2017 IEEE 4th International Conference on Cyber Security and Cloud Computing

Security Analysis of Email Systems

Tianlin Li Amish Mehta Ping Yang
Computer Science, State University of New York at Binghamton
Email: {tli16,pyang}@binghamton.edu

***Abstract*—Electronic mail (email) is universally used by businesses, government agencies, and individual users. Out of necessity, users trust their email systems to keep their emails safe and secure. However, email systems are often complex and exhaustive testing is almost impossible for such systems. As a result, email systems often contain bugs and security vulnerabilities. In this paper, we analyze the security and usability of five popular public email systems. Our analysis shows that there are several security vulnerabilities in multiple sign-in and password composition and recovery policy of some of the email systems.**

Keywords: Email, Security

I. Introduction

Electronic mail (email) is universally used by businesses, governments, and individuals. Out of necessity, users trust their email systems to keep their messages safe and secure. Due to the sensitive nature of email systems, most email vendors rigorously test their email systems for bugs and security flaws. However, email systems are often complex and exhaustive testing is almost impossible for such systems. As a result, email systems often contain bugs and/or security vulnerabilities. For example, in 2011, a bug in Gmail reset around 150,000 user accounts, which resulted in the loss of years worth of data [21]. In the same year, another bug in Gmail erased emails, chat history, labels, folders and all personal settings [14].

In this paper, we analyze the security and usability of five popular public email systems: Gmail (Bmail), Hotmail, Yahoo, 163, and Sohu. Sohu and 163 are widely used in China. Bmail is Binghamton university's email system serviced by Google, which was developed using Google Apps Education Edition. Gmail and Bmail had different login interface before January 2014. Since 2014, Gmail and Bmail used the same login interface, which fixed a vulnerability in their multiple sign-in mechanism presented in this paper. We use Bmail-old and Bmail-new to refer to Bmail versions before and after 2014, respectively.

This paper analyzes the security and usability of password management and multiple sign-in in the five email systems. In particular, we aim to answer the following questions: Does the multiple sign-in mechanism compromise the confidentiality of email systems? Do email systems provide strong password composition policies? Can an attacker discover other users' passwords using email systems' password recovery mechanism? Can confidential data such as passwords, email contents, and email subjects be stored on the disk? All results reported in this paper were obtained on machines running Windows or Linux before May 2016.

Our analysis is based on the following *assumptions and threat model*. We assume that an attacker may eavesdrop over the network, and forge or replay HTTP requests and responses. An attacker may use the computer of her friends and family members for a short period of time. We also assume that not all email users are computer security experts. After a user checks emails or uses other services provided by an email system, the user may simply close the web browser instead of logging out of her email account. Our past experience shows that this assumption is reasonable: We noticed that some users (including some Computer Science professionals) assumed that closing a web browser would automatically log users out from their email accounts. Finally, not all users would clear the browser's cache before they lend their computers to others.

While email security has been extensively studied in past literature (see Section V), our contributions are in reporting specific analysis results regarding the security of the five email service providers that have not been previously reported. We have also developed tools to automatically analyze whether an email system satisfies specific security properties. Our contributions are summarized below:

- We have performed a detailed comparison of password composition, password transmission, and password recovery/reset mechanisms in the five email service providers.
- We have developed algorithms and tools to analyze the security and usability of the multiple sign-in mechanism in Gmail and 163. Our tools found that Gmail and Bmail-old accounts may be linked together without users' consent, and data in Gmail and Bmail accounts may be mixed up without users' knowledge.
- We have analyzed the security of the login page of the five email systems. We found that 163's login page is vulnerable to a man-in-the-middle attack and Sohu provides users an option of using HTTP to log into their email accounts.
- We have analyzed whether passwords, email subjects, and email contents may be stored on the disk, which would prolong the lifetime of such data. Our analysis shows that email subjects of Gmail, Yahoo, Sohu, and 163 may be stored on the disk. In 163 and Sohu, the password is also stored on the disk, even if the user does not choose the option of remembering the password.

II. Password Management

Password is the front line of defense against attackers. Consequently, password management is critical to the security of email systems. In this section, we analyze the security of password management policies in the five email systems.

A. Password Composition Policy

Email accounts are often hacked due to weak passwords. Therefore, email systems should ideally enforce strong pass-

978-1-5090-6644-5/17 $31.00 © 2017 IEEE
DOI 10.1109/CSCloud.2017.20

TABLE I
PASSWORD COMPOSITION POLICY OF GMAIL, YAHOO, HOTMAIL, SOHU, AND 163.

