Advanced Science and Technology Letters
Vol.64 (Security 2014), pp.47-50
http://dx.doi.org/10.14257/astl.2014.64.12

J2EE Server Security Vulnerabilities Analysis Based on Java Access Permission Checking Tree

HyoSeong Park[1], YoungChan Lim[1], ChulWoo Park[1], KiChang Kim[1]JaeHyun Park[1]

[1] Information and Communication Engineering Department, Inha University
{ mongsiry013@hanmail.net, ycwithyou@naver.com, adolmn@naver.com,
kchang@inha.ac.kr, jhyun@inha.ac.kr }

Abstract. Java is being used in various places due to the fact that it can run in any platform as long as the platform is equipped with Java Virtual Machine. This strength of Java provides high potential for IoT (Internet of Things) devices. This paper suggests the use of Java Access Permission Checking Tree in the analysis of Java security and shows this technique is useful not only for the analysis of Java Virtual Machine security but also for the analysis of J2EE server vulnerabilities.

Keywords: Java Security, Java Access Permission Checking Tree, J2EE, Vulnerability

1 Introduction

Java is an interpreted language that uses Virtual Machine, and has the strength of being able to run in various platforms. Since it is designed in the beginning to work well with distributed environment, it shows high performance in especially networking area. This strength shows the possibility that Java can play major role in IoT platforms also [1].

One of the most important problems in the presence of the large number of clients and servers in IoT environment would be the security. Botnet is one of such security problems. We are already observing botnets in mobile network and we should be ready for IoT botnets. It has been reported that the vulnerabilities of Java have been exploited to build botnets [2].

Recent Java applet malwares try to compromise Java Virtual Machine Security Manager and obtain root permission [3]. The same principle can be applied to obtain root permission against Java server machine [4]. Compromising Java server machine is more serious because it can affect the numerous number of clients including PC, mobile devices, or embedded devices.

Java Security Manager is the core mechanism that protects Java Virtual Machine [5][6], and Java Access Permission Checking Tree is a tool to classify and analyze Java malwares that try to bypass it [7]. Java Access Permission Checking Tree focuses on the vulnerabilities of Java Core API. This paper shows Java Access Permission Checking Tree is versatile enough to analyze and classify the vulnerabilities in Java external libraries.

ISSN: 2287-1233 ASTL

The rest of the paper is as follows. Section 2 summarizes previous researches. Section 3 explains the adaptation of Java Access Permission Checking Tree to analyze the vulnerabilities of Java server machine. Finally Section 4 will provide a conclusion.

2 Java Access Permission Checking Tree

2.1 Bypassing Security Manager

Internet browsers activate the Security Manger before they run Java Applets. The Security Manager protects the local system from the applet. Java class can be classified into three classes : User Class, System Class, and Restricted Class. User Class cannot load Restricted Class when the Security Manager is active. System Class is one that can load Restricted Class in the presence of active Security Manager. A User Class can load a System Class and execute a method of it. If this method loads a Restricted Class and returns it, we can have a situation where a User Class effectively loads a Restricted Class. Normally the Security Manager is supposed to detect this kind of situation, but some vulnerable System Classes bypasses the detection of the Security Manager.

The Security Manager blocks User Class from loading Restricted Classes and at the same time prevents execution of methods of Restricted Class. Therefore in order to bypass the Security Manager successfully the attacker should obtain access permission to the target method after loading the Restricted Class. Obtaining access permission to target methods again exploits the vulnerable System Classes.

There is another way to bypass the Security Manager other than exploiting the vulnerabilities of the System Classes. The classes within directories designated as class path are recognized as secure classes by the Security Manager [6] and therefore these classes has the permission to access Restricted Classes. If a User Class can access these classes, it can bypass the Security Manager again by loading Restricted Classes.

2.2 Java Access Permission Checking Tree

Java Access Permission Checking Tree consists of two parts. The first part (Fig.1) shows the access path to Restricted Classes, and the second part (Fig. 2) shows the access path to restricted methods.

Fig.1 shows "forName()" method of "Class" class as a typical way of accessing Restricted Classes. The purpose of loading a Restricted Class is to use its method, therefore we can access a Restricted Class by obtaining the field of a System Class. The "getField()" and "getFields()" methods are used to obtain the fields of "Class" class. Fig.2 shows three ways of obtaining the methods of Restricted Classes: getMethod(), getDeclaredMethod(), and using lookup Class.

