Abysssec Research

1) Advisory information

Title	: Adobe Acrobat and Reader 'newfunction' Remote Code Execution Vulnerability
Version	: Adobe Reader 9.3.2
Analysis	: http://www.abysssec.com
Vendor	: http://www.adobe.com
Impact	: Critical
Contact	: shahin [at] abysssec.com , info [at] abysssec.com
Twitter	: @abysssec
CVE	: CVE-2010-2168

2) Vulnerable version

S.u.S.E. SUSE Linux Enterprise Desktop 11 SP1
+ Linux kernel 2.6.5
S.u.S.E. SUSE Linux Enterprise Desktop 11
S.u.S.E. SUSE Linux Enterprise Desktop 10 SP3
S.u.S.E. openSUSE 11.2
S.u.S.E. openSUSE 11.1
S.u.S.E. openSUSE 11.0
RedHat Enterprise Linux WS Extras 4
RedHat Enterprise Linux Supplementary 5 server
RedHat Enterprise Linux Extras 4
RedHat Enterprise Linux ES Extras 4
RedHat Enterprise Linux Desktop Supplementary 5 client
RedHat Enterprise Linux AS Extras 4
RedHat Desktop Extras 4
Adobe Reader 9.3.2
Adobe Reader 9.3.1
Adobe Reader 9.1.3
Adobe Reader 9.1.2
Adobe Reader 9.1.1

Adobe Reader 8.2.2
Adobe Reader 8.2.1
Adobe Reader 8.1.7
Adobe Reader 8.1.6
Adobe Reader 8.1.5
Adobe Reader 8.1.4
Adobe Reader 8.1.3
Adobe Reader 8.1.2
Adobe Reader 8.1.1
Adobe Reader 7.1.4
Adobe Reader 7.1.3
Adobe Reader 7.1.2
Adobe Reader 7.1.1
Adobe Reader 7.0.9
Adobe Reader 7.0.8
Adobe Reader 7.0.7
Adobe Reader 7.0.6
Adobe Reader 7.0.5
Adobe Reader 7.0.4
Adobe Reader 7.0.3
Adobe Reader 7.0.2
Adobe Reader 7.0.1
Adobe Reader 7.0
Adobe Reader 9.3
Adobe Reader 9.2
Adobe Reader 9.1
Adobe Reader 9
Adobe Reader 8.2
Adobe Reader 8.1.2 Security Update
Adobe Reader 8.1
Adobe Reader 8.0
Adobe Reader 7.1
Adobe Acrobat Standard 9.3.2
Adobe Acrobat Standard 9.3.1
Adobe Acrobat Standard 9.1.3
Adobe Acrobat Standard 9.1.2
Adobe Acrobat Standard 8.2.2
Adobe Acrobat Standard 8.2.1
Adobe Acrobat Standard 8.1.7
Adobe Acrobat Standard 8.1.6
Adobe Acrobat Standard 8.1.4
Adobe Acrobat Standard 8.1.3
Adobe Acrobat Standard 8.1.2
Adobe Acrobat Standard 8.1.1
Adobe Acrobat Standard 7.1.4
Adobe Acrobat Standard 7.1.3
Adobe Acrobat Standard 7.1.1
Adobe Acrobat Standard 7.0.8

Adobe Acrobat Standard 7.0.7
Adobe Acrobat Standard 7.0.6
Adobe Acrobat Standard 7.0.5
Adobe Acrobat Standard 7.0.4
Adobe Acrobat Standard 7.0.3
Adobe Acrobat Standard 7.0.2
Adobe Acrobat Standard 7.0.1
Adobe Acrobat Standard 7.0
Adobe Acrobat Standard 9.3
Adobe Acrobat Standard 9.2
Adobe Acrobat Standard 9.1
Adobe Acrobat Standard 9
Adobe Acrobat Standard 8.2
Adobe Acrobat Standard 8.1
Adobe Acrobat Standard 8.0
Adobe Acrobat Standard 7.1
Adobe Acrobat Professional 9.3.2
Adobe Acrobat Professional 9.3.1
Adobe Acrobat Professional 9.1.3
Adobe Acrobat Professional 9.1.2
Adobe Acrobat Professional 8.2.2
Adobe Acrobat Professional 8.2.1
Adobe Acrobat Professional 8.1.7
Adobe Acrobat Professional 8.1.6
Adobe Acrobat Professional 8.1.4
Adobe Acrobat Professional 8.1.3
Adobe Acrobat Professional 8.1.2
Adobe Acrobat Professional 8.1.1
Adobe Acrobat Professional 7.1.4
Adobe Acrobat Professional 7.1.3
Adobe Acrobat Professional 7.1.1
Adobe Acrobat Professional 7.0.9
Adobe Acrobat Professional 7.0.8
Adobe Acrobat Professional 7.0.7
Adobe Acrobat Professional 7.0.6
Adobe Acrobat Professional 7.0.5
Adobe Acrobat Professional 7.0.4
Adobe Acrobat Professional 7.0.3
Adobe Acrobat Professional 7.0.2
Adobe Acrobat Professional 7.0.1
Adobe Acrobat Professional 7.0
Adobe Acrobat Professional 9.3
Adobe Acrobat Professional 9.2
Adobe Acrobat Professional 9.1
Adobe Acrobat Professional 9
Adobe Acrobat Professional 8.2
Adobe Acrobat Professional 8.1.2 Security Updat
Adobe Acrobat Professional 8.1

