International Journal of Security and Its Applications
Vol. 11, No. 5 (2017), pp.27-38
http://dx.doi.org/10.14257/ijsia.2017.11.5.03

Mobile App User Licensing with Little or No Backend Server

Anis Prasla, Sabah Mohammad and Jinan Fiaidhi

Computer Science, Lakehead University, Thunder Bay, Canada
{Aprasla, sabahmohammad, jfiaidhi}@lakeheadu.ca

Abstract

For decades software licensing has been relaying on copyright registration and the declaration of this copyright at the software to be accepted and downloaded by the users. In this case the software developer need to work closely with legal departments and rely on the copyright laws where such laws enforcement vary from one country to other. The complexity of enforcing this licensing model largely come from the robust way of formulating the end-user license agreement (EULA) and the existence of a backend server that can monitor the usage of the software. Obviously the enforcement of this model may prove to be legally impossible as there will be many users who do not care about the software license as well as there will be an associated expenses with using the backend server. In this paper, a new method and a prototype for licensing mobile application that are uploaded on public cloud. In this method the users of the mobile app starts by using a declarative form of the License but they need to provide user specific data including the mobile unique device id, operating system and brand. The method also includes activating the application on the computing device using the device specific information. This licensing model protects software piracy and license vulnerability issues. The developed prototype for this type of licensing model has been applied for android applications as there are tons of Android apps on application stores at different domains. Experimental results show the process of integrating the licensing library with any android applications is easy without changing the existing application code and avoiding lengthy development efforts to secure mobile apps with fully licensed app and no legal overhead.

***Keywords**: Software licensing, EULA, Android License library, Software Copyright*

1. Introduction

The emergence of cloud computing is changing how organization and individuals purchase and use software. This technology promises to streamline the on-demand provisioning of software, hardware, and data as a service, achieving economies of scale in IT solutions' deployment and operation. For enterprises that rely on software to maintain a market share, the software licensing model can strongly influence the return on software investment. Yet choosing a model can be daunting, involving considerations from total licenses purchased to the resources available to install and set up licensing systems [21]. Many application fails to provide licensing security to their app due to increasing complexity of the legal process and development time to integrate and enforce the software license. To prevent the potential errors or bugs resulted from adding functionality, we designed a declarative library that can be automatically used to add a license to any application and to verify its usage. The developed prototype for this type of licensing model has been applied for android applications as there are tons of Android apps on application stores at different domains. Experimental results show the process of integrating the licensing library with any android applications is easy without changing the existing application code and avoiding lengthy development efforts to secure mobile apps with fully licensed app and no legal overhead.

2. Related Research

Software licensing has historically been based on a "trust" model in which the user (*i.e.*, licensee) is presumed to be honest and trustworthy and to abide by the legal requirements of the license. Under the trust model, a software license typically accompanies a software product to explain the terms of use. For instance, the software license might dictate that the program code is to be installed on only one computer, and may be used to make one backup copy. Common types of licenses include "shrink wrap" licenses, "online" licenses, and "site" licenses. [22]. Ache, Pence, and Napster [1] presented a simplified licensing method by acquiring the unique ID of the user device. This method works when the user request an access to the app from his/ere device, the unique ID of the device is checked at the license server. If the ID exists then user can access it otherwise if ID doesn't exist the maximum number of authorized device is checked and depending on the authorization provided the user will be allowed or denied the access. If access is allowed, then the device is added to License server and token is provided for that particular device which is checked next time when user request for access. The major drawback of this method besides using a backend server, it doesn't allow re-licensing their original the device(s) is/are lost or deactivated. In Mu, Cui, and Rao [3] the authors provided an enhanced licensing model that incorporate different types of attack and threats on mobile application which leads to breaking the software license and they have also provided the measures that need to be taken to prevent it. In Konferenzbeitrag *et al.*, [10] the authors incorporate the Google play licensing and explained how the License verification library (LVL) can be used to enforce software licensing. Google Play Licensing is a network based service that let the application include the LVL and can make a request to a service hosted by Google Play client application. The client need only to send a request to the licensing server and waits for the result. Google play licensing allows the developer to limit the number of devices that can be used by users with same applications but it's based on device count. For example, if number of devices allowed is two then once user install on the two devices the count is updated and when the user tries to install on a third device, the service will give an error. So, to access in a third device the user must uninstall from a previous device. This model helps to some extent to provide more robust licensing, however, if the licensing credentials are hacked or shared then anyone will be able to use the license. In Karvell, Donner, and Garg [11] the authors presented yet another enhanced method for software licensing by storing the user credentials for verify and providing the software license to the device based on the login information. The license file is saved locally with the necessary user information and device specific information. So, whenever user tries to access the software, it checks for the license file and verifies that the device is authorized. There are many other approaches to provide licensing and this section cannot cover all of them [*e.g.*, 23, 24, 25 and 26]. Our critical survey was based on scholarly search engines like Google Scholar and based on major search keywords described in Figure 1. Our scoping review on software licensing targeted three objectives:

1. Systems & methods for software licensing (include 17 papers)
2. Software Licensing Piracy (include 15 papers)
3. Software Licensing Piracy Prevention (include 16 papers)

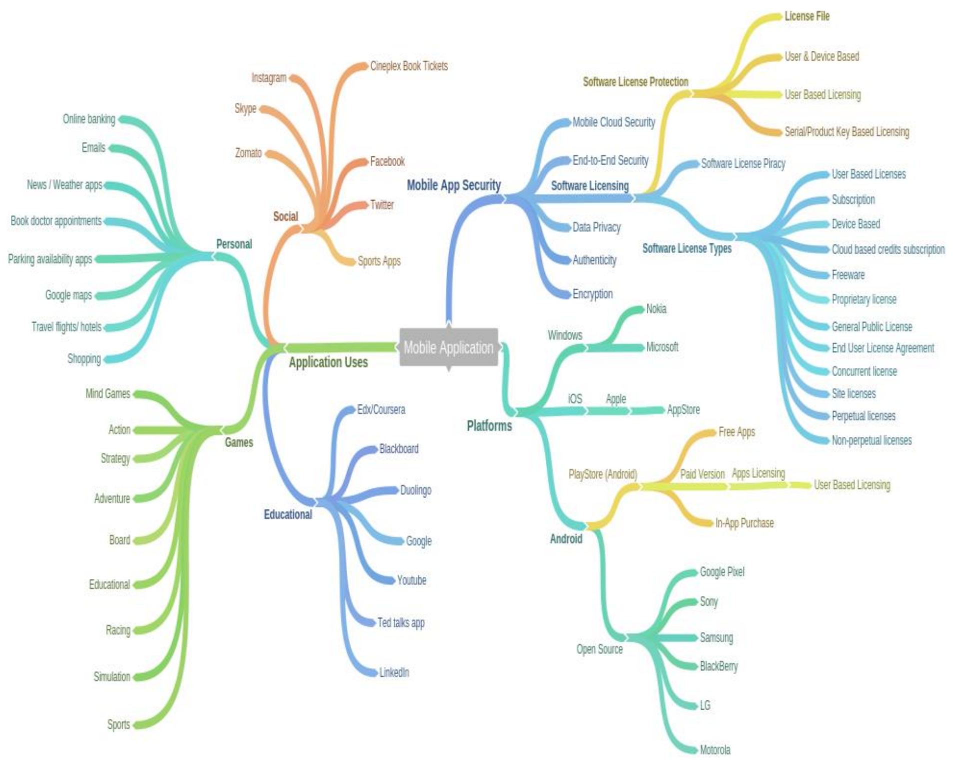


Figure 1. The Software Licensing Keywords Mind Map

In the outset of suitable licensing model for open source mobile applications this paper introduces an attempt to provide a declarative License File with every mobile application that would be used to create a specific user license and be stored at a reference storage to provide extra security to license validation.

3. Methodology of Licensing for Android

With the increase in growth of the application on mobile devices, providing license service in a way which can ease the scaling of all android. Thus, to achieve app scaling idea we have developed the Android Archive Library to provide mobile app licensing so that it can be easily incorporated to scale the existing application & for developing application.

3.1. AAR Library

AAR is Android Archive library whose structure is same as an Android app module. It can include everything needed to build an app, including source code, resource files, and an Android manifest. However, instead of compiling into an APK that runs on a device, an Android library compiles into an Android Archive (*.aar) file that you can use as a dependency for an Android app module. Unlike JAR files, AAR files can contain android resources and a manifest file, which allows you to bundle in shared resources like layouts and drawable in addition to Java classes and methods. This provide the flexibility to add different feature in AAR packages like licensing, analytics and application security.

3.2. Metadata

To automate the process of creating the license for user specific device, metadata required for library is getting the application context. License library must be developed independent of any application type, automating the process of getting the application context without asking from the developer was an important Meta data. The application context helps to read write user device storage for creating license file. Without this application context, licensing process could have not been automated. Apart from the context the other metadata need was device unique id, manufacturer, model, serial number were needed to create license file. All this metadata is generated by library and it doesn't have to be passed by third party application. The only parameter required from third party application is user email for which the license has to be generated.

