Contents

Introduction **2**
Introduction . . . 2

Version info **3**
Version info . . . 3

Definitions **3**
Definition: . . . 3

SiVa Service Overview **4**
SiVa overview . . . 4

Regulatory environment **6**
Regulatory environment . . . 6

Component diagram **6**
Component diagram . . . 6

Deployment view **7**
Deployment view . . . 7

Interfaces **8**
Interface description . . . 8

Database schema **9**
Database schema . . . 9

Use cases **9**
Use cases . . . 9

Deployment **11**
Deployment . . . 11

Logging **14**

Logging **14**
STDOUT appender . . . 15
FILE appender . . . 15
SYSLOG appender . . . 15

References **16**
References . . . 16
Appendix 1 - Non-Functional Requirements . . . 16
Appendix 2 - Functional Requirements . . . 16
Appendix 3 - Validation Policy . . . 16
Validation policy . . . 16

Appendix 4 - Validation Constraint Configuration 16
Appendix 5 - Test Case Descriptions 16

List of Test Cases **16**

Introduction

Introduction

SiVa is digital signature validation web service that provides SOAP and JSON API to validate following file types:

- Older Estonian digital signature files with DDOC extension
- BDOC containers with `TimeMark` and `TimeStamp` signatures
- Digitally signed PDF files
- X-Road security server ASiCE signature containers

Architecture document main purpose is to give overview what SiVa is. Give an overview of it's internal processes and provide information when deploying it to production environment.

SiVa architecture document sections overview

Below list will give You an overview of what each section of the SiVa architecture document will cover:

- **Overview** - gives overview what SiVa is and it's main features.
- **Regulatory environment** - legal analysis and standards that are used when building SiVa application
- **Component diagram** - gives overview of main SiVa subsystems and and and base validation Java libraries used for different validation services
- **Deployment view** - gives general overview of servers required when deploying SiVa validation web service into production
- **Interfaces** - Description of SiVa SOAP and JSON API request and response
- **Database schema** - description of SiVa validation administration service database
- **Use cases** - describes main processes in SiVa validation web service
- **Deploying** - how to build, deploy and configure SiVa web service
- **Logging** - how to configure and setup SiVa validation service logging support

Version info

Version info

Version number	Change date	Author	Description
0.1	06.05.2016	Mihkel Selgal	Initial SiVa architecture document

Definitions

Definition:

DSS Digital Signature Services is Java library to sign and validate European digital signature formats

BDOC Not defined

DDOC Not defined

PDF Portable document format is a file format that provides an electronic image of text or text and graphics that looks like a printed document and can be viewed, printed, and electronically transmitted.

SiVa is RESTful web service providing digital signature validation services for BDOC, DDOC, PDF and X-Road files

JVM The Java Virtual Machine is the runtime engine of the Java Platform, which allows any program written in Java or other language compiled into Java bytecode to run on any computer that has a native JVM.

JAR Java Archive is a package file format typically used to aggregate many Java class files and associated metadata and resources (text, images, etc.) into one file to distribute application software or libraries on the Java platform.

Fat JAR The fat JAR is the JAR, which contains classes from all the libraries, on which your project depends and, the classes of built project.

Spring Boot is a framework from the team at Pivotal, designed to simplify the bootstrapping and development of a new Spring application. The framework takes an opinionated approach to configuration, freeing developers from the need to define boilerplate configuration

Linux an operating system, based on UNIX, that runs on many different hardware platforms and whose source code is available to the public.

PüPKI Not defined

PostgreSQL is an open source relational database management system (DBMS) developed by a worldwide team of volunteers. PostgreSQL is not controlled by any corporation or other private entity and the source code is available free of charge.

SiVa Service Overview

SiVa overview

SiVa (Signature Validation) web service is continued development of PDF Validation web service. Service provides a JSON and SOAP based API web interface which purpose is to validate signatures in digitally signed BDOC, DDOC, PDF and X-Road ASiCE files according to validation policy (described in the Validation Policy section).

SiVa uses following Java libraries and command line utilities:

- EU DSS (Digital Signature Service) library is chosen for digitally signed PDF file validation
- DigiDoc4J Java library to validate BDOC containers. Supported signature types are `TimeStamp` and `TimeMark`
- JDigiDoc Java library is used to validate DDOC files starting from version
- X-Road ASiCE containers are validated using X-Road security server project provided command line utility

Validation libraries

DigiDoc4j EU DSS fork

DigiDoc4J EU DSS fork library for PDF files was chosen because it all the main validation constrains already provided and all new constraints can be added easily. For more information on EU DSS, see: https://joinup.ec.europa.eu/asset/sd-dss/description.

SiVa will use the following functionality of EU DSS library:

- PaDES Validation Functionality
- TSL loading functionality

DigiDoc4J

DigiDoc4J will be used to validate both `TimeMark` and `TimeStamp` based BDOC containers. DigiDoc4J was chosen because it's only Java library that can validate Estonian BDOC files according to SiVa validation policy. For more information on DigiDoc4J: https://github.com/open-eid/digidoc4j

SiVa will use the following functionality of DigiDoc4J:

- BDOC validation functionality

JDigiDoc

JDigiDoc provides support for DDOC files the library was chosen because it provides most complete support for all required DDOC versions. Read more about JDigiDoc: https://github.com/open-eid/jdigidoc

SiVa will use the following functionality of JDigiDoc:

- DDOC validation functionality

X-Road signature validation utility

X-Road signature validation utility is command line tool to validate X-Road Security server generated ASiCe files. The utility was chosen because it's only available packaged to tool to validate X-Road signature files.

