cryptoVision

Specification of the cryptovision TSE v2

in addition to the TR-03151

Version 2.16 • 2023-03-23

cv cryptovision GmbH • Munscheidstr. 14 • 45886 Gelsenkirchen • Germany
www.cryptovision.com • info@cryptovision.com • +49-209-167-2450

Content

1 Document Organization

This document provides the specification of cryptovision TSE with special attention on data handling, logging and functional extensions.

Extreme care was taken to define each item to be compatible with and never contradict the definitions in TR-03151 [2] or Klarstellungen [1] and to describe the extensions in their entirety.

Chapter → 2 Technical Characteristics presents the basic specifications.

Chapter → 3 ICS Supplementary Information presents technical details required by Klarstellungen [1].

Chapter → 4 additionalData Handling provides details on the handling of external and internal data.

Chapter → 5 Log Specification defines TSE audit logs.

Chapter → 6 Functional Extensions gives an overview of additional SE-API functions.

Note: Any deviations from the BSI specification in previous versions of the TSE have been removed / fixed (especially → 4.2 and → 5.2)**.**

1.1 Additional Reading

This document provides concise information on the cryptovision TSE. To keep information organized and unambiguous, this document does however **not** repeat information already available elsewhere.

So please be sure to also read the following documents:

- Technical Guideline BSI TR-03151: Secure Element API (→ [2])
- Klarstellungen und Anwendungshinweise zu BSI TR-03153 und BSI-CC-PP-0105-V2-2020 (→ [1])
- Technische Richtlinie BSI TR-03153: Technische Sicherheitseinrichtung für elektronische Aufzeichnungssysteme (→ [3])
- cryptovision SE-API documentation (→ [4] and [5])

For information on the data relevant to be signed, you may also want to consider

- Digitale Schnittstelle der Finanzverwaltung für Kassensysteme (DSFinV-K) (→ [6])

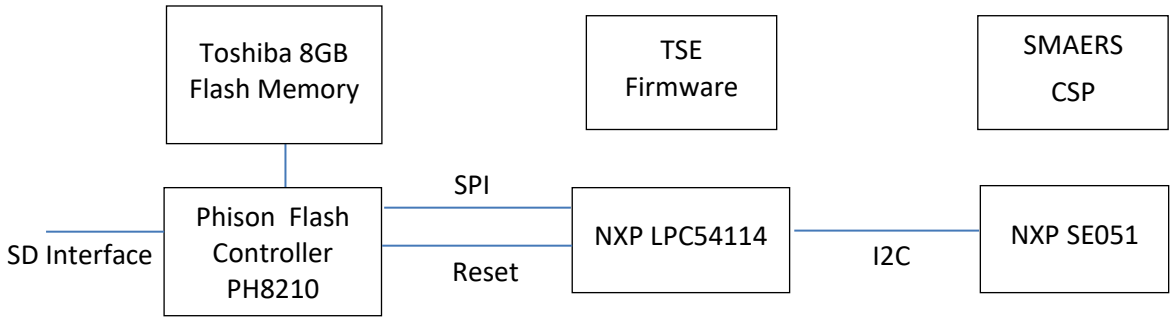
2 Technical Characteristics

This chapter provides the technical characteristics of the cryptovision TSE.

We describe the general architecture and give detailed information on several special aspects.

2.1 TSE Architecture

The cryptovision TSE is using the "platform architecture" as specified by the BSI in PP-0104 [7] (ch. 1.2) (named "Plattform-Modell" in the Klarstellungen [1] (ch. 2.1.1)).



The cryptovision TSE (version v2) consists of the following components:

- Hardware – Phison Micro-SD card in version 1.4 with:
  - Secure Element NXP SE051
  - General Purpose IC NXP LPC54114
  - Toshiba 8 GB Flash Memory
  - Phison Flash Controller PH8210
- Software:
  - cryptovision „Jacolyn" → CSP – Cryptographic Service Provider v2.0.3 Rev 18611 RC12
  - cryptovision → SMAERS – Security Module Application for Record-keeping Systems v2.0.3 Rev 18611 RC12
  - cryptovision → TSE Firmware for communication via „einheitliche digitale Schnittstelle" (SE-API) according to TR-03151 and TR-03153, v2.0 with identifier 565701
  - Phison Flash-Controller Firmware to operate the Flash Memory; Version b20

The security module NXP SE 051 runs the Java Card Operating System JCOP 4 SE051 v4.7 R3.01.11 and is the platform for CSP and SMAERS.

