Independent Work Report Spring, 2014

Convenience over safety: How authentication cookies compromise user account security on the Web

Gabriel Chen

Advised by Prof. Arvind Narayanan

Abstract

Authentication cookies allow for convenient online user authentication, but potential security problems may be encountered if a malicious adversary were able to obtain a given user's authentication cookie. The Firefox add-on, Firesheep, and cross-site scripting methods have demonstrated that attackers are capable of achieving this goal. In this study, we hope to survey user logins across the Web in order to evaluate the vulnerability of user accounts on major websites and to gain insight on potential defenses.

Our approach consists of running two browser profile instances per site: a primary crawler and a shadow crawler. For every website, the primary profile performs a legitimate login and saves the persistent cookies. The secondary profile loads the cookies and visits the same website. If the resulting webpage recognizes a user as having logged in, then the attack is considered successful. For different tests, the shadow crawler is instantiated on either the same computer as the primary crawler, or remotely, on a computer with a different browser fingerprint and IP address.

The findings of this study show that on a large scale, many sites do not yet implement additional authentication factors once an authentication cookie is obtained. We see industry beginning to develop more secure systems, and hope to encourage further progress with our results. In addition, specialized Web security companies have emerged, using more sophisticated methods to protect user accounts that will require investigation.

1. Introduction

Over the past fifty years, passwords have existed as the most widely accepted form of human-computer authentication despite evidence from researchers that a more secure and user-friendly technology is needed [2]. This model has persisted into online authentication with the rise of online service websites. Recently, we have seen these sites make progress in the area, involving the development of new and more complex systems for authentication. Therefore, research in the field of online authentication is expected to evolve from the traditional approach to the topic. As a consequence, it is important to survey the web to identify good practices and to evaluate existing strategies.

In publishing authentication techniques, we encounter the potential ethical issue that exposing vulnerabilities can also make it easier for attackers to succeed in gaining control of user accounts. However, a system that is secure through obscurity is strongly discouraged [12]. The discussion of flaws in security is intended to ultimately improve the safety of user accounts on Web by identifying susceptibilities and encouraging sites to develop better methods.

For the typical login model, a user is required to input his or her username and password in order to log into the site. The credentials are submitted and checked against the website's server and a cookie is sent in response. This cookie is then used by the site to verify the user on subsequent visits. The authentication cookie can be either a session cookie, which expires when the browser is closed, or a persistent cookie, which is recognized by the server even after a browser session ends, until a given expiration date. Persistent cookies are implemented by many websites as a "remember me" checkbox on the login page; when selected, a persistent cookie is sent to the client browser upon login.

The user authentication cookie model gives users the convenience of not having to enter login credentials every time a site is revisited. However, this ease of access trades off with security when not properly implemented. We wish to show that when the authentication cookie is not correctly protected, adversaries may be allowed to access and modify user accounts. Our evidence suggests

that many sites are not implementing further layers of defense once an authentication cookie is recognized.

2. Threat Model

Two related threat models weaken user account security. The first model depends on obtaining a user's password, while the second depends on procurement of the authentication cookie.

2.1. Stolen Password

The most basic threat model to user accounts is the case in which an adversary directly obtains a user's login credentials or more specifically, his or her password. This can be achieved using a number of well-known online and offline attacks, such as client-side malware, phishing using a spoofed site and eavesdropping the password transmission [2]. When used, these attack strategies can compromise a user account for a given site, along with any associated personal information.

A number of defenses against this threat have arisen in recent years. This includes the use of multiple factor authentication, such as the requirement of entering an authentication code that is sent to the user's phone in addition to a password. Furthermore, most websites use HTTPS when sending login credentials; this protects users from eavesdropping attacks.

2.2. Session Hijacking

Our threat model assumes that the attacker has a means of obtaining a user's authentication cookie. With the cookie, the user account is exposed to the adversary, who does not need the user's credentials as the authentication cookie contains all necessary information for the server to allow account access. Although many sites securely transmit login credentials using SSL, they fail to do the same for authentication cookies. Any authentication cookie that is not encrypted with HTTPS can be obtained over an open networking using tools such as Firesheep. A more dated approach exploits websites that use non-HTTP-only cookies; these sites are defenseless against the use of cross-site scripting attacks [15].

Using Firesheep on an open wireless network, an attacker can obtain a user's authentication

cookie for a site, which then allows the attacker to perform functions on the site on behalf of the user without knowledge of the username or password [3]. Firesheep works by performing a passive network attack, meaning that it eavesdrops on network communications between browser and server without modifying the contents [9]. The only effective fix to this problem is the use of end-to-end encryption [4].

