Code Security of Mobile Backup Modules on the Android Platform

Abstract. While many activities of business and daily lives are depending on mobile devices, more and more official files and party photos may be created or viewed in these devices. These files and photos are usually backuped to a remote service, e.g., Dropbox, Baidu Cloud, OneDrive, Goolge Drive. Applications in Android usually use a third-party SDK of one of above services to simplify these development. When the modules of these services are more and more widely called, the security of these modules are critical now. In this paper, we try to empirically study how widely these services are being used in the Android applications. Then, we investigate the differences of code security about four mainstream Android cloud storage SDKs: Dropbox SDK, Baidu Cloud SDK, OneDrive SDK and Google Drive SDKs. We analyze usage, protocol and API implement in these SDKs, and find that none of them encrypt users' files before data transmission and some of them adopt a client-side deduplication which could cause serious vulnerabilities. Based on this, after performing three security issues in today's cloud storage SDK calls, we apply the countermeasures for third-party developers to call the cloud storage SDK securely.

Key words: Mobile Cloud Storage, Code Security, Dropbox, Baidu Cloud, OneDrive, Google Drive

1 Introduction

As cloud computing becomes prevalent, cloud storage has received a lot of publicity. In recent years, cloud storage has been emerged as a novel kind of network service that provides high amount of storage space at effective cost and enables people to upload, download, control and manage their data via requesting a remote server. Current trends show an increasing number of individuals and enterprises storing their data to cloud storage services with lower energy consumption and higher resource utilization rate. Until now, there have been various cloud storage providers, such as Amazon, Microsoft, Dropbox, Google, Baidu, etc., and have appeared more and more Commercial cloud storage services, e.g., Dropbox [1], Baidu Cloud [2], OneDrive [3], Google Drive [4].

[1] https://www.dropbox.com/
[2] https://yun.baidu.com/
[3] https://onedrive.live.com/
[4] https://www.google.com/drive/

Nowadays, with the popularity of the cloud storage as well as the ubiquity of the Android smartphone, most cloud storage providers have developed mobile cloud storage applications which can support mainstream mobile systems, such as Android and iOS. People can backup their photos, address books, chat records and other sensitive data in cloud servers to mitigate the storage burden of their smart phone and synchronize their data across platforms. Besides, large quantities of applications have introduced the cloud storage service to communicate with the above given service. This allows people to make use of their personal cloud storage data via a third-party application. In order to help the third-party application development, such cloud services oftentimes provide a framework, or a SDK (Software Development Kit) [1].

However, we find that these SDKs developed by cloud storage providers have inherent coding issues. In March 2015, a severe vulnerability(identified as CVE-2014-8889) was revealed that could lead to local or remote attacks within the Dropbox SDK for Android version 1.5.4-16.1 [1]. Specifically, this vulnerability allows an attacker to insert an access token into the AuthActivity which is used for authentication. On the other hand, third-party developers may call these SDKs incorrectly even though those applications are in normal running, but there may exist undetected attacks. For instance, we found that some applications expose the app key and secret which are the equivalent of a client access token in string.xml or AndroidManifest.xml file, while an attacker can get this sensitive information easily by decompiling the apk file. Thus, the third-party applications with cloud storage SDK call could meet up with many potential challenges. This allows us to analyze four current mainstream cloud storage services from multiple aspects to discuss code security of mobile backup modules on the Android platform.

In this paper, we perform a multi-level security study across four different Android cloud storage SDK: Dropbox SDK, Baidu Cloud SDK, OneDrive SDK and Google Drive SDK. To the best of our knowledge, this is the first work on analyzing usage, protocol and API implement in these SDKs. Especially, we find some security issues which may occur in the processing of the cloud storage SDK calls by third-party developers or in the SDK itself. This allows us to propose a series of coding tips for Android developers. Our contributions in this paper can be summarized as follows:

- We perform a usage analysis on Google Play Store and Cool Store about the cloud storage SDK calls, and find that some of the third-party application developers do not follow the official development guidelines, which could cause code security issues that we summarized in this paper.
- We conduct protocol analysis based on the above four mainstream cloud storage SDKs, which include four aspects: communication protocol, encryption protocol, deduplication protocol and authentication protocol. The results show that there exist some differences in the protocols across these SDKs, while none of them encrypt users' file before data transmission and some of them adopt a client-side deduplication which could cause security issues.

