Federal Office
for Information Security

Certification Path Validation Test Tool — User Documentation

Document history

Version	Date	Editor	Description
0.1	15.09.2017	EK	Document created.
0.2	15.11.2017	EK	Changes after feedback from BSI.
0.3	05.12.2017	EK	Changes after feedback from BSI.
1.0	February 2018	BSI	Final version.
1.1	23.10.2018	EK	Update for OCSP.

Authors

Dr. Evangelos Karatsiolis
MTG AG
Dolivostraße 11
64293 Darmstadt

Federal Office for Information Security
Post Box 20 03 63
D-53133 Bonn
Phone: +49 228 9582-0
E-Mail: cpt@bsi.bund.de
Internet: https://www.bsi.bund.de

Table of Contents

Figures

Tables

1 Introduction

This document describes how to configure and use the Certification Path Validation Test Tool (CPT) implemented in the BSI-Project 242 'Erstellung einer Technischen Richtlinie und Entwicklung eines Testtools zur Zertifikatsverifikation'.

CPT is a program written in Java which creates certificates, certificate revocation lists (CRL), and OCSP responses for test cases regarding certification path validation routines. These certificates and CRLs are described in XML files and are created by the tool according to the description. After their creation they are written to the filesystem to be further used in testing procedures. The tool aids this procedure by exporting these data in different formats and variants, running HTTP or LDAP servers for locating revocation lists, and providing pre-built test reports.

The document is built up as follows. In Section 2 a guide about the installation of the tool is given. In Section 3 the configuration files of the tool and the configuration options are described. In Section 4 we describe how to run the tool and what is the expected output. Finally, in Section 5 advice about troubleshooting is given.

2 Installation

In this section we describe the requirements for running CPT (Section 2.1) and how to install it (Section 2.2).

2.1 Requirements

For running the CPT the following requirements must be met on the system:

- Installed Java 8.
- Installed Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files[1] or, depending on the Java version, properly configured Java properties[2].

The CPT requires Java 8 to run.

2.2 Installation

The delivered zip file must be unzipped. This creates the directory *certification_path_tool*. Under this directory the following files can be found:

- **cpt.ini**
  This is the configuration file of the CPT. For details see Section 3.1.
- **archives**
  Under this directory the CPT archives certificates, CRLs, and configuration data.
- **lib**
  Under this directory the binaries of the CPT and the used libraries are located.
- **resources**
  Under this directory resources used by CPT, for example pictures, are located.
- **pkiObjects**
  Under this directory the XML files that contain definitions of PKIObjects (see [AP6]) are located. In order for the tool to locate the proper file, the name of the XML file and the value of the TestDataReference tag of the test case must match.
- **setup**
  Under this directory the XML files that contain definitions of PKIObjects are located. These files specify certificates and revocation lists that are used for supporting the test procedure but are not part of certification paths to be tested. For example for testing the certification path validation of an SSL server, certificates for this server need to be created. While the server is tested using the validation of TLS client certificates created by CPT, certificates of the setup directory are used to properly configure the server. If the server does not have any certificates, testing is not possible.
- **run.bat**
  Is a shortcut for running the tool under Windows with default parameters.
- **runAsServer.bat**
  Is a shortcut for running the tool in a server mode under Windows with default parameters.
- **run.sh**
  Is a shortcut for running the tool under Unix-like systems with default parameters.

1 http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html
2 crypto.policy=unlimited in the java.security configuration file.

- **runAsServer.sh**
  Is a shortcut for running the tool in a server mode under Unix-like systems with default parameters.
- **testcases**
  Under this directory the XML files that specify test cases according to [TR-03124-2] are located. Files that do not end with *.xml* are ignored.
- **testReport.txt**
  This is a template file. The content of this file is used for producing a test report file for each test case. For more details see also Section 3.2.

3 Configuration files

In this section we describe the configuration files of the test tool. The main configuration file of the test tool *cpt.ini* is described in Section 3.1. The template for the test reports is described in Section 3.2.

