Dokumentation der TSE - Middleware

I. Haftungsausschluss

Das Softwareprodukt ist sowohl urheberrechtlich als auch durch andere Gesetze und Vereinbarungen über geistiges Eigentum geschützt.

Das Softwareprodukt wird lizenziert, nicht verkauft.

Das Softwareprodukt wurde vom Lizenzgeber mit größter Sorgfalt erarbeitet und unter Einschaltung wirksamer Kontrollmaßnahmen geprüft. Der Lizenzgeber schließt jedoch ausdrücklich eine Gewährleistung für das Softwareprodukt aus.

Das Softwareprodukt wird dem Lizenznehmer "so wie es ist" zur Verfügung gestellt, ohne Gewährleistung jeglicher Art, weder ausdrücklich noch konkludent. Das gesamte Risiko, das aus der Leistung des Softwareproduktes entsteht, verbleibt beim Lizenznehmer.

Der Lizenzgeber übernimmt keine Gewähr dafür, daß das Softwareprodukt den Anforderungen und Zwecken des Lizenznehmers genügt oder mit anderer von ihm ausgewählter Software bzw. Hardware zusammenarbeitet. Der Lizenzgeber weist ausdrücklich darauf hin, daß er weder irgendeine Haftung noch irgendeine juristische Verantwortung für Kosten und Folgekosten übernimmt, die sich aus dem Verwenden des Softwareproduktes (oder der Unmöglichkeit, das Softwareprodukt zu verwenden) ergeben.

In jedem Fall bleibt die gesamte Haftung des Lizenzgebers beschränkt auf den Betrag, den der Lizenznehmer für das Softwareprodukt bezahlt hat.

Alle durch Dritte geschützten Marken- und Warenzeichen unterliegen uneingeschränkt den Bestimmungen des jeweils gültigen Kennzeichenrechts und den Besitzrechten der jeweiligen eingetragenen Eigentümer. Allein aufgrund der bloßen Nennung ist nicht der Schluss zu ziehen, dass Markenzeichen nicht durch Rechte Dritter geschützt sind.

Der Lizenznehmer haftet für alle Schäden, die aufgrund von Urheberrechts- und sonstigen Schutzrechtsverletzungen entstehen.

Die in einigen Installationspaketen mitgelieferte und zum Betrieb erforderliche WormAPI-Bibliothek der Swissbit AG, Bronschhofen (Switzerland) unterliegt dem Copyright und den Lizenzbedingungen der vorgenannten Firma und darf nur in Verbindung mit der TSE-Middleware benutzt und eingesetzt werden.

II. Allgemeines

Die TSE-Middleware dient der einfachen Anbindung einer Swissbit-TSE an ein beliebiges ERP- oder Kassensystem. Dabei müssen keinerlei Programm- oder Codeteile in das Anwenderprogramm integriert oder gar einkompiliert werden, sondern die Anbindung erfolgt transparent über einen TCP-Socket. Somit gibt es keinerlei Abhängigkeiten von der verwendeten Programmiersprache der ERP- oder Kassenlösung, wie es z.B. regelmäßig bei der Integration eines "SDK" der Fall ist. Ebenso sind Updates auf beiden Seiten möglich, ohne daß die jeweils andere Seite geändert oder angepaßt werden muß. Selbst eine komplette Plattformänderung der ERP- oder Kassenlösung tangiert die TSE-Anbindung nicht.

Dieser unschätzbare Vorteil unserer Lösung für den Programmierer/Anwender bedeutet neben einer erhöhten Investitionssicherheit (die Lösung ist völlig transparent und jederzeit austauschbar) auch eine erhebliche Ersparnis von Kosten bei der Programmierung. Die TSE-Middleware wird als fertige Anwendung ausgeliefert, die Installation ist im nächsten Abschnitt detailliert erläutert.

Aufgrund der Architektur der TSE-Middleware lassen sich mit einer TSE beliebig viele Kassen betreiben (ohne Probleme auch mehr als zehn Kassen). Abhängig von der Hardware und den verfügbaren USB-Steckplätzen kann die TSE-Middleware auch beliebig viele TSE parallel an einem Server/PC verwalten. Somit ersetzt unsere TSE-Middleware gleichzeitig auch weitere Hard- und/oder Software-Komponenten (wie z.B. "LAN-TSE").

Nachdem sich unsere TSE-Middleware bereits seit vier Jahren in unzähligen Installation bewährt hat, möchten wir hier unsere Second-Edition mit einer Menge an Neuerungen, aber natürlich sämtlichen altbewährten Optionen vorstellen.

Die Swissbit-TSE und das zugehörige SDK entsprechen vollumfänglich allen gesetzlichen Vorgaben, wie sie u.a. in BSI TR-03151 (Secure Element API) festgelegt sind. Unsere Middleware setzt dabei auf diesem SDK auf. Zusätzlich verfügt unsere Middleware noch über die Möglichkeit, die von der TSE erzeugten Signaturen zu prüfen. Zusätzlich ist es sogar möglich, fremde Messages, Signaturen, Tar-Files und QR-Codes zu visualisieren, zu verifizieren und zu prüfen.

Außerdem bieten wir eine optionale Syntax- und Semantikprüfung für die von Ihnen an die TSE übergebenen Werte an.

Die TSE-Middleware wird aktuell für folgende Systeme und Plattformen entwickelt und bereitgestellt:

- Intel (amd64) mit Linux 64 Bit als Konsolenanwendung
- Intel (amd64) mit Windows 64 Bit (ab Windows 10) als GUI-Anwendung (Qt-basiert)
- ARM (arm32, arm64) mit Linux 32 Bit als Konsolenanwendung
- Intel (amd64) mit Linux 64 Bit als GUI-Anwendung (Qt-basiert), auf Anfrage
- Intel (amd64) mit Windows 64 Bit (für Windows 7 / 8 / 8.1) als GUI-Anwendung (Qt-basiert) in der Legacy-Version 1.x

Der prinzipielle Funktionsumfang, die zugrundeliegenden Bibliotheken und die Ansteuerung der TSE sind bei allen Betriebssystemen und auch auf auf allen Plattformen immer gleich. Dennoch gibt es, aufgrund der komplett anderen Philosophie von Linux und Windows, einige wichtige Unterschiede:

- unter Windows gibt es keine Möglichkeit, die Zugriffe auf ein (USB-)Gerät einzuschränken
- unter Windows gibt es keinen Mount-Befehl
- unter Windows ist die Herunterstufung der Rechte eines Daemons nicht vorgesehen

Trotz dieser Einschränkungen beim Betrieb unter Windows haben wir uns entschlossen, beide Lösungen parallel anzubieten, da am Markt viele Windows-ERP und/oder Kassen-Lösungen, teilweise auch als Einzelplätze, anzutreffen sind. Genau für diese Lösungen, die teilweise in Programmiersprachen wie VB5, VB6 (oder älteren Basic-Derivaten), PHP, Delphi, ja sogar Cobol oder FoxPro geschrieben sind, stellt aber unsere TSE-Middleware eine unverzichtbare Abstraktionsschnittstelle bereit.

Da die Systemsicherheit unter Linux als höher angesehen werden darf, empfehlen wir die Nutzung der TSE-Middleware unter Linux bei besonders unternehmenskritischen Anwendungen bzw. bei Vorhandensein von Netzwerken und/oder Linux-Servern.

Daneben gibt es natürlich auch Unterschiede zwischen der Konsolenversion und der GUI-Version. Während die GUI-Version eher auf kleine (Windows)-Installationen abzielt, kann die Konsolen-Version auch komplett ohne Oberfläche, d.h. programmgesteuert betrieben werden, z.B. wenn die TSE an einem Server in einem zentralen Serverraum angesteckt ist.

Durch das, je nach Plattform unterschiedlich ausgelegte Locking der TSE wird zudem zuverlässig sichergestellt, daß immer nur eine Instanz der Software zeitgleich auf die TSE zugreift. Dies ist für die reibungslose Funktion wichtig und eines der Alleinstellungsmerkmale unserer Software.

Die Visualisierung der Log-Messages, die Verifizierung der Signaturen und vor allem die Möglichkeit, fremde Tar-Files und Signaturen zu verarbeiten, beherrschen momentan nur sehr wenige Programme auf dem Markt (uns ist persönlich nur ein Einziges bekannt).

Wichtig!

Die meisten Datums- oder Zeitwerte der TSE werden als Unix-Timestamp erfaßt und ausgegeben. Dieser Timestamp entspricht den Sekunden seit dem 01.01.1970 und wird immer in UTC (Coordinated Universal Time) angegeben. Dies ist bei der Umrechnung in die lokale Zeitzone unbedingt zu beachten.

III. Installation

1. Linux (Intel 64Bit, ARM 32Bit)

Es wird ein aktuelles 64Bit Linux-System vorausgesetzt, welches folgende Mindestanforderungen erfüllen muß:

- OpenSSL Version 3.0.x (für die Legacy-Version 1.x wird Version 1.1.x benötigt)
- glibc in Version >= 6.0.30
- libarchive.so.13 (ebenfalls gegen OpenSSL Version 3.0.x gelinkt)
- libattr.so.1, liblzma.so.5, libacl.so.1, libxml2.so.2, libbz2.so.1.0
- icu (ab Version 63)

Es wird empfohlen, einen dedizierten User und eine eigene Gruppe für die TSE anzulegen. Damit wird einerseits ausgeschlossen, daß der TSE-User andere Teile des Systems beeinflussen kann und andererseits der Zugriff auf die TSE für andere Benutzer unterbunden.

Die TSE selbst wird an einen USB-Anschluß des Linux-Servers angesteckt und gemountet. Der genaue Befehl hängt von der jeweiligen Linux-Distribution ab. Wir gehen davon aus, daß die TSE unter "**/dev/sdc1**" ansprechbar ist, der Mountpoint der TSE "**/mnt**" und TSE-Benutzer und -gruppe "**tse**" lauten. Der Befehl zum Mounten der TSE lautet dann:

mount -o fmask=0177,dmask=0077,uid=tse /dev/sdc1 /mnt

Das Programmpaket besteht aus einem Tar-Archiv, welches sieben Dateien enthält, die alle den jeweiligen Copyrights unterliegen:

1. tse_admin (das TSE-Administrations-Programm)
2. tse_server (das TSE-Server-Programm)
3. tse_admin.conf (die Konfigurationsdatei für tse_admin)
4. tse_server.conf (die Konfigurationsdatei für tse_server)
5. tse.lic (die Lizenzdatei mit der signierten Lizenz)
6. libWormAPI.so (die Bibliothek der Swissbit AG zur TSE-Ansteuerung, mind. Version 5.8.1)
7. optional client.pl (ein Perl-Demoprogramm für Test und Simulation der Kommunikation)

Die fünf Dateien (1) ... (5) sollten in einen eigenen Ordner (z.B. "**/opt/tse**") gelegt werden. Die Dateien (1) und (2) benötigen das gesetzte Execute-Bit (z.B. 0755). Die Konfigdateien (3) und (4) sollten nur für den TSE-Benutzer lesbar sein (Vorgabe tse:tse 0600), da darin die Admin-PIN steht. Die Bibliothek (6) muß in ein System-Bibliotheksverzeichnis installiert werden (z.B. "**/usr/lib**"). Danach ist i.d.R. der Befehl "**ldconfig**" als root Benutzer aufzurufen. Der optionale Perl-Script (7) benötigt zum Aufruf eine installierte Perl-Umgebung (Aufruf "**perl -Tf /opt/tse/client.pl**"). Er dient hauptsächlich als Anregung zur Implementierung der Clients (Kassen).

