Simple Digital Signature Validation on PDF

Rizky Satrio · Follow
Published in Javarevisited · 4 min read · Dec 6, 2021

. . .

1. Introduction

Based on wikipedia, digital signature(DS) is a mathematical scheme for verifying the authenticity of digital messages or documents. It have been used as a tool for non-repudiation.

The type of digital signature put in PDF are varied from a message digest to another type signature (CMS Advanced Electronic Signatures or CAdES). This article will explain more on how to validate the digital signature inside a PDF file.

. . .

2. Problem Statements

- Digital Signature Validation on PDF
- Type Signature: CAdES Detached, PKCS7 Detached, or ETSI.RFC3161

. . .

3. Digital Signature Inside PDF

When you open a PDF File in a text editor, you will find structures with tag in it, somewhat like a xml file. Digital signature is placed inside the Type Sig tag(Signature Dictionary). Try to find this kind of tag /Type /Sig in a PDF File. Below is an example of it:

```
/Type /Sig
/Filter /Adobe.PPKLite
/SubFilter /ETSI.CAdES.detached
/Name [redacted]
/M (D:20210726213300+07'00')
```

Example of Digital Signature Inside PDF

In there you can also see a Filter and SubFilter tag. For further explanation of the value in that tag, you can check the PDF 32000–1:2008 document section 12.8.1. Looking further down inside the Sig dictionary, you will find a Contents tag. Inside that tag is the Based64 value of the digital signature. The example below show a CAdES signature inside the contents tag. Pay attention also the ByteRange tag, we will also use it in the digital signature validation process.

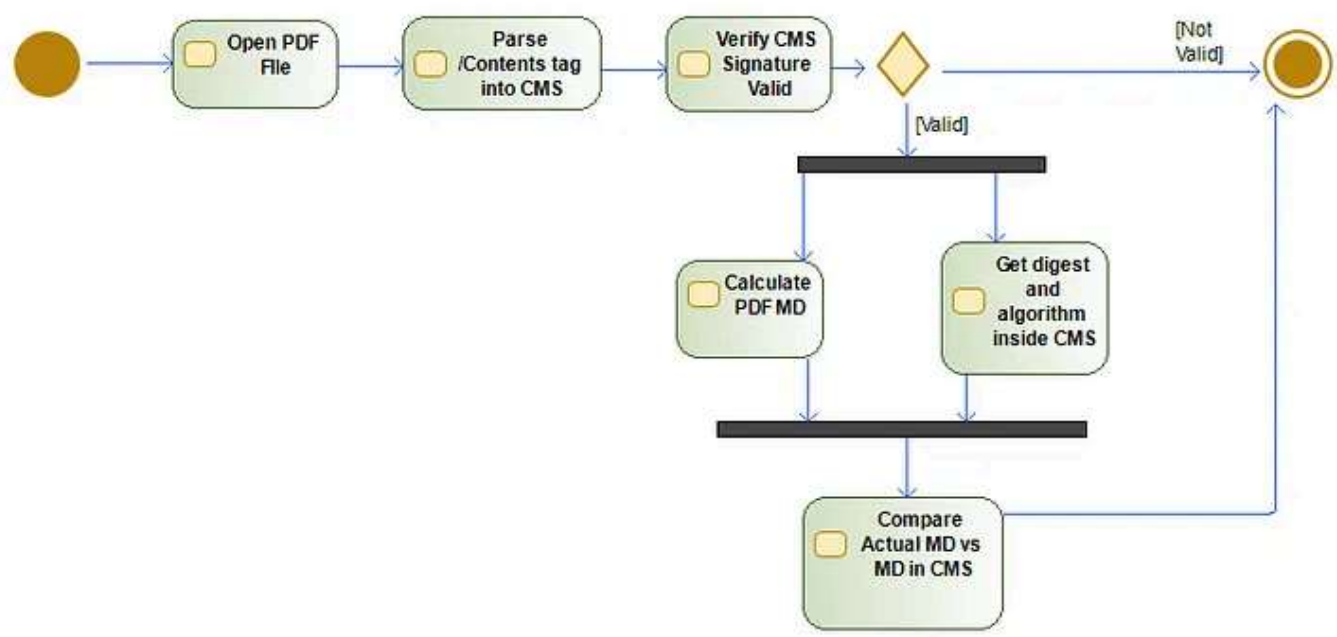
```
/Contents <30821A2B06092A864886F70D010702
/ByteRange [0 29835 48781 27288]
```

Example of digital signature contents tag

. . .

4. Digital Signature Validation

We will focus on PAdES-B Signature on PDF (/SubFilter type ETSI.CAdES.detached). It is basically using CAdES-detached signature that put into the Contents tag. The steps for validation are depicted in below figure.