3.1 cpt.ini

This is the main configuration file of CPT. In this file the parameters concerning the tool itself can be configured, for example the location of the logfile. Additionally, parameters regarding the tests can also be configured. Listing 1 shows an example of this file.

```
#####################
#### Log Section ####
#####################

## The name of the file where the log is written.
log.filename=cpt.log

## The level of logging. Typical Value: DEBUG, INFO, ERROR.
log.level=DEBUG

## The pattern of the log messages.
## See also: https://logback.qos.ch/manual/layouts.html
log.pattern=%date %-8([%level]) %logger{36}_%line [%X{TESTCASE}]
[%X{CERTIFICATE}] [%X{CRL}] %msg%n

## Shows whether the log will also be shown in the console. Allowed Values:
true or false.
log.console=true

########################
#### Report Section ####
########################

## Filename of the report template file
report.template.filename=testReport.txt

## The name of the test object
report.testobject.name=

## The version of the test object
report.testobject.version=

############################
#### Test Cases Section ####
############################

## semicolon-separated values of test case ids that must be ignored
skipTestCases=CERT_PATH_COMMON_14

## semicolon-separated values of names of profile that are executed
profiles=COMMON;COMMON_AUGMENTED;TLS CLIENT

############################
#### HTTP Server Section ####
############################
```

```
## true if CRLs are available over an HTTP server, false otherwise.
http.use=true

## Host name or IP address where the HTTP server is listening to for incoming
connections.
http.host=certpath_test_host

## The port where the HTTP server is listening to for incoming connections.
http.port=8095

## The name of the directory where the HTTP server stores its data.
http.resources.directory=httpData

############################
#### LDAP Server Section ####
############################

## true if CRLs are available over an LDAP server, false otherwise.
ldap.use=false

## Host name or IP address where the LDAP server is listening to for incoming
connections.
ldap.host=certpath_test_host

## The port where the LDAP server is listening to for incoming connections.
ldap.port=389

## The rootDN of the LDAP server.
ldap.root=dc=certpath_test_host

## The password of the LDAP user. The DN of the LDAP user is
'uid=admin,ou=system'
ldap.password=changeme

## The name of the directory where the LDAP server stores its data.
ldap.resources.directory=ldapData

####################
#### GUI Section ####
####################

## true if a window must be displayed for stopping the application.
showGUI=false

######################
#### Email Section ####
######################

## true if signed e-mails are sent to an SMTP server, false otherwise.
email.smtp.use=false

## Host name or IP address of the SMTP server to which e-mails are sent.
email.smtp.host=certpath_test_host

## Port of the SMTP server to which e-mails are sent.
email.smtp.port=25

## The sender's address included in the signed e-mail.
email.sender=test@mtg.de
```

```
## The recipient's address included in the signed e-mail.
email.recipient=test@mtg.de

## The signature algorithm used to sign e-mails.
email.signature.algorithm=SHA256withRSA

##############################
#### Replacements Section ####
####                     ####
#### These values are    ####
#### replaced in the XML ####
#### PKI Objects         ####
#### definitions         ####
##############################

## Use ${httpHost} in the PKI Objects
replace.httpHost=certpath_test_host

## Use ${httpPort} in the PKI Objects
replace.httpPort=8095

## Use ${ldapHost} in the PKI Objects
replace.ldapHost=certpath_test_host

## Use ${ldapPort} in the PKI Objects
replace.ldapPort=389

## Use ${ldapRoot} in the PKI Objects
replace.ldapRoot=dc=certpath_test_host

## Use ${serialNumber} in the PKI Objects
replace.serialNumber=1234

## Use ${issuerDN} in the PKI Objects
replace.issuerDN=CN=Test Issuer, C=DE

## Use ${publicKey} in the PKI Objects
replace.publicKey=RSA,2048

## Use ${signature} in the PKI Objects
replace.signature=1.2.840.10045.4.3.2

## Use ${extension.oid} in the PKI Objects
replace.extension.oid=2.5.29.15

## Use ${extension.name} in the PKI Objects
replace.extension.name=Key Usage

## Use ${extension.value} in the PKI Objects
replace.extension.value=BDIwMDAuoCygKqQoMCYxFzAVBgNVBAMMDkRpZmZlcmVudCBUZXN0MQs
wCQYDVQQGEwJERQ==

## Use ${extension.san.value} in the PKI Objects
replace.extension.san.value=dNSName=certpath_test_host

## Use ${extension.san.mismatch} in the PKI Objects
replace.extension.san.value.mismatch=dNSName=unknown_host

## Use ${extension.nameconstraints.san.value} in the PKI Objects
```

```
replace.extension.nameconstraints.san.value=dNSName=certpath_test_host

## Use ${extension.ku.value} in the PKI Objects
replace.extension.ku.value=digitalSignature

## Use ${ocsp.idHash} in the PKI Objects
replace.ocsp.idHash=1.3.14.3.2.26

## Use ${ocsp.signature} in the PKI Objects
replace.ocsp.signature=1.2.840.113549.1.1.11
```