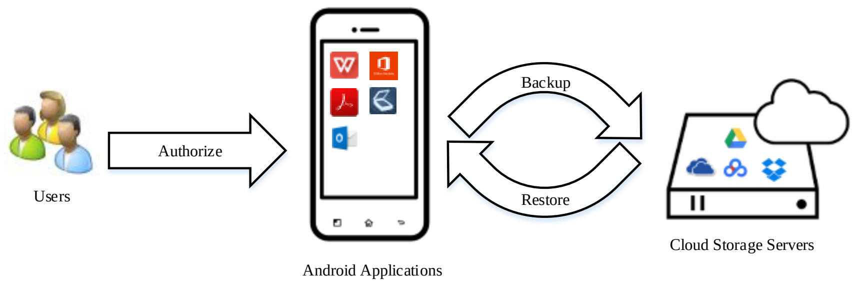

Fig. 1. Mobile Backup Storage Models

- We investigate 50 open-source Android third-party applications which call the cloud storage SDKs. Based on this, we present several coding issues by cloud storage SDK developers as well as third-party developers. This allows us to propose a series of countermeasures for developers to avoid call risk and data leakage as much as possible.

The rest of the paper is organized as follows. Section 2 introduces background as well as our research motivation. Section 3 and Section 4 present deployment investigation and code analysis across four different cloud storage SDKs on the Android platform. Section 5 performs some existing code security issues in today's cloud storage SDK calls as well as the countermeasures for third-party developers. Section 6 summarizes related work and Section 7 concludes the paper with some future works.

2 Background and Motivation

2.1 Mobile Backup Storage

To release local storage space, cloud storage service has emerged as a novel network storage platform, which can be used to create online backups of local files. Additionally, users have access to their files via computers or smartphones once the network is connected. As shown in Figure 1, users could authorize third-party Android applications, and then backup their files to cloud storage servers as well as restore the data from these servers. Dropbox, OneDrive, Google Drive and Baidu Cloud are among the most popular cloud storage service at the moment.

Dropbox has served as a file hosting service operated by American company Dropbox, Inc. since 2007, and offers file synchronization, storage service, and client software to users [2]. By March 2016, Dropbox service has hit 500 million users from more than 200 countries with 1.2 billion files uploaded daily to it. Besides, there have been 300,000 apps built on the Dropbox platform [5].

[5] http://expandedramblings.com/index.php/dropbox-statistics/

In 2008, Microsoft developed OneDrive (previously SkyDrive, Windows Live SkyDrive, and Windows Live Folders) as a file hosting service that allows users to sync files and later access them from a web browser, desktop client or mobile device [3]. There have been more than 250 million registers use OneDrive as a cloud storage server by 2016 [6].

Google Drive, which was launched in 2012, is a file storage and synchronization service created by Google [4]. Over 190 million people have used Google Drive to store their personal data until 2016 [7].

In 2012, Baidu Cloud was launched as Baidu WangPan. Today Baidu Cloud offers a cloud storage service, client software, file management, resources sharing, and Third Party Integration to users and developers [5]. The statistics show that more than 200 million users have registered in Baidu Cloud by November 2014[8].

These four cloud storage services provide mobile cloud storage SDKs for third-party developers. SDK is a library that provides easy access to cloud s-torage services. Developers can import into their own application project after downloading. These SDK frameworks hide the internal abstract implementation details, and provide succinct client-side API for application developers. This is very attractive to developers, and meanwhile, it has greatly facilitated the users' operation and management.

2.2 Motivation

Due to the astonishing popularity of Android smartphones, a large amount of sensitive information have access to various cloud storage services by individuals and enterprises. Based on this, cloud storage providers have developed their SDKs to simplify the development of third-party applications. We perform an empirical study on how widely these SDKs are being used in the Android applications in the following section 3. The results of the statistics indicate that cloud storage SDK calls are now in common use. However, during the investigation, we find that some of third-party developers did not follow the official development guidelines even though that application is in normal running. Note that there exist inherent code security issues in the SDKs as well [1, 6]. As far as we know, it can produce lots of sensitive data during the communication between third-party applications and cloud storage servers, but improper cloud storage SDK developments and calls may cause a batch of serious security problems with privacy violations and data leakage. Therefore, it is essential and meaningful to analyze code security of mobile backup modules on the Android platform to remind both third-party and cloud storage developers to avoid vulnerability.

[6] http://expandedramblings.com/index.php/microsoft-statistics/
[7] http://expandedramblings.com/index.php/google-advertising-statistics/#.u6d4muyiq5i
[8] http://expandedramblings.com/index.php/baidu-stats/5/

3 Investigation of Deployment

In order to analyze security issues in the mainstream cloud storage services, we first count the usage of several different Android cloud storage SDKs in third-party applications. By investigating developer guidelines and simples from various cloud storage official websites, such as Dropbox [9], Baidu Cloud [10], OneDrive [11] and Google Drive [12], we provide statistics as to the prevalence of the use of above SDKs in Android.

