EXHIBIT 3

Login Process Last modified Thursday, May 7, 2009 at 9:14am by Luke Shepar

Cookies

There are three critical cookies for use in the login process:

c_user
"Current user". The user ID of the logged in user. Only set when the user is actually logged in.

xs
The hashed login session key. The xs cookie is the secret key that authenticates the user. It rotates every twenty minutes, and has to match the key on the other side.

sid
The session id of the active session. This cookie is only set if persistent login is used; otherwise, the cookie is unset and it corresponds to session 0.

next_path
if you try to access a protected resource and are redirected to login.php, then this cookie is first set. after login, user is redirected here.

There are other various cookies that appear to be related to login, but are not part of the main login flow.

login_x
keeps track of the email the user last entered. Used to pre-populate the box on the login page.

h_user
"Hashed user". This is a hash of the user id of the last logged in user. Used for tracking logged out page views.

d_user
"Dark user". Set when a dark user is logged in, by clicking on an invitation link and routed through p.php. This operates the same as c_user, but can be invoked via get_logged_in_user('d_user');

login
I don't even know, it's a plus sign when you're logged in, but seems useless to me
Used to be the previous cookie we used to keep track of what email user entered. Replaced by login_x. We had to force everyone to logout at one point in time. I think it was because of cache busting.

Flow Chart

Initial Login

When someone logs in, here's what goes down:

1. User submits email / password to login.php

- Server looks up user id from email. (in login_emails table, federated)
- Checks password against hashed value (password_crypt in info table, federated)
- If it matches, create a new session in the session table on the user's database.

session table at login

	uid	sid	update_time	old_key	key	future_key
value	12345	0	now()	0	A	B
example	2901279	0	1207591313	0	241291693	509106625

- Set these cookies on the user:

cookies sent to the user

key	value	example
login_x	email used to log in	email@given.com
c_user	user id	2901279
xs	hash of 'key' from session table i.e., gen_sec_key(A,'c_user')	7bb1e0f40f6b9d0178cd97c55ff441f5
sid	only set if you clicked "remember me"	2

2. Usual Page Load (user clicking around)

User sends the **c_user, xs,** and **sid** cookies to the server. Those are checked against the contents of the session table on the user's database. If a key matches either the old_key, key, or future_key, then it is considered passed.

Great! So the user clicks around for a while, and on each click their xs cookie matches the key in the database. But if more than 4 minutes have passed since the key was created, then the key gets rotated. The keys are rotated to limit the damage of a stolen key. If a packet sniffer catches someone's cookie, then that xs cookie can only be used for up to 8 minutes before it will no longer work.

When the key is rotated, then key becomes old_key, future_key becomes key, and a new future_key is generated. The session table now looks like this:

session table after a key rotation

	uid	sid	update_time	old_key	key	future_key
value	12345	0	now()	A	B	C
example	2901279	0	1207591313	241291693	509106625	726026508

cookies sent to the user

key	value	example
xs	hash of the current key	af675fbb8dafoff5c1fee78e4d281a83

3. Login Errors

Everytime the user hits Facebook, the client sends the cookies above and is authenticated. If any of the three keys doesn't match, then the user is logged out and the process starts over.

All login-related cookies are cleared when the user logs out.

User uses multiple browsers

Suppose you log in with Firefox, and you get xs key A. Then, if you log in with Safari, the key in session 0 is deleted and a new one is generated. The next time you hit a page from Firefox, your xs cookie will not match any available key, and a logout occurs.

Code Reference

Check out lib/login.php for details, although good luck. Start with get_loggedin_user(), since that is where all of this comes from. Pay particular attention to check_session_cookie() and change_login_keys(). Also see logout_user() for a list of the cookies that are cleared on logout.

Security

- Keep in mind that cookies are typically private, but they *could* be intercepted, either by a network sniffer or by any proxies that the user passes through. For instance: http://401000.info/cookies_in_4.txt
- People pay attention to our cookies, strangely enough. For instance: http://my.opera.com/quakerdoomer/blog/2009/05/01/facebok-daughtry-song-abt-var-coookie