*International Journal on Applicationsin Information and Communication Engineering*
*Volume 2:Issue 6: June 2016, pp 34-38. www.aetsjournal.com* ISSN (Online) : 2394 - 6237

# Development Efficient Clustering Based Enforcing Security On Traffic In Android

M.Karthik, P.Nirmala Devi

***Abstract*—** The popularity and advanced functionality of mobile devices has made them attractive targets for malicious and intrusive Applications. Although strong security measures are in place of most mobile systems the area, where these systems often failed is the reliance on the user to make decisions that impact the security of a device. As our primary example, Android relies on users to understand the permissions that an app is requesting and to base the installation decision on the list of permissions. Previously research has shown that, this reliance on users is ineffective, as most users do not understand or consider the permission information's. We propose a solution that leverages a method to assign a risk score to each app and display a summary of that information to users. Results from four experiments are reported in which we examines the effects of introducing summary risk information and how best to convey such information to a user. Our results show that the inclusion of risk score information's has significant positive effects in the selection process and can also lead to more curiosity about security related information.

***Keywords*—** Android, JAVA, server, Internet, Linux

## I. INTRODUCTION

In recent years smart mobile devices have become pervasive. More than 50 percent of all mobile phones are now smart phones and this statistic does not account for other devices such as tablet computers that are running similar mobile operating systems [1]. The future are envisioned to consist of hundreds of inexpensive nodes, that can be readily deployed in physical environments to collect useful information (e.g. Seismic, acoustic, medical and surveillance data) in a robust and autonomous manner. Android devices have widespread adoption for both personal and business use. From children to the elderly, no voices to experts, and in many different cultures around the world, there is a varied user base for mobile devices [2-4].

Our entire digital lives are often stored on the devices, which contain contact lists, email messages, passwords, and access to files stored locally and in the cloud. Possible access to this personal information by unauthorized parties puts users at risk, and this is not where the risks end [5]. These devices include many sensors and are nearly always with us, providing deep insights into not only our digital lives but also our physical lives. This access means that any application that is allowed to run on the devices potentially has the ability to tap into certain aspects of the information. In the benign case the access is performed to provide useful functionalities, but in other scenarios it may be used to collect a significant amount of personal information and even as a means to have some adverse impact on a user [6].

In contrast, for mobile devices, a person often downloads and uses many apps from multiple unknown vendors, with each app providing some limited functionality. Additionally, all of these unknown vendors typically submit their apps to a single or several app stores where many other apps from other vendors may provide similar functionality [7].

In Android an app must request a specific permission to be allowed access to a given resource. Android warns the user about permissions that an app requires before it is installed, with the expectation that the user will make an informed decision. The effectiveness of such a defense depends to a large degree on choices made by the users [8]. Therefore, an important aspect of security on mobile devices is to communicate the risk of installing an app to users, and to help them make a good decision about whether to install a given app. Android's current risk communication mechanism has been shown to be of limited effectiveness. Studies have demonstrated that users tend to ignore the permissions that an app requests [9-12] and some recent work has attempted to overcome some of these limitations. Proposed several improvements, including modifying permission category headers, emphasizing risk, reducing the number of permissions, enabling customized permission lists, incorporating user reviews and thinking the timing of when and how permissions are granted. Line proposed an approach which incorporates crowd sourced [13].

When using symbols, a key decision is whether to frame the categories as varying along the dimension of risk, where more symbols represent greater risk, or the dimension of safety, where more symbols represent greater safety [4]. The addition of new security indicators not only may decrease the frequency of risky user behaviors, but it may also facilitate the use of smart phones for online transactions by more individuals [15, 16].

## II. SYSTEM MODEL

An easy risk comparison among them applications is that provides the similar functionalities. We believes that one reason why current permission information's is often ignored by users is that it is presented in a "standalone" fashion and in

M.Karthik, PG Scholar, Computer science and Engineering, Department of CSE, Annai Mathammal Sheela Engineering College, Namakkal, Tamilnadu, India.( Email: dingube@gmail.com)

P.Nirmala Devi, , Associate Professor, Computer science and Engineering, Department of CSE, Annai Mathammal Sheela Engineering College, Namakkal, Tamilnadu, India

a way that requires a lot of technical knowledge and time to distill useful information, making comparison across apps difficult.

