Study and Application of Transparent File Encryption Technology on Android Platform

Yongzhong Li [a], Lei Gu [b] and Yi Li [c]

School of computer science, Jiangsu University of Science and Technology, Zhenjiang 212003, China

[a]liyongzhong61@163.com, [b]870288285@qq.com, [c]crush_lee121@163.com

Keywords: Transparent file encryption; File encryption; Android platform.

Abstract: Aiming at the data security problem of Android platform, a transparent encryption system based on file filter driver is designed and implemented, according to the technology of file transparent encryption and decryption system based on hook transparent encryption technology and file filtering driven transparent encryption technology used on windows platform. This system is different from the traditional APP development method of Android system. By intercepting the system call function and using the secret-key converted from the host MAC address, the encryption and decryption algorithm is written into the kernel, which fundamentally guarantees the security of user information. At the same time, the user's security experience is improved by putting authentication on the screen unlocking. The system design and implementation are described in this paper from system requirement analysis to overall design and detailed design of each module. Android application development technology and cross-compiling principle are used in the coding process. The system test results show that the system can effectively transparently encrypt files and protect the privacy of mobile files.

1. Introduction

At present, file transparent encryption technology has become increasingly mature. However, it is mostly used in Windows platform, and the application market for Android mobile phone file encryption software is uneven, and users are required to enter passwords to verify every time they encrypt and decrypt files, which greatly reduce the encryption efficiency and user experience. A transparent encryption system based on Android file filter driver is designed in this paper. In the kernel layer, the encryption and decryption algorithm are written into the kernel by intercepting system calls, to improve user experience and encryption efficiency. The system's authentication is placed in the screen lock.

2. Android System Architecture

Android system architecture is based on the Linux kernel and is bottom-up structure. It is mainly divided into four layers [1], as shown in Figure 1, the Linux Kernel layer, the Library layer, the Application Framework layer and the Application layer. The Linux kernel layer provides the underlying drivers for hardware of Android devices. The system runtime layer provides the main features support for Android system through some C/C++ libraries. The application framework layer provides various APIs that may be used to build applications. The application layer includes all applications installed on mobile phones [2]. First-order subheads should be in bold caps with 2-line spaces above and 1-line space below them. Second-order subheads should be in bold with main words capitalized with 1-line space above and below them. Third-order subheads should be regular type in all caps with 1-line space above and below them. Subheads should not appear alone at the bottom of a page.

Published by CSP © 2019 the Authors

3. Principle of Transparent Encryption Technology

Transparent encryption refers to the process of encrypting and decrypting files without changing the user's operating habits. It is a passive compulsory encryption technology [3], which is insensitive to users. When the user opens or edits the specified file, the system will automatically encrypt the unencrypted file and decrypt the encrypted file. Encrypted files leave the current usage environment, which cannot automatically decrypt and protect the contents of files.

Transparent encryption technology can be divided into user-mode implementation and kernel-mode implementation according to the location of implementation. According to encryption efficiency, hook encryption technology encrypts the whole file in the application layer, and encrypts and decrypts the file relatively slowly. Driving transparency technology encrypts and decrypts the file dynamically in the driver layer, which has high efficiency. So, file filter-driven transparent encryption is used in this paper.

File Driver Encryption (IFS) technology is based on Windows File System Filter Driver (IFS) technology [4], which works in the kernel layer of Windows. Without affecting the upper and lower interfaces, it can intercept all file system requests, so that new functions can be added without modifying the upper software or the lower driver, as shown in Figure 2 [5]. It is characterized by high encryption efficiency and security, but the technical threshold is high. It is necessary to understand the Windows system kernel in depth and difficult to develop. All tables and figures with text only should be boxed in; i.e., a box should be drawn around the table or figure either by hand with a ruler or with a draw facility on.

