Hindawi
Security and Communication Networks
Volume 2021, Article ID 6684456, 7 pages
https://doi.org/10.1155/2021/6684456

Research Article

A Sensitive File Abnormal Access Detection Method Based on Application Classification

Hui Liu,[1] **Hanxing Xue**,[2] **and Hui Lu**[3]

[1]*School of Computer and Information Technology, Beijing Jiaotong University, Beijing 102603, China*
[2]*Beijing Institute of Information Application Technology, Beijing 100086, China*
[3]*Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China*

Correspondence should be addressed to Hui Lu; luhui@gzhu.edu.cn

Received 25 December 2020; Revised 4 February 2021; Accepted 1 March 2021; Published 15 March 2021

Academic Editor: BEN NIU

As the access control mode of notepad files cannot meet the requirements of risk control for sensitive file hierarchical access, this paper proposes an application classification-based detection method for abnormal access to sensitive files. The application classification and file classification, access control policy mapping, and basic and preset policy detection are designed. Combining the operating system's identification control of different applications at runtime, we monitor the abnormal access of sensitive files by hierarchical applications. The cross-access experiment of different levels of application to different sensitive files verified the effectiveness and security of hierarchical access control strategy and sensitive file abnormal access detection and reduced the risk of disclosure of sensitive files.

1. Introduction

In recent years, with the development of big data, mobile Internet, cloud computing, and other technologies, network security incidents have occurred frequently, and issues such as system and data security, consistency, integrity, and user access rights have attracted attentions [1–3]. Based on this, the directions of system risk identification, vulnerability scanning, and anomaly detection have become hot topics for researchers [4–8]. Access authority and anomaly detection for mobile terminals are also a branch of research. There are two types of access risks to sensitive files in the notebook of mobile terminals: one is that an unauthorized user (such as malware or been rooted) accesses a file that should not be accessed, and the other is that a user accesses a file with a higher security level than the user. There are several sources of this access risk. Firstly, a high-privileged user or service such as root and MTP has unauthorized access to a file of an application. Secondly, a file that is higher than the user's security level is entered in the notebook. Thirdly, the access directories of application programs are crossed and shared. Fourthly, files are not classified according to the same security level.

Lin et al. proposed a hierarchical access control scheme for shared files based on attribute encryption [9]. Huang et al. proposed a Ciphertext-Policy Attribute-Based Encryption (CP-ABE) access control scheme based on multiple authorization centers in cloud storage [10]. Li et al. proposed a hierarchical access control scheme for encrypted storage of shared files [11]. These methods mainly use encryption to control file access. Wang et al. proposed a cloud computing access control model based on task roles to achieve fine-grained file access control [12]. The abovementioned methods are mainly static protection strategies. The encryption method has a great impact on usability, and the implementation of the system is also very complicated. Fine-grained access control methods must be embedded in the application program to be effective, and external access control strategies cannot be implemented. Song et al. proposed a security level evaluation model under the cloud computing network and determined the storage file security level through evaluation [13]. Hu designed the method and device for document security level identification to identify the security level of files [14]. Zhu proposed a security analysis method for sensitive data of mobile smart terminals

[15]. Detection and evaluation methods are used to determine the legal access of sensitive files or data. It can be seen from the above that researchers have explored different solutions to improve information security capabilities [16, 17].

In order to control the hierarchical access of sensitive files on mobile terminals further and improve the usability of mobile terminals, a method for detecting abnormal access to sensitive files based on application classification is constructed. Sensitive files are statically detected, and dynamic monitoring is integrated to prevent low-level users from accessing high-security files and prevent unauthorized users from abnormally accessing high-security files.

2. The Proposed Abnormal Access Detection Method

2.1. Framework of the Detection Method. There is the principle of least privilege for Android applications and data. In Linux, UID is the user's ID, indicating which user has run the program and is mainly used for permission management, but things are different in the Android system since it is a single-user system that allocates a different UID to almost every application. Unlike traditional Linux, each user is assigned the same UID if they are the same [18].

As for file access, Android also follows the basic Linux mechanism. For sensitive files in sensitive industries, we can design a model that only a certain application can access certain sensitive files. Accessing the file through other means, whether root, MTP, or other applications, is considered a risk. Logs and marks can be made in the system, and the risk can be graded.

A sensitive file abnormal access detection method based on application classification includes eight modules, such as file access behavior abnormal detection, user security level identification and identification, sensitive file security level identification and identification, file sensitive classification and access control property mapping, and file and directory access control mechanism. The sensitive file abnormal access detection method based on application classification is shown in Figure 1.