Another related attack can be performed using cross-site scripting, in which a cookie can be accessed through the DOM of the host site. Eight years ago, HTTP-only cookies were introduced to prevent this attack. These cookies are implemented as an attribute in the cookie field, and a browser that supports them will not permit a script to access its contents. The cross-site scripting attack may be prevented by the use of HTTP-only authentication cookies, but a recent study has shown that these cookies are still not widely used [15]. Furthermore, older browsers that do not support these cookies remain completely unprotected.

A separate study has shown that third-party JavaScript is often allowed to run unsupervised in a first-party context. These results further affirm the claim that the third party can access non-HTTP-only first-party cookies and has further privacy implications [10]. This means that using JavaScript, the third party can obtain user authentication cookies to execute the session hijacking attack. There also exist ways of exploiting cross-site scripting that go beyond obtaining the authentication cookie, such as capturing user keystrokes in order to determine the password, which ties back to the first threat model in 2.1.

3. Methodology

We wish to investigate the effectiveness of the session hijacking attack on a wide array of websites, so two requirements that we look to satisfy with our methodology are scalability and automation. In addition, we wish to simulate the browser properties and conditions involved in a real user login and more importantly, to ensure that the browser is able to accept authentication cookies. The infrastructure used in this project allows for the dumping and loading of browser profiles, as well as the automated creation of browser instances that can crawl to a large number of sites.

The study tests 109 of the top 200 Alexa[1] sites, looking at only the sites for which user accounts can be created. In order to examine login techniques, a fake user account needs to be created for every site. Automated account creation is tricky due to anti-bot methods such as CAPTCHAs, so each user account was created manually. Because every tested site has a different way of handling user authentication, a means of logging into any given website is also required to automate the large-scale study of site practices.

3.1. Infrastructure

The data for the experiment is collected by a web crawler that is wrapped around the Selenium WebDriver[2]. WebDriver methods let us inspect page sources in order to determine the similarity between pages. Furthermore, it allows for the identification of username and password fields and buttons for data submission; both of the above features are critical for generic website login, which is discussed in further detail in section 3.2. In addition to identifying the login components, the WebDriver also provides commands for entering and submitting field data in a way that closely simulates a real user's data input.

In addition to the Selenium WebDriver functionality, our infrastructure adds critical features that make it appropriate for this project. In our study, we use Selenium to drive Firefox instances; using pyvirtualdisplay to interface with Xvfb, a virtual X display server, the crawler instances may be run headless. This is useful for the remote tests we conduct on the Linode to simulate different browser fingerprints and IP addresses[3].

The web crawler also supports the dumping and loading of browser profiles. This allows the crawler to visit a website, attempt a login, and then save the database of cookies associated with the browser instance if the login succeeds. A secondary profile can then use the crawler to load the profile data, and with it, the persistent cookies of the primary profile. The second profile can be loaded on either the same machine or a remote one, depending on what variables are being controlled for. Using this method, we can simulate the session hijacking attack by loading the

[1] `http://www.alexa.com/`

[2] `http://docs.seleniumhq.org/projects/webdriver/`

[3] Some infrastructure description used with permission from [11]

primary profile's authentication cookies into the secondary profile.

3.2. Generic Login

The layout of each webpage is unique, so the placement of login fields and submission buttons varies by site and cannot be easily found in a standardized way for each site. Furthermore, the login flows are inconsistent between sites; some websites support login from the home page, while others direct users to a secondary login page. This makes large-scale testing of logins difficult, because there lacks an automated way to authenticate a test user on every website in the test. Therefore, in order to test a large number of sites, we devised a method to log into a website by identifying login input fields and a submission button. This routine will be referred to as Generic Login.

3.2.1 Identifying Login Inputs

Despite the numerous possibilities for page layouts and login flows, many input fields, buttons and links associated to logging in share characteristics between sites. Generic Login looks for input HTML tags with attributes that match regular expressions to identify the fields. Username and password fields are typically easy to identify and work with because of the universal nomenclature of the words "username" or "email" and "password." Furthermore, given a set of multiple fields that we suspect to be a username field, we enter the username in all of those fields. The same applies to password fields, although another check is necessary to ensure that the sets are disjoint, as we do not want to enter both types of data into the same field. As long as the correct submission button is identified, it is not problematic if data is entered in incorrect fields.

The success of Generic Login hinges upon the correct identification of a submission button, as we only enter input once per page. If multiple submit buttons are found, only one is chosen arbitrarily, so a potential improvement on the success rate would be to try all identified submission buttons. This may also lengthen the runtime of the experiment, because data entry needs to be repeated for each potential button.

