MINISTÈRE DE LA DÉFENSE

OpenOffice / OpenDocument and MS Office 2007 / Open XML security

PacSec 2006 – 30/11/2006 – http://pacsec.jp

DÉLÉGATION GÉNÉRALE POUR L'ARMEMENT

Philippe Lagadec – DGA/CELAR
philippe.lagadec(à)laposte.net

OpenOffice and OpenDocument

- **OpenOffice.org**
  - Open-source version of Sun StarOffice
  - Nickname “**OOo**”
  - Can read/write most MS-Office documents and features

- **OpenDocument**
  - New format for OpenOffice v2 documents
    - quite similar to OpenOffice v1
  - Now used by other applications (Koffice, Abiword…)
  - XML files in a ZIP archive
  - ISO standard since May 2006, OASIS since 2005

MS Office 2007 and Open XML

- **Microsoft Office 2007**
  - Formerly known as “Office 12”
  - Future version of MS-Office, many changes
  - Beta versions already available in 2006

- **Open XML**
  - New **default** format for most Office 2007 documents (Word, Excel, PowerPoint, except Access)
  - XML files* in a ZIP archive (sounds familiar ?)
  - ECMA draft standard, work in progress

• (*): …sometimes stuffed with bits of binary OLE or BIFF files (it is Microsoft after all)

OpenDocument and OpenXML security

- Let's take a closer look at these new formats:
  - **Can they embed active content ?**
  - **Can they hide sensitive data ?**
  - **Are there new security issues ?**
  - **How can we protect information systems ?**
  - **How can we filter unwanted features ?**

- This is not a complete and definitive security analysis.
- This analysis does not focus on security features such as encryption and signature.
- OpenOffice v1.0 security analysis by F-Secure in 2003:
  - http://www.f-secure.com/weblog/archives/openoffice_security.pdf

Versions used for this analysis

- **OpenOffice.org 2.0.3 and 2.0.4**
- OpenDocument v1.0 specifications
  - http://www.oasis-open.org/committees/download.php/12572/OpenDocument-v1.0-os.pdf
- **MS Office 2007 Beta 2 “Technical Refresh”**
- Open XML specifications: ECMA working draft v1.4 and final draft (9th Oct 06)
  - http://www.ecma-international.org/news/TC45_current_work/TC45_available_docs.htm
    - Some details might change as Office 2007 is still beta software (well, for a few days now).
- Both on Windows XP SP2 fr

Specifications analysis

- One big advantage of the new ***open*** formats, compared to good-old proprietary ones:
  - Security analysis is much easier :-)

- However you have to read the specs…
- OpenDocument : **700 pages**
- Open XML final draft: **6036 pages !! ;-(**
  - and even with this, everything is not described, VBA macros for example…

Usual security issue 1: Malware inside files

- **Many usual file formats can embed <u>active content</u>, which may be malicious:**
  - EXE, COM, PIF, SCR, … : *Binary code*
  - BAT, CMD, VBS, JS, … : *Commands, Scripts*
  - HTML, XML, XHTML : *Scripts*
  - PDF : *Scripts, Embedded files, Commands*
  - Word, Excel, PowerPoint, Access, … : *Macros, OLE objects, Embedded files, Commands*
    - See http://actes.sstic.org/SSTIC03 (in French, sorry)
- **All of these are often underestimated, because many haven't been used by viruses "in the wild".**
  - **…but they can be efficient to hide a Trojan horse !**
  - **The most efficient attack against a secure system.**
- **"It's not a bug - it's a feature."**

Usual security issue 2: Data leak inside documents

- **Usual office documents may contain a lot of hidden information:**
  - User name, organization
  - History of changes, additions, deletions
  - Notes, Comments
  - Hidden text
  - A whole spreadsheet behind a simple diagram
    - (With confidential corporate figures !)
  - Sometimes even random chunks of memory
- **Something bad could happen if that information gets into the wrong hands.**

Part 1:
OpenOffice.org and OpenDocument

OpenOffice.org files

Format	Application	OOo v2 document	OOo v2 template	OOo v1 document	OOo v1 template
Text	**Writer**	**.odt**	**.ott**	.sxw	.stw
Spreadsheet	**Calc**	**.ods**	**.ots**	.sxc	.stc
Presentation	**Impress**	**.odp**	**.otp**	.sxi	.sti
Drawing	**Draw**	**.odg**	**.otg**	.sxd	.std
Database	Base	.odb			
HTML template	Writer/Web	(.html)	.oth	(.html)	.stw
Master document	Writer	.odm		.sxg	
Formula	Math	.odf		.sxm	

Only OOo v2 Text, Spreadsheet, Presentation and Draw are covered by OpenDocument v1 specifications.

OpenDocument format overview

- **A document is stored in a ZIP compressed archive**
- **XML files:**
  - content.xml: document body
  - styles.xml: style data
  - meta.xml: metadata (author, title, …)
  - settings.xml: OOo settings for document
  - META-INF/manifest.xml: files description
- **Optional files:**
  - Pictures and thumbnails: JPEG, PNG, SVG, …
  - Embedded charts/drawings/documents, OLE objects