An important feature of the mobile app ecosystem is that users often have choices and alternatives when choosing a mobile app. If a user knows that one app is significantly riskier than another but provides the same or similar functionality, then this fact may cause the user to choose the less risky one. This will in turn provide incentives for developers to better follow the least-privilege principle and request only necessary permissions [17-19].

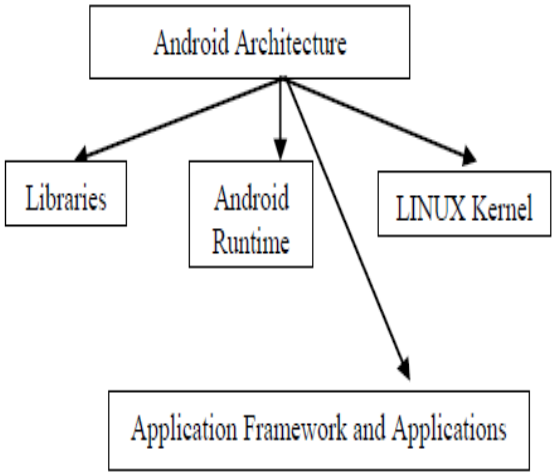


Fig 1: Android Architecture layers

*A. Attractive User Interface*

Android Operating System basic screen provides an easy and attractive user interface. In this area of internet and technology, microcontrollers are used to accelerate machine to machine (M-M) and brain-to-machine (B2M) communications. The central processing unit of the robot can be ARM (Advanced RISC Machine) processor. The LPC2148 microcontroller is a 32bit ARM7TDMI-S CPU with embedded trace supports and real-time emulation that incorporates the microcontroller with 64kB and 512kB of high speed embedded flash memory. A 128-bit wide memory interface and rare accelerator architecture permits 32 bit code execution at the ultimate clock rate. An alternative 16-bit Thumb mode reduces critical codes by more than 30 % with minimal achievement penalty.

## III. PROPOSED METHODOLOGY

*A. Android*

In the proposed scheme, the user opens the Android application in his/her Android operated device and selects search criteria i.e. either driving license search or registration search. In case of driving license search, we need to provide driving license number and in case of registration search, we need to provide the registration number and then initiate the search process which is described in the below phases. In the first phase, the driving license or registration number is sent to the web service. In the second phase, the web service searches the database on the server. In the third phase the web server forwards the search result back to the android device.

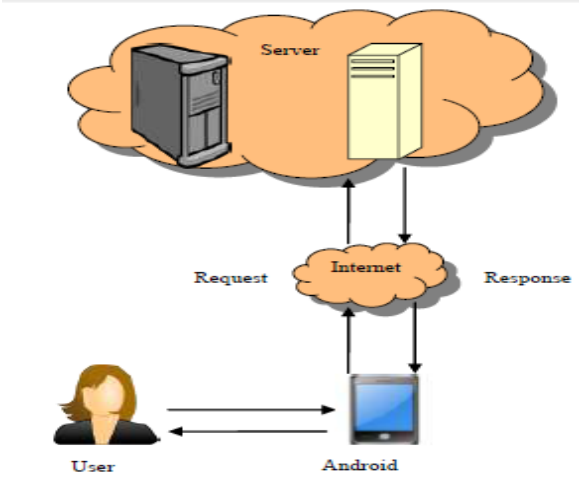


Fig. 2 User Searching Method

*B. Proposed Scheme*

This proposed scheme is composed of four parts, namely Database (DB) at server, Web Service (WS) at server, Android application (App) and user. The entire sequences of searching data details and registration details by a user from the database present at server via an Android application and WS is shown in Figure 3. In the proposed scheme, the user opens the Android application in his/her Android operated device and selects search criteria i.e. either driving license search or registration search. we need to provide the data and document details, then initiate the search process which is described in the below phases.

In the first phase, the server sent to data to the web service. In the second phase, the web service searches the database on the server. In the third phase the web server forwards the search result back to the android device.

