SECURITY OF THE JAVA EXECUTION ENVIRONMENT IN E-BUSINESS APPLICATIONS FOR SMEs

Radu BUCEA-MANEA-ŢONIŞ, Lecturer Ph.D. Cd.
radumanea.mk@spiruharet.ro
Rocsana BUCEA-MANEA-ŢONIŞ, Assistant lecturer Ph.D. Cd.
rocsanamanea.mk@spiruharet.ro
Facultaty of Financial-Accounting Management
Spiru Haret University

Abstract

Web space is currently inherent in any business activity, from SMEs to international holdings. In this context security of e-business applications becomes an issue intensively studied by the scientific literature.

Currently the Java programming language is the most popular among software developers. The paper studies new elements of security, mainly for the Java environment. Other topics studied in this article refer to attack targets, fundamental threats against computer systems, means of facilitating security attacks and Java execution security environment. The case study demonstrates the impossibility of using a Java apllet over web to access files and folders structure on the host systems. Using Java Web environment is the security strength of e-business applications.

Key-words: *Applet, Java, informational security, masquerading, IDE*

JEL Classification: L_{86}

Introduction

Web space is currently inherent in any business activity, from SMEs to international holdings. In the management activity, in the decision making process, in the communication with the customers or shareholders, the company uses web-based solutions. In this context, the security of e-business applications is a subject extensively studied by the literature.

The Java programming language is most popular among software developers. (www_1) The article brings new elements regarding security, mainly for the Java environment. Other topics studied in this article refer to the targets of the informatic attacks, fundamental threats against computer systems, means of facilitating security attacks and Java execution environment. The case study demonstrates the impossibility of using a Java web apllet to access files and folders on the host system. Using Java Web environment is a strength in the security of e-business applications.

Attacks may be aimed at specific targets or may be random, that target-of-opportunity strike randomly. Illegal access to a computer system represents a positioning of the attacker which makes possible other offenses, too.

Attacks potential on the computer systems are influenced by the following factors:

1) business process automation;
2) profitability of successful attacks;
3) access to remote procedures and methods;
4) difficulty in identifying and undertaking computer crime.

Classic scenarios of illegal computer activities focus on:

– the public exposure of confidential data and information;

– changing data and transmitting altered data to beneficiaries;

– blocking or unduly delaying legitimate users access to data.

The possibility that computer systems are insufficiently protected against certain risks or loss is called system risk. Risk is the sum of threats, vulnerabilities and value of information displayed:

Risk = Threats + Vulnerabilities + Information value

A suggestive representation of information systems security concepts and relations between them is proposed in the standard *Common Criteria for Information Technology Security Evaluation.*

Fundamental threats against computer systems can be classified into:

– ***information disclosure*** – the act of accessing and disseminating confidential information by unauthorized persons;

– ***repudiation*** – a person's ability to deny the sender's identity or content of a message or date of sending the information;

– ***denial of service*** – attacker uses thousands of remotely controlled computers to automate transmission of unsolicited data that will flood the target system;

– ***illegitimate use of data and information***.

Means of facilitating attacks are [according to Vasiu & all, 2007]:

– ***masquerading*** – the process by which an attacker assumes the identity of an authorized user. IP spoofing is how the attacker pretends to use an already known trusted IP computer to exploit the communication in order to gain privileged information or running programs (the attacker claims to be a trustworthy person known by the user);

– ***computer contaminants*** – malicious programs or scripts able to alter, erase, record or transmit data that allow the remote access to computer resources, able to alter the normal operation of a computer system without consent of the owner:

 Trojans – software with hidden features, unacceptable in terms of the user, may insert or alter data, may format disks, may intercept passwords, may remote lock, and after the malicious action they destruct itself. Examples of trojans: *droppers* that install viruses on a victim system, *injectors* that access memory placing malicious instructions, *germs* – generated by viruses,

 Logic bombs – malicious instructions from a program designed to create conditions for a computer attack;

 Viruses (from Lat. Virus = poison) – inject their own message information in an existing program that will replicate the action repeatedly, have the ability to perform unauthorized actions on behalf of the host. Can be grouped into self-replicable macros, hidden viruses (stealth) which hide the actual size of host files avoiding integrity checks on disk, encrypted viruses that hide the virus code and polymorphic viruses;

 Backdoors – mechanism for violation restrictions on disk access or write to breach confidentiality or placing Trojans;

 Spyware – malicious software placed on a host system without the knowledge of the owner to obtain vital information about the system or to capture information entered by the keyboard.

Applications with the source code unhidden are safer than the proprietary applications, because they allow to identify vulnerabilities not only for hackers, but also for specialists from the organizations that can provide a remedy. Applications requiring a high level of security are:

– views and browsers that allow you to access remote data and may allow execution of macro commands malware;

– programs run only the super-user root;

– daemons;

– CGI scripts;

– Java applets;

– setuid/setgid type applications.