Main features of SiVa validation service:

- SiVa SOAP ETSI compliant API to validate all supported signatures.
- SiVa REST ETSI compliant API to validate all supported signatures.
- SiVa handles files in PDF-format version 1.7 and later, signed with PadES-profile signatures.
- Service handles DDOC files starting from version 1.0 or later
- Service supports BDOC files starting from version 2.1 or later
- Service supports X-Road 6 security server ASiCE containers
- SiVa uses European Commission's TSL (Trusted Service Status List) for certificate chain validation for PDF and BDOC files.
  - European Commission's TSL contains references to TSLs of European Union's member states and members of the European Economic Area. This allows the PDF Validator to validate signature that has been signed with certificates issued in any of European Union's member states.
  - During the validation process, a certificate chain is created from signer's certificate up to the trust anchor (national trust list referenced by the central European Commission's trust list) for all certificates included in the signature (i.e. the signer's certificate, OCSP service's certificate, time-stamping Service's certificate).
- SiVas for DDOC and X-Road signature containers will use configured list certificates.
- Signatures with PadES-LT and PadES-LTA profile are supported.
- BDOC signatures with type BDOC-TM and BDOC-TS are supported

At the time of creating the current documentation, it is expected that SiVa will be used by the following applications:

- DigiDoc3 Client application
- Third party document management applications

Regulatory environment

Regulatory environment

!!! note Regulatory environment section will be added when analysis will be completed

Component diagram

Component diagram

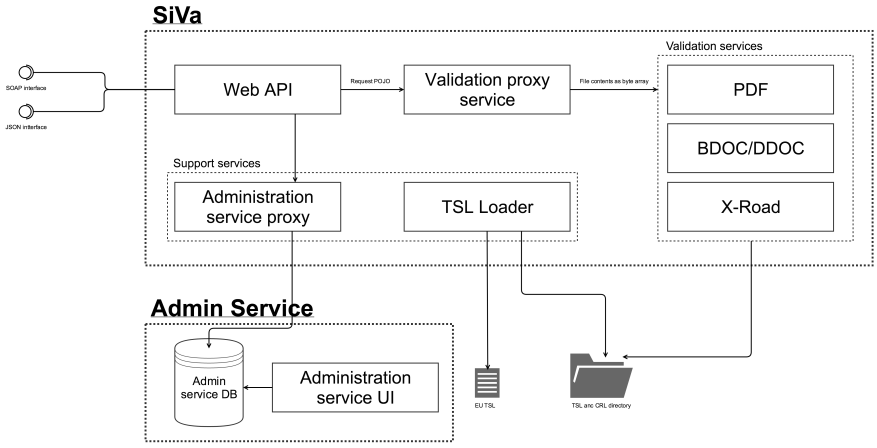

Figure 1: SiVa component diagram

- Web API is standard Spring MVC module it will take in JSON or SOAP requests sent by systems that integrate with SIVA web service API
- Validation service proxy or validation service selector is Spring module that will choose the appropriate validation service for user request
- TSL loader loads in contents of TSL file from given URL in online mode or from directory when using offline mode in predefined interval.
- PüPKI proxy converts PüPKI DB SQL results to Java objects that validation services, validation service proxy and Web API will use
- Validation services (listed below) validate different digitally signed documents
  - PDF validation service for PDF files will use **DigiDoc4J DSS forked** library
  - BDOC for ASiC compliant containers both TM and TS will latest Maven released **DigiDoc4J** library

  - DDOC for previous generation digitally signed files will use latest Maven release of **JDigiDoc**
  - X-Road Signature validation service for X-Road web service signature files will use X-Road Security server project provided X-Road signature validation command line utility.
- PüPKI DB relational database providing client information and client request counting
- PüPKI UI a separate Python application that will provide CRUD operations for clients

Deployment view

Deployment view

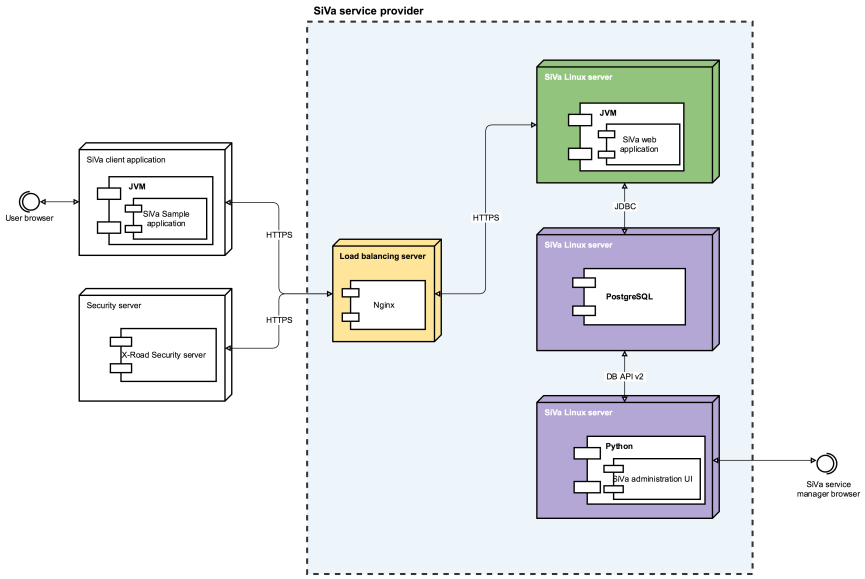

Figure 2: SiVa Deployment view

Load balancer

Load balancer can distribute traffic between SiVa nodes when there is more then one instance running. SiVa do not set any specific requirements for load balancer but in diagram the Nginx reverse proxy option is shown.

SiVa application server

SiVa validation service server that will provide the service needs to have JVM installed) both more commonly known options Oracle or OpenJDK are supported. SiVa application is built as executable JAR and can be configured like system service on Linux operating system.

Read more about running Spring Boot applications as Linux system service

SiVa validation service can run in cluster because it does not keep or create any sessions with client application or service.

!!! note The single executable JAR option may change in the future because we are considering isolating each validation service and SiVa web application into separate JVM instance

Database server

SiVa database server requirements are dictated by PüPKI application and from these requirements needs to be PostgreSQL. Currently there are no special requirements for database server nor database setup.

!!! note Database requirements section will be updated when analysis and development will begin on modification of PüPKI application

!!! development It may be possible that we will build administration user interface as part of SiVa project so the database and type of database RDBMS or NoSQL may change in the future

SiVa administration server

SiVa administration will use PüPKI administration user interface to manage authorized service users and collect information about basic service usage by authorized clients.

Only requirement currently known is that Python programming language support must be present in the server.