Zuerst sollte man sicherstellen, daß die Programme tse_admin und tse_server fehlerfrei aufgerufen werden können, also alle Bibliotheken im System vorhanden sind. Dazu führt man am besten einmal diesen Befehl aus: "**ldd /opt/tse/tse_admin**"
Wenn dabei Zeilen/Einträge mit "not found" ausgegeben werden, so müssen die betreffenden Bibliotheken nachinstalliert werden, dazu ist je nach Distribution entsprechend vorzugehen (z.B. "**apt-get install ...**").

Als nächstes überprüfen Sie bitte die Uhrzeit des Servers (z.B. mit "**date**"). Erst wenn diese korrekt ist, sollten die Programme tse_admin oder tse_server gestartet werden.

Später sollte das Mounten und der Start des Servers in den Systemstart aufgenommen werden. Hierzu können wir Ihnen, je nach verwendetem System (Systemd oder InitV) mit Beispielen für Startscripte weiterhelfen.

Wichtig!
Vor dem ersten Aufruf der Programme tse_admin oder tse_server müssen unbedingt die zugehörigen Konfigdateien angepaßt werden. Es müssen mindestens die Admin-PIN und das TSE-Mountverzeichnis eingetragen werden.

2. Windows (Intel 64Bit)

Es wird ein aktuelles 64Bit Windows-Betriebssystem vorausgesetzt, welches folgende Mindestanforderungen erfüllen muß:

- MS-Windows 10 oder MS-Windows-Server 2016
- für die Legacy-Version 1.x MS-Windows 7 oder MS-Windows-Server 2008
- MS-VisualC-Redistributable-Kit 2015-2019 (mindestens 14.38.x)

Die TSE selbst wird an einen USB-Anschluß des Windows-PC oder -Servers angesteckt. Sie muß anschließend als Laufwerk mit einem eigenen Buchstaben angezeigt werden und ansprechbar sein.

Das Programmpaket besteht hier aus einem Rar-Archiv, welches u.a. folgende Dateien enthält, die alle den jeweiligen Copyrights unterliegen:

1. tse_admin.exe (das TSE-Administrations-Programm)
2. tse_server.exe (das TSE-Server-Programm)
3. tse_admin.conf (die Konfigurationsdatei für tse_admin)
4. tse_server.conf (die Konfigurationsdatei für tse_server)
5. tse.lic (die Lizenzdatei mit der signierten Lizenz)
6. libWormAPI.dll (die Bibliothek der Swissbit AG zur TSE-Ansteuerung, mind. Version 5.8.1)
7. einige Dll's und Verzeichnisse, die die Laufzeitumgebung von "Qt" bereitstellen
8. einige Dll's, die u.a. für die Signaturprüfung und den Tar-Export notwendig sind
9. optional client.pl (ein Perl-Demoprogramm für Test und Simulation der Kommunikation)
10. optional tse_send.exe (ein Kommandozeilenprogramm für Windows, falls die verwendete Programmiersprache keine Sockets benutzen kann)

Sämtliche Dateien und Ordner des Archivs müssen in ein eigenes Verzeichnis entpackt werden (z.B. **"C:\Programme\tse"**).

Als nächstes prüfen Sie bitte die Uhrzeit des Rechners. Erst wenn diese korrekt ist, sollten die Programme tse_admin.exe oder tse_server.exe gestartet werden. Der optionale Perl-Script (9) benötigt zum Aufruf eine installierte Perl-Umgebung (Aufruf "**perl -Tf /opt/tse/client.pl**"). Er dient hauptsächlich als Anregung zur Implementierung der Clients (Kassen). Mit dem ebenfalls optionalen Programm tse_send.exe (10) kann man den TSE-Server direkt ohne eigene Kasse/ERP ansteuern und so die Kommunikation testen.

Wichtig!

*Vor dem ersten Aufruf der Programme tse_admin.exe oder tse_server.exe müssen unbedingt die zugehörigen Konfigdateien angepaßt werden. Es müssen mindestens die Admin-PIN und das TSE-Laufwerk (z.B. "**F:**", nur Buchstabe und Doppelpunkt) eingetragen werden.*

IV. Lizenzierung

Das Kernstück für die Lizenzierung der TSE-Middleware stellt die Seriennummer der TSE (dies ist der SHA-2 Hash des BitStrings vom PublicKey) dar. Diese SN wird als 32-Byte-Octet ausgegeben, d.h. als Folge von 64 Hex-Zeichen. Die Lizenzierung der TSE-Middleware erfolgt immer für genau eine TSE-Seriennummer. Dabei ist es möglich, eine Lizenz für die gesamte Laufzeit des TSE-Zertifikates (maximal ca. fünf Jahre) oder auch für einzelne Monate bzw. Jahre zu erwerben.

In der Lizenz ist dabei neben dem Start- und Ende-Datum und der SN der TSE auch der Umfang der gewährten Nutzung enthalten. Folgende Bausteine sind einzeln lizenziert:

- die Plattform (Linux und/oder Windows)
- der Anwendungstyp (Konsole und/oder GUI)
- der TSE-Typ (Entwickler- oder Produktions-TSE)
- die maximale Anzahl der zugelassenen Kassen
- die Visualisierung der Message-Logs und die interne Validierung der Signaturen der TSE
- die Prüfung von kompletten Bondaten (bzw. QR-Codes), sowie deren Signatur-Validierung
- die Visualisierung von externen Message-Logs und die Validierung von externen Tar-Dateien

Aufgrund des Aufbaus und der Funktionsweise der Lizenzdatei (die Lizenz wird mit einem privaten Schlüssel signiert, den die TSE dann mit ihrem PublicKey überprüft) lassen sich Manipulationen ausschließen, ohne daß die Lizenzdatei geheim sein muß. Da die signierte Lizenz in "Base64" kodiert ist, kann die Lizenzdatei problemlos übertragen werden (z.B. per Mail).

Damit das Programm die Lizenzdatei finden kann, wird diese in der Konfigdatei im Abschnitt "**global**" hinterlegt. Der Standardwert lautet "**lizfile = tse.lic**", wobei die Datei im gleichen Ordner wie das Programm selbst gesucht wird.

Zum Testen und für Evaluierungs-Zwecke gibt es eine besondere Lizenz, die für alle Entwickler-TSE gültig ist. Wenn Sie nach Ablauf dieser Evaluierungs-Lizenz (i.d.R. drei Monate) Ihre Entwickler-TSE weiter benutzen möchten, benötigen Sie auch dafür eine gültige Lizenz, die Sie (kostenpflichtig) erwerben müssen.

Wichtig!

Nach Ablauf der Lizenz könnte man theoretisch die Uhrzeit des Rechners zurückstellen, um weiterhin mit der Lizenz arbeiten zu können, allerdings werden dann natürlich auch alle Transaktionen mit der falschen Uhrzeit erzeugt. Ebenso verweigert das Programm die Prüfung von Signaturen und/oder den Export von Transaktionen, die außerhalb des Lizenzzeitraumes liegen.

V. Hinweise vor dem Start

Wichtig!

Es sollte unbedingt beachtet werden, daß bei allen produktiven TSE's (also alle außer den TSE's im Entwicklerpaket) kein "FactoryReset" möglich ist, d.h. eine einmal gesperrte TSE ist nicht wieder zu entsperren. Dies ist auch nicht zu umgehen!

Für eine Sperrung der TSE gibt es (leider) gleich mehrere Möglichkeiten:

1. der CredentialSeed ist falsch
2. die Admin- (oder TimeAdmin-) PIN wird dreimal falsch eingegeben, das geht schnell, wenn diese z.B. in der Konfigdatei falsch hinterlegt und das Programm dreimal gestartet wird (das läßt sich zum Glück mit der PUK reparieren).
3. die PUK wird beim Entsperren des Admin oder TimeAdmin dreimal falsch eingegeben

Der Fall (1) kann in unserer Software nicht passieren, da der CredentialSeed im Programm fest einkompiliert ist. Die Möglichkeiten (2) und (3) können nur auftreten, wenn in der Konfigdatei falsche Werte stehen oder wenn bei der Änderung von PIN/PUK aktiv falsche Werte übergeben werden.

Zudem gibt es noch eine weitere wichtige Funktion bzgl. Datum/Uhrzeit. Da die Zertifikate bei den Entwickler-TSE nur sechs Monate gültig sind, kann man einen Time-Versatz angeben, so daß der TSE immer ein "aus Zertifikats-Sicht gültiges" Datum übermittelt wird, ohne daß man das Systemdatum ändern muß. Dieses Feature ist nur aktiv, wenn das Programm eine Entwickler-TSE erkennt.

Zur Demonstration der Kassenzugriffe dient das kleine Perl-Programm "client.pl". Aus dessen Quellcode kann man recht gut die einfache Implementierung der Clients (nur vier Zeilen Quellcode pro Zugriff) entnehmen. Außerdem lassen sich damit auch alle Funktionen des Serverprogramms testen.

Wichtig!

Das gleichzeitige Ausführen von tse_admin und tse_server wird über ein Locking verhindert. Da beide Programme exklusiv auf die TSE zugreifen und zudem das Admin-Programm am Ende alle aktiven Clients abmeldet, würde das ansonsten zu einem undefinierten Verhalten führen.

Wichtig!

Der Tar-Export der TSE ist im Rahmen der Datensicherung möglichst regelmäßig auszuführen, um im Falle eines Defektes oder Verlustes der TSE keine Daten zu verlieren. Vor dem Wechsel der TSE bzw. deren Außerbetriebsetzung ist dieser verpflichtend auszuführen.

VI. Admin-Programm

Mit dem Admin-Programm werden die Einricht- und Wartungsarbeiten der TSE durchgeführt. Da es für den normalen täglichen Betrieb nicht benötigt wird, ist es auch als extra Modul entworfen worden. Somit kann man nämlich bei Bedarf den Zugriff auf den Administrator einschränken. Die Adminprogramme (tse_admin) der GUI- und der Konsolenanwendung bieten exakt dieselbe Funktionalität, unterscheiden sich aber signifikant in der Bedienung. Während bei der GUI sämtliche Eingaben interaktiv über graphische Buttons und Eingabefelder erfolgen, können und müssen bei der Konsolenvariante alle Kommandos und Parameter auf der Kommandozeile übergeben werden. Deshalb kann diese Variante (wie bei Linux üblich) auch in (Bash-)Scripte eingebunden werden.

