت خود را آشکار سازید.

بطور کلی، برای برقراری یک محیط امن، چند فاکتور اساسی باید موجود باشد. این فاکتورها عبارتند از:

جامعیت:

اطمینان از اینکه اطلاعات، صحیح و کامل است؛ یعنی دستخوردگی و عدم تغییر پیام و اطمینان از آنکه داده‌ها با اطلاعات مخرب مثل یک ویروس آلوده نشده‌اند.

محرمانگی:

اطمینان از اینکه اطلاعات تنها توسط افراد یا سازمانهای مجاز قابل استفاده است و هیچگونه فاش‌سازی اطلاعات برای افراد تشخیص و تأیید هویت نشده، صورت نخواهد گرفت.

شناسایی و احراز هویت:

گیرنده و فرستنده، هر دو باید بتوانند از هویت طرف مقابل خود مطمئن باشند و اطمینان از اینکه پیام حقیقت از کسی که تصور میکنید، رسیده باشد.

انکارناپذیری:

هیچیک از دو سوی ارتباط نتوانند مشارکت خود در ارتباط را انکار کنند؛ یعنی تمام امکانات شبکه بدون دردسر و زحمت در اختیار آنها که مجاز به استفاده از شبکه هستند، باشند و در ضمن هیچکس نتواند در دسترسی به شبکه ایجاد اشکال نماید

۱- شناسایی و احراز هویت (I & A)

شناسایی و احراز هویت دو هسته اصلی در بیشتر سیستم‌های کنترل دسترسی هستند.

شناسایی عملی است که یک کاربر برای معرفی خود به یک سیستم اطلاعاتی استفاده میکند که معمولا در قالب یک شناسه ورود یا Login ID وجود دارد. درواقع شناسایی، فرایندی است که طی آن، فرد هویتی را اظهار میکند و از این نقطه، پاسخگویی آغاز میشود. ارائه نام کاربری، شناسه ورود، شماره شناسایی یا کارت هوشمند توسط کاربر، نمایانگر فرایند شناسایی است.

احراز هویت عبارتست از فرآیند بررسی اعتبار هویت ادعاشده. احراز هویت نیازمند ارائه اطلاعات دیگری توسط فرد است که باید دقیقاً با هویت مذکور، مرتبط باشد و یکی از مهمترین مباحث در حوزه امنیت سیستمهای کامپیوتری میباشد. احراز هویت، هویت فرد را بوسیله مقایسه یک یا چند عامل با پایگاه داده‌ای از هویتهای معتبر، بررسی میکند. بصورت کلی در حوزه امنیت اطلاعات، احراز هویت یا Authentication به چهار تیپ ساده تقسیم‌بندی میشود:

۱- چیزی که شما میدانید (What You Know)

۲- چیزی که شما دارید (What You Have)

۳- چیزی که شما هستید (What You Are)

۴- آنچه که شما انجام میدهید (What You DO)

Authentication چیست؟

Authentication یا احراز هویت به فرآیندی گفته می شود که در آن ارسال کننده یا دریافت کننده اطلاعات برای همدیگر اطلاعاتی را ارائه می کنند تا مطمئن شوند آنها همانی هستند که ادعا می کنند. به نوعی در احراز هویت بررسی می شود که شخص یا ... همانی هست که ادعا می کند.

Authorization: در مورد مجوز است یعنی که شما مجاز به دسترسی به سرویس شخص ثالث برای استفاده از داده های شخصی هستید

Authentication چیست؟ فرآیندی که در اون هویتِ کاربر مشخص می شه. مثل "ورود" در خیلی از سایت ها. همیشه هم با صفحه ی Login اتفاق نمی افته. شاید با یک دستگاهِ اثرِ انگشت انجام بشه.

Authorization چیست؟ فرآیندی که در اون اختیاراتِ کاربر مشخص می شه. مثل زمانی که کاربرِ "خریدار" اختیارِ خرید کردن رو داره و کاربرِ "ادمین" اختیارِ وارد کردنِ کالاهای جدید به سایت.

