Wuhan University
Journal of Natural Sciences

2022, Vol.27 No.1, 011-016

Article ID 1007-1202(2022)01-0011-06
DOI https://doi.org/10.1051/wujns/2022271011

Security Protection Method of Energy Internet with Android

☐ ZHU Yayun[1], JIANG Lin[1], YUAN Anqi[2†], YUAN Yinghao[2]

1. China Electric Power Research Institute, Beijing 100192, China;

2. School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, Hubei, China

Abstract: As a product of the combination of information and energy technology, the energy internet is enormous and complex, and the absence of security safeguards at any aspect of it can cause incalculable damage. Aiming at the problems that Energy Internet terminals are difficult to be trusted and the integrity of massive terminals cannot be guaranteed, this paper designs and implements a set of comprehensive credibility measures and security protection schemes, isolates various malicious operations, and ensures that the system is always in a credible state. Specifically, we develop a secure TF card control program in the Android terminal application layer to realize the overall security scheme. Experimental results prove that this security protection scheme can effectively detect illegal application packages in energy Internet terminal devices, resist malicious programs, and protect key data from theft at the same time, achieving security protection in the "thing-to-thing" interconnection scenario of the Energy Internet.

Key words: Energy Internet; trusted execution environment; Android terminal; secure TF card

CLC number: TP 391

Received date: 2021-10-08

Foundation item: Supported by the State Grid Corporation of China Science and Technology Project Funding

Biography: ZHU Yayun, male, Engineer, Master, research direction: electric power network and information security. E-mail: zhuyayun@epri.sgcc.com.cn

† To whom correspondence should be addressed. E-mail: 850934851@qq.com

0 Introduction

Integrating modern communication, computer and control technologies, the Energy Internet is an energy peer-to-peer exchange and sharing network capable of coordinating power sources, energy storage devices and loads over a wide area to achieve a shift from centralized fossil energy use to distributed renewable energy use[1]. The Energy Internet has the characteristics of open sharing and various distributed devices are coordinated and optimally controlled. China currently advocates building a secure and intelligent Energy Internet to meet global electricity demand as clean and green as possible. However, with the massive access to a large number of the Internet of Things (IoT) devices brought about by the gradual expansion of the Energy Internet and the growth of confidential information and sensitive data as the volume of data continues to accumulate in the era of big data, most of the complex work in these end devices has to be done by the operating system. Compared with the traditional Internet, the security development of the Energy Internet still needs refinement, which leads to a fact that the terminal device as an important role in the Energy Internet has gradually become a new focus of the attack by the wrongdoers[2].

Due to the characteristics of low power consumption, Android devices are showing a high growth trend widely used up in the Energy Internet[3]. Ensuring the security of Android devices is a vital cornerstone of the current Energy Internet security protection.

1 Background

IoT is the foundation of the Energy Internet. The

security of the Energy Internet is guarded by IoT-related technologies. The United States is the first country to implement classified protection, and it has proposed a security plan between sensitive and non-secret level networks in its Multi-Level Information System Security Initiative (MISSI), which has meaning to the secure transmission of multi-level interconnection. However, the current foreign access control requirements and security requirements for multi-level interconnection systems are still not specific. Li[4] developed a directory protocol to solve the problems in the IoT resource sharing and open system interconnection. Zhao *et al*[5] proposed a research and implementation plan for the secure boot mechanism of embedded systems based on trusted computing technology: on the premise of the existing hardware architecture of mobile devices, a secure TF card is used to build an external trusted platform module (TPM), forming a complete trust chain from the Bootloader to the upper application, effectively ensuring that the integrity of the terminal is not damaged. Zhang *et al*[6] proposed a trusted access authentication protocol for mobile nodes in IoT environment. When the mobile sink node and the sensor node in the sensor network are authenticated, the sensor node and the mobile node complete mutual identity verification and key agreement, and there is no need for the base station to participate in the whole process. During the process of authentication, the pre-stored pseudonym and corresponding public and private keys of the mobile node are used to realize the anonymity of the mobile node, while a security certificate is given under the CK (Canetti-Krawczyk) model. Research on security early warning technology and visualization technology is also a crucial part of the research on power grid security situation awareness. Xu *et al*[7] proposed a hybrid AC-DC urban distribution network interconnection structure based on flexible DC interconnection with AC transformer as the core and DC busbar as the framework for the Energy Internet. Using the goals and characteristics of the Energy Internet as guidelines, this type of scheme meets the requirements of the Energy Internet and provides a security posture analysis of the Energy Internet network.