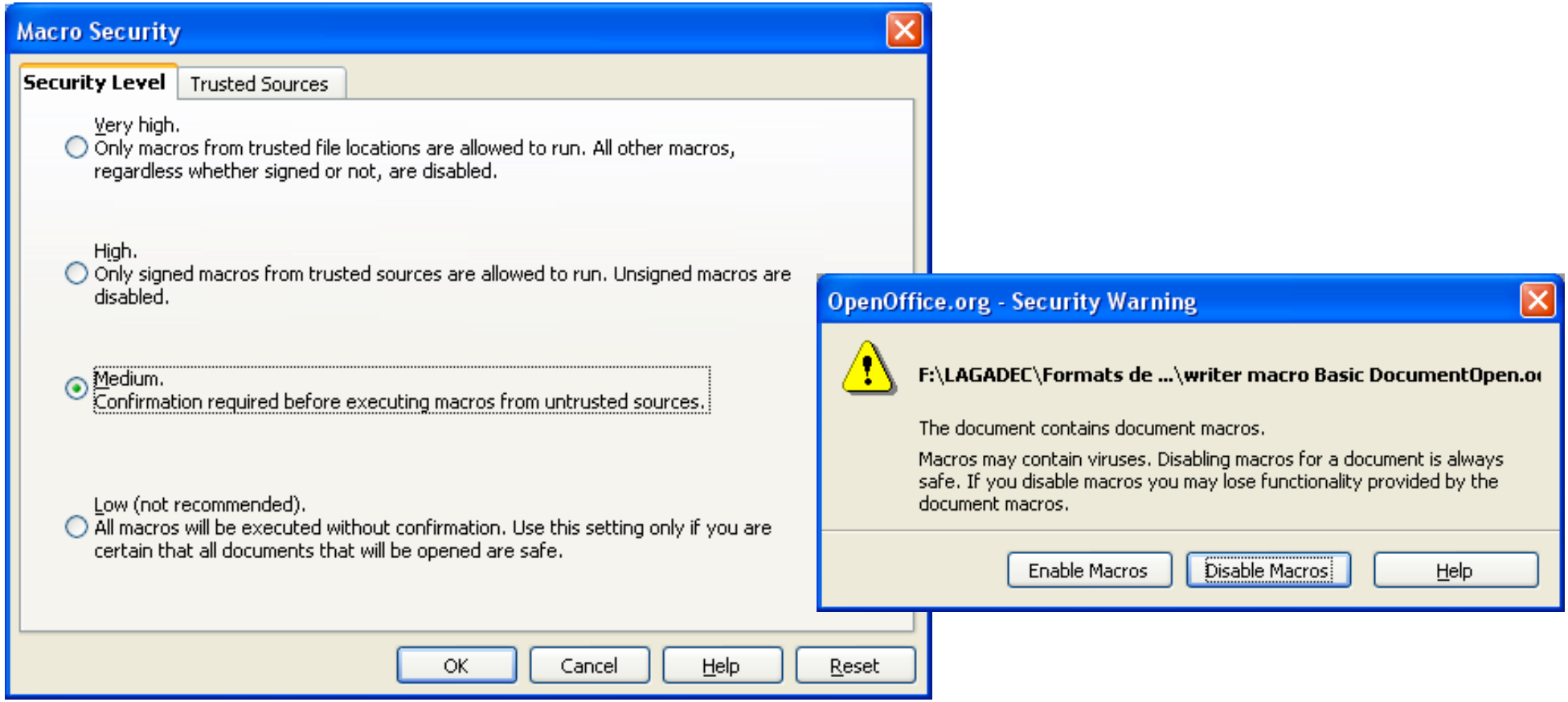
OpenOffice macros

- **OpenOffice v2.0.x has 4 available languages for macros:**
  - Basic, Javascript, Java (Beanshell), Python
  - More macro languages may be added in the future.
- **Each macro language gives access to UNO:**
  - UNO: Universal Network Objects
  - **Very powerful API**: access to OpenOffice objects and the operating system
  - **Ability to write effective malware**.
- **Macros can be assigned to events (document open, …) or forms.**

OpenOffice macros security modes

- **4 modes, quite similar to MS Office 2000-2003:**
  - **Low** (to be avoided): no protection at all
  - **Medium (default)**: macros can be enabled by the user before any access to the document.
    - Simple popup warning.
  - **High**: only signed macros or trusted directories are allowed. No warning if signature authority was already accepted or from a trusted location.
  - **Very high**: only trusted locations, no signature, no warnings.
- **Same default level as MS Office 97**
- OpenOffice 2.0.2 vulnerability: ability to bypass macros warning.

OpenOffice macro security modes



OpenOffice macros storage

- OOo **Basic** macros are stored in XML files, in the "**Basic**" directory of the archive.
- **Java, Javascript and Python** macros are stored in script files, in the "**Scripts**" dir.
- Examples:
  - **Basic**/Standard/Module1.**xml**
  - **Scripts/beanshell**/Library1/MyMacro.**bsh**
  - **Scripts/javascript**/Library1/MyMacro.**js**
  - **Scripts/python**/MyMacro.**py**

OLE objects in OpenDocument

- **OpenDocument files can embed OLE objects** (at least on Windows).
- **An OLE object is stored in a *binary file* inside the document.**
  - Microsoft OLE2 storage file (not really an open format…)
- **An OLE Package may contain any file or a command line (potential malware).**
  - If the user double-clicks on the object, the file or the command is launched by the system.

OLE objects in OpenDocument

- **OpenOffice itself doesn't warn about potential malware in OLE Package objects**
  - The warning only comes from Windows (packager.exe)
  - No confirmation on old Windows versions ! (2000 SP4)

- **Windows MS06-065 vulnerability:**
  - It is possible to spoof the command line of an OLE Package object to show a dummy filename instead:
    - cmd.exe /c […bad commands…] /joke.txt
  - http://secunia.com/advisories/20717

Other security issues

- Other potential ways to embed malware in OpenDocument files:
  - **HTML scripts**: OpenDocument allows to embed scripts (js or vbs), which are only activated when the document is saved as HTML and opened in a browser.
  - **Java applets**: Java code is executed in a sandbox from OOo, which should be quite safe.
    - But for example OpenOffice 2.0.2 had a vulnerability which permitted an escape from the sandbox.
  - **URLs**: directly launched in the default web browser.
    - Hopefully Javascript and VBscript URLs are not permitted by OpenOffice.

Other security issues

- **VBA macros** in MS Office documents are stored in comments when converted by OpenOffice. They are reactivated when saved back to MS Office format.
  - VBA code is stored as comments in an OpenOffice Basic dummy macro.
    - Same warnings as other macros.
  - Work in progress to provide direct VBA execution in future OpenOffice versions.

Other security issues

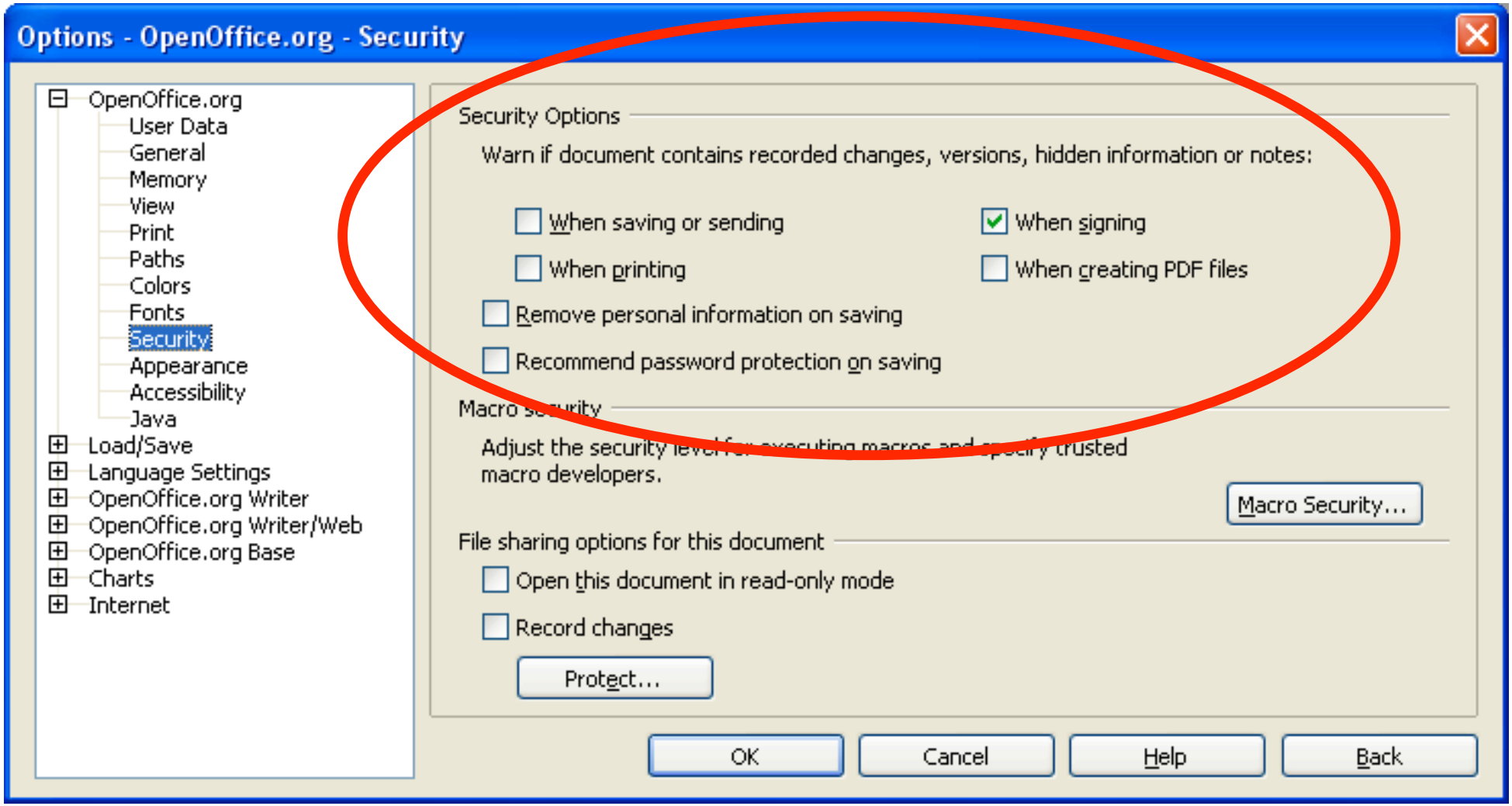
- French ESAT researchers have found that OpenOffice handling of encrypted/signed documents has conception flaws (among other things):
  - For example it is possible to replace a macro in an encrypted document by a cleartext malicious macro, without any warning.

