SAFECOR
SICHERHEIT UND AUTOMATION

SafeNet - SEPA konfigurieren und planen

E-Mail: m.stark@safecor.de

Datum: 19.08.2013

Inhaltsverzeichnis

1 SafeNet Version 9.3-01 oder höher installieren ____ 4
2 Änderungen am Datenhaushalt im Vergleich zu DTA ____ 5
3 Vorhandende Bankverbindungsdaten der zahlungspflichtigen Personen per Script zu IBAN/BIC konvertieren ____ 6
4 Prüfung der konvertierten Daten auf Richtig- und Vollständigkeit ____ 7
5 MandatIDs für die zahlungspflichtigen Personen per Script generieren ____ 8
6 Prüfung der generierten MandatIDs und Datum auf Richtig- und Vollständigkeit ____ 9
7 Erforderliche Konfigurationsdateien ____ 10
8 Weitere benötigte Dateien ____ 14
9 Erzeugte Abrechnungsdateien ____ 15
10 SEPA und DTA parallel betreiben ____ 16

Dieses Dokument beschreibt die Installation und Konfiguration von SEPA in SafeNet.

Dokument-Version	0.9 (Änderungen vorbehalten - Ein produktiver Einsatz darf erst mit Version 1.0 erfolgen)
SafeNet Version	9.3-01

1 SafeNet Version 9.3-01 oder höher installieren

Zunächst muss die erforderliche SafeNet Version (9.3-01 oder höher) installiert werden. Erst ab dieser Version ist SEPA in SafeNet integriert. Alle für die hier beschriebene Konfiguration erforderlichen Dateien werden in dem Update-Paket der jeweiligen Version vorhanden sein. Auf die Vorgehensweise eines SafeNet-Updates wird in diesem Dokument nicht weiter eingegangen.

2 Änderungen am Datenhaushalt im Vergleich zu DTA

Bisher wurde zur Abrechnung die Bankverbindungsdaten der Person verwendet. Das beutet, man konnte die Felder Bankbezeichnung, Bankleitzahl und Kontonummer über die Personenverwaltung (auch ohne Fachzuordnung) ändern und die Änderung war mit einem Mal bei jedem Vorkommen dieser Person aktiv. Mit SEPA steht die Bankverbindung der Abrechnerperson in Abhängigkeit zu einem vermieteten Fach mit einer Fachnnummer und einer MandatID. In SafeNet unterscheiden wir aus technischer Sicht die Objekte Person, Fach und Person an Fach. Letzteres beschreibt die Beziehung der Person zu einem vermieteten Fach. In diesem Objekt wird die MandatID, Mandat Unterschriftsdatum und alle weiteren SEPA-relevanten Informationen gespeichert. Aus diesem Grund findet man die SEPA-relevanten Felder beim Exportieren und Importieren des Bestandes in der Datei VermieteteFaecher_<AnlagenID>.xml und nicht, wie bisher, in der Datei personen.xml.

3 Vorhandende Bankverbindungsdaten der zahlungspflichtigen Personen per Script zu IBAN/BIC konvertieren

Aus Bankleitzahl und Kontonummer werden die entsprechenden IBAN- und BIC-Werte ermittelt und in die dafür vorgesehenen Felder eingetragen. Hierzu muss das Script CcIbanToolWithBIC.groovy in das Verzeichnis properties\groovy\de\contecon\groovy\jobs kopiert werden. Sind die Felder IBAN und BIC bereits gefüllt, so bleiben diese unangetastet und werden nicht überschrieben.

Das Script kann per Server-Konsolbefehl gestartet werden:

```
rungroovy .\properties\groovy\de\contecon\groovy\jobs\CcIbanToolWithBIC
```

Es werden keine Parameter benötigt. Alternativ kann das Script als Job angelegt und über die Jobverwaltung gestartet werden.

