A Study of the Multiple Sign-in Feature in Web Applications

Marwan Albahar[1], Xing Gao[2], Gaby Dagher[1], Daiping Liu[3], Fengwei Zhang[4], and Jidong Xiao[1]

[1]*Boise State University*
[2]*University of Memphis*
[3]*Palo Alto Networks*
[4]*SUSTech and Wayne State University*

Abstract. Nowadays, more and more web applications start to offer the multiple sign-in feature, allowing users to sign into multiple accounts simultaneously from the same browser. This feature significantly improves user experience. Unfortunately, if such a feature is not designed and implemented properly, it could lead to security, privacy, or usability issues. In this paper, we perform the first comprehensive study of the multiple sign-in feature among various web applications, including Google, Dropbox. Our results show that the problem is quite worrisome. All analyzed products that provide the multiple sign-in feature either suffer from potential security/privacy threats or are sacrificing usability to some extent. We present all issues found in these applications, and analyze the root cause by identifying four different implementation models. Finally, based on our analysis results, we design a client-side proof-of-concept solution, called *G-Remember*, to mitigate these issues. Our experiments show that *G-Remember* can successfully provide adequate context information for web servers to recognize users' intended accounts, and thus effectively address the presented multiple sign-in threat.

1 Introduction

Historically, most websites allowed users to access only one account at any given time using the same browser. As a result, users who needed to access multiple accounts (e.g., personal and business) at the same time from the same machine had to either use different browsers, or use some browser extensions [4]. In the past decade, the *multiple sign-in feature* was introduced as a solution to this problem, which enables users to sign in simultaneously using multiple accounts from the same browser. e.g., Google started offering this feature in 2010 [1]. Since then, many other well-known web applications have started to offer this feature, including Dropbox, Yahoo, Twitter, and Instagram.

However, as of now, there is no standard that defines expected behaviour for safe and secure multiple-account access, and how cookies should be shared among multiple accounts. As a result, the design and implementation of this feature varies from one web application to another. In this paper, we attempt to fill this gap by analyzing how web applications differentiate among multiple accounts connected from the same browser. To the best of our knowledge, this is the first work that studies the design and implementation details of the multiple sign-in feature in major web applications. One major finding of our study is that most products provided by Google and Dropbox lack sufficient isolation and are not able to differentiate among multiple accounts connected from the same browser, which could lead to:

1. usability issues. When the user attempts to access some web resource (R) via one account (A), which has the proper permission to access R, yet the web server mistakenly thinks the user is using another account (B), which does not have the proper permission to access R. The user's access would therefore be denied. This is a mistake, and could hurt user experience.

2. security and privacy issues. When the multiple sign-in feature is used in conjunction with capability-based access control, the problem is exacerbated. More specifically, when the capability-based access control is used, and the server fails to differentiate among the user's multiple accounts, the consequence is, one account could interfere with another account. This includes peeking into another account, gaining extra access to undisclosed information belonging to another account, or even altering the contents in files belonging to another account.

After analyzing corresponding web traffic (i.e., http requests and responses), we have identified the root cause of why the web server is not able to differentiate among the user's multiple accounts. When the user clicks a URL to access certain web resource, the web server oftentimes could not recognize the user's accounts because the context information (i.e., information about a specific account) included within the http request is inadequate. To address this problem, we have implemented a client-side proof-of-concept solution to force users to provide necessary context information such that the web server is able to identify which account it is currently dealing with. With such a solution, the aforementioned usability, security, and privacy issue would be solved.

1.1 Contribution

The contributions of this paper can be summarized as follows:

– We conduct the first systematic analysis of the design and implementation of the multiple sign-in feature among different web applications. We identify four different implementation models used by different web applications to implement the multiple sign-in feature, and we discuss why they are, or are not, able to differentiate among multiple accounts.

- For web applications that fail to differentiate among users' multiple accounts, we present what usability, security, and privacy problems could happen. We also report our major findings with respect to Google and Dropbox applications. Specifically, for the first time, we define and report a problem that we call it the **cross-account information leakage problem**.
- We implement a client side proof-of-concept solution, that includes a browser extension, to help users provide context information for the web server. Our experimental results show that our solution enables web servers to grant clients access resources using the correct account, thus avoid the aforementioned usability, security, and privacy issues.