Listing 1: An example of the cpt.ini file.

A list of all supported parameters and their function is found in Table 1.

Parameter	Function
log.filename	Specifies the name of the log file of the tool.
log.level	Specifies the level of the logging. Possible values are DEBUG, INFO, ERROR.
log.pattern	The pattern according to which the log messages are formed.
log.console	If true the application logs, additionally to the log file, also in the console.
report.template.filename	Specifies the name of the file where the template for the reports is located.
report.testobject.name	Specifies the name of the application or library under test. The configured value of this parameter is inserted in the corresponding placeholder in the report files.
report.testobject.version	Specifies the version of the application or library under test. The configured value of this parameter is inserted in the corresponding placeholder in the report files.
skipTestCases	Specifies the test cases for which PKI objects are not created. It contains the semicolon-separated values of test case ids that must be ignored.
profiles	Specifies the profiles of test cases for which PKI objects are created. It contains the semicolon-separated values of profiles. If a profile is not listed in this parameter certificates and CRLs are not created for test cases of this profile. The possible values are: {COMMON, COMMON_AUGMENTED, EMAIL, EMAIL_AUGMENTED, IPSEC, IPSEC_AUGMENTED, TLS CLIENT, TLS SERVER}
http.use	If set to “true” then a HTTP server is started if at least one certificate specifies an HTTP location in the CRL distribution points extension. The HTTP server is also used as an OCSP responder.
http.host	The hostname where the HTTP server is listening to.

http.port	The port where the HTTP server is listening to.
http.resources.directory	The name of the directory on the filesystem where the HTTP server stores its data.
ldap.use	If set to “true” then an LDAP server is started if at least one certificate specifies an LDAP location in the CRL distribution points extension.
ldap.host	The hostname where the LDAP server is listening to.
ldap.port	The port where the LDAP server is listening to.
ldap.root	The distinguished name of the root of the LDAP server.
ldap.password	The password of the LDAP server. The default password of the LDAP server is replaced by this password.
ldap.resources.directory	The name of the directory on the filesystem where the LDAP server stores its data.
showGUI	If set to “true”, then a GUI Window is shown that allows the user to stop the application. This is useful when the HTTP and/or LDAP servers are running.
email.smtp.use	If set to “true”, then signed emails are sent to an SMTP server.
email.smtp.host	Specifies the hostname of an SMTP server where the signed emails produced by CPT are sent.
email.smtp.port	Specifies the port of an SMTP server where the signed emails produced by CPT are sent.
email.sender	Specifies the sender of the signed emails produced by CPT.
email.recipient	Specifies the recipient of the signed emails produced by CPT.
email.signature.algorithm	Specifies the algorithm with which emails are signed. All algorithms supported by the cryptographic library Bouncy Castle[3] can be used here as long as the corresponding public key of the certificate matches the algorithm.
replace.<variable_name>	See Section 3.1.1.

Table 1: Configuration parameters.

3.1.1 Configuration of the test cases

In the cpt.ini file it is possible to specify parameters that configure the test cases by exchanging placeholders in the test cases with the configured values. Therefore, it is for instance possible to run the tool one time for producing certificates containing RSA keys and run it a second time for producing certificates containing EC

3 https://www.bouncycastle.org/java.html

keys. For performing this it is not necessary to change the existent test cases but to properly configure the tool. This mechanism is generic and there are no predefined variables. This works as follows:

