Collabora

LibreOffice
The Document Foundation

SUSE

OOXML / PDF Digital Signing in Draw and elsewhere

By Miklos Vajna

Software Engineer at Collabora Productivity

2020-10-16

MySQL LINUX datto ADMIN fedora TUXEDO COMPUTERS .riessapplications Required Magic advanced technology

About Miklos

From Hungary

- More details: https://www.collaboraoffice.com/about-us/

Google Summer of Code 2010 / 2011

- Rewrite of the Writer RTF import/export

Then a full-time LibreOffice developer for SUSE

Now a contractor at Collabora

Digital signing in Draw and elsewhere

ODF signing

“The” document signing we inherited from OOo

- Can sign ODT, ODS, ODP, ODG
  - MD5 and SHA1 only
  - RSA only
- Verification
  - Checks if the digest matches
  - Validates the certificate
  - Checks if the whole document is signed
  - Based on X509 certificates

OOXML signing

Added in 2016, LibreOffice 5.2

- Based on the [xmldsig-core] specification, similar to ODF
- Does not sign metadata, has separate files for each signature
- Interoperable with MSO
- Has its own “Relationships Transform Algorithm”, now libxmlsec supports this
- Leaks software / hardware details

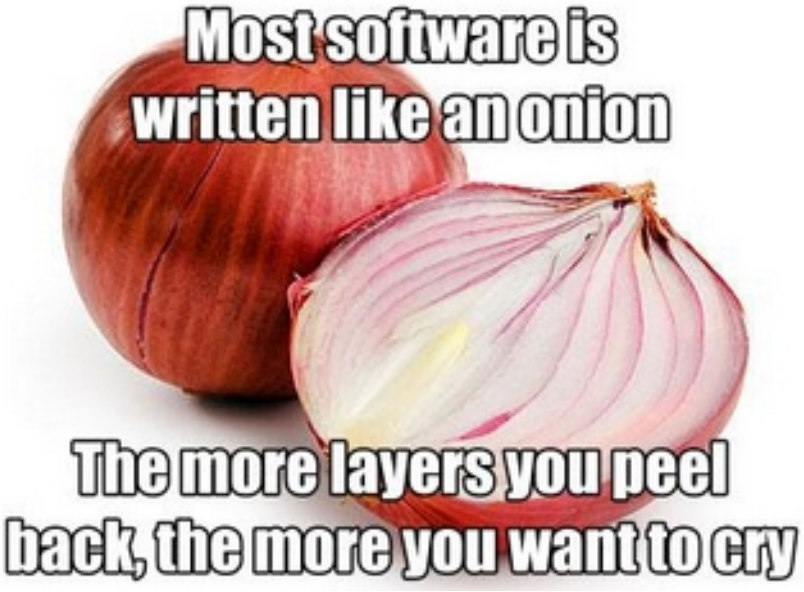
```
<WindowsVersion>6.1</WindowsVersion>
<OfficeVersion>16.0</OfficeVersion>
<ApplicationVersion>16.0</ApplicationVersion>
<Monitors>1</Monitors>
<HorizontalResolution>1280</
HorizontalResolution>
<VerticalResolution>800</VerticalResolution>
<ColorDepth>32</ColorDepth>
```

Signing during PDF export

Started by Gökçen Eraslan, finished by Tor Lillqvist in 5.0

- Started as a GSoC project
- Finished by Collabora, sponsored by Wilhelm Tux crowd-funding
- Writer a placeholder to the PDF file, then hash what's before and after
- Do a standard PKCS#7 binary signature on the hash, write hexdump to the placeholder
- Handles new PDF files, single signature

Verification of PDF signatures



Needed a whole new PDF parser, new in 5.3

- All existing ones were problematic back then:
  - Poppler is not available in MPL subset builds
  - Own boost-based parser (used for hybrid PDF) is hard to extend
  - PDFium did not have a signature API back then
- Basic verification is simple: parse the PKCS#7 hexdump, and the data before/after the signature has to be hashed
- Multiple signatures are chained by definition, and technically only the last signature can be complete, which is sad

Signing existing PDFs

Builds on top verification, since need the same PDF parser, new in 5.3

- Adds an incremental update to the document, to not break existing signatures
- Works with Acrobat-created PDF 1.5 files:
  - Supports cross-reference streams
  - Supports object streams
  - Supporst stream predictors
- Lots of corner-cases, but at the end appending signature in both Acrobat and LibreOffice in any order is meant to work

XML Advanced Electronic Signatures (XAdES)
PDF Advanced Electronic Signatures (PAdES)