*C. Architecture*

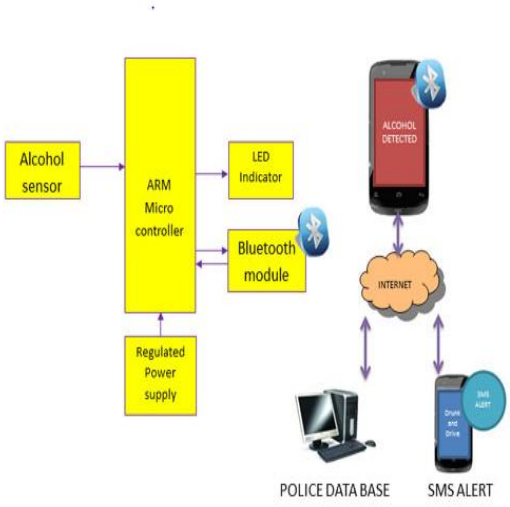


Fig.3 System Architecture

*D.Multitenant for Android*

Multi-tenancy, which means that software running on a server provides services to many users, is one of important features for cloud computing. From the viewpoint of both economy and ecology, it is beneficial to share hardware resources among users. Using a mobile OS would be more effective than using a desktop OS because the resource requirements of mobile OSs are smaller. However, to the best of our knowledge, there is still no service that uses Android as multi-tenant system. The proposed system discusses the multi-tenant architecture for Android and how to construct it.

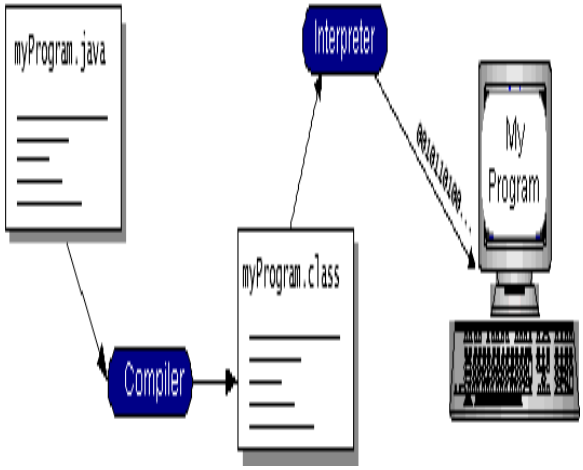


Fig.4 How Java Is Working

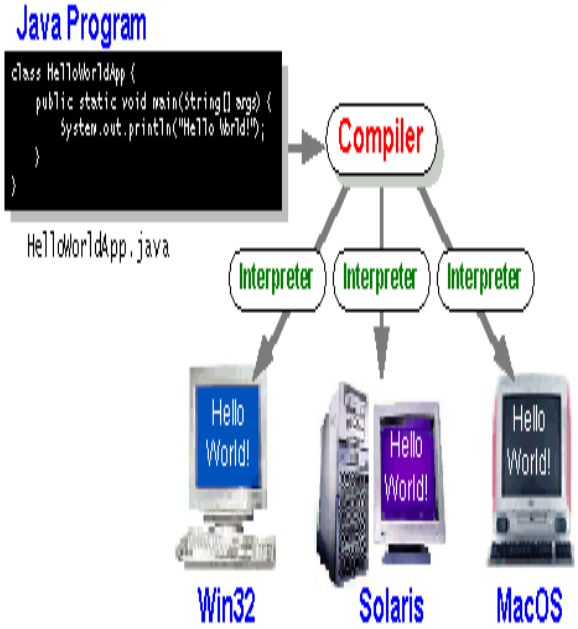


Fig.5 Java Runs In Different Platforms

## IV. THE JAVA PLATFORM

The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed.

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and Maces. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms. The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces these libraries are known as *packages*.

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and Maces. The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces these libraries are known as *packages*.

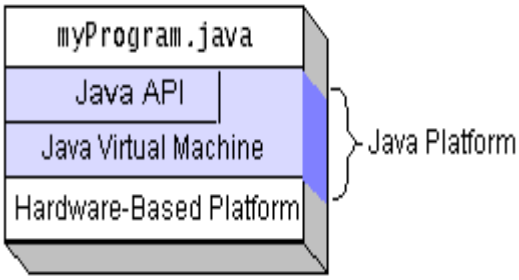


Fig.6 Program Runs In Java Platform

## V.PERFORMANCE ANALYSIS

Configuring the open VPN in the smart phone with certificates; when the certificates for the clients are packaged, it is necessary to protect them with a pass phrase.