    - De Drézigué, Fizaine, Hansma, *"In-depth Analysis of the Viral Threats with OpenOffice.org Documents"*, Journal in Computer Virology, 2006.
      - http://www.springerlink.com/content/1772-9904/?k=openoffice
    - Filiol, Fizaine, *"Le risque viral sous OpenOffice 2.0.x"*, MISC magazine n°27, 09/2006.

Hidden data in OpenDocument

- Like MS Office, OOo documents may hide sensitive data.
  - Metadata, hidden text, comments, revision marks, …
- OOo has features to warn about hidden information when signing, exporting to PDF or saving.
- However this does not include OLE objects.

Hidden data protection in OOo



OpenOffice security

- Conclusion: **OpenOffice is not *absolutely* more (or less) “secure” than MS Office, concerning malware or hidden data.**
  - Both have similar security issues, with subtle differences.
  - OpenDocument provides more ways to embed malware, but some features are more secure.
- **However, the OpenDocument format makes it much simpler to detect and filter active content or hidden data.**
  - (more on this later)

Part 2 :
MS Office 2007 and Open XML

MS Office 2007 files: Open XML

- **New Open XML default formats:**
  - **Word**: .doc**x**, .doc**m**, .dot**x**, .dot**m**
  - **Excel**: .xls**x**, .xls**m**, .xlt**x**, .xlt**m**, .xls**b**, .xla**m**
  - **Powerpoint**: .ppt**x**, .ppt**m**, .pps**x**, .pps**m**
  - **Access**: .accdb (binary, not OpenXML)

- **Compatibility mode for previous formats** (binary OLE2 files): doc, dot, xls, xlt, ppt, pps, …

- **Converter pack** to allow Office 2000, XP and 2003 to read/write new OpenXML formats.

Open XML format overview

- **A document is stored in a ZIP compressed archive**
- Open Packaging Conventions (OPC):
  - Specifications for new Microsoft formats: Open XML, XPS
  - **[Content_Types].xml**: description of all files in the archive
  - **.RELS files** (XML)**:**
    - Store relationships between “parts” in the OPC archive
- **XML data files (example for Word 2007):**
  - word/**document.xml**: document body
  - word/**styles.xml**: style data
  - word/**settings.xml**: settings for the document
  - docProps/**app.xml** and **core.xml**: metadata (author, title, …)
  - …
- **Optional binary files:**
  - Pictures and other media: JPEG, PNG, GIF, TIFF, WMF, …
  - OLE objects, macros, printer settings

Open XML and macros

- **Open XML can embed VBA macros, just like previous Office formats.**
- **But Office 2007 distinguishes “normal” from “macro-enabled” documents:**
  - Normal (default): .docx, .xlsx and .pptx
  - Macro-enabled: .docm, .xlsm, .pptm
- **Default normal “x” documents cannot embed macros.**
  - A “macro-enabled” document renamed to “normal” is rejected by Office 2007.

