J Inf Process Syst, Vol.16, No.1, pp.42~48, February 2020
https://doi.org/10.3745/JIPS.03.0130

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

A Secure Operating System Architecture Based on Linux against Communication Offense with Root Exploit for Unmanned Aerial Vehicles

KwangMin Koo*, Woo-yeob Lee**, Sung-Ryung Cho*, and Inwhee Joe***

Abstract

This paper proposes an operating system architecture for unmanned aerial vehicle (UAV), which is secure against root exploit, resilient to connection loss resulting in the control loss, and able to utilize common applications used in Linux. The Linux-based UAVs are exposed to root exploit. On the other hand, the microkernel-based UAVs are not able to use the common applications utilized in Linux, even though which is secure against root exploit. For this reason, the proposed architecture uses a virtualized microkernel on the Linux operating system to isolate communication roles and prevent root exploit. As a result, the suggested Operating system is secure against root exploit and is able to utilize the common applications at the same time.

Keywords

Architecture, Microkernel, Root Exploit, Security, UAV

1. Introduction

Lately, the usage on unmanned aerial vehicle (UAV) is increasing such as humanitarian, disaster response, search and rescue, and civilian leisure [1,2]. Accordingly, various security issues have been addressed on UAV. For example, physical security for UAV, such as collision or unauthorized location, is required and researched [3]. Privacy security resulted from video data sniffing is also needed as UAV provides a number of applications [4]. Wireless communications security is one of the considerations for UAV network as well [5].

Especially, secure operating systems of UAVs is one of the most significant security issues, in order to provide reliable and stable UAV control. The most operating systems of commercial UAVs are based on Linux [6]. However, Linux systems are vulnerable on a wireless UAV control. The root exploit offense, which is able to interfere with UAV control, is one of the most well-known security issues [7,8]. Though the microkernel-based operating systems for this issue are developed, this system is not able to use the useful applications commonly utilized in Linux systems and it is too complicated to implement. Moreover, the microkernel-based UAV is vulnerable to fall down by the forced shutdown resulted by buffer overflow offense [9]. In short, the Linux-based operating system is not secure from root exploit,

※ This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.
Manuscript received May 28, 2018; first revision January 8, 2019; accepted September 26, 2019.
Corresponding Author: Inwhee Joe (iwjoe@hanyang.ac.kr)
* Dept. of Computer Software, Hanyang University, Seoul, Korea ({codingple, kyougt}@hanyang.ac.kr)
** Dept. of Computer Science and Engineering, Hanyang University, Seoul, Korea (matias12@hanyang.ac.kr)
*** Dept. of Computer Engineering, Hanyang University, Seoul, Korea (iwjoe@hanyang.ac.kr)

Copyright© 2020 KIPS

and the microkernel-based operating system is not able to utilize the useful applications and vulnerable to the forced shutdown.

For this reason, this paper proposes a Linux-based operating system architecture, for UAV which is not only able to utilize the common applications but also robust against the vulnerabilities on UAV controls. This system uses the virtualized microkernel-based operating system on Linux to isolate wireless communication in UAVs. This enables the UAVs using the proposed architecture to prevent root exploit, falling by the forced shutdown offense, and to operate the Linux applications.

2. Related Works

The UAVs without an operating system, such as firmware-only UAVs, cannot use various functions. Since it is a pervasive stable operating system, in which diverse functions are already developed, Linux is widely adopted as the operating system for UAVs, e.g., robot operating system (ROS) [10-12].

Yet, Linux systems are known to be vulnerable to communication attacks. Since Linux is an open-source project, the data communication procedure is easily inferred. This results to loot network sessions, and hackers can insert the command evoking a shell through the buffer overflow vulnerability known as “Ghost” [7]. Using this shell, the attackers are able to exploit root privilege with “Use-After-Free” [8].

The other operating systems for UAVs are based on a microkernel, e.g., seL4 [13]. Most of recent research to enhance UAV security with operating system architecture are based on microkernel [14,15]. The microkernel-based operating system consists of only necessary components, which enable it to be lightweight. In contrast with Linux in that it needs to be implemented in person, this operating system is used for security. However, in other words, manual implementation all the time has an effect on its portability, and common applications are not able to be utilized on a microkernel-based operating system, which are easily used on Linux. In addition, as you can see [9], the UAV might be expose to shutdown offence. Some vulnerability of FreeRTOS has detected lately, which is a microkernel-based OS, and one of those is remote code execution vulnerability occurred by buffer overflow.

