Abhi Mangal Agarwal

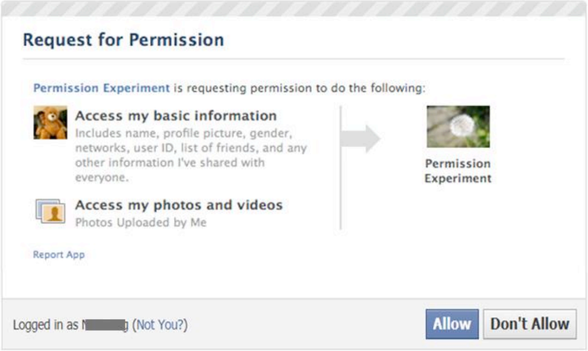
The impact of “Login with Facebook” upon the security of individuals, and its trade-off with traditional authentication systems

The emergence of social networks and the creation of OAuth technologies have led to the creation of centralized sign-on platforms such as Facebook, Twitter or Tumblr. They allow individuals to login to a broader range of websites on the Internet using authentication methods set by these platforms. Let’s take Facebook and the New York Times as an example. Say there is a non-tech savvy individual who reads the New York Times daily and uses his email and password to login to access it, and now lets say the New York Times introduces its “Login with Facebook” feature and the person decides to integrate and use this feature for the sake of convenience. This means that he now wouldn’t have to use his Username and Password, but would be authenticated by just clicking the button “Login with Facebook”. Later, as this person now understands the ease and convenience of using this, he uses it on another website- one that might not be as secure as the New York Times. Does this person understand the implications of logging in via Facebook? Does he understand that Facebook sends this website data about him? No, he doesn’t - only people who are developing and using these technologies fully understand the power, and the impact when users “Login with Facebook”. These platforms have created privacy and data security issues, which are extremely difficult to deal with by Facebook, as they can’t spend time verifying each application that uses its authentication system.

Are passwords better to use in everyday life than these centralized platforms? We have to understand what both of these technologies use in order to understand how we can rate them. Let’s understand how centralized platforms are used first, before moving onto passwords.

To fully understand the mechanisms behind these platforms, we must understand the basic technology involved. Firstly, we have to understand the concept of single sign-on systems. Single sign-on systems, or SSO systems, enable “a user to login once and gain access to multiple websites without the hassle of repeatedly typing their passwords”. Over 77% (Chen, 365) of individuals prefer these to the traditional authentication mechanisms, even though most don’t understand the underlying implication of using these systems. Knowing the fact that more and more high-value personal and business data are moving to the cloud, “authentication flaws can completely expose such information assets to the whole world” (Chen, 367).

Secondly, the standard protocol on the Internet for these centralized sign-on authentication mechanisms is called OAuth. OAuth is the standard for exchange of information between applications, and it is a mechanism that “allows users to grant a third-party application access to their account without having to provide that application with their credentials” (Paul, 2). Think of it as a set of protocols that have to be met by the website you are logging into, the third-party network (i.e. Facebook), and you. Authentication simply works as a website-to-website transfer where one website basically transfers information to another website while obeying the rules dictated by the OAuth protocol. Lets demonstrate this through an example in Figure 1:



Request for Permission

Permission Experiment is requesting permission to do the following:

Access my basic information
Includes name, profile picture, gender, networks, user ID, list of friends, and any other information I've shared with everyone.

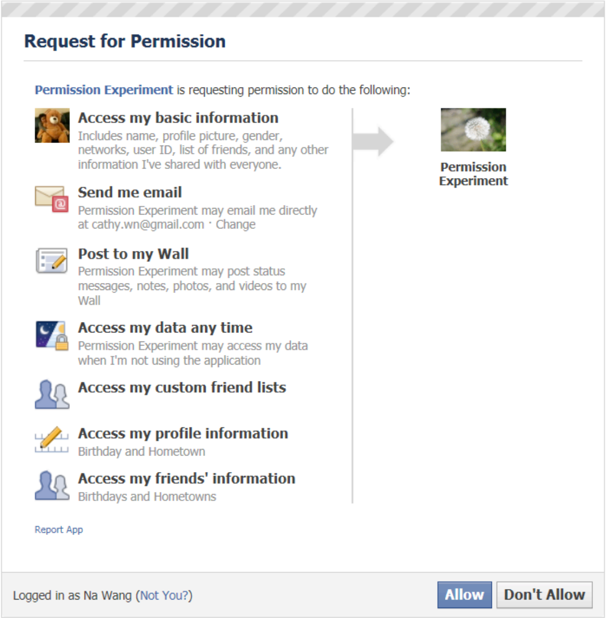
Access my photos and videos
Photos Uploaded by Me

Report App

Logged in as [redacted] (Not You?)