2 Android Terminal Security Protection Method

2.1 Threat Model

The attack of an intruder of an energy IoT Android terminal device may occur when the device is operated by a user or when the device is left unattended[8]. In this paper, we assume that the attacker of the IoT terminal is able to get hold of relevant information about the attack target, such as the system version number of the target device, the list of installed applications on the Android system, and the application operation preferences of the device user[9-11]. After that, the attacker will elaborate an intrusion plan based on the known information to the corresponding malicious program. The intrusion may be done by directly transmitting the installer of the malicious program, by providing a phishing link to lure the user to download the malicious program[12, 13] or disguising it as a normal Android application package (APK) installer and installing it by upgrading it, by indirectly transmitting the APK installer to the target device and waiting for the user to click on the installation trigger, by infecting the target device by means of mobile storage[14], and by intercepting the data information in the target system or uploading information, etc. The attackers aim to steal important data information or install some kind of Android malware on the target terminal device, and then use the implanted malicious APP as a medium to launch more attacks[15] to withdraw power, steal information, paralyze the target system, and infect more targets, thus hindering the normal and stable operation of the Energy Internet and forming the emergence of accidents during network operation[16].

2.2 Overall Framework

The boot process of the mobile terminal within the Energy Internet needs to be strictly controlled. After the mobile terminal is powered on, the boot program starts the embedded system, and by transforming the relevant code with metric function in the boot program, combined with the trusted security hardware, the corresponding metric value is stored as the starting point of the whole trust chain. With this trust root as the base, the kernel is trusted, and if it is safe and trusted, the control is given to the Kernel. Then, through the trusted status of Kernel, the trusted modules and layer applications in the middle layer of the system are metricized, thus realizing a complete trust chain from bootloader to user-state applications.

When the application requests to fetch data, it calls the middle layer driver, and the security module will verify the request legality. After the verification, whether the record exists will be checked, if yes, the data to be decrypted will be taken out from the target location of the security hardware of the record. At the same time, the

application's identity information and encrypted data will be sent to the security hardware. The security hardware retrieves the encrypted key of this application from the storage device, and the decryption module gets the key and decrypts the encrypted data. After the key is decrypted successfully, the decrypted data will be returned to the encryption module which will return the decrypted data to the secure application. Before storing data to the storage device, the secure application will call the middle tier driver which performs an identity check on the secure application. After the check is passed, the secure application will request a call to the secure hardware encryption engine. The secure hardware encryption engine encrypts the privacy data inside the secure hardware and returns it to the middle tier driver which plays a role that deposits the encrypted data into the storage device. The process of fetching privacy data from the storage device is reversed. Thus, it is possible to ensure that the data existing in the storage device is always encrypted and that the plaintext data cannot be easily accessed. The overall framework is shown in Fig. 1.

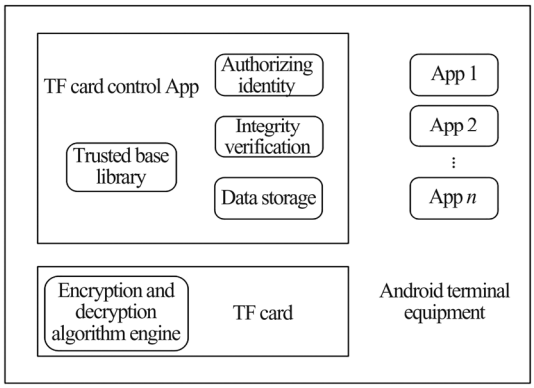

Fig. 1 Overall framework

2.3 Android Signature Verification Based on Secure TF Card

We build a trusted software base based on the secure TF card to realize the functions of identity verification, integrity checking and data storage of the trusted software base through the encryption and decryption algorithm services in the secure TF card. Identity verification and integrity checking are implemented based on new signatures to measure the legitimate identity and file integrity of applications or APKs in the system. Data encryption storage is embodied in the form of a trusted baseline repository, which is mainly used to store the baseline values after metrics.

Considering that the role of signature which containing the identification information of APP developers, this paper proposes the signature implementation scheme based on secure TF card to ensure security. Due to the requirement of higher security in the Energy Internet "thing-to-thing" interconnection scenario, this paper adopts a whitelist implementation similar to the firewall access rules in the signature verification to allow the programs that pass the verification to run. Otherwise, the program will be regarded as unknown risk software one and its related operations will be blocked. According to this principle, the security of Android system is ensured.