Name	Password length	Are all digits allowed?	Are all lower-case letters allowed?	Are adjacent keys allowed	Can password and ID be the same	Can password contain ID/user name?	Can password be the permutation of ID?
Gmail	≥ 8	yes	yes	no	no	yes	yes
Yahoo	$8--32$	yes	yes	yes	no	no	yes
Hotmail	≥ 8	no	no	no	no	yes, but can only contain user name	yes
Sohu	6-16	yes	yes	yes	no	yes	yes
163	6-16	yes	yes	yes	no	yes	yes

word composition policy to prevent users from providing weak passwords. The following passwords are generally considered weak: (1) short passwords, (2) passwords containing all digits, all lower-case letters, or all capital letters; (3) passwords containing all adjacent keys, and (4) passwords containing dictionary words [23]. Table I compares the password composition policy in Gmail, Yahoo, Hotmail, Sohu, and 163. The table shows that Gmail, Yahoo, Sohu, and 163 allow weak passwords, such as passwords containing all digits or all lower-case letters. We notice that Google provides tips for creating a strong password [1], but Gmail does not enforce such a policy. Hotmail has the strongest password composition policy, and Sohu and 163 have the weakest policy.

B. Secure Password Transmission

To prevent users' passwords from being captured by attackers, the login form of email systems should be protected using HTTPS. Table II answers the following questions: (1) Does the login page use HTTP or HTTPS? (2) Does the login page use POST or GET method to send the password? and (3) Is the password encrypted while being transmitted to the email server? The table shows that the login page of Gmail, Yahoo, and Hotmail use HTTPS while the login page of 163 uses HTTP. Sohu provides an option of using HTTP or HTTPS.

TABLE II
SECURITY OF LOGIN PAGES IN FIVE EMAIL SYSTEMS.

Name	HTTP/HTTPS	POST/GET	Password is encrypted?
Gmail	HTTPS	POST	yes
Yahoo	HTTPS	POST	yes
Hotmail	HTTPS	POST	yes
Sohu: interface (1)	HTTP	GET	no(hashed)
Sohu: interface (2)	HTTPS	GET	yes
Sohu: interface(3)	HTTPS	POST	yes
163	HTTP	POST	yes

Sohu provides three login interfaces: (1) "http://mail.sohu.com", (2) "https://mail.sohu.com", and (3) "https://passport.sohu.com". The level of security provided by these three interfaces is different. Interface (1) uses HTTP and GET to send a user's ID and hashed password (hashed using MD5) to the email server. Because the login page uses HTTP, an attacker can capture the hashed password by eavesdropping over the network, which enables the attacker to log into Sohu as shown below. First, we created a sohu account, logged into the account, and used Fiddler to capture the ID and the hashed password. Next, we provided two random strings as ID and password, respectively. We then submitted the ID and password, intercepted the corresponding HTTP request, and replaced ID and password in the request with the ID and the hashed password captured. After that, we successfully logged into the account. A video demonstrating the above attack is available at https://www.youtube.com/watch?v=LBYP-ZfgBaM.

The login page of 163 uses HTTP, but the ID and the password are sent to the server through SSL (Secure Socket Layer), which is vulnerable to the *man-in-the-middle attack* given below. First, an attacker forges a page that looks the same as the login page of 163, except that after the user logs in, the ID and the password are sent to the attacker's server. When a user requests to log into 163, the attacker intercepts the request and sends the forged page to the user. After the user enters the ID and the password in the forged page, the ID and the password are sent to the attacker.

C. Password Recovery Mechanism

The password recovery mechanism enables users who forget their passwords to recover passwords. Password recovery, if not designed properly, may enable unauthorized users to reset passwords or prevent authorized users from resetting passwords. For example, in 2014, attackers gained access to several celebrities' iCloud accounts using email addresses and a search engine (by searching for the date of birth of celebrities and the answers to two security questions) [6]. In this section, we analyze the security and usability of the password recovery mechanism in five email service providers.

Password recovery through email/phone: All five email service providers enable users to recover passwords through emails or phones. When recovering the password through emails, a link for resetting the password is sent to the recovery email account. When using phone to recover the password, a token is sent to the phone. Table III shows whether a user is required to provide a recovery email account or a phone number when creating an account (column 2), and how a user recovers the password through email or phone (columns 3 and 4). The terminologies used in the table are given below.