Several of our tests involve the dumping and loading of browser profiles between crawler instances. The profiles only store persistent cookies, so Generic Login needs to ensure that persistent cookies

are created. This is done by inspecting the webpage for persistent cookie checkboxes using the same regular expression matching as above. Depending on the site, the checkbox may be selected by default, but otherwise, for every checkbox identified that is not selected, the WebDriver selects it prior to submitting login information.

3.2.2 Identifying Successful Logins

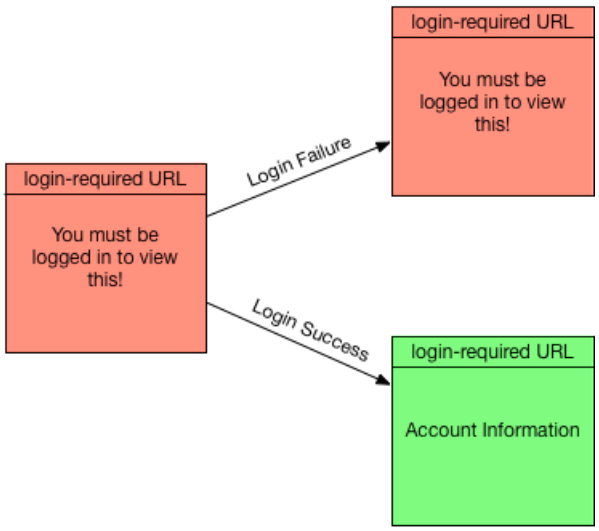


Figure 1: Differentiating between a successful and unsuccessful login on the same crawler instance using the login-required URL

Another problem with logging into different sites is determining if a login has been successfully executed. Merely checking whether or not data is entered and submitted is insufficient, as the identified button may be incorrect. Testing similarity between a webpage before and after the login is one solution, given that the appropriate webpage is chosen. In order to address this problem, we use a login-required URL associated with each tested site.

All tested websites support user authentication, so we assume that each provides a webpage for its users to change passwords or edit account-specific information. A login-required URL is such a webpage that is only viewable by an authenticated user and is typically used to edit personal data. A manual pass over all sites confirms the existence of such a URL for each site to be tested. The fact that an unauthenticated user cannot view the contents of the login-required page implies that this URL has an important property that allows us to determine login success. Figure 1 presents

further detail on the difference between login success and failure. An additional reason for using login-required URL is that it emphasizes the severity of the session hijacking attack, demonstrating that personal information may be accessed and in some cases, accounts passwords may even be changed.

3.2.3 Three Tier Login

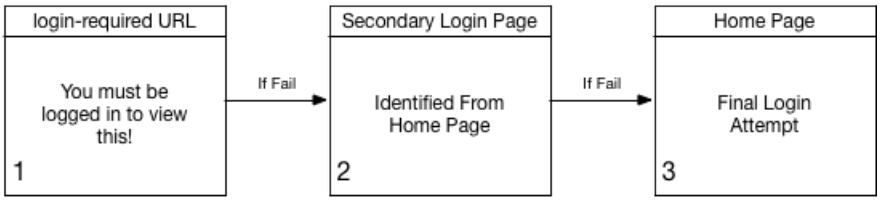


Figure 2: After 2^{nd} and 3^{rd} attempts, return to login-required URL, performing test from Figure 1

In attempting the login to a given site, we identify login inputs in three stages. First, we visit the login-required URL. Most sites redirect visitors who are not logged in to a login page when a login-required URL is accessed. If the login is unsuccessful on this page, we return to the main page of the site.

Here, we use a regular expression to determine the existence of a secondary page for login; we found that most sites choose not to support login from the main page, so finding a secondary page is a natural second tier. If such a page exists, the crawler is redirected to that page and a login is attempted using the login input identification described above.

Finally, if the login is unsuccessful on both the login-required URL redirect page and the secondary page, the login is attempted on the main page.

3.3. Experiment Design and Variables

For every tested site in this experiment, we use two crawler instances to test the efficacy of session hijacking and whether or not it uses the IP address or related defenses. The primary crawler represents the user, while the secondary represents an adversary that is utilizing the user's authentication cookies to gain access to account information.

In each test in this study, the primary crawler is initialized and makes a request to the login-required URL. The page source at this URL is saved as a control source or basis of comparison for

determining login success. A Generic Login is attempted and if it is successful, the primary crawler stores the relevant cookies and profile metadata for the secondary to load.

Data collected for each test includes the cosine similarity value between the secondary profile and the primary profile, as well as whether or not the two pages are deemed similar, as dictated by a given threshold. The corresponding HTML page sources are also saved for each site and each run of the experiment, allowing for manual inspection of select pages.

