MiniUpdate: A Smart Update Method for Smartphones

[1]Bhavik Patel, [2]Vinayak Nalwade, [3]Gauri Pagare, [4]Kishori Navale
[1]Student, [2]Student,[3]Student,[4]Student
[1]Deparment of Computer Engineering
[1]Mathoshri college of Engineering and Research Centre, Eklahare, Nashik ,India

***Abstract*— Nowadays almost everyone uses Smartphone with Android operating system. A huge variety of applications are being developed for android. These applications have some bugs, errors and limitations, to overcome this limitations updates are introduced. Updating an application increases a considerable amount of traffic in network. To overcome this shortcoming this application tries to reduce the size of application update thus reducing the network traffic using Delta encoding techniques.**

***Index Terms*— Android Update, xdelta, File Compression, Delta Encoding, Patch.**

I. INTRODUCTION

Updating an Android application is a tiresome process. More than one million applications are provided by Google Play Store. Most of the applications are provided with an update almost every week. This updates have bug fixes and new features to give a better experience to users. This update process is done by billions of users at the same time and this leads to higher traffic on the cellular network and load on the update providing servers. Due to higher load on servers time taken for updating the application is increased and the users get a bad experience out of it.

Figure 1 shows the statistics of Google Play Store application downloads [5]. The download of application has reached above 50 billion which means there is a huge amount of load on data servers and the updating of the same will increase the load on cellular traffic and the update servers.

Google has developed Google Smart Application Update to reduce traffic for application update. Google Play generates an update patch for the new version of application, this patch is then applied to the older version of application. Although this method helps in reducing traffic but the compression technique used is not optimal. The compression is done on Android Application Package (APK) level only and it limits the possible reduction of patch size.

To overcome this flaw we are going to apply delta encoding with the help of xdelta tool. This tool is an efficient way to reduce the size of application update. Furthermore we are going to extract the APK files of old and new versions and then apply delta encoding on these files. This helps us to get compression done on every file included in the APK, which will further reduce the size of update helping in reducing the traffic over the network. We can also try to share this update patches through Bluetooth enabled smartphones, thus leading to even lesser traffic on the cellular network and update servers. We can reduce the size of application update size by about 75 percent rather than the 55

Year	Month	Applications available	Downloads to date
2009	March	2,300	
	December	16,000	
2010	March	30,000	
	April	38,000	
	August	80,000	1 billion
	October	100,000	
2011	May	200,000	3 billion
	July	250,000	6 billion
	October	319,000	
	December	380,297	10 billion
2012	January	400,000	
	February	450,000	
	May	500,000	
	June	600,000	20 billion
	September	675,000	25 billion
	October	700,000	
2013	February	800,000	
	April	850,000	40 billion
	May		48 billion
	July	1,000,000	50 billion
2014	June	1,200,000	
	July	1,300,000	
	December	1,430,000	

Fig. 1 Google Play Application Statistics

percent provided by Google Smart Application Update.

But the patch implementation is a very complex process on the smartphone which requires more time. This time delay leads to more computation on the processor which directly leads to more battery consumption. The battery consumption problem can be neutralized by the lower traffic on cellular network, which indirectly leads to more battery consumption for downloading update..

Thus we have to create a server side application to create this small size patch and a client side Android application which will deploy this patch and install the updates.

II. LITERATURE SURVEY

A literature review is a critical and an evaluative summary of the themes, issues and arguments of a specific clearly defined research topic obtained from the published (and unpublished) literature.

Google developed Google Smart Application Update in the year 2012. This method uses a transparent method for reducing the size of Application Update. This method uses Delta Encoding technique for compression process [2]. A Update patch is generated and is then applied to the APK of the Application. Although this method reduces the size of update and reduces the network traffic, the compression method is not optimal. The compression is done only on the APK level which limits the possible size reduction of the update patch. The advantages of this method are that the battery consumption is low. The installation of the APK file requires a very less time.

TABLE 1. Literature Survey

Sr. no.	Title	Pros	Cons
1	"Google Smart Application Update" [3]	1) Update installation takes less time. 2) Battery consumption is low.	1) Network traffic is high. 2) Compression technique is not optimal. 3) Higher consumption of bandwidth.
2	"DELTA++: Reducing the Size of Android Application Updates" [1]	1) Network traffic is reduced. 2) It is more optimal compared to other techniques. 3) It is cost efficient.	1) Requires more time to deploy a patch. 2) Battery consumption is more.

III. PROPOSED SYSTEM

The proposed system will have a server side and client side application. The server side is responsible for generating the small size patch and the client side is responsible to implement this patch and get the application up to date.

System Architecture consists of a server side application and a client side which will be a Android Smartphone or Tablet. Server will accept old and new APK file as input. Server will process on these files and generate a update patch. This patch will be sent through a cellular network to the client. At the client side this patch will processed at the client side to generate a new APK file. The result will be generated at the client in form of reduced sized update for the respective application. This new APK file will then installed at the client side.

A. Server Side

The server side will generate small size patches with the help of delta encoding tools available free in market. This tool will compute differences of every file in the APK and then save it the patch file.

Steps for Sever Side Application:

1) Extract the old and new APK files.
2) With the help of xdelta tool we will calculate the differences of old and new version files [6].