Table 1. Cloud storage SDK usage distributions on Google Play Store and Cool Store

Cloud storage service	**Google Pay Store (top 430 apps)**	**Cool Store (top 500 apps)**
Dropbox	12(2.79%)	49(9.80%)
Google Drive	15(3.49%)	29(5.80%)
OneDrive	4(0.93%)	11(2.20%)
Baidu Cloud	0(0.00%)	3(0.60%)
Box	3(0.70%)	13(2.60%)

As shown in Table 1, we examine the top 430 applications in the Google Play Store [13] which serves as the official application store for the Android operating system. According to these statistics, the Google Drive SDK usage is the commonest (3.49%), closely followed by Dropbox SDK usage (2.79%), while 0.93% of application installations of the Google Play top 430 applications use the OneDrive SDK. Oddly, Baidu Cloud SDK has not been used by third-party applications based on our investigation results. Further, we choose Cool Store which is one of the most popular android application markets in China [14] as another tested market. Statistics suggest that Dropbox SDK has the most usage with 9.80% of app installations of the top 500 apps in Cool Store, and Google Drive takes the second most (5.80%), while the usage percentage of OneDrive SDK is 2.20%. Additionally, 0.60% of app installations use Baidu Cloud SDK. We think the major reason for this might be the popularity of Baidu in China. In total, there are 5.35% of application installations of the top 430 applications on Google Play Store as well as 18.40% of the top 500 applications on Cool Store. We note that most of these third-party applications enable to use across platforms and serve as file managers, document editors, file scanners, image editors, notebooks and calendars, etc.

[9] https://www.dropbox.com/developers
[10] https://developer.baidu.com/
[11] https://dev.onedrive.com/
[12] https://developers.google.com/drive/
[13] https://play.google.com/store
[14] http://www.coolapk.com/

4 Code Analysis

To learn about the functions of the four storage servers, we analyze their API by referring to official documents and SDK source code. As shown in Table 2, file management is the basic function which contains CRUD (create retrieve, update, and delete) operations and all servers support this function. In addition to this function, there are some other features which help simplify operations and improve system performance. Quota management allows users to query their storage space, it can help them use the remaining space reasonably or expand the capacity if necessary, and the four providers all have this function. Another useful function is batch process which enables users upload and download files in bulk. According to our survey, only Baidu Cloud has this function in their SDK, while the other three servers do not provide this function directly and third-party developers have to implement it themselves. Advanced features, which include getting file thumbnails, transcoding video files, transferring a large file instantly, off-line downloading, etc., improve the user experience. As a result, we find that Baidu Cloud is very comprehensive in these aspects, while Google Drive, Dropbox, OneDrive focus on the basic functions and pay no attention to the advanced functions as well as batch process.

Table 2. API analysis in Android SDKs of cloud storage

Cloud service	**Google Drive**	**Dropbox**	**OneDrive**	**Baidu Cloud**
File Management	Yes	Yes	Yes	Yes
Quota Management	Yes	Yes	Yes	Yes
Batch Process	No	No	No	Yes
Advanced Features	No	No	No	Yes

In order to make the whole communication process effective and secure, cloud providers apply a set of protocols to stipulate the rules between the third-party applications and cloud provider servers. We conduct a comprehensive investigation of several important protocols used by the four cloud storage services and made a comparison among them. Especially, we choose four protocols, which are communication protocol, encryption protocol, deduplication protocol and authentication protocol.

Communication protocol defines rules that allow two or more entities of a communications system to transmit information via any kind of physical quantity [7]. Encryption protocol is the conversion of original data into another form, called cipher text, which cannot be easily understood by anyone except authorized parties [8]. Deduplication protocol refers to a technique that store only a single copy of redundant data, and provide links to that copy instead of storing other actual copies of this data [9]. Authentication protocol enables a third-party application to obtain limited access to an HTTP service [10]. Next, we will discuss the usage of those four protocols among Dropbox, Google Drive, OneDrive and Baidu Cloud. The security issues will be discussed as well. Table 3 shows the comparison of protocols usage among cloud storage services.

Table 3. Protocols usage among cloud storage services

Cloud service	Communication	Encryption	Deduplication	Authentication
Dropbox	HTTPS	No	No	OAuth 1.0/2.0
Google Drive	HTTPS	No	No	OAuth 2.0
OneDrive	HTTPS	No	No	OAuth 2.0
Baidu Cloud	HTTPS	No	Yes	OAuth 2.0

4.1 Communication Protocol Analysis

Although HTTPS imposes management and performance overhead, makes web contents non-cacheable, and introduces undesired side-effects such as browser warnings about mixed secure(HTTPS) and insecure(HTTP) content [11], the four cloud storage SDKs still use secure communication protocol(HTTPS) for all network requests according to our research. Note that ensuring data transmission security is the major factor during data communication, unencrypted communication among applications and cloud servers could be eavesdropped on the network. Thus, adopting HTTPS which causes these appropriate performance sacrifice is worthy for third-party developers.