2.2 TSE Certification and Version Information

For certification and software version information, please be referred to the following chapters:

- CSP – Cryptographic Service Provider
- SMAERS – Security Module Application for Record-keeping Systems
- TSE Firmware

2.3 TSE Storage

The cryptovision TSE provides storage for transaction as well as system or audit log messages.

It uses an 8 GByte Toshiba Flash Memory chip. This is split into 2 GByte User Data which can be used with standard SD-Card commands and 6 GByte TSE internal memory for configuration and log data.

Due to special addressing which increases the robustness and long-term stability the remaining 6 GByte will be used as 3 GByte for the following data:

- Configuration: 8 KByte (8,192 Byte),
- Index Data: 447,99 MByte (469,753,856 Byte)
- Log Entries: 2,5625 GByte (2,751,463,424 Byte).

The log entry storage is used to store plain log entries and not the ".tar"-file used in export, which would require much more memory space. Thus, the ".tar"- file received in export is typically multiple times bigger than the used memory on the TSE.

2.3.1 Capacity - Number of Transactions

Memory size of a TSE is best expressed by the number of transactions that can be stored concurrently.

Some assumptions have been made to calculate this value:

- 195 bytes overhead per signature (time, counter, signature fields, ASN.1 encoding)
- 2 signatures per transaction (start, finish)
- no `processData` in startTransaction()

Taking into account the `processData` from finishTransaction(), the maximum number of transactions stored is calculated as follows:

#transactions ≈ 2,751,463,424 Byte / (`processData` + 2 * 195 Byte)

This gives these results:

- 1.98 million transactions at 1000 Byte processData
- 5.62 million transactions at 100 Byte processData
- 7.06 million transactions at 0 Byte processData

Note: Log messages can be deleted from the TSE after having been exported.
This frees memory for further log messages.

Index-Entries

Each index entry has a size of 32 Byte. Thus, 14.6 million indexes can be addressed. Due to internal memory management 14.4 million entries can be used which is sufficient to address all stored entries (see calculation of → Capacity - Number of Transactions above).

2.3.2 Reliability and Data Retention

The Toshiba Flash Memory used features the following data retention:

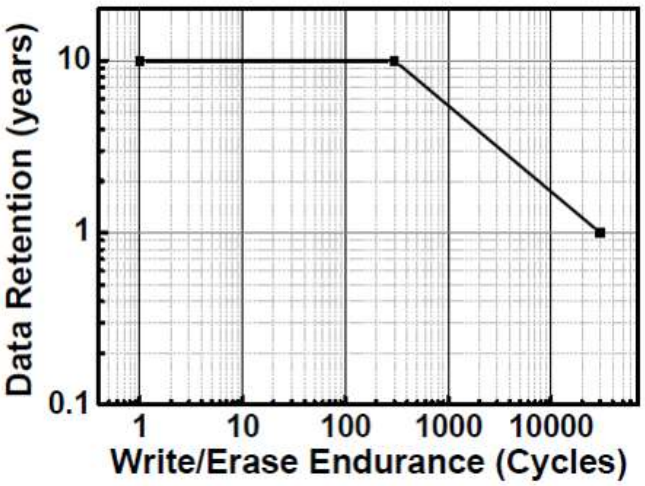


Figure 1 Flash memory data retention

Given a transaction data size of up to 600 Bytes, typically 4 (but at most 8) sectors of 512 bytes each are written to flash. The Flash-Controller uses wear-levelling to evenly distribute data written to all available memory cells. The flash vendor states a typical data retention of 10 years even when a memory cell is written three hundred times (see diagram above).

Therefor – even if 8 sectors are written during each transaction – about 236 million transactions can be performed and their data read for 10 years (which is not a required feature):

3 GByte * 300 / 512 Bytes / 8 = 235,929,600

2.4 TSE Characteristics

The following sections detail out the product's characteristics.

2.4.1 TSE Limitations

The following limitations need to be considered:

- The TSE supports up to 128 clients at the same time and up to 256 different clients over its entire lifetime.
- The TSE can start up to 320 transactions in parallel.
- The TSE uses a transaction and a signature counter of 4 bytes. The counters are implemented in the Common Criteria evaluated platform NXP JCOP 4.7 and used by the applets CSP and SMAERS. In case of a counter overflow, the platform throws an error before increasing the counter, which is then passed on to the applets and the TSE user. I.e., counter overflow is not possible and any function that tries to increase the counter will fail with an error.