1. GUI-Version

Die GUI-Version sucht ihre Konfigurationsdatei per Default im Programmverzeichnis unter dem Namen "tse_admin.conf". Dies kann über einen Aufrufparameter übersteuert werden:

-c | --cfgfile <Dateiname> der Name (und optional der Pfad) der Konfigdatei

Weitere Kommandozeilenparameter sind aktuell nicht definiert. Alle Info-Ausgaben erfolgen bei der GUI-Version in ein Textobjekt (und teilweise zusätzlich in die Zwischenablage). Da die Applikation als Multithreading-Anwendung entworfen ist, treten auch bei länger laufenden Aktionen (z.B. bei der Initialisierung oder beim Tar-Export) keine Einschränkungen der Interaktion ("keine Rückmeldung" o.ä.) auf. Die Bedienung der GUI erfolgt über Buttons und ist damit auch per Tablet gut möglich. Zuerst wird der Aktionstyp gewählt (Info, Setup, Extras), danach innerhalb dieser Aktion der Subtyp (z.B. TSE-Setup), danach wird die Aktion per Okay-Button gestartet. Dies ist eine klassische Windows-Bedienung, die sicher keiner weiteren Erklärung bedarf. Folgende Funktionen können aktuell ausgeführt werden:

- Info | TSE-Infos
  alle Infos zur TSE werden in ein Textobjekt ausgegeben
- Info | Liz-Infos
  alle Infos zur Lizenz werden in ein Textobjekt ausgegeben
- Info | Copy SN
  die Seriennummer (Hex-String) der TSE wird in ein Textobjekt und in die Zwischenablage ausgegeben
- Info | Copy PKey
  der PublicKey der TSE wird im PEM-Format in ein Textobjekt und in die Zwischenablage ausgegeben
- Info | Copy Cert
  die Zertifikats-Chain der TSE wird im PEM-Format in ein Textobjekt und in die Zwischenablage ausgegeben

- Setup | AdminPin | Okay
  ändert die Admin-PIN der TSE
  die alte und die neue PIN müssen eingegeben werden
- Setup | TAdminPin | Okay
  ändert die Time-Admin-PIN der TSE
  die alte und die neue PIN müssen eingegeben werden
- Setup | Puk | Okay
  ändert die Super-PIN (PUK) der TSE
  die alte und die neue PUK müssen eingegeben werden
- Setup | Admin | Okay
  entsperrt den Admin mit Hilfe der PUK und setzt die neue Admin-PIN
  die PUK und die neue PIN müssen eingegeben werden
- Setup | TAdmin | Okay
  entsperrt den Time-Admin mit Hilfe der PUK und setzt die neue TimeAdmin-PIN
  die PUK und die neue PIN müssen eingegeben werden
- Setup | TSE-Setup | Okay
  führt das initiale Setup der TSE aus
  die neue PUK und die neuen PIN's für Admin und TimeAdmin müssen eingegeben werden
- Setup | TSE löschen | Okay
  löscht alle Daten der TSE
  die PIN muß eingegeben werden
- Setup | FactoryReset | Okay
  setzt die TSE komplett zurück (nur Entwickler-TSE)
  die PIN muß eingegeben werden
- Setup | FW-Update | Okay
  prüft, ob ein Firmware-Update für die TSE vorliegt und führt dieses nach Rückfrage aus (die neue Firmware ist lokal in der Dll enthalten)
- Extras | Tar-Export | Okay
  exportiert den gesamten Inhalt der TSE in das angegebene Tar-File
  das Tar-Export-File muß angegeben werden
- Extras | Msg-Export | Okay
  exportiert alle Log-Messages der TSE in ein File
  das Msg-Export-File muß angegeben werden
- Extras | TSE-Check | Okay
  liest und verifiziert alle Messages und Signaturen der TSE
  ein Protokoll-Export-File muß angegeben werden, es werden nur Fehler protokolliert

- Extras | Close-Trans | Okay
  schließt eine offene Transaktion für den jeweiligen Client
  die Transaktionsnummer muß angegeben werden
- Extras | Msg prüfen | Okay
  verifiziert die Signatur des angegebenen externen Message-Files i.V. mit dem PublicKey
  das Msg-Input-File und das PKey-Input-File muß angegeben werden
- Extras | Tar-Check | Okay
  prüft das angegebene externe Tar-File und verifiziert alle darin enthaltenen Signaturen
  das Tar-Input-File und das Protokoll-Export-File muß angegeben werden
- Extras | Cert2PKey | Okay
  extrahiert den PublicKey im PEM-Format aus dem Zertifikat im PEM-Format
  das Zertifikat-Input-File und das PKey-Export-File muß angegeben werden
- Extras | Add-Client | Okay
  registriert einen zusätzlichen Client in der TSE (nur zum Testen)
  die Client-ID muß eingegeben werden
- Extras | Del-Client | Okay
  löscht einen registrierten Client in der TSE (nur zum Testen)
  die Client-ID muß eingegeben werden

Wichtig!

Nach einem Wechsel der Admin-PIN mittels "Setup / AdminPin" muß die neue PIN unbedingt vor dem nächsten Neustart des Admin-Programms in der Konfigdatei korrekt hinterlegt werden. Beim Versuch, die PIN mittels "Setup / TSE-Setup" oder "Setup / Admin" neu zu setzen, muß diese aus Sicherheitsgründen mit der in der Konfigdatei hinterlegten PIN übereinstimmen. Ebenso wird bei den Funktionen "Setup / AdminPin", "Setup / FactoryReset" und "Setup / TSE löschen" zur Sicherheit nochmals die PIN abgefragt.

2. Konsolen-Version (nur Linux)

Die Konsolen-Version sucht ihre Konfigurationsdatei ebenfalls per Default im Programmverzeichnis unter dem Namen "tse_admin.conf". Die Info-Ausgaben erfolgen dabei direkt nach stdout (also auf die Konsole).

Usage: **tse_admin [std_opt] [ext_opt]**

Die nachfolgenden Standardoptionen (std_opt) können unabhängig von der auszuführenden Aktion immer verwendet werden:

-c | --cfgfile <Dateiname>	der Name (und optional der Pfad) der Konfigdatei
-d | --loglevel { 0, 1, 2, 3, 4 }	der Loglevel (Defaultwert s. Konfigdatei)
-f | --logfile <Dateiname>	der Pfad und Name des Log-Files (Defaultwert s. Konfigdatei)
--logappend { 1 | 0 }	soll das Log-File bei jedem Start überschrieben oder erweitert werden (Defaultwert s. Konfigdatei)
-l | --lizfile <Dateiname>	das Lizenz-File (Defaultwert s. Konfigdatei)
-m | --mountpoint <Pfad>	der Mountpoint der TSE (Defaultwert s. Konfigdatei)
-v | --version	die Ausgabe von Programmversion und Copyright-Info, dies schließt alle anderen Optionen aus
-h | --help	diese Hilfeanzeige, dies schließt alle anderen Optionen aus

Alle erweiterten Optionen (ext_opt) sind, abhängig von der Aktion, nur spezifisch gültig. Folgende Aufrufe sind z.Zt. definiert:

- tse_admin print_info
  es werden alle Infos zur TSE ausgegeben
- tse_admin print_liz
  es werden alle Infos zur Lizenz ausgegeben
- tse_admin print_sn
  die Seriennummer (Hex-String) der TSE wird ausgegeben
- tse_admin print_pkey
  der PublicKey der TSE wird im PEM-Format ausgegeben
- tse_admin print_cert_chain
  die Zertifikats-Chain der TSE wird im PEM-Format ausgegeben
- tse_admin change_admin_pin --oldpin <PIN> --newpin <PIN>
  ändert die Admin-PIN der TSE
- tse_admin change_time_pin --oldpin <PIN> --newpin <PIN>
  ändert die Time-Admin-PIN der TSE
- tse_admin change_puk --oldpuk <PUK> --newpuk <PUK>
  ändert die Super-PIN (PUK) der TSE

- tse_admin unblock_admin --adminpin <PIN> --puk <PUK>
  entsperrt den Admin mit Hilfe der PUK und setzt die neue Admin-PIN
- tse_admin unblock_time --timepin <PIN> --puk <PUK>
  entsperrt den Time-Admin mit Hilfe der PUK und setzt die neue TimeAdmin-PIN
- tse_admin setup_tse --adminpin <PIN> --timepin <PIN> --puk <PUK>
  führt das initiale Setup der TSE aus
- tse_admin delete_data --tarfile <Dateiname>
  löscht alle Daten der TSE und exportiert direkt vorher alle Daten in das angegebene Tarfile (dieser Export wird vom SDK erzwungen)
- tse_admin factory_reset
  setzt die TSE komplett zurück (nur Entwickler-TSE)
- tse_admin firmware_update [--adminpin <PIN>]
  prüft auf eine neue Firmware für die TSE und führt bei Angabe der PIN das Update aus
- tse_admin add_client --client <Ident>
  registriert einen zusätzlichen Client in der TSE
- tse_admin del_client --client <Ident>
  löscht einen registrierten Client in der TSE
- tse_admin close_trans --client <Ident> --transnr <TrNr>
  schließt eine geöffnete Transkation des jeweiligen Clients
- tse_admin export_tar --tarfile <Dateiname>
  exportiert den gesamten Inhalt der TSE in das angegebene Tar-File
- tse_admin export_msg --msgfile <Dateiname>
  exportiert alle Log-Messages der TSE in ein File
- tse_admin check_tse
  liest und verifiziert alle Messages und Signaturen der TSE
- tse_admin check_ext_msg --msgfile <Dateiname> --pkeyfile <Dateiname>
  verifiziert die Signatur des angegebenen externen Message-Files i.V. mit dem PKey
- tse_admin check_ext_tar --tarfile <Dateiname>
  prüft das angegebene externe Tar-File und verifiziert alle darin enthaltenen Signaturen
- tse_admin cert_to_pkey --certfile <Dateiname> --pkeyfile <Dateiname>
  extrahiert den PublicKey im PEM-Format aus dem Zertifikat im PEM-Format

Wichtig!

Nach einem Wechsel der Admin-PIN mittels "change_admin_pin" muß die neue PIN unbedingt vor dem nächsten Neustart des Admin-Programms in der Konfigdatei korrekt hinterlegt werden. Beim Versuch, die PIN mittels "setup_tse" oder "unblock_admin" neu zu setzen, muß diese aus Sicherheitsgründen mit der in der Konfigdatei hinterlegten PIN übereinstimmen. Ebenso wird bei den Funktionen "change_admin_pin", "factory_reset" und "delete_data" zur Sicherheit nochmals die PIN abgefragt.

VII. Server-Programm

Die Serverkomponente der TSE-Middleware ist das Kernstück für die tägliche Arbeit. Das ERP- oder Kassensystem kommuniziert über eine Socketverbindung mit dem TSE-Server. Dabei werden nur wenige Daten auf unterster Ebene ausgetauscht, so daß die Verarbeitung einfach und hocheffizient erfolgt. Die Serverprogramme der GUI- und der Konsolenanwendung bieten exakt dieselbe Funktionalität. Da sie kaum Interaktion mit dem Benutzer erfordern, unterscheidet sich hier auch die Bedienung nicht wesentlich. Lediglich die Sicherheitsaspekte sind bei der Linux-Lösung höher, da hier aufgrund des Betriebssystems andere Abstraktionsschichten möglich sind. Sowohl die GUI, als auch die Konsolenversion sind zum dauerhaften Betrieb im Hintergrund vorgesehen.

