DEEPSEC

NOVEMBER 19, 2015, VIENNA

YES, NOW *YOU* CAN PATCH THAT VULNERABILITY TOO!

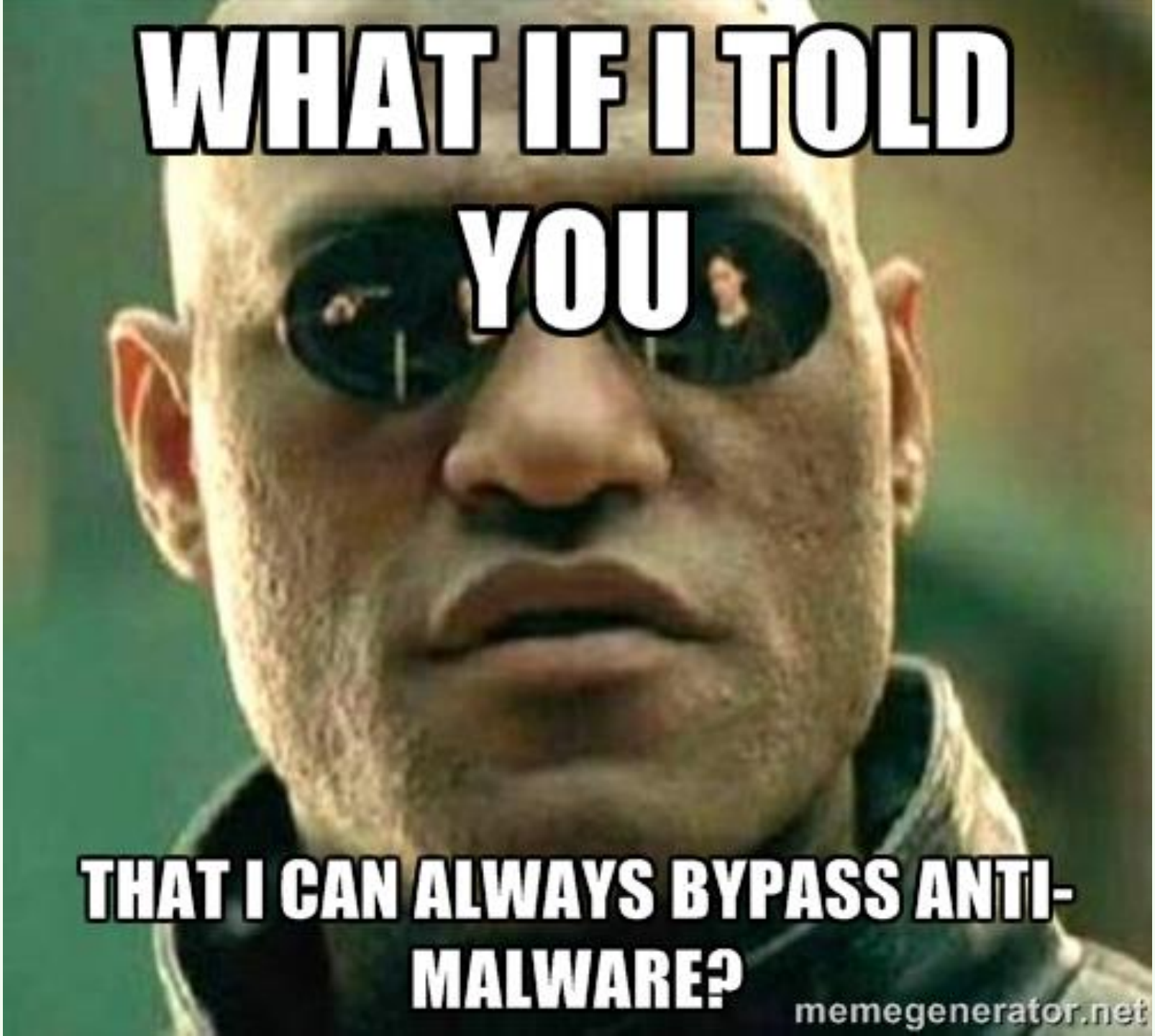
Mitja Kolšek, CEO, ACROS Security & 0patch co-founder