2 Background

2.1 Multiple Accounts

When multiple accounts are involved, cookies become more complicated. Some cookies are shared among multiple accounts, while others are non-shared and bound to a specific account. Take Google's products as an example. Shared cookies usually have the domain attribute as `google.com` and the path attribute as "/". By contrast, the domain attribute of a non-shared cookie is more specific and typically includes more subdomain information. For instance, most cookies related to Gmail accounts have the domain attribute of `mail.google.com`, and their paths are longer, like `/mail/u/0`, `/mail/u/1`, `/mail/u/2`, etc. The numbers 0, 1, and 2 denote the login order of this account, with the first signed-in Gmail account's cookies having the path of `/mail/u/0`, the second one having `/mail/u/1`, and so forth. As we will explain in Section **??**, it is because Gmail uses separate cookies and create separate URLs for different accounts that make it possible to differentiate among a user's multiple accounts connecting from the same browser. Unfortunately, most other Google products do not have such a separation. The consequence of this is Google fails to differentiate among users' multiple accounts. The same problem also occurs in Dropbox.

2.2 Capability-based Access Control: Sharing a file via a Link

Online storage service products such as Google Drive, Dropbox, Microsoft Onedrive, typically support two classic access control mechanisms: access control list (ACL) and capability-based access control. Both mechanisms provide secure access controls, and to the best of our knowledge, there is no literature proving that one is more secure than the other. However, in this work, we identify that, when the multiple sign-in feature is used in conjunction with the capability-based access control, security problems could happen.

More specifically, most problems we have identified happen when the user has multiple accounts signed in and one file is shared via a link - whoever has the link can access the file. This file could be stored in a Google drive, or in a Dropbox folder. The link is the capability in the context of capability-based access control, while in other situations, the capability could be a token, ticket, or a key [12], which gives a subject an access to an object. To ensure such a capability is not extended to a untrustworthy person, on the one hand, the owner should try to keep the link privately and only share it with a trustworthy party; On the other hand, service providers typically make such a capability hard to predict. For example, a typical Google document URL includes a randomly generated string of more than 40 characters, which makes such URLs almost unguessable to a random person. As far as we know, there is no existing literature or reports showing any evidence that such long URLs, when used in the context of HTTPS (which is exactly what Google and Dropbox have adopted), can be exploited by attackers. Furthermore, to the best of our knowledge, Google or Dropbox have not, in any of their documents, told their users that such long URLs are insecure. Therefore, it is reasonable and understandable for people to share files in such a manner - whoever has the link can access the file. In the remaining part of this paper, we assume all the file sharing situations we talk about refers to this type of sharing. Also, in Google Drive, with such a sharing method, when an account accesses a file shared by another account, the shared file would be automatically saved in this destination account's Google Drive.

In summary: our observation - web applications fail to differentiate among users' multiple accounts, plus the fact - when a shared file (shared via the above method) is accessed by another account, the shared file will be automatically saved in the other account's online drive, could cause several security problems, as described in the next section.

3 Threat Model

Overall, we consider the following three multiple sign-in scenarios where security or privacy problems exhibit. All of them involves some type of information leakage. In the following, we use Google Drive as an example, and we will present our findings in Dropboxin section 5. We use Alice to denote the victim, use **GP** to denote Alice's personal Google account, and use **GB** to denote Alice's business Google account.

– **Classic cross site request forgery (CSRF) attack**. Considering an attacker Bob, who knows the victim Alice's personal email address **GP**, but has no knowledge of Alice's work email address **GB**. We also assume at work, Alice shares that email address with several other co-workers - meaning they all have access to **GB**'s Google Drive storage. Now let us say Bob has some Alice's sensitive (e.g., sexual) videos or pictures, and he wants to distribute these videos or pictures to Alice's co-workers, but he doesn't know how to get their contact information. To achieve his malicious goal, Bob could send a link to Alice's A1 account and share the videos or pictures with Alice. And if Alice has both **GP** and **GB** account active running in the same browser, and then she opens the link (from within its **GP** account's inbox) - just as every other CSRF attack instance, the victim needs to have its account