Figure 1 - (Wang, 5)

In the figure above, you're able to see the process of logging into Facebook from an external website. It asks for a certain list of permissions, such as "Access [to your] basic information", and "Access [to my] photos and videos" so this application would be able to access your name, profile picture, friends, photos, and more. Let us see an example of an authentication request that requires more information:



Request for Permission

Permission Experiment is requesting permission to do the following:

Access my basic information
Includes name, profile picture, gender, networks, user ID, list of friends, and any other information I've shared with everyone.

Send me email
Permission Experiment may email me directly at cathy.wn@gmail.com · Change

Post to my Wall
Permission Experiment may post status messages, notes, photos, and videos to my Wall

Access my data any time
Permission Experiment may access my data when I'm not using the application

Access my custom friend lists

Access my profile information
Birthday and Hometown

Access my friends' information
Birthdays and Hometowns

Report App

Logged in as Na Wang (Not You?)

Figure 2 - (Wang, 3)

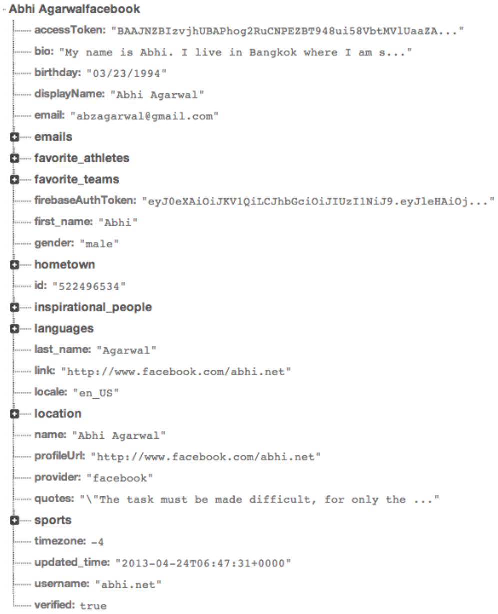
Figure 2 shows the extent of information Facebook is able to request for, and it is actually transferring more than most individuals would think it would. It is just categorizing it into these short descriptions, but behind the scenes each of these blocks gives access to over 20 things from their profiles. In this scenario it is able to "[access] my friends' information", and "access [it at] any time" (Wang, 3). It would not be able to just access your information but also the "people you share with" (Angara, 2), and "any photo or post made by an individual despite the possible fact that the privacy settings of the user who made the original post are restricted only to friends" (Angara, 2). Individuals don't usually pay attention to these settings, and the general understanding is that they only transfer information statically or only once, but once this "Allow" button has been pressed there is a link that allows information go back and forth. The information, photos, and videos of both the individual and their friends could be retrieved despite the level of privacy set on them.

Now we have an understanding that there is a protocol in place that allows individuals to login through third-parties, and that anyone with a Facebook account can login to any website that uses "Login with Facebook", let us look at some of the details that go behind in creating an application that uses this mechanism. When creating an application, you have to go to the Facebook Developers page and generate a new consumer key, and consumer secret key. Note that anyone is able to create these "apps", and the only requirement is having a Facebook account. These two keys allow third-party websites to communication with these OAuth-enabled services to "uniquely identify themselves to the service" (Paul, 2). Moreover, developers will be able to use these keys and define them in their application to be able to communicate with Facebook.

Furthermore, if both the consumer key and the consumer secret key were stolen or taken from the application, then you as an individual would be able to log you into their websites without your authentication or agreement. This is because Facebook would know that you have given certain permissions to the specific consumer key and secret key, and would allow them to access your information freely.

Let's consider a case study on the Twitter OAuth system: a few years ago an individual from ARSTechnica was able to get hold of the Consumer Key, and Consumer Secret Key that Twitter had used in its official Android application. These keys are special, and very dangerous for individuals who have used the android app as "Twitter [had] configured them to enable access to special [characteristics of your profile] which aren't generally available" (Paul, 5) to normal developers, and gaining access to these would mean they would be able to get all the information about the user when they used the "Login with Twitter". Moreover, Twitter would have also disabled the need for the secondary agreement to permissions, as it wanted the access to be quick and painless for its users.