THE STATE OF AFFAIRS

- 15 years of finding and reporting vulnerabilities
- The same types of bugs again and again
- New exploit mitigations, new bypasses
- Few vendors proactively look for vulnerabilities
- Critical security fixes are not being applied
- Feels like being a problem instead of a solution
- Nobody is happy (except the attackers)


VULNERABILITIES
VULNERABILITIES EVERYWHERE


ONE DOES NOT
SIMPLY
APPLY A VENDOR
UPDATE
memegenerator.n


WHAT IF I TOLD YOU
THAT I CAN ALWAYS BYPASS ANTI-MALWARE?
memegenerator.net

I CAN BREAK INTO ANY CORP NETWORK (BUT I SHOULDN‘T BE ABLE TO)

1. Pick any browser/reader/player vulnerability with a public PoC younger than 2 months
2. Prepare an exploit
3. Mutate the exploit until VirusTotal doesn‘t detect it any more
4. Phish until you‘re in

PATCH


FOUND A NASTY VULNERABILITY
NOT SURE WHAT TO DO WITH IT
memegenerator.net

RESEARCHER‘S DILEMMA

1. **Privately report to vendor**
   (and risk anger, silence or lawsuit)
2. **Publish**
   (and risk anger or lawsuit)
3. **Sell it**
   (and risk prosecution)
4. **Shelve it**
   (what‘s the point of your work then?)

PATCH

PATCHING
IS A HARD PROBLEM

A HARD PROBLEM

SOFTWARE VENDORS

- monopoly on patching
- direct and opportunity costs
- deploying fixes is costly
- have better things to do

USERS

- hate downtime
- updating = risk breakage
- not updating = risk ownage

SECURITY RESEARCHERS

- constant conflict with vendors
- considered part of the problem

PATCH

REINVENTING
SOFTWARE PATCHING

REINVENTING SOFTWARE PATCHING

Take **less than a minute** to install
a **small piece of software** that
will apply **tiny security patches**
in the **same way** for all applications.

Then apply and remove patches **instantly**
without disturbing users or admins.