Specify a parameter of the form *replace.<paramName>* in the cpt.ini file. Use only the name of the parameter *${<paramName>}* in the XML file of the PKI objects of one or more test cases. After reading the test cases from the filesystem, the CPT replaces the placeholders with the corresponding values configured in the cpt.ini file. For example Listing 2 shows an extract of the cpt.ini file and Listing 3 the original XML specification of PKI objects. After running the tool, a certificate for this test case is created which is using effectively the XML specification of Listing 4.

```
[...]
replace.publicKey=RSA,2048
[...]
```

Listing 2: Example of a configuration parameter specified in cpt.ini to be replaced in a test case.

```
[...]
<PublicKey type="pretty">${publicKey}</PublicKey>
[...]
```

Listing 3: XML specification of a PKIObjects in the filesystem.

```
[...]
<PublicKey type="pretty">RSA,2048</PublicKey>
[...]
```

Listing 4: Effective XML specification of a PKIObjects after replacing the configured parameter.

The supported values for public keys are:

1. RSA, <even integer>
2. ECDSA, <Bouncy Castle supported curve[4]>
3. ECDH, <Bouncy Castle supported curve>

By using this generic configuration mechanism it is possible to realise complex scenarios for testing applications and libraries.

3.2 testReport.txt

This file is the template file used as the basis for the test report that is produced in the output directory of each test case. The content of this file is copied into the corresponding directory. The user of the tool can specify the content of this file. Additionally, in this template there exist concrete parameters that contain information about the test case or the test object. A list of these parameters is shown in Table 2 and an example of this file in Listing 5. A parameter can be placed at any position in this template file. When the tool runs, the parameters are replaced with the corresponding values.

Parameter	Function
${testcase.id}	Corresponds to the content of the id attribute of the testCase tag of the XML specification of a test case.
${testcase.purpose}	Corresponds to the content of the Purpose tag of the XML specification of a test case.
${report.testobject.name}	This is the value of the parameter report.testobject.name configured in the cpt.ini file.

4 http://www.bouncycastle.org/wiki/pages/viewpage.action?pageId=362269

${report.testobject.version}	This is the value of the parameter report.testobject.version configured in the cpt.ini file.
${testcase.expectedResult}	Corresponds to the content of the Text tag of the ExpectedResult tag of the XML specification of a test case.
${testcase.severity}	Corresponds to the content of the Severity Tag of the XML specification of a test case.
${testcase.data}	Is a list of IDs of the certificates and CRLs that have been produced for a test case.

Table 2: Replaceable parameters of the test report template file.

```
Test Case: ${testcase.id}

Purpose: ${testcase.purpose}

Test Object (Name): ${report.testobject.name}

Test Object (Version): ${report.testobject.version}

Test executed at: _____________________________

Test executed by: _____________________________

Result: _____________________________

Expected Result: ${testcase.expectedResult}

Severity: ${testcase.severity}

Data:

${testcase.data}

Remarks: _____________________________

_____________________________________

_____________________________________

_____________________________________
```

Listing 5: An example of the testReport.txt file.

4 Running the CPT

In order to run the CPT you need the command line interface of the underlying operating system. The following command must be given:

java -cp "lib/*" de.mtg.certpathtest.CertificationPathTest -o <CPT output folder> -d <CPT testcases folder>

or the shortcuts *run.sh*[5] for Unix-like systems or *run.bat* for Windows with default parameters can be used. The supported command lines argument of the tool are given in Table 3.

Parameter	Function
-h	Shows a help page for the input parameters of CPT.
-o <directory name>	Specifies the output directory for the certificates, revocation lists, E-mail files, and OCSP responses; i.e. the CPT output folder. This argument is mandatory.
-d <directory name>	Specifies the input directory from which the test cases in form of XML file are read from, i.e. the CPT testcases folder. This argument is mandatory.
-c <path to file>	Specifies the location of the configuration file of CTP. Default value is *cpt.ini*. The existence of a configuration file is mandatory.
-p <pretty value according to [CPT_TS], Table 12>	Outputs the value of a public key in raw format according to [CPT_TS], Table 12. The public key is generated by this provided pretty value. This value can be used for example in PKI Objects definitions.
-s	Starts the CPT in server mode. In this mode CPT does not produce any certificates but runs an HTTP and/or LDAP server with previously created objects.

Table 3: Command line arguments.