A set of extensions to the XML-DSig recommendation / PDF spec, new in 5.3

- If all conditions met, then this can result in a legally binding signature
- SHA-256 support has to be added
- ECDSA support has to be added
- Had to make sure that the signing certificate is part of the signed data
- PAdES passes the DSS validator

Signing existing PDFs: visible signatures

Replacing a stub widget with an actually visible signature, new in 7.1

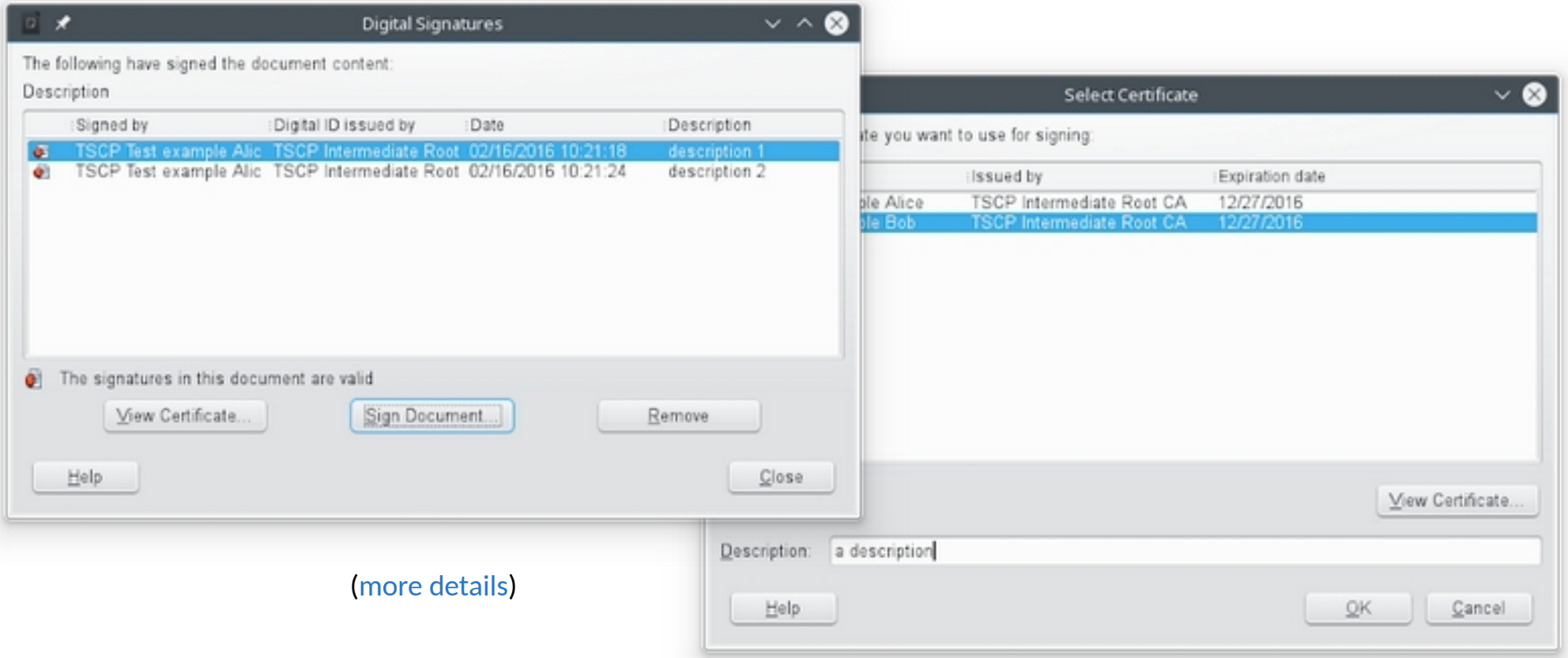
- UI is similar to signature lines, the result is a vector graphic (DocuSign doesn’t seem to do this)
- Used PDF markup associates the visible signature with the digital signature, helping a11y (DocuSign not doing this)
- The created signature shape can be resized and repositioned before signing (Acrobat has problems here)

How is this implemented?