3.3.1 Variables

After initialization of the experiment, there are a number of variables that websites may be using with regard to secure user authentication that can be tested. Our main target of study is the use of various types of cookies that facilitate user convenience by not requiring that a user enter his or her username and password during every login; these include both session and persistent cookies. Traditional cookies are specific to the browser profile that stores the cookies. In order to test this variable, we load in the secondary instance of the crawler the profile of the primary crawler.

A second factor in authentication is recognition of a set of frequently used IP addresses. A website may choose to save the normal IP address of a user in order to distinguish between a legitimate login and an attacker login. The website may enforce greater security by checking the IP address even when a user has an authentication cookie, but it may also check IP address only when a secondary login is attempted without one. Both variations are tested using a remote headless instance of the secondary crawler that is run on a Linode cloud server. To test without the cookie, Generic Login is performed on both the primary and secondary crawlers, while a test considering the authentication loads the primary profile into the remote secondary profile.

Finally, some websites may examine the user-agent string, browser fingerprint or local shared objects[4] (LSOs) of a user in order to distinguish one browser or one machine from another. Each of these potential factors are not isolated in this project, but all three differ along with IP address for the remote tests. The following sections provide further detail as to how the other variables are tested.

[4] `http://www.adobe.com/security/flashplayer/articles/lso/`

3.3.2 *Authentication Cookies, same IP, machine and browser*

In this test, both crawlers are run on the same computer, a Mac OS X 10.9, on a Firefox browser instance driven by Selenium. The experiment is conducted using the browser profile model described in 3.1, so the cookies stored in the primary crawler's profile are loaded by the secondary crawler, which then visits the login-required URL to compare the resulting page sources. The success of the secondary crawler depends on whether persistent cookies are stored on the primary crawler.

3.3.3 *Dual Generic Login, different IP, machine and browser*

In this test, after the primary crawler successfully executes a Generic Login on the OS X computer, a remote call is made using the python paramiko[5] ssh client, which connects to a Linux Ubuntu 13.10 computer. The call executes a script that instantiates a headless secondary profile and performs a second Generic Login on the same site. Both crawlers are directed to the login-required URL and the page sources are compared.

The goal here is to determine whether or not sites recognize user logins from different locations, as evidenced by manual testing of select sites. This corresponds with the stolen password threat.

3.3.4 *Authentication Cookies, different IP, machine and browser*

The setup for this test is the same as above, except instead of performing a Generic Login on the secondary crawler, the primary profile is loaded onto the secondary crawler. We use sshfs[6] in order to mount the remote file system onto the primary machine, as the profile is dumped in the context of the primary crawler's machine but must be loaded in a remote context. Both crawlers are again directed to the login-required URL and the page sources are examined.

Of the three tests, this test most closely simulates an actual session hijacking attack, where the adversary is on a different machine than the victim.

3.4. Testing Page Similarity

Testing similarity or dissimilarity between two pages is a key subproblem of both Generic Login and the overarching user authentication cookie experiments. We store the page sources of each

[5] `https://github.com/paramiko/paramiko`
[6] `http://osxfuse.github.io/`

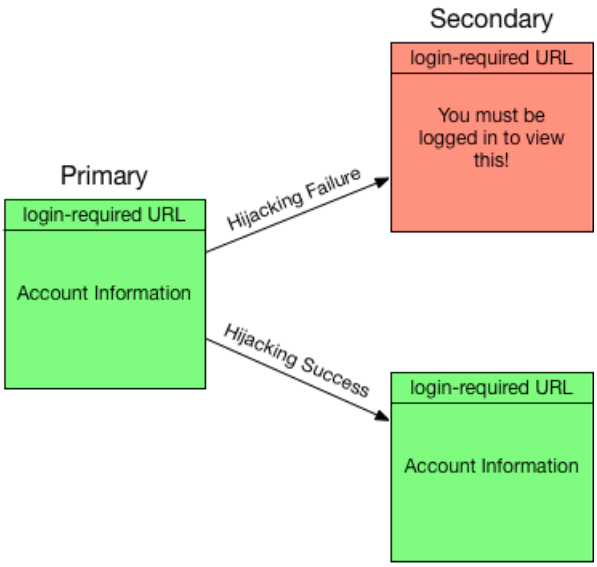


Figure 3: Differentiating between a successful and unsuccessful session hijacking attack across two crawlers using the login-required URL

website for manual inspection, but we also use an automated solution that relies on cosine similarity, which has been proven to be an effective way of comparing two texts [13]. Each webpage source is converted to a bag of words, a vector of words on the page and the number of appearances associated with it. Using the cosine similarity formula, we can compute the angle between the vectors and therefore determine if two page sources are alike.