4.2 Encryption Analysis

Since encryption is the essential way to protect the confidentiality of data stored on cloud servers and could mitigate the damage when the data is eavesdropped by attackers. As shown in Table 3, we find none of the four cloud storage SDKs encrypt data on the client side. There may be two reasons, one is HTTPS provides end-to-end protection, and is commonly suggested for mitigating attacks that manipulate network traffic, the other one is the encryption key is stored on the client side, if the key is breached and the encryption is compromised. Based on these concerns, cloud storage providers believe encryption is unnecessary.

However, it does not mean the data is absolutely secure by using HTTPS for network request. Actually, some researchers have found the weakness of android HTTPS protocol [12, 13], which may cause serious security risks. In most cases, users do not encrypt their data and encryption work should be handed over to third-party developers. Thus, in order to improve application security, integrating an encryption tool into the third-party application is a good idea, since it spends low overhead and can greatly enhance the security performance of users' personal data as well.

4.3 Deduplication Analysis

As cloud computing provides high amount of storage space, excessive amount of data being stored in the cloud. To avoid multiple copied of the same data, we need a specialized data compression technique, deduplication, which eliminates redundant data as well as improves storage and bandwidth utilization [14]. Based on this, almost all cloud storage providers implement deduplication for economic considerations.

There are two kinds of deduplication solutions, the target-based solution and source-based solution. The target-based solution handles deduplication on the server side, the client does not know the occurrence of deduplication. This kind of solution improves sever side storage utilization, but does not save bandwidth. On the other hand, the source-based solution does deduplication on client side before the data is transferred, so it can save both disk space and network bandwidth.

To confirm the deduplication solution of each cloud storage service, we conduct a experiment as follows:

1. create a new 550M encrypted file (to make sure the file is unique).
2. upload the file to Dropbox, Google Drive, OneDrive, Baidu Cloud separately.
3. monitor the network traffic and record the uploading time.
4. rename the encrypted file and upload it again to these four cloud storage servers.
5. compare the network traffic and uploading time with the first test.

The most notable feature of source-based deduplication solution is when you upload an existing file again, it uploads instantly. In our investigation, we confirm the deduplication solution of the four cloud storage platforms based on this feature. As shown in Table 3, the result shows that Dropbox, Google Drive and OneDrive use target-based deduplication solution, and Baidu Cloud adopts source-based deduplication solution. We also find that some cloud storage services handle deduplication at block level, instead of keep only a integrated single copy of each file. By monitoring the network traffic, we find that Dropbox divides the whole file into small pieces of 8MB, while OneDrive uses 160KB as a block size. Additionally, the block size of Baidu Cloud is 4MB. However, Google Drive does not apply the block level deduplication since it uploads the data at file level. In other words, for Google Drive, there are high failure rate in communication for big file, which could cause security issues, such as invisible data corruption.

Although deduplication has many benefits for both client and server, there are still some inherent risks. Especially for source-based deduplication, it can be exploited by an attacker to eavesdrop and disclose users' information. For instance, client-side deduplication can lead to side-channel attacks [9, 15], which allows attacker to learn who has stored a given file as well as profile users' usage of the service. Therefore, when third-party developers integrate a cloud storage SDK which adopts source-based deduplication into their Android application, it is better for them to prompt their users not to upload sensitive information or encrypt data before uploading. Otherwise, it may be breached.

4.4 Authentication Analysis

OAuth is an open standard for authorization, it enables a third-party application to obtain limited access to an HTTP service on behalf of the resource owners or on its own behalf [10]. Third party applications use OAuth to allow users authorize cloud storage service with their own cloud account, and then those applications have access to users' cloud resources. So far, OAuth exists two versions: OAuth1.0 and OAuth2.0. To simplify the authorization process, OAuth2.0 does a lot of improvements compared to OAuth1.0 that providing more OAuth flows to allow better support for non-browser based applications and no longer requires client applications to have cryptography [16]. Thus, some cloud storage providers obsolete OAuth1.0, such as Google Drive, One Drive and Baidu Cloud, while Dropbox still supports OAuth1.0 as an option with OAuth2.0 as the default.

