Volume 5, Issue 9, September 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Energy Efficient Network Transmission for Android

[1]Uzmamasrat F. Shaikh, [2]Dr. A. N. Banubakode
[1]Computer Engineering Department & Savitribai Phule Pune University, Maharashtra, India
[2]Professor, Department of Information Technology, Maharashtra, India

Abstract— Modern life leads to immense dependence on smartphones. So there is a swift in the number of smartphones applications too, that has impact of web services as their main building blocks. But unfortunately, these applications consumes more energy creating tribulationsin day-to-day life of common man as well as in context of battery constrained mobile devices. In this paper, we mainly focus to efficiently attain energy performance during network transmission, which would be contributed towards GreenDroid. So we are building a tool that will compress the data while performing transactions between servers and also enhancing this process by storing relevant amount of information locally inside the application to make it secure and readily available. Our results indicate that energy savings of upto 15-20% may be achieved through the sophisticated use of compression. The result of our work facilitate more energy efficient and servicebased mobile application.

Keywords— Energy Efficiency, Compress, Decompress, Server, Web Services.

I. INTRODUCTION

Android phones and tablets are becoming more and more prevalent. At the same time there has been a rush in the development and adoption of specialized programs, ranging from leisurely to mission-critical. People are using Android device to perform the work that was impossible few years ago without computers. This places a heavy emphasis on battery life of Android device. Due to which, many end users are not satisfied with the battery life of their Android devices. Thus, there is a growing need for software-quality tools in all stages of an application's life-cycle, including development, testing, auditing, and deployment. Energy is a limited resource for Android phone users.

Available energy is constrained by limits on battery size and weight, and improvements in battery technology have historically been slow. Moreover, energy demands often increase with the addition of new hardware and software features. To make matters worse, operating systems and applications frequently consume energy to perform tasks that are ultimately useless and are unknown to user viz. network data transmission. Developing a energy efficient tool is challenging and implementations can vary widely in terms of their energy consumption. As a result, battery usage has become an important, albeit informal, quality metric for marketplace application.

Another important part of distributed application are web services, which constitute a popular implementation of service-oriented architecture. Web services are software artifacts designed for machine-to-machine communication which interact via network. They provide a wide range of functionality and can be composed to more complex application. Web services are accessed by different clients using poen and standardized interfaces.

Since the recent smartphones feature fast (CPU), variable communication interfaces viz. WLAN, 3G and multiple sensor capability, developement of batteries can't cope-up with these rapid advancements in smartphone technology. The average usable time is predicted to reduce by 4.8%. between two charges of cellphone anually[1]

Web services are accessed through internet using wireless interfaces. However frequent use of these interfaces causes major part of energy consumption[6][7]. Compared to standby time, the battery lifetime is only a fraction if such interfaces are used[5].

This EnerDroid stands for Energy Android project which can be further explained as an android tool that works on top of GreenDroid processor to make use of capabilities provided by the Processor to enhance the battery life of android device. So this EnerDroid project stores some relevant amount of data in SQLite database; that is the most efficient database used by android device to provide fast mechanism to access required information efficiently to reduce the need of fetching the same information by using network bandwidth and hence enhances the battery life. To make all the data available locally, we are providing synchronization mechanisms that work on the principal of compression.

Section 2 presents the Related work done related to the different energy issues . Next section 3 presents the motivation for this work whereas section 4 defines the problem shortly to mathematically represent it. Section 5 gives implementation details. Section 6 and 7 gives results and conclusion respectively.

II. RELATED WORK

The Table I represents the papers related to our project. EnerDroid tool is implemented for the first time to make use of energy efficiently while transmitting data over network.

TABLE I RELATED WORK

Cite	Name of Paper	Description
[2]	GreenDroid	Helps developer detect battery drain issues automatically.
[3]	An investigation into Engergy-Saving Programming Practices for Android phone App Development	Results of this show that sending larger file is more energy efficient than sending smaller files.
[8]	ADEL	Automatically detects the energy leaks for Android Applications.
[10]	eLens	Gives code level estimates of energy by 324nalysing the implementation of a mobile application.
[11]	eDoctor	Diagnosis abnormal battery Drain issues automatically.
[12]	Sesame	Builds self energy model without external assistance.
[13][14]		Tracks flow of privacy sensitive information through other application as well as help marking and tracking certain information in a program at run-time.