PATCH

DEMO

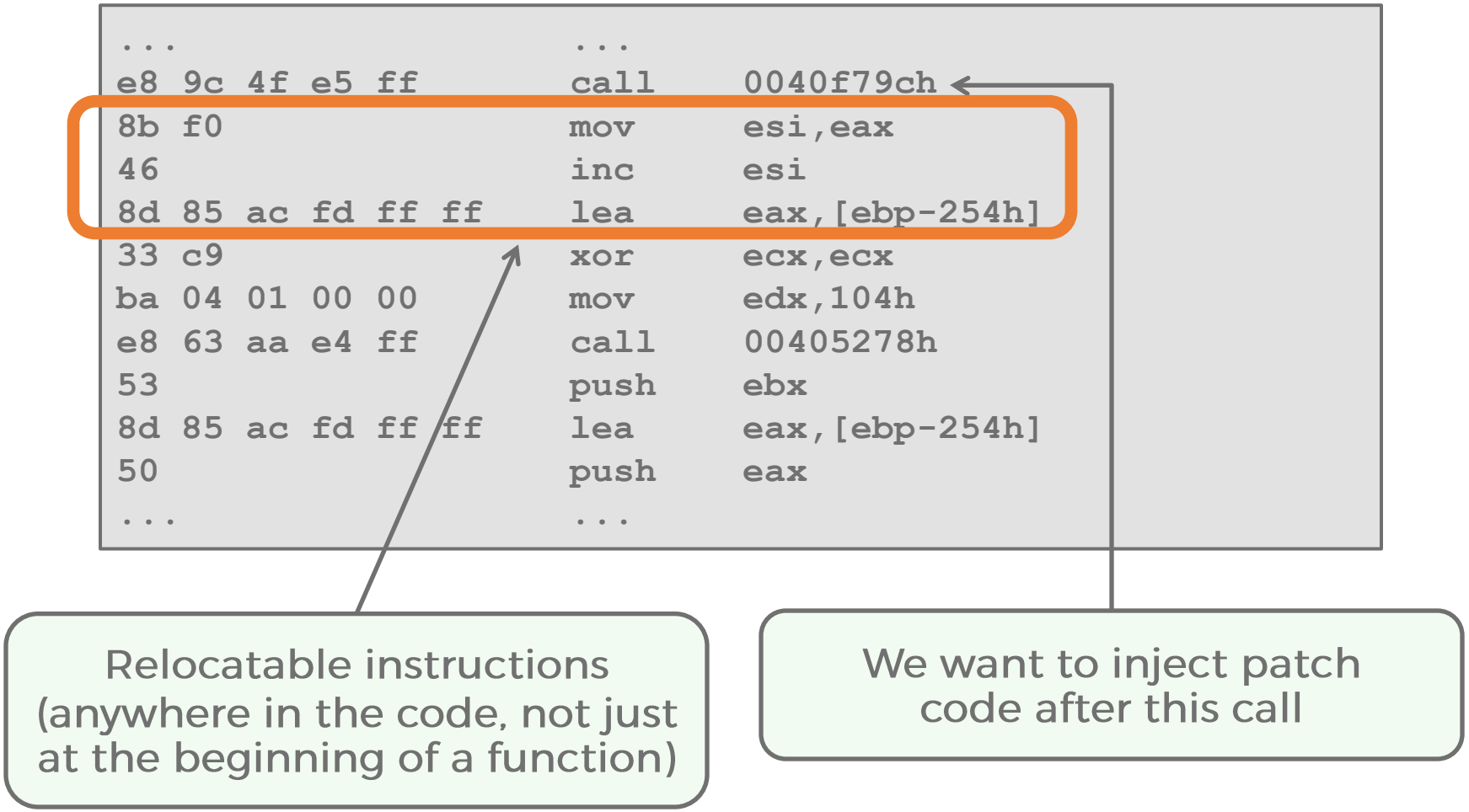
NO REBOOT, NO RELAUNCH

PATCH

TECHNOLOGY

HOW IT WORKS

FUNCTION HOOKING ON STEROIDS

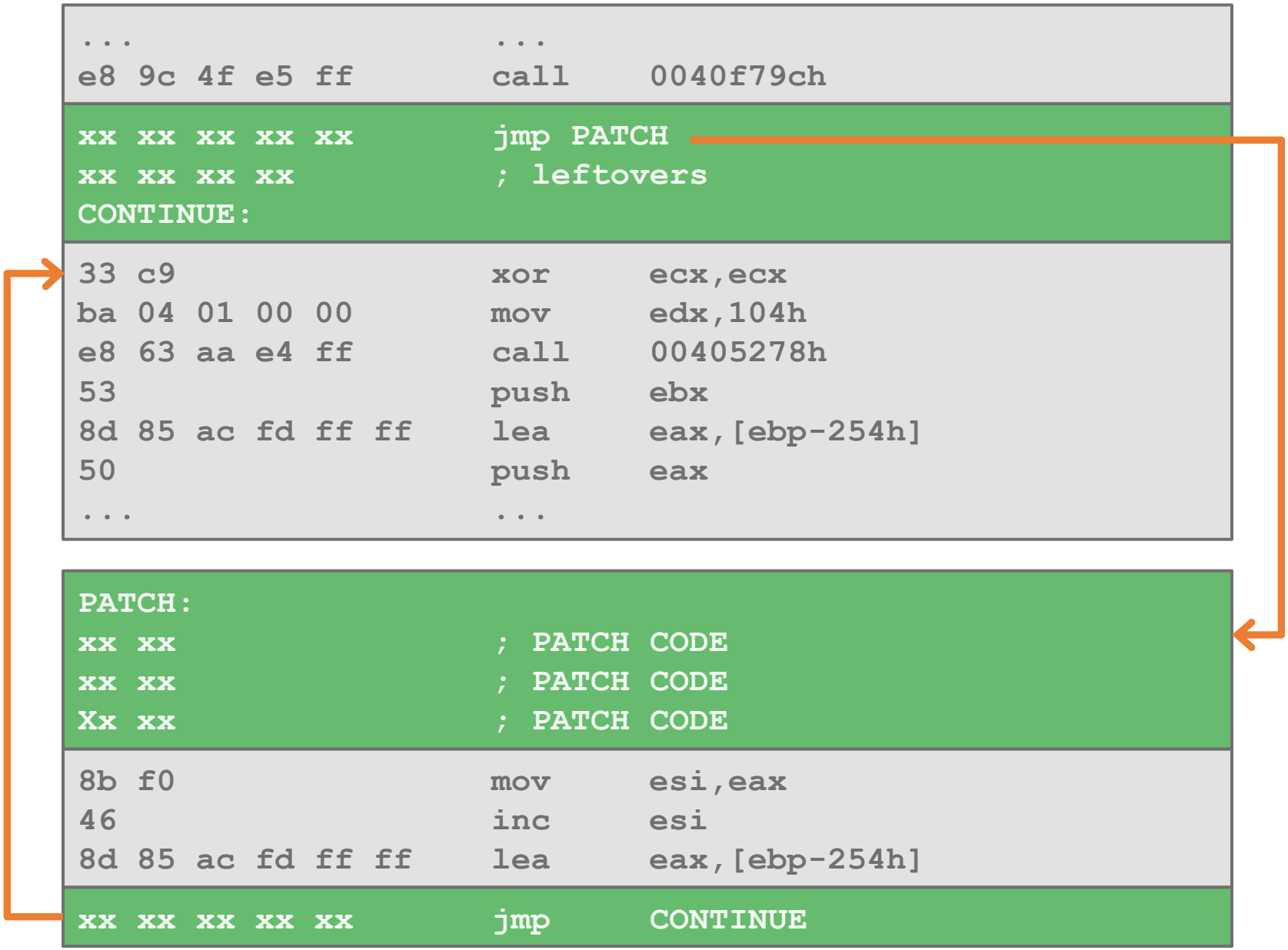


PATCH

```
...                     ...
e8 9c 4f e5 ff          call    0040f79ch
```

```
8b f0                   mov     esi,eax
46                      inc     esi
8d 85 ac fd ff ff       lea     eax,[ebp-254h]
```

```
33 c9                   xor     ecx,ecx
ba 04 01 00 00          mov     edx,104h
e8 63 aa e4 ff          call    00405278h
53                      push    ebx
8d 85 ac fd ff ff       lea     eax,[ebp-254h]
50                      push    eax
...                     ...
```


...
e8 9c 4f e5 ff
call 0040f79ch
xx xx xx xx xx
xx xx xx xx
CONTINUE:
jmp PATCH
; leftovers
33 c9
ba 04 01 00 00
e8 63 aa e4 ff
53
8d 85 ac fd ff ff
50
...
xor ecx,ecx
mov edx,104h
call 00405278h
push ebx
lea eax,[ebp-254h]
push eax
...
PATCH:
xx xx
xx xx
Xx xx
; PATCH CODE
; PATCH CODE
; PATCH CODE
8b f0
46
8d 85 ac fd ff ff
mov esi,eax
inc esi
lea eax,[ebp-254h]
xx xx xx xx xx
jmp CONTINUE

DEFINING A PATCH

1. Module hash
2. Offset of the patch inside the module
3. Patch code

PATCH SOURCE CODE

```
MODULE_PATH "C:\vulnerable_app\app.exe"
PATCH_ID 87235
VULN_ID 993

patchlet_start

 PATCHLET_ID 1
 PATCHLET_OFFSET 0x0000b979
 N_ORIGINALBYTES 5

 code_start
   xor eax, eax
 code_end

patchlet_end
```

WHAT CAN BE PATCHED

- Unchecked buffers
- Numeric over/underflows
- Use after free
- Double free
- Uninitialized variables
- Format strings
- Binary planting / DLL injection
- Data patching
- (many others)

WHAT CAN'T BE PATCHED
(or not that easily)

- Scripted (to-be-compiled) code
- Design flaws
- Windows kernel (PatchGuard)
- Apps that actively refuse to be patched

