Security Transmission Scheme of Sensitive Data for Mobile Terminal

Jicheng He[1,2](✉), Minghui Gao[1,2], Zhijun Zhang[1,2], Li Ma[1,2], Zhiyan Ning[1,2], and Jingyi Cao[3]

[1] Nari Group Corporation State Grid Electric Power Research Institute, Nanjing 211106, China
[2] Beijing Kedong Electric Power Control System Co., Ltd., Beijing 100192, China
[3] China Electric Power Research Institute, Beijing 100192, China

Abstract. In response to mobile terminal threats, this work is innovative, and gives practical solutions and prototype systems. The work first analyzes in detail the entire process of the generation, transmission and use of user-sensitive data within the Android system. In order to achieve resource sharing and data transmission between different applications and processes, the Android system provides a mechanism for inter-process communication based on Binder. Sensitive data is transmitted through Binder as a channel in the system. But sensitive data is carried out in the form of clear text during Binder transmission, which allows malware to easily intercept and tamper with sensitive data based on Binder communication, such as SMS content and GPS location information. In response to this problem, based on the above research, this work innovatively proposes and implements an adaptive transparent encryption protection scheme for sensitive data in the Android system from generation to use throughout the life cycle, effectively preventing sensitive data from being. The threat of theft and tampering by malicious third parties guarantees the privacy and integrity of user sensitive data during the internal transmission of the system. In addition, the system provides users with a simple and flexible operating experience, allowing users to independently protect specific types of sensitive data in specified applications, enhancing the ease of use and practicality of the system.

Keywords: Sensitive data · Mobile terminal · Android · Security transmission

1 Introduction

With the rapid development of mobile Internet in recent years, mobile phones, tablets and other mobile terminals have become the most closely related electronic devices [1] in people's life and work. Compared with traditional personal user computers, mobile intelligent terminals contain more user privacy information and daily sensitive data due to their rich sensors and good portability [2, 3]. However, the research on the security of intelligent terminals and the protection of user privacy data in intelligent devices are still in the initial stage, so it is of great significance to study the protection of user privacy in intelligent devices [4, 5].

X. Sun et al. (Eds.): ICAIS 2021, LNCS 12737, pp. 124–134, 2021.
https://doi.org/10.1007/978-3-030-78612-0_10

Strategy analytics released the global distribution of smartphone operating systems in the second quarter of 2019. According to the report, the market share of Android operating system in the world has reached 84.6% (the highest proportion ever). It is no doubt that Android has become the most dominant smartphone system. According to the Report on Mobile Phone Security in China in the First Quarter of 2019 released by 360 Internet Security Center, privacy theft programs accounted for 81% of the newly added malicious programs in the Android system in the first quarter of 2019, far higher than other types such as service charges. To sum up, it is of great significance to study the ecological environment security of Android system and the whole Android system.

The Android system allocates and manages resources as a unit of process. When a process needs a certain service, it must apply for the service from the corresponding manager, who will respond to the request. Among them, the request and response to the service are conducted by Binder. In the process of issuing a service request, the program will first pass the requested data and the type of service to the Binder, which will then pass it to the corresponding service manager, who will then pass back the response through the Binder. Therefore, Binder as if is used for communication between the client and the server, the postman, however, the "postman" is not very safe, because the data in the transmission process of Binder is expressly transmission, is equivalent to the postman send letters is no envelope, as long as through the attack of Binder, carry to intercept "postman" letters. Data can then be sent out via Broadcast, Intent, Content Provider, etc. Binder packet contents are stored in the form of clear text during the whole communication process. This means that once the packet is intercepted by attackers, the privacy of users will be compromised, which will bring great harm to the security of users. As for Android system, due to the serious fragmentation of its ecosystem, many devices provide or can easily obtain Root permissions, so that an attacker or malicious software can easily steal and tamper with sensitive information in Binder communication data.