The tool reads from the filesystem XML files located under the directory specified with the "-d" option of the command line. These XML files contain test cases according to [TR-03124-2]. Afterwards it reads the XML files that contain the corresponding PKIObjects definitions. These are located under the directory *pkiObjects*. Then the tool replaces the parameters that are configured in the cpt.ini file into the pkiObjects and creates the certificates, revocation lists, OCSP responses, test reports etc. These are placed under the directory specified with the "-o" command line argument. For each test case a directory containing the data for this test case is created. The name of this directory is the ID of the test case.

When all test cases and PKIObjects have been processed the tool stops, except if HTTP and/or LDAP servers are configured to be active (see Section 4.2). If any errors occurs this is specified in the log file and/or console. If HTTP is configured to be active an OCSP responder over HTTP is also started.

After the creation of certificates, revocation lists, and OCSP responses the data for this run is archived under the directory *archives*. This allows to reproduce the tests, identify possible errors, run tests against libraries in a future point in time, and also to attach the effective data of a test into a report. The files of a run are marked by the same timestamp to associate them to each other and provide a reference to the time when the run took place. The list of archive files is presented in Table 4.

5 Run *chmod u+x run.sh* to make this file executable.

Parameter	Function
<timestamp>-CERT.zip	A zip file of the CPT output folder.
<timestamp>-CONF.zip	A zip file of the CPT pkiObjects folder.
<timestamp>-cpt.ini	The configuration file cpt.ini.
<timestamp>-HTTP.zip	A zip file of the folder configured at *http.resources.directory* parameter.
<timestamp>-LDAP.ldif	A backup file of the LDAP server in the LDIF Format [RFC 2849].
<timestamp>-SETUP.zip	A zip file of the CPT setup folder.

Table 4: Files of archiving.

4.1 Structure of the output directory

The name of the output directory for each test case is the ID of the test case. In each output directory the certificates, revocation lists, E-mail files, and OCSP responses are being written. The names of the files and their function are shown in Table 5. This structure allows automating the testing procedure for a library or application.

Parameter	Function
<ID>.pem <ID>.TC.pem.key	The private key corresponding to the public key contained in the certificate with this ID in PKCS8 / PEM format.
<ID>.TC.crt	The target certificate with this ID in DER format.
<ID>.TC.pem.crt	The target certificate with this ID in PEM format.
<ID>.TA.crt	The certificate of the trust anchor with this ID in DER format.
<ID>.TA.pem.crt	The certificate of the trust anchor with this ID in PEM format.
<ID>.CA.crt	The certificate which is neither a target certificate nor belongs to a trust anchor with this ID in DER format.
<ID>.CA.pem.crt	The certificate which is neither a target certificate nor belongs to a trust anchor with this ID in PEM format.
crls	Under this directory all revocation lists are stored. When this directory exists, this means that for this test case the checking of revocation information should be activated.
crls/<ID>.crl	The revocation list with this ID in DER format.

crls/<ID>.pem.crl	The revocation list with this ID in PEM format.
paths	Under this directory information and data about the certification path of this test case are stored. This may help the tester to automate steps during the test. If the number of certificates in the PKIObjects is not equal to the number of certificates in the path, no information is written in this directory. However, if the element *Path* is specified in the PKI objects, then this information is written.
paths/issuedTo.pem	Contains the concatenated certificates in the certification path in PEM format, including the trust anchor. The first certificate is the target certificate and the last certificate is the trust anchor.
paths/issuedBy.pem	Contains the concatenated certificates in the certification path in PEM format, including the trust anchor. The first certificate is the certificate of the trust anchor and the last certificate is the target certificate.
paths/issuedToNoTA.pem	This file is for example useful when OpenVPN is the application under test.
paths/issuedByList.txt	A list containing the filenames of the certificates in the certification path, including the trust anchor. The list is given in the issuedBy direction, that is the first filename is that of the trust anchor and the last filename is that of the target certificate. The name of each file is placed in a new line.
paths/issuedToList.txt	A list containing the filenames of the certificates in the certification path, including the trust anchor. The list is given in the issuedTo direction, that is the first filename is that of the target certificate and the last filename is that of the trust anchor. The name of each file is placed in a new line.
smime	Under this directory a signed email is placed.
smime/<ID>.eml	The signed email for the test case with this ID. This email is signed with the key of the TC certificate.-
ocspResponses	Under this directory OCSP responses are written.
ocspResponses/<ID>.ocsp.der	The OCSP response with this ID in DER format.