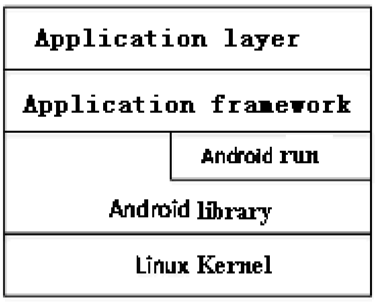


Fig. 1 Android system architecture

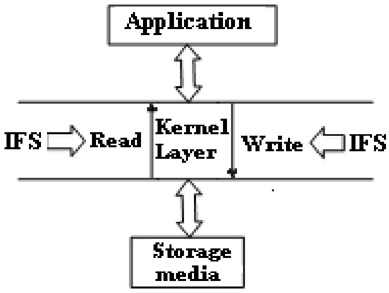


Fig. 2 Drives Transparent Encryption

4. Design and Implementation of Transparent File Encryption System

4.1 Overall Design

The system frame design of the whole system is shown in Figure 3.

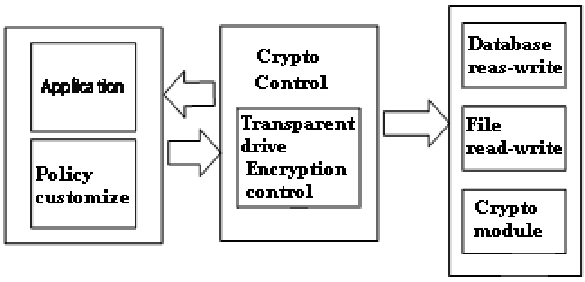


Fig.3 System Overall Design Framework

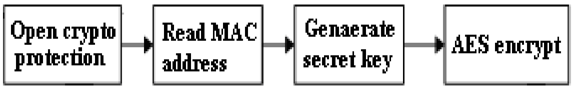


Fig. 4 Workflow of Encryption Module

The system uses MVP (Model, View, Presenter) framework. Model (model) receives the control information from the controller, completes the operation of reading and writing files and encryption and decryption. View (user interface) mainly realizes the interaction with users and updates the user's encryption policy customization to relevant database items. Presenter is responsible for logical processing, customizing and updating the monitoring list according to the user's encryption strategy, monitoring and accepting the data read and write operations applied in the list, and passing the information to Model. MVP is evolved from MVC framework [6]. It cuts off the connection between View and Model, makes View interact only with Presenter, increases readability and reusability, and reduces the cost of later testing and maintenance [7].

4.2 Design and Implementation of Encryption and Decryption Module

The performance of encryption module affects the security of transparent encrypted file system [8]. The encryption module of the system uses symmetrical encryption AES algorithm to protect file data. AES is called Advanced Encryption Standard, which is the advanced encryption standard. AES algorithm requires 128 bits or 16 bytes of plaintext packet length, and the key length can be divided into 128 bits, 192 bits or 256 bits (16, 24 or 32 bytes) [9]. The AES encryption process involves four operations: byte substitution, row shift, column mixing and extended key exclusive or. The decryption process is corresponding inverse operation [10]. Figure 4 shows the workflow diagram of the encryption module. By reading the MAC address of Android terminal, after a series of substitution transformations and other operations, it is transferred to the encryption algorithm of the kernel module as the key of the encryption algorithm of the current device. Among them, the access to the MAC address of the Android terminal needs to read the address under / sys / class / net / wlan0. Therefore, each terminal has its own unique key. Replacing the terminal will not be able to view the files of the local terminal, which plays a role in protecting the privacy of mobile files.

4.3 Design and Implementation of the Whole System

The whole system design module is divided into application layer module and kernel module. The application layer module mainly completes the function of customizing encryption strategy and interacting with users; the kernel module completes the functions of monitoring, encryption and decryption, data reading and writing according to the setting of application module.

The overall design flow chart of the system is shown in Figure 5. After the system starts to run, the user carries out the "policy customization" operation at the user level, enters the kernel layer after the policy formulation, and monitors the reading and writing operations of the files. In order to read a file, the first step is to determine whether the file is an open encrypted protected file. If it decrypts and passes the data to the user; if it is to write a file, it is still necessary to determine whether the file is an open encrypted file, and if it is encrypted and writes the data to the database or SD card. If the read-write operation file is not the file protected by the policy, then the normal read-write operation can be carried out.

