A OS Integrity Verification Using Independent Storage in Medical Device

Jae-Kyung Park*

*Senior Researcher of Cyber Security Research Center, Korea Advanced Institute of Science and Technology, Nonhyun-ro 28gil 25, Gangnam-gu, Seoul 135-854, Korea
wildcur@kaist.ac.kr

Abstract. With the recent development of Internet of Things, remote medical devices are actively introduced with much attention. These medical devices however, are not fully protected from hacking or various threats by malignant codes. This paper proposes a method for investigating the integrity of operating systems in order to prevent medical devices from being attacked by hacking or malignant codes. In the method proposed by the paper, stability of medical devices is ensured by constructing an independent storage device inside the terminal of the medical devices, and investigating the integrity of the operating system using the storage device, and making it impossible to modify the operating system by general application programs. It is expected that safer remote services of medical devices can be available through the method proposed by the paper.

Keywords: Medical Device, Integrity, Kernel, Integrity Verification, Boot Loader.

1 Introduction

Recently, medical devices are diversified in various ways along with the trends of smart era, and open structures such as remote treatment or remote adjustment are being increasingly adopted, departing from conventional closed structures. The operating system of these medical devices is also transiting from dedicated firmware to general operating systems [1,2]. From the perspective of hardware, the devices are now employing high specification units, which have been used in servers, for treatment, analysis and transmitting the results such as RF, Wi-Fi, screen, storage device, and high-performance CPU rather than mounting modules for simple treatment services. According to these changes however, more security threats on terminals of medical devices have emerged rapidly [8,13]. By supporting various network access environments through an open type platform, the security threat of PC or servers are being overcome since the objects of the treatment are patients. Although many researches and efforts have been made to handle these security threats of medical devices, only threats are being reported currently, and no perfect security schemes against malignant codes or hacking have been reported[9, 10,15].

This paper proposes a method for investigating the integrity of an operating system using an independent storage device inside the terminal to ensure a safer treatment service environment from the threat of open type operating systems of medical devices, which uses physically independent storage devices in the boot loader and terminals.

2 Related Researches

Integrity is ensuring that information of the subject is only open to those with authentication/certification, and that the information is modified by the subject only[11]. The measures taken to ensure this integrity include control on physical environments, limitation to data access, and maintenance of strict certification procedures. Recent researches which are being performed on the integrity of operating systems will be described below.

2.1 KNOX, Samsung Electronics

Samsung Electronics recently commercialized a security solution of operating systems, named KNOX. This solution is a hardware device-based multi-aspect security solution for preventing modification through the Linux kernel and Android operating system. The solution may be considered a comprehensive medical devices security platform comprising apps including the integrity investigation of the operating system of medical devices and information protection, threat response method by network infrastructure through VPN connection per app, safe management scheme of individual and corporate data according to the trend of BYOD, and management scheme of terminal loss [12,14].
Reviewing the integrity investigation method of Samsung KNOX which has been published in FOSDEM 2014, methods for integrity investigation of the operating system and integrity investigation of the block level have been proposed. Especially, it is necessary to review the integrity investigation method of the operating system at boot level comprising Secure Boot, Trusted Boot, etc. ARM TrustZone technology is used in the case of Trusted Boot for storing encrypted fingerprints of all boot loaders and operating system kernels, and this information is used for investigating integrity [16].

3 Contents

The method of verifying the integrity of operating systems of medical devices, which is proposed in this paper, will be described. Some medical devices use their own dedicated operating systems, and others use general systems such as Windows or Linux. In this study, the design will be limited to the medical devices using the Linux system. This method however, can be applied to other operating systems in the same way.

In the method proposed by the paper, an independent security memory area secured for loading integrity investigation data, and the interface which enables access to the security memory area is prepared as a kernel module. The module is loaded as a kernel at the boot loader along with operating system booting, and access to the corresponding data can be made by an integrity investigation application using the module.

3.1 Ensuring Independent Security Memory Area

The security memory area proposed by the paper means a new storage media which does not exist in conventional medical devices. The storage media can be accessed on the operating system only when device drivers are present like any other parts of the device[19]. Basically, all peripherals equipped in the PC have device drivers which control the parts respectively. This is the method by which users can use the corresponding device through a common procedure by formulating the interface controlling said device [3,4,5]. The operating system performs I/O on actual devices this way through the I/O interface defined by the device driver in response to the request of a user. These device drivers distributed by being included in the operating system, or separately managed from the operating system by managing them in the form of objects are called modules. Terminals of medical devices also use the same method as those of the PC, and device drivers exist in the same way.
The independent security memory area proposed by the paper is designed to prepare device drivers in itself and to control access to the corresponding areas, as shown in Figure 3. Access to the corresponding memory area is enabled by module Read access at boot level and operating system level. For the Write function however, the access is only possible at the boot loader area by design. The Write function is removed to prevent the possibility of cheating and modification of the corresponding security memory area while permitting the Read function when modification of the operating system is occurred by a malignant user, in which the case controlling authority of the operating system is stolen and access to the prepared security memory area becomes available.