active and has to click on the link or access some web page which includes the link. Unfortunately, Google thinks it's **GB** attempting to open the link - In section 4, we will explain why and when Google would think this way. Thus, the moment Alice opens the videos or the pictures, the videos and/or pictures will be automatically saved in **GB**'s drive. As we just mentioned, **GB** is a shared email address for work. Therefore, on a different day, Alice's co-worker Eva, or any other co-workers, signs in to **GB** from a different computer or device, would be still able to view those videos and pictures. This describes a classic CSRF attack scenario.
This same attack could also be performed so as to help attackers to spread malicious programs or virus, or ransomware, which might require more social engineering tricks. Note that even if Bob is not a bad actor - for example, Bob is Alice's friend, and is just sharing some private files between two friends, the fact that Alice's private information being automatically saved in her business account Google Drive, is still a problem. In the following, we call this the **cross-account information leakage problem**.

– **Information leakage from one user's account to another user's account.** Alice signs her personal Google account from a public or shared device, on which Bob already has one of his accounts signed in from the same browser. This case frequently occurs in public devices, like a desktop in a public library. It could also happen in a professional talk or conference presentation, where the speaker oftentimes has to login his/her account in other people's laptops (e.g., laptops provided by the conference organizer or the session chair) so as to get presentation materials from his/her email box or some online drive space. Note that in this scenario, Bob has no bad or malicious intentions, yet the cross-account information leakage problem could unexpectedly expose Alice's sensitive information or data to Bob. In the example of Google drive, once Alice visits her Google drive documents, Alice's documents could be automatically saved in Bob's Google drive folder without Alice's knowledge.
– **User's one account is hacked, while other accounts are NOT hacked.** User Alice signs multiple personal accounts, and one of which was hacked by the attacker Bob (e.g., the account and password are leaked). In recent years, credential leaking has been not rare: a dark web leaks 1.4 billion leaked passwords in 2017 [2]; twitter exposes the passwords of 330 million users in plain text [5]; and 272 million email username/password combinations are possessed by hackers in 2016 [3]. We assume the victim Alice sets different passwords for her different accounts, as this is a very basic security practice. Thus, except for the compromised account, Bob should not be able to directly obtain information from Alice's other accounts. Yet, once again, in the example of Google drive, once Alice accesses her data on Google drive, her data could be automatically saved in that compromised account's Google drive folder, which is under Bob's control.

4 Google Multiple Accounts

Our study shows most Google products fail to differentiate among multiple accounts, although Gmail is an exception. In this section, we specifically use Google Drive as a case study.

Google Drive Google *Drive* provides a file synchronization and storage service which empowers users to share and synchronize files across different devices. It also allows file sharing across different users. When sharing files with others, for each file, the owner can set the permission, indicating whether other users can view or edit the file. The owner can then send a link of the file to other people, and a common sharing scenario is whoever has the link can access the file. The problem happens when the receiver has more than one account active in the same browser, and he/she intends to use one of his/her accounts to click the link and open the file. Since such link usually does not contain any context information, Google is unable to decide the intended account, and thus opens the file with some account at its choice (i.e., the Google-chosen account) - as opposed to the user's (i.e., the user-chosen account) choice.

Determine the Google-chosen account. To determine which account would be the Google-chosen account, we further conduct experiments to understand the implicit policy used by the Google server side. We first register three regular Google accounts with the *gmail.com* domain. We also have three *Gmail* education (formerly known as G Suite for Education) accounts with the *.edu* domain. We use one regular account (denoted as G_s) and one *.edu* account (denoted as E_s) to share the file. Other accounts are signed in on the receiving end from the same browser, denoted as G_1, G_2, E_1, and E_2, respectively. We use different accounts to share the file, and change the sign-in sequence of testing accounts, to understand the policy for the *default account* (i.e., the Google-chosen account). The results are listed in Table 1.

We first share the file using the regular account G_s. We change the log-in sequence of other two regular accounts G_1, G_2, and find that the **first** log-in account is always used to open the file (acting as the *Google-chosen account*). This is also the same case when we sign in two *.edu* accounts, E_1 and E_2: the first log-in account is the *Google-chosen account.* However, if we sign in one regular account and one *.edu* account, the regular account with *.gmail.com* domain will always be the *Google-chosen account.* Changing the log-in sequence will not affect the *Google-chosen account* here.

