HYBRID ENCRYPTION FOR SECURING SHARED PREFERENCES OF ANDROID APPLICATION

K.Venkateswarlu, Associate Professor Dept.of Master of Computer Applications, Narayana Engineering College(Autonomous), Gudur.SPSR Nellore, AP, India
VK.Mohith Kumar, Research Scholar, Dept.of Master of Computer Applications, Narayana Engineering College(Autonomous), Gudur.SPSR Nellore, AP, India

ABSTRACT:

Most mobile applications generate local data on internal memory with Shared Preference interface of an Android operating system. Therefore, many possible loopholes can access the confidential information such as passwords. We propose a Hybrid Encryption approach for Shared Preferences to protect the leaking confidential information through the source code.We apply Hybrid Encryption approach combining encryption approach with Android Keystore system, for providing better encryption algorithm to hide sensitive data.

Android's Shared Preferences interface provides a general framework that allows us to access and modify key-value pairs of primitive data types. This data persists across user sessions, even if the application is closed. By default, Android stores this data in an unencrypted XML file within the app's directory on the device's file system, with permissions that allow only the app to access this file.

Tools such as Android Debug Bridge (ADB) can be used to navigate to the directory where Shared Preferences are created.

KEY WORDS: Scan image, Scan Hand Written Data

INTRODUCTION

The Android working framework is an open source and source code discharge by Google under Apache permit license, based on Linux-Kernel designed for smartphones and tablets. Android is one of the most popular operating systems for smartphones [1,2]. Designed to be a complete software stack, Android includes an operating system, middleware, and core applications. Furthermore, it comes with an SDK that provides the tools and APIs necessary to develop new applications for the platform in Java. Android does not distinguish between its core applications and new applications developed with the SDK; in particular, all applications can potentially interact with the underlying mobile device and share their functionality with other applications. Device loss is a pervasive problem with mobile devices and leads to severe attacks. Android KeyStore System minimizes drawback of encryption approach but still leak the data on the device. For securing data on the device, we propose the Hybrid Encryption Approach with case studies.

STATEMENT OF THE PROBLEM:

The problem lies with existing system is Android's SharedPreferences interface provides a general framework that allows us to access and modify key-value pairs of primitive data types. This data persists across user sessions, even if the application is closed which makes it vulnerable to access by the attackers**.**

OBJECTIVE OF STUDY:

✓ Most mobile applications generate local data on internal memory with SharedPreference interface of an Android operating system.

✓ Therefore, many possible loopholes can access the confidential information such as passwords. I propose a Hybrid Encryption approach for SharedPreferences to protect the leaking confidential information through the source code.

✓ I apply Hybrid Encryption approach combining encryption approach with Android Key store system for providing better encryption algorithm to hide sensitive data.

Copyright @ 2021 Authors

REVIEW OF LITERATURE:

Android allows create and store data within the application. This section describes Shared Preference interface, possible data security vulnerabilities with the help of test application. We discuss encryption approach taken to minimize the vulnerability and its drawback.

Android's Shared Preferences [3] interface provides a general framework that allows us to access and modify key-value pairs of primitive data types. This data persists across user sessions, even if the application is closed. By default, Android stores this data in an unencrypted XML file within the app's directory on the device's filesystem, with permissions that allow only the app to access this file. This is part of the concept known as "application sandboxing." In Android, shared preferences are used to store user's preferences for Android application such as display name, notification settings, vibration on/off, etc.

Developers can use Shared Preferences to store data on a device, and an attacker can access this data from a device as well. The need of protecting it is of much importance. Tools such as Android Debug Bridge (ADB) [4] can be used to navigate to the directory where SharedPreferences are created. ADB - Android Debug Bridge (ADB) is a versatile commandline tool that lets you communicate with a device. The ADB command facilitates a variety of device actions, such as installing and debugging apps, and it provides access to a Unix shell that you can use to run a variety of commands on a device.