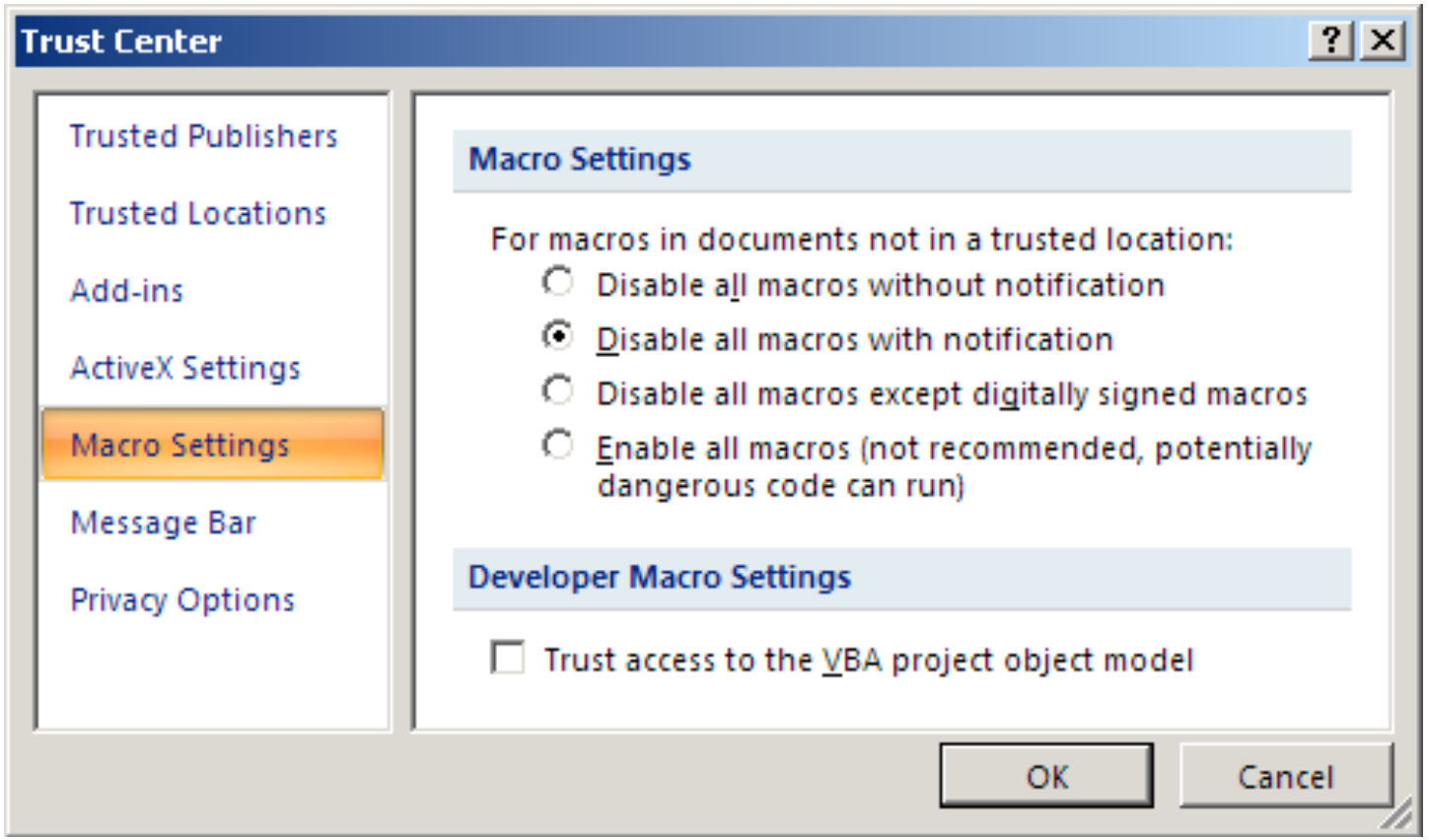
MS Office 2007 macros security modes

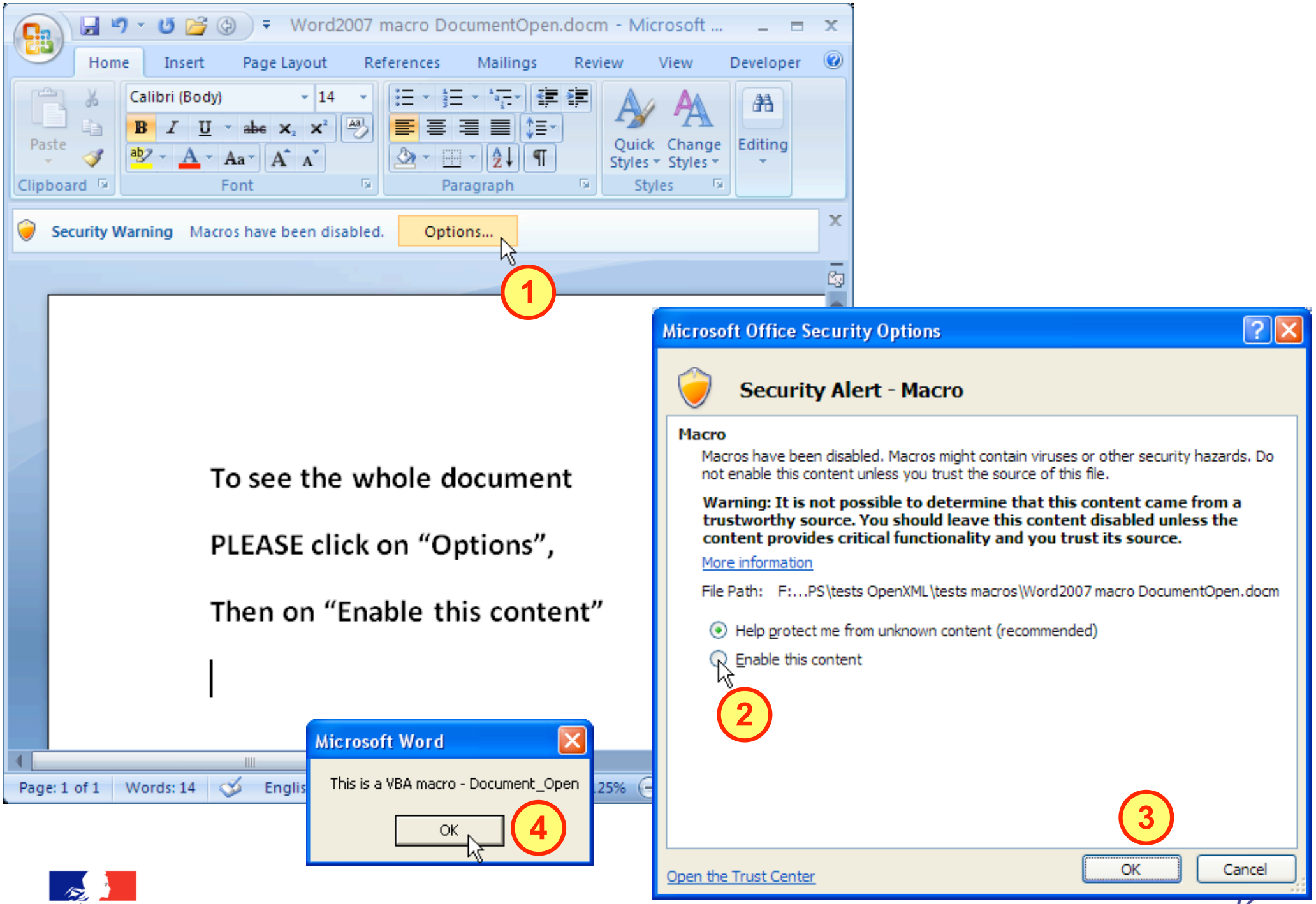
- In **MS Office 2000/XP/2003**, only signed macros can be activated with **default "high security" mode**.
  - User has to switch to "medium security" to launch unsigned macros (and to re-open the document).
  - In medium security mode, a popup Window asks to enable macros **BEFORE** the user can see the document.

- **MS Office 2007 new modes and UI change**:
  - No more "medium security" or "high security" modes.
  - **New default mode "*disable all macros with notification*"**

MS Office 2007 Macro security modes

- The new “Trust Center” gives access to all security parameters:




Word2007 macro DocumentOpen.docm - Microsoft ...
Home
Insert
Page Layout
References
Mailings
Review
View
Developer
Paste
Clipboard
Calibri (Body)
14
Font
Paragraph
Quick Styles
Change Styles
Styles
Editing
Security Warning
Macros have been disabled.
Options...
1
To see the whole document
PLEASE click on “Options”,
Then on “Enable this content”
Page: 1 of 1
Words: 14
Microsoft Word
This is a VBA macro - Document_Open
OK
4
Microsoft Office Security Options
Security Alert - Macro
Macro
Macros have been disabled. Macros might contain viruses or other security hazards. Do not enable this content unless you trust the source of this file.
Warning: It is not possible to determine that this content came from a trustworthy source. You should leave this content disabled unless the content provides critical functionality and you trust its source.
More information
File Path: F:...PS\tests OpenXML\tests macros\Word2007 macro DocumentOpen.docm
Help protect me from unknown content (recommended)
Enable this content
2
3
Open the Trust Center
OK
Cancel

New default macros security mode

- In the new default mode "*disable all macros with notification*", **the user can activate any macro with 3 clicks (even unsigned ones).**
- Furthermore, the user can **enable macros AFTER reading the document.**
  - **=> Potential social engineering !**
- As a result, the new default macros security mode is not really more secure than before…
  - For some Microsoft explanations: http://blogs.msdn.com/excel/archive/tags/Trust+Center/default.aspx
- On the other hand, macro source code can be read before enabling the macros.
  - but you must be an experienced developer to understand it.