Sensitive file classification is to mark sensitive files and design the mark of document security level in the document name and document body. The classification process can be identified by the software. Application program classification is to mark and identify the security level of the application program according to the User Identification (UID) of the application program; for example, an application program with a high UID is designed to have a high-security level, and a software can be used to identify the application program identification.

In the file access behavior abnormal detection module, static detection of the file security level in the disk can be adopted, and the sensitive file access by the application can be dynamically monitored. By measuring the credibility of the behavior, the malicious level of the application program and the user's behavior can be judged. The following is the security level mapping, consistency check, and application behavior monitoring in the method.

2.2. Hierarchical Control of Sensitive Documents. In the laptop or terminal system, there is a "one-to-one" correspondence between an application and a user, that is, an application corresponds to a user, and the authority of the application is the authority of the user. In the Android system, the UID is used to identify the application. The application inherits the user's authority, thereby giving the application corresponding access authority.

2.3. User and File Security Level Matching Relationship Mapping. Three strategies can be used to map the matching relationship between the user and file security levels: benchmark strategy, preset strategy, and monitoring strategy.

The benchmark strategy is to set the UID size benchmark for the files corresponding to the security level. Assuming that the security level of the file F is t_i, the UID of the application program is u, and the benchmark for accessing the file F by the application program is $z(t_i)$ (graded according to the file security level), which represents a UID benchmark for accessing the file F. The hierarchical protection strategy adopted for sensitive files can be described as follows:

$$p = \begin{cases} 1, & u \geq z(t_i), \\ 0, & u < z(t_i), \end{cases} \tag{1}$$

where 1 means that it can be accessed and 0 means that it cannot be accessed. Table 1 shows the sensitive file protection strategy scenarios which adopt benchmarks. It can be seen that the $u = 68501$ can access the file security level t_3 (the benchmark is 40000), but it cannot access the file security level t_5 (the benchmark is 70,000). Only the application program corresponding to the UID higher than the benchmark can access the corresponding security level file.

The preset strategy is to preset the access authority and file access of the application. Assuming that the application can access the directory $d(x, j)$)which is a set, where x represents the file security level that the application can access and j represents the directory number that can be accessed. Then, the preset strategy for using sensitive file f (security level t_i) is described as follows:

$$d = \begin{cases} 1, & f \in d(x, j), x \geq i, \\ 0, & f \in d(x, j), x < i. \end{cases} \tag{2}$$

Then, t_i, the security level of file F, should be included in this directory. The preset strategy is shown in Table 2.

Monitoring strategy is the method of controlling application access to files based on security strategy in an environment without protection strategy, as shown in Table 3. Assuming that the application program u has a security level of j, and the file F has a security level t_i. If u wants to access file F, it must satisfy the condition $j \geq i$, or it is regarded as abnormal access.

If $s \geq t$, user U can open file F. Such a setting requires the user ID to be changed to meet the access requirements.