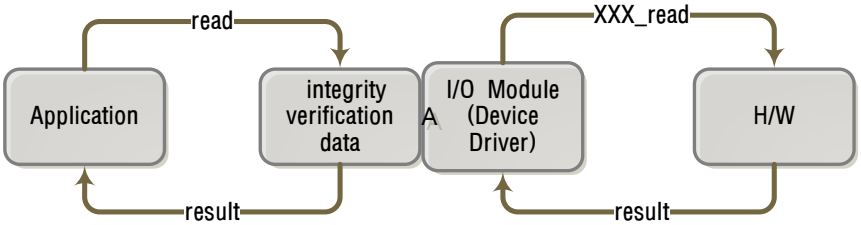

Fig. 3. Access of Integrity Verification Data

One step further, the function of decoding encryption is added to the device driver for secondary control of access to the data stored during input and output. The driver is prepared in advance in the form of a kernel module, and managed separately from the

kernel. The kernel image is loaded at boot level, and then corresponding drivers are loaded into the kernel.

3.2 Verification of Integrity Using Boot Loader

As for general boot loaders, GRUB has been used in commercial Linux operating systems after the now outdated LILO. Also, the u-boot boot loader has been used mainly in embedded devices. The boot loader plays the role of checking whether the H/W is working properly before booting into the operating system, and loading the operating system into the RAM area. Each boot loader has formalized logics, and plays the role of sequentially checking the status of devices and operating the operating system for use in the actual situation[17,18]. Therefore, the system can be booted by modified operating system taking advantage of the corresponding area. Actually, the act of rooting, which is considered the most serious problem, is the security of the terminals of medical devices, and also starts from modification of this boot area in many cases rather than exploiting loopholes of the operating system [6,7]. Therefore, many manufactures of terminals of medical devices and researchers are processing the boot level by using many techniques so that only a safe operating system can be loaded through screening by applying solutions such as used by KNOX, as described above.

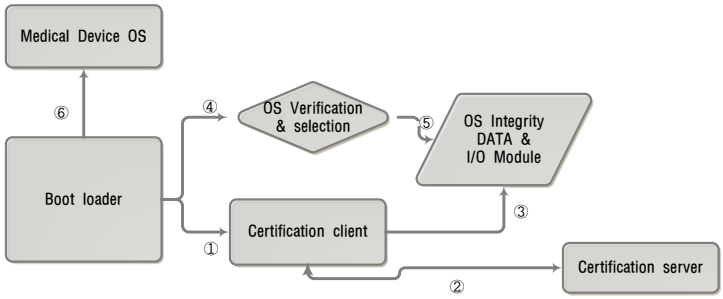

Fig. 4. Integrity Verification of Boot Level

In the method proposed by the paper, the boot loader not only initializes the process before entering a main operating system and finds and boots the operating system to be used in the system from supplementary memory such as CF memory or HDD, but provides the function that developers can program in person to add supplementary functions, as shown in Figure 4. As one example, the operating system can be downloaded from outside through a network communication to load into the main memory or to perform actions of controlling specific devices. By taking advantage of these capabilities, access can be made to the security memory area prepared above and integrity of the operating system can be certified.

The process of investigation on the integrity of the operating system proposed by the paper will be presented. Original terminals are distributed with the status that the integrity investigation data of the certified operating system image is recorded at the

memory area at the time of manufacturing, and the investigation is proceeded in the following order:

- Step ①, ②, ③: After proceeding the processes of booting sequentially during terminal booting, in order to renew the investigation data of the integrity of the operating system of the security memory area, interface and processing are proceeded so that data can be renewed through communication with the manufacturer of the terminal at the stage of investigating cheating or modification of the operating system.
- Step ④ : Functions of the investigation for identifying cheating or modifications of the operating system are executed.
- Step ⑤: For identifying cheating or modifications, determination is made on whether the data of the security memory area and the value of the actual operating system match.
- Step ⑥: Upon completing the investigation, the operating system is loaded into memory and I/O modules included in the security memory area are copied into the file system to be used by the operating system with the controlling authority transferred to the operating system.

