ITG-RKSVLight

1. Einleitung

Im Zuge der Abgabenreform 2015 wurde den Gewerbetreibenden eine Registrierkassenpflicht aufgebürdet. Mit wenigen Ausnahmen muss ab dem 1.1.2016 jeder Unternehmer mit einem Barumsatz von > EUR 7.500,-- und einem Jahresumsatz von über EUR 15.000,-- eine beim Finanzamt angemeldete Registrierkasse führen. Nach dem Jahreswechsel 2016/17 erhöhen sich die Anforderungen an die Kasse auch noch um eine nach dem SigG zugelassene Signaturerstellungseinheit.

Um eine breite Palette an Unternehmen versorgen zu können, bereitet die ITG Consulting mehrere Lösungen vor. RKSVLight kann in den folgenden Formen betrieben werden:

- WCF-Service mit einer Datenbank
- COM Library ohne Datenbank
- Unmanaged Export ohne Datenbank
- Linux unter Mono (linux-runtime für .Net)

2. RKSVLight

2.1. Kurzbeschreibung

RKSVLight ist eine Bibliothek, die es dem Hersteller der Registrierkassensoftware über einfache Funktionsaufrufe ermöglicht die Vorgaben der Finanzbehörde einzuhalten. Hierzu wird im Normalbetrieb eine einzige Funktion mit von der Behörde vorgegebenen Werten aufgerufen und entsprechende Retourwerte an die Kassensoftware geliefert. Je nach Bedarf können Journaldaten in einer von uns geführten Datenbank abgelegt oder in der bestehenden Kasse gespeichert werden. Wird unsere DB verwendet, stellt die Bibliothek etwaige Exportfunktionen zur Verfügung. Andernfalls muss der Hersteller der Kassensoftware sich um die Aufbewahrung der Daten kümmern und im für die Finanzbehörde vorgesehenen Format exportieren können. Ein Export der Daten muss zumindest vierteljährlich stattfinden und auf einem externen Datenträger 7 Jahre lang aufbewahrt werden.

2.2. Interface

Bei der Verwendung der Library über COM (Component Object Model) kommt folgendes Interface zum Einsatz

- **string[] GetCardReaders();**

  Gibt ein Array aller gefundenen Reader, die eine unterstützte Sicherheitseinrichtung darstellen (Reader, die eine Globaltrust- oder a-trust-Karte beinhalten).

- **bool IsCardReaderPresent(string readername);**

  Gibt an, ob ein bestimmter Reader gefunden werden konnte. Es werden nur Reader gelistet, die eine unterstützte Sicherheitseinrichtung darstellen (Reader, die eine Globaltrust- oder a-trust-Karte beinhalten).

- **bool IsCardNumberPresent(string cardnumber);**

  Gibt an, ob eine bestimmte Karte gefunden werden konnte.

- **string GetCardInReader(string reader);**

  Gibt die Seriennummer der Karte in einem bestimmten Reader retour.

- bool **VerifyPin**(string **readerName**, string **pin**);

  es wird überprüft ob der angegebenen PIN zur Karte passt. Wenn Ok -> true, sonst -> false

- **bool GetCardCertificate(string readerName, ref string certificateBase64, ref string certificateChainBase64);**

  Gibt das public-Zertifikat und die gesamte Zertifikatskette retour.

- **bool RKSVJwsToImage(string jws, ref string qrCode, ref string ocraCode);**

  Konvertiert die jws-kompakt-Darstellung (das Endergebnis eines Signaturvorganges) in das base64-kodierte Image sowohl den QR-Codes als auch die OCR-A Darstellung.

- **bool SaveImageToFile(string base64image, string path, string mimetype);**

  Speichert die base64-Representation als Datei im angegebenen Pfad unter der Verwendung des angegebenen mime-types. Damit ist es möglich den OR-Code, als Image an den Bondrucker zu schicken.

