

Documentation

About open-pdf-sign	2
License	2
Open-pdf-sign CLI	3
Features	3
Use open-pdf-sign	3
Run	3
Usage with Let's Encrypt certificates	4
Visible signatures	4
Usage in server mode	5
Using a config file	5
Development documentation	5
Requirements	5
Build	6
Open-pdf-sign npm module	7
Prerequisites	7
Installation	7
Usage	7
Development documentation	7
Open-pdf-sign-configurator	8
Requirements	8
Installation instructions	8
Requirements	8
Installation	8
1. SSH into your server	8
2. Install Dependencies	8
3. Install openpdfsign-configurator	8
4. Run the Configurator	9
5. Confirm that its working	9
Development documentation	9
Testing	9
Release	10
Testing a release	10

About open-pdf-sign

Our software, open-pdf-sign allows to sign PDF files digitally from CLI or a webserver. There are special certificates needed, users can just reuse their already existing certificates, e.g. from Let's Encrypt.

Files are cryptographically signed following the PAdES-B and PAdES-T standards. This lets site visitors know that PDFs were actually served by the visited server. Users can append a visible or invisible signature. Visible signatures can be further customized. Finally, open-pdf-sign easily integrates with existing Nginx web server or can be used in Node.JS projects.

Open-pdf-sign is released under the Apache 2.0 Open Source license, allowing anyone to use and modify the code. It's free to use and distribute.

License

This document is licensed under CC-BY-SA

Open-pdf-sign CLI

The open-pdf-sign CLI application allows to easily sign PDF files from the commandline. Signatures can be invisible or visible, and visible signatures can be customized.

Features

- Visible PDF signature in PDF (multi language support)
- Invoke via CLI or via starting a server
- Supported Signature type: PAdES
- Supported Signature profiles:
 - BASELINE-B
 - BASELINE-T
 - To be evaluated: BASELINE-LT, BASELINE-LTA

Use open-pdf-sign

Download the latest jar archive from the [GitHub releases page](#) or in your terminal:

```
curl -L
https://github.com/open-pdf-sign/open-pdf-sign/releases/latest/download/op
en-pdf-sign.jar -o open-pdf-sign.jar
```

Make sure that Java is installed in at least version 8.

Run

```
java -jar open-pdf-sign.jar -i input.pdf -o output.pdf -c certificate.crt
-k keyfile.pem -p key_passphrase --page -1 --locale de-AT
```

Usage:

Options:

```
-b, --binary
    binary output of PDF
    Default: false
-c, --certificate
    certificate (chain) to be used
--config
    use a configuration file
--hint
    text to be displayed in signature field
--host
    run as server with the given hostname
--image
    Image to be placed in signature block
```

```

-i, --input
    input pdf file
-k, --key
    signature key file or keystore
--left
    X coordinate of the signature block in cm
    Default: 1.0
-l, --locale
    Locale, e.g. de-AT
-o, --output
    output pdf file
--page
    Page where the signature block should be placed. [-1] for last page
-p, --passphrase
    passphrase for the signature key or keystore
--port
    run as server with the given port
    Default: 8090
--timestamp
    include signed timestamp
    Default: false
--timezone
    use specific timezone for time info, e.g. Europe/Vienna
--top
    Y coordinate of the signature block in cm
    Default: 1.0
--tsa
    use specific time stamping authority as source (if multiple given,
will
    be used in given order as fallback)
    Default: []
--width
    width of the signature block in cm
    Default: 10.0

```

Usage with Let's Encrypt certificates

PDFs can also be signed using your existing Let's Encrypt certificate.

```

java -jar open-pdf-sign.jar -i input.pdf -o output.pdf \
  -c /etc/letsencrypt/live/openpdfsign.org/fullchain.pem \
  -k /etc/letsencrypt/live/openpdfsign.org/privkey.pem


```

Visible signatures

If the page parameter is specified, a visible signature will be placed on the specified page. For example, running

```
java -jar open-pdf-sign.jar -i input.pdf -o output.pdf \
  -c certificate.crt \
  -k key.pem \
  --page -1 --logo mylogo.png \
  --hint "You can check the validity at signaturpruefung.gv.at"
```

will place a visible signature looking similar to the image below on the last page (-1) of the PDF document.