!!! note More detailed information about SiVa administration server setup will be provided when development and analysis will begin for administration service

!!! development There is alternative option that we will build administration user interface from scratch using Java and Spring Boot

Interfaces

Interface description

!!! warning The REST and SOAP API will be revised when final report is has been agreed upon

!!! development Maybe we should only display link to Swagger/Open API generated documentation to more easily keep SiVa API documentation in sync.

Database schema

Database schema

!!! note Will be added after SiVa administration system analysis has been completed.

Use cases

Use cases

Digitally signed document validation process

Digitally signed document validation process shows how SiVa chooses validation service and possible output of validation process.

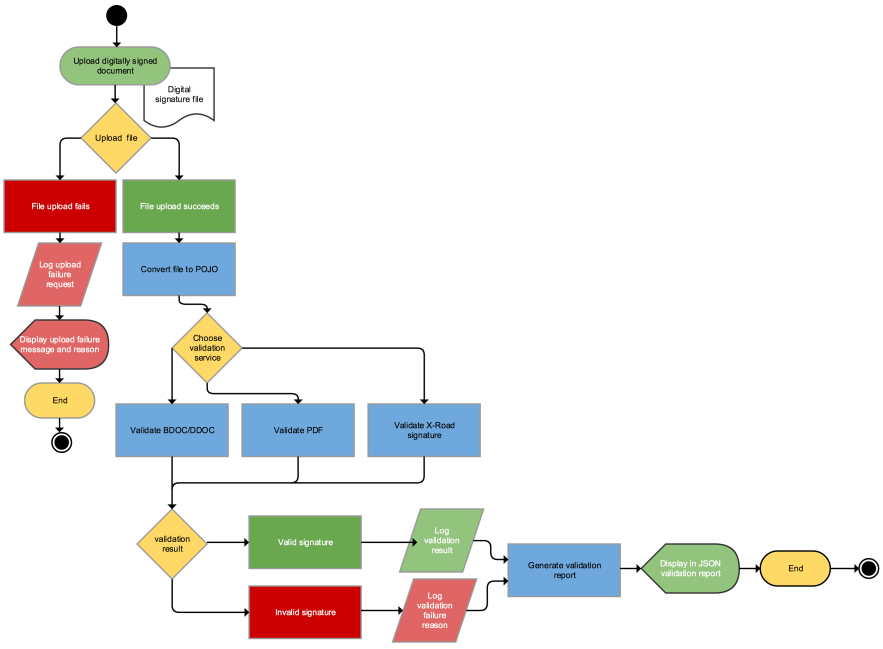

Figure 3: BDOC validation process

User of SiVa system provides digitally signed document file in form of Base64 encoded string. The validation of file and validation policy is handled by validation services underlying libraries.

- In case of PDF file it will be DSS
- For BDOC and DDOC files we will use DigiDoc4J or when required jDigiDoc
- And for X-Road signatures we will use X-road signature validation utility

We will log following failure cases: When file upload fails (request started but was not completed successfully) When request validation (JSON or SOAP) fails When user authentication fails - **not shown in diagram above** When signature validation fails – **not shown in diagram above** When increasing of request count fails – **not shown in diagram above**

Certificate loading process

All validation services require certificates to validate digitally signed documents. Below process shows how certificates are loaded into validation service. Loading process is done separably for each validation service.

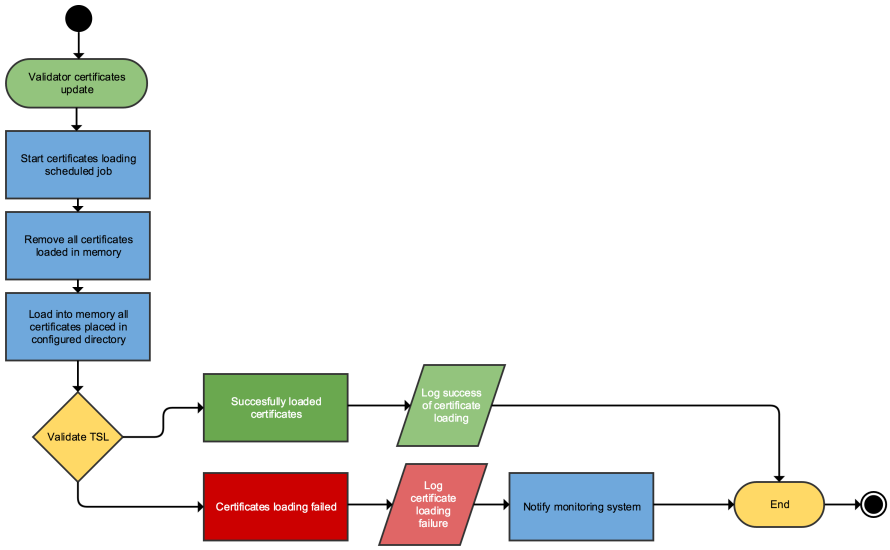

Figure 4: Certificate Loading process

Certificate loading process is scheduled cron job inside each validation service to update currently in memory loaded certificates.

This process should run after TSL loader has completed updating SiVa local copy of certificates.

X-Road 6 security server SOAP request process

X-Road validation process is brought out because we skip authentication process for X-Road security server interface and and use XML SOAP as input source.

