Yandex

CSP - the panacea for XSS or placebo?

Taras Ivashchenko
Information Security Officer

$ whoami

Information security officer at Yandex's product security team

Web application security researcher

Yet another security blogger www.oxdef.info

XSS

XSS

Why again about XSS?!

Still one of the **the most common** web application security issues

Ok, but please don't show me those **alerts**

Prevention

Input validation

Output escaping **depending on context**

`httponly` session cookie

Browser based solutions: IE filter, NoScript

?

CSP

Content Security Policy

Browser side mechanism to mitigate XSS attacks

Source whitelists for client side resources of web application

`Content-Security-Policy` HTTP header

W3C Candidate Recommendation

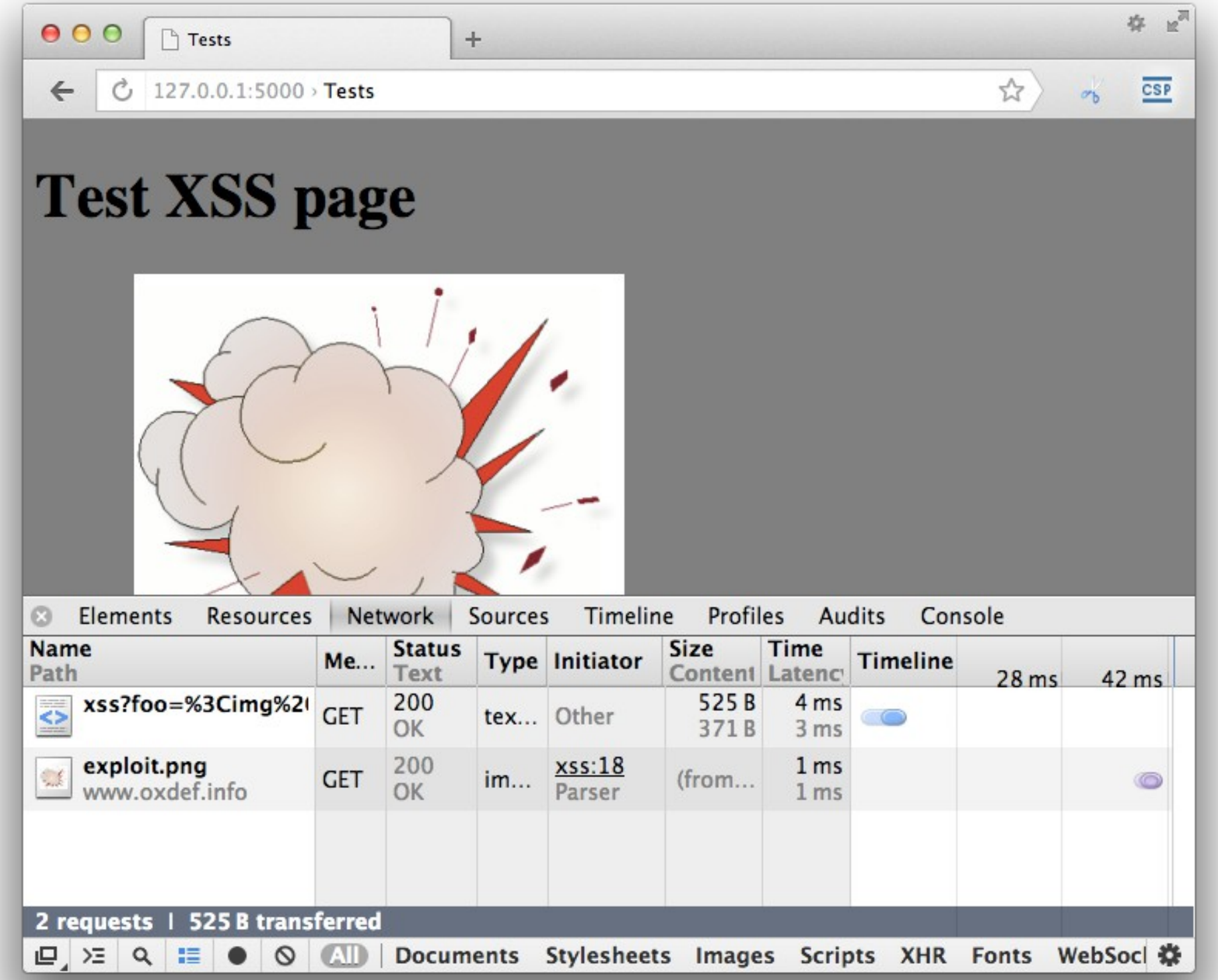
How it Works

HTML Template

```
<h1>Test XSS page</h1>
<h3>Hello, <i> {{ foo | safe }}!</i></h3>
```