Therefore, for users with high security requirements and specific application scenarios, The Binder based IPC communication mechanism of Android has some shortcomings in security, and it is in urgent need of a flexible security enhancement method to ensure users' privacy and system security.

With the popularization of 5G networks and the wide application of Android system, the mobile phone protection software industry has developed rapidly in recent years [6–8]. Various security protection software mushroomed as The Times require, such as 360 security Guard, QQ security butler, including foreign AVG phone antivirus, Norton phone security software, etc., are very popular in the market security protection software. As a new type of application software, mobile phone protection software has the greatest advantage of being convenient to use and protecting the personal interests of mobile phone users.

According to the 2018 Annual Android Security Ecological Environment Study released by 360, according to the detection data, by January 2019, the proportion of Android phones with security vulnerabilities in the detected devices reached a staggering 99.99%, which means that only 0.01% of Android phones failed to be detected with vulnerabilities. Compared with 2017, the security of Android phones has shown a significant downward trend. On the phones that were found to be vulnerable, information leaks accounted for an additional 95.6% of all vulnerabilities. In short, information breaches are by far the most common way to attack Android users.

However, AV-Comparatives, a security testing agency, has been testing about 250 domestic and foreign security apps on Android, according to Techradar. It found that more than 30% of malware detected in 2018 was successfully detected with no false positives, just a third of the time. What's more, some inferior protection apps even mark normal software as a virus.

In addition, most of the security protection software is mainly aimed at the detection and protection of mobile malware, and its main direction is the analysis of software characteristic codes [9, 10]. At present, it has basically realized the reminder and interception of all kinds of harmful behaviors, such as the short message of fee deduction, the short message of fraud, the call of charge fraud, etc. However, Binder transmission mechanism of Android system is not protected by software. As a result, there are still huge hidden dangers in sensitive information transmission for Android users.

2 Preliminaries

2.1 Binder

By analyzing the data flow of Binder, vulnerability points of Binder were found out and vulnerability points were demonstrated. The details are as follows:

(1) Binder framework layer creates an IOCtl Syscall with file descriptions as parameters and related data passed into the kernel.
(2) Binder Driver looks for the corresponding service, copies the data to the space of the server, and creates a thread to wait for processing.
(3) The server responds to the service and submits the request to the relevant hardware.
(4) Hardware response request.
(5) Pass the reply to the driver.
(6) Pass the reply to the client process.

The above model is the flow of data to the Binder of the client. For example, in the case of short message writing with the short message APP, during (1) the short message APP process will create an IOCtl SYscall and pass the relevant data (sending address and short message information) into the Binder Driver. In this process, the data is transmitted in the original clear text, that is, the malware can obtain relevant information as long as it intercepts (1) the process. And the malicious behavior may be for almost all of the Binder communication data, because in the picture above you can see in the model, each process is through the ioctl with Binder Driver to communicate, as long as stopped ioctl, text messages, to get the contact information, access to the phone call records the purpose of stealing sensitive information such as can be easily achieved.

2.2 API Hook

Through API Hook, the behavior of the method can be changed, thus affecting the behavior of the process. Its essence is to redirect the target method to other methods, perform some additional operations, and then jump back to the target method. By using API Hook technology to intercept key function calls, so as to achieve the encryption and decryption of sensitive data. The intercepting methods of these two types of APIS are described next.

Xposed Framework. Xposed framework is an android environment auxiliary tool, it can be on the premise of not modified APK affect the operation of the program or modify the system, based on the xposed framework can produce a variety of functional modules, the core Android system permissions management, clear memory, battery control, interface display, and other functions, and can be peace do not conflict with each other to function properly. Once you install the Ximproving framework in an android environment, developers will be able to use the ximproving framework's interface by writing the Ximproving module. The flag bits of some special data are set up in the Ximproving module, so you need to configure them in the corresponding project of your application.

Ptrace. Figure 1 and Fig. 2 take the Ioctl() method as an example, compare the call flow of methods before and after hook, and demonstrate the basic principle of hook.