These scenarios don't happen very often for large companies like Twitter, but smaller applications sometimes do have security flaws that can be breached by anyone with an Internet connection, and some knowledge of programming. These applications are designed by a few individuals, and so they aren't completely checked by subject-related professionals. For example, memory management tools aren't checked by individuals who had a degree in memory management but by engineers who took a memory management class in college, and therefore can only apply a basic understanding of the topic on the application. These small memory management flaws can lead to security breaches, which will inevitably lead to an exposure of your data. Let's look at the structure of the data that companies retrieve when you login. Figure 3 shows a simple database of what is saved when I login to Facebook through a website.



```
- Abhi Agarwalfacebook
  |-- accessToken: "BAAJNZBIzvjhUBAPhog2RuCNPEZBT948ui58VbtMVlUaaZA..."
  |-- bio: "My name is Abhi. I live in Bangkok where I am s..."
  |-- birthday: "03/23/1994"
  |-- displayName: "Abhi Agarwal"
  |-- email: "abzagarwal@gmail.com"
  +-- emails
  +-- favorite_athletes
  +-- favorite_teams
  |-- firebaseAuthToken: "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOj..."
  |-- first_name: "Abhi"
  |-- gender: "male"
  +-- hometown
  |-- id: "522496534"
  +-- inspirational_people
  +-- languages
  |-- last_name: "Agarwal"
  |-- link: "http://www.facebook.com/abhi.net"
  |-- locale: "en_US"
  +-- location
  |-- name: "Abhi Agarwal"
  |-- profileUrl: "http://www.facebook.com/abhi.net"
  |-- provider: "facebook"
  |-- quotes: "\"The task must be made difficult, for only the ..."
  +-- sports
  |-- timezone: -4
  |-- updated_time: "2013-04-24T06:47:31+0000"
  |-- username: "abhi.net"
  |-- verified: true
```

Figure 3 – User Information database store

Figure 3 is an of the database where my information is stored when I used “Login with Facebook” on a product I made (http://www.things.pw/nyu), or any product in general. The information is what Facebook pings back to the website, and what the website can choose or choose not to store. The image only lists the basic information of the user, but Facebook gives you access to a field called the accessToken as shown at the top, which lets you get data from the users in real time through a service called the Facebook Graph API. This accessToken can be used by anyone who has this information, even if they don’t have the consumer key & consumer secret key, and thus a security breach can cause more harm than companies would think.

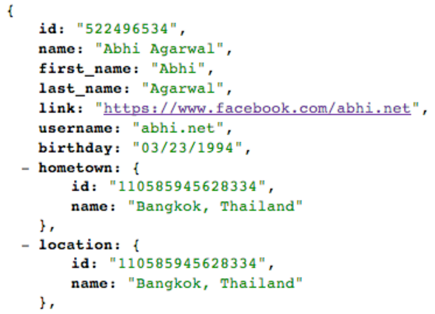
By going to this URL:
https://graph.facebook.com/USER_NAME/?fields=inbox&access_token=ACCESS_TOKEN

individuals are able to get data of the user (*USER_NAME*) if they have access to the access token (*ACCESS_TOKEN*) of the user. In this case, if I was a company who was able to go into the databases of their users and get their access tokens, I could do this to get his or her “about” page:

https://graph.facebook.com/*abhi.net*/?**about**&access_token=*BAAJNZBIzvjhUBABi5lYDciDnFueCXAZBRmvrnkblJvRJ0AdCFF2fO0cAy2fwY9eude9V3yl6fdZB9x1MJtYjQfRwfSrt86KGZBVqrJgyYny0d3lZCxxm7B0XwrhaRANTZBMZCSi5oNbZCZBxIKgX8ZAZApnJHNqmZCldY0rxlprZBQSptNtO….*
(To protect the data)

This would allow me to retrieve the ‘about’ portion of the user (abhi.net). I would be able to read through this section of this particular user; I could also gain access to the messages (inbox) portion of the user by just changing a few settings. Nowadays, this is how marketing companies often work – they get user information through the Facebook Graph API (as shown above), and analyze it through both human interaction and algorithms, which gives them a whole new understanding of their customers.