  Mimetypes:
  - „mime/jpeg“
  - „mime/bmp“
  - „mime/png“
  - „mime/gif“

- **bool VerifyJWS(string jwsCompact, string certificateBase64);**

  Verifiziert, ob die in der jws-compact-Darstellung enthaltene Signatur gültig ist.

- **bool ExportJWS(string certificate, string certificateChain, string[] jwsCompactReceipts, ref string exportResult);**

  Exportiert Signaturen der Rechnungen in ein Json-Format unter Angabe des public-Zertifikates und der zugehörigen Zertifikat-Chain sowie eines Arrays mit den Signaturen der zu exportierenden Belege. Das Ergebnis dieser Funktion wird als String zurückgegeben und muss noch in ein File exportiert werden.

- bool **ExportJWSToFile**(string **certificate**, string **certificateChain**, string[] **jwsCompactReceipts**, string **Filepath**);

  Exportiert Signaturen der Rechnungen (wie die Funktion ExportJWS(), die Signaturen warden als Array übergeben) und speichert das Ergebnis gleich in eine Datei, die als letzter Parameter angegeben wird.

- bool **ExportJWSDelimitedStringToFile**(string certificate, string certificateChain, string delimitedReceipts, string Filepath);

  Exportiert Signaturen der Rechnungen. Die Signaturen werden durch “;” (Delimiter) getrennt als sein langer String übergeben. Das Ergbnis wird in eine Datei gespeichert, die als letzter Parameter angegeben wird.

  ACHTUNG: Manche Programmiersprachen haben bei langen Strings ein Limit von 32KB. In diesen Fällen wo dieses Limit zutrifft, sollte der Export auf mehrere Dateien gesplittet warden.

  Wichtiger HINWEIS:
  Seit August 2016 wird von der Finanzbehörde ein weiterer Export von Daten gefordert. Hierbei handelt es sich um den Export von Belegpositionsdaten. Es müssen die handelsübliche Bezeichnung und die Menge exportiert werden.

  Da die Struktur bzw. das Format des Exportfiles nicht vorgegeben wurde, empfehlen wir ein CSV-Format mit folgenden Datenfeldern:

  Kassen-ID; Belgenummer; BelegdatumUhrzeit; Artikelbezeichnung; Menge; Einheit

- **bool SignReceipt(**
  string sKassenID,
  string sReaderName,
  string sCardPIN,
  string sReNummer,
  string sReBelegDatumUhrzeit,
  string sReBetragSatzNormal,
  string sReBetragSatzErmaessigt1,
  string sReBetragSatzErmaessigt2,

```
    string sReBetragSatzBesonders,
    string sReBetragSatzNull,
    string sReStandUmsatzzaehler,
    string sReSigVorigerBeleg,
    string sAesKey,
    ref string sJWSCompactresult,
    ref string sQRImageBase64,
    ref string sOCRAImageBase64,
    ref string sAntwort);
```

Dies ist die eigentliche Funktion, die laufend verwendet wird. Sie verkettet die entsprechenden Felder, verschlüsselt den Umsatzzähler, signiert das ganze Paket und gibt die zur Ablage in der Datenbank vorgesehene jws-kompakt-Darstellung retour. Gleichzeitig liefert diese Funktion die zum Andruck auf der Rechnung vorgesehenen Bilder in base64-Darstellung. Entweder der QR-Code oder die OCR-A-Darstellung müssen auf der Rechnung angedruckt werden.

