Attacking Facebook with simple Brute Force / Dictionary techniques for massive accounts harvesting

(In addition, getting full name out of an email address via Facebook - partly working)

Yaniv Miron aka Lament lament [AT] ilhack [DOT] org @lament1337

Responsible Disclosure: I have tried to communicate with Facebook regarding the issues in the following article before publishing it. It seems that they do not care. Therefore I did my part as "Responsible Disclosure of security vulnerabilities". This document was written quick & dirty, so it might contain typos etc, sorry about that. BTW I do not have a Facebook account.

Goal: Malicious hackers are trying to get access to massive amount of Facebook accounts (as fast as possible). They do not target specific users, and aiming for quantity. The reasons for executing these attack varies. Some of the reasons are:

1. Massive information gathering. Malicious hackers collects personal information, images, videos and more and trade them for money. Either with 3rd parties or by blackmailing the owner.

2. There are websites that promise to get you a Facebook account password for as low as 30$ (USD). When they get a request, they will just pull the password out of their own DB that contains account information. Therefore they are not actually hacking an account once get paid, but already hacked it in the past. If they don't have the data they either reply that the user needs to pay more or just let the user know they cannot help him with this specific account.

3. Just for fun / to prove it's possible / cause damage.

Attack steps: Two random emails were picked for this test. The first one is " ngksngksfngkfsng@gmail.com" and the second one is "david@gmail.com". I do not own any of these two email address.

Step 1: I will try to figure out, using Facebook error messages if the user "ngksngksfngkfsng@gmail.com" is being used in Facebook as an account.

I will go to www.facebook.com and try to login with "ngksngksfngkfsng@gmail.com" as user and "123456" as password. The password is just a random password that was picked, it could be any other password.

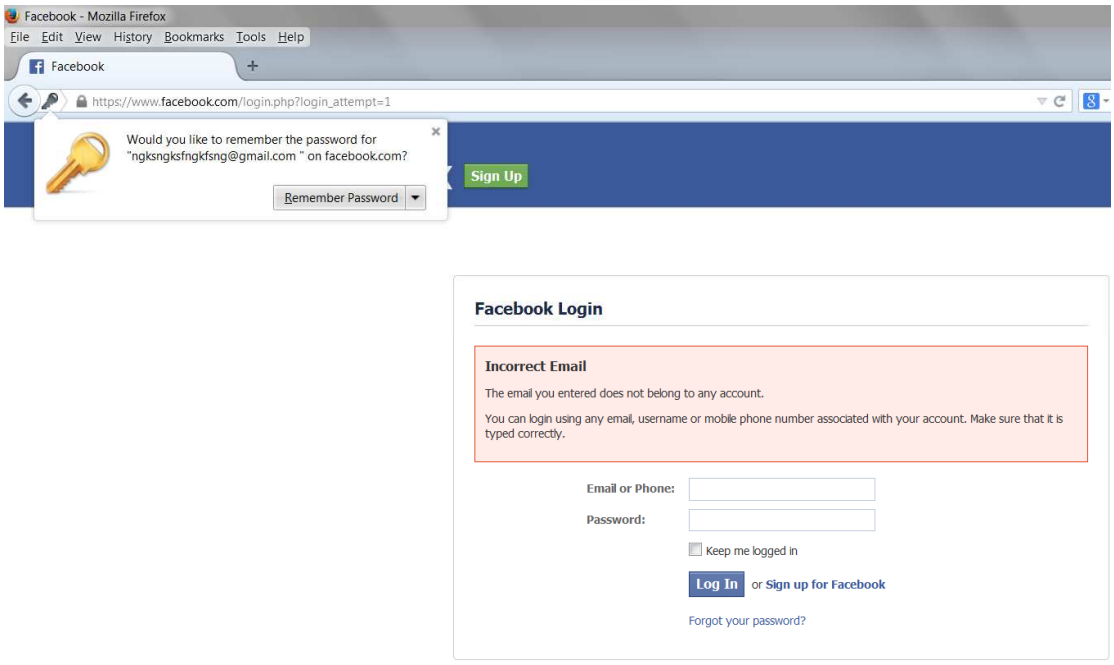
Step 2: I will get an error message from Facebook, the error message is:

" Incorrect Email

The email you entered does not belong to any account.

