
IDSECCONF 2010
INDONESIAN SECURITY CONFERENCE

Open Authorization :

OAuth for credentials security in REST API access

IDSECCONF 2010
INDONESIAN SECURITY CONFERENCE

Panggi Libersa Jasri Akadol

Web : http://www.opensecuritylab.org
Twitter : @panggi

IDSECCONF 2010
INDONESIAN SECURITY CONFERENCE

Agenda

- Web 2.0 and Data
- OAuth usage
- Useful resources

Web 2.0 and Data

Web 2.0

= Your Data

Different service = Different data

What if
you need to use your data that stored in another service provider's server ?

Yup , just take it ☺

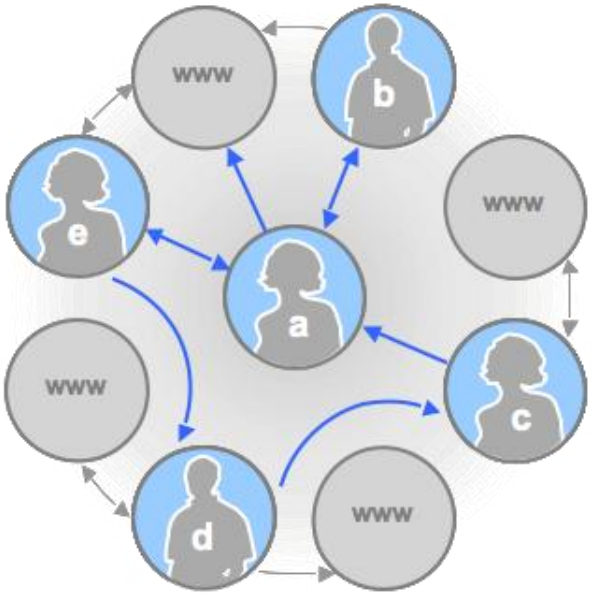

But How ??

OAUTH

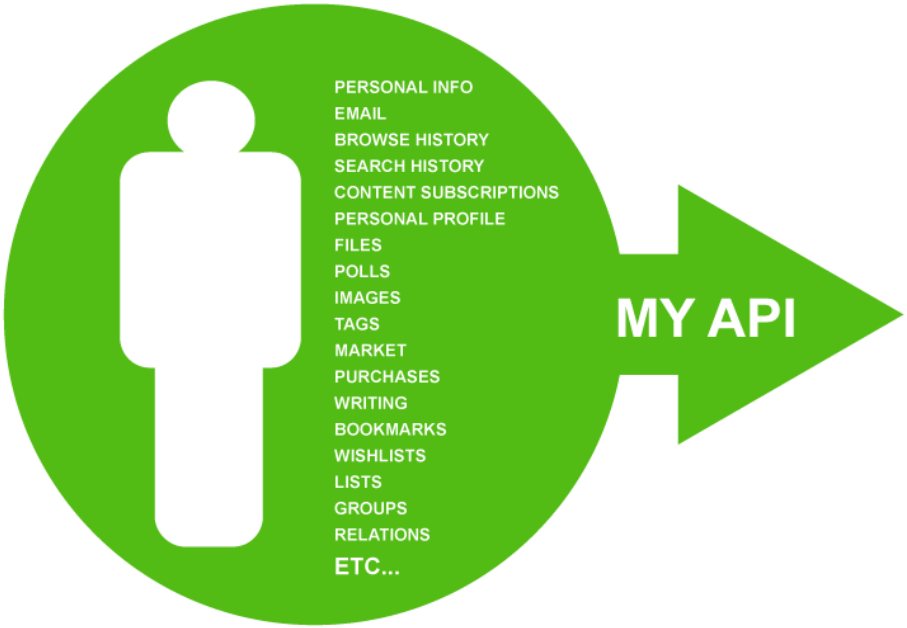
OK .. Enough with the Cute creatures :-p