In order to improve the efficiency of the signature verification process, the length of the encrypted message should be shortened before using the asymmetric encryption algorithm for the signature message, so this paper uses the digest algorithm to generate a small fixed length value, which is one of the reasons for using a secure TF card in the implementation of trusted protection for trusted end devices. The specific approaches of this paper for signature verification based on secure TF card are shown below.

Firstly, the APP to be signed will be read in the form of byte stream and copied to the location specified by the secure TF card program, then wait for the secure TF card to complete the digest algorithm and generate the corresponding result, which can be temporarily stored on the system in the form of "txt" text file. Of course, for security consideration, this summary value is also not suitable to be stored explicitly in the Android system environment, which should avoid the risk of its being stolen as an impersonation of normal applications. Then the security TF card performs the signature operation on the simplified APP information, that is, the summary information with fixed length just generated is used as the input source of the next signature algorithm, and after the encryption of the private key in its algorithm, the signature base value of the application can be obtained. At this point, the process of generating a new signature of an APP based on the secure TF card is completed.

When the secure TF card control application captures the installation behavior of the APK file, it performs signature verification checks on the application installation package and only allows the installation of applications that have passed the verification. The APK signature verification overall is to check whether it is in the system whitelist based on the signature base library after initialization. If it is not in the trusted list (i.e. whitelist), then it may be a third-party tampered file.

After the monitoring service of the secure TF card control application starts, when the APK package is in-

stalled, the application uses the interception mechanism to intercept the behavior, obtains the APK path, and invokes the service of the secure TF card according to the path. Then the interception mechanism uses the secure TF card class trust root to verify its signature, obtains APK signature information based on the binary content of the APK by using the signature verification algorithm, and compares it with the APK signature information in the benchmark library. The verification passes if the result is consistent. If the APK has been tampered with by a third party, the signature comparison will not pass, resulting in a failed verification and prohibited installation. Even with the same certificate, if the package names are different, two different applications will be found to exist at the same time when the installation is successful. The same application with the same package name but with a different certificate will lead to the failure of overwriting the installation. Therefore, only if the package name and signature information are the same, the installation is allowed to run. The APK verification flow chart is shown in Fig. 2.

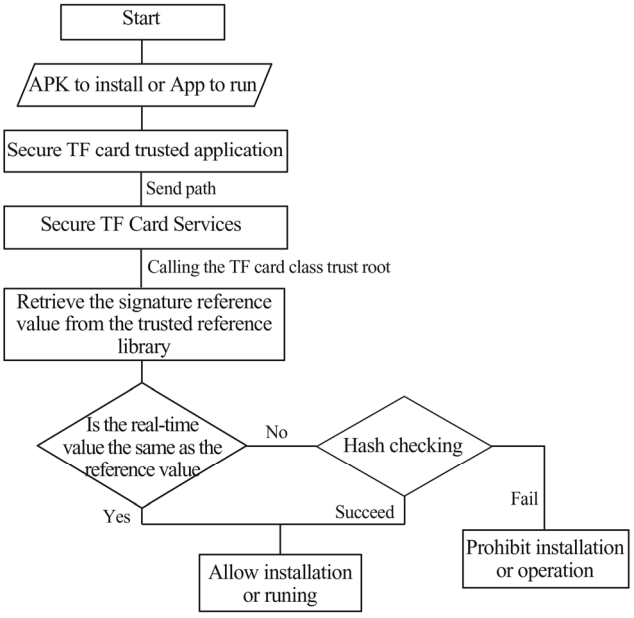

Fig. 2 Android signature verification based on security TF card

3 Experiment and Analysis

3.1 Experimental Environment

In this paper, we mainly study the security protection of Android device terminals in the Energy Internet, so the hardware devices and the experimental environment under software conditions are configured based on Android. This paper also uses Xiaomi Tablet 4 with 4 GB system memory and 64 GB storage capacity as the terminal device for experiments and analysis. During the installation of the application, the Android Studio, which is officially launched by Google, was used as the development environment. After finishing the code writing, select Generate Signed Bundle/APK in the Build menu item to generate the APK file of the control application for the secure TF card and enter some basic signing key information.

The initialization process is the generation of each digest base value. Under the security assumption that the current environment is trustworthy, the first step is to initialize the signatures of all Android applications that have been installed in the device. This step also extends the security from the secure TF card to the control applications in the device. For the convenience, users, just add some judgments in the code to distinguish whether the control application is running for the first time and whether the signature value is initialized in the trusted base library rather than do the initialization manually. In the development environment, whether the initialization process is executed successfully or not is shown through the abd log of Android Studio. If the initialization fails, the application cannot pass the verification and then will fail in running.