We then share the file using an *.edu* account E_s, and repeat the experiments. In this case, the first log-in account will always be the *Google-chosen account*, even if two different types of accounts are signed in (e.g., E_1 and G_1). The policy is implemented on the Google server side, thus obscure to users. From our experiments, we find that this policy depends on the log-in sequence and types of accounts.

Security implications. As can be seen from our experimental results in table 1, among the 16 sharing experiments, in 50% of the experiments, the

Table 1. Experiments on Google Drive to Determining the Google-chosen accounts

File sharing		Log-in sequence		Google-chosen account
From	User-chosen account	First	Second	
G_s	G_1	$\mathbf{G_1}$	G_2	$\mathbf{G_1}$
G_s	G_2	$\mathbf{G_1}$	G_2	$\mathbf{G_1}$
G_s	G_1	$\mathbf{G_2}$	G_1	$\mathbf{G_2}$
G_s	G_2	$\mathbf{G_2}$	G_1	$\mathbf{G_2}$
G_s	E_1	$\mathbf{E_1}$	E_2	$\mathbf{E_1}$
G_s	E_2	$\mathbf{E_1}$	E_2	$\mathbf{E_1}$
G_s	E_1	$\mathbf{E_2}$	E_1	$\mathbf{E_2}$
G_s	E_2	$\mathbf{E_2}$	E_1	$\mathbf{E_2}$
G_s	G_1	$\mathbf{G_1}$	E_1	$\mathbf{G_1}$
G_s	E_1	$\mathbf{G_1}$	E_1	$\mathbf{G_1}$
G_s	G_1	E_1	$\mathbf{G_1}$	$\mathbf{G_1}$
G_s	E_1	E_1	$\mathbf{G_1}$	$\mathbf{G_1}$
E_s	G_1	$\mathbf{G_1}$	E_1	$\mathbf{G_1}$
E_s	E_1	$\mathbf{G_1}$	E_1	$\mathbf{G_1}$
E_s	G_1	$\mathbf{E_1}$	G_1	$\mathbf{E_1}$
E_s	E_1	$\mathbf{E_1}$	G_1	$\mathbf{E_1}$

Google-chosen account is not the user-chosen account. Meaning Google wrongly chose an account that is not what the user intended, and this could lead to security or privacy problems. Once the shared document is opened by the *Google-chosen account*, this document will be recorded in that account's history. Even if this *Google-chosen account* is later on signed in from another device, the file is still accessible to the account. The user with control of the *Google-chosen account* can then get the information, or even tamper the file if this file is shared with write permission. In particular, suppose Eva shares a file by sending Alice's education account with a sharable link. The file is accessible by anyone who knows the link, but the link is kept privately by Eva. In this case, without knowing the link, other users are still unable to read or write the file. However, Bob successfully signs his *Gmail* account in the Alice's machine through the third cases mentioned in the threat model (Section 3). As mentioned before, the regular *Gmail* account with the *gmail.com* domain will become the *Google-chosen account.* As a result, when Alice clicks the link, Bob will get the the access of the target file. In other words, the multiple sign-in feature, when used in conjunction with the capability-based access control, could cause a file to be shared to an user against the recipient's will and without the recipient's knowledge.

Table 2. Experiments on Dropbox File Sharing to Determining the Dropbox-chosen accounts

File sharing	Log-in sequence		Sharing manner	Dropbox-chosen account
User-chosen account	First	Second		
D_B	$\mathbf{D_B}$	D_P	link, view file	$\mathbf{D_B}$
D_P	$\mathbf{D_B}$	D_P	link, view file	$\mathbf{D_B}$
D_B	$\mathbf{D_B}$	D_P	invite people, view folder	$\mathbf{D_B}$
D_P	D_B	$\mathbf{D_P}$	invite people, view folder	$\mathbf{D_P}$
D_B	$\mathbf{D_B}$	D_P	invite people, view file	$\mathbf{D_B}$
D_P	D_B	$\mathbf{D_P}$	invite people, view file	$\mathbf{D_P}$
D_B	D_P	$\mathbf{D_B}$	invite people, view folder	$\mathbf{D_B}$
D_P	$\mathbf{D_P}$	D_B	invite people, view folder	$\mathbf{D_P}$
D_B	D_P	$\mathbf{D_B}$	invite people, view file	$\mathbf{D_B}$
D_P	$\mathbf{D_P}$	D_B	invite people, view file	$\mathbf{D_P}$
D_B	D_B	$\mathbf{D_P}$	invite people, request file	$\mathbf{D_P}$
D_P	D_B	$\mathbf{D_P}$	invite people, request file	$\mathbf{D_P}$
D_B	D_B	$\mathbf{D_P}$	link, request file	$\mathbf{D_P}$
D_P	D_B	$\mathbf{D_P}$	link, request file	$\mathbf{D_P}$
D_B	$\mathbf{D_P}$	D_B	invite people, request file	$\mathbf{D_P}$
D_P	$\mathbf{D_P}$	D_B	invite people, request file	$\mathbf{D_P}$
D_B	$\mathbf{D_P}$	D_B	invite people, request file	$\mathbf{D_P}$
D_P	$\mathbf{D_P}$	D_B	invite people, request file	$\mathbf{D_P}$