Operating systems of medical devices can be divided into a kernel image and a ram disk (ramdisk.img), and an actual file system can be identified by un-packaging the ram disk.

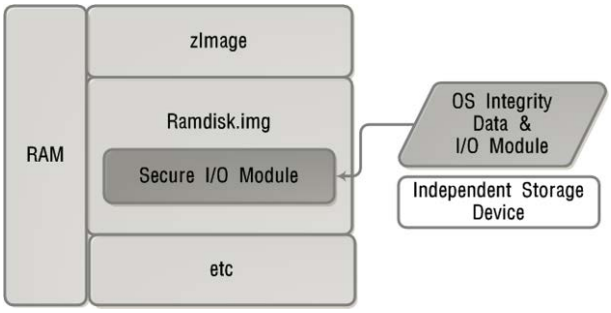

Fig. 6. Secure I/O Module OS & Merge Process

I/O modules are copied to the location of modules/locations of the file system while loading into memory at the boot loader and, after booting the operating system, access to the security memory area at related applications are enabled.

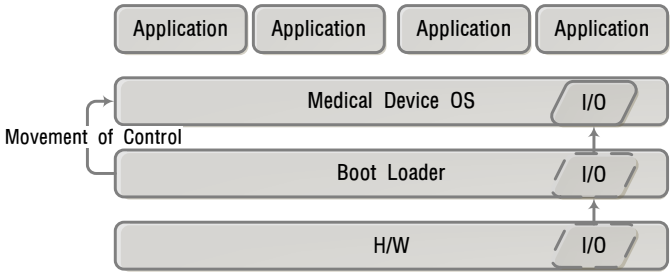

Fig. 7. Position Change of Secure I/O Module

Through the process described above, the kernel I/O module managed by the boot loader for controlling the security memory area is loaded to the operating system of medical devices, and the authority of control is transferred.

3.3 Investigation of Integrity at Operating System Level

This paper reviewed on the investigation by separately preparing necessary elements as Linux applications through formalization of the processes before booting of an OS. First, the virtual memory provided by Linux is used as a physically independent storage device part. This way, block device drivers are prepared and virtually one storage device is prepared for mounting.

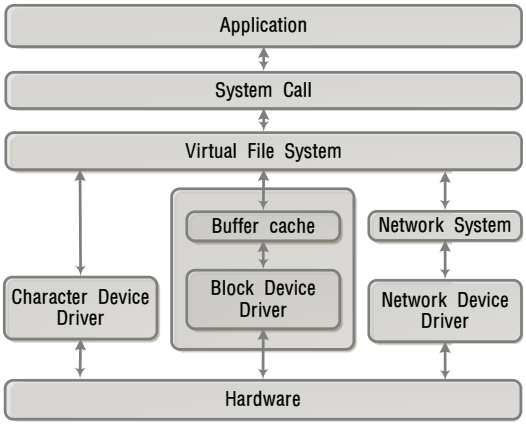

Fig. 9. Linux System Architecture

Figure 9 shows the location of the block device in the structure of the Linux system for preparing a prototype. The prototype uses a virtual ram disk which sets a part of the memory as the partition of the security memory area, the independent storage device for storing the integrity values. The virtual ram disk can set the memory like a hard disk drive that enables files to be stored in the memory, and cheating and modification of the kernel and applications can be identified by storing the integrity values at the virtual ram disk generated through the block device driver. Although the

prototype should be renewed in its integrity investigation data through communication with the manufacturer of the terminal after proceeding with the booting process sequentially during the first terminal booting, there is the problem of making an agreement with the manufacturer, and the integrity investigation data was renewed by the encrypting application instead in the prototype.