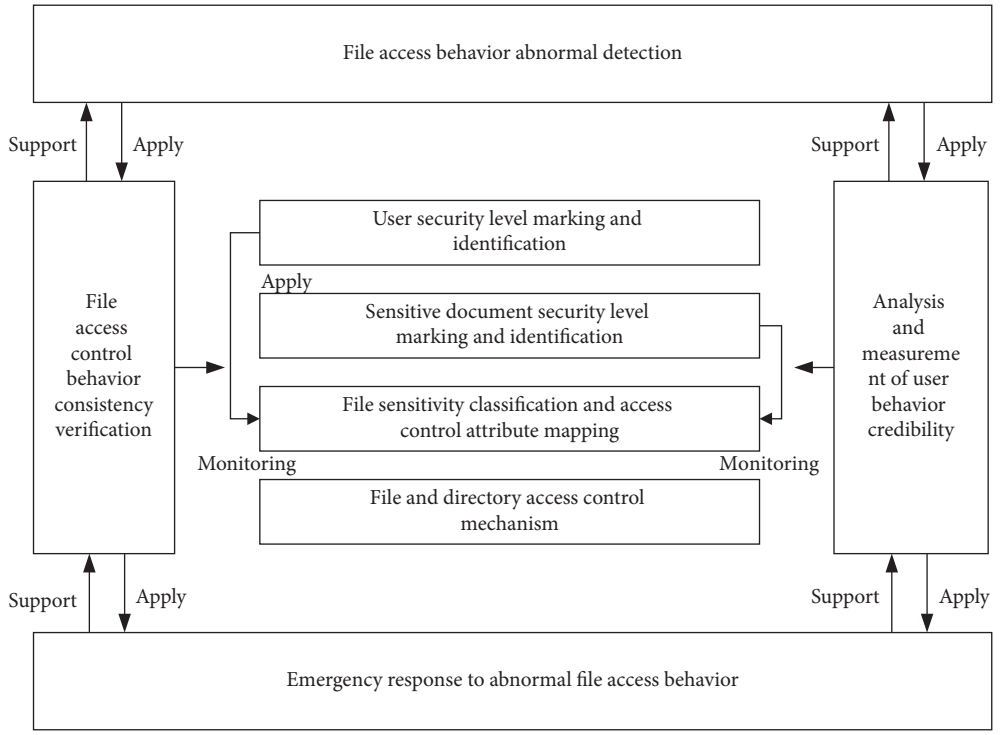

Figure 1: Composition of an abnormal access detection method for sensitive files based on application classification.

Table 1: Sensitive file protection strategy scenarios based on benchmark strategy.

Serial number	File t_i	Application UID u	File access benchmark $z(t_i)$	Access permission p
1	$F1\ (t_5)$	68501	70000	0
2	$F2\ (t_3)$	68501	40000	1
3	$F3\ (t_5)$	139929	70000	1

Table 2: Sensitive file protection strategy scenarios based on preset strategy.

Serial number	File t_i	Application UID u	Accessible directories $d(x, j)$	Access permission d and containment
1	$F1\ (t_5)$	385820	$d(3, 7)$	0, do not include
2	$F2\ (t_3)$	385820	$d(3, 4)$	1, include
3	$F3\ (t_5)$	678643	$d(6, 7)$	1, include

Table 3: Sensitive file protection strategy scenarios based on monitoring strategy.

Serial number	File t_i	Application UID u	Application security level j	Access permission
1	$F1\ (t_5)$	645830	4	0
2	$F2\ (t_3)$	645830	4	1
3	$F3\ (t_5)$	783849	8	1

2.4. Consistency Detection and Monitoring of Access Control Behavior. Sensitive file access control consistency detection means to monitor the consistency of application access sensitive files and check whether the application has access to sensitive files corresponding to the security level according to the benchmark strategy, preset strategy, and monitoring strategy.

2.4.1. Static Monitoring Based on Benchmarks and Preset Strategy. Assuming that all files $Fx = \{f_1, f_2, \ldots, f_n\}$, $n \in N$, on the terminal are the security level $Tx = \{t_1, t_2, \ldots, t_n\}$, $n \in N$, and the security level benchmark is $UBx = \{ub_1, ub_2, \ldots, ub_n\}$, $n \in N$. At the same time, suppose that the accessed application

identification UID is u, the security level is ut, and the accessible directory is $Dx = \{d_1, d_2, \ldots, d_m\}$, $m \in N$.

(1) Benchmark strategy consistency static detection method. Based on the principle that the identity of the application is greater than the benchmark identity strategy, i.e., $u \geq ub_x, ub_x \in UBx$. We detect the access permission of the application u. If it satisfied the condition $u < ub_x$ and the application has the access permission, it is determined that there is a risk of access to the application, and an alarm or warning is raised.

(2) Preset strategy consistency static detection method. According to the principle $ut \geq t_x$, $t_x \in Tx$ and the access strategy that application security level should be higher than that of sensitive files, we detect the access permission of the application u. If it satisfied the condition $ut < t_x$ and the file f_x is in the Dx directory, it is determined that there is an application access risk, and an alarm or warning is raised.

2.4.2. Access Control Dynamic Monitoring. Dynamic monitoring method: at the beginning of the application program accessing the file, according to the access policy that the application program security level is higher than the sensitive file security level, the application program u's access permission is detected. If $ut < t_x$, it is judged as abnormal access.

Analysis of user behavior is the process of credible evaluation of applications by the system. Abnormal user access behavior refers to violations of benchmark strategy, preset strategy, and dynamic strategy. Assuming that, in the interval of time t, the dynamic monitoring method is used to detect or monitor that the number c of abnormal times that the application u accesses the sensitive file is c, and then, the credibility metric is $tr = c/t$. The larger the value, the less credible it is. When the threshold TR is reached, which means $tr > TR$, it is determined that the application u is untrustworthy.