MICROSCOPIC CURES FOR BIG SECURITY HOLES

PATCHING DEMO

INTEGER OVERFLOW

CVE-2011-2371
Firefox 3.6.16 ReduceRight() Integer Overflow

```
array_extra(JSContext *cx, ArrayExtraMode mode, uintN argc,
            jsval *vp)
{
    JSObject *obj;
    jsuint length, newlen;
    jsval *argv, *elemroot, *invokevp, *sp;
    JSBool ok, cond, hole;
    JSObject *callable, *thisp, *newarr;
    jsint start, end, step, i;
    void *mark;

    obj = JS_THIS_OBJECT(cx, vp);
    if (!obj || !js_GetLengthProperty(cx, obj, &length))
        return JS_FALSE;

    switch (mode) {
      case REDUCE_RIGHT:
        start = length - 1, end = -1, step = -1;
        /* FALL THROUGH */
}
```

PATCH

CVE-2011-2371
Firefox 3.6.16 ReduceRight() Integer Overflow

```
<html>
 <body>
  <script>
  foo = new Array;
  foo.length = 0x80100000
  foo.reduceRight(function(){}, 1)
  </script>
 </body>
</html>
```

PATCH

CVE-2011-2371
Firefox 3.6.16 ReduceRight() Integer Overflow

```
array_extra(JSContext *cx, ArrayExtraMode mode, uintN argc,
            jsval *vp)
{
    JSObject *obj;
    jsuint length, newlen;
    jsval *argv, *elemroot, *invokevp, *sp;
    JSBool ok, cond, hole;
    JSObject *callable, *thisp, *newarr;
    jsint start, end, step, i;
    void *mark;

    obj = JS_THIS_OBJECT(cx, vp);
    if (!obj || !js_GetLengthProperty(cx, obj, &length))
        return JS_FALSE;

    switch (mode) {
      case REDUCE_RIGHT:
        start = length - 1, end = -1, step = -1;
        /* FALL THROUGH */
}
```

PATCH

CVE-2011-2371
Firefox 3.6.16 ReduceRight() Integer Overflow

```
array_extra(JSContext *cx, ArrayExtraMode mode, uintN argc,
```

```
6b6ab96b 56              push    esi
6b6ab96c 8d7c241c        lea     edi,[esp+1Ch]
6b6ab970 894c242c        mov     dword ptr [esp+2Ch],ecx
6b6ab974 e807240000      call    js_GetLengthProperty
6b6ab979 83c404          add     esp,4
6b6ab97c 85c0            test    eax,eax
6b6ab97e 0f84b1ce0a00    je      "return JS_FALSE"
```

```
    void *mark;

    obj = JS_THIS_OBJECT(cx, vp);
    if (!obj || !js_GetLengthProperty(cx, obj, &length))
        return JS_FALSE;

    switch (mode) {
      case REDUCE_RIGHT:
        start = length - 1, end = -1, step = -1;
        /* FALL THROUGH */
}
```

PATCH

CVE-2011-2371
Firefox 3.6.16 ReduceRight() Integer Overflow

```
6b6ab96b 56              push    esi
6b6ab96c 8d7c241c        lea     edi,[esp+1Ch]
6b6ab970 894c242c        mov     dword ptr [esp+2Ch],ecx
6b6ab974 e807240000      call    js_GetLengthProperty
6b6ab979 83c404          add     esp,4
6b6ab97c 85c0            test    eax,eax
6b6ab97e 0f84b1ce0a00    je      "return JS_FALSE"
```

After the call, array length is in `dword ptr [edi]`

Suitable bytes for overwriting

PATCH

CVE-2011-2371
Firefox 3.6.16 ReduceRight() Integer Overflow

```
6b6ab96b 56              push    esi
6b6ab96c 8d7c241c        lea     edi,[esp+1Ch]
6b6ab970 894c242c        mov     dword ptr [esp+2Ch],ecx
6b6ab974 e807240000      call    js_GetLengthProperty
6b6ab979                 and     dword ptr [edi],7FFFFFFFh
6b6ab979 83c404          add     esp,4
6b6ab97c 85c0            test    eax,eax
6b6ab97e 0f84b1ce0a00    je      "return JS_FALSE"
```