⛔ Das Kopieren von Scriptdateien in das Verzeichnis properties\groovy und dessen Unterverzeichnisse im laufenden Betrieb des Servers erfordert entweder einen Neustart des Servers oder den Server-Konsolbefehl **reloadgroovyscripts** um die neu installierten Scripte einzulesen und zu aktivieren.

4 Prüfung der konvertierten Daten auf Richtig- und Vollständigkeit

Im Zuge der Entwicklung wird ein Report erstellt, der die Zahlungspflichtigen und deren IBAN und BIC auflistet. Dieser liegt zum jetzigen Zeitpunkt noch nicht vor, wird aber mit erscheinen an dieser Stelle dokumentiert.

5 MandatIDs für die zahlungspflichtigen Personen per Script generieren

Die Felder MandatID und Mandat-Datum werden gemäß Übergabeparameter befüllt. Sind bereits Werte für diese Felder vorhanden, so bleiben diese unangetastet und werden nicht überschrieben. Hierzu muss das Script SepaSetMandatID.groovy in das Verzeichnis properties\groovy\de\contecon\groovy\jobs kopiert werden.

Das Script kann per Server-Konsolbefehl gestartet werden:

```
rungroovy .\properties\groovy\de\contecon\groovy\jobs\CcIbanToolWithBIC
mandateid=$aid$$fnr$$vermdat$$pid$ mandatedate=$dat$
```

Es werden Parameter für den Aufbau der MandatID (mandateid) als auch für das Datum des Mandats (mandatedate) benötigt. Hierzu stehen folgende Variablen zur Verfügung:

Variable	Beschreibung
aid	ID der jeweiligen Anlage
anl	Bezeichnung der jeweiligen Anlage (nicht empfohlen)
fnr	Fachnummer des jeweiligen Faches
dat	Aktuelles Datum (nicht empfohlen)
$vermdat$	Vermietdatum des jeweiligen Faches
pid	ID der jeweiligen Personen (Abrechner)

Es können anstelle der Variablen auch feste Werte angegeben, sofern dies für die Konfiguration sinnvoll ist. Alternativ kann das Script als Job angelegt und über die Jobverwaltung gestartet werden.

Gemäß den oben angegebenen Parametern wird nach Auflösung der Variablen beispielsweise eine MandatID an einem Fach **A1 00001 13082013 52** mit dem Datum 13.08.2013 vergeben, da das Fach am 13.08.2013 vermietet wurde, der 13.08.2013 das aktuelle Datum war und die Person nach Erfassung in SafeNet die PersonenID (PID) 52 zugeordnet bekam.

Teil der MandatID	Beschreibung
A1	AnlagenID
00001	Fachnummer
13082013	Aktuelles Datum
52	PersonenID

6 Prüfung der generierten MandatIDs und Datum auf Richtig- und Vollständigkeit

Es werden jeweils Ereignisse für MandatId und Mandat Unterschriftsdatum auf dem Server erzeugt:

```
MandatID gesetzt - Fach:00001 Abrechner: Michael Rettig PID: 52 MandatID: A1000011308201352
Datum Mandat Unterschrift gesetzt - Fach: 00001 Abrechner: Michael Rettig PID: 52
```

Des weiteren wird eine Logdatei dieses Scripts unter properties\groovletOut mit dem Namen SepaSetMandatID-<Zeitstempel>.html erstellt. Diese kann über das Dateisystem des Servers oder im Webportal unter Statistik - Jobergebnisse anzeigen (Name: SepaSetMandatID) eingesehen werden.