2.4.3. Exception Emergency Response. Abnormal emergency response is a measure taken to detect and monitor abnormal behavior. For statically detected abnormalities, the alarm prompts are generally used and delete abnormally sensitive files according to settings. In response to the abnormal access behavior of the application program that is dynamically monitored, measures to close the application program are taken to respond. For untrusted applications or usage behavior, a permanent shutdown strategy can be adopted.

3. Effectiveness Analysis of Abnormal Access Detection of Sensitive Files

Take Android as an example to design a sensitive file access tree. Traverse all paths, evaluate the possibility of unauthorized access, and obtain a total score through the evaluation mechanism to verify the effectiveness of hierarchical access control for sensitive files. The test environment is shown in Figure 2.

3.1. Safety Risk Analysis of Lagging Detection. Establish the effectiveness of detection methods for abnormal access to sensitive files. Static detection is to detect all files according to frequency. The feature of detecting and using at the same time will cause the detection of sensitive files to be missed. In a dynamic using environment, it is inevitable to produce file detection omissions. Assuming that the total number of files is w, because detection is a process, changed files or newly entered files may be lagging detection files. Assuming that the lagging detection time is $Dc = \{dc_1, dc_2, \ldots, dc_w\}$, $w \in N$. The lagging detection time is $Dc\ i = \{dc\ i_1, dc\ i_2, \ldots, dc_v\}$, $v \in N$, and then, the overall lagging detection safety risk is

$$\text{risk} = \frac{\sum_{i=1}^{v} dc\ i_i}{\sum_{i=1}^{w} dc_i}. \tag{3}$$

The experiments are conducted by putting 100 sensitive files to the folds in a notepad randomly. The detection software begins to check these files one by one with no interval in order to reduce the delay. The check goes with a roundtrip time, which is called lag. The lag will grow with more files to be checked. It may cost about 20 ms or more to process a file and depends on the performance of the notepad. The experiment checks 1, 2, ..., 17 files every roundtrip time, runs software to put these files in the folds, and finds abnormal access in Table 4.

According to Table 4, the maximum file number is 17. $\sum_{i=1}^{v} dc\ i_i = 493$, and $\sum_{i=1}^{w} dc_i = 1728$. The overall lagging detection security risk is risk = (493/1728) = 28.5%, which is still a high proportion. So, the hysteresis detection effect caused by static detection cannot be ignored. Therefore, a dynamic monitoring mechanism needs to be introduced.

According to the experiments, the lag time grows as the file number grows. Figure 3 shows that checking 17 files will cost almost 170 ms. If a sensitive file is an abnormal read in that interval, it will be a big risk.

The abnormal files in a roundtrip time will not grow as the reader open a file may cause more time than the check, but it may cause more risks. The experiments also verify the lag risk, which is shown in Figure 4. It shows that the lag risk slightly grows with more files to check. When the experiment has 15 files to check and also finds abnormal access, it causes more risk.

3.2. Analysis of Availability of Dynamic Monitoring and Safety Risk. The dynamic monitoring mechanism is to monitor the consistency of access control when users access files, so the monitoring performance must be higher than the user program's access performance to files; otherwise, the availability of software and privacy of files will be affected. Assuming that the monitoring time is $t\ d$ and the file access