III. MOTIVATION

As the use of Android Smartphones is becoming very common and to wide extent for performing the tasks that were earlier done by computers, laptops and desktops; it has become very essential for increasing the battery performance. So that users dont get frustrated due to low battery performance caused by Android Smartphones. Earlier work was done for various reasons viz diagnosing battery drain issues [8] [11], building self energy model [12] and also some were related to influence of compression procedure on energy consumption [4] [9] for other than Android platform. There is also a huge market to explore with this emerging new technology that is becoming famous as well as common to even ordinary people. On the other hand Yepang Liu, Chang Xu, S.C. Cheung, and Jian Liu [2] diagnosed energy problems and found out two main reasons namely missing deactivation of sensors or wake locks and cost ineffective use of sensory data which caused energy inefficiency for Android applications. Their work also introduced some scope for saving energy during data transmission over network. This was the main motivation which leads us to think of some way so that some energy would be saved while using internet over Android device.

IV. PROBLEM DEFINITION

Efficiently managing data transmission over network to increase battery life is our motive. Based on our analysis we found that energy consumption depends on time connected to server measured in seconds, Size of data measured in Mb and speed of connection i.e data connection measured in Mbps. So we formulate our approach as an tool EnerDroid on top of GreenDroid for efficient energy management of data transmission over network for Android smartphones.

V. IMPLEMENTATION DETAILS

Basically EnerDroid project depends on several research topics which include energy efficiency analysis in terms of network consumption and availability of data. This is for the first time this method is used for Android device to attain energy efficiency. GreenDroid work relates to several research topics, which include energy efficiency analysis, energy consumption estimation, resource leak detection, and information flow tracking. Some of them particularly focus on Android phone applications. In addition to these research topics which may fail in case of android application performing network transactions. For enhancing, we can create logic to make network transaction efficient that is the future scope of GreenDroid. In this section we describe the system architecture, mathematical Model and the workflow of the EnerDroid tool.

A. System Architecture

In figure 2, earlier application describes one of android smartphone application utilizing full bandwidth rate for data transferring over network which causes early battery drain, whereas other android smartphone using EnerDroid tool is showing our implementation approach. In our approach we are aiming to efficiently utilize data consumption rate and network bandwidth consumption rate so that we can extend battery life.

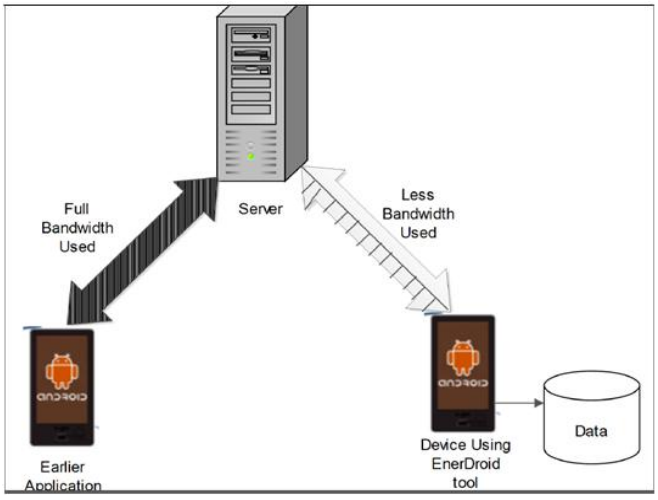

Fig. 1 System Architecture

B. Mathematical Model

Let S be the system that uses the minimum battery for performing uploading or downloading function. So, system can be represented as S = {I, O, F}

Where,

I is the input set which contains the file to be uploaded or downloaded from server.

I = {U, D, NS}

Where,

U = Select file for uploading to server.

D = Select file for downloading from server.

NS = Select the preferred network.

O is the output set which contains the result of the user's need.

O = {US, DS, NS}

Where,

US = File uploaded successfully.

DS = File downloaded successfully.

NS = Preferred network selected.

F represents the functions that are to be performed for achieving the required output when selecting the input.

F = {LC, C, DS, S}

F(LC) represents locally checking the database for a search.

F(C) represents the compression function.

F(DS) represents the data synchronization with the server.

F(S) represents the function for selecting preferred network.