We reset the top bit and keep the length below MAX_INT

Relative offset from start of module js3250.dll = B979h

PATCH

CVE-2011-2371
Firefox 3.6.16 ReduceRight() Integer Overflow

```
6b6ab96b 56                 push    esi
6b6ab96c 8d7c241c           lea     edi,[esp+1Ch]
6b6ab970 894c242c           mov     dword ptr [esp+2Ch],ecx
6b6ab974 e807240000         call    js_GetLengthProperty

6b6ab979                    cmp     dword ptr [edi],7FFFFFFFh
                            jbe     DONE
                            and     dword ptr [edi],7FFFFFFFh
                            call    PIT_ExploitBlocked
DONE:

6b6ab979 83c404             add     esp,4
6b6ab97c 85c0               test    eax,eax
6b6ab97e 0f84b1ce0a00       je      "return JS_FALSE"
```

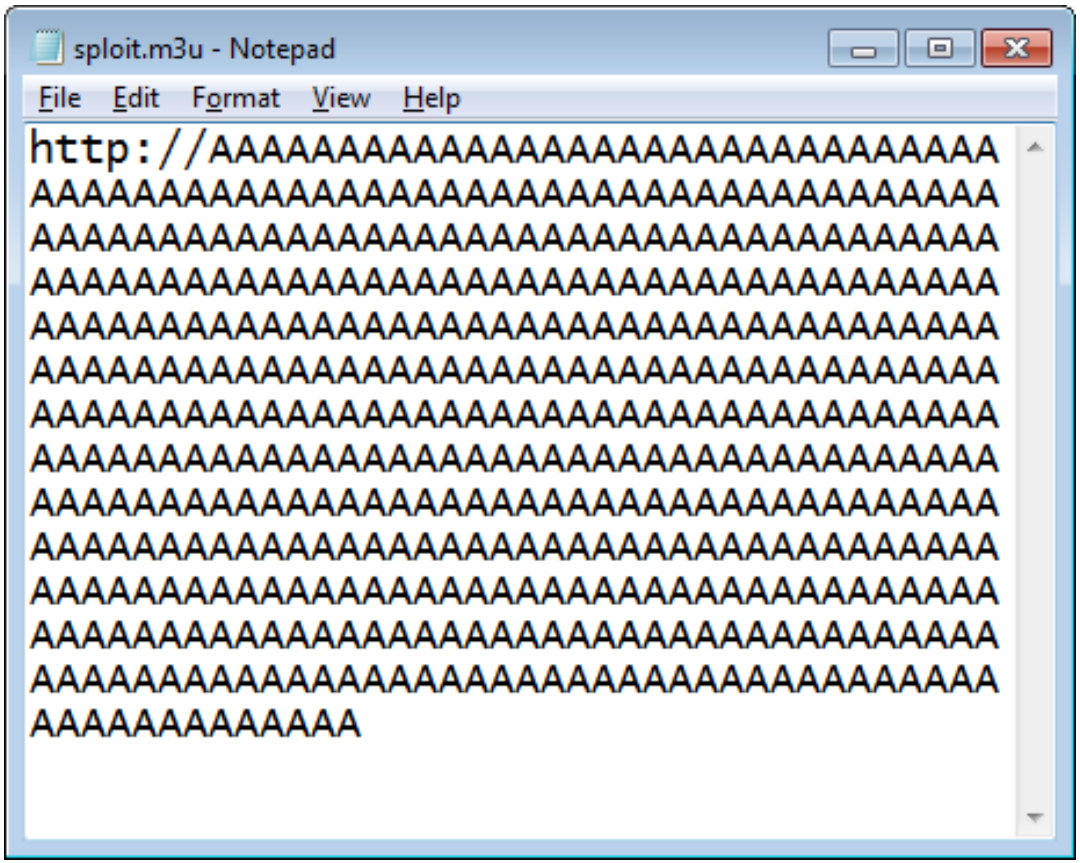
We want to display a warning to the user