2.4.2 Timing Characteristics

The timing of signatures does not change with the number of concurrently open transactions / registered clients:

Any signature operation (start/update/finish) will take approximately 125 ms.

2.4.3 TSE Description

The description of the TSE is a fixed string, which cannot be changed: “cryptovision TSEv2, 565701, X..X”, where “X..X” is a placeholder for a 32 character long and unique hardware identifier.

2.5 SE-API – Secure Element API

The programming interface for the cryptovision TSE is split into two parts, the certified → TSE Firmware with its Transport Layer interface and the → SE-API – Client-Side Implementation, available as e.g. "C" and Java library.

Note: **All data verification, validation and processing are solely performed in the → TSE Firmware.**

Therefor the cryptovision TSE is always used in certified mode regardless of the → SE-API – Client-Side Implementation. Particularly, the SE-API does not provide any functionality to create, simulate, change, or manipulate data, e.g., time, logs, counters, signatures. It only contains functions for convenient communication with the TSE. Third-party implementations are feasible and do not invalidate the certification of the cryptovision TSE.

2.5.1 TSE Firmware

The cryptovision TSE Firmware implements any data verification, evaluation and processing, as well as a low-level programming interface – namely the "TSE Transport Layer" – closely following the BSI specification in TR-03151 (→ [2]) and Klarstellungen (→ [1]). It is certified according to TR-03151 and TR-03153 in BSI-K-TR-0482.

The specification for the TSE Transport Layer is found in the SE-API documentation (→ [4] and [5]).

2.5.2 SE-API – Client-Side Implementation

The cryptovision SE-API implements the high-level programming interface as specified by BSI in TR-03151 (→ [2]) and Klarstellungen (→ [1]).

cryptovision delivers basic implementations for "C" and Java, as well as several language and operating system specific adaptions based on these.
An in-depth specification is found in the SE-API documentation (→ [4] and [5]).

Note: The cryptovision SE-API implementations are only provided for easy system integration. They might as well be replaced by custom implementations, which use the TSE Transport Layer. The TSE Transport Layer is the relevant and evaluated interface of the cryptovision TSE v2.

Note: No data verification or computation is performed in the SE-API – Client-Side Implementation. Either of the cryptovision delivered software packages just transports data between the interface and the physical TSE.

2.6 SMAERS – Security Module Application for Record-keeping Systems

The cryptovision SMAERS is a Java Card applet based on the cryptovision → CSP – Cryptographic Service Provider. It provides the functionality defined in BSI-CC-PP-105v2 (→ [8]), e.g., transaction counters, and the interface between the CSP and the outside world.

The SMAERS applet is used in version v2.0.3 Rev 18611 RC12.
It is certified according to BSI-CC-PP-105v2 in BSI-DSZ-CC-1170.

2.7 CSP – Cryptographic Service Provider

The cryptovision CSP consists of the EAL6+ certified Java Card platform using the JCOP 4 operating system and the cryptovision CSP layer. It provides the functionality specified in BSI-CC-PP-104 (→ [7]) and BSI-CC-PP-107 (→ [9]), e.g., signature counters as well as time stamp and audit functionality.

The CSP library is used in version v2.0.3 Rev 18611 RC12.
It is certified according to BSI-CC-PP-104 and BSI-CC-PP-107 in BSI-DSZ-CC-1119.

2.8 PKI Concept

The PKI concept is available in a separate document, which is provided to the certification body.

3 ICS Supplementary Information

The Klarstellungen [1] (ch. 2.1.1) require additional information on the Security Module (SM) not provided by the Information Conformance Statement (ICS). The required information is presented in → Table 1.

Table 1 ICS Supplementary Information

Required Information	cryptovision Security Module (SM) Implementation
Are several SMAERS entities combined into a single SMAERS?	The SM provides a single SMAERS entity only.
Does the architecture include several SMAERS?	The SM provides a single SMAERS only.
Are several CSP entities operated in a single CSP? Does the CSP use / provide clustered operation?	The SM provides a single CSP entity only. The CSP is operated “stand-alone”.
Does the architecture include several CSP?	The cryptovision SM provides a single CSP only.
In which way is the SMAERS connected to the CSP? Are there “remote CSP” connections?	The cryptovision SM is implemented using the **platform model**. There are **no** remote connections.
How does an ERS access the TSE?	An ERS accesses the TSE via the “Transport Layer” as documented in [5] or [4]. Any authentication, data verification (e.g., the clientId provided) or parameter evaluation is solely done in the TSE firmware and security module. In addition, cryptovision provides an SE-API in source code form (“C” or Java). This SE-API implementation by cryptovision does **not** do any authentication, verification or evaluation.

