Journal of Computer Networks, 2017, Vol. 4, No. 2, 65-68
Available online at http://pubs.sciepub.com/jcn/4/2/1

DOI:10.12691/jcn-4-2-1

Cloud-based Storage Framework for Mobile Social Apps

Abdalla Alameen*

Department of Computer Science, College of Arts and Science, Prince Sattam Bin Abdulaziz University Al-Kharj, Saudi Arabia
*Corresponding author: abda711su@gmail.com

Abstract Smart devices using the Android system are facing memory issues because of the prolonged use of social Apps. As a result, users tend to frequently reformat their smart devices, or update them with new operating system versions. To address this problem, we propose here a cloud based storage framework for social Apps. Our proposed solution has shown promising results, and can be used by any social App to solve its storage accumulation issues.

Keywords: *android, memory, apps, cloud, multimedia*

Cite This Article: Abdalla Alameen, "Cloud-based Storage Framework for Mobile Social Apps." *Journal of Computer Networks*, vol. 4, no. 2 (2017): 65-68. doi: 10.12691/jcn-4-2-1.

1. Introduction

The present situation concerning the increasing use of digital devices for work, communications, and media consumption is well summarized by the fact that "Americans now own four digital devices on average, and the average U.S. consumer spends 60 hours a week consuming content across devices" [1]. Users rely on smart devices both for their personal uses and for professional work. Smartphone manufacturers are offering very user-friendly graphical interfaces for mobile applications (Apps) [2], to the point where some of the Apps in smart devices are even superior to their counterpart desktop applications. Most applications—and especially social networking sites—are now commonly available in smart devices in the form of a dedicated App.

As a direct consequence of their permanent availability, these Apps are enabling users to be connected to their loved ones on a 24/7 basis. However, by continuously storing text, audio, and video messages on the device, they can eventually drain the smart device's memory. Memory related issues can cause serious damage to the system and the installed Apps, eventually bringing the full system down. To mitigate this problem, users tend to systematically reformat the system with either the original factory images, or new operating system (OS) versions. This approach can severely penalize device owners, because some locally saved data is inevitably damaged or permanently lost, even when care is taken to move all local data to some other storage system (which typically is, in itself, a major burden to users).

To avoid such situations, users are required to manually check the App's memory usage and then dump the data into external memory. Manual probing of storage usage is not, however, a feasible solution. Therefore, in this article, we survey the memory related issues caused by the Apps' accumulated messages—such as text, audio, and video-in the smartphone's memory and propose a solution to avoid such problems. In particular, we are proposing a memory framework that allows social Apps to abstain from accumulating multimedia and text messages in their respective smart devices.

When memory is allocated to newly created objects, the random access memory (RAM) of the Android system is shared with the other processes of the main App. RAM sharing among the processes is a direct consequence of the nature of Zygote, a daemon process at the core of the Android OS, serving as launching pad for every App present in the system. The memory framework of the Android OS is very complex and includes elements to perform the necessary garbage collection and to assess each application's memory-related behavior. Despite that, many memory-related issues are being faced in every smart device using Android. This is mainly caused by Apps that collect data during their interaction with remote Apps or servers, and is especially so for social Apps. To avoid memory issues on Android systems, some type of standardization should be put in place.

2. Apps

Android mobile OS, uses Dalvik Virtual Machine (DVM) to run apps. DVM can execute .dex files; these files are very much similar to the java jar files. DVM, optimizes the code such that it removes duplicate data entries from the class files. The multitasking is achieved because of Dalvik, which allows Android apps to run its own instances of DVM. More recently, DVM is not being used with Android; Android Runtime (ART) is used in the most recent release of Android OS. However, still user has the ability to choose the virtual machines. Dalvik uses the just in compiler, part of application will run that is compiled and while rest of the code (of the application) will be compiled later as and when needed. So hence it produces the very small memory footprints of the application and uses very less storage space. For ART based apps, their complete code will be precompiled

during installation, JIT is removed, so the application can perform better as compared to DVM.