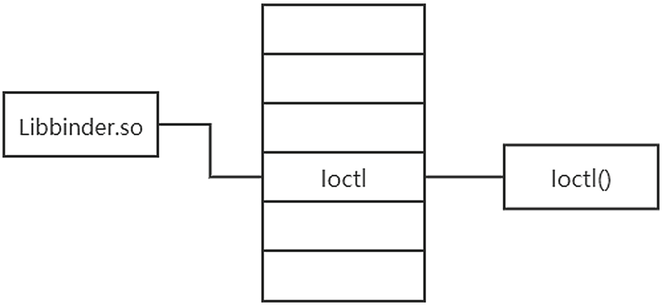

Fig. 1. Normal invocation flow of loctl

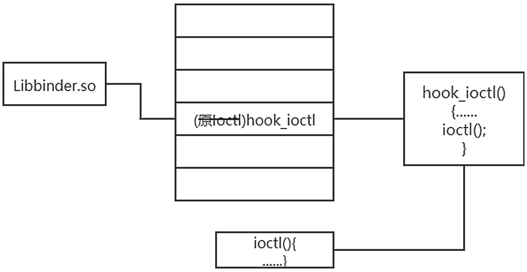

Fig. 2. Invocation flow of loctl after hook

As shown in Fig. 2, the normal invocation process is that libbinder. so calls the Ioctl method through the Ioctl address in its GOT table. After Hook, the GOT table enters the new Ioctl address, and libbinder. so will execute the hooked_IOCtl method before the Ioctl () execution.

As a shared library in Linux environment, SO library is used to provide a collection of various functional methods, and also provides a standard interface for dynamic loading. Similar to Windows system. DLL file.

(1) ShellCode is to use assembly language to write a program, the program can realize the SO library load, method search function, as well as the execution of SO library methods.
(2) ATTACH to the remote process via the remote process PID.
(3) Obtain and save the register value of the remote process, which can be used to restore the original state of the process after injection.
(4) Call dlopen, Mmap, DLSYm and other call addresses.
(5) Call process Mmap to allocate a section of storage space, and then write Shellcode and SO library path as well as other method call parameters in the space.
(6) Execute the code of process Shellcode, successfully load SO library, and implement SO library method to find and execute the method in SO library (in this method library can realize the desired method function).
(7) Restore the remote process register.
(8) Detach remote process.

3 Business Process

The security transmission system of sensitive data in Android environment aims to protect users' sensitive data from being stolen during transmission, and the whole process of protection is transparent to users, without requiring additional operations by users and without affecting their normal use.

This system does not need to register to log in, prevented as a result of database theft and lead to the disclosure of user basic information. This system encrypts where sensitive information is generated, only allows applications with decryption rights to decrypt, and allows users to freely choose the type of sensitive information to be protected and choose to grant permission to the application software, so as to ensure user privacy. This system is responsible for monitoring the basic security situation of Android phones, opening the corresponding protection module according to user input, and giving feedback and evaluation on the protection status of the whole system, as shown in Fig. 3.

The specific process involved in the system will be described in detail below:

(1) The program starts to obtain the basic status of the phone and display the current protection information.
(2) The user selects whether to turn on the safe mode.
(3) If the protection mode has been turned on, judge which functional module needs to be protected under the current situation.
(4) After determining the function module, determine the application program to be protected.
(5) After judging the success, implement the protection scheme for the corresponding program functions of the mobile phone and feedback the protection results.