3. Architecture Overview

The proposed architecture is largely composed of Linux and microkernel-based operating system virtualized on it. The microkernel-based operating system has two virtual network interface, one is linked to the Ethernet interface of UAV, another is for communication with Linux. Also, the virtualized operating system has an interface in order to connect its virtual network interfaces. In short, Linux applications of UAV communicates via the microkernel-based operating system.

This structure is beneficial for network security of UAV. When UAV with the operating system is under attack, initial network access point is not Linux itself. In other words, UAV is not exposed on the offense using the vulnerability of transport layer of Linux. Attackers are not able to exploit root of the microkernel-based operating system that became the target of attack via a network. In addition, attacked aiming buffer overflow which gives rise to the forced shutdown, the microkernel-based operating system to which network access is reachable is turned off, not Linux. This allows UAV to continue its mission without falling using applications in Linux. Fig. 1 describes the proposed architecture.

The packets from outside of UAV are entirely forwarded to the microkernel-based operating system called Communication Manager, and Communication Control Interface is defined to make two interfaces of Communication Manager be able to connect. In Linux, various applications for UAV functions are running and control UAV hardware modules, communicating with Communication Manager. For in case of communication problems, an application receiving reports from Communication Manager to check if it is running and connection loss, called Report Manager.

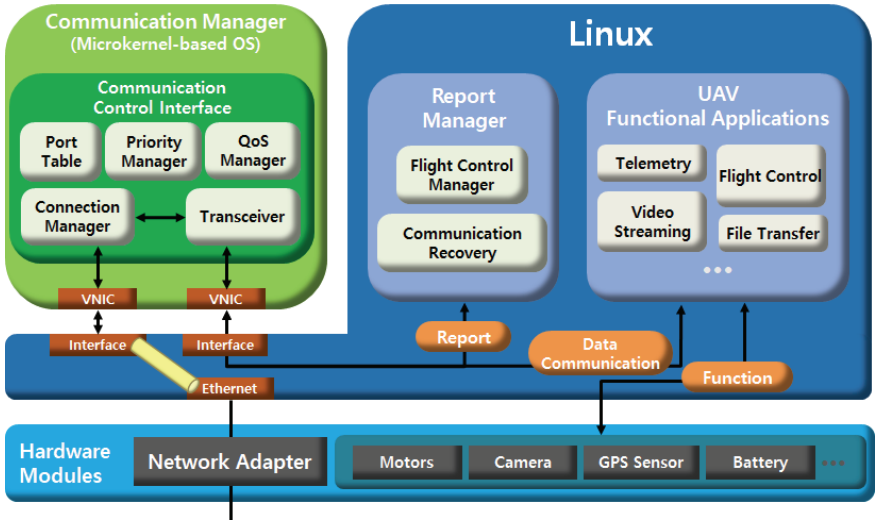

Fig. 1. Illustration of the proposed architecture.

3.1 Communication Manager

Communication Control Interface in Communication Manager is constructed of Port Table, Priority Manager, QoS Manager, Connection Manager, and Transceiver (Fig. 1). This interface is designed so that the virtual interfaces in Communication Manager are connected each other, as a result, Linux is able to establish and manage the secure connection with outside of UAV via Communication Manager.

Port Table is in charge of indicating, storing, and removing ports of communicating processes. Using Data Type field which can be identified using definitions of data types it has in advanced, Port Table parses Data Type into a registered port number in a table of applications running on Linux to route packets.

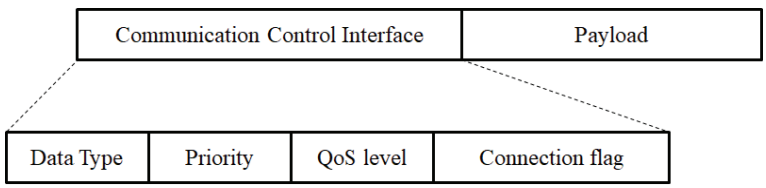

Fig. 2. The packet format of Communication Control Interface.

Priority Manager determines the priority of packets Transceiver transmits and receives. According to the priority field in packets, Priority Manager inserts the packet into relevant one of the priority queues

Transceiver reads. Reading the packet first of higher queue, Transceiver sends packets in sequence.

QoS Manager manages data for quality of service. This component configures bandwidth which should be guaranteed for data communication according to data types, such as control and multimedia. For this end, QoS manager checks QoS level field of packets as well.

Connection Manager detects a connection request by connection flag. Connection Manager confirms Connection flag in a packet and generates new Transceiver dedicating a specified session of connection. After that, the Transceiver communicates with the user who sent the request.