With ART approach, we can conclude that the applications and Android are more concerned with the speed and storage. But still we are having many issues related to the Apps. To avoid such situations, it is better to look into the memory usage at regular interval of time. There are many apps which are very much downloaded at regular intervals are also unable to perform memory checks. WhatsApp is message service app and it has users of about half billion, it allows cross platform messaging. This app is available on most of the operating systems of smartphones. Very recently it is also released web clients for the desktop users.

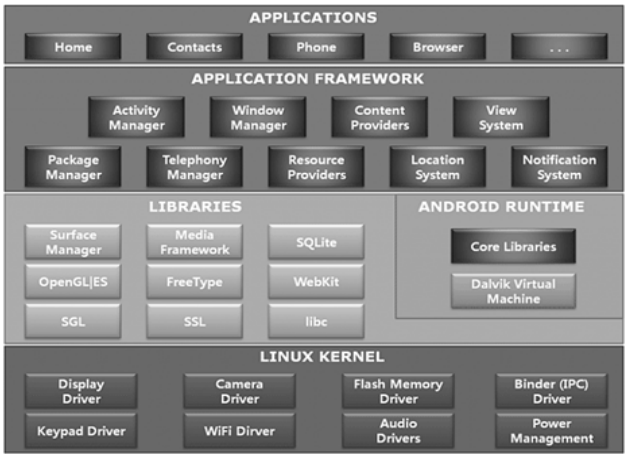

Figure 1. Android Linux kernel

Among social apps, WhatsApp has the ability to lock the users' messages during their transit of messages and only receiver has the ability to unlock the text message by using text secure protocol. Security related information is out of scope of this article. However, WhatsApp regularly updates its security loopholes and also the code auditing is performed. WhatsApp is the only App 50 billion messages are sent and received per day.

Vine is a video sharing service app, which enables users to record video messages about 5 to 6 seconds long with the followers. Recently release of vine enable the loop count which shows number of times this been watched.

Snapchat is a photo messaging application developed by Evan Spiegel and Bobby Murphy; users can use existing or create messages such as text, video, audio and drawing. These messages can be sent by the snapchat users to the list of users, messages are time bounded shares wherein sender can set the time limits for viewing, and messages are downloaded at receiver side after the time limit the messages are hidden from the user then deleted at the server side.

Response to disaster, Japanese firm developed an application by name Line, because telecommunication infrastructures were damaged during disaster. Within two years Line reached 300 million users mostly from Japan itself. However, Japanese firm says that they going to reach 700 million in next few years.

Foursquare is local location and place information's are shared among the app users. By visiting this app one can get different places information and recommendations from the users they trust.

Instagram is photo sharing, video sharing, and text messaging and social networking. Instagram allows users to share these pictures, video and text on their desired social networking websites and apps. Instagram had added hashtags to identify the photographs.

Waze is a traffic and navigation app, this is one of the applications which are bought by the Google with highest payout to the employees. Users of this app get timely alert of traffic signals, accidents or traffic jams and saves cost by showing the cheapest gas station on your route.

BlackBerry Messenger, is a personal identification number based IM service, it can perform video calls between various cross platforms. Initially it was only possible to communicate between blackberry devices, but recently app is available for other smartphones. It is also widely used app as it has more than 190 million users.

Smartphone users prefer App based text messaging than that of SMS, because obviously cost involved in it. Hence, on an average, most of the smartphone users are having these apps for communicating purpose. An average WhatsApp user is sending more than 1000 message and also receives more than double of the sent messages. If this tendency continues user will end up with less memory for his/her smartphone and user has to manually start adjusting smartphone apps by moving them to the external memory. However, WhatsApp makes regular backup of its messages at everyday 4:00 AM and stores them in the WhatsApp folder, it can be placed at local memory or external.

Similarly, Snapchat social enables the user to set time limit for the shared audio and video message after that the shared media will be deleted from the snapchat server. Hence, with this approach the shared media won't be available after its timestamp and user memory will also be consumed less by this application.

Most of the apps allocate memory dynamically; DVM and ART check these Apps's memory at regularly for garbage collection. However, this facility is further strength with help of developer only. In Linux, active memory pages/space is swapped once the RAM is full. But swap spacing facilities not available in Android systems. However, Android system uses paging and memory mapping techniques. Paging is a scheme which allows smart devices to store and retrieve data from the secondary memory; memory mapping is a part of virtual memory of each process created by the developer. Developers are allowed to use memory for development of apps for their objects and yet object remains resident of RAM and hence cannot be paged out. So it is the responsibility of the developer to release the object references for making way to the garbage collector. Allocating memory and reclaiming App memory is a part of Android memory management architecture.