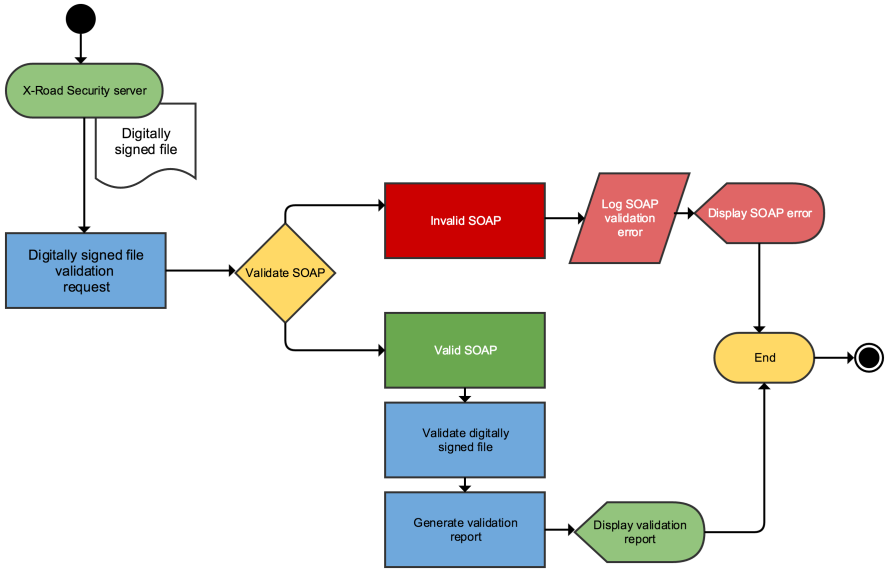

Figure 5: X-Road SOAP validation request

Validation of SOAP request XML is done in the SiVa web application module. Document validation process is described in detail in Digitally signed document validation process Validation report output id described in Interface description

Authenticate JSON REST API user

Validation of JSON request is done in SiVA web application module Document validation process is described in detail in Digitally signed document validation process Validation report output id described in Interface description

TSL loading use case

Deployment

Deployment

System requirements

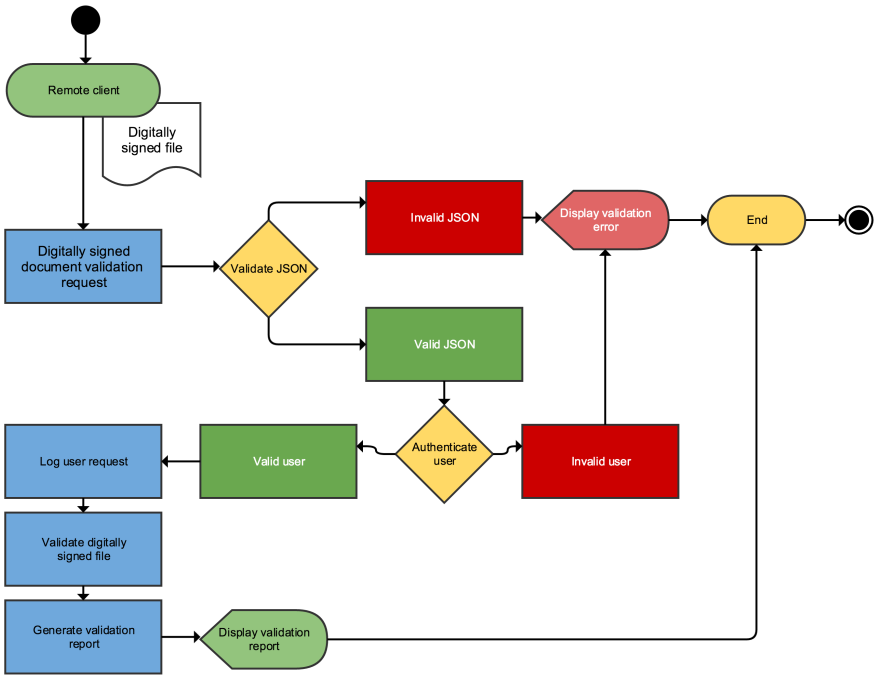

Figure 6: JSON REST validation request

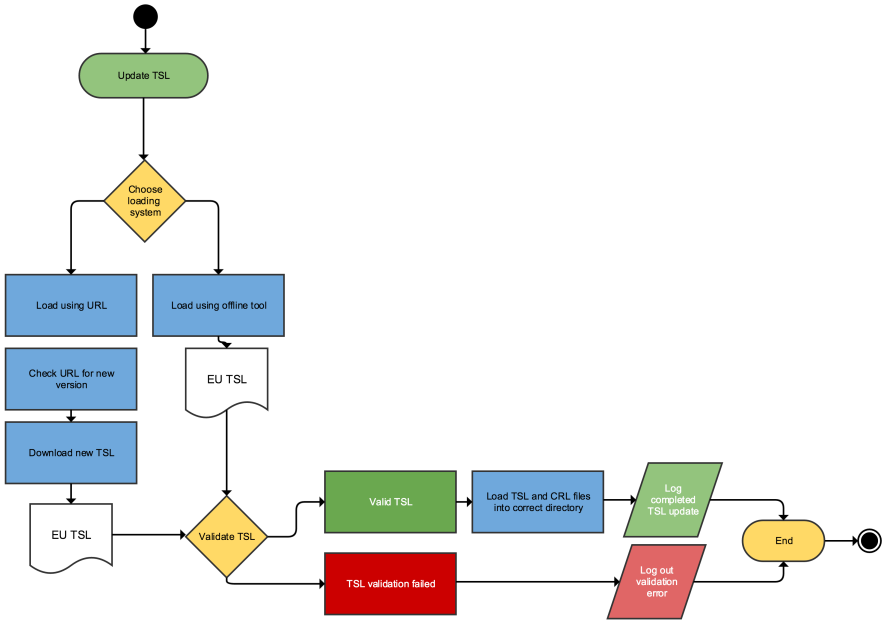

Figure 7: TSL loading process

Following are minimum requirements to build and deploy SiVa validation web service:

- Java 8 or above Oracle JVM is supported
- Git version control system version 1.8 or above is recommended
- 2GB of RAM
- 1 processor core
- Open internet connection
- 1GB of free disk space
- Supported operating system is Ubuntu 14.04 LTS

Building SiVa validation web service on Ubuntu 16.04

First we need to install Git and Java SDK 8 by issuing below commands:

```
sudo apt-get update
sudo apt-get install git -y
sudo apt-get install default-jdk -y
```

Next we need to clone the SiVa Github repository:

```
git clone https://github.com/open-eid/SiVa.git --branch develop
```

Final step is building the SiVa project using Maven Wrapper

```
cd SiVa
./mvnw install
```

Logging

Logging

Logging functionality is handled by the **SLF4J** logging facade and on top of it the **Logback** framework is used. As a result, logging can be configured via the standard Logback configuration file. By default, logging works on the `INFO` level and logs are directed to the system console as well as a log file.