3.2 Report Manager

Fig. 3 describes that Transceiver component of Communication Manager send a heartbeat message, and UAV applications reports when the connection loss occurs with Transceiver. Report Manager plays a role of communication health supervisor for resilience. Report Manager receives reports from Transceiver and UAV Applications while they are communicating. This reporting mechanism informs connection loss when one is waiting for a response from the other too long than timeout. In case that the lost connection is controlling UAV, Flight Control Manager copes with the routing path of UAV, which is already configured method, e.g., hovering and awaiting reconnection. Moreover, utilizing Transceiver reporting heartbeat, Communication Manager killed by attackers is recovered by Communication Recovery in Report Manager.

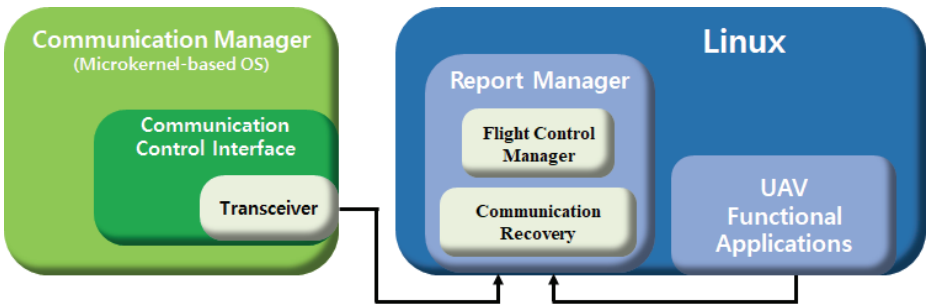

Fig. 3. Communication reports for recovery by Report Manager.

4. Implementation Issues and Analysis

To implement this architecture, there are some considerations and issues. First of all, the platform of the operating system structure must be selected, such as microkernel, hardware architecture, and virtualization method. Especially, the microkernel must contain the specified function, including communication control. In this process, some issues might occur. For example, the hardware architecture may need to support virtualization technology, such as trusted execution environment (TEE), or the microkernel must be able to run on the core architecture on the composed system. The platform we designed consists of seL4 microkernel, ARMv7 architecture, and Docker.

In the proposed architecture of the operating system of UAV, diverse applications and functions are available in ease. Besides, this structure is secure and resilient against communication offense with well-known vulnerabilities. Table 1 describes the differences among operating systems of UAV.

Table 1. Differences among operating systems of UAV

	Linux-based OS	Microkernel-based OS	Proposed architecture
Root exploit	Vulnerable	Invulnerable	Invulnerable
Common app.	Available	Unavailable	Available
Functions	Diverse	Limited	Diverse
Forced shutdown	Vulnerable	Vulnerable	Invulnerable
Connection recovery	Unavailable	Unavailable	Available

In contrast with Linux, the proposed operating system is invulnerable against root exploit and shutdown offence by separating the communication module. In addition, this operating system is able to use applications commonly used in Linux, and contains diverse functions which microkernel with minimized components does not have. Moreover, through Report Manager, the proposed architecture is able to keep the UAV from falling down and losing connection.

5. Conclusions

In this paper, a secure operating system architecture against communication offense with root exploit is designed. This structure is based on Linux so as to utilize lots of common application and functions for UAV. Simultaneously, the purposed operating system is robust due to keeping attackers from exploiting the well-known vulnerability of Linux, and resilient in that the UAV keeps its control without falling down by connection recovery.

Root exploit, control hijacking, and a crash of UAV might cause sniffing private data, to track confidential information, even to damage human life. This proposed structure is built simply and defenses these critical security issues completely and intuitively.

However, the network communication performance of this structure is inevitably deteriorated. For improving this architecture, we will study on the best way to implement network interfaces, and evaluate the network performance for demonstration.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science, ICT & Future Planning) (No. 2016R1A2B4013118).

References

[1] Z. Birnbaum, A. Dolgikh, V. Skormin, E. O'Brien, D. Muller, and C. Stracquodaine, "Unmanned aerial vehicle security using recursive parameter estimation," *Journal of Intelligent & Robotic Systems*, vol. 84, no. 1-4, pp. 107-120, 2016.

[2] J. Kwak and Y. Sung, "Beacon-based indoor location measurement method to enhanced common chord-based trilateration," *Journal of Information Processing Systems*, vol. 13, no. 6, pp. 1640-1651, 2017.