PATCH

PATCHING DEMO

BUFFER OVERFLOW

CVE-2013-7409
AllPlayer 5.8 Buffer Overflow In .M3U File



PATCH

CVE-2013-7409
AllPlayer 5.8 Buffer Overflow In .M3U File

```
005ba7fa 53                push    ebx
                           ; ebx points to source buffer (line)
005ba7fb e89c4fe5ff        call    kernel32!lstrlenW
                           ; eax is the length of the line
005ba800 8bf0              mov     esi,eax
005ba802 46                inc     esi
                           ; esi is the length of the line + 1
005ba803 8d85acfdffff      lea     eax,[ebp-254h]
005ba809 33c9              xor     ecx,ecx
005ba80b ba04010000        mov     edx,104h
005ba810 e863aae4ff        call    zero-ize_destination_buffer
005ba815 53                push    ebx
                           ; ebx points to source buffer (line)
005ba816 8d85acfdffff      lea     eax,[ebp-254h]
                           ; eax points to destination buffer
                           ; which only has 104h bytes on stack
005ba81c 50                push    eax
005ba81d e8624fe5ff        call    kernel32!lstrcpyW
```

Suitable bytes for overwriting

We want to shorten source buffer before this call

PATCH

CVE-2013-7409
AllPlayer 5.8 Buffer Overflow In .M3U File

```
005ba815 53              push    ebx
                         ; ebx points to source buffer (line)
005ba816 8d85acfdffff    lea     eax,[ebp-254h]
                         ; eax points to destination buffer
                         ; which only has 104h bytes on stack
005ba81c 50              push    eax
005ba81d e8624fe5ff      call    kernel32!lstrcpyW
```

PATCH

CVE-2013-7409
AllPlayer 5.8 Buffer Overflow In .M3U File

```
005ba815 53                push    ebx
                           ; ebx points to source buffer (line)

005ba816                   cmp esi,104h     ; esi is line length + 1
                           jbe DONE
                           mov word ptr [ebx+208h],0
                           call PIT_ExploitBlocked
DONE:

005ba816 8d85acfdffff      lea     eax,[ebp-254h]
                           ; eax points to destination buffer
                           ; which only has 104h bytes on stack
005ba81c 50                push    eax
005ba81d e8624fe5ff        call    kernel32!lstrcpyW
```

We cut the source buffer short by terminating it with a 0

Relative offset from start of module AllPlayer.exe = 1ba816h

PATCH

GUIDELINES

HOW TO FIX WITHOUT BREAKING

PATCHING GUIDELINES

1. Find a good place for patching
2. Don‘t break anything
3. Change as little code as possible
4. Execute as rarely as possible
5. Test security and functionality

PATCHING GUIDELINES

Find a good place for patching

1

- Cover all vulnerable execution paths
  (but ideally nothing else)
- Relocated original code must not be a target of calls or jumps
- Relocated original code must be easily relocatable
  (nothing that uses relative offsets)

PATCHING GUIDELINES

Don‘t break anything

2

- Make no assumptions about how your patched code can be reached
  (instead, make sure using dissassemblers, code analysis tools)
- Preserve functionality
  (don‘t cut off vulnerable code unless that‘s the only possible solution)
- Make sure there are no side-effects
  (changed registries, changed flags)

PATCHING GUIDELINES

Change as little code as possible

3

- Less code = fewer errors
- Less code = easier reviewing and testing
- Less code = less execution overhead
- Less code = less chance of race condition

PATCHING GUIDELINES

Execute as rarely as possible

4

- Avoid patching inside loops
- Sanitize user input at the beginning of an execution tree to cover all branches

PATCHING GUIDELINES

Test security and functionality

5

- PoCs and exploits should be blocked
- Legitimate use cases of the patched functionality should have exactly the same behavior as before


FOUND A NASTY VULNERABILITY
HECK, I'LL OPATCH IT
memegenerator.net


VULNERABILITIES
VULNERABILITIES EVERYWHERE


OPATCHES
OPATCHES EVERYWHERE
memeg

BETA ACCOUNTS

SEND YOUR EMAIL AND WE'LL CREATE AN ACCOUNT FOR YOU

support@0patch.com

DEEPSEC
NOVEMBER 19, 2015, VIENNA

0patch.com
@0patch

THANK YOU!

Mitja Kolšek, CEO, ACROS Security & 0patch co-founder
mitja.kolsek@acrossecurity.com