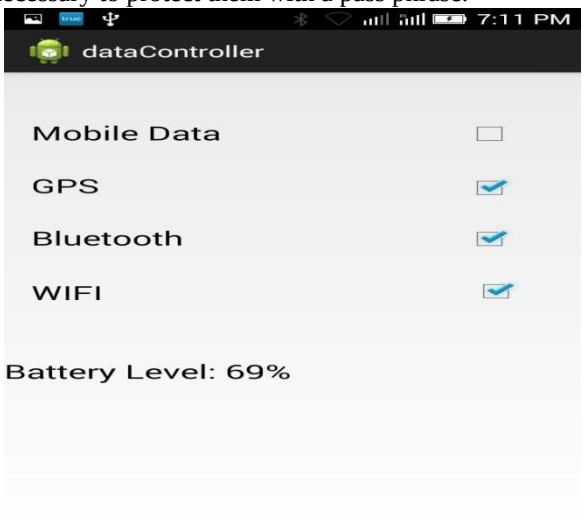


Fig.7 VPN opened page

The certificates will be copied to the SD card and imported to the smart phone. Then, it is necessary to setup the VPN creating a new connection. The parameters to configure are: _ IP of the VPN server.

_ Port and protocol: 80/tcp _ Redirect ate way: to route all the traffic through the

VPN _ LZO compression: To compress the data _ Cipher Algorithm: AES-256-CBC _ Size of the cipher: 256 bit

3) Implementing firewall policies in Android: The objective is to apply a DENY default policy. The rules are set as follow:

_ Interface lo: ACCEPT any IN/OUT

_ Interface eth0/rmnet0: DENY IN/OUT. Exception: Traffic to the VPN IP (necessary to establish the tunnel and forward the traffic)

_ Interface tun0: ACCEPT incoming SSH. ACCEPT OUT traffic to ANY to route the traffic through the VPN.

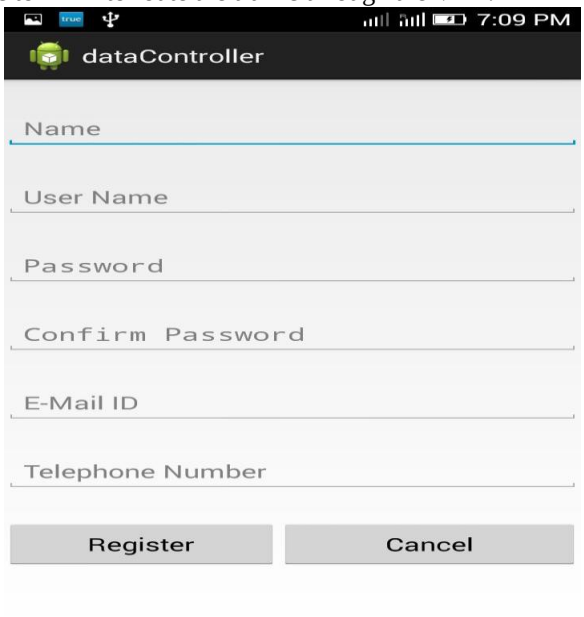


Fig.8 Security Page

With these rules we will guarantee that when connecting through a public WiFi or 3G will not go on clear, avoid sniffing, and any traffic from the local network or Internet will be denied.

### *A. Disabling the SD card*

Connecting the external device (USB stick, SD card, etc) to 'secure' the system can be a big security, risk whether it is a smart phone and a desktop or a server, etc. In the case of Android, the situation is the same or even worse. By default android mounts the SD card as a FAT file system, with 'no exec', 'no dev'. The first question is why use FAT instead of ext3 or ext4, but besides that, is the 'no exec' enough to prevent running the applications? Mario Ballano, a security researcher from Symantec, published a very interesting article about how it is possible to use the SD card to hijack in order to steal information or execute malicious code. Besides the problem described by Mario, there is an additional issue: the lack of encryption on the SD card.

## VI. Results And Discussions

### *A. Performance Analysis*

In this section, the performance of this proposed scheme in analyzed in terms of internet bandwidth, complexity of using the scheme, extensibility, availability.

***1)*** *Requirement of Internet Bandwidth:* This scheme requires less bandwidth for searching as it does not need to load the database or website. Internet is required to send the registration number or license number to the web service and get the search result from the web service.

***2)*** *Complexity of using the scheme:* Having a simple technique of searching, the complexity of the proposed scheme will be very less. All users just need to provide driving license number for searching driving license details or registration number for finding registration details.

***3)*** *Extensibility:* As web service is used, the same can be used for apps for mobiles having other OS like Windows, etc. We don't need to prepare separate back end for different platforms.

***4)*** *Availability:* For the proposed scheme we need android operated mobile devices which are low in cost and highly available.