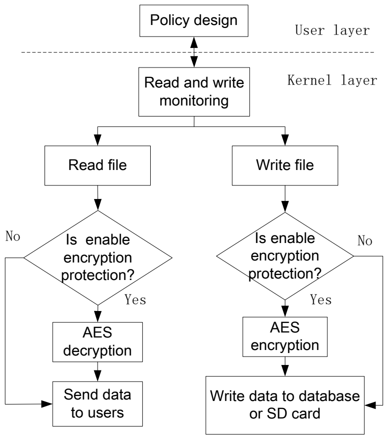

Policy design
User layer
Kernel layer
Read and write monitoring
Read file
Write file
No
Is enable encryption protection?
Yes
Is enable encryption protection?
No
Yes
AES decryption
AES encryption
Send data to users
Write data to database or SD card

Fig. 5 The overall design flow of the system

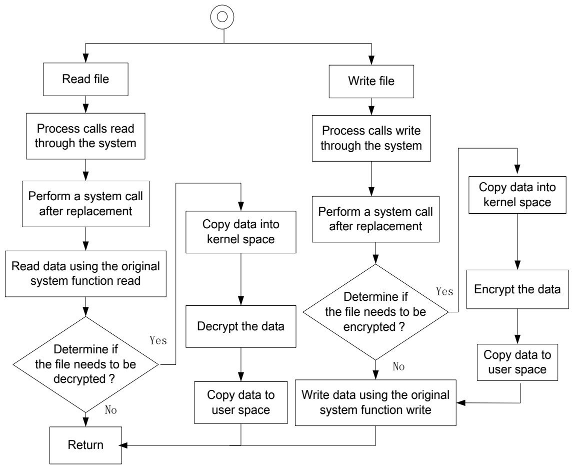

Read file
Write file
Process calls read through the system
Process calls write through the system
Perform a system call after replacement
Copy data into kernel space
Perform a system call after replacement
Copy data into kernel space
Read data using the original system function read
Yes
Determine if the file needs to be encrypted ?
Encrypt the data
Decrypt the data
Determine if the file needs to be decrypted ?
Yes
No
Copy data to user space
No
Copy data to user space
Write data using the original system function write
Return

Fig. 6 Design flow of kernel module

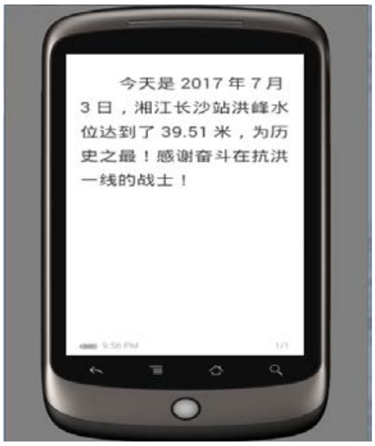
4.4 Design and Implementation of Kernel Module

System calls under Linux are implemented with soft interrupts. The interrupt program handles different system calls according to the system call number. Through the soft interrupt program, the program will be trapped in the kernel space for system call processing. In addition, Linux provides a program that can load kernel modules, namely LKM (Loadable Kernel Module), which is mainly used to dynamically extend the functions of the Linux kernel [11]. Figure 6 shows the workflow diagram of the kernel module.

When run the process of writing files, the encryption process is executed, the interception system calls write. The kernel module gets the file name and structure. Comparing with the file name and file structure in the encryption process formulated by the application layer, if the file name and file structure match, the data will be copied to the kernel space, and AES symmetric encryption operation is performed on the data through the pre-written encryption function.

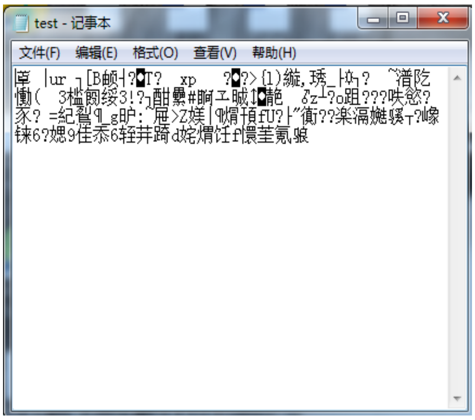
5. System Testing

The system is installed on API18 simulator and successfully implements the transparent encryption function of txt and doc file format on SD card. After the encryption is successful, the files can be viewed normally at the local terminal. Replacing the terminal and viewing it on the PC and another mobile phone is random code, thus completing the privacy protection of Android mobile phone files, as shown in Figures 7-12.