3.3. License Integration

Figure 2 illustrates the process of integrating a software license with an Android application.

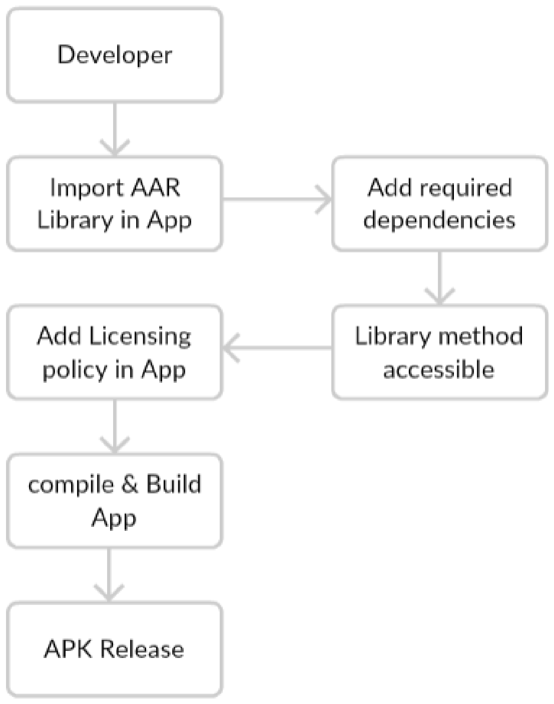

Developer
Import AAR Library in App
Add required dependencies
Library method accessible
Add Licensing policy in App
compile & Build App
APK Release

Figure 2. Flow Diagram to Integrate Library

As AAR license library is precompiled it can be imported as regular android library which keeps the code clean from the application. As shown in above figure-1 the basic steps to integrate are:

1. Import MALS AAR library in project
2. Add reference for imported AAR library in build file
3. Access license service from anywhere in projects

Thus, by following this basic steps mobile licensing services can be easily integrated in any application and developer can apply a flexible licensing policy on an application-by-application. Once the application is build it will generate a single APK file which then can be easily distributed on play store or any other app store.

4. The Licensing Prototype

This section describes about the system and process of overall mobile app licensing services combining with the distribution platform.

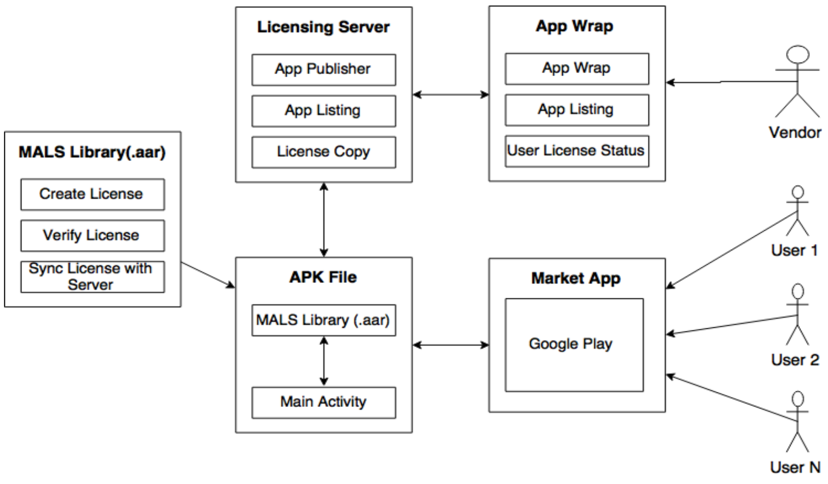


Figure 3. Mobile App Licensing Process

The Figure 3 describes the architecture of different component of the system and how they are connected to each other. App wrap is a native web application which is the initial platform for any vendors. Vendor need to register on with their app listing with unique application Id. After registration, user can download the library and include it in their application. This platform also provides admin panel for vendors to manage licenses and view the analytics based on the overall license generated.

Mobile application license service (MALS) library is a main module of the system which is responsible for creating license for all application. It is responsible to create license file depending on user device based with all the required metadata. The license file is stored on device internal directory of the device which is encrypted using AES algorithm. As license file is stored in internal data directory the file is hidden from user file explorer. This module is also responsible for storing the license data copy in the license server which is structured as per app listing by app publisher. This adds an additional security for license verification in case the license is misused in terms of duplicate license copy, license alteration and adding feature to manage license for device lost. The MAL library provide a declarative way to grant and generate licenses. Figure 4 illustrate the functionality of this library.

