Authentication

https://campus.barracuda.com/doc/96774493/

This articles provides the following topics:

Introduction

ECHOplatform's REST API uses the OAuth 2.0 protocol for authentication and authorization to protect our customer's data. The protocol supports multiple flows so users can be authenticated to the needs of the application. All requests are transmitted securely via TLS/SSL.

Once a Barracuda customer authorizes the application, the application can access the customer's owned data. The customer can authorize several permission scopes controlling how much data an application can access.

Barracuda supports OAuth 2.0 draft 10.

Client Profiles

Authentication involves requesting an access token from Barracuda's authentication server. Once an access token has been granted, it is used to request resources from the API server. Barracuda supports two profiles depending on the relationship of the client application.

If the client application is a customer of Barracuda, then they follow the client credentials profile.

Web Server

The web server profile fits those clients who are not Barracuda customers. Non-Barracuda customers should not record the Barracuda customer's credentials, but redirect the end-users to the Barracuda authentication server to handle authenticating and authorizing the client for that end-user.

Requesting an Access Token

The web server profile involves the application redirecting the end-user's agent to Barracuda's

authorization server.

After the end-user authenticates using their Barracuda username and password, the user is asked for authorization to the client application.

If allowed, the client application receives an authorization code, which the user can turn in for an access token and a refresh token.

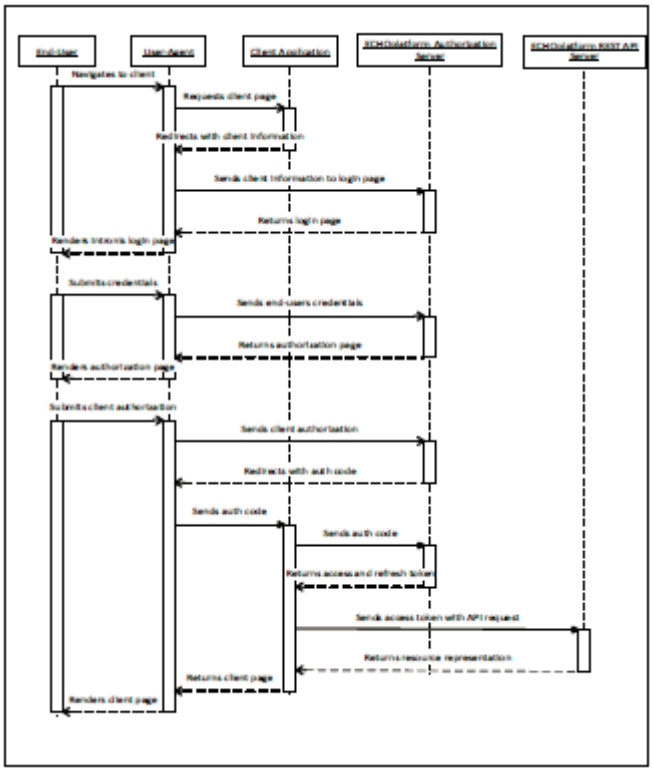

Figure 1. Web Server Profile – Authenticating from End-user.

The process works as follows:

(1) Redirect users who wish to authenticate to:

```
    https://auth.intronis.com/oauth2/authorize
?client_id={REGISTERED_CLIENT_ID}
&response_type=code
&redirect_uri={REGISTERED_REDIRECT_URI}
```

```
&scope={SCOPE}
```

(2) If the end-user grants authorization, they are redirected back to:

```
    https://{REGISTERED_REDIRECT_URI}/?code={AUTHORIZATION_CODE}
```

(3) The client server makes a request to:

```
POST /oauth2/token HTTP/1.1
Host: auth.intronis.com
Content-Type: application/x-www-form-urlencoded
client_id={REGISTERED_CLIENT_ID}&
client_secret={REGISTERED_CLIENT_SECRET}&
grant_type=authorization_code&
redirect_uri={REGISTERED_REDIRECT_URI}&
code={AUTHORIZATION_CODE}&
scope={SCOPE}
```

(4) The response is in JSON:

```
{
  "access_token": "{ACCESS_TOKEN}",
  "expires_in": "3600",
  "refresh_token": "{REFRESH_TOKEN}"
}
```

(5) The client saves this information for making API requests.