1) Using Set Theory: To mathematically model the problem we will take seven samples of E. We will take the minimum of these samples. Such that E = E1, E2, ..., E7. Suppose minimum (E) = E2: That means:

$$\frac{T_c + S_d}{D_C} \subseteq \frac{T_C + S_D}{D_C}$$

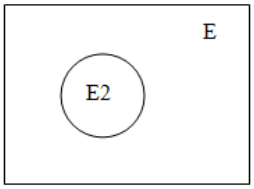

Fig. 2 Set Theory Representation

The figure 2 depicts the diagrammatic representation of set theory explained here.

C. Life Cycle of EnerDroid

The basic methods that are used during the lifecycle of an application are onCreate(), onResume() and onRestart(). Let us understand their importance as well as sequence. The very first method that starts on opening an application or start using an application is onCreate() followed by onResume(). onCreate() method is called once in its lifetime. The onCreate() method binds the business logic with the user interface. Sometimes it happens that application goes on running in background but is invisible on foreground. That is bought to run in foreground by calling onResume() method. As soon as the user finishes sub-activity, it will be stopped and onRestart() method will be called which will resume the main activity.

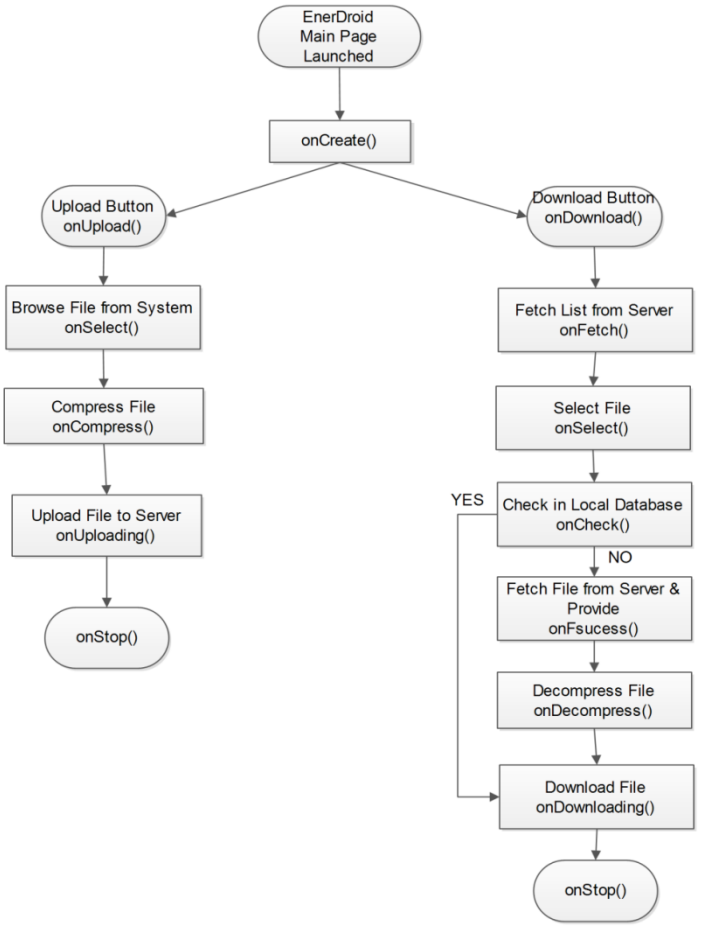

Fig. 3 Lifecycle of EnerDroid.

The above figure 3 shows the lifecycle of EnerDroid tool. On launching EnerDroid tool, two buttons will appear one for upload and other for download. On selecting onUpload() method it will lead for the process of uploading where for browsing file from system will be done using onSelect() method. File will compressed using onCompress() method for efficient use of energy. onUploading() method will upload file to server followed by onStop() method.

onDownload() method will lead for fetching list from server by using onFetch() method. onSelect() method will help in choosing file to download. Once the file is selected it will check in local database by onCheck() method if the file is present in it will be provided to user otherwise it will fetch file from server and provide using onFsuccss() method. Decompression will be done using onDecompress() method.onDownloading() method will download file and give it to user followed by onStop() method.