- Optional: Require neither email nor phone number.
- Either: Require either email or phone number.
- Phone: Require only the phone number.
- LinkWithID (LinkWithoutID): The email ID is (is not) displayed after clicking the password recovery link.
- TokenFresh (TokenNonFresh): The token sent to the phone is (is not) expired in 30 minutes.
- Unknown: we do not know the answer because a Chinese phone number is needed to recover the password.

TABLE III
THE EMAIL RECOVERY MECHANISM IN FIVE EMAIL SYSTEMS.

Name	Email/Phone required	Email recovery	Phone recovery
Gmail	Optional	LinkWithoutID email contains ID	6-digit TokenNonFresh
Yahoo	Phone	LinkWithoutID	7-letter TokenFresh
Hotmail	Either	Send token to email	7-digit TokenFresh
Sohu	Phone	LinkWithID	Unknown
163	Optional	LinkWithoutID	6-digit TokenFresh

In Sohu, if a user has provided an alternative email account, then a link for resetting the password is sent to the alternative account. After clicking the link, the user ID is shown on the webpage. In Gmail, the user ID is shown in the email containing the link. Both approaches are insecure: if the email system does not use end-to-end encryption, an attacker may be able to obtain the email containing the link, and hence can reset the user's password and log into the user's account. In other email systems, the user ID is not shown in the email or after clicking the link, which means that an attacker can only reset the password, but cannot log into the account (as the attacker does not know the ID).

When using phone to recover the password, the token expires in less than 30 minutes for all email systems except Gmail. Gmail's token is valid for at least 90 minutes.

Password recovery through online appeal: In Gmail, Hotmail, 163, and Sohu, a user can recover the password by answering a number of questions online. In Yahoo, a user needs to call the customer service to recover the password.

In Gmail, a user needs to answer two required questions: "When was the last time you were able to sign in to your Google Account?", "When did you create your Google Account?", and several optional questions to recover her password. One of the authors successfully reset the password of her relative's Gmail account (with consent from her relative) by providing approximate answers to only the two required questions. She did not know when her relative's account was created, so she entered the date in the first email she received from her relative. She also entered a recent date as the last date when the email was read (the actual date is the date when she reset the password). It seems that the IP address plays an important role: she was able to reset the password at her relative's home, but not from an IP address that her relative has never used. Therefore, Gmail's password recovery mechanism may allow family members or friends of a user to reset the user's password. In addition, the password reset link is valid for around two days, and hence an attacker can reset the password within two days after receiving the link (from any IP address). We are also able to recover the password of a newly created Gmail account from any IP address.

Hotmail's password recovery mechanism is vulnerable to the following attack. First, an attacker sends a number of emails to her friend over a period of time and then picks subjects of three emails that her friend replied. Next, the attacker tries to reset her friend's account by providing her friend's first name, last name, birth date, country, state, zip code, and the subject of three emails collected. We have successfully reset a Hotmail account created by us using the above steps. It appears that the IP address plays an important role in resetting the password: we are only able to recover the account with an IP address from which the account was accessed earlier. Hotmail also prohibits a user from resetting the password frequently.

In Sohu's password recovery mechanism, if a user has provided a security question when creating her Sohu account (which is optional), the user is prompted to answer the security question. If the answer is correct, then the user is prompted to enter a new password. Otherwise, the user can choose to appeal online. When appealing online, the user is prompted to provide her name and date of birth, and an email account for receiving the password. The user is also asked to upload a copy of her Chinese identity card. Sohu then sends the information provided by the user to the server using the HTTP protocol. Sohu's password recovery mechanism has the following drawbacks. First, sending users' information through HTTP connection is insecure. Second, Sohu does not ask users to provide their chinese ID during registration. Although a user can upload her Chinese identity card through Sohu's security settings, many users are not aware of this feature, which makes Sohu's online appealing system unusable. 163's online appealing also requires the user to enter her Chinese ID, which has the same weakness as Sohu.

D. Password Reset Mechanism

An email user can change her password if her password is compromised or used for long time. Table IV answers the following questions in the five email systems: (1) Does the password reset page use HTTP or HTTPS? (2) Can the new password be the same as a password used previously? (3) What information is needed to reset the password?