- **bool SignTrainReceipt(**

```
    string sKassenID,
    string sReaderName,
    string sCardPIN,
    string sReNummer,
    string sReBelegDatumUhrzeit,
    string sReBetragSatzNormal,
    string sReBetragSatzErmaessigt1,
    string sReBetragSatzErmaessigt2,
    string sReBetragSatzBesonders,
    string sReBetragSatzNull,
    string sReSigVorigerBeleg,
    ref string sJWSCompactresult,
    ref string sQRImageBase64,
    ref string sOCRAImageBase64,
    ref string sAntwort);
```

Diese Funktion dient Trainingszwecken und beeinflusst den Umsatzzähler nicht.

bool SignStornoReceipt(

```
    string sKassenID,
    string sReaderName,
    string sCardPIN,
    string sReNummer,
    string sReBelegDatumUhrzeit,
    string sReBetragSatzNormal,
    string sReBetragSatzErmaessigt1,
    string sReBetragSatzErmaessigt2,
    string sReBetragSatzBesonders,
    string sReBetragSatzNull,
    string sReSigVorigerBeleg,
    ref string sJWSCompactresult,
    ref string sQRImageBase64,
    ref string sOCRAImageBase64,
    ref string sAntwort);
```

Diese Funktion dient dem Stornieren von bereits signierten Rechnungen. Es dürfen nur reine Stornorechnungen signiert werden. Gemischte Rechnungen sind nicht erlaubt.

- **public static int GetCardReadersCount()**

  Mit dieser Funktion kann die Anzahl der gefundenen Reader erfragt werden.

- **public static bool GetCardReaderAt(int number, ref string reader)**

  Nachdem wir die Anzahl der gefundenen Reader erfragt haben, kann jetzt die Bezeichnung jeden Readers einzeln geholt werden.

2.3. Vorbereitung

Damit die Sicherheitseinrichtung richtig funktionieren kann müssen vor der laufenden Verwendung der Signaturfunktion bestimmte Werte in der/Ihrer Datenbank abgelegt werden.

- Public-Zertifikat
- Public-Zertifikat-Kette
- Seriennummer der Karte
- Karten-PIN
- AES-Key (siehe RksvAesGenerator.exe, damit können Sie Ihren AES-Key generieren)
- dotNet 4.0 oder höher wird benötigt
- falls ein XP Betriebssystem zum Einsatz kommt, dann ist ein SP3 erforderlich

Über die Seriennummer kann ein Signaturvorgang auf der Karte gestartet werden, die Zertifikate werden beim Export benötigt. Es gibt auch die Möglichkeit, die Zertifikate beim Export direkt von der Karte zu holen, aufgrund der Möglichkeit eines Ausfalls der Karte wird von dieser Option abgeraten.

Die Seriennummer kann erlangt werden indem der Name des entsprechenden Cardreaders als Parameter der **GetCardInReader**-Funktion gesetzt wird. Eine Liste aller verfügbaren Reader bekommt man mit der Funktion **GetReaders**.

Die Zertifikate erhält man mit der Funktion **GetCardCertificate** unter Angabe der betroffenen Seriennummer.

2.4. Aktivierung der Lizenz

Die RKSV-Bibliothek wird hardwaregebunden per PC lizenziert (die Demoversion ist 50 Tage gültig).

Hierzu muss zunächst mit der Funktion **GetMachineCode**(…) ein eindeutiger Maschinencode generiert werden. Das Ergebnis ist ein Base64-kodierter String der einem Byte[] mit 32 Byte entspricht. Dieser Code wird an die ITG-Consulting (**office@itgconsulting.at**) übermittelt, nach Überprüfung der Daten wird seitens der ITG ein Lizenzfile generiert und an den User verschickt.

2.5. Verwendung über COM (Component Object Model)

Zuerst muss die **ITGRKSVCom.dll** in ihrem Windows-Betriebssystem registriert werden. Das passiert in der Regel automatisch mit der Installation der Bibliothek. Falls das nich möglich sein sollte, kann das manuell durchgeführt werden.

Dies wird mittels Regasm**.exe** in einer als Administrator gestarteten Konsole (cmd.exe) durchgeführt.

<pfad der dll>\regasm_x86.exe ITGRKSVCom.dll /Codebase

Bitte achten Sie darauf, dass Regasm.exe im Administratormodus ausgeführt wird.