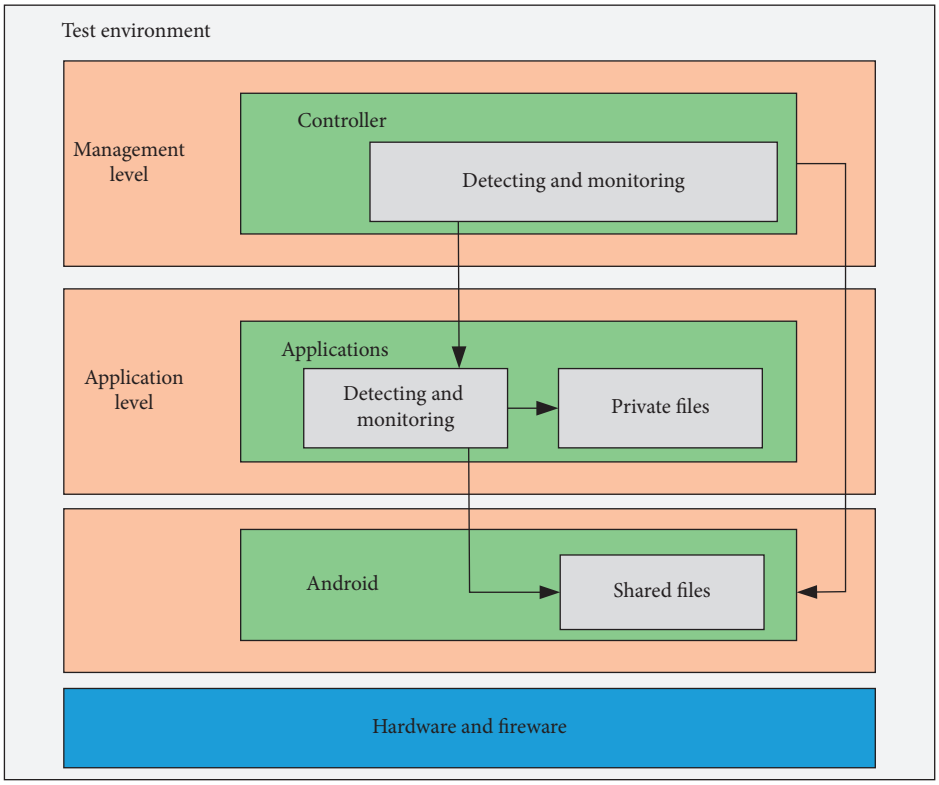

Figure 2: The test environment of Android.

Table 4: Experimental data.

Serial number	Lag	Abnormal	Abnormal lag
1	21	0	0
2	35	0	0
3	54	0	0
4	67	0	0
5	71	0	0
6	81	0	0
7	95	0	0
8	92	1	92
9	108	0	0
10	111	0	0
11	123	0	0
12	99	1	99
13	138	1	138
14	154	0	0
15	164	1	164
16	143	0	0
17	172	0	0
Summary	1728	4	493

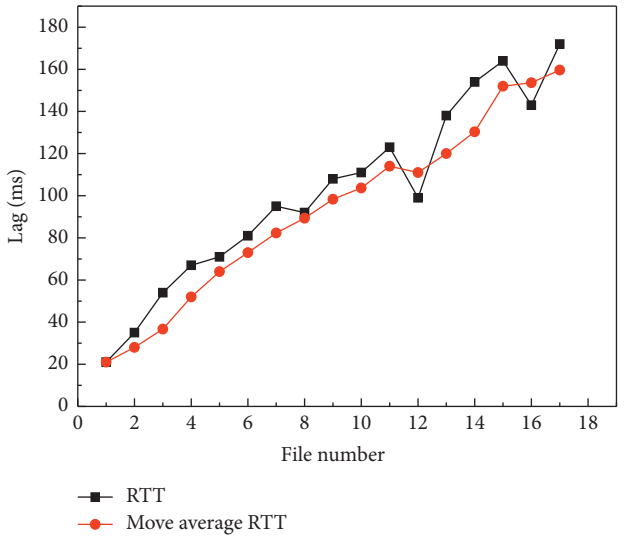

Figure 3: The lag grows with the growth in file number.

creation time is tcr, the available reference condition of the monitoring mechanism is expressed as $t\ d \leq tcr$. The risk of privacy leakage is Risk = $t\ d/tcr$. Table 5 is a case study of monitoring mechanism availability and security risk. It could be seen that the second case has the highest risk because the monitoring speed is faster than the access creation speed. Therefore, the rapid monitoring mechanism can improve the availability of applications and monitor software while reducing the security risks of sensitive data leakage.

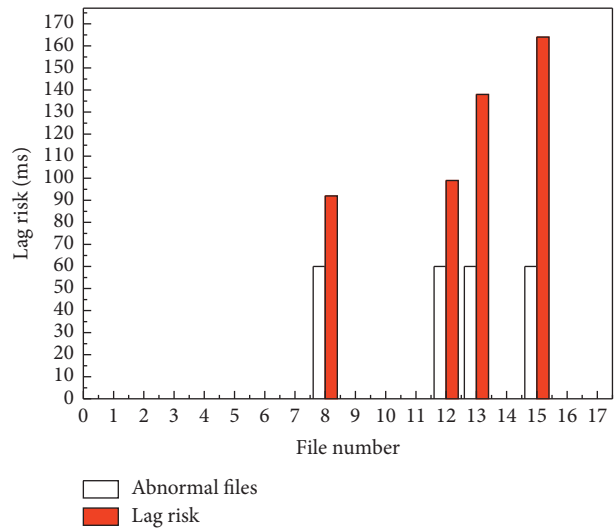

Figure 4: The lag risk slightly grows with more files to check.