When putting this URL above into the browser – you receive this:



```
{
    id: "522496534",
    name: "Abhi Agarwal",
    first_name: "Abhi",
    last_name: "Agarwal",
    link: "https://www.facebook.com/abhi.net",
    username: "abhi.net",
    birthday: "03/23/1994",
  - hometown: {
        id: "110585945628334",
        name: "Bangkok, Thailand"
    },
  - location: {
        id: "110585945628334",
        name: "Bangkok, Thailand"
    },
```

Figure 4 - Facebook JSON return

This is basically a format called *JSON* that is used by programming languages to break down the information into variables that the computer can understand. It is also easily understood by humans and broken down into categories anyone could read. By putting in a simple URL, individuals or programs are able to retrieve this information and store it in databases. These access tokens could either be encrypted or could be left insecure depending on the implementation of the databases. So there will be cases where hackers are able to get access to these access tokens very easily. If you have agreed to all the permissions the application asks you on the Facebook dialog box, and your access token from that site gets stolen, then they could potentially just re-build your version of Facebook for their viewing.

In addition, we also have to note that there are services on the Internet that allow individuals to develop “Login with Facebook” applications quickly and easily, without having to write code.

These services do the authentication through their own databases, and therefore save your data on their files. Hence, there is no certainty where the data goes after you leave the “Login with Facebook”, unless the company operating the website is large enough to have a dedicated team to both the privacy and the security of their users.

In general, the same goes for all social networks. They have the same OAuth protocol, and therefore they use the same process that goes on with Facebook OAuth. However, Facebook’s system is generally more secure, and more robust, just because more individuals use it and they have specialized teams to develop these systems.

In addition, to fully understand the impact of these OAuth systems we have to look at the backbone of these systems, and understand the privacy settings behind them. Taking information directly from the Facebook Terms of Service:

Firstly, “The copyright license **does not end when you stop using the service** unless your content has been deleted by everyone else” (TOSDr, Facebook). This means that even after individuals have deleted their accounts, Facebook still has some of your content until everyone has deleted it. For example, if you converse with your friend and you delete your account, your friend will still have the same conversations in her mailbox, and therefore Facebook still has access to your information. Secondly, “Facebook **automatically shares your information** with Bing, Pandora, TripAdvisor, Yelp, Rotten Tomatoes, Clicker, Scribd, and Docs, unless you manually opt-out” (TOSDr, Facebook). Therefore, even without you knowing, Facebook is still your sharing information with third-party sites, and this already puts you in a state where if one of these sites were compromised, your data would be compromised as well. Lastly, “You must **use your legal name publicly** on the service. Using a pseudonym or a pen name is not allowed” (TOSDr, Facebook). Individuals are forced to use their real credentials in order to operate their accounts on Facebook, and therefore individuals are forced to converse and communicate using their real world identities, which could be stolen.

However, the most important privacy issue to remember is that once your data leaves the Facebook website and enters a third-party site, the privacy settings that Facebook has set do not apply. We have to consider the fact that not all websites do obey the privacy laws set by both Facebook and the government, and that the data could be stored in any database both online or offline. There isn’t a privacy security terms of service that developers have to sign or agree to when developing and creating these applications. Moreover, for increased data on privacy, which is not going to be discussed in this paper refer to Appendix I, Figure 6.

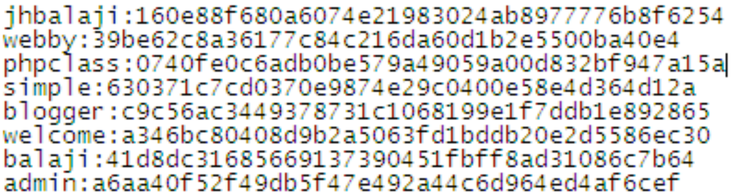
Now, let’s look at how passwords are used by people in the industry, and measure how they could impact the privacy of individuals.