Let’s dive into technical things

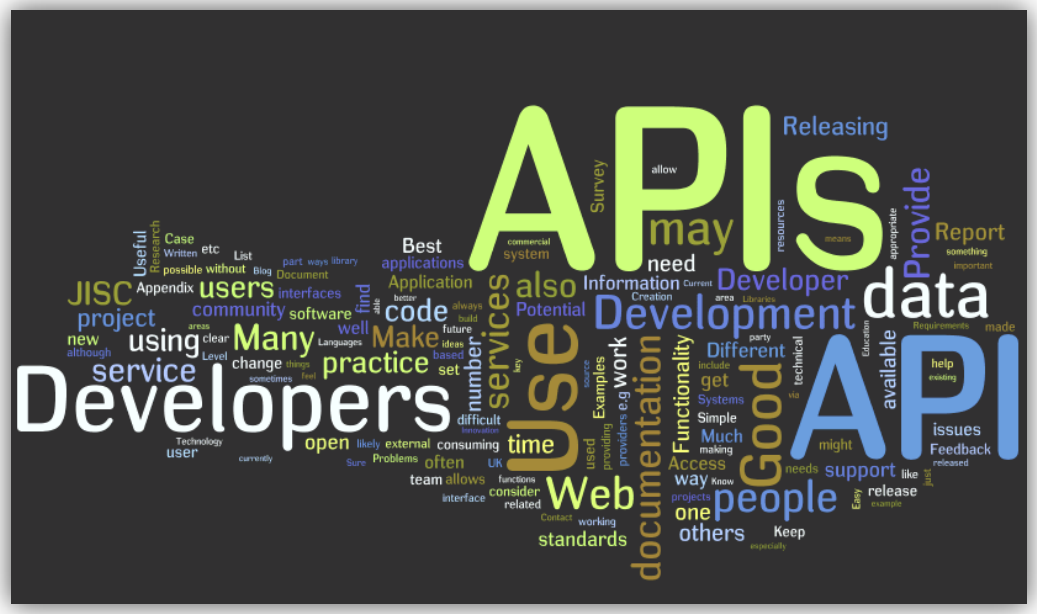
Once again.. How ?



Using API (Application Programming Interface)

