API Documentatie Ledensite.com Versie 1.0

De API bestaat uit een data download gedeelte en een single-sign on gedeelte. Het data download gedeelte is nog in ontwikkeling.

Single Sign-on

Het is mogelijk om met de accountgegevens van Ledensite.com in te loggen op andere sites. Zo kan het bijvoorbeeld dat iemand ingelogd is bij Ledensite.com en dan wil doorklikken naar een ledenforum. Als het lid dit doet moet het in de meeste gevallen met nieuwe gegevens inloggen. Met Ledensite.com hoeft dat niet meer. De inloggegevens kunnen worden meegestuurd naar de andere site, zodat de andere site meteen weet dat dit een ge-authentiseert lid is.

Om te beginnen heb je dan wel een API key nodig. Deze is te vinden in het organizer gedeelte van de site.

Inschrijven op afstand

De remote website stuurt een API request met login en password naar deze url:

http://mijnclub.ledensite.com/api/authenticate/[api key]?email=[user's email]&password=[user's password]

En krijgt een xml bestand terug (als de api key en de credentials kloppen):

```
<?xml version="1.0"?>
<ledensite>
  <version>0.4</version>
  <authentication>
    <status>OK</status>
    <name>Test User</name>
    <email>test@user.com</email>
    <user_id>100</user_id>
    <session_token>b2ef6ae88c1f8c147787e50b508500a89b4e8d2a</session_
token>
    <session_token_valid_until>2012-01-01 12:00:00
+0100</session_token_valid_until>
  </authentication>
</ledensite>
```

Deze kan dus zowel gebruikt worden om te authenticeren voor gebruik van de externe site als om in te loggen op Ledensite (door de session_token te gebruiken, hier onder uitgelegd.)

Vernieuwen of starten van een sessie

Als de deelnemer enige tijd op de externe site blijft zonder naar Ledensite te gaan zal de session_token op een gegeven moment verlopen zijn. De deelnemer is dus nog wel ingelogd op de externe site, maar de redirect werkt niet meer. Hier voor gebruik je deze url:

http://mijnclub.ledensite.com/api/initialize_session/[api key]?email=[user's email]

De reden waarom dit een aparte functie is, is omdat de externe site (als het goed is) het password van de user niet opslaat, en je wil niet de user dwingen om nogmaals in te loggen voordat hij ge-redirect kan worden. Dus voordat je redirect kun je de bovenstaande call uitvoeren om zeker te zijn dat de token niet verlopen is. Dit geeft het zelfde xml bestand terug als bij punt 1 (weer mits de api key klopt) met de nieuwe token en expiration date.

Deze functie kan ook gebruikt worden als de externe site een eigen authenticatie systeem heeft (en dus niet de user via voorbeeld 1 authentiseert.)

Valideren van de sessie

Als de externe site een session_token wil valideren kan dat met deze url (dit is nodig als er een deelnemer van ledensite naar de externe site geredirect wordt.)

http://mijnclub.ledensite.com/api/validate_session/[api key]?session_token=[session_token]

Als de api_key en session_token kloppen en niet verlopen zijn resulteert dit in dezelfde xml als bij punt 1 en 2. Normaliter gebruik je deze functie nadat een user van Ledensite is doorgestuurd naar de externe site met een session_token.

Valideren kan maar 1x, daarna is de token niet meer geldig. Dit om hijacking van sessies te voorkomen.

Expireren van de sessie

De deelnemer klikt op de externe site op "uitloggen" en de externe site wil aan Ledensite laten weten dat de sessie beëindigd is. Dat gaat zo:

http://mijnclub.ledensite.com/api/expire_session/[api key]?email=[user's email]

Dit genereert de volgende output (OK betekent "sessie is beëindigd"):

```
<?xml version="1.0"?>
<ledensite>
  <version>0.4</version>
  <session_expiration>
    <status>OK</status>
    <name>Slinger Jansen-Roijackers</name>
    <email>s.jansen@cs.uu.nl</email>
    <user_id>51</user_id>
  </session_expiration>
</ledensite>
```

Dat zijn alle calls die de externe site naar Ledensite.com kan doen.