Adobe Acrobat Professional 8.0
Adobe Acrobat Professional 7.1
Adobe Acrobat Professional 6.0
Adobe Acrobat 9.3.2
Adobe Acrobat 9.3.1
Adobe Acrobat 9.1.1
Adobe Acrobat 8.2.2
Adobe Acrobat 7.0.9
Adobe Acrobat 7.0.3
Adobe Acrobat 7.0.2
Adobe Acrobat 7.0.1
Adobe Acrobat 7.0
Adobe Acrobat 6.0.5
Adobe Acrobat 6.0.4
Adobe Acrobat 6.0.3
Adobe Acrobat 6.0.2
Adobe Acrobat 6.0.1
Adobe Acrobat 6.0
Adobe Acrobat 9.3
Adobe Acrobat 9.2

3) Vulnerability information

Class

1- Code execution

Impact

Attackers can exploit this issue to execute arbitrary code or cause denial-of-service conditions.

Remotely Exploitable

Yes

Locally Exploitable

Yes

4) Vulnerabilities detail

authplay.dll is responsible for processing flash contents in pdf files. Through processing of the newfunction(bytecode 0x40) command it faces some problem because of memory corruption error.

By running newfunction command, an object of the new function is created. This command takes an argument. The value of this argument is an index from method_info structure.(for further information about this command refer to ActionScript Virtual Machine 2 (AVM2) Overview.

Here is part of the code in the sub_30292F10 function that process this command:

```
.text:30242D54          lea     eax, [esp+18h+arg_4] ; jumptable 30242ACB cases 60,64
.text:30242D58          push    eax
.text:30242D59          call    sub_301C82B0
.text:30242D5E          mov     ecx, [esp+1Ch+arg_10]
.text:30242D62          mov     edx, [ecx+78h]
.text:30242D65          mov     ebx, [edx+eax*4]
.text:30242D68          mov     edi, [esp+1Ch+arg_0]
.text:30242D6C          add     esp, 4
```

At the beginning of this code sub_301C82B0 is called. This function takes a pointer to the buffer that contains newfunction command as an argument:

```
.text:301C82B0          push    esi
.text:301C82B1          mov     esi, [esp+4+arg_0]
.text:301C82B5          mov     ecx, [esi]
.text:301C82B7          movzx   eax, byte ptr [ecx]
.text:301C82BA          test    al, al
.text:301C82BC          js      short loc_301C82C3
.text:301C82BE          inc     ecx
.text:301C82BF          mov     [esi], ecx
.text:301C82C1          pop     esi
.text:301C82C2          retn
.text:301C82C3
.text:301C82C3 loc_301C82C3:                    ; CODE XREF: sub_301C82B0+Cj
.text:301C82C3          movzx   edx, byte ptr [ecx+1]
.text:301C82C7          shl     edx, 7
```

