The Secret Life of ActionScript

The year in Flash bugs, exploits and mitigations

Natalie Silvanovich
@natashenka
About me

- Natalie Silvanovich
  AKA natashenka AKA Flashtasha
- Project Zero member
- Previously did mobile security on Android and BlackBerry
- Flash enthusiast
- Reporter of ⅓ of Flash vulnerabilities
My goal

FIND ALL OF THE
FLASH BUGS
My goal

● Bug finding is my top priority
  ○ Mostly code review
  ○ Some fuzzing (with Mateusz Jurczyk AKA j00ru)
  ○ 1 bug per day -> 1 bug per week
  ○ Flash bugs stay gone

● Analyze external bugs and exploits
My goal

● Occasionally exploit bugs to answer questions
  ○ Is exploitation possible?
  ○ Is exploitation reliable?
  ○ How does X impact exploitability

● Work on mitigations (with James Forshaw and Mark Brand)
This talk

● Attack surface
● The year in Flash
  ○ New bugs and bug classes
  ○ 0-days, 1-days and other exploits
  ○ Mitigations
● The future?
Flash is ...

- **AS2 -- ActionScript 2**
  - Interpreted legacy Flash Scripts with own VM
  - Reduced API set
  - Generally more bugs with lower exploitability
  - Blurry boundaries between VM and APIs
Flash is ...

- **AS3 -- ActionScript 3**
  - Modern VM with JIT and interpreter
    - Extendible
    - GC Heap / Fixed Heap
    - Optimized for Flash
  - Open source VM
  - Open and closed source APIs
  - Bugs are less dense but more exploitable
Flash is ...

- Anticorpus
  - Functionality outside of script
  - MP4 parser, zlib, regex, image decoders, etc
Warning

OBJECTS IN MIRROR ARE LARGER THAN THEY APPEAR
March 2015

- One bulletin, 11 bugs, no 0-days
- MP4 and RegEx bugs
- Browser policy bypasses
- Superconstructor bugs
Superconstructor Bugs

- Type confusion in AS2 due to constructor override

AS2 ScriptObject

<table>
<thead>
<tr>
<th>void* native_data</th>
</tr>
</thead>
<tbody>
<tr>
<td>void* destroy_func</td>
</tr>
<tr>
<td>table* properties</td>
</tr>
</tbody>
</table>

SomeObject

<table>
<thead>
<tr>
<th>char* some_prop</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

function delete_SomeObject( void* data) {
  SomeObject s = (SomeObject*) data;
  delete[] some_prop;
}

{ toString : ASFunc, __proto__ : ASObject, ... }

(member types and order not exactly as shown)
Superconstructor Bugs

● Constructor flow
  ○ Fetch __proto__ property and fetch __constructor__ property
  ○ Call constructor on object
    ■ Call super
    ■ Call constructor
  ● Set native_data and destroy_func (optional)
Superconstructor Bugs

```
super();
this.__proto__={};
this.__proto__.__constructor__ = XML;
super("test");
```

**AS2 ScriptObject**
- void* native_data
- void* destroy_func
- table* properties
- ...

**XMLObject**
- char* some_prop
- ...

**Function**
```
function delete_SomeObject( void* data) {
  SomeObject s = (SomeObject*) data;
  delete[] some_prop;
}
```
Timeline

- **3/12 APSB15-05**: 11 bugs
- **4/13 0-day CVE-2015-3043**
- **4/14 APSB15-06**: 22 bugs

Jan
Dec
Nov
Oct
Sep
Aug
Jul
Jun
May
Apr

March 2015

Feb 2016
April 2015

- 0-day in FLV processing (CVE-2015-3043, reported by FireEye, limited Russian APT)
- 21 other bugs
- Many anti-corpus bugs
- First redefinition issue
CVE-2015-3039

- Redefinition issue in ConvolutionFilter (also reported by bilou)
- AS2 allows any method to be redefined in script (monkey-patching)
- Generally native methods accept any type and convert objects with valueOf, toString, object constructor, etc.
var filter = new ConvolutionFilter(...);
var n = { valueOf : ts };
var a = [];
a[0] = n;
filter.matrix = a;
function ts(){
  filter.matrix = [1];
}

```
{{ valueOf : ts }}
```
Timeline

- 3/12 APSB15-05: 11 bugs
- 4/14 APSB15-06: 22 bugs
- 4/13 0-day: CVE-2015-3043
- 5/12 APSB15-09: 18 bugs
May 2015

- 18 bugs fixed, no 0-days
- MP4 issues
- Superconstructor issues (the last)
- The redefinition continues
CVE-2015-3077

- Redefinition issue not involving `valueOf` or `toString`
- Led to perfectly* reliable exploit
var object = mc.createEmptyMovieClip(...);
var filter = new BlurFilter();
object.filters = [filter];
BlurFilter = ConvolutionFilter;
var f = object.filters;
var d = f[0]
June 2015

- Another FLV 0-day (CVE-2015-3113, reported by FireEye, Chinese)
- Several reports similar to past 0-days (FLV and shader)
- First SharedObject issue
var s = SharedObject.getLocal("test");
...
var q = {myprop :"natalie"};
s.data.fpadInfo = q;
s.flush();
...
var n = new NetConnection
n.connect.call(s.data, "");
s = 1;

AS2 ScriptObject
void* native_data
void* destroy_func
table* properties
...

{ data : { fpadinfo : { myprop : "natalie" }}}

AS2 ScriptObject

void* shared
...