انواع روش های احراز هویت:

Open id

Sso

OAuth

OAuth2

Sso چییست :

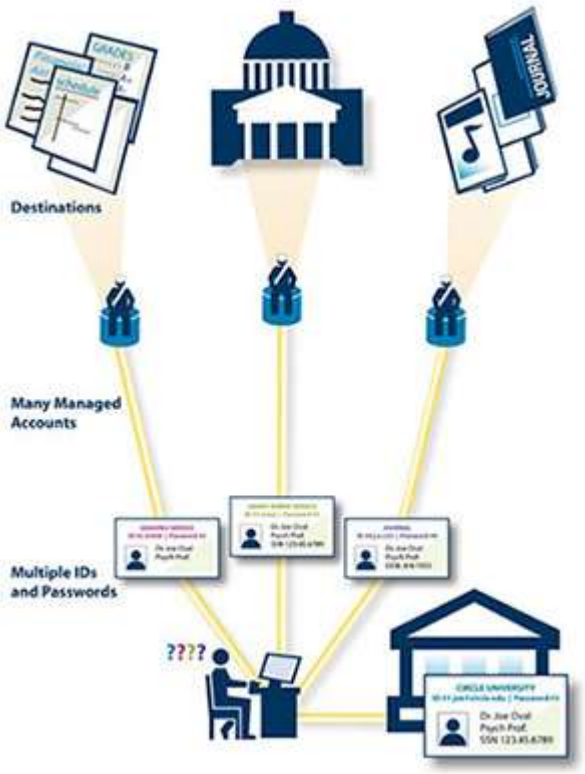
سیستم امنیت متمرکز (SSO)

SSO مخفف Single Sign On به عمل وارد شدن یک کاربر به سایتها و برنامه‌های مختلف تنها با یک نام کاربری و گذرواژه یکسان میگویند.

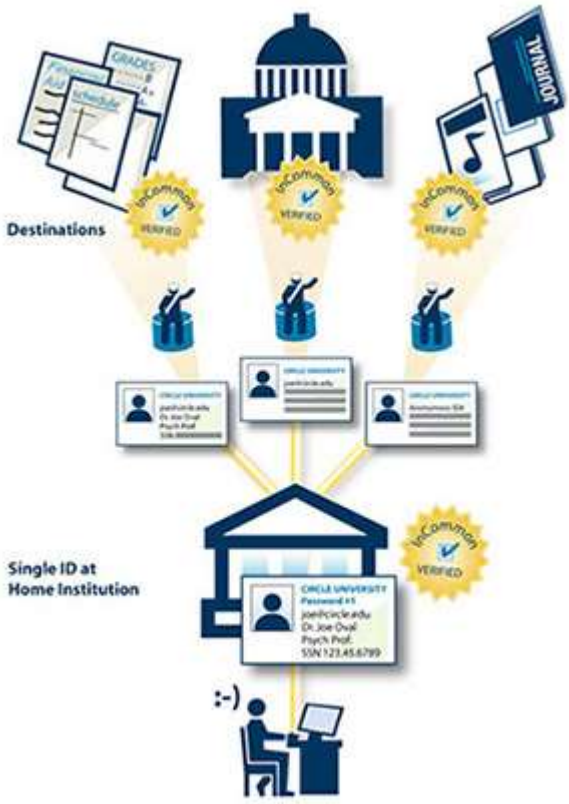
به این معنی که اطلاعات مربوط به اعتبارسنجی و تائید هویت کاربر یعنی user name و password ، در یک ناحیه امن بصورت موقت نگهداری میشود و از آن پس این کاربر بمنظور ورود به سایتها و بخشهای مختلف)دسترسی به برنامه‌های متعدد) نیازی نیست مجدداً Login نماید . این سیستم قابلیت متمرکز کردن احراز هویت انواع سیستمهای نرم‌افزاری، فارغ از مکانیزم مورد استفاده برای احراز هویت را در خود دارد. در ساختار SSO کاربر صرفا یک نام کاربری و رمز عبور یکتا در شبکه دارد که بلافاصله بعد از اینکه وارد یک سیستم در ساختار SSO شود میتواند از تمامی منابعی که برای وی در شبکه تعریف شده است استفاده کند، بدون نیاز به اینکه برای هر منبع اطلاعاتی نام کاربری و رمز عبور جدیدی وارد کند.