Security of Java execution environment

SE Java Platform is a dynamic architecture with expandable security, based on standards and interoperability. Security features of the platform include cryptography, authentication and authorization, public key infrastructure and others. Java security model is based on custom modules “sandbox” in which Java software programs can run safely, without potential risks for system or user [www_3].

Creating and implementing a Java program involves editing a source file with the extension *.java*. It is processed using a Java compiler (javac from the jdk package) into a bytecode file with the extension *.class*. It contains instructions in Java machine code and is the fundamental unit of Java implementation. The program can be executed within the execution environment of Java Runtime Environment (JRE), using Java interpreter from jdk package. JRE includes the Java class libraries (packages) and Java virtual machine (JVM).

The language was created for software development in a heterogeneous network environment. As one of the original goals of the language was to be used in embedded systems with a minimum memory source, Java is designed to be small and use a small amount of hardware [Yellin, 2010].

There are two distinct Java programs:

1. *Stand-alone Java applications* contain a main function which is executed through the Java interpreter; such applications have access to all system resources.

2. *Java applets* are executable programs within a web browser; an applet is loaded with html page via html tag APPLET:

```
<APPLET
CODE = clasaApplet
WIDTH = latimeInPixeli
HEIGHT = inaltimeInPixeli
[ARCHIVE = arhiva.jar]
[CODEBASE = URLApplet]
[ALT = textAlternativ]
[NAME = numeInstantaApplet]
[ALIGN = aliniere]
[VSPACE = spatiuVertical]
[HSPACE = spatiuOrizontal] >
[< PARAM NAME = parametru1 VALUE = valoare1 >]
[< PARAM NAME = parametru2 VALUE = valoare2 >]
...
[text HTML alternativ]
</APPLET>
```

Enrichment Web browser functions through Java applets involve a series of threats to client system security [Patricius & all, 2001]. *A Java applet code is in most cases an uncertain code, coming from an unreliable source.* An applet could write a considerable file size, could read confidential data or might launch virus-type programs.

Attacks on the client host via applets can be:

– changing the file system, execution or cessation programs, memory access and control processes during execution;

– reading the file with passwords for machine UNIX type reading the global system variables or accessing inputs from different user interfaces;

– using the printer or using the host to access other resources shared in network, the use of resident services on the host site for masquerading;

– denial of service attacks by creating threads with highest priority to monopolize all the processing capability of machine cycles, memory allocation block work overload or disk available;

– other repetitive attacks, beeps, opening windows avoided by closing the web browser.

Restrictions being placed on an applet usually concern the following:

– access to the host file system;

– network connection (socket functions and URLStream);

– accessing graphics functions – untrusted window;

– calculation system properties (user.name, user.home, user.dir, java.class.path);

– processes and threads.

JVM launches an exception type SwecurityException whenever one of these restrictions are violated, thus blocking the respective thread.

The general structure of an applet is as follows:

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
public class StructuraApplet extends Applet {
public void init() {
}
public void start() {
}
public void stop() {
}
public void destroy() {} }
```

Methods *init() start(), stop()* and *destroy()* are called automatically by the browser and are not called explicitly in the program.

Case study – Java applet

Building a Java NetBeans apllet [www_2] that accesses the password file on the root disk. Steps are:

Step 1. Open a new project and choose its type:

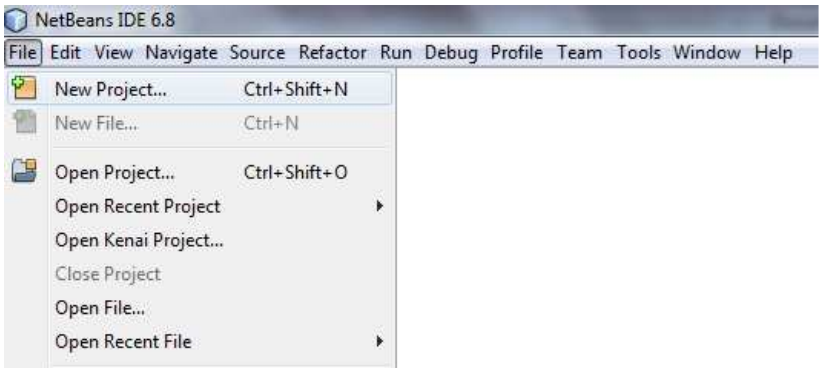

Fig. 1. *Integrated Development Environment (IDE) NetBeans*

Step 2. Writing the source code:

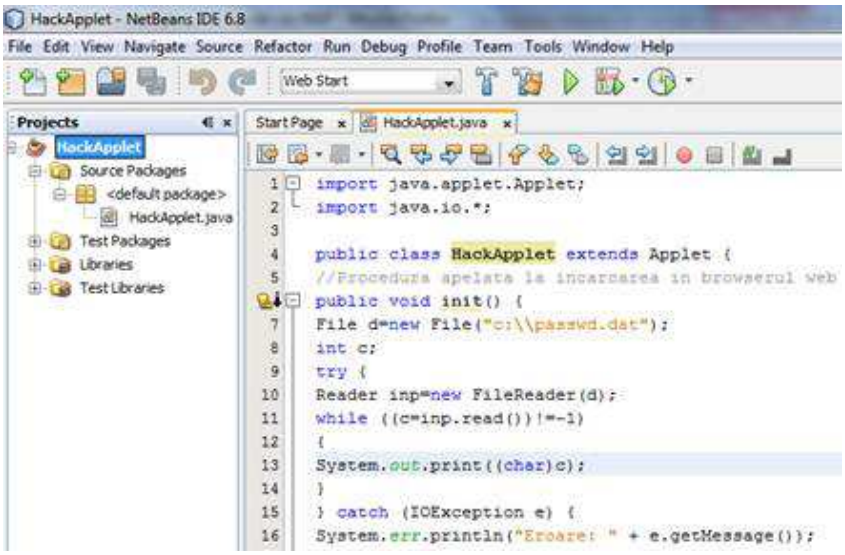

Fig. 2. *Code editor window and browser solutions*

Step 3. Running project development environment and find AlbacaZapada password, as shown in fig. 3:

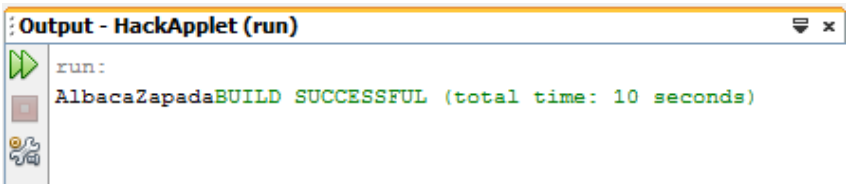

Fig. 3. *Java applet result in NetBeans environment*

Step 4. Trying to appeal apllet from the browser the system returns the following error:

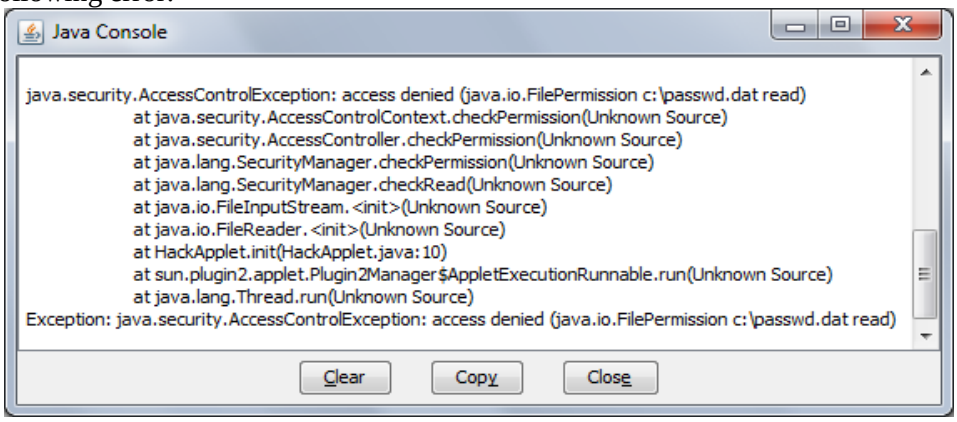

Fig. 4. *Consola Java with exception error message*

The exception is launched when trying to open the file *passwd.dat*. FileReader constructor that takes as parameter the file descriptor *d*, calls the method checkRead of the Security Manager installed by the browser. This method is possible to check the file and starts reading the above exception. In the worst-case-scenario the checkRead can be overridden in order to access the system, so an applet cannot create or install your own security manager.

Conclusions

In conclusion, the safest method of configuration browser against malicious applets is disabling Java option. This is done in Internet Explorer by opening the configuration window from the Options submenu of Tools menu. In the dialog box the user has to select the Security tab where he has to check the option Disable Java Programs.

REFERENCES

- V.V. Patriciu, M. Ene-Pietrosanu, I. Bică, C. Văduva, N. Voicu, *Securitatea comerţului electronic*, ALL Publishing House, Bucharest, 2001.
- I. Vasiu, L. Vasiu, *Afaceri electronice, aspecte legale, tehnice şi manageriale*, Albastra Publishing House, Cluj-Napoca, 2007.
- Frank Yellin, Low Level Security in Java, 2010, http://74.125.77.132/search?q= cache: aRGtJknE2NwJ:www.w3.org/Conferences/WWW4/Papers/197/40.html +java+security+papers&cd=10&hl=ro&ct=clnk&gl=ro&client=firefox-a.
- *Study: Java still top programming language*, 2008. http://news.cnet.com/8301-13505_3-10009669-16.html.
- *Sursa mediului integrat de dezvoltare NetBeans*, 2010. http://netbeans.org/.
- *Oracle Sun Developer Network (SDN)*, 2010 http://java.sun.com/javase/technologies/security/index.jsp.