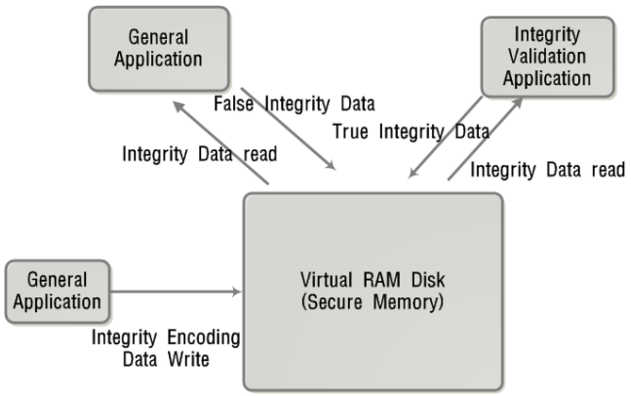

Fig. 10. Prototype Model Diagram

Figure 10 shows the diagram constructed with prototype models, which consists of a virtual ram disk and an application. An encrypting application extracts hash values of general files or binary files to be checked for cheating or modification, and stores the values in a generated virtual ram disk, so that cheating or modification can be detected in integrity checking applications in the future. General applications are the applications other than the integrity checking applications, which are not certified for accessing a virtual ram disk, and integrity checking applications read in the integrity value stored in the virtual ram disk by an encrypting application in real time, that detect cheating or modification by comparing the values with the hash values of the file to be currently checked for integrity.

4 Conclusion

The investigation of the integrity of the operating system of medical devices as proposed in this paper adopted a design, in which a separate storage device is prepared for performing primary investigation at boot level before an operating system is executed, and continuous investigation of integrity is performed after a controlling authority is transferred to the operating system using the same investigation data. Also, the device driver performing I/O to the storage device is prepared as a kernel module in advance and managed separately from the kernel. The module is loaded together with the operating system which is loaded into memory by a boot loader at boot level rather than distributed by including it in an operating system, making it possible to ensure safety against leak of firmware. This design scheme is considered to provide n advantageous method of maintaining integrity safely when applied to the environment of medical devices.

References

[1] http://www.phonearena.com/news/Samsung–Knox-found-to-have-a-serious-vulnerability_id50670

[2] http://secunia.com/advisories/product/15863.

[3] http://www.cvedetails.comvulnerability-list/vendor_id-6276/XEN.html.Last accessed April 4, 2012.

[4] http://www.fda.gov/NewsEvents/Newsroom/PressAnnouncements/ucm416809.htm

[5] http://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/UCM356190.pdf

[6] http://digitalcommons.law.scu.edu/cgi/viewcontent.cgi?article=1578&context=chtlj

[7] http://mirian.kisti.re.kr/futuremonitor/view.jsp?cont_cd=GT&record_no=240568

[8] http://www.zvei.org/Publikationen/ProtectionofMalwareinMedicalTechnologyandHospitalIT.pdf

[9] http://www.trendmicro.com.br/cloud-content/us/pdfs/home/cs_medical-center-central-georgia-tms.pdf

[10] http://www.ieee-security.org/TC/SP2014/papers/SoK_c_SecurityandPrivacyinImplantableMedicalDevicesandBodyAreaNetworks.pdf

[11] http://www.himss.org/files/himssorg/content/files/medical-defendingnemawhitepaper.pdf

[12] Integrity Protection Solutions for Embedded Systems, FOSDEM 2014 Brussels, Belgium, February , 2014

[13] Survey of Security Threats and Countermeasures on Android Environment. Joonhyouk Jang 2013.12

[14] RAISE. Enye lkm rookit modified for ubuntu 8.04. http://packetstormsecurity.com/files/75184/Enye-LKM-Rookit-Modified-For-Ubuntu-8.04.html. Last accessed Sep 4, 2012.

[15] Vulnerability report: Xen3.x.http://secunia.com/advisories/product/15 86 3. Last accessed April 4, 2012.

[16] Sangorrin, Daniel, Shinya Honda, Hiroaki Takada. "Dual operating system architecture for real-time embedded systems." In Proceedings of the 6th International Workshop on Operating Systems Platforms for Embedded Real-Time applications (OSPERT), Brussels, Belgium, pp. 6-15. 2010.

[17] A. M. Azab, P. Ning, Z. Wang, X. Jiang, X. Zhang, and N. C. Skalsky. Hypersentry: enabling stealthy in-context measurement of hypervisor integrity. In Proceedings of the 17th ACM conference on Computer and communications security, CCS '10, pages 38{49, New York, NY, USA, ACM, 2010.

[18] DINABURG, A., ROYAL, P., SHARIF, M., AND LEE, W. Ether: malware analysis via hardware virtualization extensions. In Proceedings of the 15th ACM conference on Computer and communications security (New York, NY, USA, 2008), pp. 51–62. CCS ’08, ACM, 2008.

[19] D. Clarke, G. E. Suh, B. Gassend, M. van Dijk, and S. Devadas. Checking the integrity of a memory in a snooping-based symmetric multiprocessor (smp) system. Technical report, MIT LCS memo-470, http://csg.csail.mit.edu/pubs/memos/Memo-470/smpMemoryMemo.pdf, 2004.