Demo URL

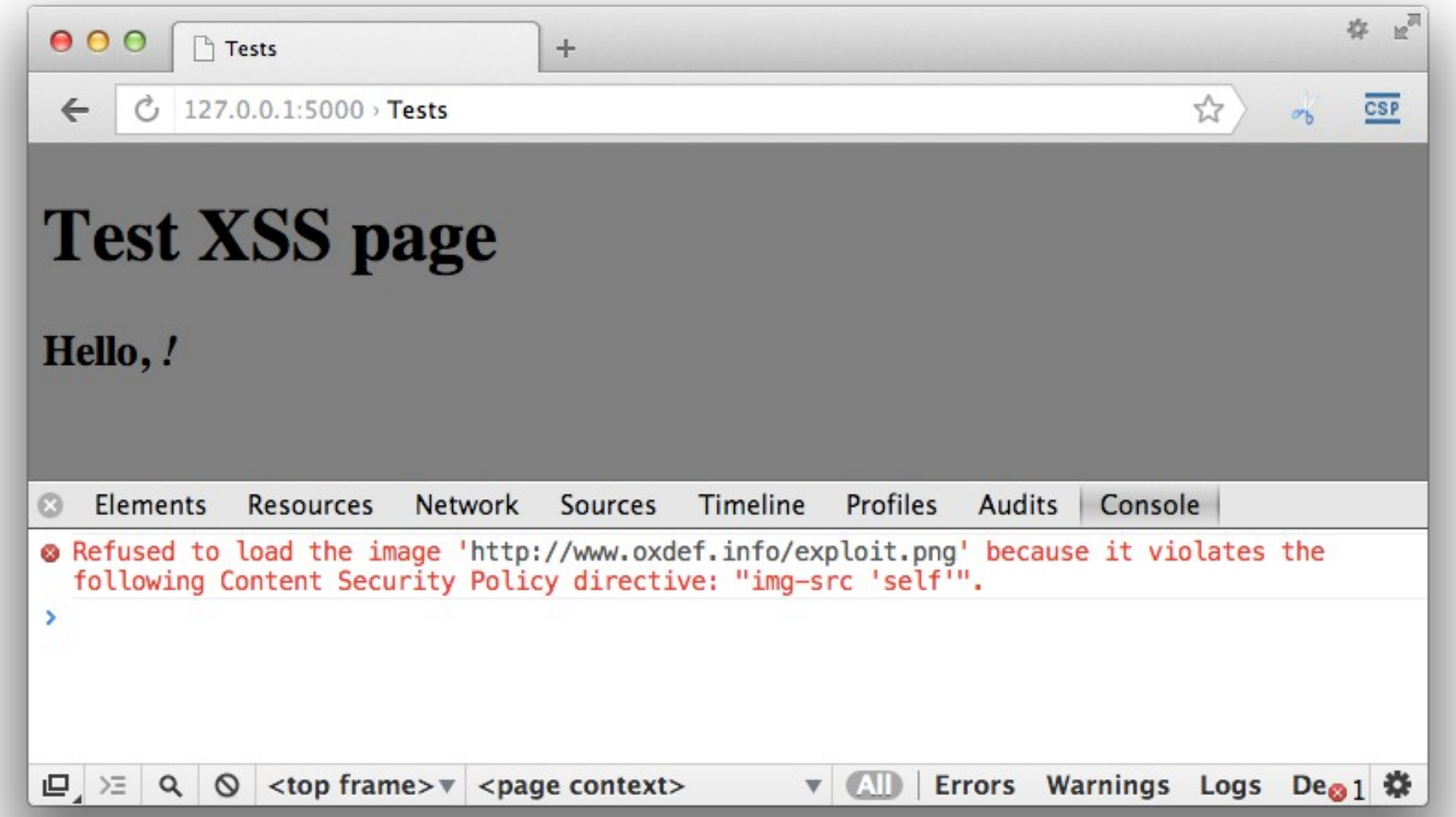
```
http://127.0.0.1:5000/xss?foo=
 <img src="http://www.oxdef.info/exploit.png">
```