4 additionalData Handling

The cryptovision TSE handles additinalExternalData and additionalInternalData fields as specified in TR-03151 [2] and Klarstellungen [1].

4.1 additionalExternalData in StartTransaction and FinishTransaction Commands

The parameter `additionalData` of the StartTransaction (→ [2], ch. 4.4.1) and FinishTransaction (→ [2], ch. 4.4.3) commands are defined as “Reserved for future use”. It is to be included in the TSE signature as `additionalExternalData` (→ [2], ch. 2.3.1).

The cryptovision TSE does support this parameter.

When specified / non-null, it will be included in the signature as specified in TR-03151 [2].
When not specified / null, `additionalExternalData` will be omitted from the signature input.

This fulfills the specification in TR-03151 and Klarstellungen (→ [1], ch. 3,3) exactly.

4.2 additionalInternalData

`additionalInternalData` is not used / written to meet the specification in Klarstellungen (→ [1], ch. 3.3.

Note: In particular, when compared to TSE version 1, the `Initialize` function (→ [2], ch. 4.3.1) **does not write `additionalInternalData`.**

5 Log Specification

The cryptovision TSE generates audit and system logs as specified in PP-0105-V2 [8], TR-03151 [2] and Klarstellungen [1].

5.1 Audit Logs

Audit entries are generated for the events listed in → Table 2 and are stored and exported as audit logs.

The table is complete; seemingly missing IDs are not used in a CSP configured for the SMAERS application.

Table 2 Audit types

ID	Name	Description
01	AuditStartup	At finalization of CSP pre-personalization. However, in case of TSE, no audit key exists at this point, so this audit log will never occur. Instead, AuditConfigure will be the first audit log.
03	RetriesExhausted	When exceeding the retry-counter for a PIN
04	AuditConfigure	On creation, deletion, or modification of audit configuration. This audit record will be created during SMAERS pre-personalization and will be exported as first audit log later.
06	SetTime	At each update of the time
07	UCP	During installation of an “Update Code Package” (UCP)

Audit logs are formatted according to TR-03151 (→ [2], ch.2.1); seAuditData is defined in → Table 3.

Table 3 Contents of seAuditData

Data field	Tag	Data type	Mandatory?	Description
Audit type	0x8A	BYTE	m	ID from → Table 2
Result	0x8B	BYTE	m	success (0x01), error (0x00)
Time stamp	0x8C	INTEGER	m	Unix Time encoded in an 8-byte INTEGER
Identity	0x8D	PrintableString	o	Empty if “Unidentified user”, otherwise Subject as configured in personalization
Additional data	0x8E	Table 4 -7	o	Additional data depending on the audit type (Table 4 - 7)

5.1.1.1 Additional Data in RetriesExhausted Audit Logs

Table 4 Additional data of the "RetriesExhausted" audit log

Data field	Tag	Data type	Mandatory?	Description
Additional data	0x8E	INTEGER	m	ID of authentication object

5.1.1.2 Additional Data in ConfigureAudit Audit Logs

Table 5 Additional data of the "ConfigureAudit" audit log

Data field	Tag	Data type	Mandatory?	Description
Additional data	0x8E	OCTET STRING	m	ID || old configuration || new configuration

5.1.1.3 Additional Data in SetTime Audit Logs

The "Additional data" field in SetTime Audit Logs (→ Table 3) will be present in case of a successful time update only.

Table 6 Additional data of the "SetTime" audit log

Data field	Tag	Data type	Mandatory?	Description
Additional data	0x8E	INTEGER	m	Unix Time encoded in an 8-byte INTEGER

5.1.1.4 Additional Data in Update Code Package (UCP) Audit Logs

Table 7 Additional data of the "UCP" audit log

Data field	Tag	Data type	Mandatory?	Description
Additional data	0x8E	OCTED STRING	m	current CSP version (2 byte) || version of update package (2 byte)

5.2 System Logs

System logs are generated as specified in TR-03151 [2], PP-SMAERS [8], and Klarstellungen [1].

Note: The cryptovision TSE does not generate any additional system logs.

Additional log types as previously generated by TSE Version 1 have been replaced by logs defined in Klarstellungen [1].