Refreshing an Access Token

Access tokens have a short lifetime while the refresh token has a long lifetime. After the access token expires, the client application can turn in the refresh token for a new access token without having to involve the end-user.

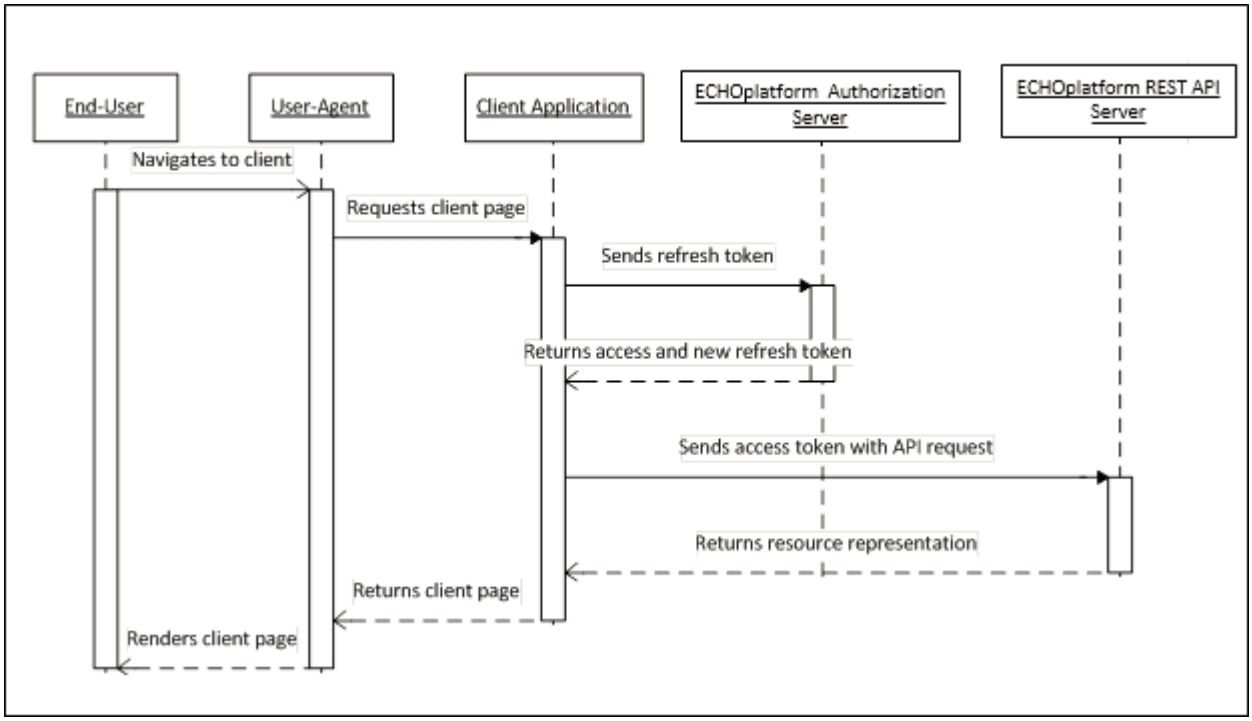

Figure 2. Web Server Profile – Authenticating from Refresh Token.

The process works as follows:

(1) The client server makes a request to:

```
POST /oauth2/token HTTP/1.1
Host: auth.intronis.com
Content-Type: application/x-www-form-urlencoded
client_id={REGISTERED_CLIENT_ID}&
client_secret={REGISTERED_CLIENT_SECRET}&
grant_type=refresh_token&
refresh_token={REFRESH_TOKEN}&
```

```
scope={SCOPE}
```

(2) The response is in JSON:

```
{
  "access_token": "{ACCESS_TOKEN}",
  "expires_in": "3600",
  "refresh_token": "{REFRESH_TOKEN}"
}
(3) The client saves this information for making API requests.
```

Client Credentials

The client credentials profile fits those clients who are Barracuda customers. They should not need to send end-users to barracuda to authenticate and authorize the client because the client owns that customer's information.