MS Office 2007 macros storage

- Macros are stored in a **binary OLE2 file**:
  - Word: **word/vbaProject.bin**
  - Excel: **xl/vbaProject.bin**
  - Powerpoint: **ppt/vbaProject.bin**

- **This is not described in the current Open XML draft specifications.**
  - **or have I missed one of the 6036 pages ?**
  - **And OLE2 is not really an open format.**

- Example: automatic launch of a macro from a Word 2007 document (.docm)
  - You only have to name the macro “Document_Open”
  - Word adds a tag in word/vbaData.xml:
    - `<wne:eventDocOpen/>`

OLE objects

- **Open XML documents can embed OLE objects.**
- For example you can store a **macro-enabled** Excel workbook in a **macro-free** Word document.
  - When activated, Excel will ask to enable/disable macros, even if you chose “disable all macros with notifications” !

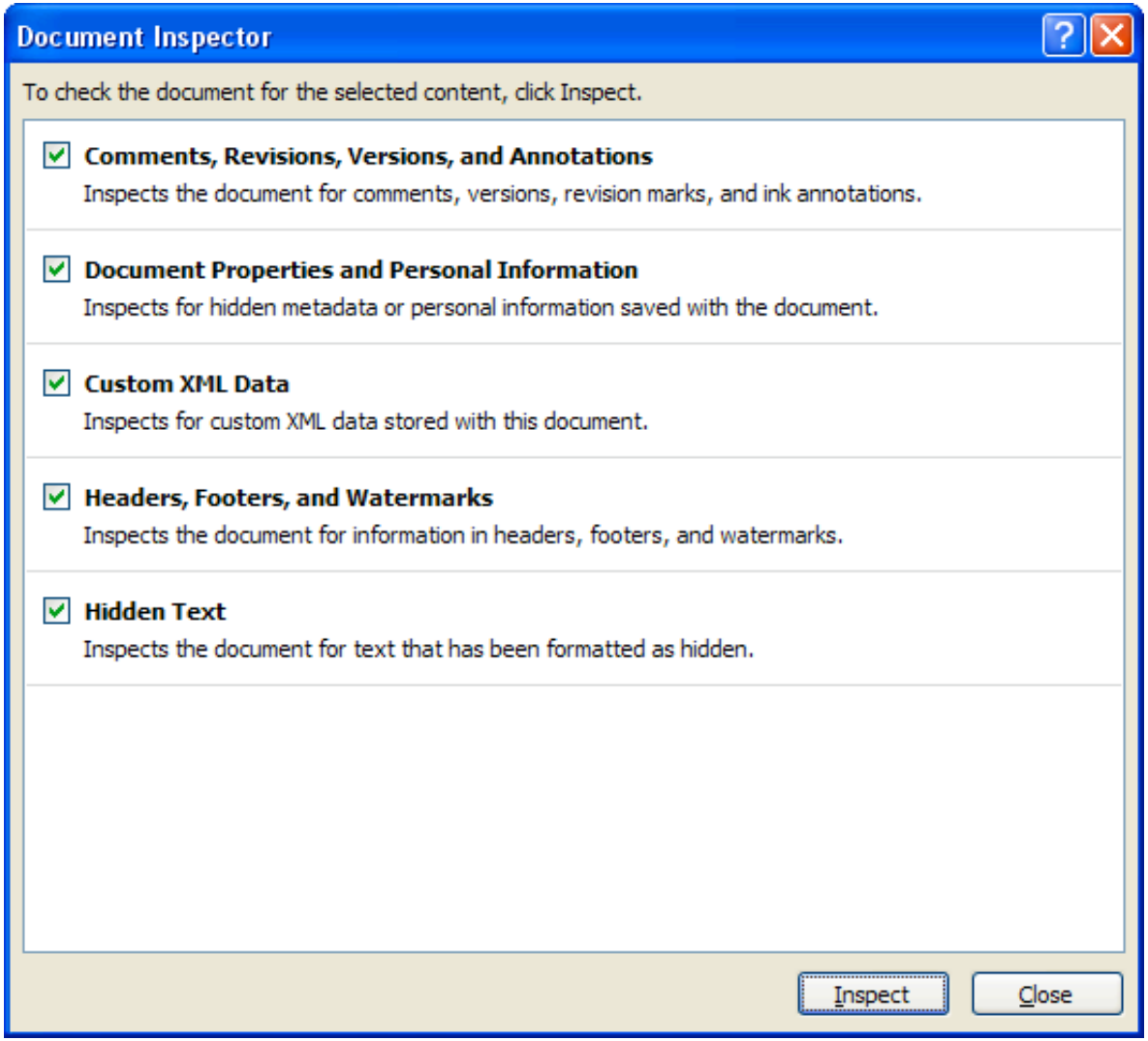
- **An OLE Package object may contain any file or a command line (potential malware).**
  - If the user double-clicks on the object, the file or the command is launched.
  - User warning is up to the operating system (packager.exe).

OLE objects storage

- For example in a Word document, OLE objects are stored in **word/embeddings** inside the archive.
- OLE objects are stored in their native format (for example xlsx in docx).
- **OLE Package objects are stored as binary MS OLE2 files.**

Hidden data removal

- MS Office 2007 provides new features to remove hidden data from documents.
  - “Document Inspector”, improvement of the RHDtool for Office 2003/XP.
- However, OLE objects are not detected as potential hidden data.



MS Office 2007 (beta) security

- Conclusion: **Overall, MS Office 2007 has the same security issues as previous versions (macros, OLE objects, …)**
- **The new *default* macro security mode seems less strict.**
  - **Ability to launch unsigned macros.**
- **Open XML files may contain binary files with a proprietary format : VBA macros, OLE objects, .xlsb, … (not described in current Open XML specifications)**
- **New OpenXML format distinguishes "normal" from "macro-enabled" documents by their name.**
- **Office 2007 provides improved features to remove hidden data from documents.**
- **Open XML makes it easier to detect and filter active content.**