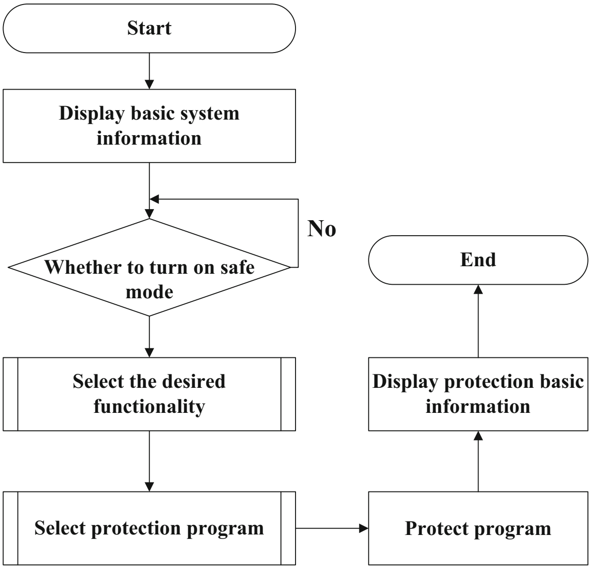
Start
Display basic system information
Whether to turn on safe mode
No
Select the desired functionality
Select protection program
Protect program
Display protection basic information
End

Fig. 3. Business process

4 System Architecture

The business process of the sensitive data security transmission system under the Android environment involves the client user and the server. The system adopts three-tier architecture, including the application layer, the business layer and the data layer. The system logic architecture is shown in Fig. 4.

(1) Data layer: Provides the data access interface for the business layer and accesses the service side through data access.
(2) Business layer: provide the logic realization of various functions of sensitive data security transmission system, and encapsulate four functions of application management, task planning, logging and security configuration;
(3) Application layer: the client user issues a request to the server; The server replies to the client after processing the request, in which the packets are encrypted through hash operation to generate the secret key before transmission.

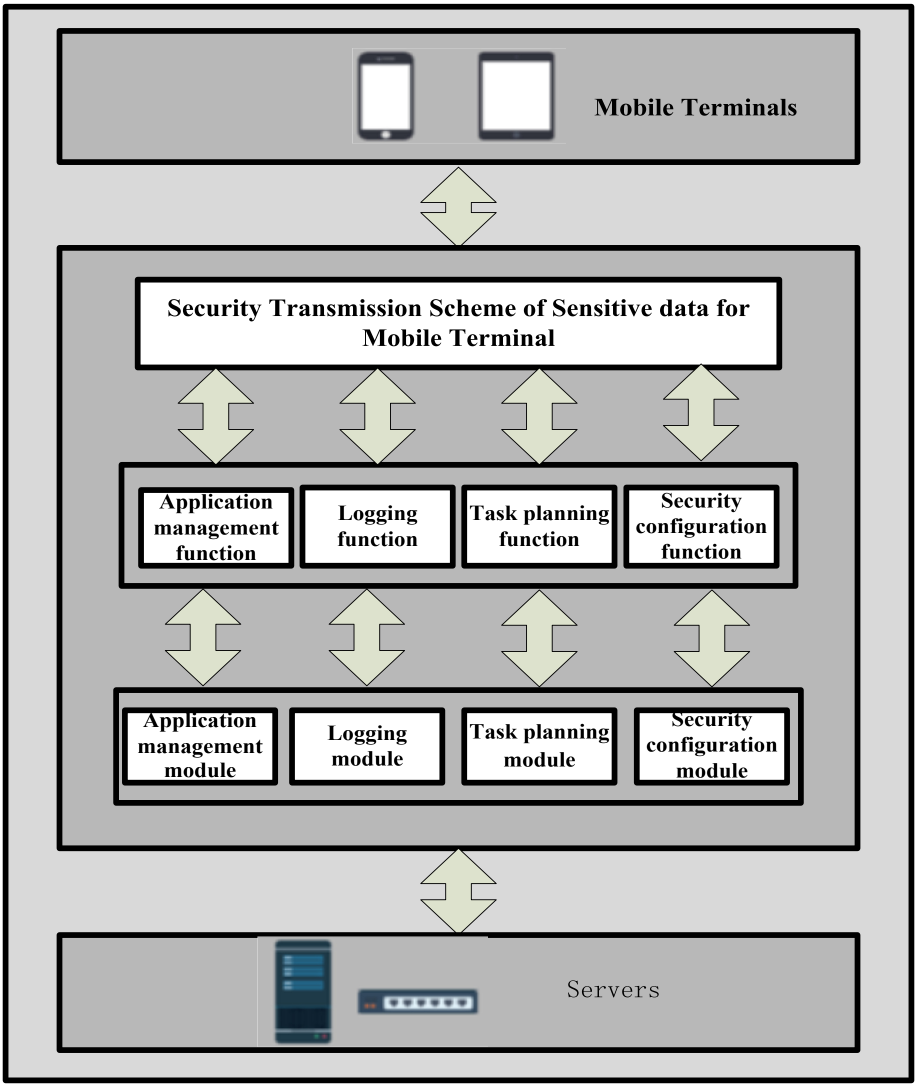
Mobile Terminals
Security Transmission Scheme of Sensitive data for Mobile Terminal
Application management function
Logging function
Task planning function
Security configuration function
Application management module
Logging module
Task planning module
Security configuration module
Servers