Tests
127.0.0.1:5000 › Tests
CSP
Test XSS page
Elements
Resources
Network
Sources
Timeline
Profiles
Audits
Console
Name
Path
Me...
Status
Text
Type
Initiator
Size
Content
Time
Latency
Timeline
28 ms
42 ms
xss?foo=%3Cimg%2(
GET
200
OK
tex...
Other
525 B
371 B
4 ms
3 ms
exploit.png
www.oxdef.info
GET
200
OK
im...
xss:18
Parser
(from...
1 ms
1 ms
2 requests | 525 B transferred
All
Documents
Stylesheets
Images
Scripts
XHR
Fonts
WebSoc

Without CSP

CSP in Action

```
Content-Security-Policy: img-src 'self'
```



Control JavaScript

Policy

```
Content-Security-Policy: default-src 'self'; script-src
'self' static.example.com
```

HTML

```
<!doctype html><html><head>
    <meta charset="utf-8">
    <script src="/js/jquery-1.10.2.js"></script>
    <script src="//evil.net/evil.js"></script>...
```

console.log

```
Refused to load the script 'http://evil.net/evil.js'
because it violates...
```

Unsafe-inline and unsafe-eval

- `unsafe-inline` **allows:**
  - **Inline scripts and styles**
  - `onclick="..."`
  - `javascrtipt:`
  - You **should not** include it in the policy!
- `unsafe-eval` **allows:**
  - `eval()`
  - `new Function`
  - `setTimeout`, `setInterval` **with string as a first argument**
  - You **should not** include it in the policy!

Other Directives

`media-src` — audio and video

`object-src` - plugin objects (e.g. Flash)

`frame-src` — iframe sources

`font-src` — font files

`connect-src` — XMLHttpRequest, WebSockets, EventSource

Reporting

Policy

```
Content-Security-Policy-Report-Only: ...; report-uri csp.php
```

Log contents

```
{
    "csp-report": {
        "violated-directive": "img-src data: ...
*.example.com",
        "referrer": "",
        "blocked-uri": "https://static.doubleclick.net",
        "document-uri": "https://example.com/foo",
        "original-policy": "default-src ...; report-uri
csp.php"
    }
}
```

Browser Support

`Content-Security-Policy` 25+ 23+ 1.7+

`X-Content-Security-Policy` 4 - 22 10 (sandbox)

`X-WebKit-CSP` 14 - 25 5.1+

Mobile browsers: 7.0+ 28+ 23+

Bypass

Manipulating HTTP response headers

Implementation bugs: MFSA 2012-36: Content Security Policy inline-script bypass

JSONP

XSS without JS

See in the Next Version: nonce-source

Policy

```
Content-Security-Policy: script-src 'self' nonce-Nc3n83cnSAd
```

HTML Code

```
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <script src="/js/jquery.min.js"></script>
</head>
<body>
    <script nonce="Nc3n83cnSAd">
    // Some inline code here
    </script>
```

