The Image that called me

Active Content Injection with SVG Files

A presentation by Mario Heiderich, 2011

Introduction

- Mario Heiderich
  - Researcher and PhD student at the Ruhr-University, Bochum
  - Security Researcher for Microsoft, Redmond
  - Security Consultant for XING AG, Hamburg
  - Published author and international speaker
  - HTML5 Security Cheatsheet / H5SC
  - PHPIDS Project

Today

- SVGs and the modern web
  - What are SVGs?
  - What are they capable of?
  - Which browsers “understand” SVG?
  - Why there are conflicted areas?

- **And what does that have to do with security?**

SVG Images

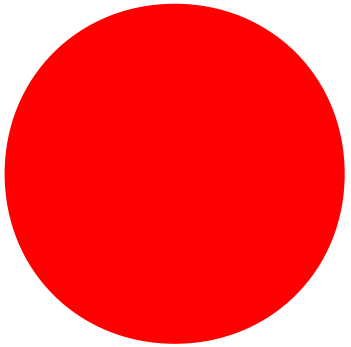
- Scalable Vector Graphics
- XML based, therefore
  - Versatile
  - Accessible
  - Compressible
  - “Stylable” w. CSS
  - Open
- Great for mobile devices
- Easy to parse and process
- Ancient format, older than 10 years
- Relations to ~~HTML5~~, *the living standard*

SVG History

- Proposed by several W3C members in 1998
- Derived from Adobe Postscript and VML
- Developed in 1999
- Currently at version 1.1
  - Version 1.2 still a working draft
  - Might be overtaken by SVG 2.0
- Good browser support
  - Gecko, Webkit, Presto, and Trident

Basic Example

```
<svg xmlns="http://www.w3.org/1999/svg">
  <circle r="40" fill="red"></circle>
</svg>
```



SVG Family

- **SVG Tiny 1.2**
  - Designed for cellphones and smart-phones
  - 47 Tags
- **SVG Basic 1.1**
  - Designed for handhelds, tablets and net-books
  - 71 tags
- **SVG Full 1.1**
  - Full feature set
  - 81 tags

Features

- Geometrical shapes
  - Circles, ellipses, squares, lines and more
  - SVG fonts
- Font specific formatting and glyph styles
- **Links**
- Animations and Transformations
- Gradients and Effects
- Meta-data
- **Scripting and Events**
- **Inclusion of arbitrary objects**

SVG in Action

RUB

Scripting

- The following SVG executes JavaScript

```
<svg xmlns="http://www.w3.org/1999/svg">
  <script>
    alert(1)
  </script>
</svg>
```

- More examples?

More Scripting

```
<svg xmlns="http://www.w3.org/2000/svg">
   <g onload="javascript:alert(1)"></g>
</svg>

<svg xmlns="http://www.w3.org/2000/svg">
   <animation xlink:href="javascript:alert(1)"/>
</svg>

<svg xmlns="http://www.w3.org/2000/svg">
   <foreignObject xlink:href="javascript:alert(1)"/>
</svg>

<svg xmlns="http://www.w3.org/2000/svg">
   <set attributeName="onmouseover" to="alert(1)"/>
</svg>

<svg xmlns="http://www.w3.org/2000/svg">
   <handler
      xmlns:ev="http://www.w3.org/2001/xml-events"
      ev:event="load"
   >alert(1)</handler>
</svg>
```

Deploying SVGs

- Several ways of deploying SVGs, implemented by modern browsers
- **Five important ones are:**
  - Opening the file directly
  - Deployment via `<object>` or `<embed>`
  - Deployment via `<img>` or `<image>`
  - Deployment via CSS `background`/`list-style`/`content`/`cursor`
  - In-line SVG

Security Boundaries

- SVG capabilities based on deployment method
- A model, based on expectations
- Heterogeneous implementations

- **And a whole new world of bugs and vulnerabilities**

XSS

- SVGs deployed via `<img>` and `<image>` tag **should** not execute JavaScript
- Same goes for SVGs used via CSS
- Or SVG fonts

- SVGs deployed via `<iframe>`, `<embed>` or `<object>` should, though
- So browsers need different approaches

- Learning by fixing?

