Sicurezza Informatica

Il Protocollo OAuth

Anno Accademico 2010/2011

Riccardo Queri

Luca Mancini

Cos'è OAuth?

* OAuth (Open Authorization) è un protocollo open che permette ad Applicazioni di chiamare in modo sicuro ed autorizzato API messe a disposizione da un Servizio Web.

* Permette di accedere alle risorse protette di un utente senza che esso debba condividere le sue credenziali (username e password).

Prima di OAuth

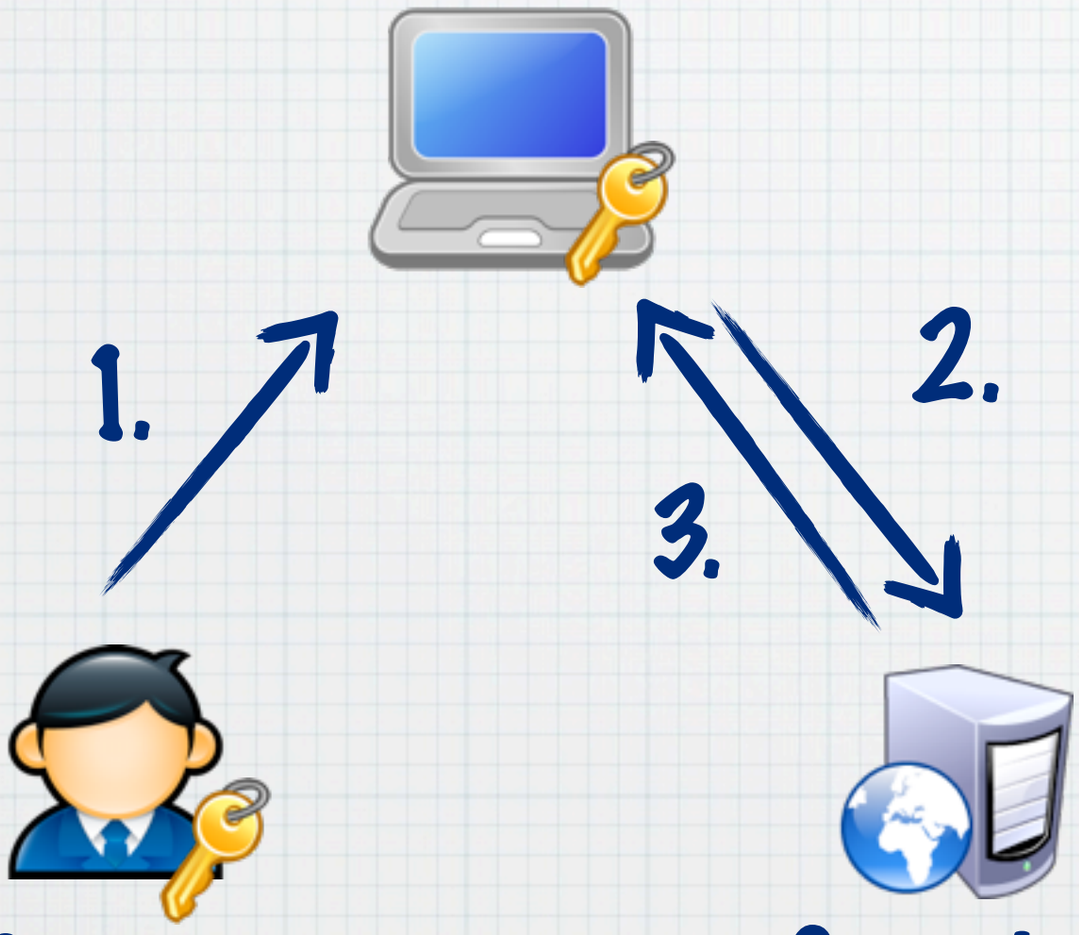
* Esistevano protocolli proprietari come Google AuthSub, AOL OpenAuth, Yahoo BBAuth, Flickr API.
* OAuth è stato ideato da Blaine Cook nel 2006, mentre lavorava a Twitter OpenID
* Nel luglio 2007 esce la prima verisone ufficiale.