You can login using any email, username or mobile phone number associated with your account. Make sure that it is typed correctly."

Screenshot:



Step 3: By reading the error message above, it is possible to understand that the email "ngksngksfngkfsng@gmail.com" is not being used in Facebook. Let keep that info for now.

Step 4: I will try to figure out, using Facebook error messages if the user "david@gmail.com" is being used in Facebook as an account.

I will go to www.facebook.com and try to login with "david@gmail.com" as user and "123456" as password. The password is just a random password that was picked, it could be any other password.

Step 5: I will get an error message from Facebook, the error message is:

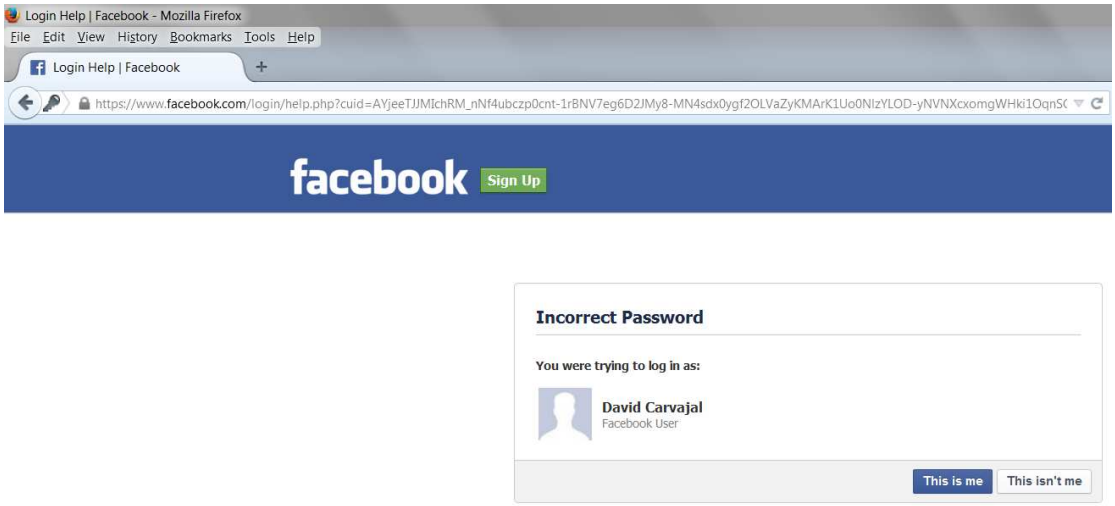
"Incorrect Password

You were trying to log in as:

David Carvajal

Facebook User"

Screenshot:



https://www.facebook.com/login/help.php?cuid=AYjeeTJJMIchRM_nNf4ubczp0cnt-1rBNV7eg6D2JMy8-MN4sdx0ygf2OLVaZyKMArK1Uo0NlzYLOD-yNVNXcxomgWHki1OqnSGrxgnPGvLqEbUUrzTt22O1MSv-5HXFUSRNq7mek_rcDiwuqhNCyiby&st=ambiguous

(now does not exist since Facebook removed it).