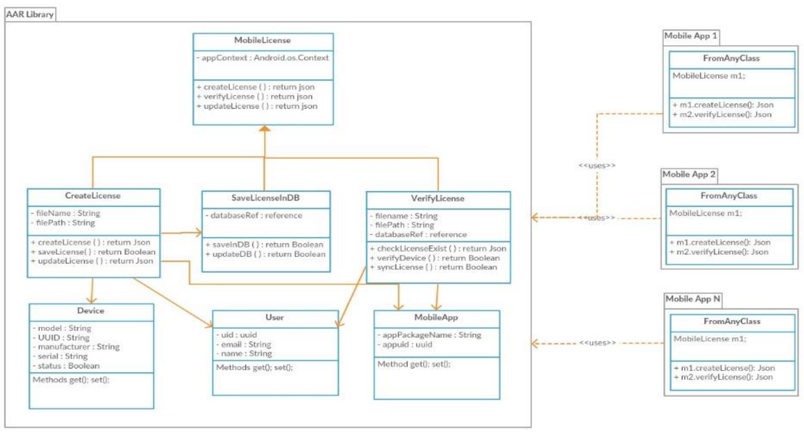

AAR Library
MobileLicense
- appContext : Android.os.Context
+ createLicense ( ) : return json
+ verifyLicense ( ) : return json
+ updateLicense ( ) : return json
CreateLicense
- fileName : String
- filePath : String
+ createLicense ( ) : return Json
+ saveLicense( ) : return Boolean
+ updateLicense ( ) : return Json
SaveLicenseInDB
- databaseRef : reference
+ saveInDB ( ) : return Boolean
+ updateDB ( ) : return Boolean
VerifyLicense
- filename : String
- filePath : String
- databaseRef : reference
+ checkLicenseExist ( ) : return Json
+ verifyDevice ( ) : return Boolean
+ syncLicense ( ) : return Boolean
Device
- model : String
- UUID : String
- manufacturer : String
- serial : String
- status : Boolean
Methods get(); set();
User
- uid : uuid
- email : String
- name : String
Methods get(); set();
MobileApp
- appPackageName : String
- appuid : uuid
Method get(); set();
Mobile App 1
FromAnyClass
MobileLicense m1;
+ m1.createLicense(): Json
+ m2.verifyLicense(): Json
<<uses>>
Mobile App 2
FromAnyClass
MobileLicense m1;
+ m1.createLicense(): Json
+ m2.verifyLicense(): Json
<<uses>>
Mobile App N
FromAnyClass
MobileLicense m1;
+ m1.createLicense(): Json
+ m2.verifyLicense(): Json
<<uses>>

Figure 4. The Mobile Application License Library Structure

The basic steps required to create a license using our prototype is quite simple (see Figure 5):

1. Import MAL AAR library in project
2. Add reference for imported AAR library in build file
3. Access license service from anywhere in projects