## VII. Conclusion And Future Work

The results from four user studies validated our hypothesis that when risk ranking is presented in a user friendly fashion, eg., translated into categorical values and presented early in the selection process, it will lead users to select apps with lower risk. The majority of participants preferred to have such a risk metric in Google Play Store. We expect that adding a summary risk metric would cause positive changes in the app ecosystem. When users prefer lower-risk apps, developers will have incentives to better follow the least-privilege principle and request only necessary permissions.

It is also possible that the introduction of this risks core will cause more users to pay for low risk apps. Thus, this creates an incentive for developers to create lower risk apps that do not contain invasive ad networks and in general over-request permissions. Our studies are not the last word on the question of how to best present risk information. For example, we have also not examined how the risk score interacts with other factors to affect a user's choice, such as user ratings in the natural setting and whether an app is free or not. Also of interest is a show user behave when choosing among a list of search results (as opposed to choosing between two options).These topics are important ones for future research.

## References

[1] M. Bhardwaj, T. Garnett, and A.P. Chandrakasan. Upper Bounds on the Lifetime of Sensor Networks. In Proceedings of International Conference on Communications, 2001.

[2] J.H. Chang and L. Tassiulas. Energy Conserving Routing in Wireless Ad-hoc Networks. In Proceedings of IEEE INFOCOM, 2000.

[3] J.H. Chang and L. Tassiulas. Maximum Lifetime Routing in Wireless Sensor Networks. In Proceedings of Advanced Telecommunications and Information Distribution Research Program, College Park, MD, 2000.

[4] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. Max–flow Min–cut Theorem. In Introduction to Algorithms, MIT Press, 1998.
[5] J. Edmonds. Edge –disjoint branching. In Combinatorial Algorithms, Academic Press, 1973.
[6] W. Heinzelman, A.P. Chandrakasan, and H. Balakrishnan. Energy - Efficient Communication Protocols for Wireless Micro sensor Networks.In Proceedings of Hawaiian International Conference on Systems Science, 2000.
[7] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In Proceedings of 5th ACM/IEEE Mobicom Conference, 1999.
[8] C. Intanagon wiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In Proceedings of 6th ACM/IEEE Mobicom Conference, 2000.
[9] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Mobile Networking for Smart Dust. In Proceedings of 5th ACM/IEEE Mobicom Conference, 1999.
[10] K.Kalpakis, K.Dasgupta, and P. Namjoshi. Maximum Lifetime Data Gathering and Aggregation in Wireless Sensor Networks. To Appear in Proceedings of IEEE Networks'02 Conference, 2002.
[11] B. Krishnamachari, D. Estrin, and S. Wicker. Modeling Data-Centric Routing in Wireless Sensor Networks. In Proceedings of IEEE Info com, 2002.
[12] X. Lin and I. Stojmenovic. Power–aware Routing in Ad Hoc Wireless Networks. In University of Ottawa, TR-98-11, 1998.
[13] S. Lindsey and C. S. Raghavendra. PEGASIS: Power Efficient Gathering in Sensor Information Systems. In Proceedings of IEEE Aerospace Conference, 2002.
[14] S. Lindsey, C. S. Raghavendra, and K. Sivalingam. Data Gathering in Sensor Networks using the Energy*Delay Metric. In Proceedings of the IPDPS Workshop on Issues in Wireless Networks and Mobile Computing, 2001.
[15] L. Lov_asz. On two minimax theorems in graph theory. In Journal of Combinatorial Theory Series B, Vol. 21, 1976.
[16] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler. Supporting Aggregate Queries Over Ad-Hoc Wireless Sensor Networks. In Proceedings of 4th IEEE Workshop on Mobile Computing and Systems Applications, 2002.
[17] R. Min, M. Bhardwaj, S.H. Cho, A. Sinha, E. Shih, A. Wang, and A.P. Chandrakasan. Low-Power Wireless Sensor Networks. In VLSI Design, 2001.
[18] J. Rabaey, J. Ammer, J.L. da Silva Jr, and D. Patel. PicoRadio: Adhoc Wireless Networking of Ubiquitous Low-Energy Sensor/Monitor Nodes. In Proceedings of the IEEE Computer Society Annual Workshop on VLSI, 2000.
[19] S. Singh, M. Woo, and C. Raghavendra. Power–aware Routing in Mobile Ad Hoc Networks. In Proceedings of 4th ACM/IEEE Mobicom Conference, 1998