In Gmail and Yahoo, a user needs to enter both her current and new passwords in order to reset her password. After resetting the password, an email notifying the change of password will be sent to the user's recovery email account, if such an account is provided. Sohu's password reset mechanism is similar to that of Gmail and Yahoo, except that Sohu does not check whether the new password is the same as a previously used password.

To reset the password in 163, a user needs to select three security questions among 11 questions provided by 163. 163 does not allow the user to define her own security questions. In addition, it is possible for an attacker to find the answers to some of the security questions online if the attacker knows the owner of the email account. Such questions include e.g. "Your spouse/mother/father/kid's name". If the user answers security questions correctly, then a 6-digit code will be sent to the user's cellphone. After entering the code, the user can change the password.

In Hotmail, when a user resets the password, a 7-digit code is sent to the user's phone or alternative email account. After the user enters the code, the user can change the password. The code expires in less than 30 minutes and cannot be reused.

III. ANALYSIS OF MULTIPLE SIGN-IN MECHANISM

It is common for a user to have multiple email accounts. For example, a user may use one email account for business/work,

TABLE IV
THE PASSWORD RESET MECHANISM IN FIVE EMAIL SYSTEMS.

Email	HTTP/HTTPS?	Can new password be the same as old password?	Information needed for reseting the password
Gmail	HTTPS	no	current password
Yahoo	HTTPS	no	current password
Hotmail	HTTPS	no	current password, has access to either the alternative email account or the phone of the user
Sohu	HTTP	yes	current password
163	HTTPs	no	current password, has access to the phone of the user

another for personal use, and a third account for purchasing goods online from commercial websites. A user may want to sign into all accounts simultaneously in order to access all her emails in a timely manner. If two email accounts of a user belong to different service providers (e.g. one is Hotmail and the other is Gmail), then the user can sign into both accounts using the same browser. Otherwise, the user needs to use two different browsers to sign into each email account unless the email system supports the multiple sign-in mechanism.

Gmail/Bmail allows users to sign into multiple Gmail/Bmail accounts simultaneously using the same web browser by clicking the button "Add Account". 163 also supports multiple sign-in. Hotmail used to support multiple sign-in, which was replaced with alias in 2013 due to security concerns [7]. Yahoo does not support multiple sign-in.

While multiple sign-in may improve the usability of email systems, it also complicates the design of email systems and may introduce new security vulnerabilities. Multiple sign-in should satisfy the following security requirements: (1) Files sent to one email account should not be stored in another account. (2) Multiple sign-in should not compromise the usability of email systems. For example, multiple sign-in should not prevent users from using services that can be accessed without multiple sign-in. In this section, we analyze whether the above two properties hold in Gmail, which supports multiple sign-in.

A. Adding Two Accounts Without Users' Consent

In email systems that support multiple sign-in, two email accounts should not be added together without users' consent. It is impractical for email users to manually check whether the above holds periodically. To address this issue, we developed an algorithm to automatically detect whether two Gmail accounts are added together without users' consent. The pseudocode is given in Algorithm 1.

We use Fiddler to capture HTTPS sessions, each of which comprises an HTTP request and an HTTP response, and contains the host address and the URL of the request. We notice that, when a user signs in or logs out from her Gmail account, the host address of the session is "accounts.google.com". When a user signs in her Gmail account, the HTTPS response header contains the string "Google-Accounts-SignIn:$email = xxx@gmail.com$, $sessionindex = yy$", where $xxx@gmail.com$ is the user's email address and $sessionindex$ represents session index, which is used to identify the Gmail account that the user has signed in. The value of $sessionindex$ of the first account signed in is 0. Everytime the user adds an account, the value of $sessionindex$ increases by 1. We also notice that, when a user clicks "Add Account", the URL of the HTTP response contains "/AddSession". When a user logs out from her Gmail accounts, the URL contains "/Logout" and the HTTP response header contains "Google-Accounts-SignOut". In addition, when a user clicks the "Log Out" button, she is logged out from all her Gmail accounts.