The logback xml configuration file can be found at:

`pdfValidator/pdf-validator-webapp/src/main/resources/logback.xml`

and when compiled the file will reside at

`WEB-INF/classes/logback.xml`

within the packaged war. There is also a possibility to set the location of the default configuration file with a system property `logback.configurationFile` as a JVM argument. The value of this property can be a URL, a resource on the class path or a path to a file external to the application.

```
java -Dlogback.configurationFile=/path/to/config.xml
```

In this configuration file there are three appenders: `STDOUT` (logs to standard output), `FILE` (logs to a file) and `SYSLOG` (logs to syslog server over the network). To disable certain appender from logging, commenting out its `appender-ref` is sufficient, but it is *recommended* that the appender itself should also be commented out. For example to disable `SYSLOG` appender (which is the default configuration), then one can use following configuration:

```
<!--
<appender name="SYSLOG" class="ch.qos.logback.classic.net.SyslogAppender">
    <syslogHost>enter\_ip\_or\_hostname\_here</syslogHost>
    <port>514</port>
    <facility>USER</facility>
    <filter class="ch.qos.logback.classic.filter.ThresholdFilter">
        <level>INFO</level>
    </filter>
    <suffixPattern>%-5level %logger{0}:%L \[%thread\] - %msg</suffixPattern>
</appender>
-->

<root level="DEBUG">
```

```
<appender-ref ref="STDOUT"/>
<appender-ref ref="FILE"/>
<!--<appender-ref ref="SYSLOG"/>-->

</root>
```

Logback configuration manual: http://logback.qos.ch/manual/

STDOUT appender

- Default the log level is set to DEBUG
- Appender output pattern is: `%d{HH:mm:ss.SSS} %-5level %logger{0}:%L [%thread] - %msg%n`

FILE appender

- Default log level is set to `INFO`
- uses RollingFileAppender configured with `TimeBasedRollingPolicy`. Current configuration makes a seperate logfile for each day and each file is kept for *30 days*.

PS! keep in mind when using relative destination file path, then the path is added at the end of the currently working directory, i.e. where the application was started. (Current day's logfile path: `logs/pdf-validator-webapp.log`, prievious days pattern:

`logs/pdf-validator-webapp.%d{yyyy-MM-dd}.log)`

- Appender output pattern is: `%d{HH:mm:ss.SSS} %-5level %logger{0}:%L \[%thread\] - %msg%n`

  `-Dlogback.configurationFile=config.xml`

SYSLOG appender

- Default log level is set to `INFO`
- Target's ip/hostname and port are configurable
- Syslog messsages' severity is configurable
- Syslog messages' payload's timestamp and hostname part are created implicitly and the suffixpattern is: `%-5level %logger{0}:%L \[%thread\] - %msg`

References

References

Appendix 1 - Non-Functional Requirements

Appendix 2 - Functional Requirements

Appendix 3 - Validation Policy

Validation policy

!!! note Information will be added after analysis has been completed and agreed upon

Appendix 4 - Validation Constraint Configuration

Appendix 5 - Test Case Descriptions

List of Test Cases

BdocValidationFail.java

TestCaseID: Bdoc-ValidationFail-1

- TestType: Automated
- RequirementID:
- Title: Bdoc with single invalid signature
- Expected Result: The document should fail the validation
- File: IB-3960_bdoc2.1_TSA_SignatureValue_altered.bdoc

TestCaseID: Bdoc-ValidationFail-2

- TestType: Automated
- RequirementID:
- Title: Bdoc with multiple invalid signatures
- Expected Result: The document should fail the validation
- File:

TestCaseID: Bdoc-ValidationFail-3

- TestType: Automated
- RequirementID:
- Title: Bdoc with multiple signatures both valid and invalid
- Expected Result: The document should fail the validation
- File: BdocMultipleSignaturesMixedWithValidAndInvalid.bdoc

BdocValidationPass.java

TestCaseID: Bdoc-ValidationPass-1

- TestType: Automated
- RequirementID:
- Title: Bdoc with single valid signature
- Expected Result: The document should pass the validation
- File: Valid_ID_sig.bdoc

TestCaseID: Bdoc-ValidationPass-2

- TestType: Automated
- RequirementID:
- Title: Bdoc with multiple valid signatures
- Expected Result: The document should pass the validation
- File: Valid_IDCard_MobID_signatures.bdoc

DdocValidationFail.java

TestCaseID: Ddoc-ValidationFail-1

- TestType: Automated
- RequirementID:
- Title: Ddoc with single invalid signature
- Expected Result: The document should fail the validation
- File:

TestCaseID: Ddoc-ValidationFail-2

- TestType: Automated
- RequirementID:
- Title: Ddoc with multiple invalid signatures
- Expected Result: The document should fail the validation
- File:

TestCaseID: Ddoc-ValidationFail-3

- TestType: Automated
- RequirementID:
- Title: Ddoc with multiple signatures both valid and invalid
- Expected Result: The document should fail the validation
- File:

DdocValidationPass.java

TestCaseID: Ddoc-ValidationPass-1

- TestType: Automated
- RequirementID:

- Title: Ddoc with single valid signature
- Expected Result: The document should pass the validation
- File: 23734-ddoc13-13basn1.ddoc

TestCaseID: Ddoc-ValidationPass-2

- TestType: Automated
- RequirementID:
- Title: Ddoc with multiple valid signatures
- Expected Result: The document should pass the validation
- File: igasugust1.1.ddoc

DocumentFormatTests.java

TestCaseID: DocumentFormat-1

- TestType: Automated
- RequirementID: (TBD)
- Title: Validation of pdf document acceptance
- Expected Result: Pdf is accepted and correct signature validation is given
- File: hellopades-pades-lt-sha256-sign.pdf

TestCaseID: DocumentFormat-2

- TestType: Automated
- RequirementID: (TBD)
- Title: Validation of bdoc document acceptance
- Expected Result: Bdoc is accepted and correct signature validation is given
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: DocumentFormat-3