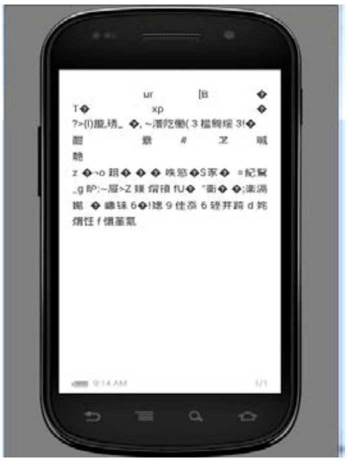
今天是 2017 年 7 月 3 日，湘江长沙站洪峰水位达到了 39.51 米，为历史之最！感谢奋斗在抗洪一线的战士！

Fig.7 Views the encrypted TXT



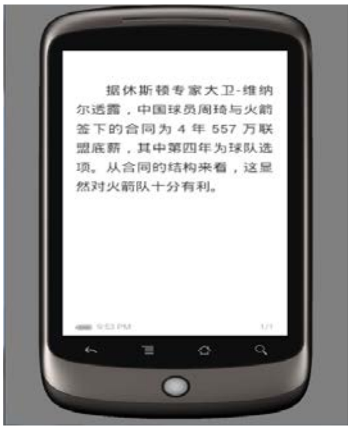
```
嶂 |ur ┐[B峋┤?�T?  xp   ?�?>{1)緃,琇_├办┐?  ~潸陀
慟(  3檻阌绥3!?┐酣纍#眴ㄒ眡↕�靘  ǒz┴?o跙???呋慾?
豕? =紀翟¶_g肸:~屉>Z媄|ฯ熁禎fU?┝"衜??楽滆媸骙┬?嚎
铼6?媤9佳忝6轻茾蹄d姹熁饪f愢董氪骕
```

Fig. 8 Views the encrypted TXT file on the PC



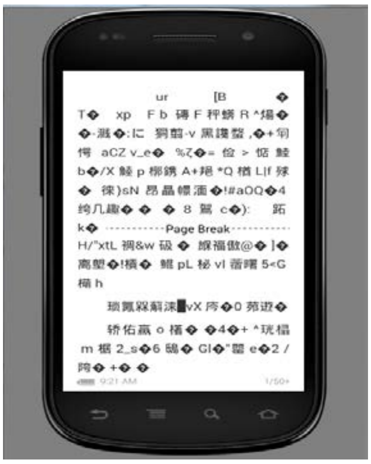
```
            ur       [B       �
T�          xp                �
?>(l)龇,琇_ �, ~潸陀慟( 3 檻阌绥 3!�
酣          纍     #    ㄒ    眡
靘
z �¬o 跙� � � 呋慾�S豕�  =紀翟
_g 肸:~屉>Z 媄 熁禎 fU�  "衜� �;楽滆
媸  � 骙铼 6�!媤 9 佳忝 6 轻茾蹄 d 姹
熁饪 f 愢董氪
```

Fig. 9 Views the encrypted TXT file on another terminal



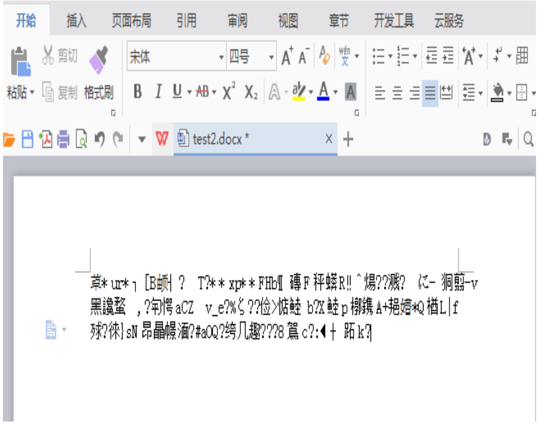
据休斯顿专家大卫-维纳尔透露，中国球员周琦与火箭签下的合同为 4 年 557 万联盟底薪，其中第四年为球队选项。从合同的结构来看，这显然对火箭队十分有利。