2.1. Frame Work for App Development

The proposed solution is as follows. Figure 2 shows a diagram of social App groups involved in multimedia and text based activities. Each group has an administrator, who is responsible for creating the group. Each group has a maximum of 200 members and a minimum of 2 members. All these members have the capability to access the memory shared between them (in the form of multimedia

and text messages) directly from the cloud itself. For each social group Gi, i = {1, 2, 3, …, n}, there exists a memory block P in the Cloud, which is shared among the members of the group. Based on these storage partitions, a virtual machine (VM) exists for each social group Gi. These VMs have two parts. The first one is for shared data (data that is being shared between the group users). Shared data can be manipulated in two forms: some users will wish to delete the data, whereas other users will want to keep the data permanently. The VM memory (VMmi) at any given moment of time is given by (1).

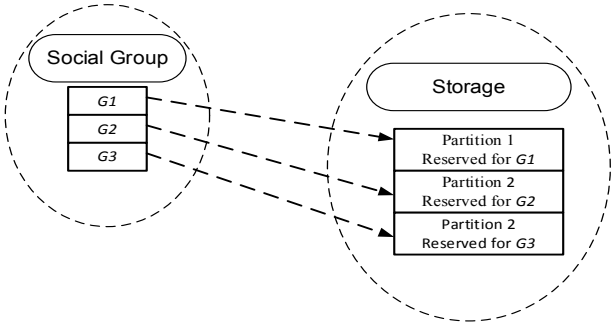

Figure 2. Social groups store data into the cloud

$$Vmmi = Up + \sum_{i=2}^{n} G_i, \quad (1)$$

where $\boldsymbol{U_p}$ is the unused partition block of $\boldsymbol{VM_m}$. The size of $\boldsymbol{U_p}$ depends on the size of the shared data. If the users of social group $\boldsymbol{G_i}$ delete shared data, the fraction $\boldsymbol{S}$ of occupied shared memory will decrease. The size of the unused partition $\boldsymbol{U_p}$ can therefore be written as follows,

$$Up = Vmmi(1 - S), \quad (2)$$

which clearly represents the fact that the unused memory size $\boldsymbol{U_p}$ depends on the memory consumed by the multimedia data in social group $\boldsymbol{G_i}$. The factor $\boldsymbol{S}$ (which we shall name status data) takes values between 0 and 1. If the shared data is completely deleted, $\boldsymbol{S}$ can become 0. Similarly, S can be calculated as follows.

$$S = 1/n, \quad (3)$$

where $\boldsymbol{n}$ is the number of group members.

To meet further memory demands on $\boldsymbol{G_i}$, we can combine the left-over memory generated by shared-data prior deletions. If the $\boldsymbol{G_i}$ is not obtaining enough memory to accommodate its shared data, our framework can provide additional memory by merging memory partitions.

3. Evaluation

Initially, the server from the group member will receive the shared data, and will send it to the users only on request. However, the server will notify the clients about the arrival of new data, along with the name of the group member that generated the data. With this approach, smart devices can receive the shared data by tapping on the message, or just swipe across to delete it. The algorithm used by the server to share data to the rest of members is presented in Figure 3, which illustrates the message passing algorithm that enables data sharing to the user via the server. With this approach, social Apps can reduce memory utilization at the end-user devices. We compared our approach with that of the very famous social App WhatsApp. Our findings are illustrated in Figure 4.

```
Begin
  message_num←0
  while msge_recv_num<num_incoming do
      Receive(msge_pkt, msge_pkt_size) from the member
    If msge_pkt is empty then
      Break;
    end if
    foreach token of msge_pkt do
      send token to each user in the group;
    end foreach
      msge_recv_num ← msge_recv_num+ msge_pkt_size
  end while
end
```