5 Dropbox Multiple Accounts

5.1 How Dropbox Multiple Accounts Works

Dropbox is mainly for online storage sharing. Dropbox allows users to have a personal account and a business account; users can have both accounts active in the same browser. Users can access their person account by visiting the URL https://www.dropbox.com/personal and access their business account by visiting the URL https://www.dropbox.com/work.

Dropbox uses a cookie called "`Last_active_role`" to record the **last active** account, which could be the personal account, or the business account. For example, when both pages are open, if the user refreshes the personal account page, the personal account will be considered as the last active account; if the user then refreshes the business account page, the business account will become the last active account.

5.2 Main Problem

The main problem of Dropbox occurs when resource sharing is happening. At the time of this study, Dropbox supports three types of resource sharing: regular file, paper, showcase. We perform various experiments to measure each of these three services, and we find several issues. Since both Dropbox and Google Drive are storage sharing products, most problems we identify in Dropbox are similar to those problems in Google Drive. They exhibit in a similar manner: i.e., there is a mismatch between the user-chosen account and the server-chosen account. In the following, we use the term "*Dropbox-chosen* account" to represent this server-chosen account. We also define the "user-chosen account" in the Dropbox context as follows: in order to use Dropbox, users need to register an account with their email address. Thus, each Dropbox account is essentially bound with an email address. Therefore, the "user-chosen account" in the Dropbox context is similar to the Google Drive situation - the resource recipient opens the resource from within its email box. We conduct different experiments to determine the Dropbox-chosen account. In the following, we use **D1** to denote the business account, and **D2** to denote the personal account.

Dropbox File Similar to Google drive, Dropbox allows users to share files in different ways. The owner can generate a link and send the link to the recipient over either an email or an instant message. The owner can also select "invite people", which will automatically constructs an email to notify the recipient. Users can either share a single file, or a folder containing multiple files. We also notice there is a feature called "request file", which allows a user to request a file from another user. We test all of these scenarios and record our results in Table 2. As illustrated in the table, it can be seen that in nearly 30% of situations the Dropbox-chosen account does not match with the user-chosen account.

Dropbox Paper Dropbox *paper* is a paper collaboration service, which allows multiple people to edit the same paper simultaneously. Dropbox *paper* allows users to send an invitation to collaborators, and the owner can specify whether the recipient should have the edit permission or just the comment permission. We test both of these two scenarios and record our results in Table 3. As can be seen from Table 3, when sharing a paper with someone who has two accounts alive from the same browser, no matter which account is the paper shared to, the personal account will always be used to access to the paper.

We also notice another interesting and surprising issue with the Dropbox *paper* feature. When a paper is shared with a business account in the comment mode only, if the business account is active with a personal account from the same browser, the personal account will get the permission of both commenting and editing.

Dropbox Showcase Dropbox *showcase* is a service that allows users to share a project to other people on a single page. Similarly, Dropbox *showcase* allows users to either send an invitation to other people, or send a link to other people. For both cases, the recipient can make comments about the shared project. We test both scenarios and record our results in Table 4.