Fig. 4. System logic architecture

5 Key Technologies

Application management module is one of the core functions of the four modules of the sensitive data security transmission system under the whole Android environment. After the APP is authorized by the user, open the application management function, and you can see the list of sensitive software on the phone. Click one of the software at will, and the system can get the icon, name, version, software permission and so on of the software. Click on the CAMERA function, for example, after can see version

3.0, the APP permissions include android. Permission. CAMERA, android. Permission. RECORD_AUDIO permission, etc.

The main functions related to the above operations are information acquisition and data presentation. In addition, the most important role of application management module is to add protection to the application. Open the drop-down menu under this function, you can see that all the software is divided into GPS app and SMS app. Then click anyone, such as GPS app, and you can find that all the GPS-sensitive information involved in the operation of the app is in the GPS app. Then you can add protection to the GPS application and view details. After the corresponding application is used to add protection, the protection APP can protect the application without interruption for 24 h without affecting the normal use.

In this module, the protection of user data is implemented through the Xhook specific method operating, so the key can be passed into the method requiring hooks in the form of parameters, so as to achieve the key release and management.

In addition, for the selection of the key in the process of data protection, we carry out a hash operation on the data and take the operation result as the key, which can guarantee the randomness of the key and prevent attackers from using common statistical attack methods such as dictionary attack to decipher the ciphertext.

As shown in Fig. 5, taking SMS and GPS as examples, SMS and GPS services are protected service types. The encryption module intercepts the data sent by SMS and GPS services. After encryption, encrypted data is given to the binders, in which the encrypted data is delivered. When Binder data transmission is requested by application software with authority (such as official SMS and Baidu map in the figure above), it shall be decrypted to ensure that the application software with authority can obtain plaintext information. Applications without authority will obtain ciphertext even if the Binder is requested by them, or application software that attacks the Binder maliciously (such as stealing information locating APP). The following model can be applied to the process of applying the Binder to request sensitive data, for example, requesting memos, viewing call records, etc. SMS and GPS are just the tips of the iceberg.

During Binder transmission, another data flow is transmitted from the application software to the corresponding service manager, such as the process of editing and sending SMS messages with SMS application. In this information transmission model, the server is the application software and the client is the service manager. See Fig. 6.

Figure 7 describes the workflow of the data protection module, where function A and function B represent the services in the Android system, such as SMS service and GPS positioning service. The corresponding program, program represents the program that invokes the corresponding service, such as information APP, Baidu Map, etc.

The relevant steps of the system to protect the private data and feedback the results are as follows:

(1) Turn on the protection function and determine the functional modules to be protected.
(2) Determine the APP to be protected.
(3) Distribute the key for the Server side of the corresponding service and APP that needs to be protected.