The OAuth2.0 protocol has been proven secure by several approaches based on formal methods [17, 18, 19], and the OAuth working group write formal guidelines (OAuth Threat Model [20]) for developers to avoid potential risks which include eavesdrops sensitive information on the network traffic, XSS attacks, cross-site request forgery and so on [21] during developments. However, in practical situations, many developers do not follow the guidelines, they use an insecure way to transfer sensitive information, for instance, password, access token, etc. Based on this, we conduct an examination and try to find some sensitive information by monitoring the network requests. Surprisingly, we find that some applications put access token which contains the security credentials for a login session and identifies the user as an attribute into their http requests in plaintext. Consequently, attackers could get the access token easily by intercepting the http requests and use this access token to steal users' resources successfully. So we strongly suggest developers follow the official guidelines to avoid unnecessary data loss and leakage.

5 Security Issues and Countermeasures

Since more and more Android applications integrate a third party cloud storage SDK, the security issues of mobile backup modules can be critical now. However, due to the improper coding issues by cloud storage developers and incorrect usage of cloud storage SDKs by third-party developers, there still exist many security issues. By conducting a widely investigation, we discuss these code security issues of the cloud storage SDKs in this section. After that, in order to avoid these risks, we give the corresponding countermeasures to these developers.

5.1 Improper coding issues by cloud storage SDK developers

To simplify the application development, cloud storage services supply SDKs for third-party developers, which is a library that provides easy access to cloud services. Although using these SDKs can greatly reduce the development lifecycle and benefit users' operation, there are vulnerabilities which may be unintentional or purposive inside the SDKs all the time.

A recent study revealed a severe vulnerability (identified as CVE-2014-8889) within the Dropbox SDK for Android version 1.5.4-16.1 [1], it allows an attacker to insert an access token into the Dropbox SDK AuthActivity which is used for authentication. Since AuthActivity consumes an Intent extra parameter named INTERNAL_WEB_HOST [15] , this activity is exported and browsable and accepts arbitrary Intent extras from both malware and website. Based on this, an attacker could exploit this vulnerability by replace a users access token with his own access token. Consequently, when a user upload a given file, he actually uploaded this file into the attackers account.

Another security issue comes from Baidu and could be artificial. An Android SDK named Moplus [16] created by Baidu which has brought to the cusp for its security problems. According to the investigation by researchers, Moplus SDK has nothing to do with these vulnerabilities, it has backdoor functionality which allow attackers perform malicious behaviors, such as pushing phishing pages, sending SMS, uploading local files to remote servers, etc. Unfortunately, Baidu Cloud uses the Moplus SDK. Since installing Baidu Cloud is a prerequisite for third-party applications to integrate Baidu Cloud SDK [17], third-party applications which call Baidu Cloud SDK to backup users' data can also be in risk due to the vulnerability of Moplus SDK.

Note that it is unlikely to avoid these inherent security issues in the SDK for third-party developers, but when the vulnerability is discovered, cloud storage SDK developers always fix the loophole as soon as possible and release a new SDK version. Therefore, we strongly remind third-party developers to update their applications with the newest SDK periodically to avoid damage caused by these revealed security issues.

5.2 Incorrect usage of cloud SDK by Android third-party developers

Incorrect usage of cloud SDK by third-party developers is another potential risk while integrating cloud SDK into their applications. Since all cloud storage services have provided official guidelines to help third-party developers complete their development, and ensure data integrity in some extent, many third-party developers do not follow these security instructions because of the coding negligence, simplification and performance concerns during the development process, which could always lead to security risks according to our investigation. What's more, we give these developers some advice to make their applications more secure.

A. App_key and App_secret

[15] parameter INTERNAL_WEB_HOST controls the host that the browser surfs to.

[16] http://blog.trendmicro.com/trendlabs-security-intelligence/setting-the-record-straight-on-moplus-sdk-and-the-wormhole-vulnerability/

[17] if the device does not install Baidu Cloud, there will be a prompt to install it first or Baidu Cloud is disabled

Before integrating cloud storage SDK into an application, third-party developers should firstly register the application in the corresponding cloud storage service and they can receive a pair of app key and secret from cloud server. Usually, when an application wants to call the authorization interface, app key and app secret are required, so that the cloud storage service knows which application it is. However, if they are leaked, an attacker can use it to publish information, plus interest and other malicious actions. Since the cloud storage services could limit the permission of the applications when those abnormal behaviors are detected which can affect the normal use of the API calls, it is quite important for third-party developers to protect app key and app secret to ensure their applications working securely.

However, after examining 31 applications in the Google Play Store as well as 79 applications in China Cool Store [18], we find that some developers still do not follow the official guidelines. According to the result, 2 applications (6.4%) in the Google Play Store and 5 applications (6.3%) in the Cool Store expose the app key and secret in the string.xml or AndroidManifest.xml file. That means that an attacker can get this sensitive information easily by simply decompiling the apk file. To avoid this problem, it is better to code the app key and secret into the java file and obfuscate, while another effective method is keeping the app key and app secret on the server side.