- TestType: Automated
- RequirementID: (TBD)
- Title: Validation of txt document rejection
- Expected Result: Txt document is rejected and proper error message is given
- File: hellopades-pades-lt-sha256-sign.txt

TestCaseID: DocumentFormat-4

- TestType: Automated
- RequirementID: (TBD)
- Title: Validation of p7s document rejection
- Expected Result: P7s document is rejected and proper error message is given
- File: hellocades.p7s

TestCaseID: DocumentFormat-5

- TestType: Automated

- RequirementID: (TBD)
- Title: Validation of zip document rejection
- Expected Result: Zip document is rejected and proper error message is given
- File: 42.zip

TestCaseID: DocumentFormat-6

- TestType: Automated
- RequirementID: (TBD)
- Title: Validation of doc document rejection
- Expected Result: Doc document is rejected and proper error message is given
- File: hellopades-pades-lt-sha256-sign.doc

TestCaseID: DocumentFormat-7

- TestType: Automated
- RequirementID: (TBD)
- Title: Validation of XML document rejection
- Expected Result: XML document is rejected and proper error message is given
- File: XML.xml

TestCaseID: DocumentFormat-8

- TestType: Automated
- RequirementID: (TBD)
- Title: Validation of png document rejection
- Expected Result: Png document is rejected and proper error message is given
- File: Picture.png

LargeFileTests.java

TestCaseID: PDF-LargeFiles-1

- TestType: Automated
- RequirementID:
- Title: Larger signed PDF files (PAdES Baseline LT).
- Expected Result: Bigger documents with valid signature should pass
- File: scout_x4-manual-signed_lt_9mb.pdf

TestCaseID: PDF-LargeFiles-2

- TestType: Automated
- RequirementID:
- Title: Larger signed PDF files (PAdES Baseline LT).
- Expected Result: Bigger documents with valid signature should pass
- File: scout_x4-manual-signed_lta_9mb.pdf

TestCaseID: PDF-LargeFiles-3

- TestType: Automated
- RequirementID:
- Title: Larger signed PDF files (PAdES Baseline LT).
- Expected Result: Bigger documents with valid signature should pass
- File: digidocservice-signed-lt-1-2mb.pdf

TestCaseID: PDF-LargeFiles-4

- TestType: Automated
- RequirementID:
- Title: Larger signed PDF files (PAdES Baseline LT).
- Expected Result: Bigger documents with valid signature should pass
- File: digidocservice-signed-lta-1-2mb.pdf

TestCaseID: PDF-LargeFiles-5

- TestType: Automated
- RequirementID:
- Title: Larger signed PDF files (PAdES Baseline LT).
- Expected Result: Bigger documents with valid signature should pass
- File: egovernment-benchmark-lt-3-8mb.pdf

TestCaseID: PDF-LargeFiles-6

- TestType: Automated
- RequirementID:
- Title: Larger signed PDF files (PAdES Baseline LT).
- Expected Result: Bigger documents with valid signature should pass
- File: egovenrment-benchmark-lta-3-8mb.pdf

PdfBaselineProfileTests.java

TestCaseID: PDF-BaselineProfile-1

- TestType: Automated
- RequirementID:
- Title: The PDF has PAdES-B profile signature
- Expected Result: Document validation should fail
- File: hellopades-pades-b-sha256-auth.pdf

TestCaseID: PDF-BaselineProfile-2

- TestType: Automated
- RequirementID:
- Title: The PDF has PAdES-T profile signature
- Expected Result: Document validation should fail
- File: some_file.pdf

TestCaseID: PDF-BaselineProfile-3

- TestType: Automated
- RequirementID:
- Title: The PDF has PAdES-LT profile signature
- Expected Result: Document validation should pass
- File: hellopades-pades-lt-sha256-sign.pdf

TestCaseID: PDF-BaselineProfile-4

- TestType: Automated
- RequirementID:
- Title: The PDF has PAdES-LTA profile signature
- Expected Result: Document validation should pass
- File: some_file.pdf

TestCaseID: PDF-BaselineProfile-5

- TestType: Automated
- RequirementID:
- Title: The PDF has PAdES-LTA profile signature
- Expected Result: Document validation should pass
- File: some_file.pdf

TestCaseID: PDF-BaselineProfile-6

- TestType: Automated
- RequirementID:
- Title: PDF document message digest attribute value does not match calculate value
- Expected Result: Document validation should fail
- File: hellopades-lt1-lt2-wrongDigestValue.pdf

TestCaseID: PDF-BaselineProfile-7

- TestType: Automated
- RequirementID: DSS-SIG-MULTISIG
- Title: PDF file with a serial signature
- Expected Result: Document signed with multiple signers with serial signatures should pass
- File: hellopades-lt1-lt2-Serial.pdf

TestCaseID: PDF-BaselineProfile-8

- TestType: Automated
- RequirementID:
- Title: PDF document signed with multiple signers parallel signature
- Expected Result: Document with parallel signatures should pass
- File: hellopades-lt1-lt2-parallel3.pdf

TestCaseID: PDF-BaselineProfile-9

- TestType: Automated
- RequirementID:

- Title: PDF document signed with multiple signers parallel signature without Sscd
- Expected Result: Document with no qualified and without SSCD should fail
- File: hellopades-lt1-lt2-parallel3.pdf

PdfSignatureCryptographicAlgorithmTests.java

TestCaseID: PDF-SigCryptoAlg-1

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: SHA512 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with SHA512 algorithm should pass
- File: hellopades-lt-sha512.pdf

TestCaseID: PDF-SigCryptoAlg-2

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: SHA1 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with SHA1 algorithm should pass
- File: hellopades-lt-sha1.pdf

TestCaseID: PDF-SigCryptoAlg-3

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: ECDSA algorithms (PAdES Baseline LT)
- Expected Result: Document signed with ECDSA algorithm should pass
- File: hellopades-ecdsa.pdf

TestCaseID: PDF-SigCryptoAlg-4

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: ECDSA224 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with ECDSA224 algorithm should pass
- File: hellopades-lt-sha256-ec224.pdf

TestCaseID: PDF-SigCryptoAlg-5

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: ECDSA256 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with ECDSA256 algorithm should pass
- File: hellopades-lt-sha256-ec256.pdf