Algorithm 1 Multiple sign-in without users' consent.

```
1:  AddSession = SessionIndex = 0;
2:  for each https session s do
3:      if s.host == "accounts.google.com", s.url contains "/AddSession", and s.response.header contains both "email" and "sessionindex" then
4:          AddSession++;
5:      else if s.url doesn't contain "Logout?" and s.response.header contains both "email" and "sessionindex" then
6:          SessionIndex++;
7:          if SessionIndex > AddSession+1 then
8:              error "Accounts are added together"
9:          end if
10:     else if s.url contains "Logout?" and s.response.header contains both "email" and "sessionindex" then
11:         AddSession = SessionIndex = 0;
12:     end if
13: end for
```

In Algorithm 1, we use two variables – $AddSession$ and $SessionIndex$ – to record the number of times a user clicks "Add Account" and the number of email accounts the user has signed in, respectively. Initially, both variables have value 0. When a user clicks "Add Account", $AddSession$ increases by 1 (lines 3–4). When a user signs in an email account, $SessionIndex$ increases by 1 (lines 5–6). When a user logs out from one of her email accounts (lines 10-12), all email accounts are logged out automatically. Thus, the value of both $SessionIndex$ and $AddSession$ becomes 0. If $SessionIndex$ is greater than $AddSession + 1$, then the algorithm reports that two accounts have been added together without users' consent (lines 7–9).

To automate the analysis process, we prepared a file that contains the URLs of Gmail, Bmail, and other Google services (Youtube, Google+, Google blog, etc.). Our tool automatically started two tabs of the Chrome browser, picked two URLs stored in the file, opened the corresponding web pages, signed into two different Gmail (Bmail) accounts, and checked whether two Gmail (Bmail) accounts are add together without

users' consent.

Results: We found that when a user performed the following actions, two email accounts gc_1 and bc_1 were added together without explicitly clicking the "Add Account" button.

1) Log into www.youtube.com using Gmail account gc_1.
2) Log into Bmail-old using account bc_1.

A video showing the above violation is given at https://www.youtube.com/watch?v=dFAdOfVJjwo. Bmail-new does not add the two accounts after the above actions.

B. Storing Files in Wrong Google Drive

A Gmail user U_1 can share a Google Doc file with another user U_2 by emailing the URL of the file to U_2 or clicking the "share" button to share the file with U_2. Below, we present an algorithm for automatically checking whether the file opened in one Google account is stored in the Google Drive of another account. We refer to the account from which a user clicks a URL/file as "source account" and the Google Drive account where the file is stored as "destination account".

Algorithm 2 Algorithm for checking whether a file is stored in wrong Google Drive

```
1: for each HTTPS session s do
2:   if s.host == "accounts.google.com" and s.response.header
     contains "Google-Accounts-SignIn" then
3:     sindex = session index in s.response.header;
4:     emails[sindex]= email address in s.response.header;
5:   end if
6:   if s.host == "mail.google.com", s.request contains
     "req0_focused=1", and s.url contains "mail/u/si" then
7:     sourceindex = si;
8:   end if
9:   if s.host == "docs.google.com", s.response.header con-
     tains "Content-Type" and "html", and s.response contains
     "Google Account:addr" then
10:    destination = sddr;
11:    source= emails[sourceindex];
12:    if source!=destination and source != null and
       destination != null then
13:      print "File is stored in different account.";
14:    end if
15:  end if
16: end for
```

Algorithm 2 gives the pseudocode for checking whether the source account and the destination account are the same. The source account is identified as follows. When a user logs into a Gmail account, the HTTPS response header contains "Google-Accounts-SignIn:email=xxx@gmail.com, sessionindex=yy", where $xxx@gmail.com$ is the user's email address and yy is the session index. We then record the email address and the session index (lines 2–5). If the user signs into multiple accounts using the same browser, then each email account is displayed in a separate tab. When the user opens an email, the email account in the currently active tab is the account from which the email is opened. We notice that, when we switch tabs of two emails accounts, the tab that is currently active sends an HTTP request containing "req0_focused=1". We then record the session index si of $mail/u/si$ in the URL of the request that contains "req0_focused=1" (line 7). The account with session index si is the source account.

The destination account is identified as follows: when a Google Doc file is opened in Google drive, the host of the session is "docs.google.com" and the HTTP response from the server contains a string "Google Account:". The destination account follows the string (lines 9–15).

Results: Our tool found that, When a user logs into both Gmail and Bmail (either Bmail-old or Bmail-new) accounts, the Google Doc file opened in Gmail will be stored in the Google Drive of Bmail if the link to the file contains the domain of Bmail, and vice versa. When a user signs into two Gmail (or two Bmail) accounts, then the file is always stored in the Google Drive of the first account the user logged in.