Terminologia
(RFC 5849)

* Service Provider
* Resource Owner o Utente
* Consumer o Client
* Consumer Key e Consumer Secret
* Request Token
* Access Token e Token Secret

Client Server vs OAuth (1)

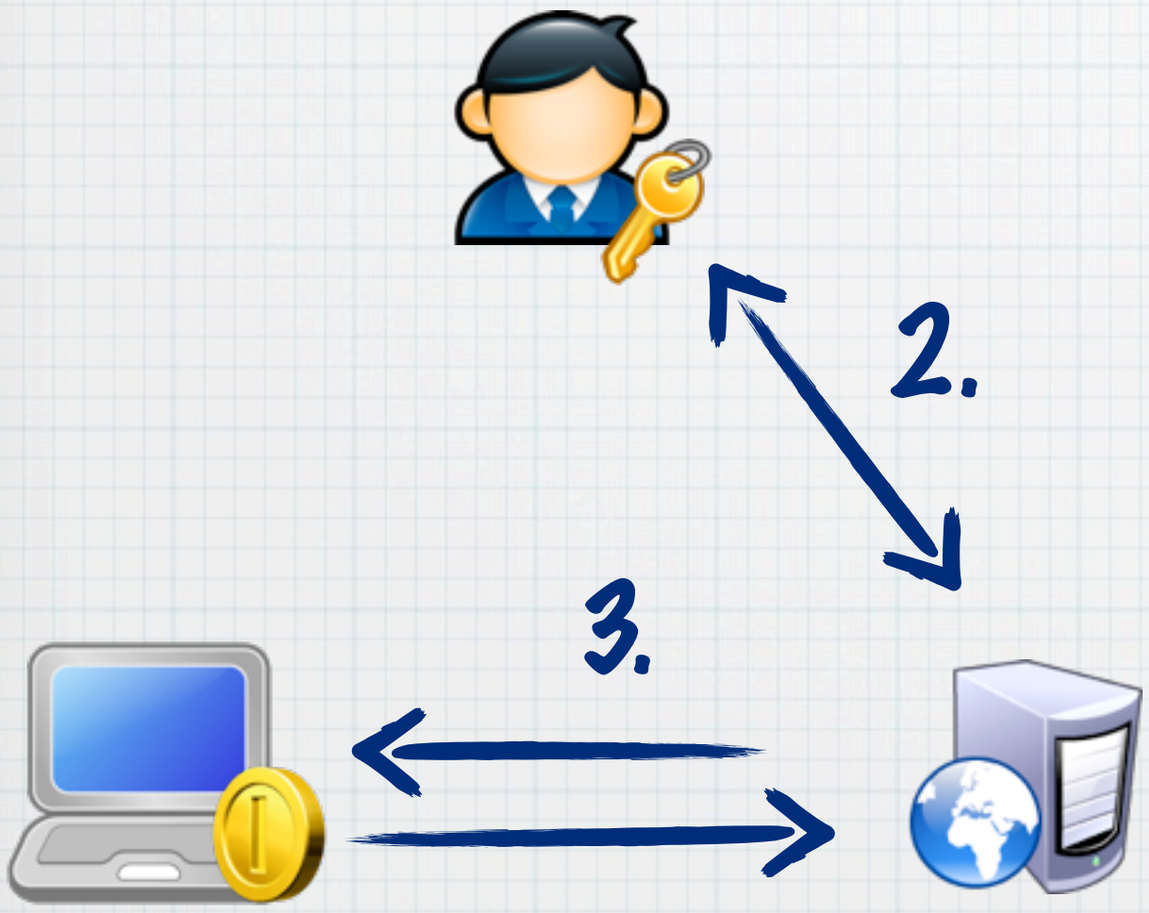


L' utente fornisce username e password al client (1.) che le usa (2.) per accedere alle sue risorse nel server (3.).

↓

Problema di sicurezza!

Client Server vs OAuth (1)



Il client chiede l'accesso alle risorse in nome dell'utente (1.). che lo autorizza loggandosi al SP (2.). Il SP restituisce al client un token per accedere ai dati richiesti.

↓

Il Client NON conosce le credenziali utente!

Fasi del protocollo
-Registrazione-

Client

Il Consumer si iscrive come sviluppatore al Service Provider.

S.P.

Fornisce una coppia di chiavi:

* Consumer Key
* Consumer Secret (Shared Secret)

Fasi del protocollo
-Autenticazione-

E' divisa in 3 step, in ognuno dei quali il SP espone un end-point:

* Richiesta credenziali temporanee → Temporary Credential Request
* Redirezione dell'utente → Resource Owner Authorization
* Richiesta del Token → Token Request

Autenticazione
- HTTP -

* Acronimo di Hyper Text Transfer Protocol, è un protocollo a livello applicazione (7 ISO/OSI)
* E' il principale metodo di trasferimento per informazioni nel web
* Meccanismo Richiesta/Risposta (client/server), con due tipi di messaggi differenti.