Part 3 :
How to protect information systems

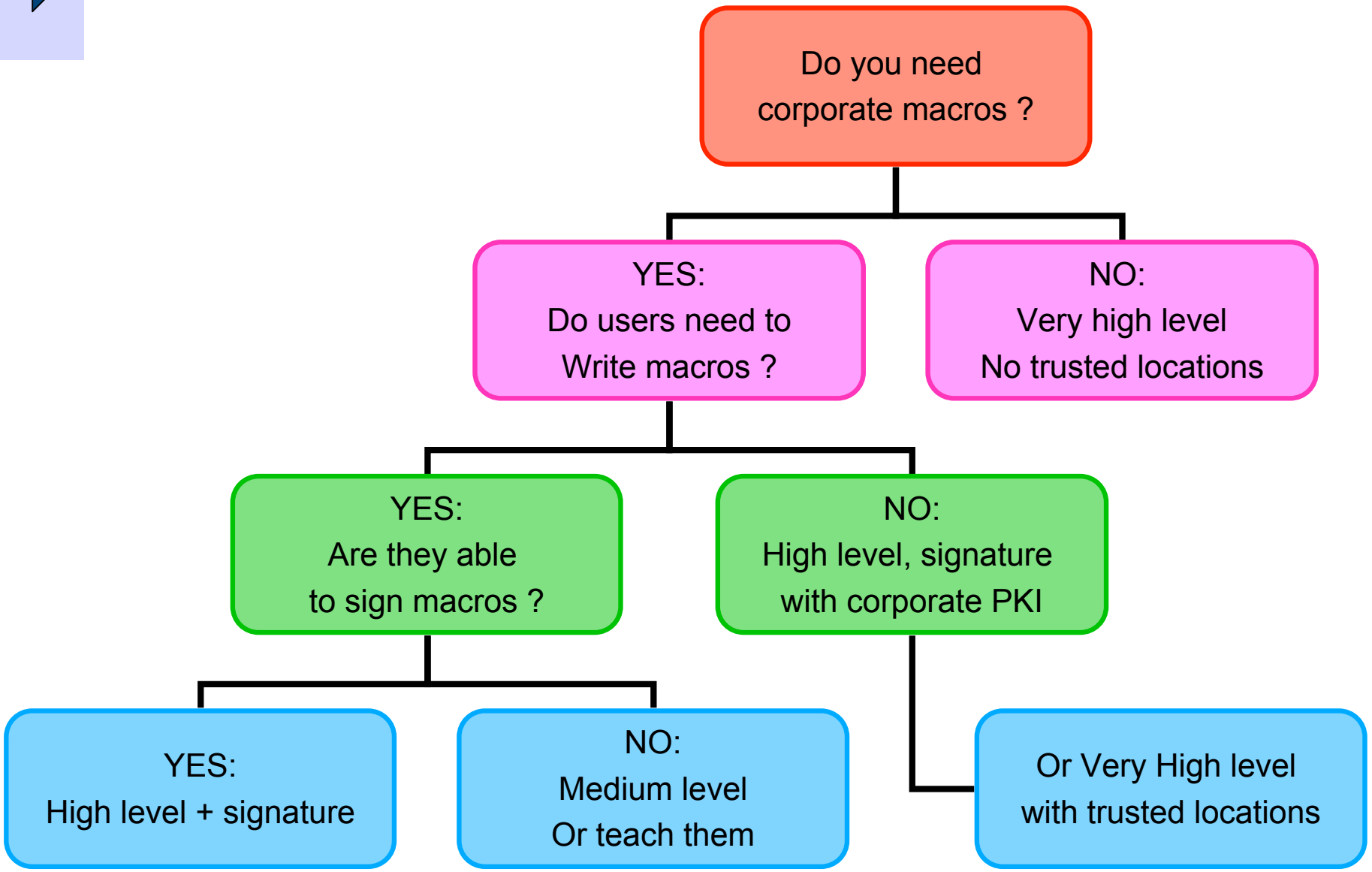
Protection

- 2 ways to protect against security issues:
  - **Secure configuration** of OpenOffice and MS Office 2007.
  - **Filter incoming and outgoing documents**.
    - On a gateway: SMTP, HTTP, FTP proxy
    - On removable media

OpenOffice and MS Office secure configuration

- **Of course, install any security patch or service pack.**
- **Set security parameters according to corporate needs.**
  - Security modes for macros, ActiveX, …
  - Trusted locations
  - Browser security
- **Protect security parameters and trusted locations from end-users.**
  - They should only be writable by admins.
- **Restrict execution permissions of** C:\Windows\System32**Packager.exe if OLE Package objects are not used.**

OpenOffice secure configuration

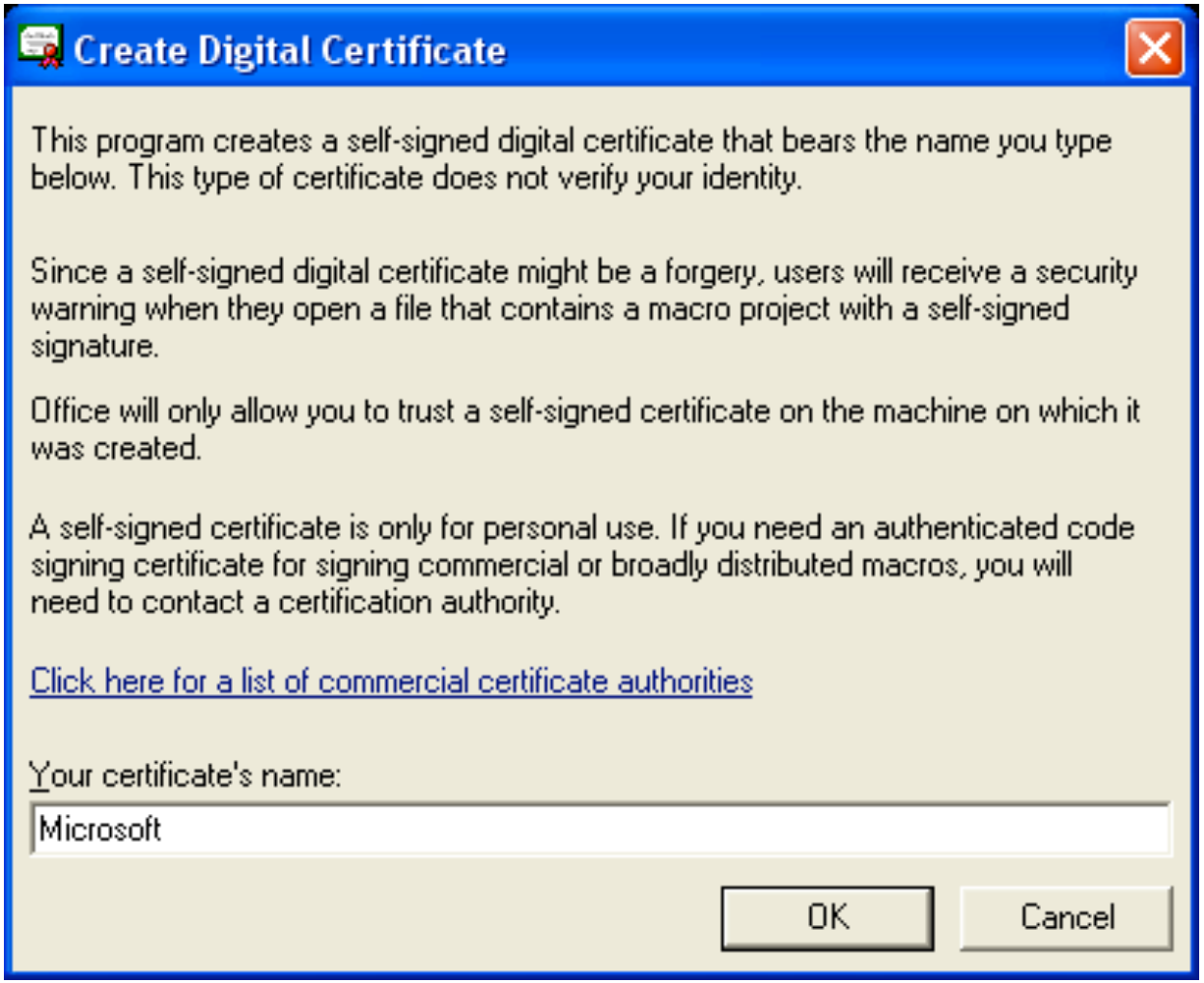


MS Office 2007 secure configuration

- **Choose the highest modes for Macros and ActiveX security.**
  - Macros : choose “disable all macros without notification” if possible.
  - Or “disable all macros except digitally signed” if signature is used.
  - And disable the Message bar notifications to block unsigned macros.
- **Disable Trusted locations if not used**.
  - At least remove user-writable trusted locations, unless users need to write macros and cannot sign them.
- **Use HKLM registry keys to prevent user from changing security parameters.**
  - Wait: Currently this does not work with Office 2007 Beta 2 TR…
- **Use GPO to deploy secure settings (see future Office 2007 Resource Kit)**
  - http://www.microsoft.com/office/ork

What do self-signed certificates certify ?