For Generic Login, it is necessary to determine whether a legitimate login succeeds or not on any website. We achieve this by comparing whether or not the login-required URL page is different before and after the login attempt, as in Figure 1. Comparing just any page, such as the home page, does not always yield correct results, due to the dynamic nature of some pages or the minimal changes a logged in user sees for certain pages. The unique properties of the login-required URL discussed in 3.2.2 make it an appropriate basis of comparison for testing success.

For each of the experiments, the method for determining whether or not a secondary profile has successfully logged in relies on comparing the page source of the secondary profile to that of the primary profile, either after parallel Generic Login attempts or after cookies are loaded into the secondary browser profile. Again, we use the login-required URL to ensure a reliable comparison. However, the differentiation between profiles is used instead of contrasting the before and after

sources of the login-required URL because a site that detects an unusual activity may direct a user to a page different from the login-required URL, which still represents a failed attack. In more detail, the approach differs from the one described above for Generic Login in that instead of the dissimilarity between the control page and the post login page, we look for the similarity between the primary and secondary profiles' sources, as shown in Figure 3. This ensures that if an additional factor is used, such that a website detects the misuse of authentication cookies by an attacker, a different page would not result in a false positive.

4. Results

For comprehensive results for each website, see A.1. For each of the following tests, we identify sites for which a secondary crawler login is possible under corresponding variable conditions.

4.1. Generic Login Success Rate

Using Generic Login, we succeed in logging into 55 out of 109 sites. The tool is most effective on static webpages that do not rely heavily on JavaScript to load hidden elements, so that login inputs can be easily identified. Being able to log in on the primary crawler is essential for our tests, so websites for which Generic Login fails are excluded from this study.

4.2. Authentication Cookies, same IP, machine and browser

In this test, both the primary and secondary crawlers are run on the same machine, using separate browser profiles. Of the 55 sites for which Generic Login was successful, 42 allowed the secondary crawler to log in using the loaded profile from the primary crawler. This means that at least 42 sites give complete control to a visitor with an authentication cookie. However, according to our threat model, session hijacking is conducted by a party on a different computer with a different browser fingerprint, in the case of Firesheep, or by a party with a different IP address, in the case of cross-site scripting. The conditions in this test do not model the attack realistically, so this result is primarily used to compare the effects of other variables in subsequent simulations.

With regard to persistent cookies, we obtain another important interpretation of this result. For

this test, the success of the secondary profile in using the primary profile's authentication cookies depends on whether or not persistent cookies are created. In other words, we know a primary crawler that visits the aforementioned 42 sites correctly dumps persistent cookies. For the remaining 13 sites, either persistent cookies are not supported on the site, identification of the "remember me" box fails, or the site is using an uncommon heuristic for deterring user authentication cookie login, even if the attack is conducted on the same computer. An explanation for why the cookie login failed can be concluded without further manual inspection.

4.3. Dual Generic Login, different IP, machine and browser

In this test, the secondary crawler is run on a remote machine, but both crawlers perform Generic Logins and do not rely on cookies. The goal is to identify whether any sites identify users by IP address or other browser conditions in order to recognize the more standard stolen password attack. Of 55 sites that Generic Login succeeded on using the primary crawler, 52 were also successful on the secondary crawler with no distinction given due to a different IP address. Login failed remotely for the remaining 3 sites. For two of these sites, we are unable to conclude whether IP address played a role, or if Generic Login failed for other reasons. This is explained in more detail in 5.2. Further analysis of the third site, `linkedin.com`, is in 4.5.

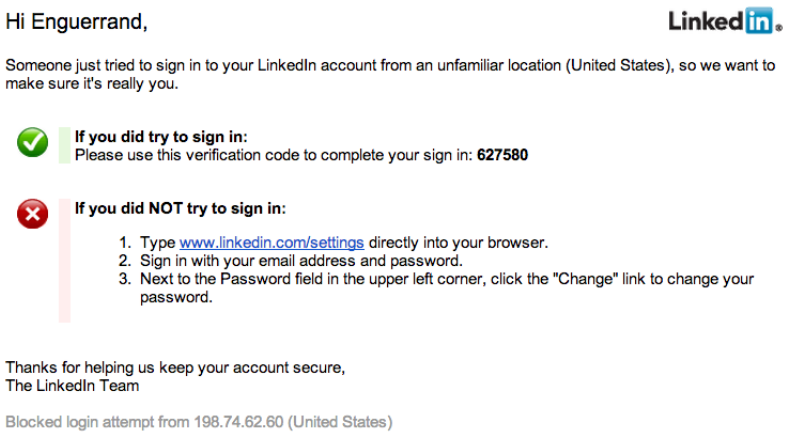
4.4. Authentication Cookies, different IP, machine and browser

This test adds several variables that are held constant between the crawlers in 4.2. Given that only 42 of 55 persistent cookie attacks were guaranteed by the earlier test, we expect fewer successful results here. Of 55 sites that Generic Login succeeded on, 33 allowed logins on the secondary profile when the site is visited on a remote headless crawler. From A.1, we see that for these 33 sites, a visit from the secondary crawler using database cookies succeeds regardless of whether or not it is run remotely.