Requesting an Access Token

The client application can send their client credentials for an access token for their Barracuda customer without having to involve the end-user.

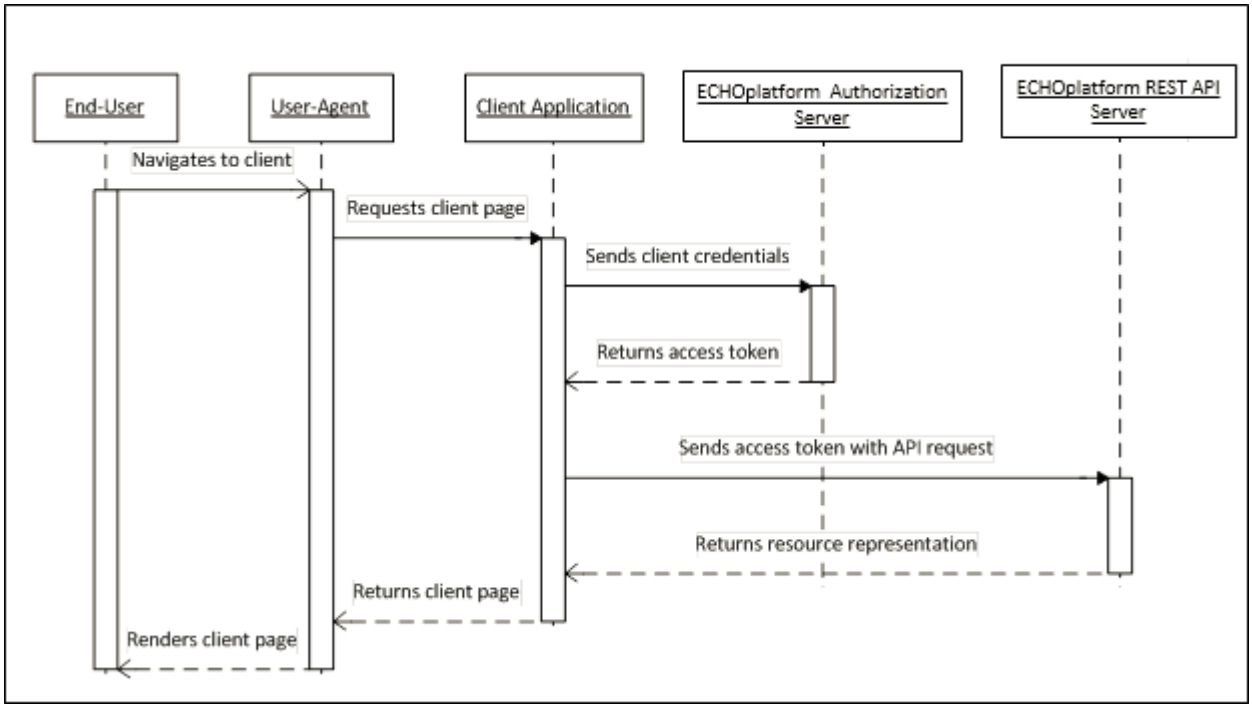

Figure 3. Client Credentials – Authenticating.

The process works as follows:

(1) The client server makes a request to:

```
POST /oauth2/token HTTP/1.1
Host: auth.intronis.com
Content-Type: application/x-www-form-urlencoded
client_id={REGISTERED_CLIENT_ID}&
client_secret={REGISTERED_CLIENT_SECRET}&
grant_type=none&
scope={SCOPE}

(2)The response is in JSON:

{
  "access_token": "{ACCESS_TOKEN}",
  "expires_in": "3600"
}

(3)The client saves this information for making API requests.
```

Scope

Resources are protected by various scopes and the end-user must authorize the client for a particular scope. For example, the end-user might only to want grant the client a read-only scope. The client must send the scopes to be authorized when requesting an access token. The client can send multiple scopes that are space delimited (separated by a '+' when formatted in application/x-www-form-urlencoded).

Errors

In addition to the OAuth2 required ‘error’ parameter, Barracuda returns additional information for developers in the optional ‘error_description’ parameter.

Figures