Security Vulnerability Notice

SE-2012-01-ORACLE-6

[Security vulnerabilities in Java SE, Issue 50]

2011 Security Explorations. All Rights Reserved.

DISLAIMER

INFORMATION PROVIDED IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW NEITHER SECURITY EXPLORATIONS, ITS LICENSORS OR AFFILIATES, NOR THE COPYRIGHT HOLDERS MAKE ANY REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OR THAT THE INFORMATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS, OR OTHER RIGHTS. THERE IS NO WARRANTY BY SECURITY EXPLORATIONS OR BY ANY OTHER PARTY THAT THE INFORMATION CONTAINED IN THE THIS DOCUMENT WILL MEET YOUR REQUIREMENTS OR THAT IT WILL BE ERROR-FREE.

YOU ASSUME ALL RESPONSIBILITY AND RISK FOR THE SELECTION AND USE OF THE INFORMATION TO ACHIEVE YOUR INTENDED RESULTS AND FOR THE INSTALLATION, USE, AND RESULTS OBTAINED FROM IT.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL SECURITY EXPLORATIONS, ITS EMPLOYEES OR LICENSORS OR AFFILIATES BE LIABLE FOR ANY LOST PROFITS, REVENUE, SALES, DATA, OR COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, PROPERTY DAMAGE, PERSONAL INJURY, INTERRUPTION OF BUSINESS, LOSS OF BUSINESS INFORMATION, OR FOR ANY SPECIAL, DIRECT, INDIRECT, INCIDENTAL, ECONOMIC, COVER, PUNITIVE, SPECIAL, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND WHETHER ARISING UNDER CONTRACT, TORT, NEGLIGENCE, OR OTHER THEORY OF LIABILITY ARISING OUT OF THE USE OF OR INABILITY TO USE THE INFORMATION CONTAINED IN THIS DOCUMENT, EVEN IF SECURITY EXPLORATIONS OR ITS LICENSORS OR AFFILIATES ARE ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS.

2011 Security Explorations. All Rights Reserved.

Security Explorations discovered a critical security issue in Java SE Platform, Standard Edition. It is caused by insecure implementation of a serialization functionality implemented by `com.sun.corba.se.impl.io.ObjectStreamClass` class. A table below, presents its technical summary:

ISSUE #	TECHNICAL DETAILS	
50	origin	`com.sun.corba.se.impl.io.ObjectStreamClass`
	cause	the possibility to use same `serialPersistentFields` value for classes with incompatible fields layout
	impact	arbitrary type confusion condition
	type	complete security bypass vulnerability

The problem with `com.sun.corba.se.impl.io.ObjectStreamClass` class stems from the fact that it can reuse information about class fields layout declared in an incompatible class and cached by the `translateFields` method in a `translatedFields` hashtable. For the purpose of serialization, classes can explicitly declare, which fields take part in this process. This is done by the means of a proper `serialPersistentFields` value declaration. For `com.sun.corba.se.impl.io.ObjectStreamClass` class, if two different classes use the same reference for the value of their `serialPersistentFields`, the layout of the fields declared by that table will be the same for both classes. This will be the case regardless of the fact that both classes could be actually incompatible (the types of corresponding fields in the layout could be different) and that the names declared by a shared `serialPersistentFields` value may not actually denote a valid field from a target class. As a result, it is possible to use a different `fieldID` (object field offset) value of `ObjectStreamField` class instance as an argument to insecure `putObject` method of `sun.corba.Bridge` class. This can directly lead to arbitrary type confusion condition.

It should be also mentioned that there is another class (`com.sun.corba.se.impl.orbutil.ObjectStreamClass_1_3_1`) that has the same, vulnerable implementation regarding the handling of cached `serialPersistentFields` values as the one described above. This second class does not seem to be however used.

Issue 50 was tested in the environment of latest versions of Java SE 5, 6 and 7 software. We verified that it can be successfully used to achieve a complete JVM sandbox bypass in a target system.

Attached to this report, there is a Proof of Concept codes that illustrates this. It has been successfully tested in a fully patched Windows 7 OS environment and with Java SE 5 Update 22 (build 1.5.0_22-b03), Java SE 6 Update 35 (build 1.6.0_35-b10), Java SE 7 Update 7 (build 1.7.0_07-b10).

About Security Explorations

2011 Security Explorations. All Rights Reserved.

Security Explorations (http://www.security-explorations.com) is a security start-up company from Poland, providing various services in the area of security and vulnerability research. The company came to life in a result of a true passion of its founder for breaking security of things and analyzing software for security defects. Adam Gowdiak is the company's founder and its CEO. Adam is an experienced Java Virtual Machine hacker, with over 50 security issues uncovered in the Java technology over the recent years. He is also the hacking contest co-winner and the man who has put Microsoft Windows to its knees (vide MS03-026). He was also the first one to present successful and widespread attack against mobile Java platform in 2004.

2011 Security Explorations. All Rights Reserved.