Figure 3. Algorithm for sending messages to the server

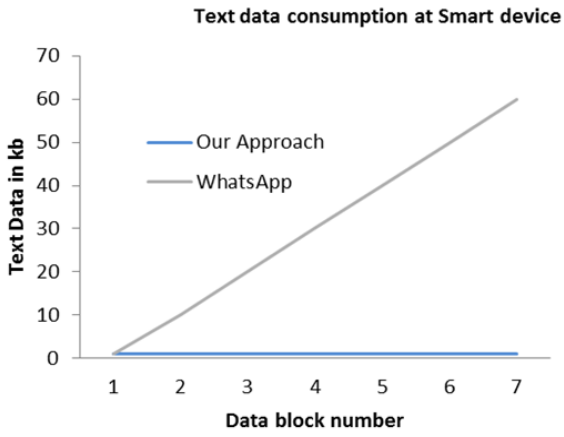

Figure 4. Text message transmission between server and clients

When we use WhatsApp for text messaging using variable text data sizes such as 10 kb, 20 kb, 30 kb, 40 kb, etc., the smart device's memory stores that information and is therefore constantly filling up, whereas our approach preserves the existing free memory in the smart device, unless the user wishes to download the shared data from the server. Memory availability is a vital requirement for mobiles to function normally. Therefore, the prolonged use of social Apps or any other memory-consuming activities will degrade the smart device's performance. By using our approach, we can at least choose the important messages that deserve to be stored in our device; the remaining messages can be discarded.

When storing data in the cloud, special care should be taken with information security. Threats and data breaches have always been a concern for cloud service providers (CSPs). There are many kinds of threats to the CSP activity. These threats do not only directly hamper cloud services, but also lead to the cloud being considered an untrustworthy platform for data storage. The Cloud Security Alliance released an online magazine called The Notorious Nine discussing the threats faced by cloud providers. Data loss is considered to be the top ranked threat. Furthermore, it was also reported that even data stored on reliable clouds can be in danger and could eventually be lost for various reasons, including accidental deletion by the cloud provider. Natural calamities, such as

earthquakes, fires, and other natural disasters could also cause user data deletion from the cloud. CSPs can use cloud auditing techniques to ensure the privacy and integrity of the remote data. Cloud auditing requires two to three entities to perform data auditing. These entities are classified as user/client, auditor, and server (where the data is stored). Users issue requests to the auditor to perform the auditing; the auditor then issues a challenge to the server, which in turn releases the proof for the challenge. Finally, verification is performed on the block of data. All these cloud auditing techniques use cryptographic methods such as message authentication codes (MAC), homomorphic authentication, or Boneh-Lynn-Shachan signature schemes [3]. The CSP-held data privacy and integrity is at stake, and this has opened new doors to innovative research opportunities. To cope up with existing threats, both CSPs and cloud users should prepare in advance to act against those threats before they even materialize and actually hit. It has also been suggested that a strategic document about risk management should be prepared, and that this document should be reviewed by the stakeholders. Additionally, we should apply continuous pressure on the threat actors and be attentive to the privacy and security issues arising from policy changes. This is also an opportunity for researchers to apply new scientific methods to overcome cloud data breach issues caused by the threat actors.

We may summarize this paper by saying that it proposed an alternative memory framework for smart devices. In our approach, we have empowered the servers, making them vital components of the message sharing architecture. In the future, we would like to investigate the use of adaptive transmission links to serve end-users under fluctuating transmission services.

Acknowledgements

We greatly appreciate the Deanship of Research of the Prince Sattam University for allowing us to carry out our research. The authors would also like to thank the editors and reviewer's efforts in improving the manuscript, which we much appreciated.

References

[1] Nielsen. "The U.S. digital consumer report," *Nielsen Publishing* Available:http://www.nielsen.com/us/en/insights/reports/2014/the-us-digital-consumer-report.html [Accessed: 25 August 2015].

[2] Zheng, P. and Ni, L. (2010) Smart phone and next generation mobile computing. Morgan Kaufmann, Massachusetts.

[3] Kolhar, M., Abu-Alhaj, M. M., and El-atty, S. M. (2017). Cloud Data Auditing Techniques with a Focus on Privacy and Security. IEEE Security & Privacy, 15(1):42-51.