D. Modules

Based on these we have to design three modules as:

1) Module 1 : Compression : Compression will be done automatically. User will have no information about this process neither the user will be asked for or will we be seeking permission from them. Algorithms for compression process are mentioned in subsection
   The coded output of algorithm i is given to algorithm iii, which compresses the coded data. Algorithm ii is used to decode the data that is coded using algorithm i whereas the process to decompress is mentioned below algorithm iii.
2) Module 2: Synchronizing Online: When second condition mentioned above (i.e data is unavailable locally) is found true the flag will be maintained on server. If data updates it will set the flag and then with tool synchronization with server it checks for flag. Data is downloaded if and only if flag found to be set.
3) Module 3: Network Selection: This would be a user defined event where user will be asked for selecting the network so as to increase the energy efficiency. Particularly, we are most interested in the practices of making HTTP request as these are the most important among many methods for accessing the Internet. Android phones visit various web applications through HTTP requests. Furthermore, many modern Android applications adhere to the REST architectural style and they treat remote resources as URL links. Thus, these RESTful applications need to make HTTP requests to visit remote resources and databases.

E. Algorithm

The algorithm used for compression are given below. Algorithm 1 generates the code that is decoded using algorithm Compression of code generated using algorithm 1 is done by algorithm 3

Algorithm 1 Algorithm for generating code for data

Input: Data to be downloaded or uploaded.

Output: Data in coded form.

1: Count the number of codes for each code length.
2: Let bl-count $[N]$ be the number of codes of length N, N $\geq$ 1.
3: Find the numerical value of the smallest code for each code length:
4: code $= 0$;
5: bl-count $[0] = 0$;
6: For bits $= 1$; bits $\leq$ MAX$_B ITS; bits\pm$
7: code = code + bl$_c ount[bits1]$ ¡¡ 1;
8: next-code$[bits] = code$;
9: Assign numerical values to all codes, using consecutive values for all codes of the same length with the base values determined at step 2. Codes that are never used $(whichhaveabitlengthofzero)$ must not be assigned a value.
10: For n $= 0$; n $\leq$ " max-code; n++
11: len $=$ tree $[n]$.Len; if len $\neq 0$
12: tree$[n]$.Code $=$ next-code$[len]$;
13: next-code$[len]$++;
14: Code generated

Algorithm 2 Algorithm for Decoding

Input: Coded data generated from algorithm 1.

Output: Data uploaded or downloaded.

1: Do
2: read representation of code trees
3: Do$(untilendofblockcoderecognized)$
4: decode literal/length value from input stream
5: ifvalue ≤ 256
6: copy value $(literalbyte)$ to output stream end of block(256)
7: break
8: Else$(value$ = 257....258
9: decode distance from input stream
10: move backwards distance bytes in the output
11: stream, and copy length bytes from this
12: position to the output stream.
13: While not last block

Input :- The concatenated code words generated from algorithm 1 are:
00010111000101111100010100001101101110011110101111111 0
and the final encoded values are:-
(0; 0),(1; 0),(0; 1),(1; 1),
(3; 1),(2; 0),(3; 0),(5; 1),
(5; 0),(2; 1),(4; 0),(6; 1),
(7; 1),(13; 1),(7; 0),(8; 1),
(11; 1),(16; 1),(3; 0).

Table II: Dictionary for Algo 2

0	NULL	-
1	0	(0, 0)
2	00	(1, 0)
3	1	(0, 1)
4	01	(1, 1)
5	11	(3, 1)
6	000	(2, 0)
7	10	(3, 0)
8	111	(5, 1)
9	110	(5, 0)
10	001	(2, 1)
11	010	(4, 0)
12	0001	(6, 1)
13	101	(7, 1)
14	1011	(13, 1)
15	100	(7, 1)
16	1111	(8, 1)
17	0101	(11, 1)
18	11111	(16, 1)
19	10	(3, 0)

Algorithm 3 Algorithm for Compression

Input: Code Generated from Algorithm 1.

Output: Compressed Data.

1: Initialize the dictionary to contain all strings of length one.
2: Find the longest string W in the dictionary that matches the current input.
3: Emit the dictionary index for W to output and remove W from the input.
4: Add W followed by the next symbol in the input to the dictionary.
5: Go to Step 2.

The decoding algorithm works by reading a value from the encoded input and outputting the corresponding string from the initialized dictionary. At the same time it obtains the next value from the input, and adds to the dictionary the concatenation of the string just output and the first character of the string obtained by decoding the next input value. The decoder then proceeds to the next input value (which was already read in as the ”next value” in the previous pass and repeats the process until there is no more input, at which point the final input value is decoded without any more additions to the dictionary. The result is the concatenated code words i.e.
000101110001011111000101000011011011100111101011111110.