Das Server-Programm kümmert sich auch um die laufende Aktualisierung der Uhrzeit und um den täglichen Selbsttest. Dieser wird zu einer konfigurierbaren Zeit aufgerufen, was ein weiteres Alleinstellungsmerkmal unserer TSE-Middleware ist.

Sobald sich ein Client (Kasse) verbindet, wertet der Server das gesendete Token (GUID) aus, registriert bei Erfolg die Kasse und startet die angeforderte Aktion.

Aus Sicherheitsgründen können hierüber keine destruktiven administrativen Aufgaben (z.B. Setup oder Daten löschen) erledigt werden, dies übernimmt das Admin-Programm. Folgende Befehle sind aktuell im Server-Programm implementiert:

- eine Transaktion starten
- eine Transaktion updaten
- eine Transaktion beenden
- die Verfügbarkeit der TSE prüfen (KeepAlive)
- den PublicKey der TSE auslesen (z.B. für den Bondruck)
- die SerialNumber der TSE auslesen (z.B. für den Bondruck)
- den Signatur-Algorithmus und das LogTimeFormat der TSE auslesen (z.B. für den Bondruck)
- TSE- und Programm-Infos ermitteln (Zertifikats- und Lizenzende, freie Blöcke, verbleibende Signaturen, nächster Selbsttest, SDK- und Programm-Version, Plattform)
- die offenen Transaktionen des aktiven Clients auslesen (z.B. für Kassenabschluß)
- die SN des aktuellen und/oder aller aktuell angemeldeten Clients ermitteln
- den Tar-Export aufrufen
- die kompletten Bondaten (Signatur, SN TSE, SN Kasse, Uhrzeit etc.) prüfen
- einen QR-Code aus den Bondaten erzeugen
- einen Client registrieren und löschen (wird nur intern benötigt)
- den Serverprozess sauber beenden

Aus dem vom Client gesendeten Request werden vorab abschließende CR, LF und Leerzeichen ausgefiltert.

Die Clients müssen beim Verbinden einen Aufrufcode (ein Groß-Buchstabe), gefolgt von einem Bindestrich und der GUID, sowie optional danach die evtl. Nutzdaten als Request senden. Der Response erfolgt immer mit "@" als Trennzeichen. Zwischen der GUID und den Nutzdaten darf im Request nun generell auch ein "@" (statt wie bisher ein "-") stehen (s. z.B. Code "L").
Folgende Aufrufcodes sind aktuell definiert:

A startet eine Transaktion
- es werden die Transaktions-Nr und die TSE-Zeit zurückgeliefert
- das Zeitformat kann über den Konfigeintrag "timeformat" eingestellt werden
- Response: "<TransNr>@<StartTime>"

B macht ein Transaktions-Update
- nach der GUID und einem Bindestrich müssen die Transaktions-Nr und die Nutzdaten angegeben werden
- die Client-ID muß identisch zum Aufruf "A" sein, es werden die Transaktions-Nr, die TSE-Zeit, der Signaturzähler und die Signatur selbst zurückgeliefert
- das Signatur-Format kann über den Konfigeintrag "signatur" eingestellt werden
- das Zeitformat kann über den Konfigeintrag "timeformat" eingestellt werden
- je nach Einstellung des Konfigeintrages "checkprocdata" werden die übergebenen Nutzdaten vorher analysiert und ggf. auf Syntax und Semantik geprüft
- die Werte für ProcessType und ProcessData sind mit einem "@" zu trennen
- bei Zahlenwerten ist als Dezimaltrennzeichen der Punkt zu benutzen
- mehrere Positionszeilen sind mit "\r" (CR bzw. 0x0D) zu trennen
- Beispiele für Requests:
  B-<GUID>-123@Bestellung-V1@1;"Mineralwasser";3.90
  B-<GUID>@456@Bestellung-V1@2;"Radeberger Pils";4.85
- Response: "<TransNr>@<UpdTime>@<SigNr>@<Signatur>"

C beendet eine Transaktion
- nach der GUID und einem Bindestrich müssen die Transaktions-Nr und die Nutzdaten angegeben werden
- die Client-ID muß identisch zum Aufruf "A" sein
- es werden die Transaktions-Nr, die TSE-Zeit, der Signaturzähler und die Signatur selbst zurückgeliefert
- das Signatur-Format kann über den Konfigeintrag "signatur" eingestellt werden
- das Zeitformat kann über den Konfigeintrag "timeformat" eingestellt werden
- je nach Einstellung des Konfigeintrages "checkprocdata" werden die übergebenen Nutzdaten vorher analysiert und ggf. auf Syntax und Semantik geprüft
- die Werte für ProcessType und ProcessData sind mit einem "@" zu trennen
- bei Zahlenwerten ist als Dezimaltrennzeichen der Punkt zu benutzen

- mehrere Positionszeilen sind mit "\r" (0x0D) zu trennen
- Beispiele für Requests:
  C-<GUID>-9@Kassenbeleg-V1@Beleg^120.00_0.00_0.00_0.00_0.00^120.00:Bar
  C-<GUID>@10@Kassenbeleg-V1@AVTransfer^0.00_0.00_0.00_0.00_55.55^55.55:Unbar
  C-<GUID>-456@Bestellung-V1@2;"Radeberger Pils";4.85
- Response: "<TransNr>@<EndTime>@<SigNr>@<Signatur>"

D testet, ob der TSE-Server erreichbar ist (keepalive)
- Response: "ping okay" oder "TSE busy"

E liefert den PublicKey der TSE als Base64-String zurück

F liefert die SerialNumber der TSE als Octet-String zurück

G liefert einige statische Daten als String zurück
- den TSE-Signatur-Algorithmus
- das intern verwendete TimeFormat der TSE
- die SDK-Version (direkt aus der Worm-Library)
- die Programmversion der TSE-Middleware (direkt aus der Software)
- das System (Windows, Linux, Arm), dabei handelt es sich um den einkompilierten Wert der Software, nicht um das aktuelle Betriebssystem
- Response: "<SigAlgo>@<TimeFormat>@5.8.1-offline@2.0.2@WINDOWS"

H liefert einige aktuelle TSE-Infos zurück
- den Zeitpunkt des Zertifikatsendes der TSE
- den Stand des Signaturzählers
- die Anzahl freier Blöcke der TSE
- die Zeit des nächsten Selbsttests
- den Zeitpunkt des Lizenzendes der TSE-Middleware
- die SN des aktuellen Clients anhand der Konfigdatei
- die Zeit und das Ergebnis des letzten Tar-Exports
  bei Erfolg die Datenmenge in Blöcken, bei einem Fehler diesen als negativen Wert
- das Zeitformat kann über den Konfigeintrag "timeformat" eingestellt werden
- Response: "<ZertifikatsEnde>@<Signaturzähler>@<freie Blöcke>@<NextSelftest>
  @<LizenzEnde>@<aktuelle Client-SN>@<Zeit letzter Tar-Export>
  @<Ergebnis letzter Tar-Export>"

I liefert die Anzahl und die Nummern der offenen Transaktionen des aktuellen Clients zurück
- wenn es keine offenen Transaktionen gibt, wird nur "0@" zurückgegeben
- Response: "<Anzahl>@<TransNr1>;<TransNr2>"

J ruft den Tar-Export der TSE auf
- nach der GUID und einem Bindestrich muß angegeben werden, ob der Server nach dem Export beendet werden soll (1) oder nicht (0)
- nach dem Trennzeichen '@' muß die Exportdatei (optional mit Pfad) angegeben werden
  bei allen Plattformen ist einheitlich ein Schrägstrich ("/") als Pfadseparator zu verwenden
- die hier angegebene Exportdatei muß zwingend auf ".tar" enden und ist immer relativ zum Pfad in der Konfigdatei
- aus diesem Grund ist der Parameter "tarexportdir" in der Konfigdatei verpflichtend
- Request: J-<GUID>-0@subdir/export.tar

K liefert die Anzahl und die Seriennummern der aktuell angemeldeten Clients zurück
- zu beachten ist hierbei, daß Clients, die noch offene Transaktionen besitzen, von der TSE nicht wieder abgemeldet werden (auch nicht bei Beendigung des Serverprozesses)
- da immer mindestens der aktuelle Client angemeldet ist, sollte es normal nie zu einer Rückgabe von "0@" kommen
- Response: "<Anzahl>@<SN Client1>;<SN Client2>"

L prüft die übergebenen Bon-Daten auf Korrektheit
- es kann entweder ein QR-Code im PBM-ASCII-Format (P1) oder ein bereits decodierter QR-Code angeliefert werden
- Request: L-<GUID>@P1<CRLF>25<CRLF>25<CRLF>010011100....010010010
  Request: L-<GUID>@V0;<Kasse>;....;<TSE-PublicKey>
- Response: "<okay|not valid>@<SN TSE in Octet>@<SN Kasse>"

Z beendet den Serverprozess sauber
- Response: "shutdown initiated"

Wichtig!

Da es leider in der Firmware der TSE keine Möglichkeit gibt, auszulesen, welche Client-ID eine Transaktion geöffnet hat und die TSE aber ein "finishTransaction" nur von derselben Client-ID akzeptiert, sollte man sich unbedingt die offenen Transaktionen im ERP/Kasse genau merken. Optional kann man natürlich auch alle angemeldeten Clients "durchprobieren".

1. GUI-Version

Das Server-Programm wird entweder per Aufruf aus dem Startmenü oder per Autostart ausgeführt. Die GUI-Version sucht ihre Konfigurationsdatei per Default im Programmverzeichnis unter dem Namen "tse_admin.conf". Dies kann über einen Aufrufparameter übersteuert werden:

-c | --cfgfile <Dateiname> der Name (und optional der Pfad) der Konfigdatei

Weitere Kommandozeilenparameter sind aktuell nicht definiert. Die GUI öffnet standardmäßig (die Position ist konfigurierbar) in der rechten unteren Ecke des Desktops ein kleines Fenster, welches den Status der Initialisierung anzeigt. Sobald die Lizenz geprüft ist, wird ein Icon im SystemTray erstellt, welches beim Start rot dargestellt wird. Nach der Initialisierung wechselt dieses Icon auf grün und das Fenster wird nach weiteren 30 Sekunden ausgeblendet. Es kann jederzeit mit einem Rechtsklick auf das Icon wieder eingeblendet werden. Ebenso läßt sich darüber auch der Server beenden. Ein Linksklick wechselt die Anzeige des Serverprogramms direkt zwischen ein- und ausgeblendet. Das Server-Programm lauscht auf dem konfigurierten Port auf Zugriffe von Clients und dient damit gleichzeitig als Semaphore für den (synchronisierten) Zugriff mehrerer Kassen auf eine TSE, indem es die gesamte Kommunikation mit der TSE übernimmt.

Wichtig!

Das Server-Programm sollte nach Möglichkeit entweder über den Aufrufcode "Z" oder über die GUI beendet werden, damit es seine Aufräumarbeiten (Logfile schreiben, Clients abmelden) korrekt erledigen kann.