Fig. 10 Views the encrypted doc file on the local terminal



ur [B �
T� xp F b �castle F 枰蛴 R ^煬�
�-濈�:に 狪翦-v 黑譢蜜 ,�+旬
愕 aCZ v_e� %ζ�= 俭 > 惦 鲑
b�/X 鲑 p 榔鎸 A+絶 *Q 楯 L|f 殏
� 徕}sN 昂晶幉湎 �!#aOQ�4
绔几趣� � � 8 篇 c�): 跖
k� ------------Page Break------------
H/"xtL 裥&w 砐 � 娓福做@�]�
离塑�!楨� 鲱 pL 秘 vl 蔷曙 5<G
楠 h
琐氮槑蔛涑█vX 庤�0 蒣迣�
轿佑藏 o 楯� �4�+ ^珖榀
m 楣 2_s�6 鷊� GI�"噐 e�2 /
骻� +� �

Fig. 11 Views encrypted doc files on PC



茸* ur* ┐ [B蛕| ? T?* * xp* * FHb¶ 磚F 秤蟒R‼ ^煬??溅? に- 洞翦-v
黑譢鳌 ,?旬愕 aCZ v_e?%ζ??俭>惦鲑 b?X 鲑 p 榔鎸 A+絶嬉*Q 楯L|f
殏?徕}sN 昂晶幉湎?#aOQ?绔几趣???8 篇 c?:◀┼ 跖 k?|

Fig.12 Views encrypted doc files on another terminal

6. Conclusion

According to the test results, when the encryption is completed, the files can be viewed normally on the local mobile phone after the authentication of screen lock, but not in other environments. The transparent encryption function of files under Android platform has been successfully implemented. The system uses file filtering to drive transparent encryption technology. By intercepting system calls and using keys converted from MAC address of host computer, the encryption and decryption algorithm is written into the kernel. In the process of encryption, the plaintext of files only appears in the kernel layer, which has the characteristics of security, stability and efficiency. However, the software user interface for this system can also be beautified, and the type of file data for encryption protection can also be increased, which will become the next research content.

Acknowledgments

This paper is supported by Jiangsu postgraduate research innovation project (SJCX18_0784). and Research fund of University of Jiangsu Province and Jiangsu University of Science and Technology’s Basic Research Development Program (No. 2005DX006J).

References

[1] HUANG Ji-hua. Research and Application of Android System Architecture[J]. Electronic Technology & Software Engineering,2016(07):49.

[2] GUO Lin. The First Line of Code-Android[M]. Second Edition. Beijing: The People's Posts and Telecommunications Press,2016.

[3] YANG Di, YE Peng, FANG Zhen-lin. The Application of Transparent Decryption in Trusted Storage of Electronic Documents[J]. Electronic Science and Technology,2017,04(04):147-150.

[4] SUN Xiao-yu, YANG Tao, HU Xiao-qin. A File System Protection Scheme Based on File Filter Driver and TrueCrypt[J]. Modern Computer(professional),2016(03):77-80.

[5] ZHOU Sheng-tao. The Reveal and Prevention in Android Security Technology[M]. Beijing: The People's Posts and Telecommunications Press,2015.

[6] LIN Ya-ming. Application of MVVM Design Pattern and MVP Design Pattern Based on ZK[J]. Journal of Chongqing University of Arts and Sciences (Natural Science Edition), 2012 (06) :72-74,78.

[7] ZENG Lu. Application Research of MVP for Android[J]. Computer Engineering and Software, 2016(06):75-78.

[8] FU Cun-jun. Design of Transparent Encryption System for Documents Based on Windows Kernel[J]. Journal of Chongqing University of Education,2015,28(03):171-173.

[9] ZHU Song-bai. Design and Implementation of AES Algorithm Based on FPGA[D]. Nanchong: China West Normal University,2016.

[10] LUO Zi-yu. Improvement and Application of the AES Algorithm Based on Multi-core Android Platform[D]. Shanghai: Shanghai Normal University,2016.

[11] WANG Yan. Design and Implementation of Security System for Documents on Linux Platform[D]. Xi’an: Xidian University,2014.