Introduction Speaker

❖ Stefan Horlacher
  • Founder of Arcus Security
    o Penetration Testing
    o Web Application Reviews
    o Remediation of vulnerabilities
    o www.arcus-security.ch
Agenda

- Foreword
- Introduction
- Sandboxes
- Local Shared Object
- Configuration Overview
- Client Side Configuration
- Embedding Flash
- Cross Domain Policies
- Common Vulnerabilities
- Questions
Foreword

- This presentation summarises security relevant material from a client side perspective
- A lot of public knowledge can be found on
  - Adobe web site
  - OWASP web site
Introduction
Introduction

✧ SWF pronounced
  • Swif
✧ Dialect of ECMAScript
  • JavaScript alike
✧ ActionScript
  • 2000 ActionScript 1.0
  • 2003 ActionScript 2.0
  • 2006 ActionScript 3.0
Introduction

✧ Purposes
  - Animations
  - Games
  - Rich Internet Applications (RIA)
  - ...
Files and execution environments

- Browser Plug-ins
  - Plays SWF files
  - Embedded in <object> / <embed> tag
  - Direct call to the SWF file
    - Browser Plug-in creates the DOM
  - Plug-ins use the browser’s SSL / TLS implementation
  - Byte Code (JIT Compiler)
Introduction

✧ Files and execution environments
  • Standalone player
    o Plays SWF files
    o SSL / TLS available
  • Projector
    o .exe files
    o Standalone player containing the SWF
      • Specific version
    o No SSL / TLS
    o Creation not supported in Standalone player anymore
Introduction

Cookies

- HttpOnly flag prevents Flash from reading the cookie
- Requests send cookies if any are present
  - HttpOnly flag has no influence in this case
Sandboxes
Sandboxes

- local-with-file-system
  - Access to local resources
  - Access to UNC network path

- local-with-networking
  - Access to network resources

- local-trusted
  - Not restricted
Sandboxes

remote

- No access to local file system
  - Except Local Shared Objects
  - Except Upload / Download API calls
    - Have to be called in a mouse or keyboard event
Sandboxes

- SWF loaded locally
  - Sandbox setting is part of the binary (flag)
    - local-with-file-system
    - local-with-networking
Sandboxes

- SWF loaded locally
  - local-trusted has to be granted by
    - Installer
    - Administrator
    - User
    - (or Projector files)
  - Local trusted files are defined in configuration files
    - Global / User FlashPlayerTrust directory
    - Or in settings.sol
Local Shared Objects
Local Shared Objects

✧ Also known as Flash Cookies
  • Used to persist on computer
  • Private browsing deletes Flash Cookies!
    o Since Flash Player 10.1
✧ The only type of persistent storage
✧ Limited disk space
  • Default 100KB
Local Shared Objects

✧ Object encoding is AMF
  • Actionscript 1.0 / 2.0
    o AMF0
  • Actionscript 3.0
    o AMF3
    o Downgrade to AMF0 is possible
Local Shared Objects

- Stored locally in the user’s app data directory
  - Directory name is #SharedObjects
  - Path contains a random value (LLWKHP8Z)
  - Example
    - C:\users\<user>\AppData\Roaming\Macromedia\Flash Player\#SharedObjects\LLWKHP8Z\<domain>\<SWF Name>\<LSO Name>.sol
  - Local (Standalone player) SWFs contain the sandbox in the path
    - ..\LLWKHP8Z\#localWithNet\<SWF path>
Local Shared Objects

- `SharedObject.getLocal(name, path, secure)`
  - Name of the LSO
  - Path of the LSO
    - ‘/’ is equal to the sandbox
      - Browser => domain name
      - Standalone player => local sandbox
        - E.g: #localWithNet
Local Shared Objects

- **SharedObject.getLocal(name, path, secure)**
  - **Path of the LSO**
    - Path has to be part of the URL
      - www.example.com/files/myswf/path.swf
        - Only access to ‘/’, ‘/files’ and ‘/files/myswf’
      - www.example.com/files/anotherswf/path2.swf
        - Has access to /files
        - Doesn’t have access to ‘/files/myswf’
  - Without path specification an additional directory is created with name of the SWF
    - ../example.com/files/myswf/path.swf/<LSO Name>.sol
Local Shared Objects

- `SharedObject.getLocal(name, path, secure)`
  - Secure
    - Access to the LSO is only possible if the SWF is served over HTTPS
Local Shared Objects