B. Access Token

Access token is another sensitive parameter during the cloud storage SDK call, which contains the security credentials for a login session and user authentication. Generally, to avoid repeated authentication request, the applications always keep the access token locally to maintain the session state. However, where the access token stores and whether it is secure enough are worth studying. In order to perform these issues, we choose 50 open-source Android applications which call the cloud storage SDKs and analyze their source codes. According to the result of our analysis, all applications use a java class named SharedPreferences to store the access token. Note that the SharedPreferences class could create a xml file in the application data folder (/data/data/your_packet_name/shared_prefs/your_prefs_name.xml). Because of the sandbox mechanism [19] of Android system, other programs are constrained to access to the data which belongs to another application. In most cases, it is secure to store the access token in the SharedPreferences class. However, the data stored in SharedPreferences could be leaked once meeting the following two issues:

1. Application uses SharedPreferences to store unencrypted access token
2. The Mobile device is rooted

The second condition seems very harsh, but the reality is that more than a fifth of users (22%) rooted their Android devices according to Tencent 2015

[18] these applications all perform a cloud storage SDK call

[19] a sandbox mechanism is a security mechanism for separating running programs, it typically provides a tightly controlled set of resources for guest programs to run in, such as scratch space on disk and memory.

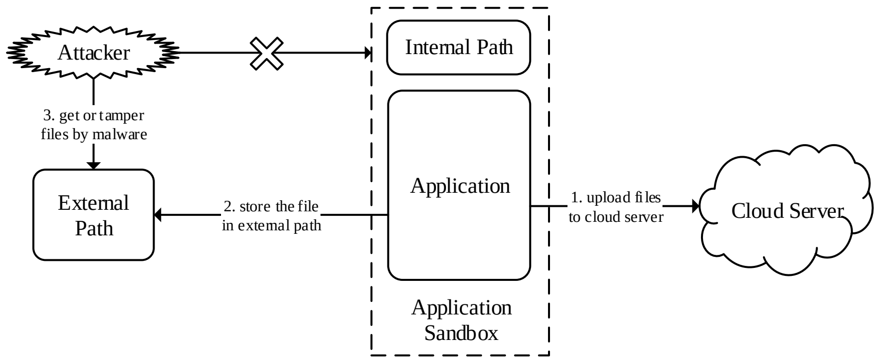
Attacker
Internal Path
3. get or tamper files by malware
Application
External Path
2. store the file in external path
1. upload files to cloud server
Cloud Server
Application Sandbox

Fig. 2. Attack caused by external download path

Internet Plus White Paper [20]. Thus, if the third-party developers want to store the access token or other sensitive information in SharedPreferences, they should consider this security factors. A good method is to encrypt the access token before storing it in SharedPreferences, so that an attacker cannot decrypt it even if he has got the SharedPreferences file. Besides, another effective method is to store the access token on the server side. That means before calling a cloud SDK API, third-party application need to send a request to the server to get the access token first. The above two methods can protect the access token more effectively.

C. The default saving path

When an application downloads a file from the cloud storage server on behalf of the resource owner and then save the file locally, the third-party developers always set a default path to simplify users' operation, as shown in Figure 2. By analyzing the source code of the applications, we find that the default download path is always set as an external path, such as the root path of SD card. This provides an extremely indetectable attack for users since an attacker can access to these files without any restrictions, while these files may contain some sensitive information, e.g. salary, physical condition, credit card numbers, etc. Once the information exploited by attackers, it will cause a great damage to the users. What's more, since it is difficult to detect this malicious behavior because the users usually do not check the file before downloading it from a cloud server, the attackers could tamper the information without any awareness by users.

To avoid this security problem, we recommend the third-party developers to set the default save path as an internal path, e.g. in the application sandbox. Besides, if users choose a external path as the download path, it is better to remind them to ensure that the file does not contain any sensitive information.

[20] `http://qzonestyle.gtimg.cn/qzone/vas/opensns/res/doc/2015Internet\_PLUS\_Product\_White\_Paper1020.pdf`

6 Related Work

The cloud storage security issues have been and currently are widely discussed in both the industry and the academia [22]. Many researchers have focused on analyzing if the cloud storage provider still preserve the clients file, and whether a file has been leaked out or modified. S. Subashini et al. describe various security issues of cloud computing due to its service delivery models and lists some of the current solutions which partly target the security challenges posed by the cloud [23]. In a paper by Richard Chow at al. [24] characterizes the problems and their impact on adoption that arise with the usage of cloud storage service. Especially, they introduce third-party data control as a security concern. The authors of [25] present a model for provable data possession (PDP) that provide verification to users that whether the cloud storage server possesses the data without retrieving it. Furthermore various researches perform more improved storage protocols [26, 27, 28, 29].