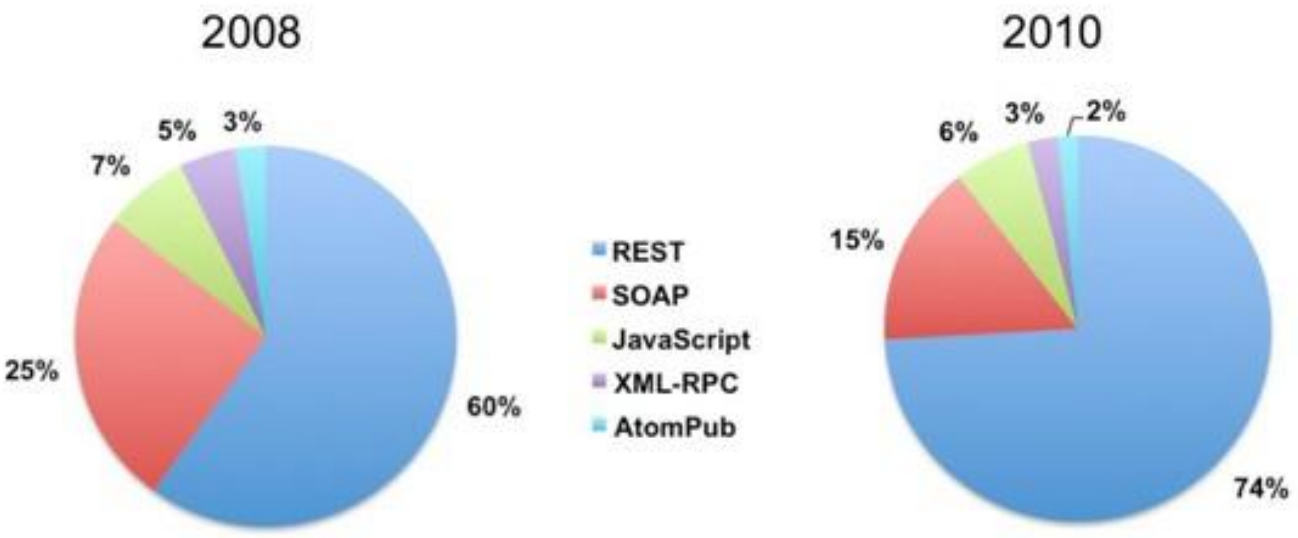


An **application programming interface** (**API**)
is an interface implemented by a software program that enables it to interact with other software. It facilitates interaction between different software programs similar to the way the user interface facilitates interaction between humans and computers. (via http://en.wikipedia.org/wiki/API)



IDSECCONF 2010
INDONESIAN SECURITY CONFERENCE

REST vs. SOAP: Simplicity wins again



Distribution of API protocols and styles

Based on directory of 2,000 web APIs listed at ProgrammableWeb, May 2010

REST Representational State Transfer

- Provide every resource with a unique ID, for example, a URI
- Link resources with each other, establishing relationships among resources
- Use standard methods (HTTP, media types, XML)
- Resources can have multiple representations that reflect different application states
- The communication should be stateless using the HTTP

URL

http://api.twitter.com/*version*/statuses/home_timeline.*format*

Supported formats

json, xml, rss, atom

Supported request methods

GET

Requires Authentication

true About authentication »

Rate Limited

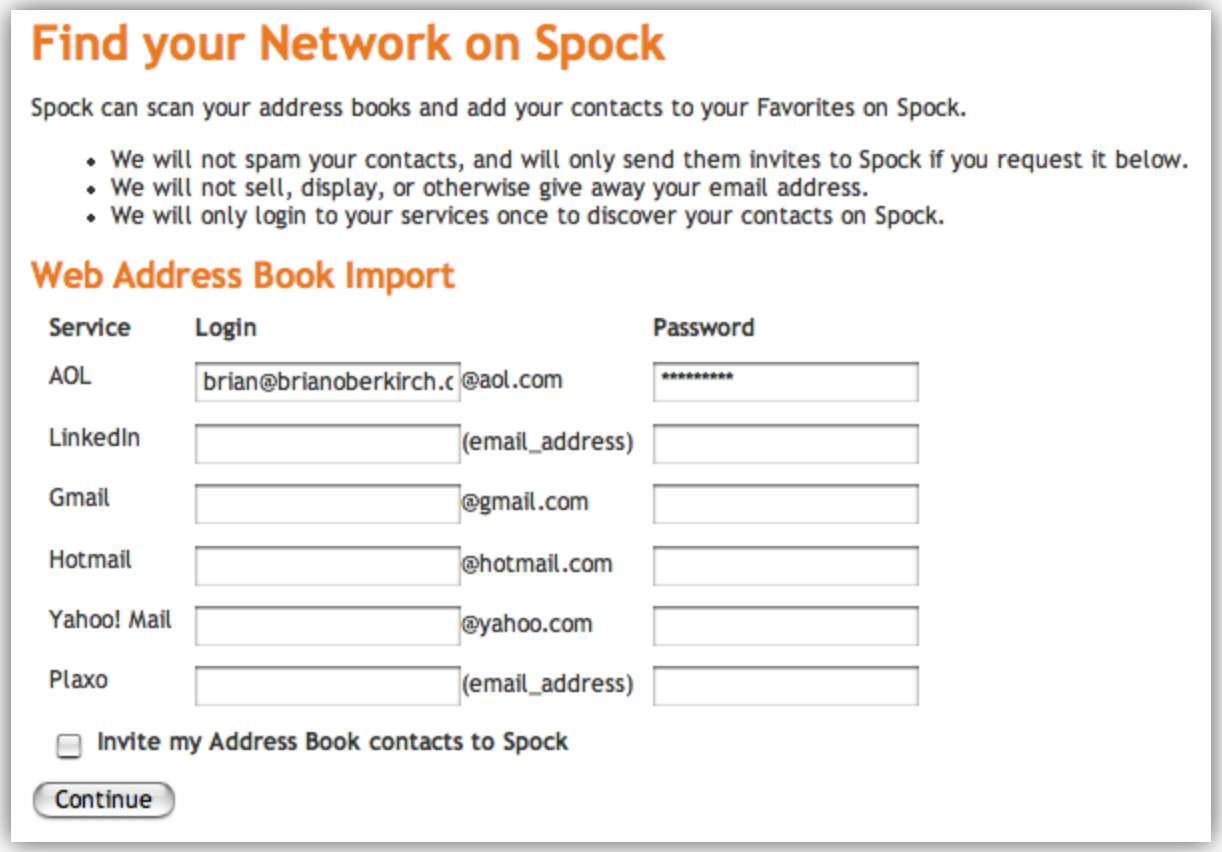
true About rate limiting »