Local SVGs

- SVGs opened directly are allowed to script
- Imagine the following attack:
  - Attacker uploads an image with an exciting motive to a server
  - Victim navigates to the image, likes it, saves it locally, downloads folder or desktop
  - Victim wants to watch the image again and double-clicks it
  - Image is an SVG and executes JavaScript locally
  - **Attacker can read local files (same directory, sub-folders)**
  - Attacker can even load and start Java applets or worse
- Very likely too be used in real life attacks!
- Porn sites, Email attachments, Malware

In-line SVG

- Suggested by the HTML5 specs
- Working on all modern browsers – except Opera
- No strict XML parser anymore
  - `<svg><circle r=40 fill=red></svg>`
  - See – no quotes, no trailing slash
- Reduced feature set
- <svg> introduces many new XSS vectors
- XSS filter bypasses

Scoping

- SVG images are treated by browsers as **XML**
- Same is for in-line SVG blocks
- **XML treats plain-text tags differently**
  - Entities and canonical character representations are treated equally
  - 0-Day filter bypasses ahead
- This enables a new attack technique on Firefox

- **DEMO**

- And it's even worse
- In-line SVG “self-terminates” open HTML elements

Opera

- A long history of SVG flaws
  - JavaScript execution via SVG fonts
  - XSS via CSS background images
- Now SVGs deployed via CSS/<img> cannot script anymore
- But - not all kinds of attacks need scripting to succeed

- **DEMO**

Other Browsers

- Firefox 4 crashed badly on SVGs embedding JS
- Chrome produces weird things when using <foreignObject> and <iframes
- Opera deploys Java applets via SVG fonts
- And what about other XML related attack patterns?
  - External entities
  - SVG Tiny 1.2 Java Events
  - Entity bombs
  - Etc. etc.
- Some browsers support SVG Masks, perfect for click-jacking

Wrap-Up

- SVGs are **not just images** but mini-applications
- <img> tags can now deploy Java, PDF and Flash – and call you on Skype
- In-line SVG creates small XML islands enabling XML attacks on HTML websites
- SVG and XSLT work too, enabling DoS and other attacks
- Web-security and XML security, they meet again!
- And XXE is back – remember 2002's advisories?

- **SVG is not getting enough attention in the security community**
- **SVG provides a lot of room for more security research**

Defense

- More difficult than one might assume
  - No existing filter libs
  - No good documentation
  - XSS vectors are hard to comprehend
  - New vectors coming up weekly

- SVG files should not be perceived as *images*
- Allowing SVG for upload == allowing HTML for upload
- SVG can embed, link or reference any kind of content over cross domain borders
- SVG provides new ways of payload obfuscation

Future Work

- **SVG Purifier**
  - Based on HTMLPurifier 4.2.0
  - Still very young, and so far unpublished
- More articles on the HTML5 Sec Cheatsheet Wiki
- **Publications, to raise awareness**
  - Academic publication is in preparation
- More demo vectors on the H5SC to demonstrate impact

- OWASP research and documentation?

Links

- Wikipedia on SVG http://en.wikipedia.org/wiki/Scalable_Vector_Graphics
- W3C SVG Working Group http://www.w3.org/Graphics/SVG/
- SVG Full 1.1 (W3C) http://www.w3.org/TR/SVG11/
  - SVG Basic 1.1 and SVG Tiny 1.2 http://www.w3.org/TR/SVGMobile/
  - SVG 2.0 http://dev.w3.org/SVG/profiles/2.0/publish/intro.html
- Adobe's SVG Zone http://www.adobe.com/svg/

  H5SC http://html5sec.org/
- XSLT and SVG http://scarybeastsecurity.blogspot.com/20...riousity.html
- Opera SVG Bug http://heideri.ch/opera/
- HTMLPurifier http://htmlpurifier.org/
- JSBin http://jsbin.com/
- More SVG fun http://maliciousmarkup.blogspot.com/20...re-xml-fun.html

Thanks

- Thanks for listening
- Questions || Comments?
- Discussion and tool preview?

- Thanks to
  - Gareth Heyes and Manuel Caballero from UNH
  - Alexey Silin / LeverOne
  - Dave Ross