VI. RESULTS

Based on our analysis and using above mentioned equation, we have got the below shown graphs assuming certain values. We assume the constant k = 1. Graph shows the results for energy consumed when the size of data being transferred is 100 Mb and the speed of data connection is 1 Mbps. So the results show that as the time to transfer data increases the energy consumption rate also increases.

Graph 2 shows the energy consumed when the time to transfer data over network is 10 min and the speed of data connection is 1 Mbps. So the results show that as the size of being transferred increases the energy consumption rate also increases.

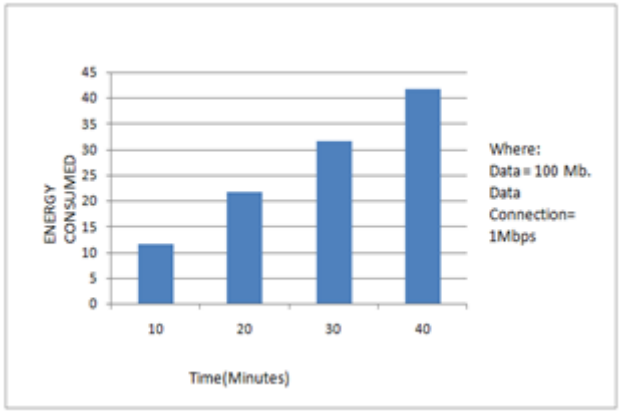

Fig. Graph 1

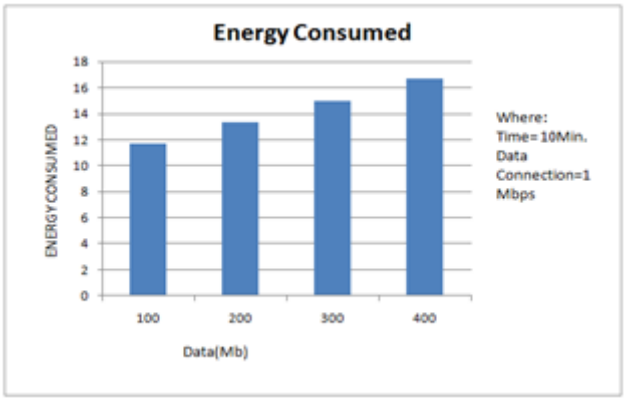

Fig. Graph 2

Graph 3 shows the energy consumed when the time to transfer data over network is 10 min and the size of data is 100 Mb. So the results show that as the speed of data connection increases the energy consumption will be less.

So we prove that intelligent management of the factors affecting the battery drain can make battery life long. Based on above graphs we can define these research questions as how we can reduce time to transfer data over network,

make efficient use of data as well as how can we increase the speed of data connection. As a solution to above research question we will make use of compression and decompression techniques, data synchronization and user defined selection of network to efficiently manage the battery drain issues by implementing a tool named EnerDroid. On an average if a user downloads 100-200 mb data from server daily it will impact about 30-40% of battery, so implementing module we can reduce battery consumption and hence enhance the performance by 30-40%,when connected to Wi-Fi network of speed 1mbps.

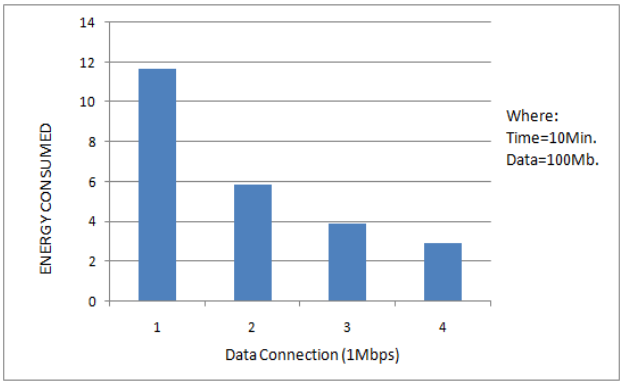

Fig. Graph 3

VII. CONCLUSION

Based on our research we found that android application identified two types of coding phenomena that caused energy wastages viz. inefficient use of network bandwidth and inefficient use of data. Based on these finding we proposed an approach for solving energy problem caused while data transmission over network. Our approach systematically explores an applications data consumption rate and network bandwidth consumption rate. So the battery performance is increased to an inevitable amount and relieving the user to use Android device without anxiety.