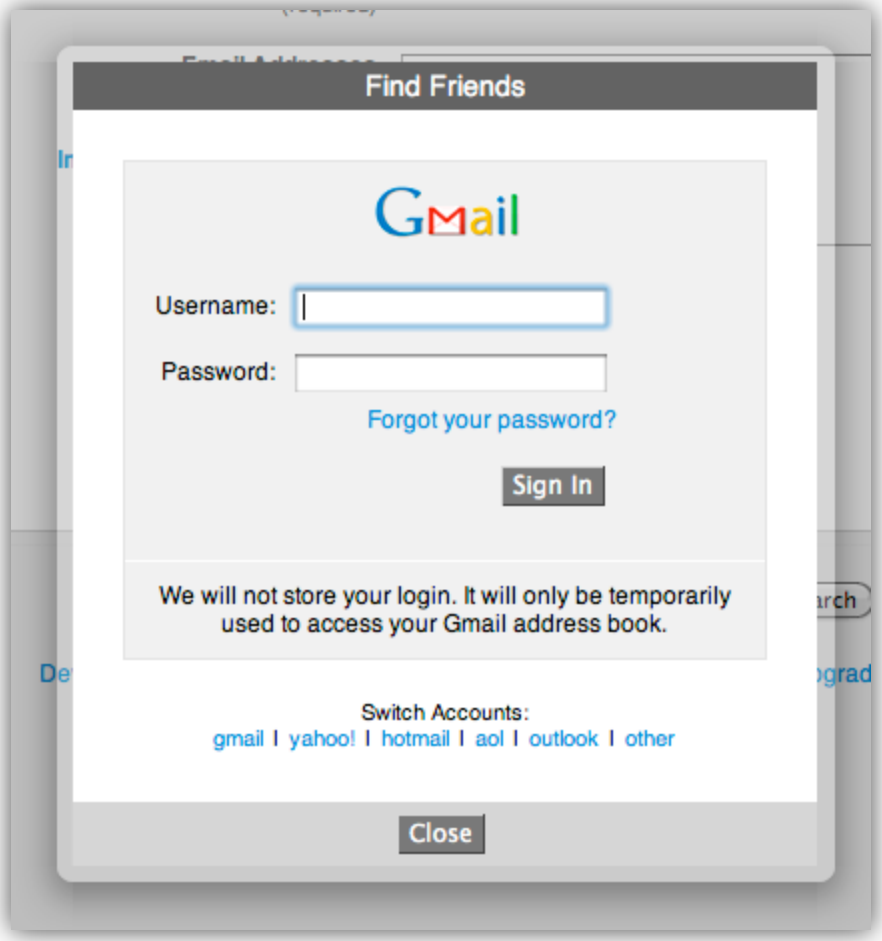
Parameters

Optional

- **since_id**
  Returns results with an ID greater than (that is, more recent than) the specified ID. There are limits to the number of Tweets which can be accessed through the API. If the limit of Tweets has occured since the since_id, the since_id will be forced to the oldest ID available.
  http://api.twitter.com/1/statuses/home_timeline.json?since_id=12345

Accessing API





IDSECCONF 2010
INDONESIAN SECURITY CONFERENCE

Brightkite & Twitter

Twitter username

Your Twitter username.