6 Functional Extensions

The SE-API as specified in the TR-03151 [2] and Klarstellungen [1] has been cautiously extended.

These functions allow configuration of the TSE or provide detailed information on the TSE. The added functions do not interfere with the BSI SE-API ([2],[1]). Especially, there are no functions to manipulate current, future, or past logs and there are also no functions that prohibit proper use of the TSE.

Note: The naming in this chapter only gives the method names used in the Java SE-API;
for the "C" SE-API the function names have to be prepended with **`se_`**;
for the "extended" C SE-API an **`Ex`** is appended to the "C" SE-API function names.

Detailed information on each of these functions is found in the cryptovision SE-API documentation (→ [5] and [4]).

6.1 Extensions to the Export function

The cryptovision SE-API provides additional export functions / methods to provide I/O to a file (specified by its `fileName`, “C” and Java), an `OutputStream` (Java only) or via a call-back function (“C” only).

6.1.1 exportData

`exportData()` has been implemented in additional variants (see above).

6.1.2 exportMoreData and deleteStoredDataUpTo

`exportMoreData()` can be used to export data beginning at the last seen log entry.
`deleteStoredDataUpTo()` can be used to delete data up to a specific log entry. Unexported data cannot be deleted.

These two functions - counterparts of each other - speed up data export considerably. Additional variants of `exportMoreData()` exist like for `exportData()`.

6.1.3 getExportSize

`getExportSize()` determines the size of data to be exported. The parameters are identical to those in `exportData()` or `exportMoreData()` respectively.

6.1.4 getNextSignatureCounter

`getNextSignatureCounter()` returns the value of the signature counter of the first log message stored in memory but not yet exported.

6.1.5 getMinSignatureCounter and getMaxSignatureCounter

`getMinSignatureCounter()` and `getMaxSignatureCounter()` determine the minimal and maximal value of the signature counter within the log messages stored in memory.

6.1.6 exportPublicKey

`exportPublicKey()` returns the public signature key from the certificate.

6.2 Legacy Extensions

The functions / methods in this group have been retained to some extend allow backward compatibility with the TSE v1.

Due to differing requirements, parameters to the functions may differ from those they map to. Nevertheless, the legacy extensions are only used for mapping names and do not provide additional functions.

6.2.1 mapERStoKey and unmapERS

These commands directly map to `registerClient()` and `deregisterClient()` as defined in Klarstellungen [1].

6.2.2 deactivateTSE and activateTSE

These commands directly map to `lockTransactionLogging()` and `unlockTransactionLogging()` as defined in Klarstellungen [1].

6.3 TSE Status

The cryptovision TSE provides detailed information on its current status.

6.3.1 getOpenTransactions

`getOpenTransactions()` retrieves the list of dangling ("open") transactions from the SE.

6.3.2 getTransactionCounter

`getTransactionCounter()` returns the latest transaction number.

6.3.3 getSignatureCounter

`getSignatureCounter()` returns the current signature counter.

6.3.4 listClients

`listClients()` returns a list of the currently mapped cash registers (clientID).
It goes with the `registerClient()` and `deregisterClient()` commands.

6.3.5 getERSMappings

`getERSMappings()` returns an ASN.1 encoded sequence of mappings of ERSs (clientID) to key serial numbers (keyID).

6.3.6 getLifeCycleState

`getLifeCycleState()` returns the current life cycle state.

6.3.7 getTemperature

Returns the device temperature as measured by LPC54114.

6.4 TSE Memory

Several functions / methods are provided to determine the TSE memory status.

6.4.1 getTotalLogMemory

`getTotalLogMemory()` returns the size of the log memory.

6.4.2 getAvailableLogMemory

`getAvailableLogMemory()` returns the remaining free log memory.

6.4.3 getWearIndicator

`getWearIndicator()` returns an indicator for data retention.

6.4.4 selftest

`selftest()` performs and in depth test of stored data and returns health indicators.

6.5 TSE PIN Handling

The functions / methods in this chapter provide information on PIN status and allow PUK initialization.

6.5.1 getPinStatus

`getPinStatus()` retrieves the current "transport" status of the PINs and PUKs defined for the separate users (namely the `Admin` and `TimeAdmin` users) of the TSE.

6.5.2 initializePuk

`initializePuk()` switches a PUK from "transport" status to "use" status and assigns the PUK value.

PINs are initialized by `unblockUser()`.

6.6 TSE Information

Functions / methods in this group provide static information on the TSE.