Statistically, over 60% (Danchev) of users use the same password for all their online accounts. If individuals are to use the same passwords for all websites, then it lowers their security, and gives hackers the same access to all their user account like getting their Facebook account. Nowadays most websites require you to have a password with a letter, a capital letter, and a number, this increases the security of your information as it becomes harder for hackers to decrypt and try to crack your password. Let’s try to understand how this process works:

The hackers design an algorithm that inputs multiple generated passwords into Facebook, and Facebook replies with if they are correct or wrong. If the passwords are all just numbers, they will just generate a bunch of numbers and try them. For example if your password is 2 numbers and only includes 0 and 1, it can only be 00, 01, 10, 11, and in the worst case they would need 4 tries to get this password. Now if we set it to include one small letter or one capital letter it could be 00a, 01A, 10A, 11a, etc. Therefore now it becomes more complicated as they now have to try more combinations. There are only 2 choices for each the 0 or the 1, but 52 choices for the small letter or the capital letter (26 small + 26 capital), and this exponentially increases the processing time. Now the password is more secure, but using multiple processing computers this process would still work, and that strength checker uses different combinations of algorithms and checks if it is above a certain number in time to decrypt.

Therefore, if an individual has an easy password, and uses the same password (as most individuals do), then his information could easily be stolen from multiple websites. Moreover, after the development of tools that allow information to be transferred from one website another to get that information, individuals can input one password and get information for each person very easily.

Large encryption systems are used to encrypt passwords in databases when a user signs up. There are systems like MD5, and SHA-2 that are primarily designed to encrypt and decrypt passwords quickly and efficiently when the user enters them, and are entered into a database. There are vast amount of resources in mathematics to design these encryption systems, and it is extremely difficult to decrypt some of this information unless you have the decryption keys. They are decrypted primarily using prime numbers, and as we know prime numbers don’t have a divisor except itself and 1, and therefore a machine would need to generate these prime numbers to decrypt them. The problem is that these prime numbers are 100 or 200 digits long, and therefore would take a large processing power to decrypt. Lets look at an example:



```
jhbalaji:160e88f680a6074e21983024ab8977776b8f6254
webby:39be62c8a36177c84c216da60d1b2e5500ba40e4
phpclass:0740fe0c6adb0be579a49059a00d832bf947a15a
simple:630371c7cd0370e9874e29c0400e58e4d364d12a
blogger:c9c56ac3449378731c1068199e1f7ddb1e892865
welcome:a346bc80408d9b2a5063fd1bddb20e2d5586ec30
balaji:41d8dc316856691373904 51fbff8ad31086c7b64
admin:a6aa40f52f49db5f47e492a44c6d964ed4af6cef
```

Figure 5 - http://www.phpclasses.org/browse/view/image/file/31749/name/Sample-Encrypted-Passwords.png

Figure 5 shows the username and password in a username:password format. Lets take the username “admin” as a base case. We’re not able to tell what the password is in this case as it has been encrypted into something we can’t read. It has been translated into “a6aa40f52f49db5f47e492a44c6d964ed4af6cef” by mathematical formulas into something else, which the mathematical formula can reverse as well when checking if the passwords are the same. By ourselves, we aren’t able to tell what kind of mathematical formula or encryption method have been used, but there is software that is able to do that.

We can safely assume that passwords saved in databases by most companies are extremely secure, but we also need to note that not all websites encrypt the passwords, and in some scenarios individuals who design the websites can see your password.

Now that we have seen how both passwords, and single sign-on systems operate, we are able to compare them to understand which ones to use in which scenarios.

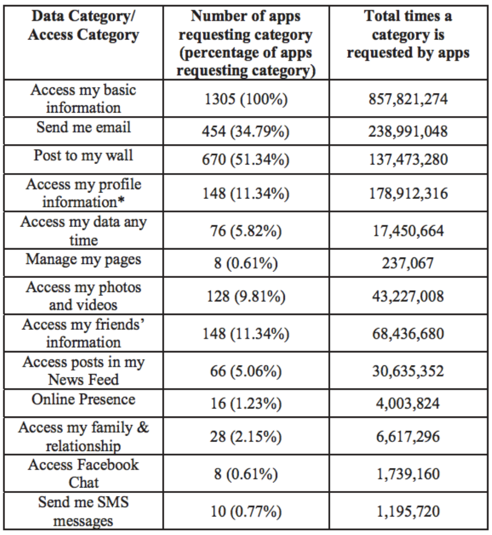
Firstly, if an individual is using a single sign-on system they would only have one password, which would mean that if this particular password got stolen, then the particular person would be able to login to all their websites. Let’s consider a case study that could lead to this example: in “2005, Jones and Soltren identified serious flaws in Facebook’s setup that would facilitate **privacy breaches and data mining**. At the time, nearly 2 years after Facebook’s inception, users’ passwords were still being sent without encryption, and thus could be easily intercepted by a third party” (Debatin, 84). Data mining is a technique where algorithms are able to go to different pages and just retrieve information and store them or analyze them. In this scenario a security flaw would lead to the password being exposed on Facebook, and there are many methods through which a hacker is able to gain access to your computer and get your password. This one password would be able to gain you access into all your accounts with a single “Login with Facebook” authentication box.