Autenticazione
- HTTP -

* Messaggio di Richiesta
  * Request Line (GET/POST, URI)
  * Header (Host, User Agent, Authorization)
    * Body
* Messaggio di Risposta
  * Status Line
  * Header
  * Body

Autenticazione
-Credenziali Temporanee-

* Sono usate per identificare univocamente la richiesta di accesso

```
POST/ https://Temporary.Credential.Request.aspx
oauth_consumer_key= "jd83jd92dhsh93js",
oauth_signature_method= "PLAINTEXT",
oauth_callback= "http%3A%2F%2Fclient.net%",
oauth_signature= "ja893SD9%26"
```

* Se la richiesta è valida il server restituisce le credenziali temporanee (ID e shared secret)

Autenticazione
-Autorizzazione dell'utente-

* Il client indirizza, attraverso l'User Agent, l'utente al Resource Owner Endpoint (server) per autorizzare la richiesta
* Https://GET/authorize_access?
  oauth_token=hdk48Djdsa HTTP/1.1
* L'utente dovrà inserire le sue credenziali
* Il Server ritorna alla callback URI un codice di verifica e il token temporaneo

Un prodotto desidera connettersi al tuo account

Il prodotto **Droid Hattrick Manager** di **lucky88** desidera connettersi al tuo account in Hattrick con i seguenti permessi:

- **Accesso in sola lettura**

Utente:

lucky88

Password:

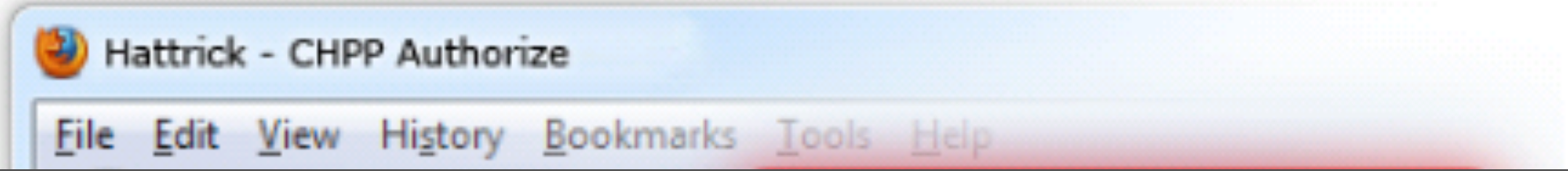
••••••••

Il tuo utente e la password sono usati solamente per verificare la tua identità qui in Hattrick ma non vengono passati al prodotto.

Nega o Consenti

A riguardo di phishing

Non dovresti fornire i tuoi login e/o password a nessuno all'infuori di Hattrick. Prima di effettuare il login assicurati che questo non sia un sito fasullo verificando nella barra indirizzi del tuo browser che ti trovi su chpp.hattrick.org.



Autenticazione
-Autorizzazione dell'utente-

* Il client indirizza, attraverso l'User Agent, l'utente al Resource Owner Endpoint (server) per autorizzare la richiesta
* Https://GET/authorize_access?
  oauth_token=hdk48Djdsa HTTP/1.1
* L'utente dovrà inserire le sue credenziali
* Il Server ritorna alla callback URI un codice di verifica e il token temporaneo

Autenticazione
-Rilascio Credenziali-

* Il Client effettua una richiesta per ottenere le credenziali autenticate:

```
POST/ https://Credential.Request.aspx
oauth_consumer_key= "jd83jd92dhsh93js",
oauth_token="hdk48Djdsa",
oauth_signature_method="PLAINTEXT",
oauth_verifier="473f82d3",
oauth_signature= "ja893SD9%26"
```

* Il Server, se la richiesta è valida, rilascia un Token e un Token Secret per accedere alle risorse

Richieste Autenticate
-Costruzione della richiesta-

* Il Client attribuisce valori ai seguenti parametri:
  * oauth_consumer_key
  * oauth_token
  * oauth_signature_method
  * oauth_timestamp
  * oauth_nonce

Richieste Autenticate
-Firma della richiesta-

* Tramite i valori dei parametri precedenti, viene costruita una stringa che sarà l'input degli algoritmi di firma:
  * HMAC-SHA1 Chiave simmetrica
  * RSA-SHA1 Chiave pubblica/privata
  * PLAIN-TEXT Non firmato
* L' output di questi algoritmi sarà il valore del parametro "oauth_signature"
* Viene costruita la richiesta http POST

Richieste Autenticate
-Verifica-

* Ricevuta la richiesta il Server deve validarla:
  * Ricalcola e verifica la firma
  * Verifica la combinazione nonce-timestamp-token
  * Verifica la legittimità delle richieste rispetto alle autorizzazioni del token
* Se la richiesta è valida concede al client le risorse.

Considerazioni Finali

* GARANTISCE integrità e autenticità
* NON GARANTISCE confidenzialità
* Uso di SSL o TLS a livello trasporto
* Spoofing di falsi Server
* Phishing Attacks
* Differenziazione sui diritti del token
* Evitare Automatic Processing of Repeat Authorizations

GRAZIE DELL'ATTENZIONE!