[3] Z. Birnbaum, A. Dolgikh, V. Skormin, E. O'Brien, D. Muller, and C. Stracquodaine, "Unmanned aerial vehicle security using behavioral profiling," in *Proceedings of 2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, Denver, CO, 2015, pp. 1310-1319.

[4] E. Rivera, R. Baykov, and G. Gu, "A study on unmanned vehicles and cyber security," 2014; http://students.cse.tamu.edu/emy/report.pdf.
[5] A. R. Sobhy, R. A. Sadek, and A. Hashad, "Secure routing in UAV," *International Journal of Computer Science and Information Security*, vol. 11, no. 6, pp. 109-122, 2013.
[6] D. Stiawan, M. Y. B. Idris, and A. H. Abdullah, "Penetration testing and network auditing: Linux," *Journal of Information Processing Systems*, vol. 11, no. 1, pp. 104-115, 2015.
[7] Common Vulnerabilities and Exposures, "CVE-2015-0235 (Ghost)," https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2015-0235.
[8] Common Weakness Enumeration, "CWE-416: Use After Free," 2019; https://cwe.mitre.org/data/definitions/416.html.
[9] "Amazon IoT operating system FreeRTOS has serious vulnerabilities," 2018; https://blog.360totalsecurity.com/en/amazon-iot-operating-system-freertos-has-serious-vulnerabilities/.
[10] H. Choi, M. Geeves, B. Alsalam, and F. Gonzalez, "Open source computer-vision based guidance system for UAVs on-board decision making," in *Proceedings of 2016 IEEE Aerospace Conference*, Big Sky, MT, 2016, pp. 1-5.
[11] L. C. B. Da Silva, R. M. Bernardo, H. A. De Oliveira, and P. F. Rosa, "Multi-UAV agent-based coordination for persistent surveillance with dynamic priorities," in *Proceedings of 2017 International Conference on Military Technologies (ICMT)*, Brno, Czech Republic, 2017, pp. 765-771.
[12] F. Yu, G. Chen, N. Fan, Y. Song, and L. Zhu, "Autonomous flight control law for an indoor UAV quadrotor," in *Proceedings of the 29th Chinese Control and Decision Conference (CCDC)*, Chongqing, China, 2017, pp. 6767-6771.
[13] S. H. VanderLeest, "The open source, formally-proven seL4 microkernel: considerations for use in avionics," in *Proceedings of 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, Sacramento, CA, 2016, pp. 1-9.
[14] A. Lackorzynski and A. Warg, "Demo abstract: timing aware hardware virtualization on the L4Re microkernel systems," in Proceedings of *2016 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, Vienna, Austria, 2016, pp. 1-1.
[15] X. Wang, R. Habeeb, X. Ou, S. Amaravadi, J. Hatcliff, M. Mizuno, M. Neilsen, S. Raj Rajagopalan, and S. Varadarajan, "Enhanced security of building automation systems through microkernel-based controller platforms," in *Proceedings of 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, Atlanta, GA, 2017, pp. 37-44.

KwangMin Koo **https://orcid.org/0000-0003-3484-4095**

He is with the Department of Computer Software and Engineering from Hanyang University as a M.S. candidate since March 2017. His current research interests include deep learning, big data processing, delay tolerant network, and blockchain.

Woo-yeob Lee **https://orcid.org/0000-0001-9688-6691**

He received his B.E. and M.E. degrees in Computer Science and Engineering from Hanyang University. He is with the Department of Computer Science and Engineering from Hanyang University as a Ph.D. candidate, Seoul, Korea in 2009. He is currently pursuing the Ph.D. degree in computer and software at Hanyang University, Seoul, Korea. His current research interests include 5G, Internet of Things, CPS, LPWA and reinforcement learning.

Sung-Ryung Cho **https://orcid.org/0000-0002-2884-6054**

He received the B.S. degree in electrical engineering and computer science from Kookmin University, Seoul, Korea, in 2012. He is currently pursuing the Ph.D. degree in computer and software at Hanyang University, Seoul, Korea. His research interests include convex optimization, probability theory, wireless sensor network, energy harvesting and wireless-powered communication networks.

Inwhee Joe **https://orcid.org/0000-0002-8435-0395**

He received his Ph.D. degree in Electrical and Computer Engineering from Georgia Institute of Technology, Atlanta, GA in 1998. Since 2002, he has been a faculty member in the Department of Computer Science at Hanyang University, Seoul, Korea. His current research interests include Internet of Things, 5G cellular systems, LPWA, DTN, embedded system, security and machine learning.