TECHNICAL SESSION 3 (14:30~15:45)

• Usability of Software Weakness Discovery based on the Binary File Visualization
/ Keon-Ho Park, Sang-Hoon Choi, Chul-Woo Kim, Ki-Woong Park
/ Sejong University, Pace University / Korea

• Education Measurement and Academic Planning Case Study of Information and Production Technology Management Department
/ Hathairat Ketmaneechairat, Rathartip Jaijing, Anusorn Kalapakdee, Kannika Kaewthong
/ King Mongkut's University of Technology North Bangkok / Thailand

• Detecting Ransomware with File System-Awareness Scheme in Cloud Computing Environment
/ Woo-Jin Jeon, Sang-Hoon Choi, Ki-Woong Park
/ Sejong University / Korea

• Integration of Curvelet transform and Probabilistic principal component analysis for detection of Alzheimer's disease and pathological brain image
/ Debesh Jha, Goo-Rak kwon
/ Chosun University / Korea

• Interactive Dance Performance System Based on Wireless Wearable Gesture Sensing Device
/ Guan-I Lin, Chien-Wen Cheng
/ National Taipei University of Technology / Taiwan

Usability of Software Weakness Discovery based on the Binary File Visualization

Keon-Ho Park, Sang-Hoon Choi, Ki-Woong Park*
SysCore Lab., *Dept. Information Security
Sejong University
Seoul, Korea
neipmelon@gmail.com, csh0052@gmail.com, woongbak@sejong.ac.kr

Chul-Woo Kim
Dept. Computer Science
Pace University
New York, United State
Ck90134n@pace.edu

***Abstract*— Software weaknesses that occur in software's architecture, design, and code are able to allow hacker to gain access to a system or network illegally. It is not easy to prepare everything in spite of knowing the vulnerability in advance through Common Weakness Enumeration (CWE) which is a formal list or dictionary of common software weaknesses. In this paper, we present a framework for software weakness discovery based on the binary visualization and verify its performance. To demonstrate the performance of our framework, we use CWE 190 Integer Overread from Juliet Test Suite C/C++ provided by National Institute of Standards and Technology (NIST). We compile CWE 190 source code in two cases. One is with software weakness. The other is without software weakness. We visualize binary of portable executable (PE) file and train images with convolutional neural network (CNN). In our experiment result, we identify up to 99% accuracy in classifying images what belong to which case with software weakness or without.**

Keywords*— Software Weakness, Visualization, Deep learning, Convolutional Neural Network(CNN)

I. Introduction

Attacks on computer systems begin with vulnerability in the code caused by developer mistakes. According to the security audit conducted by security auditor, Positive Technologies^{TM} in 2016, a serious vulnerability was found in 47% of enterprise systems and among them, vulnerability found in 1999 still exists in the system [2]. The vulnerability is led by software weaknesses. If we eliminate software weakness, we can eliminate quite a lot of vulnerability. The software weakness discovered is continuously released on CWE for reduce the problem of software weakness. Despite this effort, there are many cases which software weakness remains in the software caused by implementation mistake in development process. If the known software weakness remains in the system, it is essential to discovery software weakness in advance because software weakness lead to software vulnerability and prone to attack. There are multiple techniques used to find weakness or vulnerability remaining in the system. Representatively Fuzz testing used for weakness or vulnerability discovery is an automatic method of judging whether software is crashing by inputting random data. The developer wants to discover in advance software weakness using discovery method such as Fuzz. For prevent attacks and to develop safe software, software weakness discovery method is necessarily required. Recently artificial intelligence (AI) technology attracts attention and works on new technologies applying AI have been advanced in various fields. Also, participation of various researchers brings on AI algorithm progress. Especially CNN algorithm of AI shows excellent usability in image data processing and is widely used in the field of processing visualization data.

In this paper, we try to examine the usability of software weakness discovery based on the binary file visualization using CNN. We visualize CWE 190 in two cases which is PE file with software weakness and without software weakness. We train and evaluate two cases' images with CNN. And we will measure the accuracy in classifying whether case has software weakness or not. This paper is organized as follows: Section 2 described related work with software weakness discovery. Section 3 describes the framework. Section 4 describes experiment, Section 5 concludes the paper.