Twitter password

The password you use to log into Twitter. We need this so we can post updates on your behalf.

Also post to Twitter

- [ ] When I post a note.
- [ ] When I post a photo.
- [ ] When I check in at a place, post my checkin to Twitter.
- [ ] When I check in at a place, set my Twitter location accordingly (NEW).

Choose which Brightkite actions you want to post to your Twitter account. **Keep in mind that this might easily annoy your followers, so choose wisely.**

Save

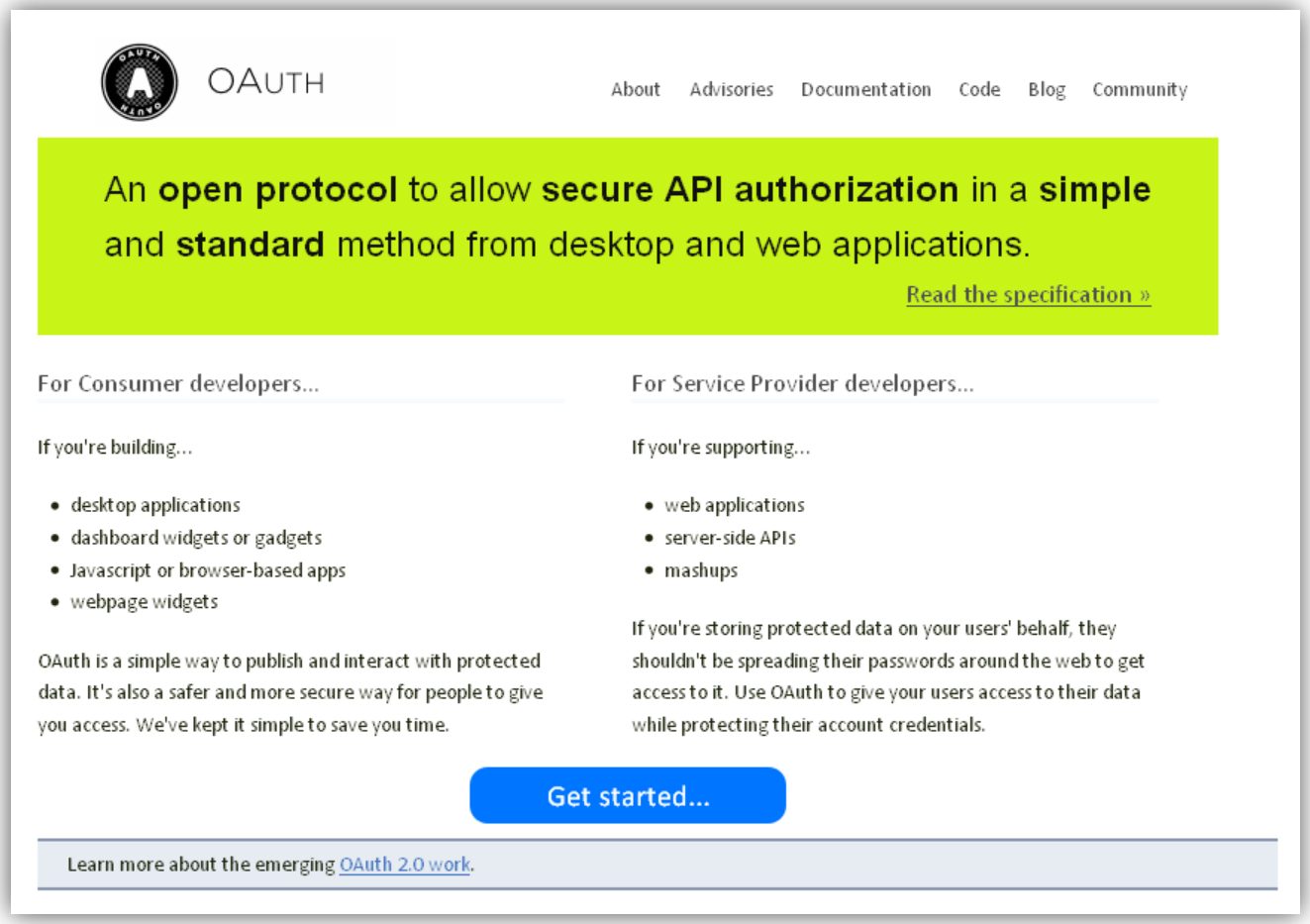
What's on your mind ?