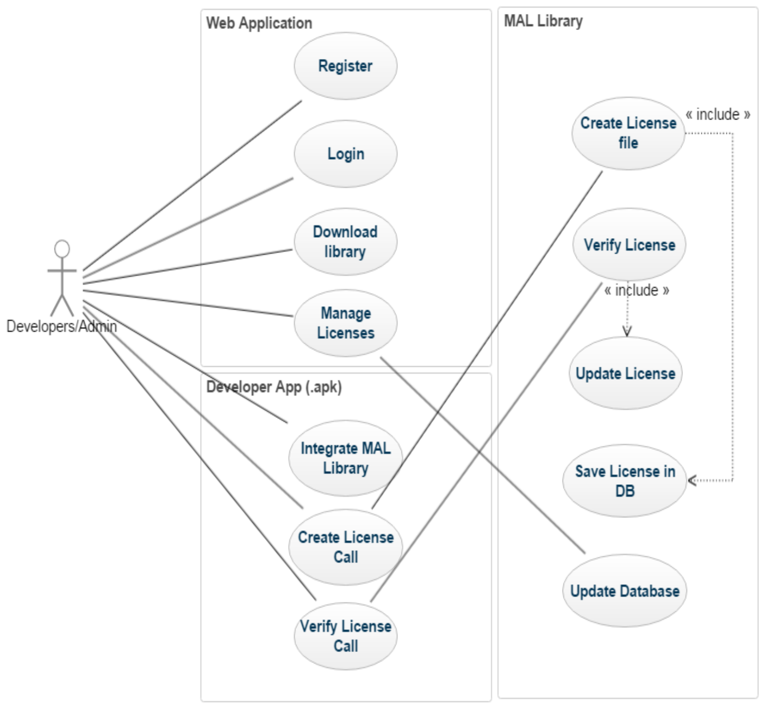


Figure 5. The Use Case Diagram for Creating Mobile App License

However, creating a light storage for licenses requires an admin role a light server presence. This can be enforced through our license server module that uses the firebase Google cloud server where all the license copies are maintained. Firebase is a mobile platform that helps you quickly develop high-quality apps, grow your user base, and is made up of complementary features that you can mix-and-match to fit your needs. Each feature works independently, and they work even better together. Cloud Platform services always encrypt customer content stored at rest, without any action required from the customer. For example, any new data stored in persistent disks is encrypted under the 256-bit Advanced Encryption Standard, and each encryption key is itself encrypted with a regularly rotated set of master keys. The same encryption and key management policies, cryptographic libraries, and root of trust used for your data in Google Cloud Platform are used by many of Google's production services, including Gmail and Google's own corporate data.

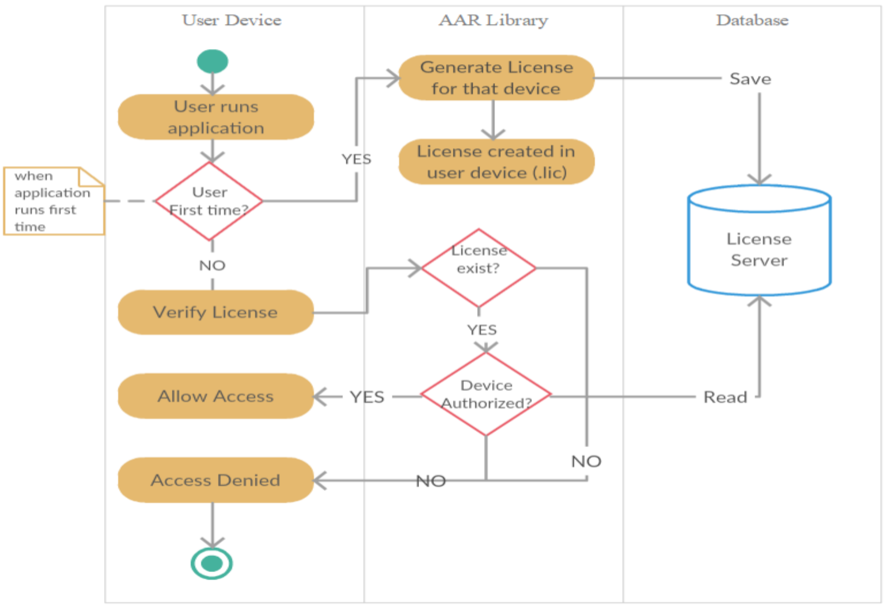
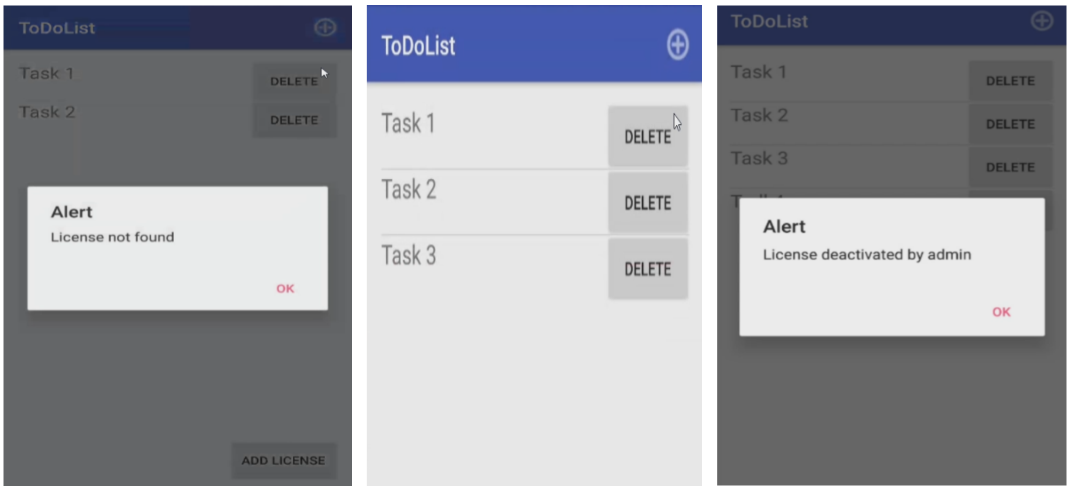


Figure 6. Activity Flow Diagram

APK file module shown here is a third-party vendor application who will be integrating the library in their application. It shows the license library integrated in the application and referenced as the part of their main activity to generate & verify license. Once the application is build it will generate an APK file with license library incorporated and which can be now published in play store so that all user can download and install the application in their device. After installation, the above figure 6 explain the use case flow of how the licensing service works.