However, using single sign-on mechanisms would allow individuals to be signed into other applications while signed onto Facebook, and most of the time, they wouldn’t even have to put in their own passwords to authenticate. This would not only increase convenience, but also safety as now individuals would only need to remember one password that matches the security standards rather than many, and most importantly this password would only be used on one platform. Moreover, if we take past data we can see that most individuals only have one password in general, and centralizing this password to be stored on one platform is an excellent idea, with that benefit of not having to fill in “Full Name”, “Email”, “Address” into all these different registration pages but clicking one button and everything loading for you.

On the other hand, we have to concentrate and understand that there are disadvantages to this transfer of information. We are unaware of what happens to our information after we authenticate and give the website permission to take it. It is difficult to understand which permissions they are asking for, as it is narrowed down into many non-specific descriptions and headers.

In conclusion, after all this information, it is still extremely difficult to compare and contrast between having passwords or utilizing the central sign-on systems. There are strings attached even with using these central sign-on systems with large companies as they use most of the data for marketing, and build applications to analyze that data. But centralized sign-on systems will save time on a daily basis, and remove the need for multiple passwords, which is extremely important, and this would avoid the need to give every website all your information. Therefore, we have to understand the implications of using these tools, but make the decision of using either using a centralized sign-on system or a username/password model ourselves

Appendix I



Data Category/ Access Category	Number of apps requesting category (percentage of apps requesting category)	Total times a category is requested by apps
Access my basic information	1305 (100%)	857,821,274
Send me email	454 (34.79%)	238,991,048
Post to my wall	670 (51.34%)	137,473,280
Access my profile information*	148 (11.34%)	178,912,316
Access my data any time	76 (5.82%)	17,450,664
Manage my pages	8 (0.61%)	237,067
Access my photos and videos	128 (9.81%)	43,227,008
Access my friends' information	148 (11.34%)	68,436,680
Access posts in my News Feed	66 (5.06%)	30,635,352
Online Presence	16 (1.23%)	4,003,824
Access my family & relationship	28 (2.15%)	6,617,296
Access Facebook Chat	8 (0.61%)	1,739,160
Send me SMS messages	10 (0.77%)	1,195,720

Figure 6 - Image to show some data on privacy

Bibliography

Debatin, Bernhard, Jennette P. Lovejoy, Ann-Kathrin Horn, and Brittany N. Hughes. "Facebook and Online Privacy: Attitudes, Behaviors, and Unintended Consequences." *Journal of Computer-Mediated Communication* 15.1 (2009): 83-108. Print.

Paul, Ryan. "Compromising Twitter's OAuth Security System." *Ars Technica*. ArsTechnica, 3 Sept. 2010. Web. 17 Apr. 2013.

Wang, Rui. "Signing Me onto Your Accounts through Facebook and Google: A Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services." *IEEE Xplore*. IEEE Xplore. Web.

Angara, Alexander. "INFS1300–Reflective Paper Social Networking, Privacy and Managing Your Digital Footprint."

Roy, Hugo. "Terms of Service; Didn't Read." *Terms of Service; Didn't Read*. 2011 Chaos Communication Camp, June 2012. Web. 12 May 2013.

"Facebook Terms of Service." *Facebook*. N.p., n.d. Web. 12 May 2013.

Debatin, Bernhard, Jennette P. Lovejoy, Ann-Kathrin Horn, and Brittany N. Hughes. "Facebook and Online Privacy: Attitudes, Behaviors, and Unintended Consequences." *Journal of Computer-Mediated Communication* 15.1 (2009): 83-108. Print.

Danchev, Dancho. "Survey: 60 Percent of Users Use the Same Password across More than One of Their Online Accounts." *ZDNet*. Paypal, 30 Sept. 2011. Web. 12 May 2013.