"**Giving your email account password to a social network site so** they can look up your friends is the same thing as going to dinner and **giving your ATM card and PIN code to the waiter when it's time to pay.**"

- oauth.net

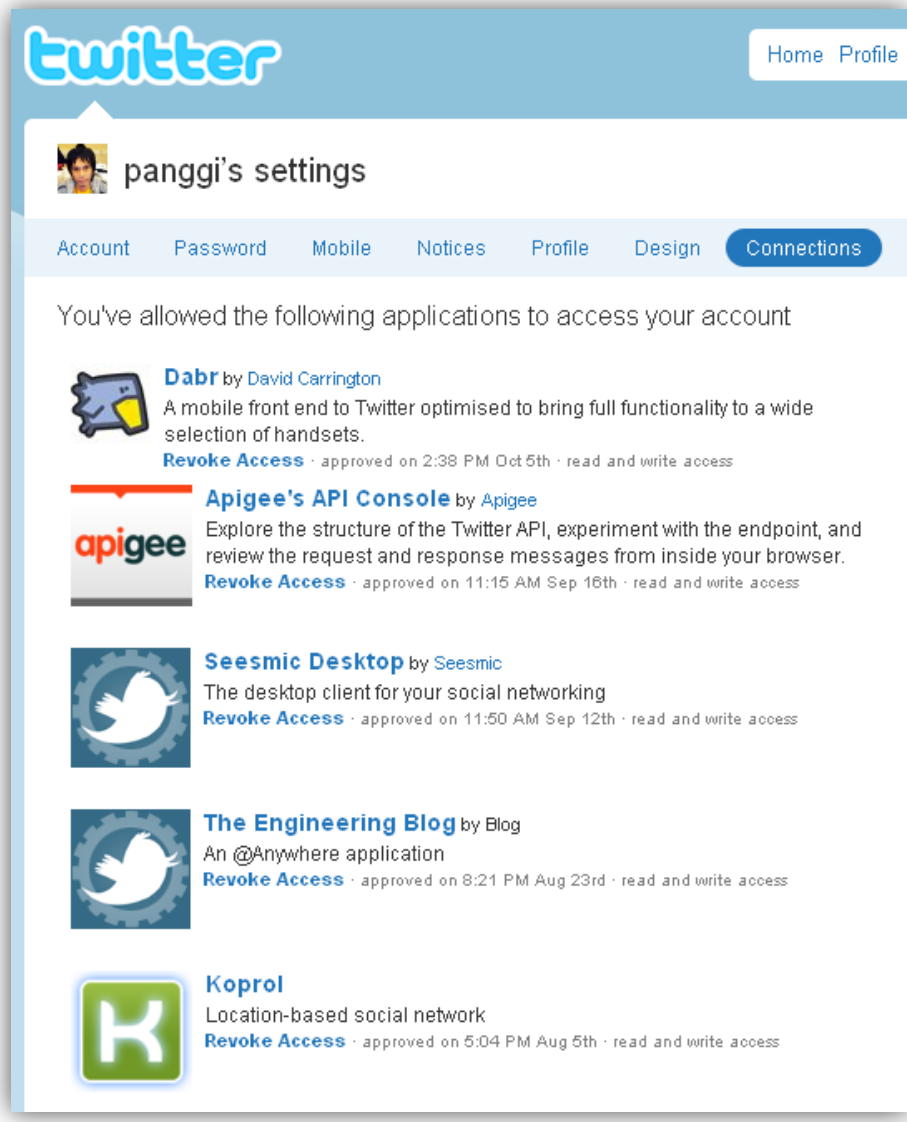
we need an easy,
user-friendly standard
for third party api security