در SSO امنیت کلمه عبور بسیار حائز اهمیت است و اکانتهایی که بین سیستمها انتقال پیدا مینمایند باید بصورت رمزنگاری شده بکار گرفته شوند. سهولت در ویرایش اکانتها)مثل تغییر رمز عبور) و حذف حسابهای کاربری در مدیریت سیستمها، از مزیتهای SSO به شمار میایند.

معماری احراز هویت بدون sso



معماری احراز هویت با sso



معماری احراز هویت به روش opoen id

OpenID به شما اجازه می‌دهد با استفاده از اکانت (نام کاربری) که در یک سایت دارید بتوانید به سایت‌های متفاوتی وارد شوید (لاگین کنید) بدون این که نیاز به ثبت نام دوباره در آن سایت‌ها داشته باشید.

نمونه بارز آن می‌توان به سایت هایی مانند StackOverflowاشاره کرد.

OpenID به شما اجازه می‌دهد شما در یک نام کاربری که برای خود ایجاد کرده اید اطلاعاتی را که دارید با دیگر وبسایت‌ها به اشتراک بگذارید.

با OpenID کلمه عبور شما فقط توسط سرویس دهنده گرفته میشود و سرویس دهنده هویت شما را برای بازید از سایت دیگر تایید می‌کند.از طرف دیگر سرویس دهنده شما تا شما اجازه ندهید هیچ وب سایتی کلمه عبور شما را به هیچ وب سایتی نمی‌بینند. بنابراین نیازی نیست در مورد کلمه عبور خود نگرانی به خود راه دهید.

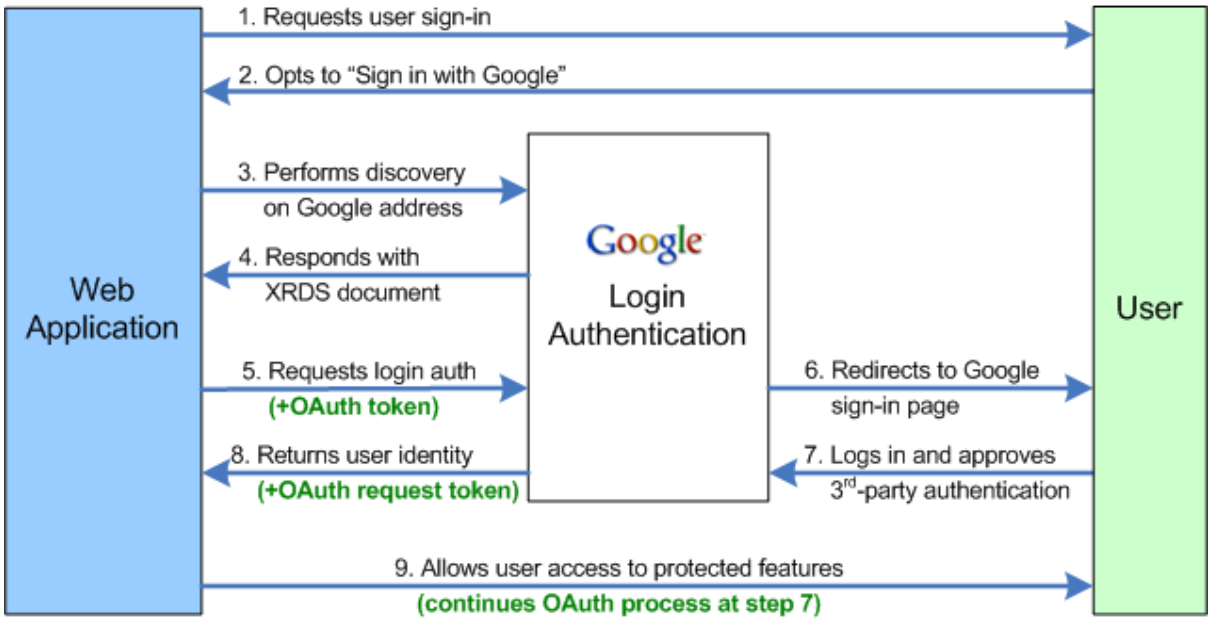
OpenID به سرعت در حال گسترش بروی وب استف در حال حاضر بیش از یک میلیارد نام کاربری (اکانت) وجود دارد و بیش از ۵۰۰۰۰ سایت OpenID را پذیرفته و با آن کار می‌کنند. چندین سازمان بزرگ موضوع OpenID را پذیرفته اند، سازمان هایی مانند Google, FaceBook,Yahoo!,Microsoft,AOL,MySpace,Sears, Universal Music Group, France Telecom, Novell, Sun, Telecom Italia

و بسیاری از سازمان‌های دیگر.