✧ Remote Shared Objects exist as well
  • They require Adobe Flash Media Server
  • Share data between clients
  • Share data in real time
Configuration Overview
Configuration Overview

✧ Client Side
  • Administrative configuration
  • User configuration

✧ Flash Application
  • Author / Developer

✧ Remote
  • Cross domain policies
Client Side Configuration
Client Side Configuration

✧ mms.cfg (Administrative configuration)
  • Deployed by an administrator
  • Enforcing corporate
    o Security settings
    o Privacy settings
  • Location is OS / architecture dependent
    o E.g. Windows 64 bit:
      • %WINDIR%\SysWow64\Macromed\Flash
Client Side Configuration

✧ mms.cfg
  • Privacy options
    o AWHardwareDisable
    o AWHardwareEnabledDomain
    o DisableDeviceFontEnumeration
  • User interface options
    o FullScreenDisabled
Client Side Configuration

- mms.cfg
  - Data loading and storage options
    - LocalFileReadDisable
    - FileDownloadDisable / FileDownloadEnabledDomain
    - FileUploadDisable / FileUploadEnabledDomain
    - LocalStorageLimit
    - ThirdPartyStorage
    - AssetCacheSize
Client Side Configuration

- mms.cfg
  - Update Options
    - AutoUpdateDisable
    - AutoUpdateInterval
    - SilentAutoUpdateEnable
    - SilentAutoUpdateServerDomain
    - SilentAutoUpdateVerboseLogging
    - DisableProductDownload
    - ProductDisabled
Client Side Configuration

- `mms.cfg`
  - Security options
    - LegacyDomainMatching
    - AllowUserLocalTrust
    - FullScreenInteractiveDisable
Client Side Configuration

❖ User Configuration

- Deprecated way to configure Flash

Note: The Settings Manager that you see above is not an image; it is the actual Settings Manager. Click the tabs to see different panels, and click the options in the panels to change your Adobe Flash Player settings.
Client Side Configuration

- User Configuration
  - Display
  - Privacy
    - Microphone / Camera
  - Local Storage
  - Microphone
  - Camera
Client Side Configuration

- User Configuration
  - Also available through the Control Panel
Embedding Flash
Embedding Flash

payer

SWFs can be embedded in different ways

• Directly in the HTML
• Directly calling the SWF in the browser
  o This will generate a DOM in the background
• JavaScript libraries
  o E.g.: swfobject
Embedding Flash

✧ allowScriptAccess
  • Defines if the SWF is allowed to run scripts in the context of the embedding web site
  • Values
    o always (default in older Flash versions)
      • Regardless of the SWFs location, it is allowed to communicate with the embedding web site
    o sameDomain (default in newer Flash versions)
      • SWF may communicate with the embedding web site if they are hosted on the same domain
    o never (deprecated)
Embedding Flash

- allowNetworking
  - Defines how the SWF is allowed to make network calls
  - Values
    - all (default)
      - No restrictions
    - internal
      - Restricted network calls
    - none
Embedding Flash

 didSelect allowNetworking supersedes allowScriptAccess
 • E.g.: ‘allowNetworking = none’ makes allowScriptAccess obsolete
DO NOT host an untrusted SWF on a trusted domain

- allowScriptAccess and allowNetworking won’t prevent exploitation of vulnerabilities
- SWF object may be directly invoked (URL)
  - The created DOM has the default values for allowScriptAccess and allowNetworking
  - Therefore, these settings may be bypassed!
- Reason why allowScriptAccess = never is deprecated
Cross Domain Policies

❖ Request / Send on same domain
  • No policy required

❖ Send to foreign domain
  • GET don’t require a cross domain policy
  • POST require a cross domain policy
    ○ E.g.: sendToURL() method
Cross Domain Policies

✧ Requests to foreign domains
  • Require cross domain policy
Cross Domain Policies

Why do we need a cross domain policy?
  • Flash would have full access to the foreign domain’s content
    o Reading sensitive content
    o Access to the functionality of the application in the context of the current user
Cross Domain Policies

✧ Cross Domain Request Restrictions
  • Cross-Protocol-Scripting (XPS) Prevention
    o Several ports are blocked by default
      • E.g.: SMPT, telnet
  • A policy file has to allow access
    o URL connections need an URL policy file
      • crossdomain.xml
    o Socket connections need a socket policy file
Cross Domain Policies