5. Evaluation

The license library created was evaluated by integrating with two different application and the experimental results shows that licensing library can be easily integrated with any android applications in minutes without changing existing code and avoiding lengthy development efforts to secure mobile apps with fully productized app.



Alert
License not found

Alert
License deactivated by admin

Figure 7. (a) (b) (c)

First app was To-Do app, which is an example of an in-app purchase. The user can add only two to-do item in the list for free. If user wants to add more items, then user need to do the purchase because without purchase there wouldn't be any license file created. So, when user tries to add third item in the list it won't allow and alert user as shown in Figure 7(a). Now, assuming the user did the purchase and depending on the purchase response license can be generated for that user device. In this example, add license button was provided where user provide email and for that user device license was generated. Once the license is created user can add unlimited item in the list as shown in Figure 7(b). The admin can also manage the license and deactivate it whenever needed if any problem. So, when user try to add any more item it won't allow and alert with the message as shown in Figure 7(c).

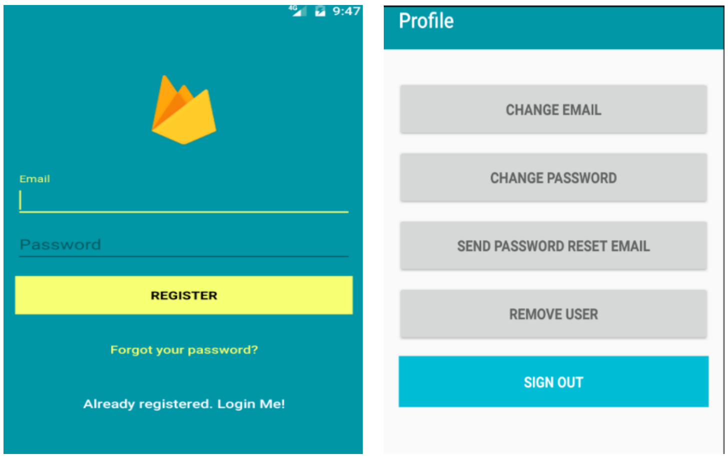
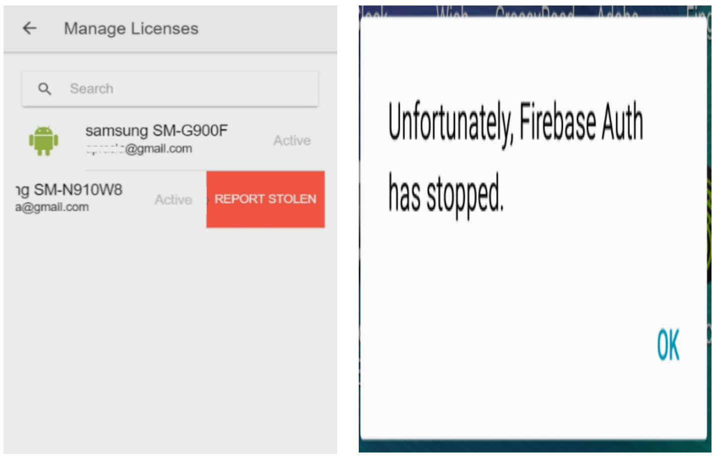


Figure 8. (a) (b)

Second app was an authentication app, the license was generated once after the user registers successfully from registration page as shown in Figure 8(a). After registration license file is created and the copy is also stored on license server. When user logins or access the application, the license is validated and allow user to access application as shown in Figure 8(b). License file is also synched with license server if any changes made by admin. So, as shown in Figure 9(a) if the user report to admin that the device was stolen then admin can revoke the access to that user device and the application on user device will be forced closed as shown in Figure 9(b) and license file will also be deleted protecting the user data from been misused.



Unfortunately, Firebase Auth has stopped.

Figure 9. (a) (b)

Below scenarios were achieved by evaluating on above two different applications:

License Piracy: The above created application was installed on different devices. User registered on one device and license for that device was created. When the user tries to access with same credentials on other device the application access was denied as there was no license file found on another unregistered device.

License Duplication: In continuation with above scenario user tries copying the license file in another device. So, when user try to get access to the application the access will be denied. As the license file is created for specific user device, the device unique won't match with the licenses data and it will stop user from accessing the application. This helps any vendor for solving license duplication problem.