C. Effect of Multiple Sign-in On Gmail Services

Multiple sign-in should not prevent a user from using services provided by her email accounts. Bmail does not support several services provided by Gmail, including Maps, Youtube, Play, News, Google+, Translate, etc. We noticed that, when a user logs into Bmail and then Gmail using multiple sign-in, the user is not able to sign into Youtube using her Google account. In addition, when a user logs into both Bmail and Gmail accounts, only Bmail and Gmail accounts that are first signed in can use Google Doc.

IV. Leak of Emails on Disk

Email contents and subjects are considered confidential. Storing email contents and subjects on disk prolongs the lifetime of such data, but not necessarily improving the usability and performance of email systems. In this section, we present two tools for analyzing whether email subjects/contents may be stored on the disk.

a) IOMonitor: IOMonitor records a list of files created, deleted, moved, or modified by web browsers. It was implemented using Pyinotify [4] (Linux) and resource monitor [5] (Windows). We run IOMonitor while performing operations such as writing, reading, and sending emails, and then check if any files modified contain email subjects or contents.

TABLE V
Files that store email subjects in five email systems.

Name	Firefox	Google Chrome
Gmail	places.sqlite, sessionstore.js	Current session, Current tab, Last session, Last tab, History
Yahoo	No	Current session, last session
Hotmail	No	No
Sohu	places.sqlite, sessionstore.js	Current session, Current tab, Last session, Last tab, History
163	No	Current session, Current tab, Last session, last tab

We found that the password is stored in file "Current session" after a user logs into 163 and Sohu, even if the user does not choose to remember the password. Email subjects are stored in a number of files as shown in Table V, because email subjects are displayed in the title of the page or are included in the URL when sending emails.

b) CacheCrawler: Browser caching reduces the time on loading web pages by storing such pages in web browsers. Thumbnails are reduced-sized images of web pages stored

in cache memory, which enables quick access to the corresponding web pages. To prevent confidential data from being cached or stored in thumbnails, web developers can disable caching of a confidential page by including "no-cache, no-store" directives in its HTTP response header.

We developed a tool *CacheCrawler* to check whether all HTTPS pages of Gmail and Yahoo contain "no-cache" and "no-store" cache directives. CacheCrawler was developed using Java and Goldium [2]. CacheCrawler first checks if the login page contains these two directives and then prompts the user to enter her ID and password. This step is necessary because many HTTPS pages can be accessed only after logging into the account. Once CacheCrawler logs into the account, it recursively crawls all web pages and checks if the cache-control response headers of the pages contain "no-cache" and " no-store". Our tool found that most HTTPs pages in Gmail and Yahoo have "no-cache" and "no-store" directives. We have manually checked web pages that do not have such directives and found that none of them contain confidential data. The login page of Hotmail has neither "no-cache" nor "no-store" cache directive. This means that the ID and the length of the password may be cached or stored in the thumbnail. CacheCrawl is not able to log into Hotmail, possibly because Microsoft blocks automated services, such as Goldium. Sohu and 163 use HTTP for all pages.

V. Related Work

A number of vulnerabilities in email systems were reported in the past. In 2008, Noiumkar et al. showed that Gmail, Inbox Mail and Hotmail were vulnerable to session hijacking attacks [18]. In 2013, Castillo [13] discovered a cross-site scripting vulnerability in the mail attachment feature of Gmail on iOS. Hafif [19] discovered cross-site request forgery vulnerabilities in Gmail's password reset and recovery mechanism in 2013. Goodman [9] showed that an attacker can bypass Google's two-step login verification and reset a user's master password by capturing the user's application-specific password. In 2006, Grossman [15] discovered a vulnerability in Gmail, which enables an attacker to access the contact list of users' Gmail accounts. In 2011, a vulnerability in Hotmail was discovered, which enables an attacker to steal a user's emails and contacts when the user opens a malicious email sent from the attacker [3]. In 2012, Prince [17] reported that a hacker successfully accessed a customer's account on CloudFlare, which uses Google Apps for email, and changed that customer's DNS records. Yahoo also disclosed that more than one billion Yahoo user accounts have been hacked in August 2013 [8].