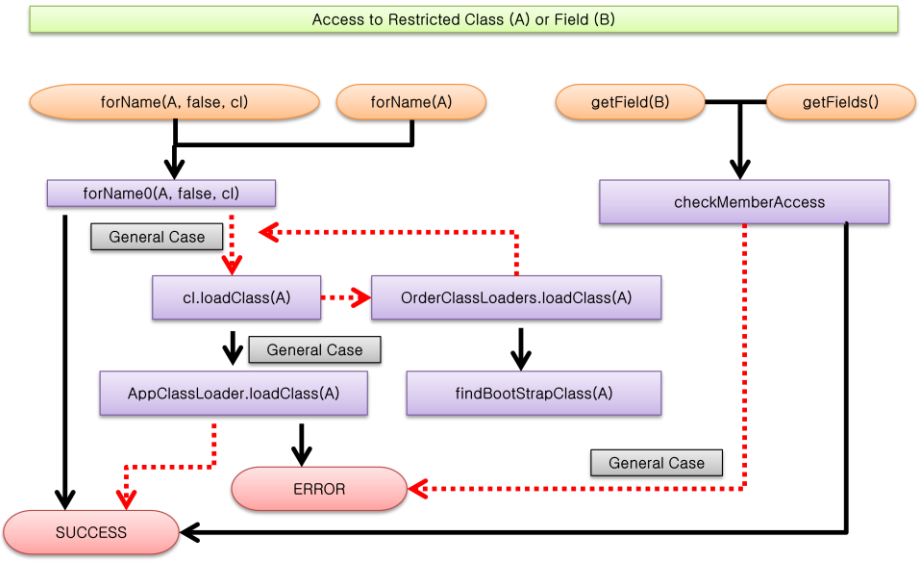

Fig.1. Java Class Access Permission Checking Tree

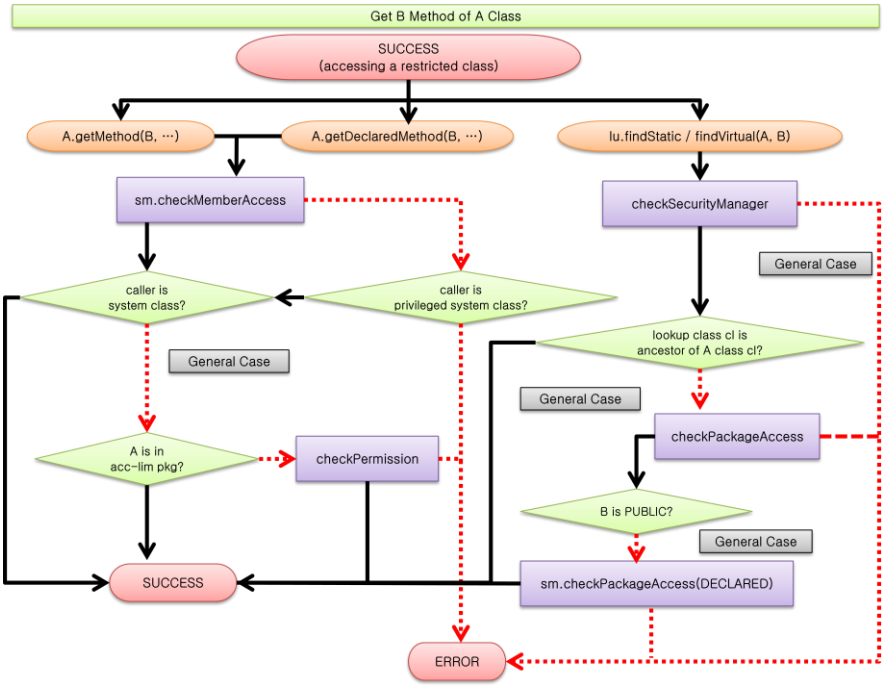

Fig.2. Java Method Access Permission Checking Tree

3 Conclusion

Java Core API security principle also applies to external libraries with system-level authorities. Therefore we need to extend the power of Java Access Permission Checking Tree to the external libraries as well as Core API when analyzing Java vulnerabilities. We also need to extend vulnerability analysis to the Java server considering the services provided by Cloud Java systems. This paper adds checking field path in the Java Access Permission Checking Tree for this purpose and shows an example of malware that uses external libraries.

5 Reference

1. Oracle, http://www.oracle.com/us/solutions/internetofthings
2. F-Secure, Threat Report H2, 2012
3. Security Explorations, “Security Vulnerabilities in Java SE Technical Report”, 2012
4. Security Explorations, “Security Vulnerability Notice – Security vulnerabilities in Oracle Java Cloud Service”, 2013
5. Wallach, Dan S., and Edward W.Felten. Understanding Java stack inspection." Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium on. IEEE, 1998
6. Scott Oaks, “Java Security 2nd Edition”, O’REILLY’, pp.21-112, May 24, 2001
7. Chan Lim, Hyo-Seong Park, Chul-Woo Park, “Java Security Vulnerability Analysis Based on Java Access Permission Checking Tree”, Advanced Science and Technology Letters Vol.58 (Security 2014), pp.52-55, 2014