Sollten Sie damit auch keinen Erfolg haben, dann versuchen Sie die die mitgelieferte ITGRKSVCom.reg in Ihre Registry zu importieren (muss auch als Administrator ausgeführt warden)

Zusammen mit der Dll wird von uns auch eine zugehörige type-library mitgeliefert (ITGRKSVCom.tlb).
Aus diesem File kann jetzt das RKSVCOM-interface generiert werden. Es dürfte in ihrer IDE auch die Möglichkeit geben direkt nach dem .tlb-file zu browsen und die zugehörige dll über die IDE zu registrieren.

Anhand der Entwicklungsumgebung „Gupta Team Developer“ werden nun die generierten Klassen kurz erläutert. In anderen Umgebungen gibt es ähnliche Mechanismen um auf COM-Objekte zugreifen zu können.

- Unter Tools/ActiveX-Explorer wird die entsprechende COM-Library (RKSVCOM) bzw. das ITGRKSVCom.tlb rausgesucht
- Das Interface wird generiert
- Wir erhalten ein include-file (im TD sind dies *.apl-files)
  Darin enthalten sind mehrere Klassen:
  - Mscorlib__Object (Functional Class)
  - Object (Functional Class)
  - OleErrorInfo (Functional Class)
  - ITGRKSVCom_RKSVCom (Functional Class)
  - ITGRKSVCom_IRKSVCom (Functional Class)
  - ITGRKSVCom_RKSVCOM (COM Proxy Class)
  - SafeArray (Functional Class)
  - Variant (Functional Class)

Im Fall des Team Developers wird die COM Proxy Class verwendet. Diese ist von der RKSV_Single_COM_IRKSVCOM abgeleitet und enthält alle Funktionen des Interface und zusätzlich die Create-Funktion, die das Objekt erst initialisiert (entspricht einem „new“ in C#).

Objekt anlegen: ITGRKSVCom_RKSVCOM: cRKSV

Objekt initialisieren: Call cRKSV.Create()

Signierung einer Rechnung:

Set bOk = cRKSV.SignReceipt(dfKassenID, dfReaderName, dfPIN, dfBelegnummer, dfBelegDatumUhrzeit, dfBetragSatzNormal, dfBetragSatzErm1, dfBetragSatzErm2, dfBetragSatzBesonders, dfBetragSatzNull, dfUmsatzzaehler, dfSigVorigerBeleg, dfAesKey, sJWSText, sQRText, sOCRAText, sRetVal, bOk1)

Wenn bOk1 true ist befindet sich das Ergebnis der Signaturerstellung in der Variable sJWSText, das Base64-kodierte Bild des QR-Codes in der Variable sQRText und OCR-A in sOCRAText. Wenn bOk1 false ist, steht in sRetVal eine Fehlermeldung.

Wenn bOk auf false steht hat es Probleme beim Invoke der Funktion über COM gegeben.

2.6. Verwendung als .Net-dll

Die RKSV.Single.COM.dll kann in einem .Net-Projekt einfach als Verweis hinzugefügt werden. Dabei muss die dll nicht mittels regsvr32.exe registriert werden.

Bespielcode (C#):

```
RKSVCOM myComObject;

public Form1()
{
    InitializeComponent();
    myComObject = new RKSVCOM();
    myComObject.ActivateLicense();
 }

private void btnSign_Click(object sender, EventArgs e)
{
    string aes = tbAes.Text;
    string sAntwort = string.Empty;
    string jwt = string.Empty;
    string qr = string.Empty;
    string ocra = string.Empty;

    if(!myComObject.SignReceipt(tbKassenID.Text, tbReaderName.Text, tbCardPIN.Text,
tbBelegnummer.Text,     tbBelegDatumUhrzeit.Text, tbBetragSatzNormal.Text,
        tbBetragSatzErm1.Text, tbBetragSatzErm2.Text, tbBetragSatzBesonders.Text,
        tbBetragSatzNull.Text, tbStandUmsatzzaehler.Text, tbSigVorigerBeleg.Text,
        aes, ref jwt, ref qr, ref ocra, ref sAntwort))
    {
        MessageBox.Show(sAntwort);
    }
    else
    {
        byte[] bmpBytes = Convert.FromBase64String(qr);
        System.IO.MemoryStream ms = new System.IO.MemoryStream(bmpBytes);
        Image bmp = Image.FromStream(ms);
        pbQrCode.Image = bmp;

        tbJWT.Text = jwt;
```

```
            tbSigVorigerBeleg.Text = jwt;
        }
    }
```