This data will not change during TSE lifetime unless the TSE firmware is updated.

6.6.1 getCertificationId

`getCertificationId()` returns the (BSI) certification ID of the cryptovision TSE.

6.6.2 getTimeSyncInterval

`getTimeSyncInterval()` retrieves the proposed update interval for the CSP time base.

6.6.3 getCertificate

`getCertificate()` returns the certificate attesting the signature key – as opposed to the entire chain returned as tar file by `exportCertificates()`.

6.6.4 getCertificateExpirationDate

`getCertificateExpirationDate()` returns the certificate expiration date.

6.6.5 getSignatureAlgorithm

The command `getSignatureAlgorithm()` returns the signature algorithm used.

6.7 Client-Side extensions

SE-API Client-Side implementations provide additional APIs, which do not interact with the TSE device or firmware but provide additional information, either about the Client-Side Implementation or information returned by the TSE earlier. Methods are listed here for completeness only:

6.7.1 getImplementationVersionString

Returns the version of Client-Side Implementation as `String`.

6.7.2 getImplementationVersion

Return the version of Client-Side Implementation in three bytes.

6.7.3 getUniqueId

Returns the unique hardware identifier as also seen in the “description”.

6.7.4 getFirmwareId

Returns the firmware identifier as also seen in the “description”.

6.7.5 getHardwareVersion

Returns v1 or v2 respectively depending on the version of used TSE device.

6.7.6 getOsStatusOfErrorIO

To ease error handling, the cryptovision SE-API Client-Side "C" Implementation can provide the operating system error code.

The helper function `getOsStatusOfErrorIO()` returns the system error code (`errno` / `GetLastError()`) of the latest failed TSE I/O operation.

6.8 TSE Servicing

In case of future need, the TSE supports protected updates.

6.8.1 updateFirmware

`updateFirmware()` loads an encrypted and signed binary into the TSE. Only after successful signature verification and decryption, the new firmware is stored permanently.

6.9 TSE Hardware Operation

6.9.1 TSE Startup

This command powers up the Secure Element and initializes TSE RAM. Any other command is rejected after a power cycle or after `shutdown`() and before this command is executed.

6.9.2 TSE Shutdown

This command should be send to the TSE if it is no longer in use, e.g. if the cash register software is closed. TSE powers off the Secure Element to save power. This is especially interesting on mobile / battery powered devices.

References

[1] BSI, “Klarstellungen und Anwendungshinweise zu BSI TR-03153 und BSI-CC-PP-0105-V2-2020,” 26 November 2020. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03153/TR-03153_Anwendungshinweise.html.

[2] BSI, “Technical Guideline BSI TR-03151 Secure Element API (SE API) - Version 1.0.1,” 20 December 2018. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03151/TR-03151.html.

[3] BSI, “Technische Richtlinie BSI TR-03153 Technische Sicherheitseinrichtung für elektronische Aufzeichnungssysteme,” 20 December 2018. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03153/TR-03153.html.

[4] cryptovision, “cryptovision SE-API ("C"),” 2022. [Online]. Available: https://support.cryptovision.com/jira/servicedesk/customer/kb/view/6324324.

[5] cryptovision, “cryptovision SE-API (Java),” 2022. [Online]. Available: https://support.cryptovision.com/jira/servicedesk/customer/kb/view/6324324.

[6] Bundeszentralamt für Steuern, “Digitale Schnittstelle der Finanzverwaltung für Kassensysteme (DSFinV-K),” 13 November 2019. [Online]. Available: https://www.bzst.de/DE/Unternehmen/Aussenpruefungen/DigitaleSchnittstelleFinV/digitaleschnittstellefinv_node.html.

[7] BSI, “BSI-CC-PP-0104-2019 - Common Criteria Protection Profile Cryptographic Service Provider,” 28 Feb 2019. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Zertifikate_CC/PP/aktuell/PP_0104.html.

[8] BSI, “BSI-CC-PP-0105-V2-2020 - Common Criteria Protection Profile - Security Module Application for Electronic Record-keeping Systems - Version 1.0,” July 2020. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Zertifikate_CC/PP/aktuell/PP_0105_0105_V2.html.

[9] BSI, “BSI-CC-PP-0107-2019 - Common Criteria Protection Profile Configurations - Cryptographic Service Provider – Time Stamp Service and Audit (PPC-CSP-TS-Au),” 8 April 2019. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Zertifikate_CC/PP/aktuell/PP_0107.html.