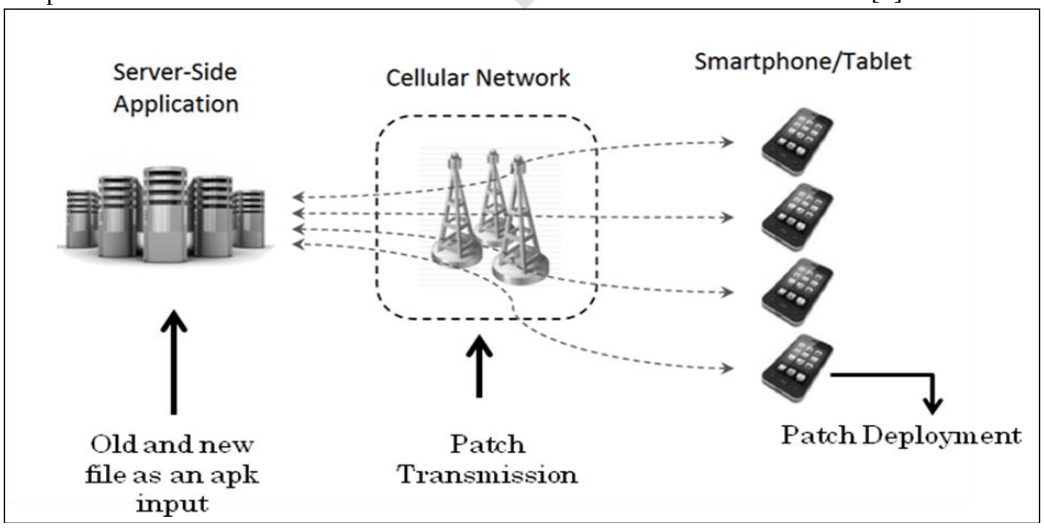

Fig 1. System Architecture

3) The files that are no longer required will be deleted, the files with same names but with few changes will get deltas computed for it, the new files will be copied as it is.
4) The patch will be generated and will further get compressed to get more small size.
5) Generated compressed patch will be transferred through the cellular network to the client application [4].

B. Client Side

The client side will download the patch generated at the client side. With the help of delta encoding and the old version APK we will generate a new version APK and install it.

Steps for Client Side Application:

1) Download the patch from the server side and decompress it.
2) Fetch the older version of APK and decompress it in a new folder.
3) Delete the files that are no longer required.
4) Again with the help of delta encoding update the files with the deltas provided from the patch.
5) Copy the new files from patch to the new folder.
6) With the help of the files in this new folder we get a new version of decompressed new APK.
7) Construct the new version APK with the help of the new folder.
8) Install the new APK with the help of Android Package Installer.

IV. EXPECTED RESULTS

MiniUpdate will reduce the size of update by at least 75 percent of the original update size. It will reduce the cellular traffic by over 50 percent and also the load on update servers. This will help in getting a traffic free network and will further contribute to get a better experience for Android users. The Bluetooth sharing feature will further help in getting lesser traffic on the network.

We have constructed and deployed patches for the free applications which are popular on the basis of the number of download. MiniUpdate shows considerable reduction in the patch size as compared to the Google Smart Application Update.

It is expected that this technology won't work for paid applications as it requires access to Google Play Store server.

The battery consumption is an issue as it takes more time for the complex patch to be deployed on the client side. But this disadvantage can be ignored as the traffic reduction feature will negate it. As the battery consumption will be more if the traffic is high and will take more time to download which in turn takes more power as well as time.

V. CONCLUSION

We have introduced a android Smartphone application with the name MiniUpdate. The application will reduce the size of updates which are provided for all the application which are developed up to the date. The network traffic will be reduced to a great extent which reduces the load of the servers which holds these applications. MiniUpdate will prove to be very helpful for all the Android Smartphone users as their internet data will also be saved. The trade-off will be the increase in the deployment time of the application as every module of the Apk file will be compressed. The battery usage problem is also negligible.

Thus, MiniUpdate will prove to be a effective application and save lot of money which is being spent on the application updates.

ACKNOWLEDGMENT

We take immense pleasure in thanking our college management for having permitted us to carry out this project work. Words are inadequate in offering our thanks to our respected co-operation in carrying out project work in the starting phase of the project like requirement analysis and literature survey. We also express our at most gratitude to the department of Computer Engineering, for this valuable co-operation in selecting the project topic and guiding us stepwise ahead. We also express our sincere gratitude towards Savitribai Phule Pune University for giving an opportunity to publish our ideas. Inspiration and guidance are invaluable in every aspect of life, especially in field of education, which we have received from our respected project guide who has helped a lot In first two phase of project like topic selection then information gathering and guide us to throughout the project work and gave earnest co-operation whenever required. We would like to express sincere gratitude towards him. At last, we would like to take this opportunity to convey thanks to all my staff members, who directly or indirectly encouraged and helped us project work. Finally yet importantly, we would like to express heartfelt thanks to our beloved parents for their blessings, our friends/classmates for their and helps and wishes.

REFERENCES

[1] N. Samteladze and K. Christensen, "Delta++: Reducing the size of Android Application updates" *IEEE Internet Computing*, 2014, pp. 50-57
[2] N. Samteladze and K. Christensen, "DELTA: Delta Encoding for Less Traffic for Apps," *Proc. IEEE Conf. Local Computer Networks*, 2012, pp. 212–215.
[3] S. Musil, "Google Play Enables Smart App Updates, Conserving Batteries," *CNET News*, 16 Aug. 2012; http:// news.cnet.com/8301-1023_3-57495096-93/google-playenables-smart-app-updates-conserving-batteries/.
[4] Torsten Suel and Nasir Memon, "Algorithms for Delta Compression and Remote File Synchronization" Work supported by NSF CAREER Award NSF CCR-0093400 and by Intel Corporation.
[5] "Google Play Application statistics" http://en.wikipedia.org/wiki/Google_Play
[6] "Information, working and to download xDelta tool" http://xdelta.org