ACKNOWLEDGMENT

The author would like to thank RSCOE Computer Engineering Department as well as their researchers for making their resources readily available and teachers for their guidance. We are thankful to the authorities of Savitribai Phule Pune University and members of cPGCON2015 conference organized by MET Bhujbal Knowledge City for their constant guidelines and support. We are also thankful to the reviewers for their valuable suggestions. We also thank the authorities for providing required infrastructure and support. Last but not the least we would like to extend a heartfelt gratitude to family members and friends for giving their efforts.

REFERENCES

[1] S. Analytics, ”Cellphone Energy gap is widening”, SEPTEMBER 2014.

[2] Yepang Liu, Chang Xu, S.C. Cheung, and Jian Lu, ”GreenDroid: Automated Diagnosis of Energy Inefficiency for Android phone Applications”,IEEE TRANSACTIONS ON SOFTWARE ENGINEERING. VOL. 40, NO. 9, SEPTEMBER 2014.

[3] Ding Li and William G. J. Halfond, ”An Investigation into Energy-Saving Programming Practices for Android Android phone App Development”,India Copyright 14 ACM 978-1-4503-2844-9/14/06. GREENS14, June 1, 2014.

[4] Ronny Hans, Manuel Zahn, Ulrich Lampe, Ralf Steinmetz, Apostolos Papageorgiou, ”Energy-efficient Web Service Invocation on Mobile Devices: The Influence of Compression and Parsing”, In Proceedings of 2nd International Conference of Mobile Services. (MS 2013).

[5] Saswat Anand, Mayur Naik, Hongseok Yang,, ”Automated Concolic Testing of Android phone Apps”, SIGSOFT12/FSE-20, November 11 16, 2012, Cary, North Carolina, USA. Copyright 2012 ACM 978-1-4503-161 9/12/11.

[6] G. P Perrucci, F.H Fiitzek, G Sasso, W. Kellerer and J. Widmer, ”On the impact of 2G and 3G network usagefor mobile phones’ battery life ”, In Proceedings of 15th European Conference of Mobile Services. (EWC2009).

[7] T. Pering, Y. Agarwal, R. Gupta, R. Want, ”CoolSpots:Reducing the power consumption of wireless mobile devices with multiple radio interfaces”, In Proceedings of 4th International Conference of Mobile Services. (MobiSys 2006).

[8] Lide Zhang, Mark S. Gordon, Robert P. Dick, Z. Morley Mao, Peter Dinda, Lei Yang, ”ADEL: An Automatic Detector of Energy Leaks for Android phone Applications”, Copyright 2012 ACM 978-1-4503-1426- 8/12/09.. CODES+ISSS12, October 712, 2012,Tampere, Finland.

[9] Aleksandar Milenkovi, Armen Dzhagaryan, Martin Burtscher, ”Performance and Energy Consumption of Lossless Compression/Decompression Utilities on Mobile Computing Platforms”, IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems. 2013.

[10] Shuai Hao, Ding Li, William G. J. Halfond, Ramesh Govindan, ”Estimating Mobile Application Energy Consumption using Program Analysis”,ICSE 2013, San Francisco, USA, 978-1-4673-3076-3/13/2013.

[11] Xiao Ma, Peng Huang, Xinxin Jin, Pei Wang, Soyeon Park, Dongcai Shen, Yuanyuan Zhou, Lawrence K. Saul and Geoffrey M.Voelker, "eDoctor: Automatically Diagnosing Abnormal Battery Drain Issues on Android phones.", . 2013.

[12] Mian Dong and Lin Zhong, "Self-Constructive High-Rate System Energy Modeling for Battery-Powered Mobile Systems", Copyright 2011 ACM 978-1-4503-0643-0/11/06. MobiSys11, June 28-July 1, 2011, Bethesda, Maryland, USA.

[13] James Clause, Wanchun Li, and Alessandro Orso, "Dytan: A Generic Dynamic Taint Analysis Framework", Copyright 2007 ACM 9781595937346/07/0007. ISSTA07, July 912, 2007, London, England, United Kingdom.

[14] William Enck, Peter Gilbert, Byung-Gon Chun, "TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Android phones"

[15] Kyu-Han Kim, Alexander W. Min, Dhruv Gupta, Prasant Mohapatra, Jatinder Pal Singh, "Improving Energy Efficiency of Wi-Fi Sensing on Android phones"

[16] Muhammed Fatih Bulut, Murat Demirbas, "Energy Efficient Proximity Alert on Android"