Table 3. Dropbox Paper Sharing

File sharing	Log-in sequence		Sharing manner	Dropbox-chosen account
User-chosen account	First	Second		
D_B	D_B	$\mathbf{D_P}$	invite people, edit	$\mathbf{D_P}$
D_P	D_B	$\mathbf{D_P}$	invite people, edit	$\mathbf{D_P}$
D_B	D_B	$\mathbf{D_P}$	invite people, comment	$\mathbf{D_P}$
D_P	D_B	$\mathbf{D_P}$	invite people, comment	$\mathbf{D_P}$
D_B	$\mathbf{D_P}$	D_B	invite people, edit	$\mathbf{D_P}$
D_P	$\mathbf{D_P}$	D_B	invite people, edit	$\mathbf{D_P}$
D_B	$\mathbf{D_P}$	D_B	invite people, comment	$\mathbf{D_P}$
D_P	$\mathbf{D_P}$	D_B	invite people, comment	$\mathbf{D_P}$

Table 4. Dropbox Showcase Sharing

File sharing	Log-in sequence		Sharing manner	Dropbox-chosen account
User-chosen account	First	Second		
D_B	$\mathbf{D_B}$	D_P	invite people	$\mathbf{D_B}$
D_P	$\mathbf{D_B}$	D_P	invite people	$\mathbf{D_B}$
D_B	$\mathbf{D_B}$	D_P	link	$\mathbf{D_B}$
D_P	$\mathbf{D_B}$	D_P	link	$\mathbf{D_B}$
D_B	D_P	$\mathbf{D_B}$	invite people	$\mathbf{D_B}$
D_P	D_P	$\mathbf{D_B}$	invite people	$\mathbf{D_B}$
D_B	D_P	$\mathbf{D_B}$	link	$\mathbf{D_B}$
D_P	D_P	$\mathbf{D_B}$	link	$\mathbf{D_B}$

The main insights we gain from this set of experiment is, when we share a showcase with someone who has two accounts (one business and one personal account) alive from the same browser, the *Dropbox-chosen* account is always the last active account. In other words, when the recipient opens the showcase, the last active account will always be used to open the showcase - and when we were performing the experiments for Table 4, the last active account was the business account.

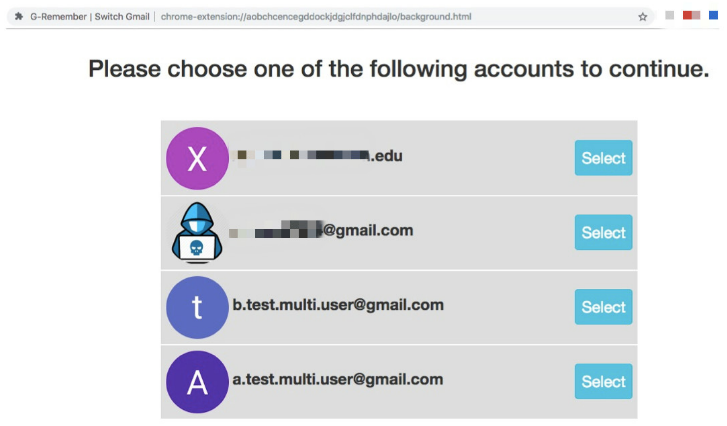
Privacy implications. For all the three main services Dropbox provides, as we can see, there is always a decent chance (30% to 50%) that the Dropbox-chosen account is not the user-chosen account. This could lead to some privacy leakage issue. Next we will describe an example in which this privacy leakage issue could hurt Dropbox users. Let us suppose user Bob wants to share a business file with his co-worker Alice. Bob creates a link, and sends this link to Alice via email. Alice has two Dropbox accounts signed in the same browser. The link goes to her business account email box, and she tries to open it, but Dropbox does not know it is opened from her business account email box. So a Dropbox-chosen account will be used, which at this moment could happen to be her personal account. So the file will be opened from her personal account, and she does not realize this. After viewing the file, she writes a comment about this file - and this comment will be visible to everybody in the business group - meaning that everybody would see Alice's personal account and its profile picture. This could lead to two problems. First, Alice would be embarrassed if her personal profile picture is an inappropriate picture. Second, Alice might be violating the company's policy and be punished for using her personal account to access business resources. Yet in her defense, she does not expect any of her personal account's information to be exposed to her colleagues or business partners, and she does not have any intention to access business resources with her personal account. The whole procedure happens without her knowledge.