II. Related Work

In this chapter, we introduce related work on vulnerability detection technology based on binary data of PE file. Work mainly focused on detecting vulnerability led by software weakness. Based on the binary data of the PE file, previous works have addressed problem of vulnerability detection by converting machine language to assembly language. Work on detecting vulnerabilities in a static environment [1, 6] is to convert binary data into assembly language and find symbolic execution region on assembly language. Vulnerabilities such as overflow require verification of input values, so it is effective that find symbolic execution for vulnerability detection. In addition, work [3, 4] combining the above work with Fuzz technology was conducted. These work, to detect vulnerability, looks for regions where vulnerability may occur by static analysis and inputs random values to regions. This is on the improvement from previous Fuzz technology.

As in the framework presented in this paper, there is a case used in the malware classification study [5] to visualize and analyze the binary data. By distinguishing the type of malware through binary file visualization it is possible to classify malware as visualization data without conversion process of assembly language.

*: Corresponding Author (Ki-Woong Park, woongbak@sejong.ac.kr)
This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP)
(NRF-2017R1C1B2003957)

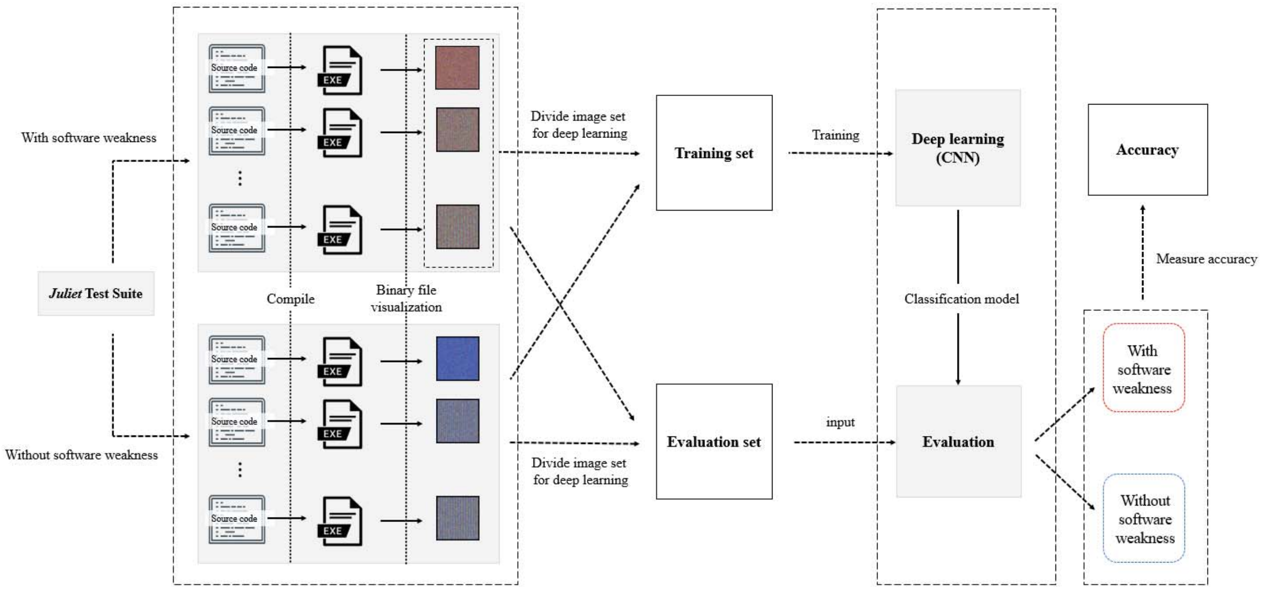

Fig. 1. Framework for Software Weakness Discovery over the Binary File Visualization

III. Framework for Software Weakness Discovery

The software weakness discovery framework is as shown in *Fig 1*. This framework visualizes binary data of PE file. And this framework trains and evaluates images with CNN to classify each case. First, we use two case source code which include algorithm for doing same work. One is source code with software weakness, the other is without software weakness. At that time, the entire binary data of each case maximizes the similar structure to make it not to be classified into areas other than secure coding. And we compiled with each case code respectively. Then this convert image using binary data of the PE file through the same visualization process.