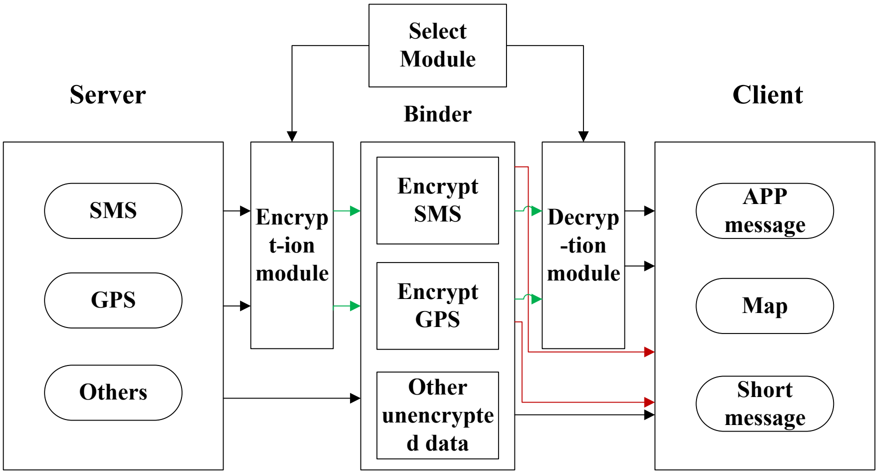

Fig. 5. Encryption process of sensitive information

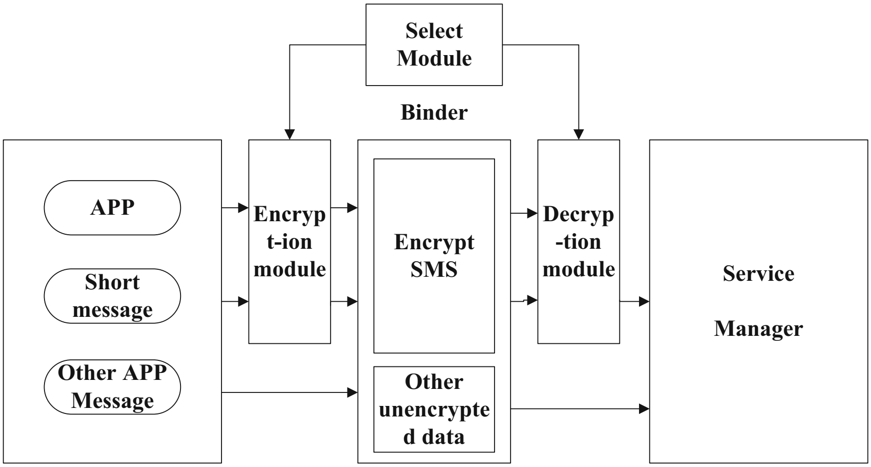

Fig. 6. SMS encryption process

(4) Select the corresponding Hook method to protect the sensitive data in Binder packet according to the functional module of protection.

(5) Call SM4 encryption and decryption module to process sensitive data.