As with the other tests, it is difficult to determine the cause of failure without further testing. It is possible that of the 22 sites, there exist sites that are checking factors such as browser fingerprint, user-agent string and IP address. The failures may also be attributed to complications in the remote

testing methodology.

4.5. An Idiosyncratic Result


Hi Enguerrand,

LinkedIn

Someone just tried to sign in to your LinkedIn account from an unfamiliar location (United States), so we want to make sure it's really you.

If you did try to sign in:
Please use this verification code to complete your sign in: **627580**

If you did NOT try to sign in:

1. Type www.linkedin.com/settings directly into your browser.
2. Sign in with your email address and password.
3. Next to the Password field in the upper left corner, click the "Change" link to change your password.

Thanks for helping us keep your account secure,
The LinkedIn Team

Blocked login attempt from 198.74.62.60 (United States)

Figure 4: An email received when a remote generic login was attempted

A manual inspection of the email inbox associated with test user accounts on many websites revealed the above message. The finding corresponds with the data obtained from the automated experiment involving a dual generic login, as seen in 4.3 and A.1. Here, it is evident that `linkedin.com` is likely using an IP address-based heuristic to identify suspicious activity, an effective defense against the stolen password attack. The user is required to enter an identification code sent via email in addition to his or her password in order to complete the login. However, as shown by the site's results for the other two tests, possession of the user authentication cookie is enough to bypass this security check. Although not included in this study, `yammer.com` is known to use the same technique.

5. Discussion

In the following discussion, we will treat successful logins on the secondary crawler as positive results, and failed attacks as negative results. In general, results show that sites do not defend against session hijacking during this stage of the attack; whether or not end-to-end encryption or HTTP-only cookies are used is a part of the study, but may be surveyed with further testing.

5.1. False Positives

One priority of our project is to minimize and eliminate false positives, where an unsuccessful login is counted as a positive result. This is significant in both the treatment of Generic Login results as well as the experimental methodology.

For Generic Login, in choosing the threshold for cosine similarity comparisons, we choose a value low enough such that every positive result is correctly recorded. A low similarity threshold imposes a strict requirement on the test described in Figure 1, so that the login-required URL page contents must differ significantly after the login from before the attempt. For every site in A.1, valid results are dependent on correct Generic Login results, because the similarity between page sources of two failed attempts is high; this result must be distinguishable from the case where both crawler attempts are successful, and the page sources are also similar.

As a consequence, the Generic Login tool's reported success rate is lower than its actual rate, and the experiments can be fine-tuned further to incorporate more sites into the study. Sites that are excluded have login-required URL pages that are similar in source before and after a legitimate login is performed. These webpages typically rely on scripting to conditionally load elements of the page, so that the HTML source is not easily distinguishable.

For each test, the similarity threshold between primary and secondary login-required URL pages is chosen to be high, in order to ensure that the two pages are truly highly similar before drawing conclusions. Both this threshold and the Generic Login thresholds should be adjusted in a more rigorous manner in future experiments.

Finally, whenever either the primary or secondary crawler exits without having successfully completed its computation, the result is also treated as negative. This can be due to more secure practices, but can also be due to inconsistencies in the infrastructure and WebDriver.

In our presentation of the results, we are hesitant to conclude that a site with negative results is using additional authentication factors. However, we would like to emphasize that the identification of positive results is necessary in encouraging the development of better systems. The false positive approach is an essential requirement for correct analysis.

5.2. Limitations

A few limitations of the study are evident from the methodology and the treatment of false positives. The results contain a large number of negative or inconclusive observations for the user authentication cookie tests. Causes for failure are not easily determined in both the single machine and the remote experiments, and more variables need to be controlled for in order to identify specific techniques. Additional constraints include the inability to test certain types of high-security sites.

For the single machine tests, failure to find a "remember me" box and thus the inability to dump persistent cookies is not distinguishable from a failure to log in due to heightened site security. Potential improvements to the methodology include better heuristics for guaranteeing persistent cookie acquisition, such as an improved regular expression for checkbox matching. The reopening of the same browser profile and access of the website can also be used to reveal whether or not persistent cookie storage has been completed.