Table 5: Cases of monitoring mechanism availability and security risk analysis.

Serial number	Monitoring time	Access creation time	Availability	Risk
1	23	58	1	0.396552
2	22	18	0	1.222222
3	23	63	1	0.365079
4	21	128	1	0.164063
5	20	46	1	0.434783

4. Conclusion

A sensitive file abnormal access detection method based on application classification includes application classification and sensitive file classification, application permission control, security control strategy of sensitive file classification, abnormal access detection of sensitive file classification, consistency evaluation, and validity evaluation of sensitive file classification detection. The abnormal access detection of sensitive files is realized through the cross and combination access test between the permission control of notepad and its operating system, hierarchical application, differentiated sensitive files, and the matching control between hierarchical control and application. Through the dynamic monitoring of sensitive documents, the security control availability and safety evaluation of sensitive documents are realized, and the effectiveness of the method is verified through the test and evaluation of important indicators.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the Guangdong Province Key Area R&D Program of China (Grant no. 2019B010137004) and the National Natural Science Foundation of China (Grant no. 61972108).

References

[1] Z. Tian, C. Luo, J. Qiu, X. Du, and M. Guizani, "A distributed deep learning system for web attack detection on edge devices," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1963–1971, 2020.

[2] X. Jing, Z. Yan, and W. Pedrycz, "Security data collection and data analytics in the Internet: a survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 586–618, 2019.

[3] M. Li, Y. Sun, H. Lu, S. Maharjan, and Z. Tian, "Deep reinforcement learning for partially observable data poisoning attack in crowdsensing systems," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6266–6278, 2020.

[4] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A survey on access control in the age of Internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020.

[5] Z. Yang, R. Wang, D. Luo, and Y. Xiong, "Rapid Re-identification risk assessment for anonymous data set in mobile multimedia scene," *IEEE Access*, vol. 8, pp. 41557–41565, 2020.

[6] H. Lu, C. Jin, X. Helu, C. Zhu, N. Guizani, and Z. Tian, *AutoD: Intelligent Blockchain Application Unpacking Based on JNI Layer Deception Call*, pp. 1–7, IEEE Network, New York, NY, USA, 2020.

[7] G. Pu, L. Wang, J. Shen, and F. Dong, "A hybrid unsupervised clustering-based anomaly detection method," *Tsinghua Science and Technology*, vol. 26, no. 2, pp. 146–153, April 2021.
[8] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. Tian, "Toward a comprehensive insight into the eclipse attacks of tor hidden services," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1584–1593, 2019.
[9] X. Lin and Y. Han, "Hierarchical access control scheme of shared file based on attribute encryption," *Journal of Yanshan University*, vol. 41, no. 5, pp. 450–456, 2017.
[10] Y. Huang, X. Wu, and J. Ling, "CP-ABE access control scheme based on multi-authorities in cloud storage," *Software Engineering and Applications*, vol. 9, no. 3, pp. 216–227, 2020.
[11] L. Li, G. Shi, X. Wang et al., "Implementation of shared file encrypted storage hierarchical access control scheme," *Chinese Journal of Network and Information Security*, vol. 2, no. 7, pp. 26–32, 2016.
[12] X. Wang and Y. Zhao, "A task-role-based access control model for cloud computing," *Computer Engineering*, vol. 38, no. 24, pp. 9–13, 2012.
[13] W. Song, Y. Wang, Y. Huang, and Z. Liu, "Simulation of security level evaluation model in cloud computing network," *Pioneering with Science & Technology Monthly*, vol. 28, no. 18, pp. 103-104, 2015.
[14] W. Hu, C. Li, C. Zhang, and D. Tang, *Patent: File Security Grade Identification Method and Device*, China Nation Intellectual Property Administration, Beijing, China, 2020.
[15] K. Z. Thesis, *Research on the Security of Sensitive Data in Mobile Intelligent Terminals*, Southeast University, Nanjing, China, 2014.
[16] H. Lu, C. Jin, X. Helu et al., "Research on intelligent detection of command level stack pollution for binary program analysis," *Mobile Networks and Applications*, 2020.
[17] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine-learning techniques," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242–3254, 2021.
[18] J. Zhao, "The access control of file base on Linux," *Computer Knowledge and Technology*, vol. 15, no. 18, pp. 300-301, 2019.