- It’s easy to forge a certificate with any name…



To avoid sensitive data leak

- **Use OOo and Office 2007 new features to detect and remove hidden data.**
- **Replace OLE objects by static pictures.**
- **If possible, export outgoing documents as PDF.**
  - But beware: PDF may still embed hidden data.

Part 4 :
How to filter OpenDocument and OpenXML files

Antivirus / Active content filter

- An **antivirus** has to analyze all the document contents to detect known malware.
- An **active content filter** is designed to remove all active content (macros, scripts, objects, applets…) from documents.
- **Now both will have to be able to handle OpenDocument and Open XML files.**

OpenDocument active content filter

- To remove active content from OpenDocument:
  - **Macros**: Remove any file in the “Basic” and “Scripts” directories.
  - **OLE objects**: Remove any “Object*” file
  - In “content.xml”:
    - Remove **OLE objects**: `<draw:object-ole>`
    - Remove **scripts :** `<text:script>`
    - Remove **applets :** `<draw:applet>`
    - Remove **plugins :** `<draw:plugin>`
    - Update any tag linked to macros, like **events:** `<office:event-listeners>`

Open XML active content filter

- To remove active content from Open XML:
  - **Macros**: remove all vbaProject.bin and vbaData.xml files
    - (and all other vbaProject / vbaData parts, according to [Content_Types].xml)
    - Update any tag linked to macros: entryMacro, exitMacro, …
  - **OLE objects**: remove all *.bin files
    - (and all other oleObject parts)
  - **Update relationships**

Open XML simple filenames filtering

- At first glance it seems very simple to detect and filter “macro-enabled” Open XML documents:
  - .doc**x**, .xls**x** and .ppt**x**: OK, no macros.
  - .doc**m**, .xls**m**, .ppt**m**: NOK, macros.
- **But this only applies to macros, not to other security issues like OLE.**
- And one can rename a .docm file to .docx, then trick the user into renaming the file before opening it… (otherwise Word won’t open it at all)
- Or worse: **rename .docm to .doc**, the document is silently opened as if it was a .docm…

Quick and dirty filter

- We could simply remove unwanted files from ZIP archives, for example with zip:
  - OpenOffice: `zip -d mydoc.odt` **`Scripts/* Basic/* object*`**
  - Open XML: `zip -d mydoc.docm` **`*.bin`**

- …but beware: zip –d is **case sensitive**, whereas office suites are not !
  - « sCriPts/* » wouldn't be removed
  - To avoid this it would be possible to patch zip source code.

- And we might get some annoying error messages, due to references in XML content.

Another quick and dirty filter

- Another simple filter in Python :
  - (slower due to recompression, but safer)

```
import zipfile, sys
try:
        infile = zipfile.ZipFile(sys.argv[1], "r")
        outfile = zipfile.ZipFile(sys.argv[2], "w")
except:
        sys.exit("usage: %s infile outfile" % __file__)
for f in infile.infolist():
        fname = f.filename.lower()
        if  not fname.startswith("scripts") \
        and not fname.startswith("basic") \
        and not fname.startswith("object") \
        and not fname.endswith(".bin") :
                data = infile.read(f.filename)
                outfile.writestr(f, data)
```

OpenOffice macros renaming

- Macro files can be renamed with **any extension**, if manifest.xml and content.xml are modified accordingly.
  - Example: Scripts/python/BadMacro**.txt**

- **A macro filter should not rely on file extensions for OOo.**
  - Hopefully, we only have to remove everything in the Scripts and Basic directories.

Office 2007 macros renaming

- **Due to the modular structure of Open XML, renaming the VBA macros storage is possible.**
- Example for Word:
  - Rename vbaProject.bin to dummy.txt
  - Update word/_rels/document.xml.rels
  - In [Content_Types].xml, replace “bin” by “txt”
  - …and the macros will work fine !
- **=> Antivirus and filters should not rely only on filenames in Office 2007 documents !**
  - **XML parsing or content analysis is mandatory.**

Open XML « ASCII 7 bit » obfuscation

- Like Internet Explorer, Office 2007 has a rather strange way to handle XML files with ASCII (7 bits) encoding:
  - 8th bit of each character is just silently removed and parsing goes on… !
  - To hide tags you just have to add the « obfuscation bit »:

```
<?xml version="1.0" encoding="us-ascii" standalone="yes"?>
¼HIDDENTAG¾ malware[…] ¼/HIDDENTAG¾
```

  - http://www.securityfocus.com/archive/1/437948

Open XML UTF-7 encoding

- It is also possible to use UTF-7 encoding to hide tags:

```
<?xml version="1.0" encoding="UTF-7" standalone="yes"?>
+ADw-HIDDENTAG+AD4- malware[…] +ADw-/HIDDENTAG+AD4-
```

- **But according to the ECMA specifications, Open XML should only allow UTF-8 or UTF-16 encodings !**
  - Not UTF-7 nor US-ASCII
  - Fortunately MS Office 2007 is still beta…
- (OpenOffice seems to have a stricter XML parser)

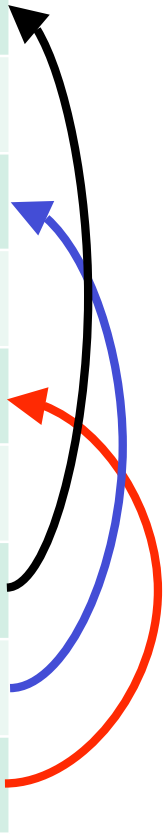
ZIP filenames obfuscation

- **In a ZIP archive, filenames (and other file information) are duplicated:**
  - In each **file header** (before each file compressed content)
  - In the **central directory**, at the end of the ZIP

- **Problem: some applications rely on the central dir, some on the headers…**
  - Very few apps check if both are the same.

ZIP filenames obfuscation: example

File 1 header	Name: **Document.xml**, size: 4000
File 1 content	*(compressed)*
File 2 header	Name: **vbaProject.bin**, size: 1024
File 2 content	*(compressed)*
File 3 header	Name: **HiddenMalware.exe**, size: 16000
File 3 content	*(compressed)*
Central Dir	File 1: **Document.xml**, size: 4000
	File 2: **nothing.xml**, size: 1024
	File 3: **nothing2.txt**, size: **0**



ZIP filenames obfuscation for OOo and MS Office 2007

- **OpenOffice only relies on ZIP central directory filenames:**
  - Any filter/antivirus relying only on ZIP header filenames may be bypassed.
- **MS Office 2007 asks for restoration of the document if there is any incoherence between ZIP filenames.**
  - …if the user confirms, macros come back to life !
  - If a filter/antivirus relies only on central dir OR header filenames, it may be bypassed.