6 Defense

The root cause of all the problems we have identified in the multiple sign-in process is due to the shortcomings in current cookies mechanism. When a shared link is opened in a browser where multiple accounts have signed-in, there is insufficient context information in the cookies about the accounts. Specifically, if the link is clicked inside one account (e.g., *Gmail*), the server side has to open the *default account* (e.g. Google-chosen or Dropbox-chosen) because the current cookie mechanism lacks the account information. This could be solved by enhancing servers with new cookies containing accounts information. However, if a link is clicked outside web browsers, the browsers will not be able to know which account should be used to open this link. For instance, a user simply copies and pastes a Google *Drive* link to the browser's address bar. In this case, the server side has no idea on which account is the "correct" account. Therefore, it would open the link with a *default account*, which might not be the user's intended account. As a result, we argue that this account selection procedure must be in some way explicitly delegated to users.

6.1 Server Side Defense

Ideally, such a delegation mechanism should be implemented by the service providers, i.e., Google, Dropbox, etc. We have reported the issues we found to Google. After reading our report, the corresponding security team at Google told us that our finding is surprising, but instead of fixing the security issues, they stressed that the capability-based access control should not be used to share a document if the document is very confidential. We do not agree with Google, as no existing research or literature has shown that the capability-based access control is less secure than the ACL based access control. As we have stated before, the security issues do not manifest just because of the capability-based access control, they arise when both the multiple sign-in feature and the capability-based access control are used.


G-Remember | Switch Gmail | chrome-extension://aobchcencegddockjdgjclfdnphdajlo/background.html

Please choose one of the following accounts to continue.

X	[redacted].edu	Select
	[redacted]@gmail.com	Select
t	b.test.multi.user@gmail.com	Select
A	a.test.multi.user@gmail.com	Select

Fig. 1. A snapshot of G-Remember.

6.2 Client Side Defense

Since the server side is out of our control, we consider to demonstrate that the issue could also be mitigated from the client side, even though that is not ideal. We propose a proof-of-concept solution for Google products by allowing end-users to make the decision on which account should further proceed upon opening links without context information. We call our solution *G-Remember*, which is a browser extension, and is implemented with JavaScripts+HTML+JSON. *G-Remember* collects all accounts' information, intercepts URLs, and reproduces the link sent to remote server by adding some user input (account choice). As a result, *G-Remember* enables the user to choose the appropriate account. Specifically, *G-Remember* consists of four parts.

– First, *G-Remember* collects the account information by recording a unique identifier (e.g., session index), the profile picture, as well as the email address. The unique identifier works as a trusted parameter for verifying accounts' identities. The email address and the profile picture will be presented to the user for account selection. All of these pieces of information are automatically collected while the user signs in.
– When a link is opened, *G-Remember* intercepts the HTTP request, extracts and analyzes the URL information to determine the target service and product. This function is accomplished by keyword and structure matching.
– After figuring out the target product, *G-Remember* will display a customized web page with all of the accounts' details (e.g., picture and email address from step one) included, and ask the user to select one account to proceed.
– Finally, after the user's selection, *G-Remember* inserts the the unique identifier in the correct place in the URL, and sends corresponding request to the remote server. Figure 1 shows a snapshot of *G-Remember* using our tested emails.

Our experimental results show that *G-Remember* enables web clients to send context information to the Google web server, and the Google server is therefore able to recognize the intended account. During our experiments, we consistently observe that the Google-chosen account matches with the user-chosen account. As a proof-of-concept solution, *G-Remember* supports Google products only, but it is trivial to extend its support for other companies such as Dropbox's products.

7 Related Work

To the best of our knowledge, we are the first to study cookie issues in the context of multiple accounts. Our work is related to web security, especially cookies related security issues.

Cookies enable web servers to store the states of clients, and thus they are widely used by first-party and third-party websites [14]. Previous large-scale measurements [11, 8] found that cookies in practice are much more sophisticated than the standard. Ill-managed cookies could be exploited by attackers to obtain private data or track users. As a result, cookies have attracted numerous attention. Sivakorn et al. [16] presented a comprehensive study on the HTTP cookie hijacking attack. They showed that such attacks not only disclose private and sensitive information, but also can gain access to protected account functionality. Historiographer [9] demonstrates that the web search history of Google users could be reconstructed from the personalized suggestions. Englehardt et al. [10] showed that third-party cookies could be used as unique identifiers to track users even with different IP addresses. Even worse, cookies are prone to be leaked due to cross-site scripting (XSS) attacks [15]. To mitigate the risk of client side scripts accessing cookies, HTTP-only cookies are introduced. Unfortunately, Zhou et al. [17] demonstrated that such a mechanism cannot completely eliminate XSS vulnerabilities. Cookies are also widely used as fingerprinting to track users [6, 7].Mendoza et al. [13] studied the inconsistencies between mobile and desktop HTTP security response, and showed that the inconsistencies on the same website can cause various vulnerabilities. Our work also studies the fragility in existing cookies design. However, we focus on the scenarios of multiple-accounts, which have not been studied in previous work.