در کل می‌توان گفت ما در یک سایت خاص مانند یاهو ، گوگل، و... یک نام کاربری خواهیم داشت سپس برای ارتباط سایت مقصد (مثلا همین سایت) با نام کاربری ما در گوگل به این صورت عمل می‌شود که ابتدا از طریق این سایت ما به سایت گوگل هدایت می‌شویم در آنجا از ما یک تاییدیه جهت استفاده از سرویس OpenID از کاربر میگیرد. در صورت تایید کاربر سایت گوگل از این لحظه جهت احراز هویت کاربر برای ورود به سایت مقصد استفاده می‌کند .

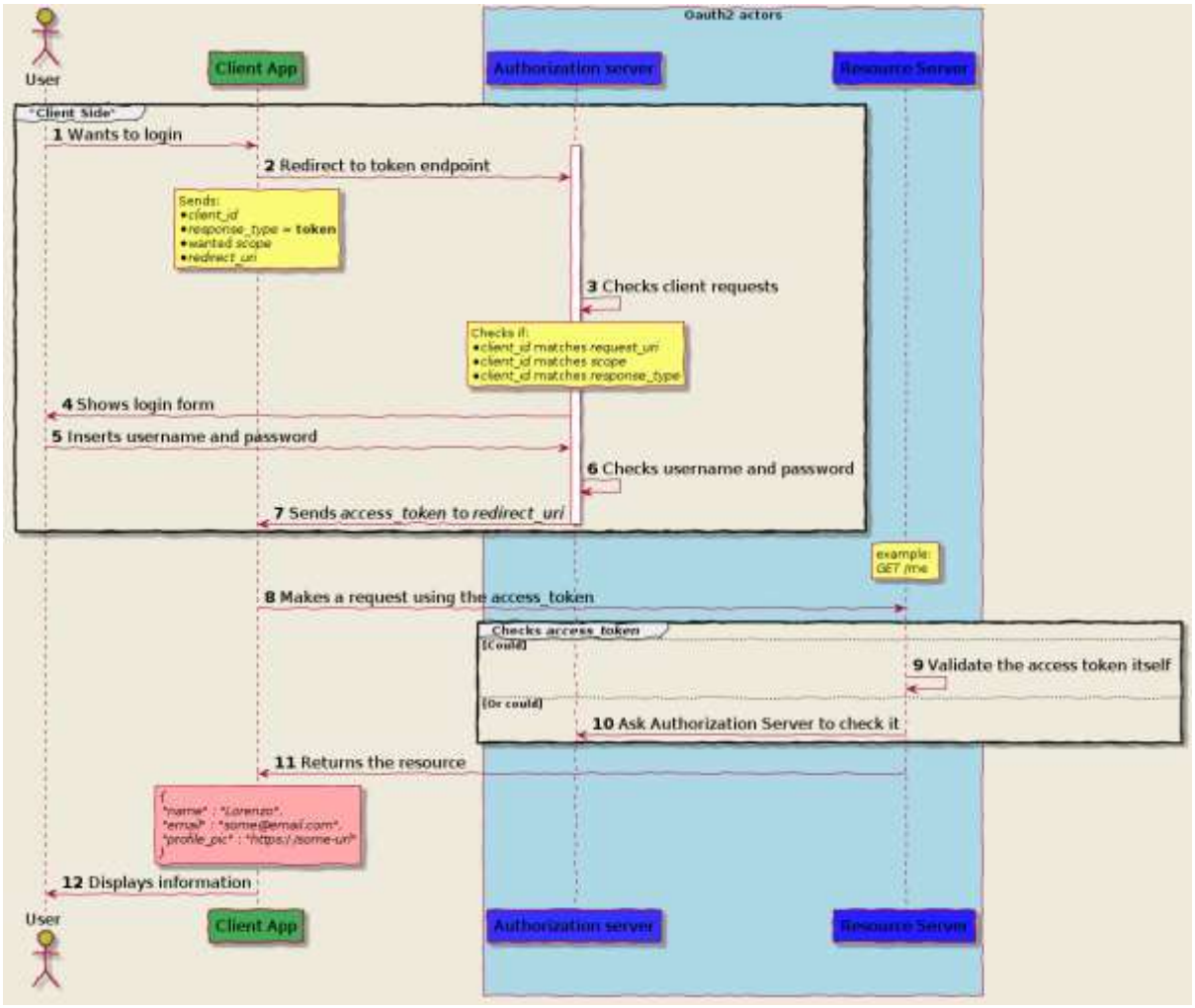
زمانی که کاربر می‌خواهد به این سایت لاگین کند سایت نام کاربری و کلمه عبور او را (در صورتی که قبلا به گوگل لاگین کرده باشد نیازی نیست و وارد سایت می‌شود) به گوگل می‌فرستد و پس از تایید هویت در صورت صحیح بودن اجازه می‌دهد کاربر به آسانی وارد سایت مقصد شود.