TestCaseID: PDF-SigCryptoAlg-6

- TestType: Automated

- RequirementID: DSS-CRY-ALGOEXP
- Title: RSA1024 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with RSA1024 algorithm should pass
- File: hellopades-lt-sha256-rsa1024.pdf

TestCaseID: PDF-SigCryptoAlg-7

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: RSA1023 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with RSA1023 algorithm should pass
- File: hellopades-lt-sha256-rsa1023.pdf

TestCaseID: PDF-SigCryptoAlg-8

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: RSA2047 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with RSA2047 algorithm should pass
- File: hellopades-lt-sha256-rsa2047.pdf

TestCaseID: PDF-SigCryptoAlg-9

- TestType: Automated
- RequirementID: DSS-CRY-ALGOEXP
- Title: RSA2048 algorithms (PAdES Baseline LT)
- Expected Result: Document signed with RSA2048 algorithm should pass
- File: hellopades-lt-sha256-rsa2048.pdf

PdfValidationFail.java

TestCaseID: PDF-ValidationFail-1

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with certificate that is expired before signing (PAdES Baseline LT)
- Expected Result: Document signed with certificate that expired before signing should fail.
- File: hellopades-lt-rsa1024-sha1-expired.pdf

TestCaseID: PDF-ValidationFail-2

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with expired certificate (PAdES Baseline LT)
- Expected Result: Document signed with certificate that is expired should fail.
- File: IB-3691_bdoc21-TS-old-cert.bdoc

TestCaseID: PDF-ValidationFail-3

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with revoked certificate (PAdES Baseline LT)
- Expected Result: Document signed with certificate that is revoked should fail.
- File: hellopades-lt-sha256-revoked.pdf

TestCaseID: PDF-ValidationFail-4

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with certificate that missing signed attribute (PAdES Baseline LT)
- Expected Result: PDF-file validation should fail
- File: missing_signing_certificate_attribute.pdf

TestCaseID: PDF-ValidationFail-5

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with certificate which has no non repudiation key usage attribute (PAdES Baseline LT)
- Expected Result: The PDF-file validation should fail with error.
- File: hellopades-pades-lt-sha256-auth.pdf

TestCaseID: PDF-ValidationFail-6

- TestType: Automated
- RequirementID:
- Title: hellopadess been signed with an expired certificate, where signing time is within the original validity
- Expected Result: Document signed with expired certificate should fail
- File: hellopades-lt-sha256-rsa2048-expired.pdf

TestCaseID: PDF-ValidationFail-7

- TestType: Automated
- RequirementID:
- Title: hellopadess been signed with an expired certificate, where signing time is within the original validity
- Expected Result: Document signed with expired certificate should fail
- File: hellopades-lt-sha256-rsa1024-expired2.pdf

TestCaseID: PDF-ValidationFail-8

- TestType: Automated
- RequirementID:

- Title: hellopadess been signed with an expired certificate, where signing time is within the original validity
- Expected Result: Document signed with expired certificate should fail
- File: hellopades-lt-sha1-rsa1024-expired2.pdf

PdfValidationPass.java

TestCaseID: PDF-ValidationPass-1

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with certificate that is expired after signing (PAdES Baseline LT)
- Expected Result: Document signed with certificate that expired after signing should pass.
- File: hellopades-lt-sha256-rsa1024-not-expired.pdf

TestCaseID: PDF-ValidationPass-2

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with certificate that will expire in 7 days after signing (PAdES Baseline LT)
- Expected Result: Document signed with certificate that expired after signing should pass.
- File: hellopades-lt-sha256-rsa2048-7d.pdf

TestCaseID: PDF-ValidationPass-3

- TestType: Automated
- RequirementID:
- Title: Certificate contents are include in response (PAdES Baseline LT)
- Expected Result: The PDF-file validation should pass
- File: hellopades-lt-sha256-ocsp-15min1s.pdf

SignatureRevocationValueTests.java

TestCaseID: PDF-SigRevocVal-1

- TestType: Automated
- RequirementID: DSS-SIG-OCSP-TS-WARN
- Title: The PDF-file has PAdES-LT profile signature and an OCSP confirmation that is more than 15 minutes later than the signatures Time Stamp.
- Expected Result: Document with ocsp over 15 min delay should pass but warn
- File: hellopades-lt-sha256-ocsp-15min1s.pdf

TestCaseID: PDF-SigRevocVal-2

- TestType: Automated
- RequirementID: DSS-SIG-OCSP-TS-WARN
- Title: The PDF-file has PAdES-LT profile signature and an OCSP confirmation that is more than 15 minutes later than the signatures Time Stamp.
- Expected Result: Document with ocsp over 15 min delay should pass but warn
- File: hellopades-lt-sha256-ocsp-15min1s.pdf

TestCaseID: PDF-SigRevocVal-3

- TestType: Automated
- RequirementID: DSS-SIG-OCSP-TS-ERROR
- Title: The PDF-file has PAdES-LT profile signature and an OCSP confirmation more than 24 hours later than the signatures Time Stamp.
- Expected Result: Document with over 24h delay should fail
- File: hellopades-lt-sha256-ocsp-28h.pdf

TestCaseID: PDF-SigRevocVal-4

- TestType: Automated
- RequirementID: DSS-SIG-OCSP-TS-ERROR
- Title: The PDF-file has PAdES-LT profile signature and an OCSP confirmation more than 24 hours later than the signatures Time Stamp.
- Expected Result: Document with over 24h delay should fail
- File: hellopades-lt-sha256-ocsp-28h.pdf

TestCaseID: PDF-SigRevocVal-5

- TestType: Automated
- RequirementID:
- Title: The PDF-file has been signed with PAdES Baseline LTA profile signature, the signature contains CRL.
- Expected Result: Document with no ocsp or crl in signature should fail
- File: hellopades-lta-no-ocsp.pdf

TestCaseID: PDF-SigRevocVal-6

- TestType: Automated
- RequirementID:
- Title: PDF signature has OCSP confirmation before Time Stamp
- Expected Result: Document signed with ocsp time value before best signature time should fail
- File: hellopades-lt-sha256-rsa2048-ocsp-before-ts.pdf