PDF Signature Validation Process

1. Parse the content in /Contents tag into CMS (Cryptography Message Syntax)
2. Verify that the signature inside CMS is valid (compare it with the signed attribute inside CMS, using certificate in signerInfo Type)
3. If valid, then get the message digest from the signed attribute inside CMS. Also get the digest algorithm used (SHA1,SHA256,etc)
4. Calculate the actual message digest inside PDF (using the byterange and digest algorithm from no.3)
5. Compare number (3) and no(4), if it is valid, then the digital signature value is valid

To be noted the steps above did not handle digital certificate validation, CRL checking, or OCSP checking. We will talk about that in another article. So, let's breakdown the steps above into java code. The actual code can be seen in my github. We will be using 2 external libraries: pdfbox and bouncycastle.

4.1 Parsing CMS Content inside /Contents tag

First open the pdf and get the signature with this line of code:

```
ByteArrayInputStream pdfBytes=new ByteArrayInputStream(
                    Files.readAllBytes(Paths.get(pdfFile.getAbsolutePath())));

pdfDoc=PDDocument.load(pdfFile);

pdfDoc.getSignatureDictionaries().forEach(signature-> {
                try {
                     //Get PKCS#7 Data
                     CMSSignedData signedData=new CMSSignedData(signature.getContents());
                    }
```

OpenPdf.java hosted with ❤ by GitHub view raw

Then get the /Contents tag inside the signature:

```
//Get PKCS#7 Data
        CMSSignedData signedData=new
CMSSignedData(signature.getContents());
```

4.2 Verify CMS Signature is valid

First, we acquired the signerInfo inside CMS:

```
//Get SignerInfo
SignerInformation
signerInfo=signedData.getSignerInfos().iterator().next();
```

Then, we acquired the Public Key inside CMS:

```
//Getting PublicKey
Collection<X509CertificateHolder> matches = signedData.getCertificates().getMatches(signerInfo.get
byte[] pubByte=matches.iterator().next().getSubjectPublicKeyInfo().getEncoded();

X509EncodedKeySpec keySpec=new X509EncodedKeySpec(pubByte);
KeyFactory kf = KeyFactory.getInstance("RSA");
PublicKey pubKey=kf.generatePublic(keySpec);
```

GetCMSPublicKey.java hosted with ❤ by GitHub view raw

Then, we acquired the signature algorithm:

```java
if(signerInfo.getEncryptionAlgOID().trim().equals("1.2.840.113549.1.1.1")) {
            encAlgo="RSA";
 }

if(encAlgo!=null)   {
if(digest.getAlgorithm().equals("1.3.14.3.2.26")) {
    encAlgo="SHA1withRSA";
}
else if(digest.getAlgorithm().equals("2.16.840.1.101.3.4.2.1"))    {
    encAlgo="SHA256withRSA";
}
else if(digest.getAlgorithm().equals("2.16.840.1.101.3.4.2.2"))    {
    encAlgo="SHA384withRSA";
}
else if(digest.getAlgorithm().equals("2.16.840.1.101.3.4.2.3"))    {
    encAlgo="SHA512withRSA";
}
```

GetSignatureAlgorithm.java hosted with ❤ by GitHub view raw

Then, we check the validity of the signature inside CMS:

```
Signature rsaSign=Signature.getInstance(encAlgo);
        rsaSign.initVerify(pubKey);
        rsaSign.update(signerInfo.getEncodedSignedAttributes());
        boolean
cmsSignatureValid=rsaSign.verify(signerInfo.getSignature());
```

4.3 Get the Message Digest Algorithm and the message digest data inside CMS

```java
MessageDigest digest=MessageDigest.getInstance(signerInfo.getDigestAlgOID());
//Get Attribute
Attribute attribute1 =signerInfo.getSignedAttributes().get(PKCSObjectIdentifiers.pkcs_9_at_message
Attribute attribute2=null;
if(signerInfo.getUnsignedAttributes()!=null)    {
    attribute2 =signerInfo.getUnsignedAttributes().get(PKCSObjectIdentifiers.id_aa_signatureTimeSt
}
messageDigest=Base64.getEncoder().encodeToString(
                Hex.decode(attribute1.getAttributeValues()[0].toString().substring(1)));
```

MDAlgorithm.java hosted with ❤ by GitHub view raw

4.4 Calculate the Message Digest inside PDF

First, we calculate the byterange data on the PDF

```java
byte[] contentToSigned=getByteRangeData(pdfBytes, signature.getByteRange());

private  byte[] getByteRangeData(ByteArrayInputStream bis,int[] byteRange)    {
    int length1=byteRange[1]+byteRange[3];
    byte[] contentSigned=new byte[length1];
    bis.skip(byteRange[0]);
    bis.read(contentSigned, 0, byteRange[1]);
    bis.skip(byteRange[2]-byteRange[1]-byteRange[0]);
    bis.read(contentSigned, byteRange[1], byteRange[3]);
    bis.reset();
    return contentSigned;

}
```

CalculateMDPdf.java hosted with ❤ by GitHub view raw

Then, we calculate the Message Digest on the PDF

```
//Calculate MD in PDF

        String
mdPdf=Base64.getEncoder().encodeToString(digest.digest(contentToSigne
d));
```

4.5 Compare the message digest from CMS and from calculation in PDF

If it is the same, then the signature is valid. On the other hand, if it is not the same, then the signature is not valid.

```
if(mdPdf.equals(messageDigest)) {
            logApp.info("Message Digest Signature ID {} is valid,
data integrity is OK",signatureSID);
        }
        else    {
            logApp.info("Message Digest Signature ID {} is invalid,
data integrity is NOT OK",signatureSID);
        }
```

. . .