How to design a safe filter / antivirus

- **Use a robust ZIP file library with safety checks**
  - Strict comparison of the central dir and file info (file names, sizes must match)
  - Reject any invalid or unhandled file (Zip64 compression, encryption, …)
- **Use a robust XML parser**
  - Never use simple string or pattern matching, use a real XML parser.
  - Only allow normal encodings (UTF-8, …)
  - Reject any invalid character (strict parsing)
- **Use XML schemas (XSD, RNG) to validate XML files**
- **Reject any incoherent structure:**
  - Open XML named .doc, …
- …easier said than done. ;-)

Conclusion

A quick comparison: OpenOffice OpenDocument and MS Office 2007 Open XML (1)

- Both formats are mainly based on **XML files in a ZIP archive**.
- Both can embed **macros**, with possible automatic launch and ability to write malware.
  - With their default security level, the user might launch any unsigned macro.
  - OOo: with 1 click before seeing the doc content.
  - Office 2007: with 3 clicks after seeing the doc content.
- Both can embed **OLE Package objects** (stored in MS OLE2 binary files), and rely on OS to protect the user.

A quick comparison: OpenOffice OpenDocument and MS Office 2007 Open XML (2)

- **Open XML format seems much more complex than OpenDocument**. (specifications, relationships, features, …)
  - Security analysis and antivirus/filters development are more difficult.
- Both provide interesting **hidden data removal features**.
  - But that does not cover hidden data inside embedded/OLE objects
  - Both provide PDF export
- They allow **different obfuscation tricks (ZIP, XML)** to bypass content filters and antiviruses.

Conclusion

- **OpenDocument and Open XML are very promising new document formats for office suites :**
  - Open specifications, standardization
  - ZIP, XML with provided schemas
- **OpenOffice and Office 2007 provide improved features to remove hidden data from documents**
  - personal information, metadata, …
- **However there are still several ways to embed malware inside these new formats.**
  - Macros, OLE objects, HTML scripts, Applets, ActiveX, …

Conclusion

- **There are also new security issues:**
  - Some **ZIP and XML tricks** may allow obfuscations (against antivirus or filters)
  - **“restore malformed documents”** feature may allow attackers to obfuscate malware.
  - **The new default Office 2007 macros security mode** seem less strict than before.
  - **It will take time before these new formats are safely handled by all antivirus and content analysis software.**

Conclusion

- **Thanks to open formats, it's now easier to detect active content inside documents.**
  - **Designing filters seems straightforward thanks to ZIP and XML, but it's not.**

- OOo and MS Office developers ought to fix a few bugs to avoid some issues, and they should provide more security features in their products.
  - (Strict ZIP/XML parsing, OLE blocking, administrator parameters to deploy a hardened configuration, …).

The End

<? Any questions ?>

See http://pacsec.jp/pastevents.html for updates

(Random) Bonus slides

Data leak real life example

- One day we were looking at a Powerpoint file, coming from a well-known vendor.
- double-clicked on a nice 3D diagram
- it was a complete Excel spreadsheet in an OLE object.
  - Full of confidential figures and equipment prices !
  - Fortunately we're not bad guys. ;-)

Reminder from the “past”: Some MS Office 97-2003 security issues

- **Macros**
  - Powerful API which allows to write malware
  - Possible automatic launch (doc open/close, …)
- **OLE objects**
  - “Package” objects may contain any file, or launch any command with cmd.exe
- **Data leak**
  - Metadata, revision marks, comments, hidden text, fields, embedded documents, …

OpenOffice - UNO

- **Each macro language gives access to UNO:**
  - UNO: Universal Network Objects
  - **Very powerful API**: access to OpenOffice objects and the operating system
  - **Ability to write effective malware**.

- UNO can also be used from programs outside documents (from C++, .NET, Java, Python, …)
  - OpenOffice acts as a server, controlled by a client application through UNO calls.
    - (outside the scope of this analysis)

OPC: Open Packaging Conventions

- Open XML documents are **OPC files**:
  - **XML files in a ZIP archive**
    - (and optionally binary files)
  - **Modular structure with “relationships”**
    - Indirect references between objects (parts).
    - Relationships in “.rels” XML files
  - A little more complex than OpenDocument

- Other OPC formats:
  - **XPS**: new Microsoft format, similar to PDF.

Excel 2007 binary files

- .xlsx : workbook without macros (default)
- .xlsm: macro-enabled workbook
- **.xlsb: binary workbook**
  - Designed for better performance than XML
  - Same ZIP structure as .xlsx/.xlsm
  - **XML data files are replaced by binary files** (BIFF format)
    - Except relationships (.rels), metadata, …
  - **May contain macros (just like .xlsm)**

HTML Scripts in Office documents

- **MS Office 2003** allows the storage of HTML scripts in documents (with Script Editor)
- These scripts are only activated when the document is saved as HTML/MHTML, and opened in a browser.
  - Just like OpenOffice.
- **MS Office 2007 B2TR** does not currently give access to Script Editor.
  - For now it seems impossible to store HTML scripts in Open XML documents, but…
- Scripts are still handled when an Office 2003 document is saved to HTML by Office 2007.

How to sign a trusted macro for Office 2007

- Use « MS Office tools / Digital Certificate for VBA projects » to **create a self-signed certificate**.
  - These certificates also work for OpenOffice.
- Then **sign your trusted macro** with VBA Editor / Tools
- And **approve your certificate**.

How to use an unsigned trusted macro

- If you can't / don't want to sign a macro:
- **Put the document into a trusted location.**
- By default (example for Word):
  - C:\Program Files\Microsoft Office\Templates
    - Writable by administrators and power users only
  - C:\Documents and Settings\user\Application Data\Microsoft\Templates
    - Writable by user only
  - C:\Documents and Settings\user\Application Data\Microsoft\Word\Startup
    - Writable by user only
- **In a corporate environment, it would be wise to trust only admin-writable locations.**
- **And to protect MS-Office security parameters from users: use HKLM registry instead of HKCU.**

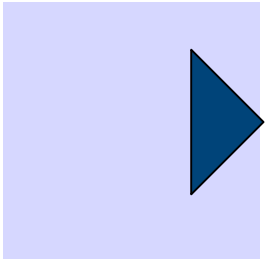
Obfuscation techniques

- Malicious people goal: to bypass antivirus and gateways checks

- Each file format may allow obfuscation features to fool filters.
- Examples for HTML:
  - UTF-8 encoding (with illegal encoded ASCII characters)
  - Fake script tags inclusion
    - `<SCR<script>remove me</script>IPT>…`

.NET API: System.IO.Packaging

- The new System.IO.Packaging API provides Open Packaging Conventions files access to .NET applications.
  - Open XML and XPS documents.
- Allows to develop Open XML security filters.
  - For example, Microsoft provides a VBA removal code snippet.
- But this only works on Windows, whereas many gateways are based on other systems.
- And of course this is not usable for OpenDocument.



OOoPy

- OOoPy is a useful Python package to open and modify OpenDocument files.
  - A simple combination of a ZIP reader/writer with a XML parser.
  - Can be used to handle Open XML files also.
- May be used to design portable security filters.

ZIP compression

- Recently new compression algorithms were added to the ZIP standard.
  - **Open XML specification explicitly allows the use of Zip64 compression.**
- But most free ZIP libraries only support standard Deflate compression.
  - Any unsupported archive should be rejected.

XML schema validation

- OpenDocument and Open XML specifications provide XML schemas (XSD or Relax-NG).
- It should be easy to validate XML content.
- You have to use a validating XML parser, with XML namespaces handling.
- It’s even possible to use modified schemas without unwanted tags.
- But beware of documents with custom schemas or Open XML “Markup compatibility and extensibility” features.