Hinweis: siehe auch die Tester.exe, in der die Funktionen als Beispiel umgesetzt wurden. Der zugehörige Source-Code befindet sich im c# Verzeichnis.

2.7. Exportfunktion

Zumindest **vierteljährlich** muss ein Export aller Rechnungen auf einen externen Datenträger stattfinden. Hierzu wird ein SafeArray mit allen zu exportierenden Rechnungen (JWS-Kompakt-Darstellung) erstellt und samt dem Zertifikat und der Zertifikatskette an die Funktion ExportJWS übergeben. Diese Funktion liefert einen String retour, der das exportierte Paket darstellt.

3. Begriffserklärung

3.1. Base64

Base64 beschreibt ein Verfahren zur Kodierung von 8-Bit-Binärdaten. Es kommen nur lesbare, Codepageunabhängige ASCII-Zeichen zum Einsatz. Die RKSVLight-Bibliothek verwendet Base64 für jegliche Arten von Binärdaten. Dies wären die Repräsentationen der Bilder der QR- und OCRA-Codes. Bilder müssen vor der Anzeige bzw. dem Andruck in Binärdaten überführt werden. Mit der Funktion SaveImageToFile(…) können Bilder direkt aus der Base64-Repräsentation in ein Bildformat auf einem Datenträger überführt und dann weiter verarbeitet werden wie z.B. an den Bondrucker schicken.

3.2. Steuersätze

In die Felder der Signaturfunktionen kommen ausschließlich Bruttobeträge hinein. Daher werden auch Bruttobeträge dem Umsatzzähler hinzugerechnet oder abgezogen (Storno).

Bei der Aufteilung der Beträge nach Steuersätzen kommen die folgenden zum Einsatz:

- Normal (20 %)
- Ermäßigt-1 (10 %)
- Ermäßigt-2 (13 %)
- Besonders (19 %)
- Null (0 %)

3.3. Besondere Belege

3.3.1. Trainingsbuchungen

Bei einer Trainingsbuchung wird statt dem Verschlüsselten Umsatzzähler, die Base64-kodierte Zeichenkette „TRA“ eingefügt.

Die Funktion SignTrainReceipt(…) dient diesem Zweck und bekommt weder Umsatzzähler noch einen AES-Key als Parameter.

3.3.2. Stornobuchungen

Stornobuchungen werden markiert indem die Base64-kodierte Zeichenkette „STO“ statt dem Umsatzzähler dargestellt wird. Der Umsatzzähler wird im DEP (Datenerfassungsprotokoll) sehr wohl verändert und muss vom Kassenprogramm mitgeführt werden (Stand 10.02.2016).

Die Funktion SignStornoReceipt(…) dient diesem Zweck und bekommt weder Umsatzzähler noch einen AES-Key als Parameter.

3.3.3. Weitere wichtige aktuelle Hinweise finden Sie auf:

https://www.wko.at/registrierkassen

und

https://www.bmf.gv.at/top-themen/Registrierkassen.html

sowie im PDF-Dokument: **FESTLEGUNGEN DES BMF ZU DETAILFRAGEN DER REGISTRIERKASSENSICHERHEITSVERORDNUNG (RKSV)**