Case-study

About the Service

One of the most popular mail services in Russia

Over **12 million** email messages daily

Lots of client side code and hosts to communicate with

CSP Tester

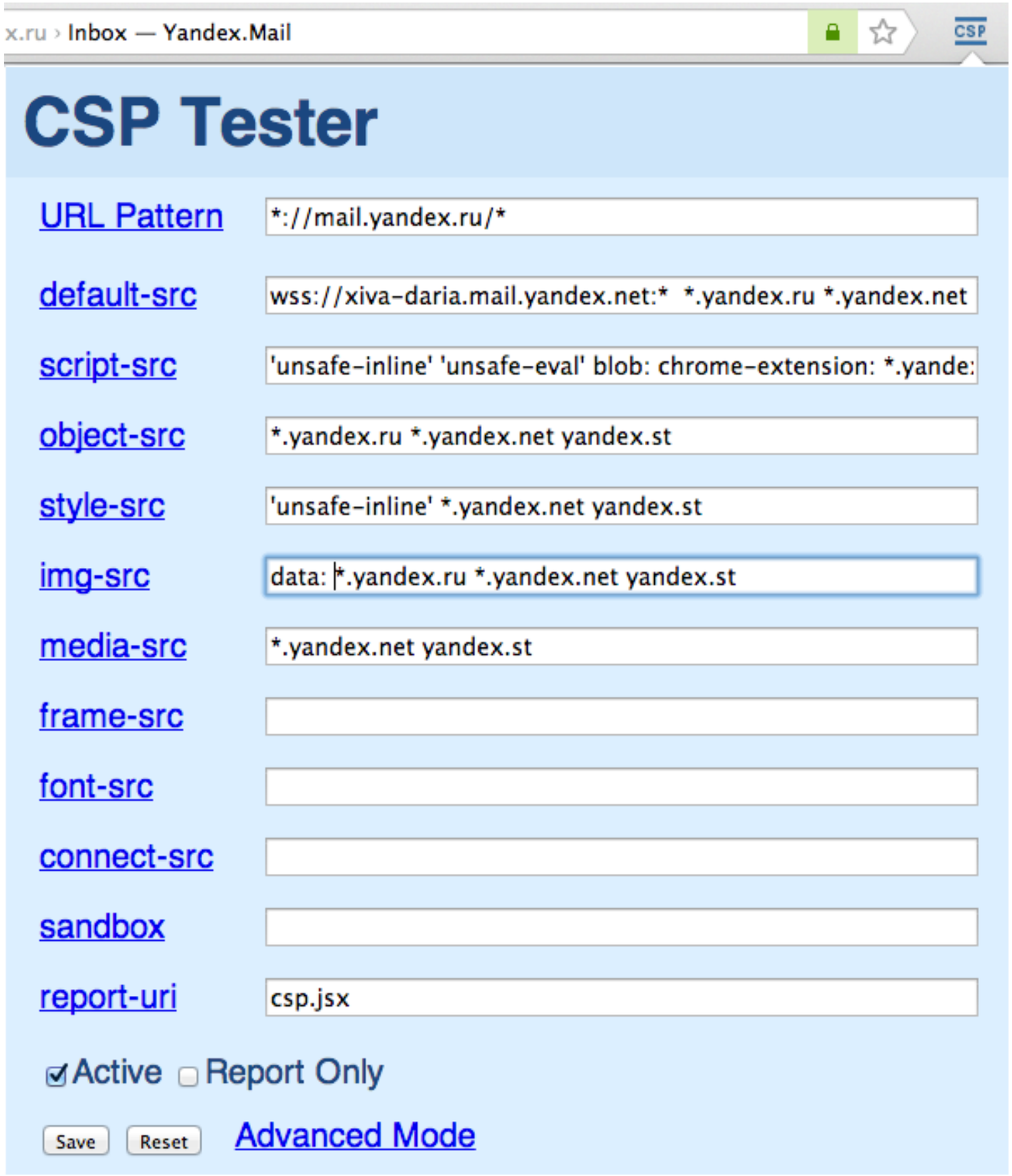
Extension for Chromium based browsers

Simple and Advanced modes

`Content-Security-Policy` **and** `X-WebKit-CSP` **headers**

Help links for directives

https://github.com/oxdef/csp-tester



CSP Tester in action

The Plan

1. Test it on the corporate mail
2. It's ok - let's try it on production in `Report-Only` mode
3. Analyze tons of logs ;-(
4. Fix bugs and improve the policy
5. Switch to `block` mode
6. Profit! :-)

Changes in service

Try to remove all inline code

Log Analysis

`awk`, `grep`, `sort`, `head` for **gigabytes** of logs?

Yes, but we can do it in more complex way with help of Python

Charts for directives and blocked URIs

Problems

Browser implementations differ

3rd party JS libraries

Inline styles in HTML letters

Browser extensions

What is that *** external code doing in our DOM?