2. Konsolenversion (nur Linux)

Das Server-Programm sollte später dauerhaft als Daemon laufen. Man kann es aber ebenso, z.B. zum Testen, manuell in einer Konsole starten. Es muß in jedem Falle als root gestartet werden und wechselt anschließend (nach dem Öffnen des Ports) in den Kontext des konfigurierten TSE-Users. Zum Beenden des Programms kann man diesem ein SIGTERM oder SIGINT-Signal senden. Dazu benötigt man die Prozess-Nummer, die man einfach aus dem PID-File ermitteln kann (z.B. "**kill $(cat <pidfile>)"**). Wenn das Programm in der Konsole läuft, kann man zum Beenden auch einfach "**Ctrl-C**" drücken.

Die Konsolen-Version sucht ihre Konfigurationsdatei per Default im Programmverzeichnis unter dem Namen "tse_server.conf". Das Server-Programm lauscht auf dem konfigurierten Port auf Zugriffe von Clients (beliebig viele, max. fünf zur exakt gleichen Zeit) und dient damit gleichzeitig als Semaphore für den (synchronisierten) Zugriff mehrerer Kassen auf eine TSE, indem es die gesamte Kommunikation mit der TSE übernimmt.

Sollten Sie zusätzliche Hilfe bei der Integration der Software in Ihren Server benötigen (Mount-Script, Start-/Stop-Scripte, Monitoring etc.), kontaktieren Sie uns bitte. Wir können Ihnen eine Vielzahl an Scripten, bis hin zu fertigen OpenVPN- und DynDNS-Lösungen über unsere eigenen Server anbieten.

Usage: **tse_server [std_opt]**

Die nachfolgenden Standardoptionen (std_opt) können verwendet werden:

-c | --cfgfile <Dateiname>	der Name (und optional der Pfad) der Konfigdatei
-d | --loglevel { 0 | 1 | 2 | 3 | 4 }	der Loglevel (Defaultwert s. Konfigdatei)
-f | --logfile <Dateiname>	der Pfad und Name des Log-Files (Defaultwert s. Konfigdatei)
--logappend { 1 | 0 }	soll das Log-File bei jedem Start überschrieben oder erweitert werden (Defaultwert s. Konfigdatei)
-l | --lizfile <Dateiname>	das Lizenz-File (Defaultwert s. Konfigdatei)
-m | --mountpoint <Pfad>	der Mountpoint der TSE (Defaultwert s. Konfigdatei)
-v | --version	die Ausgabe von Programmversion und Copyright-Info (schließt alle anderen Optionen aus)
-h | --help	diese Hilfeanzeige (schließt alle anderen Optionen aus)
-D | --daemon	das Programm wird im Hintergrund als Daemon ausgeführt

Wichtig!

Das Server-Programm sollte nach Möglichkeit entweder über den Aufrufcode "Z" oder über das Signal "SIGTERM" beendet werden, damit es seine Aufräumarbeiten (Logfile schreiben, Clients abmelden) korrekt erledigen kann.

VIII. Konfiguration

Die meisten Parameter für den normalen Betrieb werden aus der Konfigdatei gelesen. Diese Konfigdatei wird standardmäßig im aktuellen Programmverzeichnis gesucht. Der Vorgabename lautet

- tse_admin.conf für das Adminprogramm
- tse_server.conf für das Serverprogramm

Per Kommandozeilenparameter (-c oder --cfgfile) kann auch ein anderer beliebiger Name und Pfad für die Konfigdatei angegeben werden. Die Konfigdatei ist nach dem üblichen Schema für Konfigurationsdateien aufgebaut. Es gibt Sektionen (in eckigen Klammern) und innerhalb dieser Sektionen Key-Value-Einträge (key = value). Dabei sind beliebig viele Leerzeichen vor und hinter dem Key, dem Gleichheitszeichen und dem Value erlaubt. Die Datei ist in drei Sektionen gegliedert:

- "global" (hier stehen allgemeine Werte)
- "log" (die Angaben zu Logfile, Loglevel etc.)
- "clients" (nur in der Server-Konfig, die Angaben zu den Kassen, die sich verbinden dürfen)

1. Einträge der Sektion "global":

1.1 Linux

- mountpoint
  der Mountpoint der TSE
  mandatory, eine absolute Pfadangabe
- lizfile
  das zugehörige Lizenzfile
  mandatory, entweder relativ zum Programmdirectory oder eine absolute Pfadangabe
- adminpin
  die Admin-PIN der TSE
  mandatory, muß unbedingt korrekt gesetzt sein
- timeofset
  der Zeitversatz der TSE gegenüber der Systemuhr
  optional, nur bei Entwickler-TSE möglich
  mit diesem Eintrag kann ein Zeitversatz angegeben werden, um den die interne TSE-Uhr gegenüber der Systemzeit nachgehen soll
  der Wert wird in Sekunden angegeben, also 1 Tag = 86400, 1 Jahr = 31536000
- port
  der Port, auf dem der Server "lauscht"
  mandatory, nur beim Server notwendig
  da der Server den Port als root öffnet, können hier auch beliebige freie Ports unterhalb von 1024 genutzt werden
- selftest
  die Uhrzeit, zu welcher der Selbsttest täglich (bevorzugt) ausgeführt werden soll
  optional, nur beim Server möglich
  spätestens aller 25 Stunden muß ein Selbsttest der TSE angestoßen werden, ansonsten verweigert die TSE den Dienst, da der Selbsttest bis zu 60 Sekunden dauert, kann man diesen z.B. auf die Nachtstunden verlegen, so vermeidet man eine Störung oder Verzögerung innerhalb der Kernarbeitszeit
  sollte der nächste notwendige Selbsttest der TSE vor dieser Uhrzeit liegen, wird direkt beim Start des Programms ein einmaliger Selbsttest ausgeführt
- daemon
  legt fest, ob sich das Programm von der Konsole abkoppelt
  optional, nur beim Server möglich
  0 = default, das Programm bleibt im Vordergrund und protokolliert in die Konsole
  1 = das Programm wird "deamonisiert", d.h. es wird von der Konsole abgekoppelt und läuft im Hintergrund weiter, die Konsole kann geschlossen werden

- pidfile
  das File, welches die PID des Prozesses enthält
  mandatory, nur beim Server notwendig
  relativ zum Programmdirectory oder eine absolute Pfadangabe
  das Serverprogramm erzeugt diese Datei zwar beim Start, da sie jedoch erst nach dem Switch in den User-Kontext beschrieben wird, muß sie auch für den konfigurierten User schreib- und löschbar sein
  Wichtig! das PID-File wird unabhängig von dem Wert im Eintrag "daemon" erzeugt
- user
  der User, unter dem der TSE-Server laufen soll
  mandatory, nur beim Server notwendig
  es kann ein Name oder eine UID angegeben werden (Nutzer mit uid=0 müssen per Name spezifiziert werden), dieser User sollte mit dem Mount-Befehl korrespondieren
- group
  die Gruppe, unter der der TSE-Server laufen soll
  optional, nur beim Server möglich
  es kann ein Name oder eine gid angegeben werden (Gruppen mit gid=0 müssen per Name spezifiziert werden), wenn die Gruppenangabe fehlt, wird die primäre Gruppe des Users genutzt
- tarexportdir
  das Basis-Verzeichnis für den Tar-Export, der über den Server ausgeführt wird
  für das Kommando "J" mandatory, nur beim Server notwendig
  hier muß unbedingt ein gültiges Verzeichnis (relativ zum Programmdirectory oder ein absoluter Pfad) angegeben werden
  aus Sicherheitsgründen ist es dem tse_server-Prozess nicht erlaubt, Daten oberhalb dieses Verzeichnisses abzulegen, somit bestimmt diese Angabe das Wurzelverzeichnis für den Tar-Export
- checkprocdata
  schaltet eine zusätzliche Prüfung der übergebenen Nutzdaten vor der Weiterleitung an die TSE ein
  optional, nur beim Server möglich
  0 = default, keine Prüfung
  1 = das Feld "processtype" wird gegen die möglichen Typen laut DSFiNV-K geprüft
  2 = wie 1, zusätzlich wird die Syntax im Feld "processdata" laut DSFiNV-K geprüft (z.B. die erlaubten Transaktionstypen, die Zahlenformate, die Trennzeichen, die Reihenfolge der Zahlungen)
  3 = wie 2, zusätzlich werden bei "processtype=Kassenbeleg-V1" die Bruttosummen-Felder gegen die Felder mit den Zahlungen auf Gleichheit geprüft (außer bei Zahlung in Fremdwährung)

Die meisten Werte können auch per Kommandozeilenparameter übersteuert werden.

1.2 Windows

- mountpoint
  der Mountpoint der TSE
  mandatory, nur der Laufwerksbuchstabe gefolgt von einem Doppelpunkt
- init_timeout
  die Zeit in Sekunden, den das Serverprogramm auf die Bereitschaft des Mountpoint wartet, bevor es zu einem Fehler kommt
  optional, nur beim Server möglich
  damit kann der Server auch problemlos gestartet werden, wenn ein USB-Port oder -Hub erst verzögert aktiviert wird (z.B. im "Autostart")
- lizfile
  das zugehörige Lizenzfile
  mandatory, entweder relativ zum Programmdirectory oder eine absolute Pfadangabe
- adminpin
  die Admin-PIN der TSE
  mandatory, muß unbedingt korrekt gesetzt sein
- timeofset
  der Zeitversatz der TSE gegenüber der Systemuhr
  optional, nur bei Entwickler-TSE möglich
  mit diesem Eintrag kann ein Zeitversatz angegeben werden, um den die interne TSE-Uhr gegenüber der Systemzeit nachgehen soll
  der Wert wird in Sekunden angegeben, also 1 Tag = 86400, 1 Jahr = 31536000
- port
  der Port, auf dem der Server "lauscht"
  mandatory, nur beim Server notwendig
- selftest
  die Uhrzeit, zu welcher der Selbsttest täglich (bevorzugt) ausgeführt werden soll
  optional, nur beim Server möglich
  spätestens aller 25 Stunden muß ein Selbsttest der TSE angestoßen werden, ansonsten verweigert die TSE den Dienst, da der Selbsttest bis zu 60 Sekunden dauert, kann man diesen z.B. auf die Nachtstunden verlegen, so vermeidet man eine Störung oder Verzögerung innerhalb der Kernarbeitszeit
  sollte der nächste notwendige Selbsttest der TSE vor dieser Uhrzeit liegen, wird direkt beim Start des Programms ein einmaliger Selbsttest ausgeführt