Signature descriptions

ODF: store a description next to the date

- OOXML & PDF already had markup for this
- Also called comment or reason



(more details)

OOXML signature import

Performing exactly the same hashing as MSO, needed:

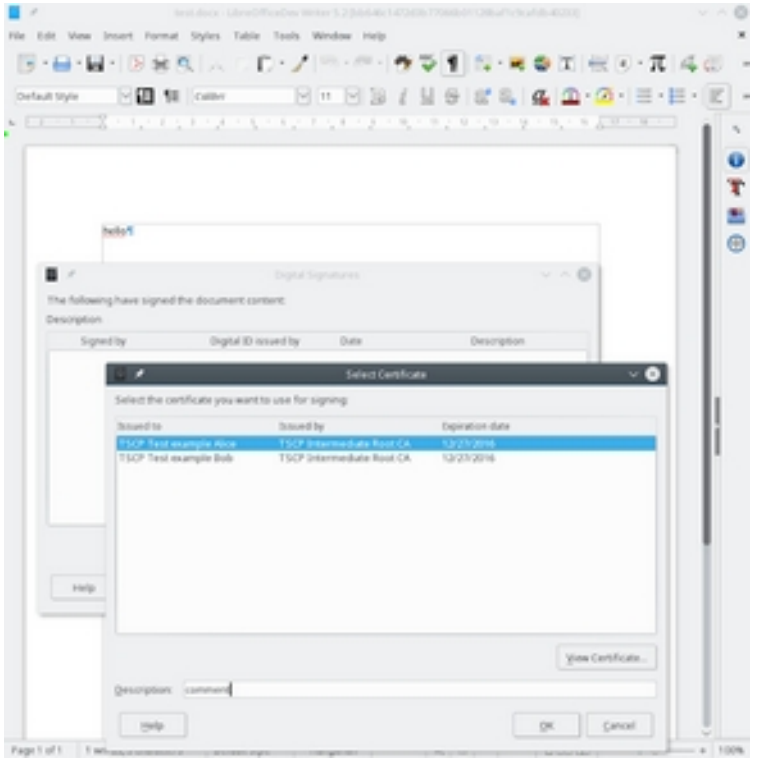
- support for the Relationships Transform Algorithm (described in ISO/IEC 29500-2:2012) in libxmlsec
- an actual XML parser for the OOXML signature in xmlsecurity/
- a new filter flag, so that our code no longer assumes "is ODF" means "supports digital signing" and
- some refactoring in xmlsecurity/, so that our digital signature code doesn't assume that multiple signatures are always written to a single file

(via ascertia, more details)

OOXML signature export

Builds on top of the import side:

- signing a previously unsigned document
- appending a signature to an already signed document
- removing a signature from a document with multiple signatures
- removing the last signature of a signed document, turning it into an unsigned one
- [Content_Types].xml has to mention the .sigs extension and the individual /_xmlsignatures/sigN.xml streams
- _rels/.rels has to refer to _xmlsignatures/origin.sigs, which refers to the individual signatures
- DigitalSignaturesDialog has less code, factored out to a DocumentSignatureManager, so it can be tested from cppunit