A limitation of the remote profile loading experiment is that factors for an unsuccessful remote result are not isolated. The negative results can be attributed to either IP address, fingerprint, user-agent, flash cookies. These results may also be due to remote failure of the WebDriver and crawler infrastructure. Further testing with different controls needs to be conducted in order to ascertain which user authentication factors are being used, if at all. Another problem with the methodology for remote testing is associated with the repeated execution of remote tests. Sites that use learning techniques to identify regular behavior may recognize the remote address as one that is frequently used, which could be detrimental to some positive results.

A related factor that is not automatically examined is the identification of emails or text messages such as the one from `linkedin.com` in 4.5. One potential implementation of the test is to parse inbox data in order to identify relevant messages by timestamp. A final aspect of user authentication that is also omitted is the consideration of session cookies. A test for session cookies was implemented early in the study using WebDriver methods, but these methods restrict the copyable cookies by domain. This method proved to be unreliable, for `google.com` was found to rely on cookies from separate domains. A comprehensive automated test was not conducted, but it could

potentially reveal interesting results.

The last restriction to mention is derived from the scalability requirement and the information associated with a fake user account. For the purposes of testing every site, we use a manufactured identity, which does not have access to information such as a phone number, credit card or other sensitive data. As a result, high-security websites, specifically banking websites, are excluded from this study. Disclosure of information from these sites can result in serious harm to its users, so many claim to be using more sophisticated fraud-prevention technology. A study of these authentication security methods may be informative. Nonetheless, much of the user information on the sites in this study is private and can still have negative implications on the user if an adversary is to access it.

5.3. Future work

To summarize, this project can be taken further in several directions to address its limitations. More variables can be controlled for in the remote authentication cookie test in order to identify specific user login methods. Improvements can be made to Generic Login and other infrastructure methods. Finally, a small scale study can be conducted on high-security websites such as commercial banking sites. Each of these potential projects will enhance our results and allow for deeper and more comprehensive analysis of how user authentication cookies are implemented across the Web.

6. Related Work

This project is relevant to several recent advancements in industry and adds security implications to many existing studies in academia.

6.1. Industry Research

We see the beginnings of industry progress in the development and improvement of authentication systems in some of the websites surveyed in this study. Most of these sites are online service sites promising account security to attract more users, so authentication techniques are implemented for private site use.

A market is also emerging for companies that provide fraud detection and prevention services to

other websites. For example, the company 41st Parameter relies on device recognition to prevent account takeover [1]. Related companies such as Kount and ThreatMetrix also work on detecting fraudulent logins and claim to be using techniques similar to ones discussed in this study [7, 14]. These companies market their products to websites that need to protect sensitive client data; in their product descriptions, they claim to implement defenses against stolen credentials and session hijacking attacks [14].

Given an overview of the defenses that each company has to offer, it is clear that relevant research results are being produced in the private sector. However, these techniques are highly proprietary and as a result are obscure. More academic work in the field needs to be conducted in order for wider adoption of well-designed Web user authentication systems.

6.2. Academic Research

This project uses techniques similar to those used in the field of web privacy measurement, building upon a common infrastructure for scalability and automation. However, instead of considering privacy threats, we consider methods of security enhancement, which may be achieved by similar means of user identification such as IP tracking and browser fingerprinting.

Past studies have examined device and browser fingerprinting through a privacy lens. One paper evaluates browser uniqueness by taking its fingerprint from Panopticlick[7], viewing the technology as a privacy concern [6]. Another study examines several companies that claim to be using device fingerprinting techniques in their products, including ThreatMetrix, which is also mentioned above. Another inspected company, BlueCava[8], does not use fingerprinting for security but appears to use it as a marketing solution. This survey of commercial fingerprinting identifies several fingerprinting techniques, such as the use of browser plugins and the detection of system fonts [8].

Sometimes published works make distinctions between third-party and first-party tracking. Above, we see general evaluations of browser fingerprinting and its privacy implications. A few studies point out that user tracking can be much more concerning if the ability of the third party to access

[7] `panopticlick.eff.org`
[8] `http://bluecava.com/`

first-party data is considered.

Concerning user authentication, research has been conducted with respect to the stolen credential attack and its relevant defenses [2]. Other relevant work includes the identification of the attack as demonstrated by Firesheep and several studies on cross-site scripting; these two initial security breaches spurred our investigation of authentication cookie-based attacks. Other research groups propose solutions to the session hijacking attack, such as the use of one-time cookies and yet others evaluate the use of certain types of authentication cookies over the Web [5, 15].