Lost Device: If the user lost the device then it can contact the admin as admin as rights to activate and deactivate devices. Once the device license was reported as stolen the license file for that device will be deleted and resulting in revoking immediate access by forcefully closing the application. User also gets flexibility for adding new device by providing unique device id generated by app to the admin so that it generate license for new device and no more access to the stolen device.

6. Conclusion & Future Work

The emergence of cloud computing is changing how organization and individuals purchase and use software. This technology promises to streamline the on-demand provisioning of software, hardware, and data as a service, achieving economies of scale in IT solutions' deployment and operation. For enterprises that rely on software to maintain a market share, the software licensing model can strongly influence the return on software investment. Yet choosing a model can be daunting, involving considerations from total licenses purchased to the resources available to install and set up licensing systems. This more critical when we are dealing with mobile applications. There no clear effective model that can achieve flexible licensing without heavily relying on a backed server and carefully crafted copyright agreement. The only frameworks that we have seen at the time of developing our method is ether relying on the Google Play LAL library or uses recent development model like the Oracle Mobile Application Framework Mobile.[1] In this paper we have introduced an app license library that provide a declarative general license to each app and can be transferred to a dedicated software license according to the user and the mobile device uniquely. The declarative library was built for Android apps and is known as Android Archive Library (AAL) which can be included with any android app. We have evaluated the scalability of this library by trying to implement licensing on two different applications. The evaluation result was a clear evident that the mobile license is easily integrated without changing any app code. Our future research work will be focused on integrating the dedicated license with the use of the compiled APK file by incorporating some Delvik Virtual Machine[2] programming.

References

[1] Marc Ache, William Pence, and Napster Llc. Patent US20060272031 - system and method for unlimited licensing to a fixed number of devices, issued May 24, **(2005)**.

[2] Aaron Michael Burry, Peter Paul, Joel Eagle, and Xerox Corporation. Patent US20130204719 - systems and methods for license plate recognition to enable advance order pickup, issued February 2, (**2012).**

[1] http://www.oracle.com/technetwork/developer-tools/maf/overview/index.html

[2] https://source.android.com/devices/tech/dalvik/

[3] Jiayi Mu, Ailing Cui, and Jingyu Rao. Android Mobile Security – Threats and Protection. Proceedings of the International Conference on Computer, Networks and Communication Engineering (ICCNCE 2013), **(2013)**.

[4] Robert Willison, Mikko Siponen, "Software Piracy: Original Insights from a Criminological Perspective", Hawaii International Conference on System Sciences Proceedings of the 41st Annual, pp. 266-266, **(2008)**, ISSN 1530-1605.

[5] Diana Gabriela Noemí Benítez-Mejía, Gabriel Sánchez-Pérez, Linda Karina Toscano-Medina. Android applications and security breach, Digital Information Processing Data Mining and Wireless Communications (DIPDMWC) 2016 Third International Conference on, pp. 164-169, **(2016)**.

[6] Protsenko Mykola, Sebastien Kreuter, and Tilo Muller. Dynamic Self-Protection and Tamper proofing for Android Apps Using Native Code. 2015 10th International Conference on Availability, Reliability and Security, October **(2015)**. doi:10.1109/ares.2015.98.

[7] Jyun-Yao Huang, I-Hui Li, I-En Liao. A Software Licensing Authorization Scheme Based on Hardware Component Identifiers - IEEE Xplore Document. n.p., **(2017)**.

[8] Ons Mlouki, Foutse Khomh, Giuliano Antoniol. On the Detection of Licenses Violations in the Android Ecosystem - IEEE Xplore Document. n.p., **(2017)**.

[9] Jesse Obiri-Yeboah and Man Qi. Data Security of Android Applications - IEEE Xplore Document. n.p., **(2017)**.

[10] Nils Konferenzbeitrag, Yixiang Chen, Uwe Baumgarten, and Sejun Song. Securing License Verification by using Native Code, Fusing Options and Indirect Method Triggering on Android, International Symposium on Ambient Intelligence and Embedded Systems, **(2016)**. Available Online: http://amies.international-symposium.org/proceedings_2016/Kannengiesser_Chen_Baumgarten_Song_AmiEs_2016_Paper.pdf

[11] Li Karvell, Robert Donner, and Sanjay Garg. Patent US20150007340 A1- User based licensing for applications, issued January 1, (**2015**).

[12] Anthony George Myers, Thomas Louis Adrian, and Bmc Software. Patent US8646093 - method and system for configuration management database software license compliance, issued December 9, **(2009)**.