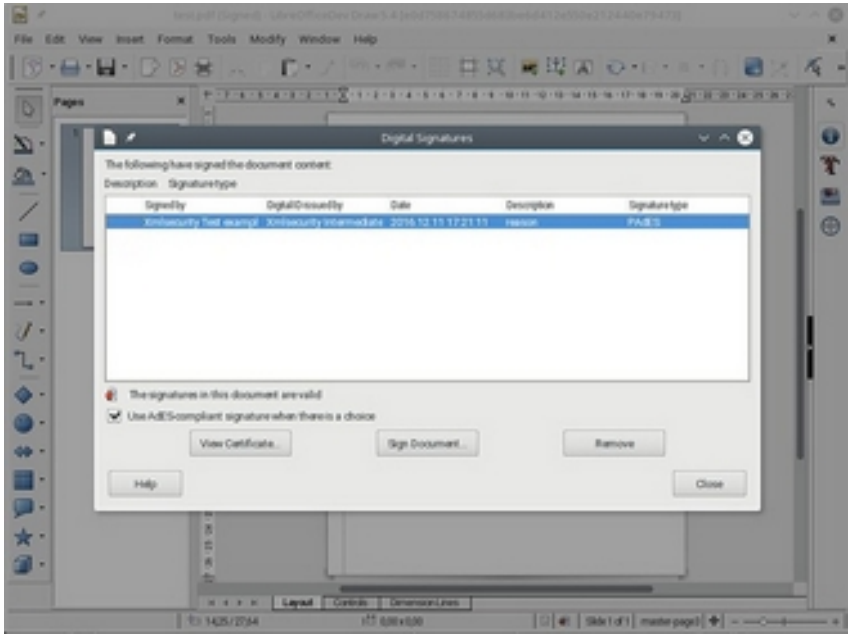


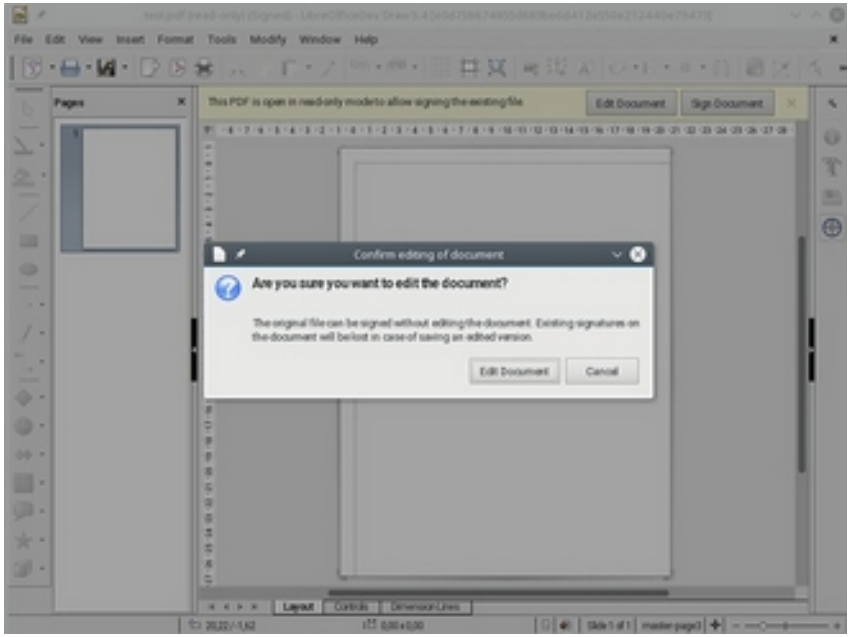
(more details)

Verifying existing PDF signatures

Parsing the PDF, then verify the signature

- File → Digital signatures → Sign exiting PDF
- Verification happens unconditionally, when opening a PDF file
- Discouraging editing when the file is opened for signing



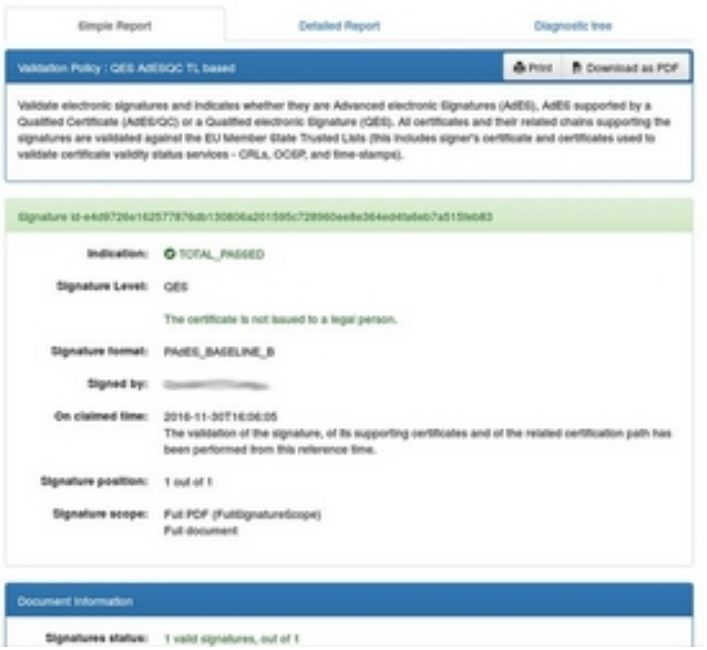


(more details)

Adding PAdES support

Towards a legally binding signature:

- PDF signature creation now defaults to the stronger SHA-256 (instead of the previously used weaker SHA-1), and the PDF verifier understands SHA-256
- the PDF signature creation now embeds the signing certificate into the PKCS#7 signature blob in the PDF, so the verifier can check not only the key used for the signing, but the actual certificate as well
- the PDF signature import can now detect if such an embedded signing certificate is present in the signature or not

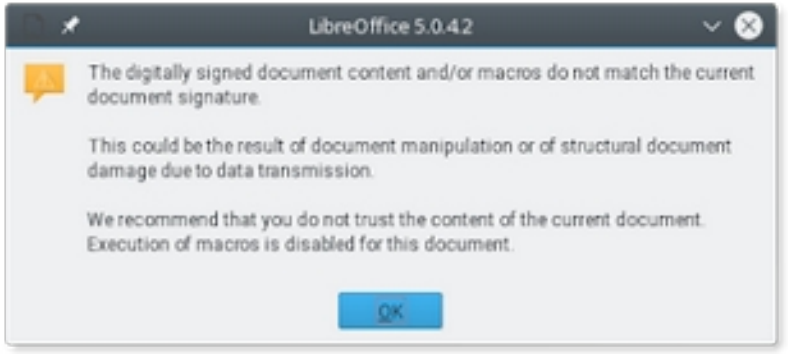


The DSS validator (more details)