Several researchers have analyzed the security of email systems. Foster et al. [12] analyzed whether email providers support TLS at each point in the email message path. Schechter et al. [22] analyzed the security of using security questions to recover accounts in four webmail providers. Pandove et al. [20] discussed a number of existing email security issues and provided strategies to protect users' emails. Li et al. [16] identified a few vulnerabilities of several web-based password manager, which may lead to the stolen of users' credentials. Argyros et al. [10] discovered randomness vulnerabilities in password reset token generation of PHP. Clair et al. [11] stated that it may be possible to recover 8-character random passwords via the brute force attack with powerful cluster machines. None of the above work reported the vulnerabilities discovered in this paper.

VI. Conclusion

This paper analyzes the security of five popular public email service providers with focus on password management and multiple sign-in. Our analysis results show that several email systems have security vulnerabilities.

References

[1] Creating a strong password. https://support.google.com/accounts/answer/32040?rd=2.

[2] Goldium. https://code.google.com/p/goldium/.

[3] Hotmail vulnerability for stealing e-mail. http://www.spamfighter.com/News-16229-Hotmail-Vulnerability-for-Stealing-E-Mail.htm.

[4] Pyinotify: monitor filesystem events with python under linux. http://pyinotify.sourceforge.net.

[5] Resource monitor. https://en.wikipedia.org/wiki/Resource_Monitor.

[6] This is how easy it is to hack someone's icloud with their security questions. https://www.washingtonpost.com/news/the-intersect/wp/2014/09/03/this-is-how-easy-it-is-to-hack-someones-icloud-with-their-security-questions/.

[7] Use outlook.com's aliases to hide your true email address from prying eyes. http://www.pcworld.com/article/2095505/use-outlook-coms-aliases-to-hide-your-true-email-address-from-prying-eyes.html.

[8] Yahoo discloses hack of 1 billion accounts. https://techcrunch.com/2016/12/14/yahoo-discloses-hack-of-1-billion-accounts/.

[9] Adam Goodman. Bypassing google's two-factor authentication. https://www.duosecurity.com/blog/bypassing-googles-two-factor-authentication.

[10] G. Argyros and A. Kiayias. I forgot your password: Randomness attacks against php applications. In *USENIX Security*, pages 81–96, 2012.

[11] L. S. Clair, L. Johansen, W. Enck, M. Pirretti, P. Traynor, P. McDaniel, and T. Jaeger. Password exhaustion: Predicting the end of password usefulness. In *Information Systems Security*, pages 37–55, 2006.

[12] I. Foster, J. Larson, M. Masich, A. C. Snoeren, S. Savage, and K. Levchenko. Security by any other name: On the effectiveness of provider based email security. In *CCS '15*, 2015.

[13] Glenn Darmanin. Xss vulnerability injected through google analytics, executed in ios's gmail application. http://www.acunetix.com/blog/web-security-zone/articles/xss-vulnerability-injected-google-analytics-executed-ioss-gmail-application/.

[14] IBTimes Staff Reporter. Google fixing gmail bug. http://www.ibtimes.com/google-fixing-gmail-bug-271359.

[15] Jeremiah Grossman. Advanced web attack techniques using gmail. http://jeremiahgrossman.blogspot.com/2006/01/advanced-web-attack-techniques-using.html.

[16] Z. Li, W. He, D. Akhawe, and D. Song. The emperor's new password manager: Security analysis of web-based password managers. In *USENIX Security Symposium*, pages 465–479, 2014.

[17] Matthew Prince. Post mortem: Today's attack; apparent google apps/gmail vulnerability; and how to protect yourself. http://blog.cloudflare.com/post-mortem-todays-attack-apparent-google-app.

[18] P. Noiumkar and T. Chomsiri. Top 10 free web-mail security test using session hijacking. In *International Conference on Convergence and Hybrid Information Technology*, pages 486–490, 2008.

[19] Oren Hafif. Google account recovery vulnerability. http://www.orenh.com/2013/11/google-account-recovery-vulnerability.html.

[20] K. Pandove, A. Jindal, and R. Kumar. Email security. *International Journal of Computer Applications*, 5(1):23–26, August 2010.

[21] Paul McNamara. Google apologizes for gmail bug that shook 150,000 users. http://www.networkworld.com/news/2011/030111-google-gmail-apology.html.

[22] S. Schechter, A. J. B. Brush, and S. Egelman. It's no secret: Measuring the security and reliability of authentication via 'secret' questions. In *IEEE Symposium on Security and Privacy*, 2009.

[23] J. J. Yan. A note on proactive password checking. In *Proceedings of the 2001 Workshop on New Security Paradigms*, pages 127–135, 2001.