To migrate storage space and communication bandwidth in cloud storage, Deduplication [30] has emerged as a major technique with only one actual file on the server. Further, to enforce data confidentiality as well as ensure storage saving, convergent encryption is proposed by J.R. Douceur et al [31]. After that, many papers focus on improving this system with deduplication and security [32, 33, 34]. Additionally, Harnik et al. make a summing up of the attacks of side channels [15] with server-side deduplication in cloud storage [9]. They suggest users to encrypt their data before upload to turn off the deduplication, and present a randomized solution with unpredictable deduplication. In a paper by Martin Mulazzani et al. [35], client-side data possession proofs was conducted to avoid hash manipulation attacks with practical evaluations in Dropbox for their discovered attacks.

In recent years, with a rapid growth of cloud storage service and mobile devices, many researchers shift their attention to mobile cloud storage security issues. Based on this, application in Android usually call a third-party SDK, which is developed by cloud storage provider, to simplify their development. Roee Hay et al. present a severe vulnerability in Dropbox SDK on the Android platform [1]. This vulnerability can develop local and remote attacks. Besides, nowadays many security communities has been devoted to discovering the vulnerability within cloud storage SDK. For instance, WooYun.og [21] discover Wormhole by Baidus Moplus SDK includes Baidu Cloud SDK that allows attackers to access the content stored on affected applications through an open HTTP server without notifying users [6]. Consequently, this vulnerability put 100 million Android devices at risk.

However, these work above do not make an overall code analysis about Android SDK in cloud storage. In other words, they always focus on uncovering a specific vulnerability in a certain SDK or presenting a more secure system protocol. To our knowledge, our work is different from others by paying attention to the code security analysis of mobile backup modules on the Android platfor-

[21] http://www.wooyun.org/

m. In this paper, we perform a multi-level security study across four different Android cloud storage SDKs: Dropbox SDK, Baidu Cloud SDK, OneDrive SDK and Google Drive SDK. After analyzing usage, protocol and API functions in different Android cloud storage SDKs, we find some security issues which could occur in the processing of the cloud storage SDK call by third-party developers or in the SDK itself. Consequently, we make coding suggestions for third-party developers when they call these cloud storage SDKs.

7 Conclusion and Future Work

In this paper, we have conducted the first multi-level empirical study on code security issues of four mainstream Android cloud storage SDKs: Dropbox SDK, Baidu Cloud SDK, OneDrive SDK, and Google Drive SDK. After counting the usage of these SDKs in third-party applications in Google Play Store and Cool Store, we compared the API functions of these four SDKs and performed code analysis focused on four protocols: communication, encryption, deduplication and authentication. We found that none of them encrypt users' files before data transmission and some of them adopt a client-side deduplication which could cause serious vulnerabilities. Besides, we present improper coding issues in cloud storage SDKs and Android third-party applications. Based on this, we proposed countermeasures for their developers.

For the future work, we will pay attention to discover more influential security issues on mobile backup modules on the Android platform. In this paper, our summarized code security issues are only based on empirical study, we plan to conduct lots of experiments to enforce the influence of these security issues. Last but not least, we will consider moving those analysis methods to another security research area.

References

1. Roee Hay and Or Peles. Remote exploitation of the dropbox sdk for android.
2. Wikipedia. Dropbox (service) — wikipedia, the free encyclopedia, 2016. [Online; accessed 18-June-2016].
3. Wikipedia. Onedrive — wikipedia, the free encyclopedia, 2016. [Online; accessed 18-June-2016].
4. Wikipedia. `GoogleDrive---Wikipedia{,}TheFreeEncyclopedia`, 2016. [Online; accessed 18-June-2016].
5. Wikipedia. Baidu cloud — wikipedia, the free encyclopedia, 2016. [Online; accessed 18-June-2016].
6. Scott Wu@0xid Min(Spark) Zheng, Magic Ding. Baidu patched sdk for over 100 million android devices affected by wormhole, 2015. [Online; accessed 18-June-2016].
7. Wikipedia. Communications protocol — wikipedia, the free encyclopedia, 2016. [Online; accessed 18-June-2016].