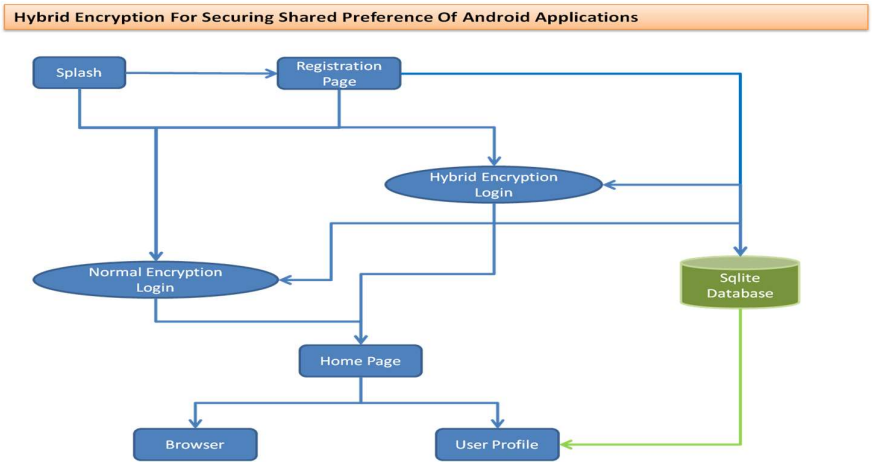
ADB is included in the Android SDK Platform-Tools package. An attacker can download this package with the SDK Manager, which installs it at location android_sdk/platform-tools/. As it provides access to Unix shell of Android device, it helps an attacker to navigate to data directory and read or modify any unencrypted data including SharedPreferences

RESEARCH METHADOLOGY:

System Analysis is first stage according to System Development Life Cycle model. This System Analysis is a process that starts with the analyst.Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. One aspect of analysis is defining the boundaries of the system and determining whether or not a candidate system should consider other related systems. During analysis, data are collected on the available files, decision points, and transactions handled by the present system.

Logical system models and tools that are used in analysis. Training, experience, and common sense are required for collection of the information needed to do the analysis.

ARCHITECTURE:



MODULES:

- ✓ User Module
- ✓ Encryption Module
- ✓ Android Keystore Module
- ✓ Decryption Module

Copyright @ 2021 Authors

DESCRIPTION:

User

In this module, User can register and login to the system.Here, he can view the profile information and view the shared preference of this an android app.He can show that how the shared preferences are created in the android application.In the proposed system the user performing two types of logins, one normal login and second one is hybrid login.

Encryption Module

In this module, user can perform the hybrid login the system automatically encrypt the login information before storing in shared preferences.The proposed method uses symmetric encryption algorithm to encrypt the login information and store the private key in android keystore.

Android Keystore Module

In this module, the encryption key is stored in the android key store.
Android key store is root memory of android operating system which cannot accessed by external applications or external users.This is secure to store the data.

Decryption Module

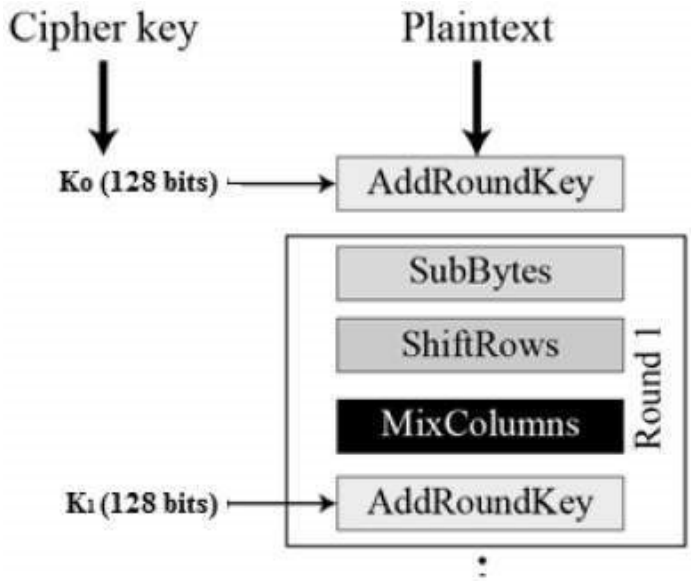
In this module, when user performs login the app access the android key store and decrypt the data and verifies the data.This module is very efficient and it cannot be leak any data to third party applications.