From Report-Only to Block mode

Fix bugs from CSP logs

Use only standard CSP HTTP header

Allow browser extensions

`unsafe-inline` for `style-src`

`unsafe-eval` for `script-src`

Tips

Teach your front-end developers

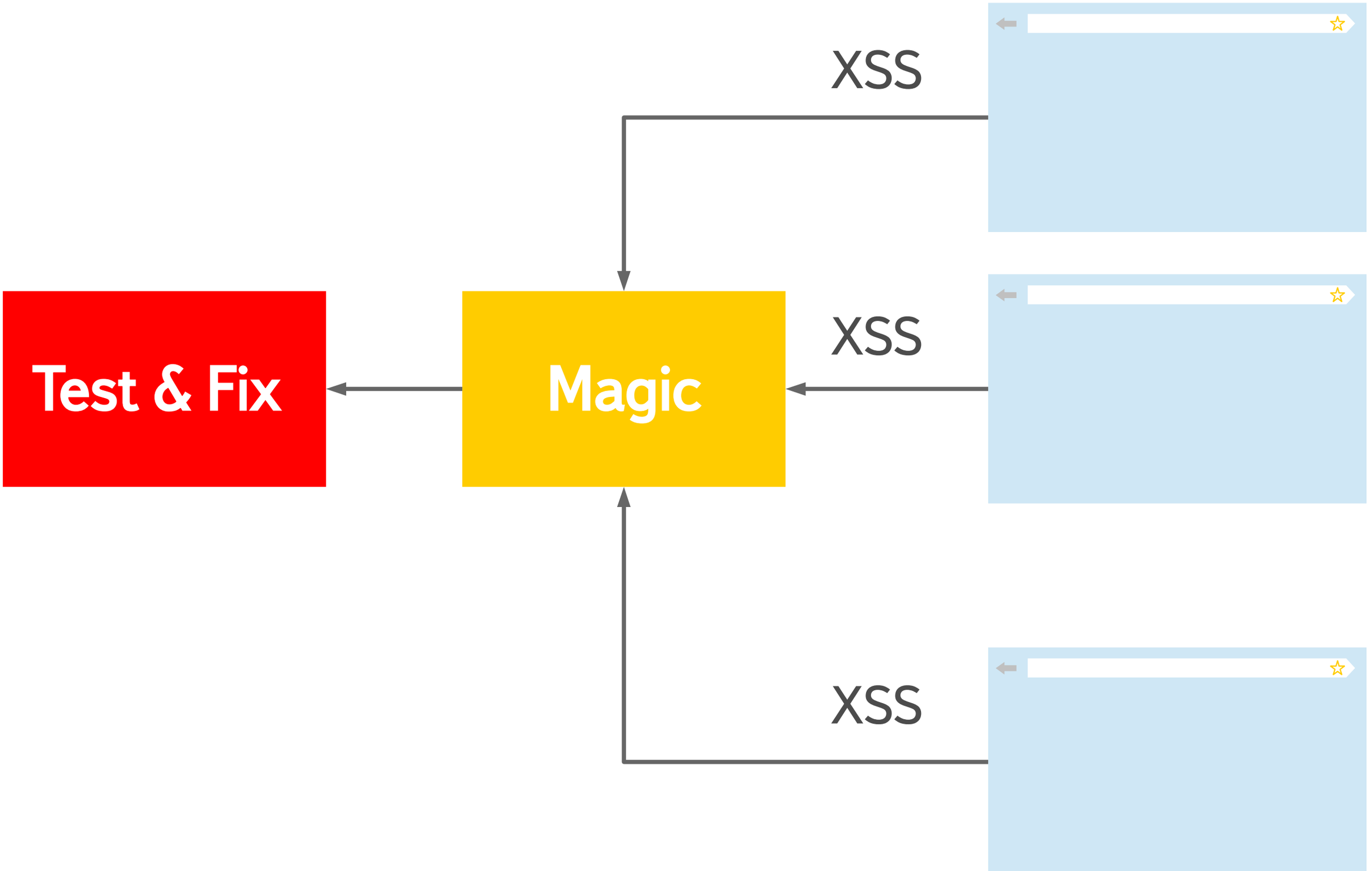
Add CSP as security requirement for new products

Don't forget about mobile versions!

Research your core front-end components to support CSP

Assign developer responsible for CSP

CSP Based IDS



Conclusion

CSP is not a panacea

but it's a **good** «yet another **level**» to **protect** your **users** against XSS attacks

To be continued ;-)

Taras Ivashchenko

Information Security Officer

oxdef@yandex-team.ru

http://company.yandex.com/security

Thanks