✧ Same Origin Policy
  • Prevents active content from accessing resources residing on a different origin
    o Based on protocol, port and FQDN
Cross Domain Policies

- Same Origin Policy
  - JavaScript
    - Has only access to the DOM of the embedding page
    - Isn’t able to read content from its origin
      - Exception: Cross Origin Resource Sharing (CORS)
Cross Domain Policies

✧ Same Origin Policy
  • ActionScript
    o Access to the embedding page’s DOM depends on allowScriptAccess / allowNetworking
    o Is able to read content from its origin
      • Without requesting a cross domain policy!
Cross Domain Policies

✧ Same Origin Policy
  • ActionScript
    o The embedded SWF file is in the (remote) sandbox of its origin.
    o Therefore, it has the origin’s trust relationship with other domains!
Cross Domain Policies

Example:

1. Example.com
   - Index.html
2. www.arcus-security.ch
   - arc.swf
3. trusted.arcus-security.ch
   - crossdomain.xml
4. confidential.php

Trust Relation

Untrusted
Cross Domain Policies

- crossdomain.xml
  - Stored in the server’s web root directory
    - Master policy file
  - By default further policies aren’t permitted

```xml
<?xml version="1.0"?>
<cross-domain-policy>
  <allow-access-from domain="*"/>
</cross-domain-policy>
```
Cross Domain Policies

✧ Further policies may be delivered
  • Master policy has to permit meta-policy files
    o Master policy's meta-policy specification may be overridden by a meta-policy specified in the HTTP response header
      • X-Permitted-Cross-Domain-Policies
    o Meta-policy files have to be loaded manually by the SWF application
      • Security.loadPolicyFile(url:String)
Cross Domain Policies

- Possible meta-policies (master policy configuration)
  - All
  - by-content-type (Content-Type: text/x-cross-domain-policy)
  - by-ftp-filename (/crossdomain.xml)
  - master-only
  - None (ignores even the master policy file)
  - none-this-response (HTTP response header)
Cross Domain Policies

✧ Meta-policy restrictions
  • Content-Type
    o text/*
    o application/xml
    o application/xhtml+xml
Cross Domain Policies

1. Example SWF file (example.com)
2. Crossdomain.xml file
3. Index.html file

www.arcus-security.ch

/subdir/
somepolicy.txt
Cross Domain Policies

- Cross Domain Configuration
  - cross-domain-policy
    - site-control
      - permitted-cross-domain-policies
    - allow-access-from
      - domain
      - to-ports (only in socket policies)
      - secure
Cross Domain Policies

✦ Cross Domain Configuration
  • cross-domain-policy
    o allow-access-from-identity
  • signatory
    - Certificate
      • fingerprint-algorithm
      • fingerprint
Cross Domain Policies

✧ Cross Domain Configuration
  • cross-domain-policy
    o allow-http-request-headers-from
      • domain
      • headers
Cross Domain Policies Demo
Common Vulnerabilities
Common Vulnerabilities

✧ Passing variables to the SWF
  • Flashvars
  • Declared in `<embed>` / `<object>` tags
  • Passed as URL parameters
    o `myswf.swf?a=a&b=b`
Common Vulnerabilities

- Reputational Damage
  - Direct embedding
    - Pictures
    - Text
    - Movies
  - ../vulnerable.swf?image=http://a.com/image.gif
Common Vulnerabilities

- Improved Phishing
  - Change of data flow
    - Sensitive data is being sent to an attacker
  - ../vulnerable.swf?configuration=http://a.com/my.conf
Common Vulnerabilities

✧ Redirection Attacks
  • Redirect to
    o Phishing site
    o Malware
    o …
  • ../vulnerable.swf?
    url=http://a.com/
Common Vulnerabilities

🔹 Cross Site Scripting
  • Access to the DOM
  • Stealing session cookies
  • Redirection attacks
  • Malware Infection
  • Abuse of exposed ActionScript functions
  • ../vulnerable.swf?text=
    <a href="javascript:alert(1)">click here</a>
  • javascript:
Common Vulnerabilities

✧ Cross Site Flashing
  • Achieves the same as Cross Site Scripting
  • May run ActionScript code in the context of the vulnerable application
    o Implementation depended of the vulnerable application
    o Therefore, may have the same Security Sandbox
Common Vulnerabilities Demo I

Trust Relation
Common Vulnerabilities Demo II
Questions?