Although work has been done in device identification and its privacy concerns as well as user authentication security, few studies have put the two together. It appears that there is a potential tradeoff between privacy and security, in that the first-party use of device tracking may infringe upon user privacy but can also help identify fraudulent users. Here, it can be agreed upon that third-party tracking is detrimental to both security and privacy.

7. Conclusion

Our study sought to perform a large-scale study of user authentication practices on the Web. In order to achieve this, we built upon an existing web measurement infrastructure and developed a Generic Login strategy to automate user authentication. Using this method, we conducted several experiments to evaluate existing authentication techniques, specifically examining the identification of a remote machine and other security checks given the possession of an authentication cookie.

Results from our study show that many of these online service sites do not employ sophisticated systems for user authentication that protect accounts from this particular type of session hijacking. When a user authentication cookie is presented, these major websites often display sensitive account information to the corresponding browser, giving an attacker that has procured the cookie complete access to the account. Select websites appear to perform some user identification, using the IP address as a proxy for location; for `linkedin.com`, this strategy is only present in the defense of the stolen credentials attack and does not resurface in the remote authentication cookie test.

This study is important because it identifies security implications of third-party tracking, adding

to existing privacy violations and highlighting the severity of this attack. It is also highly relevant to other discussions about user tracking and identifies a tradeoff between security and privacy in the use of these techniques. The testing infrastructure can be improved upon to perform similar studies on a wide array of sites, and more variables about browser fingerprinting and device identification can be tested using the same two crawler methodology.

We hope that our findings will prove beneficial to users by increasing awareness of session hijacking attacks and how they might be defended against. Websites need to understand that security considerations arise when using the standard cookie authentication model. Ultimately, we wish to encourage further development of secure user authentication systems for online service websites and to make these techniques easier to study.

References

[1] 41stParameter. (2014) Fraud prevention for account takeover. [Online]. Available: http://www.the41.com/solutions/account-takeover

[2] J. Bonneau *et al.*, "The past, present, and future of password-based authentication on the web," 2014.

[3] E. Butler. (2010) Firesheep. Available: http://codebutler.com/firesheep/

[4] E. Butler. (2010) Firesheep, a day later. Available: http://codebutler.com/firesheep-a-day-later/

[5] I. Dacosta *et al.*, "One-time cookies: Preventing session hijacking attacks with disposable credentials," 2011.

[6] P. Eckersley, "How unique is your web browser?" 2010.

[7] Kount. (2014) Applications - account takeover. Available: http://www.kount.com/use-cases/account-takeover

[8] N. Nikiforakis *et al.*, "Cookieless monster: Exploring the ecosystem of web-based device fingerprinting," 2013.

[9] C. Palmer and Y. Zhu. (2010) How to deploy https correctly. Available: https://www.eff.org/https-everywhere/deploying-https

[10] D. Reisman, "Cookie crumbs and unwelcome javascript: Evaluating the hidden privacy threats posed by the mashed-up web," 2014.

[11] D. Reisman *et al.*, "Cookies that give you away: Evaluating the surveillance implications of web tracking," 2014.

[12] K. Scarfone, W. Jansen, and M. Tracy, "Guide to general server security," 2008.

[13] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Eng. Bull.*, vol. 24, no. 4, pp. 35–43, 2001.

[14] ThreatMetrix, "Prevent account takeover," 2014.

[15] Y. Zhou and D. Evans, "Why aren't http-only cookies more widely deployed?" 2010.

A. Appendix

A.1. Results For Each Website

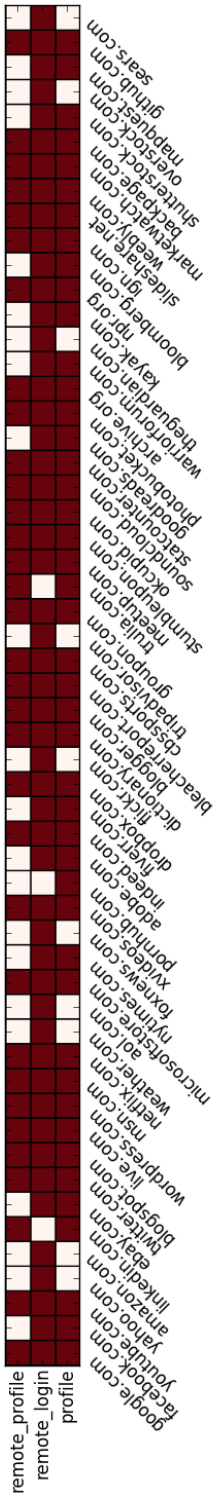


Figure 5: Filled boxes indicate successful secondary crawler login. Empty boxes are inconclusive results. The rows $remote\_profile$, $remote\_login$ and $profile$ correspond to 4.4, 4.3 and 4.2, respectively.