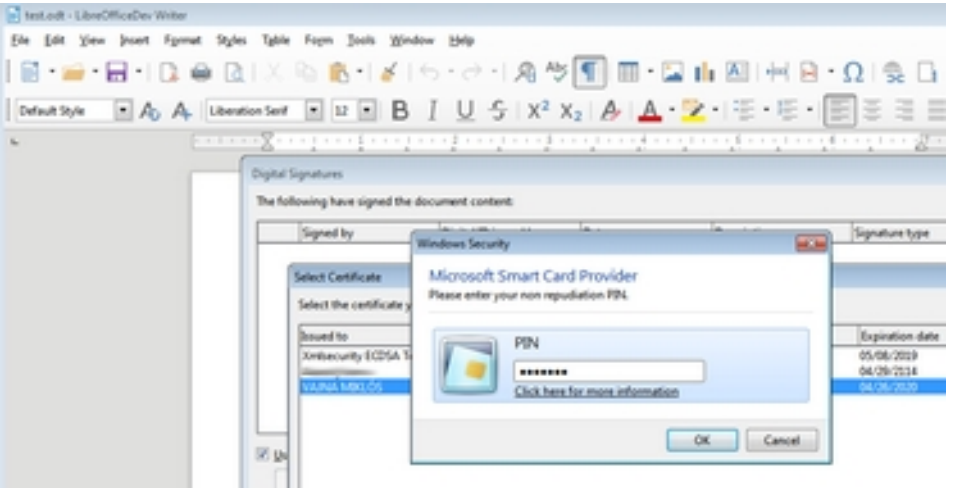
Adding support for SHA-256 and ECDSA

Because real-word HW-based certificates often use those:

- SHA-256 can be a signature method or a digest method, different URIs
- First added patches to our old bundled libxmlsec to support this, on top of SHA-1, then updated libxmlsec
- ECDSA support some generic work, and also required switching to MSCNG on Windows, rewriting half of the xmlsecurity/ Windows code



(SHA-256 details)

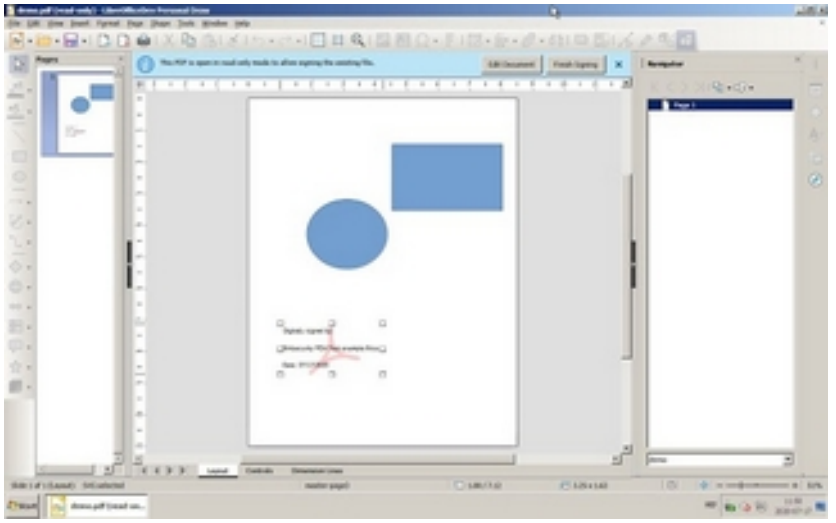


(ECDSA details, CNG details)

Visible PDF signing

As always tries to reuse existing code:

- Signature lines were already working in Writer and Calc, this effort brings them to Draw, improving consistency.
- The generated object is locale-aware when it comes to the actual signature string and date format.
- The feature works for multiple signatures and multiple pages as well.
- Draw the signature rectangle, finalize it, then Finish signing:



(more details)

Thanks

Collabora is an open source consulting and product company

- What we do and share with the community has to be paid by someone

The Dutch Ministry of Defense in cooperation with Nou&Off

- Made this work by Collabora possible

Summary

Good digital signature support of ODF, OOXML & PDF

- Including signature descriptions, XAdES & PAdES
- Modern hash & encryption algorithms: SHA-256 & ECDSA
- Interoperable with MS Office & Adobe Acrobat
- Latest news is visible PDF signatures

Thanks for listening! :-)

- Slides: https://people.collabora.com/~vmiklos/slides/