تصویر زیر نمایانگر این روش می‌باشد



.

معماری احراز هویت به روش oauth2:



معماری احراز هویت به روش oauth2

شرح روش

بیاید به همراه جزییات نحوه ی کار این روش لاگین رو در ادامه بررسی کنیم:

۱- کاربر می خواهد لاگین کند

یک سناریوی کلاسیک برای این روش توسط کاربر در مرورگر اجرا می شه. کاربر روی دکمه ی «لاگین با OAuth» کلیک می کنه و درخواست کاربر توسط کلاینت (در این مثال سایت) به سمت سرور احراز هویت ارسال می شه.

۲- کاربر به سرور احراز هویت منتقل (redirect) می شود

کلاینت درخواست لاگین رو به سرور احراز هویت می فرستد. این request به صورت فرم HTTP redirect و اطلاعات آن با متد GET به این سرور ارسال می شود

پارامتر های ارسالی:

client_id: برای شناسایی و اطلاعات مربوط به اپلیکیشن کلاینت

redirect_uri: آدرسی که سرور احراز هویت در حین انتقال به آن کد احراز هویت را می فرستد (بعد از لاگین کاربر به همراه کد به این آدرس ریدایرکت می شود)

response_type: نوع پاسخ برگشتی سرور احراز هویت را مشخص می کند

scope: لیستی از بخش های اپلیکیشن که کاربر اجازه ی دسترسی به ان ها راخواهد داشت و کاربر درخواست لاگین را برای دسترسی به آن اطلاعات ارسال کرده است. (مثلا ایجاد مطلب یا خواندن پیام ها و ...) و این پارامتر برای سرور منابع جهت دسترسی دادن به اطلاعات لازم و مفید است. (تا در جاهایی که کاربر دسترسی ندارد ارور ۴۰۳ عدم دسرسی را بتواند ارسال کند)

state: پارامتری اختیاری که در هنگان برگشت به کلاینت بازگردانده می شود و برای بررسی اطلاعات استفاده می شود

۳- اعتبار سنجی درخواست لاگین (Request validation)

سرور احراز هویت بایستی تمامی پارامتر های لاگین را بررسی کند:

آیا کلاینتی با آی دی client_id وجود دارد؟ و آیا این کلاینت مجاز به ارسال request برای لاگین هست یا خیر؟

آیا کلاینت می تواند از redirect_uri استفاده کند؟ آیا با کلاینت مربوطه مرتبط است؟

آیا کلاینت اجازه ی درخواست برگشت با response_type را دارد؟

آیا کلاینت اجازه ی این استفاده از این نوع احراز هویت را برای دسترسی به scope مربوطه دارد؟

پیشاپیش ذخیره ی کلاینت های در دسترس برای بررسی و هندل کردن این موارد ضروری است. و البته نگه داری و مدیریت اطلاعات کلاینت ها از حیطه ی وظایف OAuth خارج است.