In this paper, to verify usability of software weakness discovery based on the binary file visualization, we visualize all binary data of each PE file without optional selecting. The encoding scheme from binary file to image is as shown in *Fig 2*. In the process, the size of the image is fixed and generated at 256 by 256. Since the PE file of the experiment data does not exceed 140 KB, all binary value can be expressed in the image fixed size. After reading the entire binary data of each compiled sample by 3 bytes, we map each 3 bytes to RGB and draw a pixel in order. In this process, if the binary does not exist, the value of the RGB parameter is fixed at 255.

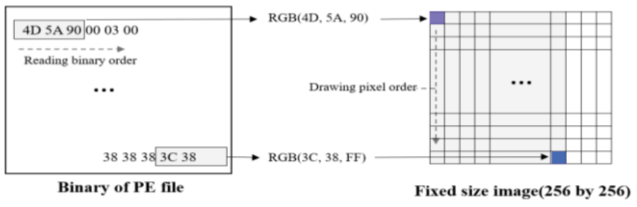

Fig. 2. Encoding Scheme from Binary File to Image

IV. Experiment

In this experiment, we create experiment samples by using CWE 190 Integer Overread from *Juliet* Test Suite C/C++ which is provided by NIST. This software weakness occurs when an integer value is read without validation. These source codes have more than one good function without software weakness and one bad function with software weakness. We compile two times respectively. First is that we create good case PE file that does not have software weakness compiled with good function in source code. Second is that we create bad case PE file that contains software weakness compiled with bad function in source code. To make the binary file structure of each case similarly, when we create good cases, we use only the first calling good function. As a result, we create 1,031 good case and 1,293 bad case.

We measure the accuracy of proposed framework using experiment samples. In our experiment result, this framework classified Good case and Bad case of evaluation set with 99% accuracy based on the binary file visualization. It means system can discover software weakness with 99% accuracy.

V. Conclusion

In this paper, we verify usability of software weakness discovery based on the binary file visualization using CWE 190 and CNN deep learning algorithm. In this paper, Experiment proved that proposed framework can discover software weakness with 99% accuracy. So, we confirmed that our approach is effective based on proof. Furthermore, we plan to verify the framework using all CWE data from *Juliet* Test Suite C/C++. We expect this work is not only worth to try, but also possible to usable in the filed for discovering software weakness.

References

[1] Cova, Marco, et al. "Static detection of vulnerabilities in x86 executables." Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual. IEEE, 2006.

[2] Positive Technologies “Industrial Control Systems 2016 Report: Connected and Vulnerable” http://blog.ptsecurity.com/2016/10/industrial-control-systems-2016-report.html, 2016

[3] Lanzi, Andrea, et al. "A smart fuzzer for x86 executables." Software Engineering for Secure Systems, 2007. SESS'07: ICSE Workshops 2007. Third International Workshop on. IEEE, 2007.

[4] Liu, Guang-Hong, et al. "Vulnerability analysis for x86 executables using genetic algorithm and fuzzing." Convergence and Hybrid Information Technology, 2008. ICCIT'08. Third International Conference on. Vol. 2. IEEE, 2008.

[5] Seok, Seonhee, and Howon Kim. "Visualized Malware Classification Based-on Convolutional Neural Network." Journal of the Korea Institute of Information Security and Cryptology 26.1 (2016): 197-208.

[6] Wang, Tielei, et al. "IntScope: Automatically Detecting Integer Overflow Vulnerability in X86 Binary Using Symbolic Execution." NDSS. 2009.

[7] *Juliet* Test Suite http://samate.nist.gov/SRD/tesuite.php