	Signee	commonName=open-pdf-sign test,organizationName=open-pdf-sign test,stateOrProvinceName=Vienna,countryName=AT
	Timestamp	2022-07-24T19:27:59.752+02:00
	Hint	You can check the validity of this certificate at https://www.signaturpruefung.gv.at

Usage in server mode

You can also run open-pdf-sign as a server application in order to only load certificates once and easily integrate it in applications where CLI invocations are not possible. Simply add the --port or --hostname parameters, e.g.

```
java -jar open-pdf-sign.jar -i input.pdf -o output.pdf \
  -c /etc/letsencrypt/live/openpdfsign.org/fullchain.pem \
  -k /etc/letsencrypt/live/openpdfsign.org/privkey.pem
  --port 8090 --hostname 127.0.0.1
```

Then, PDFs can be signed via the [specified](#) /pdf endpoint:

```
curl --location --request POST 'http://localhost:8090/' \
  --header 'Content-Type: application/json' \
  --data-raw '{
  "input": "/path/to/pdf.pdf"
}'
```

Using a config file

Instead of CLI parameters, you can also submit a configuration file with the same parameters and the possibility to lead multiple keys, as shown in [this example](#)

```
java -jar open-pdf-sign.jar --config /path/to/config.yaml
```

Development documentation

Requirements

- Maven (<https://maven.apache.org/>)
- JDK 8

Build

The package can be built running

```
mvn package
```

During the build, all unit tests are executed.

Open-pdf-sign npm module

Node.js wrapper for the [open-pdf-sign CLI](#). This allows you to easily sign PDF files from your node.js applications.

Prerequisites

- nodejs
- java

Installation

```
npm i open-pdf-sign --save
```

Usage

See [demo/sample.js](#) file for how to use with.

You can use all command line parameters from the [open-pdf-sign Java CLI](#), including visible signatures, signatures text, etc.

```
const OpenPdfSign = require("open-pdf-sign")
```

```
await OpenPdfSign.sign("-i demo.pdf", "-o demo.signed.pdf", "-k  
key.pem", "-c cert.pem")
```

You can also use open-pdf-sign in your TypeScript files:

```
import { OpenPdfSign } from "open-pdf-sign"
```

```
await OpenPdfSign.sign("-i demo.pdf", "-o demo.signed.pdf", "-k  
key.pem", "-c cert.pem")
```

Development documentation

After installing all dependencies and downloading the latest version of the *open-pdf-sign* CLI application by running `npm i`, run `npm run build` to build the typescript files.

Open-pdf-sign-configurator

open-pdf-sign-configurator is a tool that simplifies the process of setting up and configuring open-pdf-sign on an nginx server. It automates many of the steps involved in the installation and configuration process, making it quick and easy for users to get open-pdf-sign up and running on their web servers. This can help save time and reduce the potential for errors, allowing users to quickly and easily sign PDFs on their web server.

After you install and activate it, all PDFs served by your Nginx server will automatically be signed by your SSL certificate.

Requirements

- Java 8 or higher
- python 3.9 or higher
- Webserver (currently only Nginx supported)
- SSL certificate (tested with letsencrypt)

Installation instructions

Following these instructions will make all PDFs served by your webserver signed by your SSL certificate

Requirements

- comfort with the command line
- a Nginx webserver that is online and running debian or Ubuntu
- a LetsEncrypt or other SSL certificate installed with Nginx
- SSH access to your server with sudo rights

Installation

1. SSH into your server

SSH into the server running your HTTP website as a user with sudo privileges.

2. Install Dependencies

```
apt-get install default-jre python3 python3-pip
```

3. Install openpdfsign-configurator

```
pip install openpdfsign-configurator
```


4. Run the Configurator

opsconfig

You can select the domain names where open-pdf-sign will be installed

5. Confirm that its working

Go to the URL where you are currently serving a PDF Download it and check your signature i.e. here: <https://ec.europa.eu/cefdigital/DSS/webapp-demo/validation>

Development documentation

- go to the "installer" folder
- create a virtual environment
- do pip install -r requirements.txt

In order to develop, you can simply run the main.py file, just make sure to change the "stage" from "prod" to "dev"

Testing

This will create a docker container with:

- Nginx server running
- A certificate for test.com
- all dependencies installed

go to the file main.py in directory installer and change "prod" to "test"

```
cd installer
```

```
python -m build
```

```
cp dist/*.whl ../test/installerdev
```

```
cd ../test/installerdev
```

```
vim Dockerfile #(change the version of the whl file if necessary)
```

```
docker build -t nginx .
```

```
docker run --name installertest -d -p 80:80 -p 443:443 nginx
```

```
docker exec -it installertest bash
```

```
opsconfig
```

```
java -jar /etc/openpdfsign/openpdfsign.jar --config  
/etc/openpdfsign/config.yml
```

visit: <https://127.0.0.1/test1.pdf>

Release

- `python -m build`
- `twine upload -r testpypi dist/*`
- `twine upload -r pypi dist/*`

Testing a release

This will test the pypi test pip package. In order to test you can run the following commands.

```
cd test/installerrelease
```

```
docker build -t nginx .
```

```
docker run --name installertest -d -p 80:80 -p 443:443 nginx
```

```
docker exec -it installertest bash
```

```
opsconfig
```

```
java -jar /etc/openpdfsign/openpdfsign.jar --config  
/etc/openpdfsign/config.yml
```

visit: <https://127.0.0.1/test1.pdf>