5. Remarks

Basically what we discuss in this blog is a very simple example of digital signature validation inside a PDF file. We hope that this simple example is enough to be a starting point in understanding how the validation works, and also how digital signature in PDF work. If you have any questions or suggestions, please do give comments below.

. . .

6. References

- www.adobe.com/go/pdfreference

- https://github.com/rsatrio/PDF-Signature-Check/

Digital Signatures | Pdf | Java | Pdfbox | Cryptography

Responses (1)

What are your thoughts?

Respond

김경재
Jun 14, 2022

Step 4.2 and 4.3 is little bit wierd. Can you fix it please?

1 reply Reply

Creating Locally-Running LLM Chatbot using Java and Spring...

This article will explain how to create a chatbot that interacts with a pre-trained LL...

Aug 3, 2024 55 1

Because I Didn't Know the Difference Between Exception an...

My articles are open to everyone; non-member readers can read the full article by...

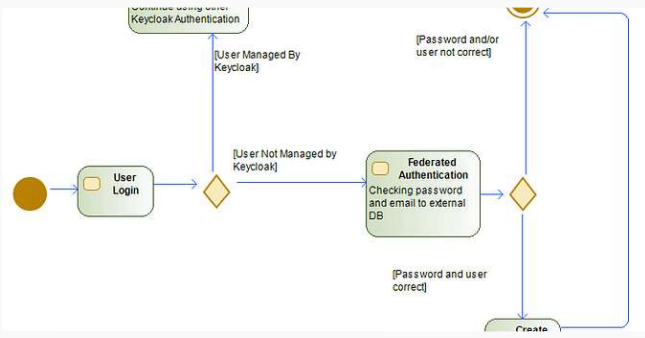
Dec 9, 2024 925 26

In Javarevisited by Rahul Soni

How to Prevent Duplicate Requests in REST APIs and Why Spring Say...

Handling duplicate requests in a REST API is essential, especially for actions that create,...

Nov 11, 2024 238 2



In Javarevisited by Rizky Satrio

Keycloak Integration with External Existing Database

I've used Keycloak many times as Identity and access management solution. One of the...

Dec 9, 2021 27 1

See all from Rizky Satrio

See all from Javarevisited

Recommended from Medium

In Byte of Knowledge by Mayur Koshti

Handling Large File Uploads in Laravel with Chunking and...

Best Practices for Efficient File Uploads in Laravel

Jan 6

Andrew Zuo

Just-In-Time Languages Are Taking Over

I've been using PocketBase a lot and, by extension, the Go programming language. G...

Jan 16 197 16

Lists

General Coding Knowledge

20 stories · 1879 saves

data science and AI

40 stories · 320 saves

Natural Language Processing

1889 stories · 1551 saves

Staff picks

802 stories · 1578 saves

Avinash Maheshwari

Image Processing Tool with Python

In this article, we'll walk through the creation of an intuitive image processing and real tim...

Jan 12 · 48 · 1

In Level Up Coding by Jacob Bennett

The 5 paid subscriptions I actually use in 2025 as a Staff Software...

Tools I use that are cheaper than Netflix

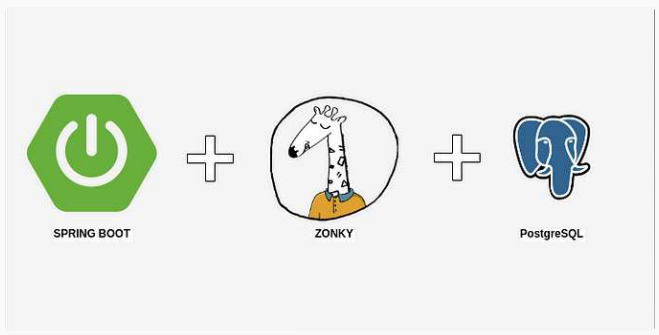
Jan 7 · 6K · 127

PURRFECT SOFTWARE LIMITED

How I Found My Developer Superpower by Stopping Tutorials

I Quit Learning Python and Discovered True Developer Growth

4d ago · 53



Eric Anicet

Spring Boot Embedded PostgreSQL Database for Testing

In this story, we'll learn how to implement Embedded PostgreSQL for Spring Boot...

Sep 16, 2024 · 20 · 1

See more recommendations