[13] Christopher Vendome, Mario Linares-Vásquez, Gabriele Bavota, Massimiliano Di Penta, Daniel M. German, Denys Poshyvanyk, "When and why developers adopt and change software licenses", Software Maintenance and Evolution (ICSME) 2015 IEEE International Conference on, pp. 31-40, (**2015**).

[14] William J Driscoll, William V Stevenson, Chad M Larsen, Driscoll William J, Stevenson William, and Larsen Chad M. Patent WO2004057580A2 - copy protected optical media storage device, along with methodologies for manufacturing and authenticating the same, issued December 17, **(2003)**.

[15] Neetesh Saxena, Narendra S. Chaudhari, "EasySMS: A Protocol for End-to-End Secure Transmission of SMS", Information Forensics and Security IEEE Transactions on, vol. 9, pp. 1157-1168, (**2014**), ISSN 1556-6013.

[16] Ladislav Sobr, Petr Tuma. SOFAnet: Middleware for Software Distribution over Internet - IEEE Xplore Document. n.p., **(2017)**.

[17] Jesse Obiri-Yeboah and Man Qi. Data Security of Android Applications - IEEE Xplore Document. n.p., **(2017)**.

[18] Jyun-Yao Huang, I-Hui Li, I-En Liao, "A software licensing authorization scheme based on hardware component identifiers", Information Science Electronics and Electrical Engineering (ISEEE) 2014 International Conference on, vol. 3, pp. 1673-1676, **(2014)**.

[19] Yanwu Yang, Fen Xia, Wensheng Zhang, Xian Xiao, Yiqun Li, Xuhui Li, "Towards Semantic Requirement Engineering", Semantic Computing and Systems (**2008**). WSCS '08. IEEE International Workshop on, pp. 67-71, 2008.

[20] Mohab Usama, Mohamed Sobh, "Software Copy Protection and Licensing based on XrML and PKCS#11", Communications Computers and Signal Processing (PacRim) **(2011)** IEEE Pacific Rim Conference on, pp. 856-861, 2011, ISSN 2154-5952.

[21] Daniel Ferrante. "Software licensing models: what's out there?." IT Professional 8.6 (2006).

[22] Pradyumna K. Misra, Bradley J. Graziadio, and Terence R. Spies. "System and method for software licensing." U.S. Patent No. 7,171,662. 30 Jan. 2007.

[23] Pradyumna K. Misra, Bradley J. Graziadio, and Terence R. Spies. "System and method for software licensing." U.S. Patent No. 7,171,662. 30 Jan. 2007.

[24] Shiliang Wu, Hans Wortmann, Chee-wee Tan, "A pricing framework for software-as-a-service", Innovative Computing Technology (INTECH) 2014 Fourth International Conference on, pp. 152-157, 2014.

[25] Ming-Wei Wu, and Ying-Dar Lin. "Open source software development: an overview." Computer 34.6 (2001): 33-38.

[26] Andrew M. St. Laurent, Understanding open source and free software licensing: guide to navigating licensing issues in existing & new software. "O'Reilly Media, Inc.", 2004.

Authors

Anis Prasla, He is pursuing MSc. Computer Science at Lakehead University, Thunder Bay, ON, Canada. He received is B.E. degree in 2010 from Department of Computer Engineering at Rajiv Gandhi Institute of Technology affiliated to University of Mumbai. His current research includes mobile application licensing services to scale all applications.

Sabah Mohammed, He received the M.S. degree in 1981 from Department of Computing of Glasgow University, UK and Ph.D. degree in 1986 from the Department of Computer Science of Brunel University, UK. Since 2001, Dr. Mohammed is a Full Professor with the Department of Computer Science at Lakehead University. Dr. Mohammed is also an adjunct Professor with the University of Western Ontario. His current research interests include Security of Health Data, Web Intelligence, Machine Learning, Data Science, Cloud Computing, Social Networking and Enterprise Mobility.

Jinan Fiaidhi, she is a full Professor and the Graduate Coordinator with the Department of Computer Science, Lakehead University, Ontario, Canada since 2001. She is also an Adjunct Research Professor with the University of Western Ontario. She received her graduate degrees in Computer Science from Essex University (PgD 1983) and Brunel University (PhD, 1986). Dr. Fiaidhi research is focused on mobile and ubiquitous learning utilizing the emerging technologies (e.g. Deep Learning, Calm Computing, Learning Analytics, Mobile Learning, Personal Learning Environment, Social Networking, Enterprise Mashups, Big Data and Semantic Web).