۴- فرم لاگین

سرور احراز هویت فرم لاگین را به کاربر نشان می دهد و کاربر نام کاربری و رمز عبور خود را برای ورود به برنامه وارد می کند (مرحله ی ۵ در عکس)

بعد از بررسی اطلاعات توسط سرور احراز هویت (مرحله ی ۶ در عکس) اجازه ی دسترسی به scope مشخص شده را به کاربر می دهد و request خارج از اون scope تعیین شده مجاز نیست (چون ممکنه در استفاده از اپلیکیشن محدود شده باشه)

۵- انتقال (redirect) به کلاینت با کد مجوز authorization_code

سرور احراز هویت کد authorization code را ایجاد می کند و آن را روی redirect_url تعیین شده به سمت کلاینت بر می گرداند.

تمامی این مراحل در سمت کلاینت اتفاق می افتد (روی مرورگر یا اپلیکیشن موبایل) و در تمامی این مراحل کلاینت باید با بک اند در ارتباط باشد.

یه نظر من این موارد یکی از مهم ترین جنبه های انتخاب روش مناسب OAuth 2.0 برای پروژه هاست که بیا نیاز هامون منطبق بشه.

۸ , ۹- اعتبار سنجی کد مجوز (authorization code)

در اینجا کلاینت باید سرور احراز هویت را برای بررسی کد دریافت شده فراخوانی کند. برای این کار ارسال پارامتر های زیر لازم است:

authorization_code: برای بررسی

client_id: این پارامتر در کنار مورد بعدی جهت اطمینان از صحت کلاینت مورد استفاده قرار می گیرد و این که از دزدیده نشدن توکن مطمئن بشیم

client_secret: به عنوان رمز عبور کلاینت است و به صورت امن در سمت کلاینت ذخیره می شود

سرور احراز هویت تمامی اطلاعات بالا را از جهت صحت داشتن و همخوانی بررسی می کند. بررسی کد مربوطه باید از جهات کامل بودن و خطر دار نبودن و منتقضی نبودن و تعلق داشتن به کلاینت بررسی شود.

۱۰- کلاینت access_token را دریافت می کند

سرور احراز هویت access_token را ایجاد کی کند و آن را به سمت کلاینت بر می گرداند. انواع مختلفی برای ایجاد توکن وجود دارد. این ها می توانند تاریخ انقضا داشته باشند یا رفرش شوند. عموما همراه access_token اطلاعات زیر نیز به کلاینت برگردانده می شود:

token_type: نوع توکن

expires_in: تاریخ منقضی شدن توکن

refresh_token: توکنی دیگر برای ایجاد دوباره ی توکن اصلی هنکام انقضای آن

۱۱- کلاینت از access_token استفاده می کند

از این مرحله به بعد کلاینت access_token رو دریافت کرده و از آن برای دسترسی و شناسایی توسط سرور منابع (همون جایی که اطلاعات رو از دیتابیس می گیره و به کلاینت بر می گردونه) مورد استفاده قرار می گیرد.

سرور منابع یک بخشی از وب اپلیکیشن سمت سرور ماست که فرقی نمی کنه که از چه نوعی باشه می تونه RESRT API یا SOAP یا هر چیز دیگه ای باشه ... متد درخواست تغییر access_token بستگی به نوع منابع تون داره. متد متداول استفاده از REST API هست (در حال حاظر) با بهره گیری از HTTP header:

Authorization: شامل نوع توکن و خود توکن می شود

۱۲ و ۱۳- اعتبار سنجی توکن

هر وقتی که سرور منابع درخواستی را دریافت می کند بایستی توکن از لحاظ اعتبار خود توکن و کلاینت آن اطمینان حاصل کند.

اعتبار سنجی توکن بسته به نوع ایجاد آن می تواند به روش های مختلفی صورت گیرد. این اعتبار سنجی می تواند در همان سرور منابع یا با فراخوانی سرور احراز هویت انجام شود.

به هر حال به معماری برنامه تون بستگی داره و در سمت کلاینت تفاوتی نشون داده نمی شه. اگر می خواید از روش OAuth برای احراز هویت کاربران استفاده کنید بهتره به نمونه های مربوط به نوع معماری برنامه تون مراجعه کنید تا انتخاب آگاهانه تر و کد های مناسب تری داشته باشید.

۱۴ و ۱۵- دسترسی و نمایش منابع اختصاصی

اطلاعات مخصوص با دسترسی محدود به کلاینت در هر بار اعتبار سنجی توکن توسط سرور منابع به کاربر نمایش داده خواهد شد.