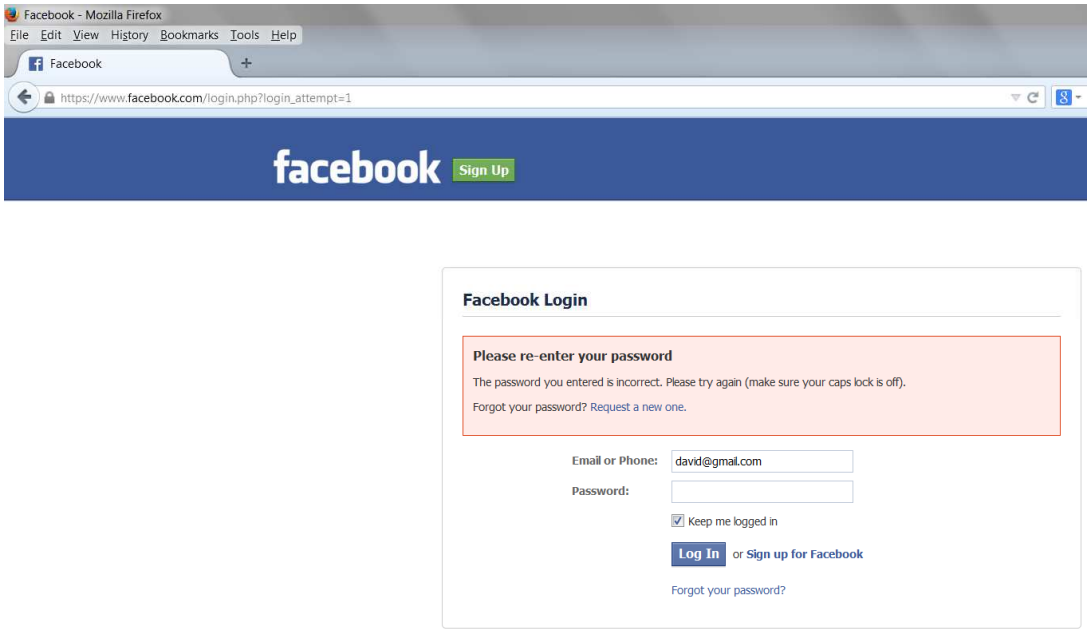
NOTE: Facebook changed the error message after my report, but did not gave me credit nor gave me a bounty. Nevertheless the issue still stands. Here is the new error message by Facebook

"Please re-enter your password

The password you entered is incorrect. Please try again (make sure your caps lock is off).

Forgot your password? Request a new one."

Screenshot:



Step 6: By reading the error message above, it is possible to understand that the user "david@gmail.com" is being used in Facebook. Let keep that info for now.

Step 7: Now that we know how easy it is to know if an email address is a Facebook user or not, we can cut Brute Force / Dictionary attacks time/requests massively. Now how's that? Well, let's say that we would like to do massive attack on Facebook users. First we need to collect emails from the net. Now let's say that we have 100,000 email addresses. And that we would like to try 3 passwords on each which are "123456", "1234567", "12345678". We would have to try 100,000 * 3 = 300,000 requests to test them all (well probably less as some of them might hit on first try, but just to keep it simple). The thing is, that we do not know if those 100,000 emails that we have are even Facebook users. They might be, they might be not. It could be that we are wasting our Brute Force time for nothing as none or little of them are actual Facebook users. Now this paper comes handy. As with the method in this paper a malicious hacker will first check if the email is a valid Facebook user and then try to Brute Force it. Now let's see the math here. We have 100,000 emails, first we try them all to see if they are valid or not. It means 100,000 * 1 = 100,000 requests. Now let's say 25,000 emails are Facebook users. That means 25,000 * 3 = 75,000 requests. All in all we have 175,000 requests instead of 300,000 requests. Almost half. That save us valuable time and efforts.

Now, there is another issue that I noticed.

If you'll take a look on the screenshots it is possible to see that when I've used the david@gmail.com email (which I do NOT own) Facebook replied with this users full name: "*David Carvajal*". This is an easy way to get full name out of an email. Sometimes it works, sometimes is doesn't. I'm not sure how it works in Facebook and not going to spent time to check. But I do know that for some people it works as my example shows.

Final notes: It's a pity that Facebook ignores these things, you should think twice if you want to use their services. In addition Facebook did not paid me any bounty, so think twice before reporting an issue to Facebook, maybe it's better to keep it for yourself / sell it / share it with the security community.

Timeline: 21 July 2014 - Issue Found.

22 July 2014 - Issue reported to Facebook.

23 July 2014 - Facebook replied that it's not an issue.

24 July 2014 - Trying to explain the issue again to Facebook.

24 July 2014 - Facebook replied that it's not an issue.

27 July 2014 - Publishing this document. Shame on you Facebook.

 # e [0] f #