OAuth usage



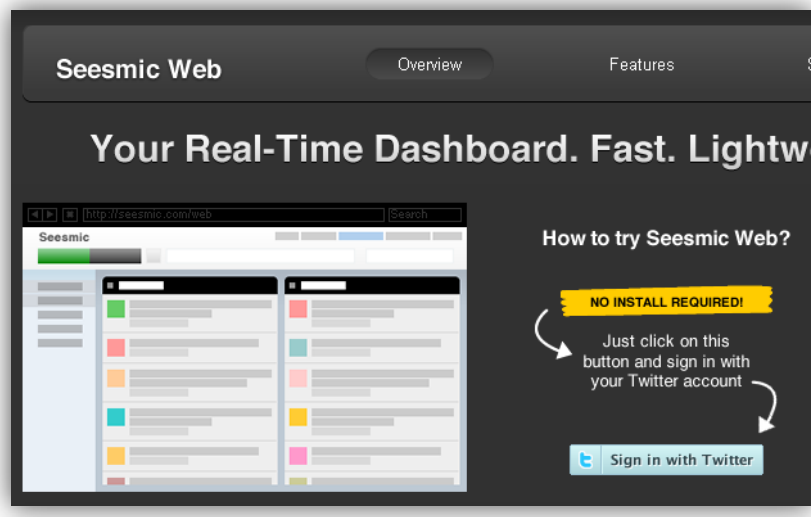
OAuth puts the user back in control

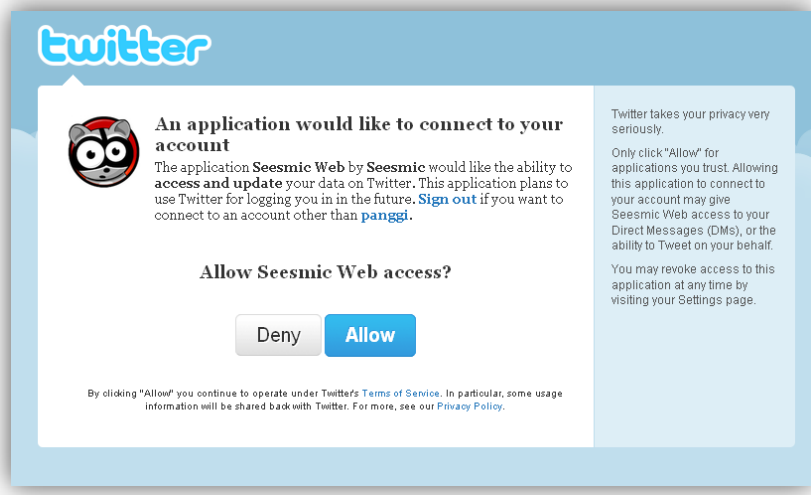
You choose who you share your data with

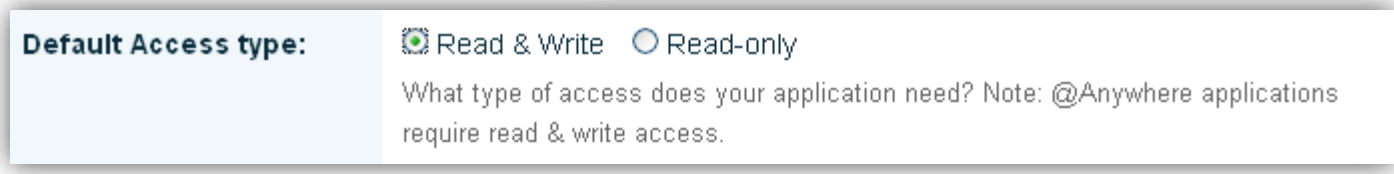


OAuth is secure

No need to give Username and Password







Big Name Adoption

Google
OpenSocial
MySpace
SmugMug

Yahoo!
Netflix
twitter
GetSatisfaction *and more...*

OAuth

Love triangle

End user
(Resource Owner)

twitter

Service provider

Consumer

IDSECCONF 2010
INDONESIAN SECURITY CONFERENCE

OAuth

Protected resources
are exposed by service providers
and used by consumer applications
on behalf of users

OAuth

My Twitter Status
Is exposed by Twitter
And used by Seesmic
On my behalf

OAuth Terminology

- Provider is the application that exposes the secure API and user's identity
- Consumer is the application that is written against the Provider's API, intended for Provider's users.
- Users or resource owners are registered users of the Provider
- Consumer Key is an identifier for the consumer
- Consumer Secret is a shared-secret between the provider and the consumer
- Signature Methods are encryption methods used by OAuth communication. Methods suggested are PLAINTEXT, RSA-SHA1 and HMAC-SHA1
- OAuth Endpoints are endpoints exposed by the provider to facilitate OAuth dance
- Callback URL is an endpoint at the Consumer that is invoked by the Provider once the user authorizes the Consumer. If none, the value is oob, or Out-of-Band

Tokens

- Request Token
  - Short lived identifiers which start the handshake
  - Must be converted to Access Token in order to gain access to a user's resources

- Access Token
  - Long lived identifiers that are tied to the user's identity
  - Are used to access a user's resources (data) at the Provider on behalf of the user

Endpoints

- Get request token
- Authorize token
- Get access token

Get Request Token

- The endpoint provides consumers to get an unauthorized request token by providing their consumer key and other parameters as a signed request