ValidationReportJsonStructureVerification.java

TestCaseID: Bdoc-ValidationReport-1

- TestType: Automated

- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Bdoc valid single signature)
- Expected Result: All required elements are present according to BdocDoc-SimpleReportSchema.json
- File: Valid_ID_sig.bdoc

TestCaseID: Bdoc-ValidationReport-2

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Bdoc valid multiple signatures)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: Bdoc-ValidationReport-3

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Bdoc invalid single signature)
- Expected Result: All required elements are present according to BdocDoc-SimpleReportSchema.json
- File: IB-3960_bdoc2.1_TSA_SignatureValue_altered.bdoc

TestCaseID: Bdoc-ValidationReport-4

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Bdoc indeterminate status)
- Expected Result: All required elements are present according to BdocDoc-SimpleReportSchema.json
- File: test1-bdoc-unknown.bdoc

TestCaseID: Bdoc-ValidationReport-5

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: Check for optional subindication and error elements
- Expected Result: Error and subindication elements are present
- File: IB-3960_bdoc2.1_TSA_SignatureValue_altered.bdoc

TestCaseID: Bdoc-ValidationReport-6

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: Check for optional warning element
- Expected Result: Warning element is present
- File: 23154_test1-old-sig-sigat-NOK-prodat-OK-1.bdoc

TestCaseID: Pdf-ValidationReport-7

- TestType: Automated

- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Pdf valid single signature)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: hellopades-lt-sha256-ec256.pdf

TestCaseID: Pdf-ValidationReport-8

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Pdf valid Multiple signatures)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File:

TestCaseID: Pdf-ValidationReport-9

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Pdf invalid signature)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: hellopades-lt-b.pdf

TestCaseID: Pdf-ValidationReport-10

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (Pdf indeterminate status)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: hellopades-lt-rsa1024-sha1-expired.pdf

TestCaseID: Ddoc-ValidationReport-11

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (ddoc valid single signature)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: 18912.ddoc

TestCaseID: Ddoc-ValidationReport-12

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (ddoc valid Multiple signatures)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: igasugust1.1.ddoc

TestCaseID: Ddoc-ValidationReport-13

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (ddoc invalid signature)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: test1-ddoc-revoked.ddoc

TestCaseID: Ddoc-ValidationReport-14

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: JSON structure has all elements (ddoc indeterminate status)
- Expected Result: All required elements are present according to SimpleReportSchema.json
- File: test1-ddoc-unknown.ddoc

TestCaseID: Ddoc-ValidationReport-15

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: Check for optional subindication and error elements
- Expected Result: Error and subindication elements are present
- File: test1-ddoc-unknown.ddoc

TestCaseID: Ddoc-ValidationReport-16

- TestType: Automated
- RequirementID: Validation report - WIP (TBD)
- Title: Check for optional warning element
- Expected Result: Warning element is present
- File: test1-ddoc-unknown.ddoc

ValidationRequestTests.java

TestCaseID: ValidationRequest-1

- TestType: Automated
- RequirementID:(TBD)
- Title: Input random base64 string as document
- Expected Result: Error is returned stating problem in document
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-2

- TestType: Automated
- RequirementID:(TBD)
- Title: Input a request with empty values
- Expected Result: Errors are returned stating the missing values

- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-3

- TestType: Automated
- RequirementID:(TBD)
- Title: Request with not base64 string as document
- Expected Result: Error is returned stating encoding problem
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-4

- TestType: Automated
- RequirementID:(TBD)
- Title: Verification of wrong report type as input
- Expected Result: Error is returned stating wrong report type
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-5

- TestType: Automated
- RequirementID:(TBD)
- Title: Verification of valid (but not simple) report type as input
- Expected Result: Correct report is returned (currently simple report is returned in all cases)
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-6

- TestType: Automated
- RequirementID:(TBD)
- Title: Verification of wrong document type as input
- Expected Result: Correct error code is returned
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-7

- TestType: Automated
- RequirementID:(TBD)
- Title: Mismatch in documentType and actual document (bdoc and pdf)
- Expected Result: Error is returned
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-8

- TestType: Automated
- RequirementID:(TBD)
- Title: Mismatch in documentType and actual document (asice and bdoc)
- Expected Result: Error is returned
- File: TS-11_23634_TS_2_timestamps.asice

TestCaseID: ValidationRequest-9

- TestType: Automated
- RequirementID:(TBD)
- Title: Verification of case insensitivity in document type
- Expected Result: Report is returned
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-10

- TestType: Automated
- RequirementID:(TBD)
- Title: Verification of filename value (filename do not match the actual file)
- Expected Result: The same filename is returned as sent in the request
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-11

- TestType: Automated
- RequirementID:(TBD)
- Title: Request has invalid character in filename
- Expected Result: Correct error code is returned
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-12

- TestType: Automated
- RequirementID:(TBD)
- Title: Request has invalid key on document position
- Expected Result: Error is returned
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-13

- TestType: Automated
- RequirementID:(TBD)
- Title: Request has wrong key on report type position
- Expected Result: Error is returned
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-14

- TestType: Automated
- RequirementID:(TBD)
- Title: Request has XML as document type (special case, XML is similar to ddoc)
- Expected Result: Error is given
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-15

- TestType: Automated
- RequirementID:(TBD)
- Title: Request has long (38784 characters) in filename field

- Expected Result: Report is returned with the same filename
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-16

- TestType: Automated
- RequirementID:(TBD)
- Title: Totally empty request body is sent
- Expected Result: Error is given
- File: None

TestCaseID: ValidationRequest-17

- TestType: Automated
- RequirementID:(TBD)
- Title: Request with more parameters than expected is sent
- Expected Result: Error is given or extra parameters are ignored?
- File: Valid_IDCard_MobID_signatures.bdoc

TestCaseID: ValidationRequest-18

- TestType: Automated
- RequirementID:(TBD)
- Title: Request with special chars is sent
- Expected Result: Validation report is returned
- File: Valid_IDCard_MobID_signatures.bdoc