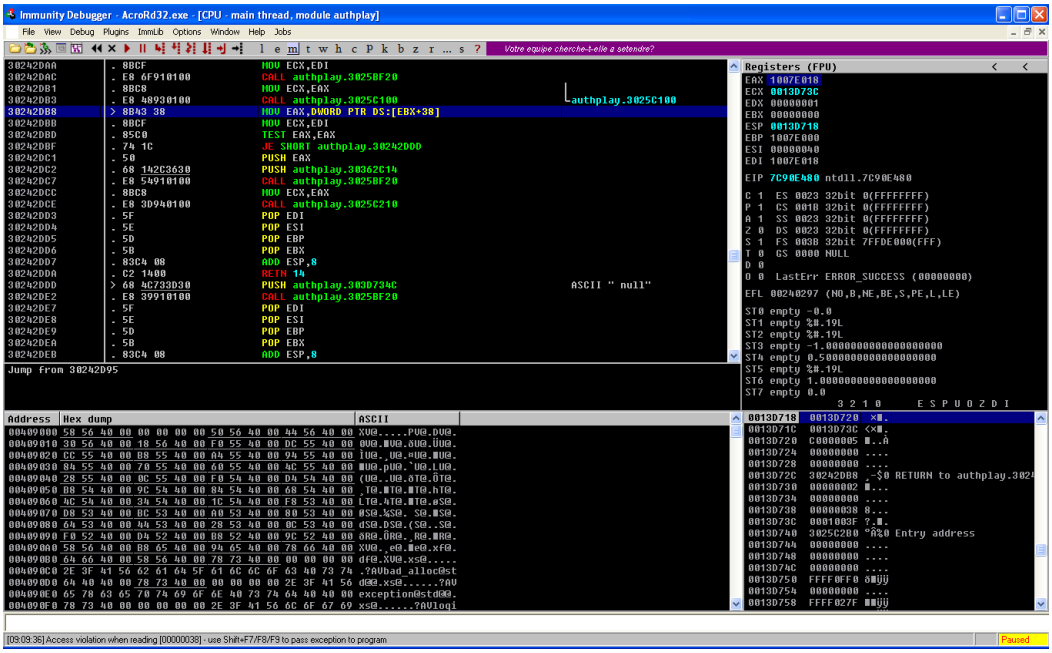
```
.text:301C82CA          and     eax, 7Fh
.text:301C82CD          or      edx, eax
.text:301C82CF          test    edx, 4000h
.text:301C82D5          jnz     short loc_301C82E0
.text:301C82D7          add     ecx, 2
.text:301C82DA          mov     [esi], ecx
.text:301C82DC          mov     eax, edx
.text:301C82DE          pop     esi
.text:301C82DF          retn
....
```

In this function the first byte after bytecode 40 which is equal to newfunction command is read. If it is greater than zero the next bytes also will be read. The value of the second byte is multiplied by 128 and added with the value of the first byte. If the result is greater than 16384 it will go on the third byte. This process is continued until the fifth bye after bytecode 0x40.

There problem here is not properly checking the value. sub_301C82B0 function return the above result. After executing the sub_301C82B0 function remaining code will be followed in sub_30292F10 function. then value of edx is added to the return value of sub_301C82B0 function and is stored in a buffer.

The value that is stored in the buffer and is under our control is used in the next instructions that can cause memory corruption.

```
...
.text:30242D92          cmp     esi, 44h
.text:30242D95          jnz     short loc_30242DB8
.text:30242D97          lea     ecx, [esp+18h+arg_4]
.text:30242D9B          push    ecx
.text:30242D9C          call    sub_301C82B0
.text:30242DB8          mov     eax, [ebx+38h]
.text:30242DBB          mov     ecx, edi
.text:30242DBD          test    eax, eax
.text:30242DBF          jz      short loc_30242DDD
.text:30242DC1          push    eax
.text:30242DC2          push    offset asc_30362C14 ; " "
.text:30242DC7          call    sub_3025BF20
.text:30242DCC          mov     ecx, eax
.text:30242DCE          call    sub_3025C210
```



Exploit

Exploiting this bug is difficult but possible because the DEP (permanent) in Adobe Reader. According to the above explanation I will present the way of exploitation.

As we discussed sub_301C82B0 function return some controllable value:

```
.text:30242DF5          push    edx
.text:30242DF6          call    sub_301C82B0
.text:30242DFB          mov     ecx, [esp+1Ch+arg_10]
.text:30242DFF          mov     edx, [ecx+9Ch]
.text:30242E05          mov     eax, [edx+eax*4]
```

We should set values after bytecode 0x40 which in result the return value of sub_301C82B0 and finally result of [edx+eax*4] expression direct us to our controllable code. Then take the advantages of other codes that use this value to gain control of the program. After gaining control of the execution we should take the stack and bypassing the DEP by implementing the ROP method to execute the shellcode.