...
July 2015

- Hacking Team dump contained two 0-days and two fixed bugs
  - ByteArray/OpaqueBackground -- 0-day UaFs due to valueOf redefinition (CVE-2015-0349 and CVE-2015-05122)
  - ConvolutionFilter issue shown earlier (CVE-2015-3039/CVE-2015-0349)
  - Integer overflow in Function.apply -- reported via Chromium VRP before use (CVE-2015-0387)
  - NULL pointer in BitmapData, not exploitable (CVE-2015-05123)
July 2015

- `valueOf/toString` bugs receive increased attention
  - Many similar bugs reported in next few months
  - Adobe starts efforts to pre-emptively fix similar bugs
- 33 bugs in regular update
- Vector mitigations implemented
Vector Mitigation

"I don't afraid Adobe analysts at all" -- Vitaly Toropov

- Adds checksums to Vectors that are checked before doing sensitive functions
- Some Vectors are also on their own heap page
- Reduced the reusability of exploit code
- Generally increases the quality of bug needed for an exploit
- Substitution of ByteArray or BitmapData is possible, but not as good
CVE-2015-3130

- Redefinition issue involving `valueOf` that’s not a UaF
CVE-2015-3130

```javascript
var s = 1;
var rec_array:Array = new Array();
rec_array.push({name: "john", city: "omaha"});
rec_array.push({name: "bob", city: "omaha"});
rec_array.length = {valueOf : gl};
rec_array.sortOn(['name', 'city']);

function gl(){
  if(s < 3){
    s++;
    return 100000;
  }else{
    return 2;
  }
}
```
Many more bugs similar to HT bugs
MC UaFs pour in
CVE-2015-5550 (MovieClip UaFs)

- Very common AS2 bug, 100+ reported this year
  - Small variety of freed object
- Also works with TextFields
- Root cause is that display fields are freed outside of garbage collection
  - Always, for real, even if there are references (in AS2)
CVE-2015-5550 (MovieClip UaFs)

- Happens when function parameters are converted after local variables are initialized, but before they are used
- Fixed by enforcing convert -> initialize -> use order
var clip1 = this.createEmptyMovieClip("clip1", 1);
var clip2 = this.createEmptyMovieClip("clip2", 2);
var n = {toString: func};
clip1.swapDepths(n);

function func(){
    clip1.removeMovieClip();
    return "clip2";
}

SO *s = GetObject()
MC *m = native_data[10];
September/October 2015

- 23 bugs in September updates and 20 in October
  - Mostly UaFs and other redefinition bugs
- 0-day immediately after October update (reported by TrendMicro, NATO targets)
CVE-2015-7645

- Reported two weeks before it was found in the wild
- Type confusion in serializations, due to weird AVM behaviour
- Two other variants also reported and fixed in emergency patch
- None of these bugs compile
CVE-2015-7645

From the AVM:

```c
// In theory we should reject duplicate slots here;
// in practice we don't, as it causes problems with some existing content
//if (basetb->findBinding(name, ns) != BIND_NONE)
//  toplevel->throwVerifyError(kIllegalOverrideError, toplevel->core()->toErrorString(qn), toplevel->core()->toErrorString(this));
```

tl;dr a method can be overridden by a var

Most natives don’t make assumptions, but some do. Especially interfaces.
class superclass{
    ...
    public function writeExternal(){
        return 1;
    }
}

class subclass extends superclass{
    public var writeExternal:uint = 7;
    ...
}

CVE-2015-7645

From the AVM:

```
Multiname mn(core->getPublicNamespace(t->pool),
    core->internConstantStringLatin1(kWriteExternal));
m_functionBinding = toplevel->getBinding(t, &mn);
```

and later:

```
MethodEnv* method =
    obj->vtable->methods[AvmCore::bindingToMethodId(info->get_functionBinding())];
method->coerceEnter(argc, argv);
```
How was this bug exploited?

- Traits property array is variable-sized
- Corrupted ByteArray to get R/W access to entire memory space
November and December 2015

● Huge Dec update, 79 bugs, mostly MC UaF
  ○ Structural changes to AS2 to make broad fixes

● New mitigations
  ○ Checksumming on ByteArray
  ○ Isolated Heap
  ○ NOP slide mitigations

● Exploit kit 1-day and 0-day
CVE-2015-8446

- 1-day in Angler
- Similar to CVE-2015-5560
- Integer overflow in ID3 allocation
  - Controllable size
  - Controllable overwrite
- Exploited using BitmapData
CVE-2015-8651

- Integer overflow leading to heap overflow in JIT (reported by Huawei)
CVE-2015-8651

- SWF contained two exploits
  - Typical vector exploit
  - Post Isolated Heap exploit including such elements as
    - Long if statements nested almost 100 times
    - Using both a media file and an image to fill heap slots at different points in the exploit
    - Triggering the bug ~600 times
    - Final results was memory space access via ByteArray
Conclusions

- Finding bugs in Flash is generally getting harder
  - 1 bug per day versus 1 per week
- Certain bug classes are drying up, but others are taking their places
- Flash mitigations are making it more difficult to exploit bugs, especially with low-quality bugs
The Future (What’s left?)

- MC UaFs (and AS2) probably still exist, but getting hard to exploit
  - Eventually similar bugs will have marginal utility
  - Display UaFs in AS3?
- Redefinition bugs are no longer ‘deep’
- More AS3 bugs?
The Future (What’s left?)

- More anticorpus bugs / use of anti-corpus?
  - Media (MP4, FLV)
- Open source AVM?
- Platform-specific code
- Flash deprecation
  - Browsers?
Thank You

● Adobe
Questions?

http://googleprojectzero.blogspot.com/
@natashenka
natalie@natashenka.ca