- windowposx
  bestimmt die x-Koordinate des Serverfensters
  optional, nur beim Server möglich
  0 = default, das Fenster wird ganz recht unten angezeigt
  -1 = das Fenster wird horizontal zentriert auf dem Bildschirm angezeigt
  >0 = Anzahl Pixel vom linken Bildschirmrand bis zur linken Fensterkante
  <0 = Anzahl Pixel vom rechten Bildschirmrand bis zur rechten Fensterkante
- windowposy
  bestimmt die y-Koordinate des Serverfensters
  optional, nur beim Server möglich
  0 = default, das Fenster wird ganz recht unten angezeigt
  -1 = das Fenster wird vertikal zentriert auf dem Bildschirm angezeigt
  >0 = Anzahl Pixel vom oberen Bildschirmrand bis zur oberen Fensterkante
  <0 = Anzahl Pixel vom unteren Bildschirmrand bis zur unteren Fensterkante
- tarexportdir
  das Basis-Verzeichnis für den Tar-Export, der über den Server ausgeführt wird
  für das Kommando "J" mandatory, nur beim Server notwendig
  hier muß unbedingt ein gültiges Verzeichnis (relativ zum Programmdirectory oder ein absoluter Pfad) angegeben werden
  aus Sicherheitsgründen ist es dem tse_server-Prozess nicht erlaubt, Daten oberhalb dieses Verzeichnisses abzulegen, somit bestimmt diese Angabe das Wurzelverzeichnis für den Tar-Export
- checkprocdata
  schaltet eine zusätzliche Prüfung der übergebenen Nutzdaten vor der Weiterleitung an die TSE ein
  optional, nur beim Server möglich
  0 = default, keine Prüfung
  1 = das Feld "processtype" wird gegen die möglichen Typen laut DSFiNV-K geprüft
  2 = wie 1, zusätzlich wird die Syntax im Feld "processdata" laut DSFiNV-K geprüft (z.B. die erlaubten Transaktionstypen, die Zahlenformate, die Trennzeichen, die Reihenfolge der Zahlungen)
  3 = wie 2, zusätzlich werden bei "processtype=Kassenbeleg-V1" die Bruttosummen-Felder gegen die Felder mit den Zahlungen auf Gleichheit geprüft (außer bei Zahlung in Fremdwährung)

2. Einträge der Sektion "response":

- timeformat
  bestimmt das Format, in dem die Uhrzeiten im Response zurückgegeben werden
  optional, nur beim Server möglich
  0 = default, der Response wird als Unix-Timestamp (UTC) übermittelt
  1 = es wird die lokale Zeit im Format "dd.mm.yyyy hh:mm:ss +0100", also mit Angabe der Abweichung zu UTC, übermittelt, die Darstellung entspricht dabei der deutschen Locale, nicht aber der ISO-Norm
  2 = es wird die lokale Zeit im Format "yyyy-mm-ddThh:mm:ss+0100" nach ISO 8601-1:2019, also mit Angabe der Abweichung zu UTC, übermittelt
  3 = es wird die UTC-Zeit im Format "yyyy-mm-ddThh:mm:ssZ" nach ISO 8601-1:2019 übermittelt, das "Z" steht dabei für Zulu-Zeit (also UTC)
- signatur
  bestimmt die Kodierung der vom Befehl "C" zurückgelieferten Signatur
  optional, nur beim Server möglich
  0 = default, die Signatur wird ab v0.99 in Base64-Kodierung, vorher in Octet-Kodierung zurückgeliefert
  1 = die Signatur wird in jedem Fall in Octet-Kodierung zurückgeliefert
- appendchar
  bestimmt die Zeichen, die an jeden Response angehangen werden
  optional, nur beim Server möglich
  leer = default, der Response wird "as it is" gesendet
  LF = es wird an jeden Response ein LF ("\n") angehängt
  CR = es wird an jeden Response ein CR ("\r") angehängt
  CRLF = es wird an jeden Response ein CRLF ("\r\n") angehängt
  Wichtig! beim Request werden unabhängig von dieser Einstellung in jedem Fall alle abschließenden CR, LF und Leerzeichen entfernt

3. Einträge der Sektion "log" :

- logfile
  legt das File fest, wohin die Log-Daten ausgegeben werden
  optional
  muß bei Linux mit daemon=1 immer eine absolute Pfadangabe sein
  wenn die Angabe fehlt oder das File nicht schreibbar ist, erfolgt die Ausgabe entweder auf die Konsole (unter Linux bei daemon=0) oder ins "Nirwana"
- logappend
  bestimmt die Erzeugung des Logfiles
  optional
  0 = default, das Logfile wird bei jedem Start neu erstellt (vermeidet zu große Logdateien)
  1 = das Logfile wird jeweils fortgeschrieben (kann sehr groß werden)
- loglevel
  legt den Loglevel fest
  optional
  folgende Loglevel sind definiert:
  0 = default, kein Log (es werden nur Fehler ausgegeben)
  1 = produktiv (wenige Ausgaben, für den produktiven Einsatz empfohlen)
  2 = info (mehr informative Ausgaben)
  3 = notice (sehr viele detaillierte Ausgaben)
  4 = debug (nur zur Debug- und Entwicklerzwecken, es werden sehr viele Daten ausgegeben)
  Wichtig! da während des laufenden Betriebes eine Menge an Daten anfallen können, wählen Sie für den produktiven Einsatz unbedingt einen loglevel von 1 oder 0

Bei der Konsolenversion können die meisten Werte per Kommandozeilenparameter übersteuert werden.

4. Einträge der Sektion "clients":

In der Sektion "clients" werden die Kassen (Clients) eingetragen, die sich verbinden dürfen. Dabei ist der Key die eindeutige Client-ID, also die Seriennummer der Kasse, wie sie vom Kassenhersteller vergeben und von der TSE gefordert wird. Der Value (Wert) ist eine frei wählbare, aber ebenfalls (pro TSE) eindeutige GUID. Der Aufbau entspricht dem bekannten Format von GUID's (s. Glossar). Diese GUID dient ausschließlich der Kommunikation und Authentifizierung zwischen dem ERP/Kasse und dem Serverprogramm. Dadurch ist der Request (unabhängig von der Länge der Client-ID) immer gleich lang und die Sicherheit ist ebenfalls immer gleich hoch. Anschließend "übersetzt" das Serverprogramm diese GUID wieder in die Client-ID, welche dann an die TSE gesendet wird.
Beispiel:
Kasse-0815 = 3CCEB42E-99CD-49D6-8B8A-536B8C3BE1D2
hierbei ist "Kasse-0815" die SN der Kasse (Client-ID), welche auf dem Bon und in der TSE benutzt wird

Wichtig (nur Linux)!
Die Konfigdatei sollte nur für den User, der das Programm tse_admin ausführt, also z.B. root oder den TSE-User, les- und schreibbar sein, denn darin steht die PIN im Klartext.
chown tse:tse /opt/tse/tse.conf; chmod 0600 /opt/tse/tse.conf

IX. QR-Code und Bondruck

Aufgrund wiederholter Anfragen möchten wir Ihnen hier nochmals einige Infos zum Aufbau des QR-Codes, sowie zu den Angaben auf dem Bon geben.
Seit 2021 kann auf den Druck des "für jedermann ohne maschinelle Unterstützung lesbaren" Teils des Bondruckes verzichtet werden, wenn ein geeigneter QR-Code abgedruckt wird, der alle notwendigen Informationen enthält.
Es ist zu beachten, daß ohne den Druck des QR-Codes keine Möglichkeit besteht, die Signatur auf dem Bon zu prüfen, weshalb sich eine Kassenachschau ggf. aufwendiger und wohl mindestens auf den Tar-Export erstrecken dürfte.

1. Aufbau des QR-Code

V0 (fixe Kennung)
<SNr der Kasse> also die ClientID
<processtype> z.B. "Kassenbeleg-V1"
<processdata> z.B. "Beleg^20.00_0.00....^20.00:Bar"
<Transaktions-Nr>
<Signaturzähler>
<Startzeit> z.B. als JJJJ-MM-DDThh:mm:ss.cccZ oder JJJJ-MM-DDThh:mm:ss+0x00
<Endzeit> z.B. als JJJJ-MM-DDThh:mm:ss.cccZ oder JJJJ-MM-DDThh:mm:ss+0x00
<TSE-Signatur-Algorithmus> z.B. "ecdsa-plain-SHA384"
<TSE-Timeformat> z.B. "unixTime"
<BitString der Signatur als Base64> (i.d.R. 128 Zeichen)
<BitString des PublicKey als Base64> (i.d.R. 132 Zeichen)

Alle Bestandteile sind mit Semikolon zu trennen. Es sind keine Zeilenumbrüche einzufügen.
Beim Zeitformat ist zu beachten, daß es sich entweder direkt um UTC-Zeit handelt, oder daß sich die lokale Zeit eindeutig in UTC umrechnen läßt (z.B. durch Angabe der Zeitverschiebung).
Der QR-Code muß in einer maschinenlesbaren Größe gedruckt werden, aktuell geht man dabei von ca. 20 bis 25 mm als Minimum aus. Eine Vorgabe der Fehlertoleranz gibt es nicht.

2. Notwendige Angaben auf dem Bon

Wenn kein QR-Code gedruckt wird, müssen die folgenden Angaben zwingend auf dem Bon gedruckt werden. Die Reihenfolge und Größe des Drucks ist dabei frei wählbar, der Text muß aber "ohne maschinelle Hilfsmittel lesbar" sein.

<SNr der Kasse> also die ClientID
<SNr der TSE als Octet-String> (somit genau 64 Hex-Zeichen)
<processtype> z.B. "Kassenbeleg-V1"
<processdata> z.B. "Beleg^20.00_0.00....^20.00:Bar"
<Transaktions-Nr>
<Signaturzähler>
<Startzeit> z.B. als JJJJ-MM-DDThh:mm:ss.cccZ oder JJJJ-MM-DDThh:mm:ss+0x00
<Endzeit> z.B. als JJJJ-MM-DDThh:mm:ss.cccZ oder JJJJ-MM-DDThh:mm:ss+0x00
<TSE-Signatur-Algorithmus> z.B. "ecdsa-plain-SHA384"
<TSE-Timeformat> z.B. "unixTime"
<BitString der Signatur als Base64> (i.d.R. 128 Zeichen)

X. Glossar

TSE	Technische Sicherheitseinrichtung, als Technische Sicherheitseinrichtung wird ein Sicherheitsmodul in elektronischen Registrierkassen bezeichnet, das der lückenlosen und unveränderbaren Aufzeichnung aller Kassenvorgänge dient.
KassenSichV	deutsche Kassensicherungsverordnung, welche ab 1. Januar 2020 die vollständige, unveränderte und manipulationssichere Speicherung von Geschäftsvorfällen und einiger weiterer Vorgänge verlangt
tse_server	das Server-Programm der TSE-Middleware
tse_admin	das Admin-Programm der TSE-Middleware
tse_server.conf	die Konfigdatei für das Server-Programm der TSE-Middleware
tse_admin.conf	die Konfigdatei für das Admin-Programm der TSE-Middleware
Client-ID	die Identifikation der Kasse gegenüber der TSE (die Seriennummer der Kasse) darf nur die folgenden Zeichen enthalten: A-Z, a-z, 0-9, Bindestrich und darf maximal 30 Zeichen lang sein wird vom tse_server-Programm anhand der Konfigdatei aus der GUID ermittelt
AdminPIN	die 5-stellige PIN des TSE-Administrators, wird für verschiedene Befehle benötigt (default=12345)
TimeAdminPIN	die 5-stellige PIN des TSE-Time-Administrators, wird für die TSE-Middleware nicht benötigt (default=12345)
PUK	die 6-stellige Notfall-PIN der TSE, hierüber kann man die Admin- und die TimeAdmin-PIN neu vergeben und so geblockte User wieder entsperren, nach dreimaliger Falscheingabe ist die TSE für immer gesperrt (default=123456)
TSE-Selbstest	eine interne Routine der TSE, die von deren Firmware ausgeführt wird, damit soll die korrekte Funktion der TSE sichergestellt werden
TSE-TarExport	ein Export aller TSE-Daten in einem genau definierten Format, es wird dabei von der TSE pro Message eine Datei erzeugt, anschließend werden diese Dateien in einem kompakten File abgelegt, der Tar-Export ist z.B. für Prüfvorgänge (Finanzamt) oder vor dem Löschen der TSE-Daten notwendig, zusätzlich sollte der Tar-Export im Rahmen einer Datensicherung zyklisch ausgeführt werden
FactoryReset	eine Möglichkeit, die TSE komplett auf Werkseinstellungen zurückzusetzen, geht nur bei sog. Entwickler-TSE's mit einer Developement-Firmware
Signatur	mit Hilfe eines kryptographischen Vorgangs wird eine Nachricht "signiert", später kann anhand dieser Signatur zweifelsfrei erkannt werden, ob die Nachricht nachträglich verändert wurde