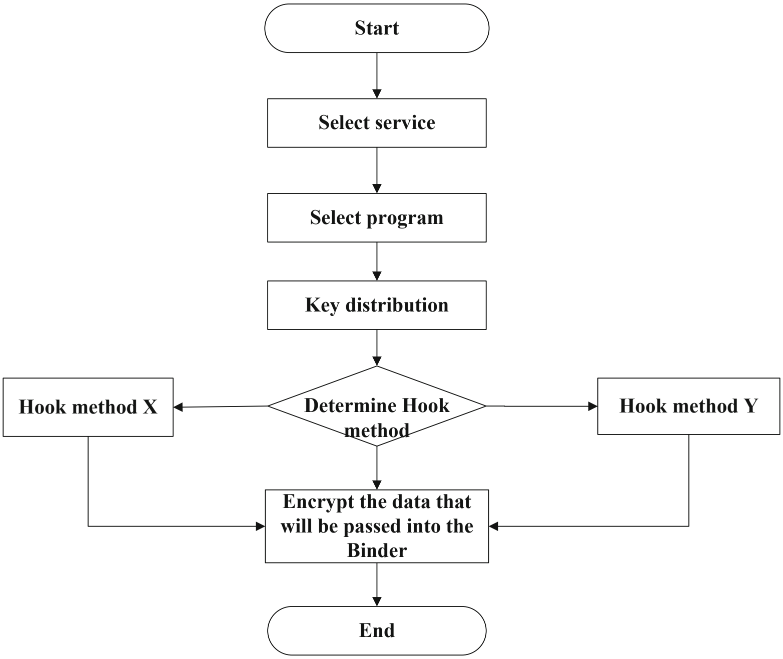

Fig. 7. Data protection business processes

6 Conclusion

From Android IPC communication mechanism, the innovation was put forward and implemented for the Android system sensitive data from production to use the entire life cycle of the adaptability of the transparent encryption protection scheme, effectively prevent the sensitive data by malicious third party to steal and tamper with the threat, guarantee the user sensitive data within the system the privacy and integrity in the process of transmission. Based on the realization of basic functions, the upper interface encapsulation provides users with simple and flexible operation experience, which enables users to protect the specific type of sensitive data in the specified application independently and enhances the usability and practicability of the system.

Acknowledgement. This research was funded by the Science and Technology Project Funding of State Grid Corporation of China (Research on Key Technologies of Energy Internet Mobile and Internet Security in 2019–2021, Contract no.: 5700-210955463A-0–0-00).

References

1. Choe, C., Chen, C.T., Nagao, S., et al.: Real-time acoustic emission monitoring of wear-out failure in SiC power electronic devices during power cycling tests. IEEE Trans. Power Electron. **36**, 4420–4428 (2021). https://doi.org/10.1109/TPEL.2020.3024986

2. Xu, J., Wei, L.W., Wu, W., et al.: Privacy-preserving data integrity verification by using lightweight streaming authenticated data structures for healthcare cyber-physical system. Future Gener. Comput. Syst. **108**, 1287–1296 (2018). https://doi.org/10.1016/j.future.2018.04.018
3. Xu, J., Wang, A.D., Wu, J., et al.: SPCSS: social network based privacy-preserving criminal suspects sensing. IEEE Trans. Comput. Soc. Syst. **7**, 261–274 (2020). https://doi.org/10.1109/TCSS.2019.2960857
4. Bello, O., Zeadally, S.: Intelligent device-to-device communication in the internet of things. IEEE Syst. J. **10**, 1172–1182 (2016). https://doi.org/10.1109/JSYST.2014.2298837
5. Xu, J., Liu, H., Wu, D.X., et al.: Generating universal adversarial perturbation with ResNet. Inf. Sci. **537**, 302–312 (2020). https://doi.org/10.1016/j.ins.2020.05.099
6. Al-Turjman, F.: Intelligence and security in big 5G-oriented IoNT: an overview. Future Gener. Comput. Syst. **102**, 357–368 (2020). https://doi.org/10.1016/j.future.2019.08.009
7. Rajy, L., Kiram, M.A., Ghazouani, M.E.: New security risk value estimate method for android applications. Comput. J. **63**, 593–603 (2020). https://doi.org/10.1093/comjnl/bxz109
8. Kelec, A., Djuric, Z.: A proposal for addressing security issues related to dynamic code loading on android platform. Comput. Syst. Sci. Eng. **35**, 271–282 (2020). https://doi.org/10.32604/csse.2020.35.271
9. Feng, R.T., Chen, S., Xie, X.F., et al.: A performance-sensitive malware detection system using deep learning on mobile devices. IEEE Trans. Inf. Forens. Secur. **16**, 1563–1578 (2021). https://doi.org/10.1109/TIFS.2020.3025436
10. Qamar, A., Karim, A., Chang, V.: Mobile malware attacks: review, taxonomy & future directions. Future Gener. Comput. Syst. **97**, 887–909 (2019). https://doi.org/10.1016/j.future.2019.03.007