- The credentials can be passed via HTTP Header, POST body or GET QueryString
- The request includes an oauth_signature which is calculated by following the steps defined in the spec. Use libraries instead of writing your own signing implementations.
- The response has an unauthorized request token as well as a token secret, and a flag indicating if the callback was accepted.

Authorize Token

- The step authorizes an unauthorized request token retrieved via previous request.
- The endpoint takes the unauthorized request token – or the user can enter one manually if supported.
- The Authorize Token endpoint then redirects the user to the Provider's login page
- The user logs in, and is asked to authorize the consumer (and hence the request token)
- Once the user authenticates, and authorizes access to the consumer, the provider calls the callback URL provided earlier with a verifier code. This verifier code, along with other credentials is used to get an Access Token.

Get Access Token

- At this step, the now authorized request token is exchanged for an access token
- The access token acts as a user's credential for any further transactions
- The endpoint takes the request token and the verifier code returned via the callback, or manually if callback is not supported. The request is signed with consumer secret and the request token's secret.
- The Provider returns an access token and a token secret.
- The token secret is used to sign the requests along with the consumer secret.

Access User's Resources

- Now that the consumer has the access token, the user's resources can be requested via signed requests to the provider.
- The user should be able to unauthorize the consumer by revoking the access token.
- The access token has a time to live which is typically longer than the request token

Useful Resources

- http://tools.ietf.org/html/rfc5849
- http://oauth.net/code/
- http://hueniverse.com/oauth/
- http://code.google.com/p/oauth/
- http://opensecuritylab.org/tag/oauth