8. Tanya Vladimirova, Roohi Banu, and M Sweeting. On-board security services in small satellites. In *MAPLD Proceedings*, 2005.
9. Danny Harnik, Benny Pinkas, and Alexandra Shulman-Peleg. Side channels in cloud services: Deduplication in cloud storage. *Security & Privacy, IEEE*, 8(6):40–47, 2010.
10. Dick Hardt. The oauth 2.0 authorization framework. 2012.
11. San Tsai Sun, Kirstie Hawkey, and Konstantin Beznosov. Systematically breaking and fixing openid security: Formal analysis, semi-automated empirical evaluation, and practical countermeasures. *Computers & Security*, 31(4):465–483, 2012.
12. Sascha Fahl, Marian Harbach, Thomas Muders, Lars Baumgärtner, Bernd Freisleben, and Matthew Smith. Why eve and mallory love android: An analysis of android ssl (in) security. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 50–61. ACM, 2012.
13. Sascha Fahl, Marian Harbach, Henning Perl, Markus Koetter, and Matthew Smith. Rethinking ssl development in an appified world. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 49–60. ACM, 2013.
14. Ms Akanksha V Patil and Mr Navnath D Kale. Survey on secure authorized de-duplication in hybrid cloud. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(11):3574–3577, 2014.
15. Tobias Pulls. (more) side channels in cloud storage. In *Privacy and Identity Management for Life*, pages 102–115. Springer, 2011.
16. E Hammer-Lahav. Introducing oauth 2.0. *Hueniverse, May*, 2010.
17. Suhas Pai, Yash Sharma, Sunil Kumar, Radhika M Pai, and Sanjay Singh. Formal verification of oauth 2.0 using alloy framework. In *Communication Systems and Network Technologies (CSNT), 2011 International Conference on*, pages 655–659. IEEE, 2011.
18. Suresh Chari, Charanjit S Jutla, and Arnab Roy. Universally composable security analysis of oauth v2. 0. *IACR Cryptology ePrint Archive*, 2011:526, 2011.
19. Q Slack and R Frostig. Oauth 2.0 implicit grant flow analysis using murphi. *URL: http://www. stanford. edu/class/cs259/WWW11/(: 05.02. 2013)*, 2011.
20. Torsten Lodderstedt, Mark McGloin, and Phil Hunt. Oauth 2.0 threat model and security considerations. 2013.
21. San-Tsai Sun and Konstantin Beznosov. The devil is in the (implementation) details: an empirical analysis of oauth sso systems. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 378–390. ACM, 2012.
22. Diogo AB Fernandes, Liliana FB Soares, João V Gomes, Mário M Freire, and Pedro RM Inácio. Security issues in cloud environments: a survey. *International Journal of Information Security*, 13(2):113–170, 2014.
23. Subashini Subashini and Veeraruna Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of network and computer applications*, 34(1):1–11, 2011.
24. Richard Chow, Philippe Golle, Markus Jakobsson, Elaine Shi, Jessica Staddon, Ryusuke Masuoka, and Jesus Molina. Controlling data in the cloud: outsourcing computation without outsourcing control. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, pages 85–90. ACM, 2009.
25. Giuseppe Ateniese, Randal Burns, Reza Curtmola, Joseph Herring, Lea Kissner, Zachary Peterson, and Dawn Song. Provable data possession at untrusted stores. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 598–609. Acm, 2007.

26. Ari Juels and Burton S Kaliski Jr. Pors: Proofs of retrievability for large files. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 584–597. Acm, 2007.
27. Kevin D Bowers, Ari Juels, and Alina Oprea. Proofs of retrievability: Theory and implementation. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, pages 43–54. ACM, 2009.
28. Giuseppe Ateniese, Roberto Di Pietro, Luigi V Mancini, and Gene Tsudik. Scalable and efficient provable data possession. In *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, page 9. ACM, 2008.
29. C Chris Erway, Alptekin Küpçü, Charalampos Papamanthou, and Roberto Tamassia. Dynamic provable data possession. *ACM Transactions on Information and System Security (TISSEC)*, 17(4):15, 2015.
30. Sean Quinlan and Sean Dorward. Venti: A new approach to archival storage. In *FAST*, volume 2, pages 89–101, 2002.
31. John R Douceur, Atul Adya, William J Bolosky, Dan Simon, and Marvin Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, pages 617–624. IEEE, 2002.
32. Mihir Bellare, Sriram Keelveedhi, and Thomas Ristenpart. Message-locked encryption and secure deduplication. In *Advances in Cryptology–EUROCRYPT 2013*, pages 296–312. Springer, 2013.
33. Sriram Keelveedhi, Mihir Bellare, and Thomas Ristenpart. Dupless: server-aided encryption for deduplicated storage. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 179–194, 2013.
34. Jian Liu, N Asokan, and Benny Pinkas. Secure deduplication of encrypted data without additional independent servers. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 874–885. ACM, 2015.
35. Martin Mulazzani, Sebastian Schrittwieser, Manuel Leithner, Markus Huber, and Edgar R Weippl. Dark clouds on the horizon: Using cloud storage as attack vector and online slack space. In *USENIX Security Symposium*, pages 65–76. San Francisco, CA, USA, 2011.