RESULTS

The result analysis describes that the entire project was executed successfully and also having quality and performance by analyzing the flow of data and output screens. In my project the modules like User, Encryption, android keystroke and Decryption are independent modules. Because my project follows the top down approach and bottom up approach.

AES Algorithm

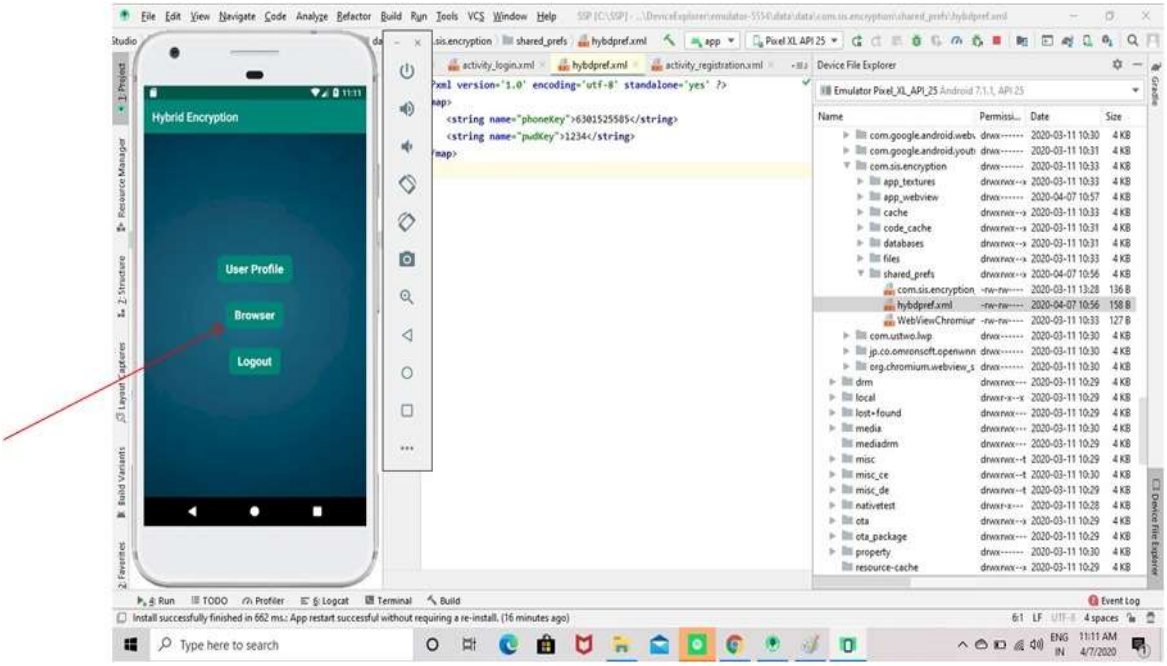
In this project to protect the video data we adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES).AES is an iterative rather than Feistel cipher. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).