Transaktion	eine Verarbeitung der TSE, bei der eine vom Client gesendete Zeichenfolge in der TSE gespeichert und signiert wird, diese Zeichenfolge ist damit nachträglich nicht mehr veränderbar, ohne daß die Signatur ungültig wird
ProcessType	beim Abschluß der Transaktion werden der TSE die zugehörigen (Nutz-)Daten übergeben, dabei gibt es z.Zt. drei Typen: "Bestellung", "Kassenbeleg" und "Sonstiger Vorgang"
ProcessData	beim Abschluß der Transaktion werden der TSE die zugehörigen (Nutz-)Daten übergeben, dabei gibt es (abhängig vom ProcessType) verschiedene Formate der Nutzdaten: bei "Bestellung" müssen Menge, Artikel und Preis, bei "Kassenbeleg" Transaktionstyp, Summen und Zahlungen übergeben werden, während beim "Sonstigen Vorgang" die Daten frei wählbar sind
GUI	Graphical-User-Interface, ein Programm mit einer "Fensterausgabe"
SDK	Software-Developement-Kit, ein Programmpaket, welches (meist vom Hersteller) mitgeliefert wird und in die Anwendungssoftware integriert ("einkompiliert") werden muß, diese SDK werden i.d.R. nur in ausgewählten Programmiersprachen bereitgestellt (bei der Swissbit-TSE sind dies C/C# und Java)
Qt	ein Anwendungsframework und GUI-Toolkit zur plattformübergreifenden Entwicklung von Programmen und grafischen Benutzeroberflächen
ERP	Programm für "Enterprise-Ressource-Planning", also eine "Warenwirtschaft" im weiteren Sinne
QR-Code	ein QR-Code (2D-Barcode) besteht aus einer quadratischen Matrix von schwarzen und weißen Quadraten, die die kodierten Daten binär darstellen. Eine spezielle Markierung in drei der vier Ecken des Quadrats gibt die Orientierung vor. Die Daten im QR-Code sind durch einen fehlerkorrigierenden Code erweitert. Dadurch wird der Verlust von bis zu 30% des Codes toleriert
QR-Code-Test	Website zum Testen des QR-Codes oder der Bondaten im Klartext, z.B. https://kassen-qr-code-test.de
PBM-Format	PBM (Portable Bitmap) ist ein sehr einfaches und unkomprimiertes Format zur Speicherung einer zweifarbigen Bilddatei, hierbei wird jedes Pixel des Bildes in einem Byte (P1 = ACSII-PBM) oder einem Bit (P4 = Binär-PBM) dargestellt, wobei das Bild zeilenweise von oben links durchlaufen wird
UTC	Universal Coordinated Time, früher auch GMT (Greenwich Mean Time) genannt, hierbei handelt es sich um die sogenannte Normalzeit, d.h. eine weltweit eindeutige Uhrzeit, die auf dem Null-Meridian von Greenwich basiert

LocalTime	die lokale Uhrzeit, die sich aus der Verschiebung der aktuellen Region zur UTC ergibt, Deutschland nutzt eine Verschiebung von einer Stunde (UTC +1), diese Zone wird auch als MEZ (Mitteleuropäische Zeit) oder CET (Central European Time) bezeichnet, während der Sommerzeit erhöht sich dieser Versatz auf zwei Stunden (UTC +1), die Zeitzone heißt dann CEST (Central European Summer-Time) oder MESZ (Mitteleuropäische Sommerzeit)
ISO 8601	diese Norm ist ein internationaler Standard, der Empfehlungen über numerische Datumsformate und Zeitangaben enthält. Der (eingedeutschte) Titel der Norm lautet: "Datenelemente und Austauschformate - Informationsaustausch - Darstellung von Datum und Uhrzeit"
Unix-Timestamp	eine in der Unix-Welt gebräuchliche Darstellung für die Uhrzeit, hierbei werden die Sekunden seit dem 01.01.1970 in einer Zahl angegeben, dieser Wert ist immer in UTC (Universal Coordinated Time)
GeneralizedTime	eine in ISO8601 definierte Darstellung der Uhrzeit, dabei wird folgender String genutzt: "YYYYMMDDHHMMSS[.fff](Z|[+/-]HHMM)" vielfach wird auch die folgende Darstellung genutzt: "YYYY-MM-DDTHH:MM:SS[.fff](Z|[+/-]HHMM)", an dem "T" in der Mitte erkennt man diese Darstellung recht gut
UTC-Time	eine in ISO8601 definierte Darstellung der Uhrzeit, dabei wird folgender String genutzt: "YYMMDDHHMM[SS](Z|[+/-]HHMM)", hierbei wird nur eine zweistellige Jahreszahl genutzt, es sind keine Sekundenbruchteile erlaubt
Sommerzeit	während des Winterhalbjahres (Standard) beträgt die Differenz der lokalen Zeit in Deutschland genau plus eine Stunde zur UTC-Zeit zur Sommerzeit, wenn also die Uhren vorgestellt sind und wir uns in der MESZ (Mitteleuropäische Sommerzeit) befinden, wächst diese Differenz auf plus zwei Stunden an die Winterzeit wird auch als CET (CentralEuropeanTime) und die Sommerzeit als CEST (CentralEuropeanSummerTime) bezeichnet
SHA-2 / SHA-256	Hash oder Prüfwert, beliebig lange Daten erzeugen immer die gleiche, kompakte Größe des Hashs, die Änderung an nur einem Bit hingegen führt dazu, dass ein komplett anderer Hashwert entsteht
Octet	besondere Darstellung von (nichtdruckbaren) ASCII-Zeichen, hierbei wird jedes Zeichen im Bereich von 0...255 durch seinen Hexcode dargestellt, es ergibt sich damit eine (durch zwei teilbare) Folge von Hex-Werten z.B. "B1A2\n" = "423141320A"

Base64	ein Verfahren zur Kodierung von 8-Bit-Binärdaten in eine Zeichenfolge, die nur aus lesbaren, Codepage-unabhängigen ASCII-Zeichen besteht, es wird z.B. für Mail-Anhänge benutzt, dabei werden aus jeweils drei Zeichen Input vier Zeichen Output erzeugt, weshalb die kodierten Daten ca. 33% mehr Platz beanspruchen
GUID	https://de.wikipedia.org/wiki/Globally_Unique_Identifier, eine 128 Bit (also 36 Zeichen) lange Zeichenfolge, die (praktisch) weltweit einmalig ist, die Darstellung erfolgt i.d.R. im Format XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX die GUID wird nur für die Kommunikation zwischen Kasse/ERP und Middleware genutzt, von der Middleware wird später die "übersetzte" Client-ID an die TSE gesendet
CR / LF	"CR" = CarriageReturn (0x0D) bewegt den Cursor nach links an den Zeilenanfang, während "LF" = LineFeed (0x0A) einen Zeilenvorschuib ausführt, ohne die horizontale Cursorposition zu ändern. In aktuellen Betriebssystemen kennzeichnen diese Zeichen einen Zeilenumbruch, werden aber leider unterschiedlich verwendet. Während Unix/Linux ein "LF" und Mac-OS ein "CR" nutzt, wird bei Windows "CRLF", also zwei Zeichen, benutzt.
Daemon	ein bekannter Begriff unter Linux, mit dem Dienste bezeichnet werden, die im Hintergrund, losgelöst von der Konsole, arbeiten
PID-File	eine Datei, die bei Linux von Daemons erzeugt wird und die Prozessnummer des Dienstes enthält, sie wird u.a. zum Beenden des Dienstes verwendet
Firmware	die TSE besitzt eine eigene Software, die für die "Intelligenz" der TSE verantwortlich ist, diese ist fest in der TSE verankert; mit unserer TSE-Middleware ist ein Update dieser Firmware ohne Internetzugriff möglich

XI. ReleaseNotes

Version 2.0.8 (28.06.2024)

- der Sicherheitspuffer für die Prüfung auf Ablauf des Zertifikates wurde auf zwei Tage verkürzt
- beim Server-Kommando "H" werden nun zusätzlich auch die Zeit und das Ergebnis des letzten Tar-Exports zurückgegeben
- aus dem Server-Kommando "G" wurde die Architektur (32 oder 64 Bit) entfernt, da die Software nur noch für 64Bit angeboten wird
- neues Server-Kommando "M" zur Erstellung von QR-Code-Daten in Vorbereitung

Version 2.0.6 (02.06.2024)

- neues Server-Kommando "L" zur Prüfung der Signaturdaten von decodierten QR-Codes
- neue Funktionen zum Erstellen (Nachbilden) der Messages der TSE, sowie zum Parsen der Uhrzeit aus einem String
- im Request ist nun statt "-" optional auch ein "@" zwischen der GUID und den Nutzdaten erlaubt
- neue Sektion "response" in der tse_server.conf
- die Fehlernummern und Lizenztypen sind überarbeitet

Version 2.0.2 (16.04.2024)

- Versionsnummer auf 2.x erhöht
- Update auf Qt-Creator 13.x und Qt 6.5.x
- Konformität mit der neuesten Spezifikation "C++20"
- diese Version setzt für die Windows-GUI mindestens MS-Windows 10 64Bit und für Linux bzw. ARM mindestens 64Bit, libc6 2.30 und OpenSSL 3.1.x voraus
- alle Demo-Lizenzen, die in Verbindung mit einer produktiven TSE eingesetzt werden, unterliegen nun im tse_server der Runtimebegrenzung auf 30 Minuten, der Tar-Export kann hier ebenfalls nur im gültigen Lizenz-Zeitraum ausgeführt werden und liefert absichtlich verkürzte Daten
- beim Server-Kommando "H" werden nun zusätzlich auch das Lizenzende und die SN des aktuellen Clients zurückgeliefert
- neue Funktion zum Auslesen und zur verschlüsselten Übermittlung der SN von Fremd-TSE
- die Parameter "response" und "transnullok" aus der tse_server.conf sind entfernt, der Parameter "responsechar" ist in "appendchar" umbenannt

Version 1.1.10 (12.07.2024)