3.2 Simulated Attacks

In the IoT environment of hydropower stations such as power grids and other critical facilities, the use of Android devices is often used as entry points by attackers. In the threat model, it has been pointed out that Android devices mainly face various attacks launched through applications, such as disguising as normal applications to gain users' trust and then implanting on the system, or replacing the file information of installed applications in the system by external removable storage. This experiment simulates the following attack experiments: using the APK file with tampered contents to simulate the intruded malware and trying to install it on the Android terminal device; replacing the directory file of the installed application in the storage path of Android system and waiting for the user to perform the relevant operation to trigger the malicious attack. The purpose of the experiment is to test whether the authentication and integrity checking functions of the security scheme are successfully implemented.

Through the simulated attack experiment and its performance effect, it can be seen that the design scheme has the ability to resist the APK of unknown applications

and tampered applications, satisfies the functions of authentication and integrity checking of the trusted software base, and achieves the design goal of security boundary determination and security checking.

3.3 Performance Evaluation

3.3.1 Performance overhead

The current mainstream security software needs to be resident in the Android system, listening to a large number of files such as SMS, audio and video, etc., and consuming resources to upload and analyze the recent files in the process. As the Android terminal protection method of Energy Internet environment designed in this paper is implemented based on the security TF card, the application service controlling the security TF card mainly listens to the operation of APK files and APP in the background, which consumes less resources.

In Android Studio development environment, one can use adb tool to connect to the experimental device, and then use top command to view the occupied system resources, just like the operation on Linux system, and the object of the view is the processes running in the current system. The main purpose of using this command is to dynamically monitor the system resource allocation, which is characterized by the ability to display each process in order for a single system resource attribute, such as CPU, memory, disk IO, etc. The results of the experiment are shown in Fig. 3 below.

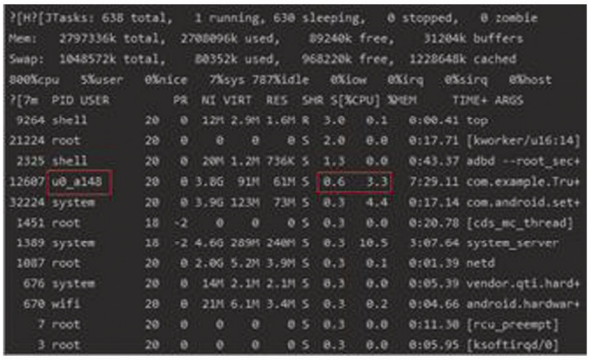

Fig. 3 System resource occupation

3.3.2 Time overhead

In the development environment, functions such as the system's application startup time are used to calculate the time consumed for signature verification, and the statistical results are shown in Table 1.

Table 1 The startup time of the application with and without check

ms

Number	Browser application		Camera application		Contacts application	
	With check	Without check	With check	Without check	With check	Without check
1	475	434	483	445	337	298
2	517	463	487	440	498	429
3	469	458	477	410	412	395
4	459	455	466	435	401	390
5	477	460	463	436	416	400
Avg	479.4	454	475.2	433.2	412.8	382.4

When a user uses an application on Android or installs an APK file, the service of the secure TF card controlling the application is automatically triggered to start verifying its new signature value. In terms of the whole process, the core is to compare the real-time value obtained by using the signature algorithm of the secure TF card and the benchmark value. The acquisition of the real-time value does not take long in practice, and this step can be completed in less than 1 s, while the user does not feel any significant difference when comparing the original start-up time. As for the benchmark value, it can be obtained from the benchmark library, and the time required is negligible.

3.4 Experimental Results and Analysis

From the above out experimental results, it can be seen that Xiaomi Tablet PC installed with the Android end device security protection scheme proposed in this paper can successfully resist the attack of suspicious application APK. The design motive of this solution originates from the trusted computing technology, using the secure TF card with high security as the trust base and extending it to the application layer to strengthen the security of Android system. Specifically, this solution uses the secure TF card to generate a new "signature" that is different from the digital certificate and signature in APK, so as to achieve the verification and validation of the trusted software base, through which the security of the Android system is improved. The experimental results show that the device has certain defense ability against foreign unknown APK files and unverified APPs, and can identify suspicious programs and stop them from running.