Copyright @ 2021 Authors

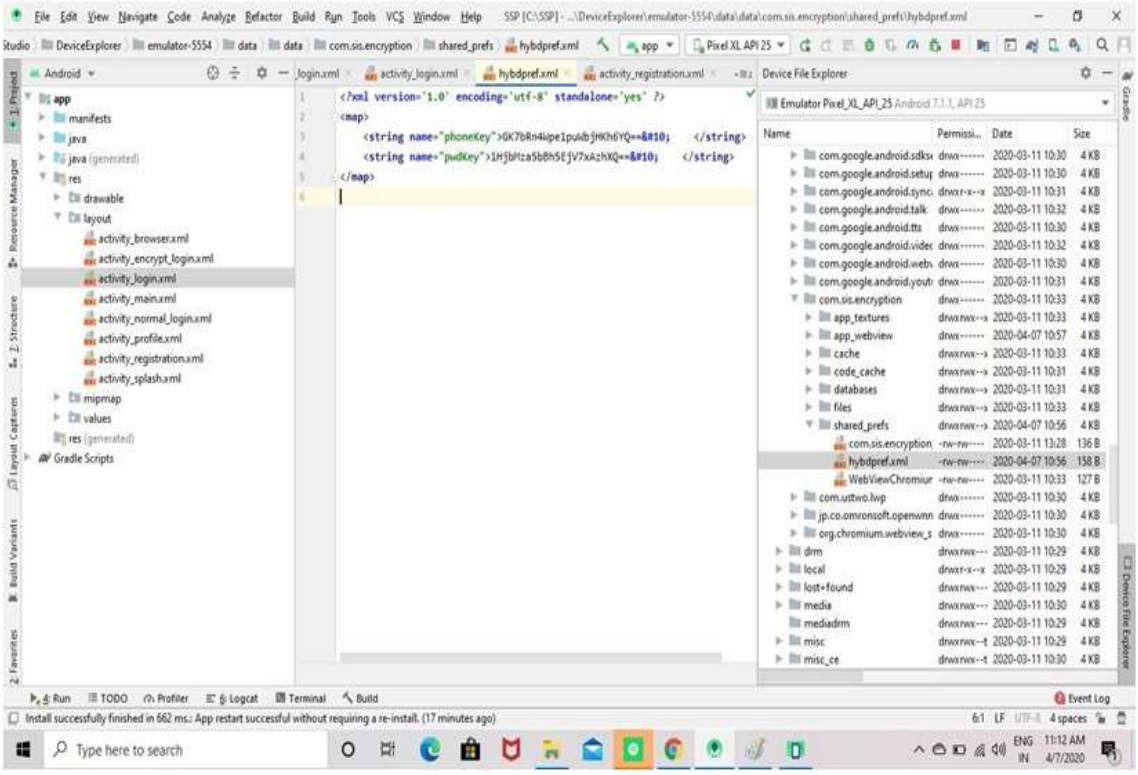
OUTPUT SCREENS

SCREEN



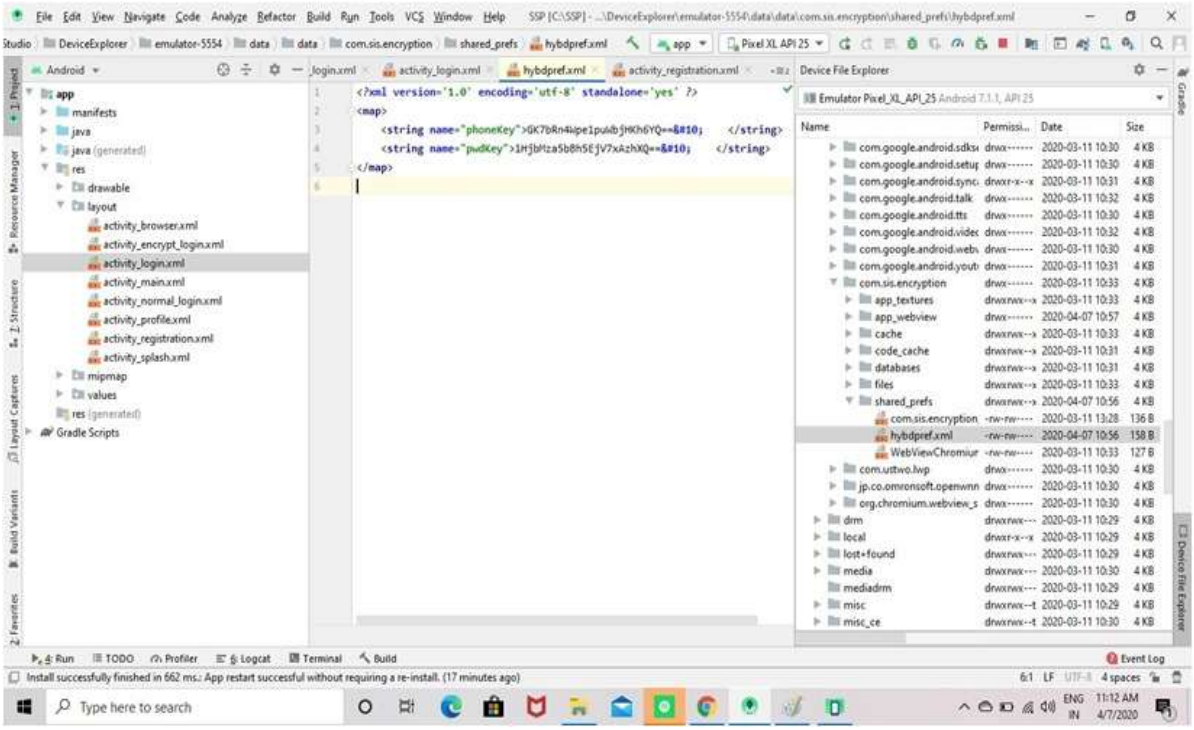
1

SCREEN 2

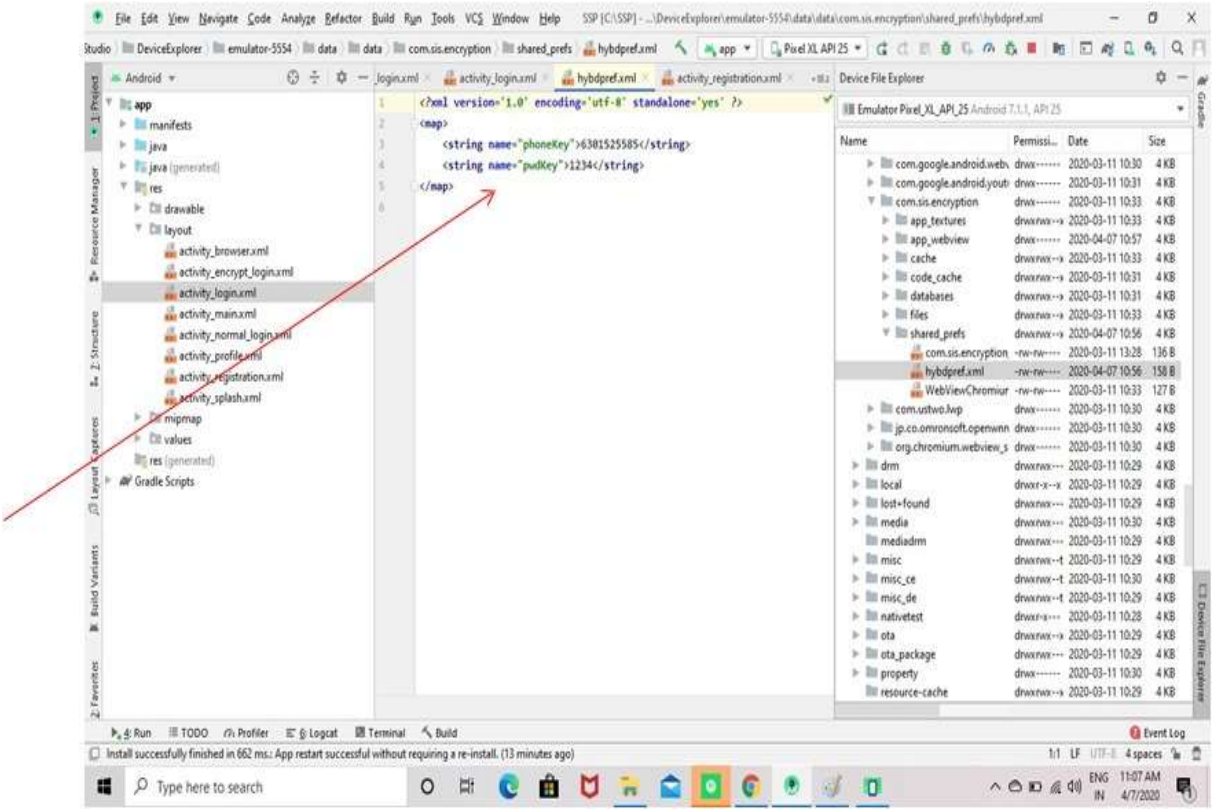


Copyright @ 2021 Authors

SCREEN 3

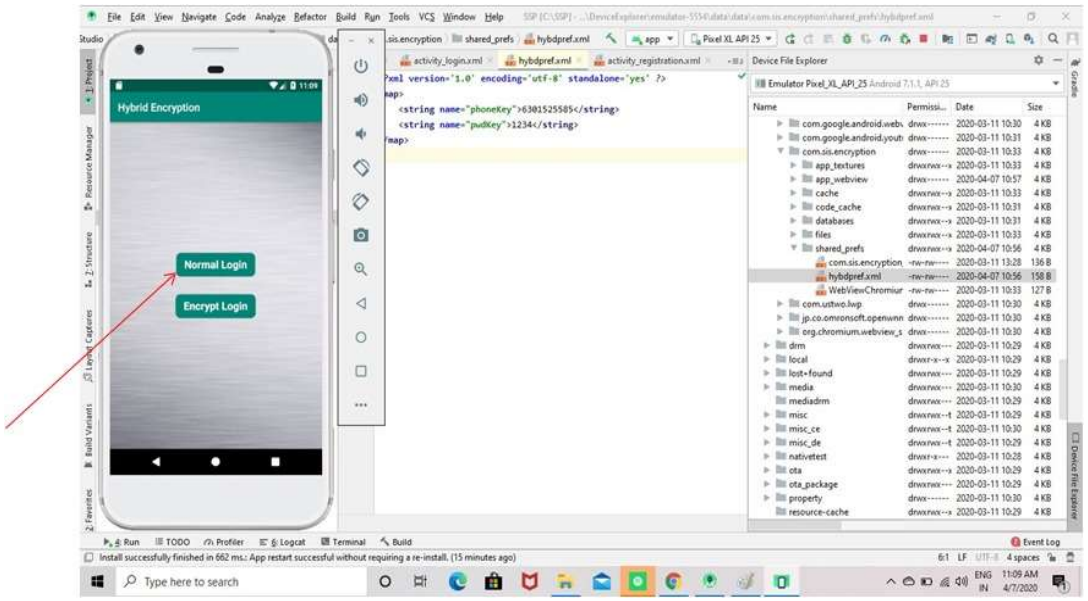


SCREEN 4

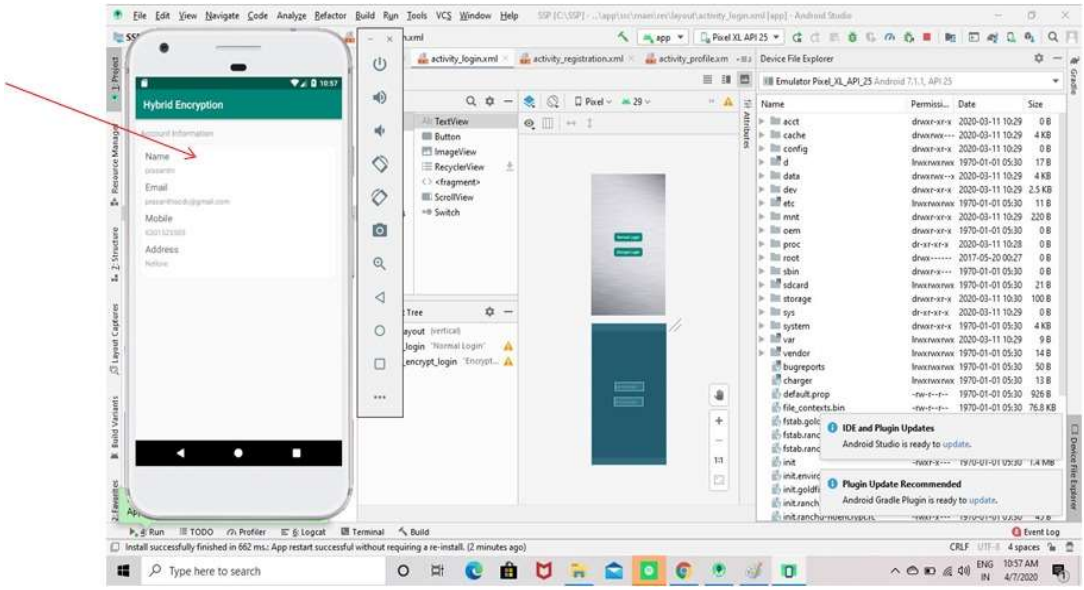


Copyright @ 2021 Authors

SCREEN 5



SCREEN 6



CONCLUSION:

This project discussed the security leaks of an Android application due to XML files generated within the internal memory of application. This paper demonstrated the leak of the encrypted confidential data on internal memory with SharedPreference interface. We proposed a Hybrid Encryption approach for SharedPreferences combining encryption approach with Android Keystore system to protect the leaking confidential information from the Android device. The test results indicated that proposed Hybrid Encryption approach enables to secure local data on the device.