8 Conclusion

In this paper we study the multiple sign-in feature in several web applications, including Google, Dropbox, Yahoo and Postman. We identify four different models used by web applications to implement the multiple sign-in feature, and report various security, privacy, and usability concerns regarding its implementation in Google and Dropbox applications. We investigate the root cause and present a proof-of-concept client solution to alleviate these concerns. Until the service providers fix the problems on the server side, we recommend users to be cautious when using those web services that provide the multiple sign-in feature.

References

1. Access two gmail accounts at once in the same browser. https://gmail.googleblog.com/2010/08/access-two-gmail-accounts-at-once-in.html.
2. File with 1.4 billion hacked and leaked passwords found on the dark web. https://www.forbes.com/sites/leemathews/2017/12/11/billion-hacked-passwords-dark-web/1d2ef9ec21f2.
3. Hold security recovers 272 million stolen credentials from a collector. https://holdsecurity.com/news/the$_{c}ollector_{b}reach$/.
4. Sessionbox. https://sessionbox.io/discover.
5. Twitter advising all 330 million users to change passwords after bug exposed them in plain text. https://www.theverge.com/2018/5/3/17316684/twitter-password-bug-security-flaw-exposed-change-now.
6. Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 674–689. ACM, 2014.
7. Gunes Acar, Marc Juarez, Nick Nikiforakis, Claudia Diaz, Seda Gürses, Frank Piessens, and Bart Preneel. Fpdetective: dusting the web for fingerprinters. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 1129–1140. ACM, 2013.
8. Aaron Cahn, Scott Alfeld, Paul Barford, and S Muthukrishnan. An empirical study of web cookies. In *Proceedings of the 25th International Conference on World Wide Web*, pages 891–901. International World Wide Web Conferences Steering Committee, 2016.
9. Claude Castelluccia, Emiliano De Cristofaro, and Daniele Perito. Private Information Disclosure from Web Searches. In *International Symposium on Privacy Enhancing Technologies Symposium (PETS)*, 2010.
10. Steven Englehardt, Dillon Reisman, Christian Eubank, Peter Zimmerman, Jonathan Mayer, Arvind Narayanan, and Edward W Felten. Cookies That Give You Away: The Surveillance Implications of Web Tracking. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*, 2015.
11. Roberto Gonzalez, Lili Jiang, Mohamed Ahmed, Miriam Marciel, Ruben Cuevas, Hassan Metwalley, and Saverio Niccolini. The Cookie Recipe: Untangling the Use of Cookies in the Wild. In *2017 IEEE Network Traffic Measurement and Analysis Conference*, 2017.
12. Henry M Levy. *Capability-based computer systems.* Digital Press, 2014.
13. Abner Mendoza, Phakpoom Chinprutthiwong, and Guofei Gu. Uncovering HTTP Header Inconsistencies and the Impact on Desktop/Mobile Websites. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web (WWW)*, 2018.
14. Franziska Roesner, Tadayoshi Kohno, and David Wetherall. Detecting and Defending against Third-Party Tracking on the Web. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation (NSDI)*, 2012.
15. Kapil Singh, Alexander Moshchuk, Helen J Wang, and Wenke Lee. On the Incoherencies in Web Browser Access Control Policies. In *2010 IEEE Symposium on Security and Privacy (SP)*, 2010.
16. Suphannee Sivakorn, Jason Polakis, and Angelos D Keromytis. Cookie hijacking in the wild : Security and privacy implications. BlackHat, 2016.
17. Yuchen Zhou and David Evans. Why aren't http-only cookies more widely deployed. *Proceedings of 4th Web 2.0 Security and Privacy*, 2010.