Stap voor Stap Workflows

De vier workflows die kunnen worden doorlopen zijn: externe site redirect naar ledensite, ledensite redirect naar externe site, deelnemer logt uit op ledensite, deelnemer logt uit op de externe site.

1. ***De externe site redirect een deelnemer naar Ledensite.***

   Eerst vraagt de externe site een session_token aan via api call 1 of 2 in de lijst boven. Daarna redirect de externe site de deelnemer naar dit adres:

   http://mijnclub.ledensite.com/api/login/[session_token]

   Als de api key en session_token kloppen wordt de deelnemer ingelogd, net alsof hij zijn email en wachtwoord ingevuld had. We vragen hier geen api_key omdat de user deze redirect url (even) te zien krijgt. De api key mag natuurlijk nooit bekend worden bij derden.

   De session token kan maar 1x gebruikt worden op deze manier. Dit omdat als de session_token eenmaal in de url 'publiek' weergegeven is deze gevoelig is voor hijacking. Als later nogmaals geredirect moet worden zal de session_token vernieuwd moeten worden (call 2 bovenaan.)

2. ***Ledensite redirect een user naar de externe site.***

   De organisatie stelt eerst in hun configuratie de 'API login url' in in hun configuratie. Daarna verwerken zij ergens in hun site de volgende link:

   http://mijnclub.ledensite.com/remote_login

   De deelnemer wordt dan ge-redirect naar de remote_login_url die de organisatie heeft ingesteld met als parameter de session_token, bijvoorbeeld:

   http://my.site.com/login.php?session_token=abcdef123456

   Daarna valideert de externe site de token via punt 3 bovenaan. (nogmaals: na validatie zal de token niet meer geldig zijn om hijacking te voorkomen.)

3. ***Klant logt uit via de externe site.***

   De externe site handelt zelf zijn eigen sessie af en stuurt dan een request naar ledensite om de sessie te expireren (beschreven in punt 4 van de calls bovenaan)

   http://mijnclub.ledensite.com/api/expire_session/[api key]?email=[user's email]

   De session token wordt nu uit het systeem verwijderd en de browser sessie van de klant zal niet meer geldig zijn als hij terugkeert naar ledensite, totdat hij weer handmatig inlogt of totdat de externe site een nieuwe token aanvraagt en de klant daarmee naar Ledensite redirect.

4. ***Klant logt uit op Ledensite***

   De organisatie stelt in zijn configuratie een 'API loguit url' in waarmee het beëindigen van de sessie kan worden gecommuniceerd.

   Ledensite handelt zijn eigen sessie af en vervolgens wordt een request gestuurd naar de remote site die er als volgt uitziet:

   http://my.site.com/logout.php?session_token=abcdef123456

Specificatie van de Beveiliging

Bij deze de specificaties van de encryptie van de API calls. Bij alle calls die een api_key vragen wordt nu ook een validation_hash verwacht. Ik heb hiervoor voorlopig SHA2 gebruikt (default 256 bits). Deze wordt als volgt berekend:

1. Authenticate:

   Digest::SHA2.hexdigest(['test_key', 'test@domain.com', 'password', 'test_salt'].join)

2. Initialize:

   Digest::SHA2.hexdigest(['test_key', 'test@domain.com', 'test_salt'].join)

3. Validate:

   Digest::SHA2.hexdigest(['test_key', 'session_token', 'test_salt'].join)

4. Expire:

   Digest::SHA2.hexdigest(['test_key', 'test@domain.com', 'test_salt'].join)

Dus altijd eerst de api_key, gevolgd door de parameters, gevolgd door de salt string. Overal is 'test_key' de api key, 'test_salt' de api salt (ingesteld in de organisatie configuratie) en 'session_token' de huidige session_token van de user.

De validation hash wordt gewoon met de rest van de parameters meegestuurd, b.v.:

`http://vaan.ledensite.com/api/authenticate/[api key]?email=[email]&password=[password]&validation_hash=abcdef123456.....`