- dies ist die letzte Version für Linux-Plattformen mit OpenSSL 1.x
- dies ist auch die letzte Version für Windows 7/8/8.1 und/oder 32Bit-Unterstützung
- wir werden sie als Legacy-Version (ohne Support und Weiterentwicklung) solange bereitstellen, wie es notwendig und technisch möglich ist

Version 1.01.7 (15.07.2023)

- Korrektur der checkprocdata-Prüfung von negativen Cent-Beträgen in den Bruttosummen
- Erweiterung der checkprocdata-Prüfung auf ungültige Werte ("0.00:Bar" und "-0.00")

Version 1.01.2 (08.08.2022)

- das Server-Kommando "B" akzeptiert nun dieselben Input-Daten wie das Kommando "C", die Einstellung für "checkprocdata" gilt ebenfalls analog
- neues Server-Kommando "K" zur Ausgabe der angemeldeten Clients

Version 1.00.8 (02.05.2022)

- in der Konsolenversion wurde der Befehl "delete_data" korrigiert, es wird nun direkt vor dem Löschen ein Tar-Export ausgeführt (dieser wird vom SDK gefordert)

Version 1.00.6 (28.04.2022)

- Korrektur der Auswertung der Felder in processdata bei negativen Werten mit Nachkommastellen

Version 1.00.4 (18.04.2022)

- Korrektur von Transaktionen mit Typ "Beleg" und leeren Zahlungsdaten (z.B. Differenzbons)

Version 1.00.2 (30.03.2022)

- Versionsnummer auf 1.x erhöht
- neuer Konfigeintrag "checkprocdata" im tse_server, damit können die via Server-Kommando "C" übergebenen Werte für processtype und processdata vor der Übermittlung an die TSE mehrstufig geprüft werden, bei Fehlern wird die Transaktion nicht beendet
- das Server-Kommando "C" akzeptiert nun im Input auch CR (0x0D), welches z.B. beim processtype "Bestellung" Verwendung findet
- die Fehlernummern in den Linux-Dateien sind überarbeitet
- erweitertes Code-Review mit cLang und cTidy

Version 0.99.5 (15.03.2022)

- beim Server-Kommando "G" wird neben der Programm- nun auch die SDK-Version zurückgeliefert
- neuer Konfigeintrag "timeformat" im tse_server, damit kann das Format der Rückgabe der TSE-Zeiten gesteuert werden, neben UTC ist hier auch die Ausgabe von lokaler Zeit möglich
- neue Defines für die ARM-Plattform zur Prüfung der Lizenz
- Status-Flag (Open/Close) bei der Verarbeitung der Tar-Dateien

Version 0.99.3 (10.02.2022)

- beim Server-Kommando "G" wird nun auch die Programmversion und die Systemarchitektur (Linux/Windows/Arm, 32/64Bit) zurückgeliefert, hierbei handelt es sich um einkompilierte Werte
- Anpassung am Logging beim Tar-Export

Version 0.99.1 (05.02.2022)

- Update auf Qt-Creator 6.0.2
- die Signatur wird nun per default in Base64-Kodierung zurückgeliefert, über den Konfigeintrag "signatur=1" kann die bisherige Octet-Kodierung erzwungen werden
- neuer Konfigeintrag "tarexportdir" im tse_server, der Tar-Export via Serverkommando "J" schreibt nur noch in Verzeichnisse unterhalb dieses Konfigeintrages
- Logging und Verzeichnisabfrage beim Tar-Export unter Windows verbessert
- neben CR/LF werden nun auch abschließende Leerzeichen aus dem Request entfernt
- mehrere Konfigeinträge korrigiert
- alle Inputs der Server-Kommandos werden via RegExp geprüft
- der TSE-Mountpoint wird nun per RegExp ausgewertet
- Dokumentation überarbeitet

Version 0.98.3 (20.12.2021)

- fataler Fehler im Tar-Export unter Windows beseitigt

Version 0.98.2 (28.09.2021)

- Aktualisierung auf Qt-Creator 5.03
- Makefile bei ARM32 geändert (-Wno-psabi)
- gemeinsame Klasse für die beiden Komponenten tse_server und tse_admin
- Optimierungen und Bereinigungen am Code für die Vereinheitlichung (Templates)
- Meldungen erweitert und verbessert
- Konfigfile kann auch bei der GUI über Parameter geändert werden
- der TSE-Infoblock wird vor der Ausgabe nochmals aktualisiert
- tse_server und tse_admin erzeugen ein Locking, welches sicherstellt, daß nur eine Instanz pro TSE läuft
- alle Fehlernummern überarbeitet
- der Mountpoint wird normiert (ohne Slash am Ende)

Version 0.97.2 (03.09.2021)

- komplette Überarbeitung der Anfangsinitialisierung im tse_admin
- Höhe des Admin-Fensters geändert, Anordnung der Buttons optimiert
- FactoryReset ist nun eine MultiThreading-Aktion
- Integration des neuen SwissBit-SDK 5.8.1
- Aktualisierung der Includes für openssl (1.1.1j)
- CommandLineParser in GUI integriert
- direktes Firmware-Update im tse_admin ohne Internet möglich
- Überarbeitung und Vereinheitlichung von internen Funktionen
- neuer Funktionsaufruf "J" im tse_server für den Tar-Export
- Dokumentation überarbeitet

Version 0.96.0 (19.08.2021)

- bei Auswahl einer Aktion werden die Edit-Felder focussiert
- das Fenster von tse_server wird nach der Anzeige von Meldungen nun je nach Situation wieder ausgeblendet
- Verarbeitung von Serveranfragen während des Busy-Zustandes
- Korrektur des Konfigeintrages "logappend", neuer Konfigeintrag "init_timeout" bei tse_server
- neue TSE-Stati UNAUTH und BLOCK hinzugefügt
- factory_reset, unblock_admin, unblock_time_admin, change_puk ist nun auch bei geblockter TSE oder bei Status UNAUTH möglich
- im tse_admin wird nun vor jeder Aktion der TSE-Status geprüft
- die TSE-SN wird nur noch bei gültiger Lizenz im Logfile vermerkt
- Aktualisierung auf Qt-Creator 4.15.2 und auf die LTS-Qt-Version 5.15.1

Version 0.95.2 (15.04.2021)

- die Linux-Dateien im Git wurden aktualisiert
- Icons für "Admin entsperren" und "TimeAdmin entsperren" geändert
- Überarbeitung der Ausgabe der TSE-Messages und TSE-Info (u.a.Tausendertrennzeichen)
- nichtdruckbare Zeichen werden im Logfile maskiert
- die Programm-Version wird ins Konfigfile geschrieben, Leerzeile nach jeder Meldung eingefügt
- "Add-Client" und "Del-Client" unter Extras verlegt
- neue Funktion "CloseTrans" im tse_admin zum Schließen von offenen Transaktionen
- alle Keys und Sektionen im Konfigfile werden nun beim Start geprüft
- die LogLevel der Serveraktionen sind auf Prio 2, die der Konfigeinträge auf Prio 3, die vom Start/Stop-Timer auf Prio 4 geändert
- neue Option "responsechar" zur Erweiterung der Response um ein abschließendes CR/LF
- die Option "response=0" ist nun die Vorgabe (Fehlerunterdrückung mit "response=1")
- die Position des Serverfensters kann mit den zwei neuen Konfigeinträgen "windowposx" und "windowposy" frei bestimmt werden
- Aktualisierung auf Qt-Creator 4.14.2
- Dokumentation überarbeitet

Version 0.94.2 (26.03.2021)

- die TSE-Info wird auch vor der Initialisierung korrekt angezeigt
- die Client-ID wird nun besser geprüft, Unterstriche werden nicht mehr akzeptiert
- in der TSE-Info wird die letzte Transaktionsnummer angezeigt
- die Linux-Daten wurden aktualisiert, die Version wurde nach Debian, Raspi und Win32 portiert
- Infoausgaben verbessert, Ausgabebreite unter Linux verlängert
- der Loglevel des Timers wurde auf Prio 3 geändert
- die Textbreite im Server- und Adminfenster wird nun automatisch berechnet
- das Fenster vom tse_server wird automatisch wieder ausgeblendet
- abschließende CR und/oder LF im Datenstrom werden abgeschnitten
- Update Visual-Studio-Compiler 2019

Version 0.93.2 (07.03.2021)

- Linux-Dateien aktualisiert, Makefile hinzugefügt
- Response in Linux-Version eingebaut
- chown bei Linux-Logfile, reopen korrigiert
- MemoryLeak in cTseMsg::factory beseitigt
- emit, slots, signals in Qt-Schlüsselworte gewandelt
- SIGNAL- und SLOT-Makros aus connect entfernt

Version 0.93.0 (23.02.2021)

- eigenständige Dokumentation für die TSE-Middleware erstellt
- Include-Dateien ins Git aufgenommen
- neuer Konfigeintrag "transnullok" im tse_server als Workaround für Transaktionsnummer 0
- beim Setup (Initialisierung) wird sofort die erste Transaktion mit Nummer 0 erzeugt
- Projekt auf 32Bit erweitert, server.pro und admin.pro auf 32Bit angepaßt
- Optimierung der GUI (Aktivierung der Buttons)
- neuer Konfigeintrag "response=2" gibt auch den Fehler über den Socket zurück

Version 0.92.4 (03.02.2021)

- TSE-SN wird ins Logfile ausgegeben, Auswertung der Demo-SN ist korrigiert
- alle Info-Routinen sind nun auch ohne Lizenz möglich
- Runtime-Begrenzung in tse_server bei Demo-Lizenz auf 30 Minuten geändert
- interne Klassen umbenannt, Projekt umbenannt, Start/Stop Timer optimiert
- neue Icons für tse_server und tse_app
- Korrektur der Fenstergröße in tse_server
- neue Statuscodes für die TSE und die GUI

Version 0.91 (03.01.2021)

- erstes produktives Linux-Release und erstes Windows-Beta-Release
- Semaphore in Logging eingebaut
- neue File-Auswahl-Dialoge in Extras
- Funktionen in Extras-Panel fertiggestellt
- Initialisierung, FactoryReset, Pin/Puk in eigene Funktionen ausgelagert
- Start für uninitialisierte TSE ermöglicht
- Ausgabe von Lizenzdaten, PublicKey und Cert-Chain eingebaut
- Prüfung der TSE bei jedem Timer-Event
- Serverkommandos beachten ebenfalls den TSE-Status
- neue Buchstaben für die Funktionsaufrufe im tse_server
- Animation und SysTrayIcon im Server-Programm
- neues Loggingmodul und neue GUI für das tse_server Programm
- neues Projekt tse_server hinzugefügt
- Cygwin-Version wegen fehlender Funktionalität (Brainpool) endgültig entfernt

Version 0.90 (02.12.2020)

- Lizenzierung hinzugefügt
- Statuscodes in cServer, Timeranpassungen
- Multithreading beim Init von cServer
- Aufruf einiger TSE-Funktionen korrigiert
- Sicherheitsabfragen eingebaut

Version 0.88 (17.11.2020)

- TSE-Initialisierung in eigenen Thread ausgelagert

Version 0.87 (01.11.2020)

- erstes funktionsfähiges Linux-Release