Although the results show that the scheme is designed to achieve malware protection, the research scheme in this paper has the following limitations: In terms of the current attacks on Android system, most of them are through the injection of malware to invade the device, and there are even web pages, scripts and other ways to guide or trick users to download the APK file of malware, which is essentially a malicious program as a springboard to make more serious attacks. Tighter restrictions on the regulation of applications can significantly reduce the likelihood of attacks. However, there is a paradox between security and convenience, and the negative impact of enhanced security with reduced convenience is unavoidable. However, the design proposed in this paper compares with Google's initial proposal of backing up phone data, which is measured in hours, and the former's time overhead is similar to the time it takes to launch an app in Android.

4 Conclusion

In the new energy era, the issue of trusted interconnection of the Energy Internet is of great concern, and trusted computing is also a top priority for social and technological development. The security protection framework opens up a new path for the "thing-to-thing" interconnection security protection method for the Energy Internet. Based on the theoretical analysis for different security issues and the results of the above summary and analysis, this paper uses trusted computing technology to enhance security for Android device terminals in the Energy Internet, constructs a trusted protection system based on secure TF cards, and thus adds trusted network connections, trusted real-time metrics, and trusted control technologies to form a trusted terminal protection framework for Android in the Energy Internet. This paper simulates the security protection and malware detection experiments of Android devices in the Energy Internet, conducts security and feasibility analysis, and proves that the scheme proposed in this paper can achieve security protection in the Energy Internet "thing-to-thing" interconnection scenario.

References

[1] Sun Q Y, Teng F, Zhang H G. Energy Internet and its key control issues [J]. *Journal of Automation*, 2017, **43**(2): 176-194.

[2] Liu Y B, Wang Q, Zeng Q, *et al*. Key technologies and prospects for energy consumption control in 5G networks in the context of energy internet [J]. *Power System Automation*, 2021, **45**(12): 174-183.

[3] Meng H, Thing V L L, Cheng Y, *et al*. A survey of Android exploits in the wild [J]. *Computers & Security*, 2018, **76**: 71- 91.

[4] Li X C. *Design and Implementation of a Management System for IoT Connected Light Directory Protocols and Services* [D]. Beijing: Beijing University of Posts and Telecommunications, 2015(Ch).

[5] Zhao B, Fei Y K, Xiang S, *et al*. Research and implementation of secure boot mechanism for embedded systems [J]. *Computer Engineering and Applications*, 2014, **10**: 72-77.

[6] Zhang X, Yang X Y, Zhu R, *et al*. Trusted access authentication protocol for mobile nodes in the Internet of Things environment [J]. *Computer Applications*, 2016, **11**: 3108-3112.

[7] Xu C, Liang R, Cheng Z H, *et al*. Smart distribution network security situational awareness for energy internet [J]. *Power Automation Equipment*, 2016, **36**(6): 13-18.

[8] Gasparis I, Qian Z, Song C, *et al.* Detecting android root exploits by learning from root providers [C]// *Proceedings of the* 26*th USENIX Conference on Security Symposium*. New York: ACM, 2017: 1129-1144.

[9] Imtiaz S I, Rehman S U, Javed A R, *et al*. DeepAMD: Detection and identification of Android malware using high-efficient Deep Artificial Neural Network [J]. *Future Generation Computer Systems*, 2021, **115**: 844-856.

[10] Korkmaz I, Metin S K, Gurek A, *et al*. A cloud based and Android supported scalable home automation system [J]. *Computers & Electrical Engineering*, 2015, **43**: 112-128.

[11] Zhang Y Q, Wang K, Yang H, *et al*. Survey of Android OS security [J]. *Journal of Computer Research and Development*, 2014, **51**(7): 1385-1396(Ch).

[12] Watanabe T, Akiyama M, Kanei F, *et al*. Understanding the origins of mobile app vulnerabilities: A large-scale measurement study of free and paid apps [C]//2017 *IEEE*/*ACM* 14*th International Conference on Mining Software Repositories* (*MSR*). Piscataway: IEEE, 2017: 14-24.

[13] Shen D, Li Z, Su X, *et al*. TinyVisor: An extensible secure framework on android platforms [J]. *Computers & Security*, 2018, **72**: 145-162.

[14] Yan J, Qi Y, Rao Q, *et al*. LSTM-based hierarchical denoising network for Android malware detection [J]. *Security and Communication Networks*, 2018, **2018**: 1-18.

[15] Shaheen J A, Asghar M A, Hussain A. Android OS with its architecture and Android application with Dalvik virtual machine review [J]. *International Journal of Multimedia and Ubiquitous Engineering*, 2017, **12**(7): 19-30.

[16] Xiang Y, Tian X X, Zhou S, *et al*. Construction and application of digital creative platform for digital creative industry based on smart city concept [J]. *Computers & Electrical Engineering*, 2020, **87**: 106748.

□