Table 5: Directories and files in the CPT output folder.

4.2 CRL Server

When at least one *CRL* Element in the PKIObjects with a *Location* element set exists and at least one of the parameters *http.use* or *ldap.use* in the configuration file is set to true, then the CPT starts an HTTP and/or

LDAP server that makes available certificate revocation lists. These can be downloaded by applications that evaluate the CRL distribution points extensions. If *http.use* is set to true, also an OCSP responder starts.

The HTTP server and/or an LDAP server start after the creation of certificates and revocation lists. CPT does not end operation and runs until it is stopped by the user (see Section4.2.2). The DN of the administrator of the LDAP directory is *uid=admin,ou=system*.

The OCSP responder always provides a response based on the specifications made in the respective test case definition. The OCSP request it receives is completely ignored. When the HTTP server is started in the same run of the CPT in which the test data is created, then the OCSP responses' time fields are set with respect to the point of time of the sending of the response. This is different if the CPT is started in server mode, as elaborated in the following subsection.

4.2.1 Server Mode

CPT starts in the server mode by setting the "-s" option in the arguments of the program. In this mode certificates, revocation lists, and OCSP responses are not created, but previously created revocation lists become available over an HTTP server and/or LDAP server, depending on the values of the *http.use* and *ldap.use* parameters. This is useful when CPT has already created certificates and revocation lists, but has been stopped by the user, who however wants to continue the tests. For using the revocation lists of previous runs of CPT the parameters *http.resources.directory* and *ldap.resources.directory* must point to the corresponding directories of the appropriate run. In server mode, only previously created revocation lists, which are static data are provided. OCSP responses are not available from the HTTP server in server mode.

4.2.2 Stopping CPT

There are two methods to stop CPT when an HTTP and/or LDAP server run.

The first method is by deleting the file *cpt.lck*. This file is created by CPT in the directory where the user started it. CPT scans periodically for the existence of this file. When this file does not exist, then CPT shuts down the servers and stops operation. This mode is helpful when the system where CPT runs is headless, i.e. no graphical environment exists.

The second method can be applied when the parameter *showGUI* is set to true. In this case the window shown in Figure 1 is displayed. By pressing the "Stop Application" button it is possible to stop the tool. Additionally, in this window the status of the HTTP server and the LDAP server is displayed.

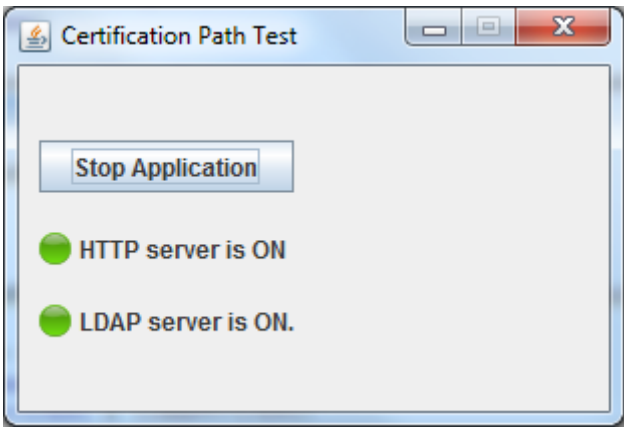

Figure 1: Window enabling the user to stop the application

5 Troubleshooting

The tool creates a detailed output about its function in form of a log file. Please consult the log file for identifying any issues. Especially ERROR log entries are helpful for isolating any problems. CPT exits without creating any certificates if the XML file of a test case or a PKIObjects contain technical errors. These must be corrected before running the tool. The tool provides proper hints where a mistake is located.

Reference Documentation

[CPT-TS] Federal Office for Information Security: Certification Path Validation Test Tool - Test Specification, Januar 2018.

[RFC 2849] G. Good, The LDAP Data Interchange Format (LDIF) - Technical Specification, IETF Request For Comments 2849, June 2000.

[TR-03124-2] Federal Office for Information Security, Technical Guideline TR-03124-2 eID-Client – Part 2: Conformance Test Specification, Version 1.2.