REFERENCES:

1. https://www.tutorialspoint.com/android/android_resources.htm
2. https://developer.android.com/guide/index.html
3. https://www.engineersgarage.com/articles/what-is-android-introduction
4. http://www.beginandroid.com/intro.shtml

Copyright @ 2021 Authors

5. http://www.gcflearnfree.org/androidbasics/intro-to-android-devices/1/
6. https://en.wikipedia.org/wiki/Android

REFERENCES

[1] "Number of Google play store apps 2016 statistic," Statista, 2014. [Online]. Available: http://www.statista.com/statistics/266210/numberofavailable-applications-in-the-google-play-store.
[2] "Smartphone users worldwide 2014-2020 statistic," Statista, 2016. [Online]. Available: https://www.statista.com/statistics/330695/numberofsmartphone-users-worldwide.
[3] "SharedPreference" [Online] Available: https://developer.android.com/reference/android/content/SharedPreferences.html
[4] "ADB" [Online] Available: https://developer.android.com/studio/command-line/adb.html
[5] "CheatDroid application" [Online] Available: https://play.google.com/store/apps/details?id=com.felixheller.sharedprefseditor&hl=en
[6] "Android Keystore System" [Online] Available: https://developer.android.com/training/articles/keystore.html#SecurityFeatures
[7] Suchita Tayde, Seema Siledar, "File Encryption, Decryption Using" 2015 International Journal of Advanced Research in Computer Science and Software Engineering. [Online] Available: https://www.researchgate.net/publication/315669325_File_Encryption_Decryption_Using_AES_Algorithm_in_Android_Phone
[8] Rohan Rayarikar, Sanket Upadhyay, Priyanka Pimpale, "SMS Encryption using AES Algorithm on Android" 2012 International Journal of Computer Applications [Online] Available: http://www.ijcaonline.org/archives/volume50/number19/7909-1038
[9] Tim Cooijmans, Joeri de Ruiter, Erik Poll, "Analysis of Secure Key Storage Solutions on Android."2014 Proceeding SPSM '14 Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices [Online] Available: https://dl.acm.org/citation.cfm?id=2666627
[10] Poonguzhali P., Prajyot Dhanokar, M. K. Chaithanya, and Mahesh U. Patil, "Secure Storage of Data on Android Based Devices." 2016 IACSIT International Journal of Engineering and Technology, Vol. 8, No. 3 [Online] Available: http://www.ijetch.org/vol8/880-ST011.pdf8 m
[11] Penchalaiah, P., & Rajasekar, P. An Efficient Multi-User Hierarchical Authenticated Encryption Using Simultaneous Congruence for Highly Secure Data. International Journal of Future Generation Communication and Networking (WoS), ISSN, 2233-7857.

Copyright @ 2021 Authors