7 Erforderliche Konfigurationsdateien

Zunächst wird in den globalen Parametern die Abrechnung auf SEPA umgestellt.

properties\parameter.properties

```
# Umstellung von DTA auf SEPA
SettlementType=sepa_de
```

Nun müssen die globalen (anlagenübergreifenden) Parameter für SEPA gesetzt / überprüft werden. Eine detaillierte Erläuterung zu den hier verwendeten Parametern ist unter
http://www.bundesbank.de/
verfügbar.

properties\SRV_SEPA_DE.properties

```
# Konstante für Währung (z.Zt. nur EUR)
# default: EUR
sepaParameter.amtCcy=EUR

# Wenn gesetzt, werden für interne Kunden keine Mandate benötigt (2 Dateien werden erzeugt)
# default:
sepaParameter.bicInternal=HYVEDEMM475

# Gebühren werden entsprechend gemäß den im ServiceLevel vereinbarten Regeln berechnet
# default: SLEV
sepaParameter.chargeBearer=SLEV

############ Sektion Gutschriften ############

# BIC des Kreditinstitut des Auftraggebers
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.creditTransferDebitorBic=HYVEDEMM475

# International Bank Account Number (IBAN) des Auftraggebers
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.creditTransferDebitorIban=DE92660202861457032621

# Name des Auftraggebers oder des Kontoinhabers
# Name ist begrenzt auf 70 Zeichen
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.creditTransferDebitorName=UNICREDIT BANK KARLSRUHE

############ Sektion Lastschriften ############

# Eindeutige Identifikation eines Kreditinstituts (Zahlungsempfängers)
```

```
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.directDebitBic=HYVEDEMM475

# Konto des Zahlungsempfängers
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.directDebitIban=DE92660202861457032621

# Name des Zahlungsempfängers
# Name ist begrenzt auf 70 Zeichen
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.directDebitName=Name EmpfaengerSRV

# Eindeutiges Identifikationsmerkmal des Zahlungsempfängers
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.creditorId=UNICREDIT BANK KARLSRUHE

############ Ende Sektion Lastschriften ############

# (In Entwicklung)
sepaParameter.hideBlzAccount=true

# Name des Einreichers (InitiatingParty)
# Name ist begrenzt auf 70 Zeichen
# Falls hier nicht gesetzt, muss dieser Parameter bei den entsprechenden Anlagen (z.B.
A1_SEPA_DE.properties) gesetzt werden
# default:
sepaParameter.initiatingParty=InitBKSRV

# Local-Instrument-SEPACode
# Nur CORE (SEPA-Basislastschrift) und B2B (SEPA-Firmenlastschrift) ist zulässig
sepaParameter.localInstrumentCd=CORE

# Alle Buchungen in einer Datei zusammenfassen
# default: true
sepaParameter.onefileonly=true

# PaymentMethod Überweisung (Zahlungsinstrument)
# default: TRF
sepaParameter.paymentMethodPain001=TRF

# PaymentMethod Bankeinzug (Zahlungsinstrument)
# default: DD
sepaParameter.paymentMethodPain008=DD

# Requested-Collection-Date (Lastschrift)
# Int-Wert beschreibt die Tage die zum Erstellungsdatum hinzugerechnet werden sollen
# Faelligkeitsdatum der Lastschrift bei Ersteinzug
# default: 1
sepaParameter.requestedCollectionDateDaysFirst=3
```

```
# Requested-Collection-Date (Lastschrift)
# Int-Wert beschreibt die Tage die zum Erstellungsdatum hinzugerechnet werden sollen
# Faelligkeitsdatum der Lastschrift bei Folgeeinzug
# default: 1
sepaParameter.requestedCollectionDateDaysRecur=3

# RequestedExecutionDateDays (Gutschrift)
# Int-Wert beschreibt die Tage die zum Erstellungsdatum hinzugerechnet werden sollen
# Vom Kunden gewünschter Ausführungstermin
# default: 1
sepaParameter.requestedExecutionDateDays=3

# sepaHandlerClass beschreibt die Klasse (Version) die zur Sepa-Generierung benutzt wird
# de.contecon.sepa.SepaHandler_V1 --> Ab 4. November 2013  Version 2.7
# de.contecon.sepa.SepaHandler_V0 --> bis 3. November 2013  Version 2.5, 2.6
# default: de.contecon.sepa.SepaHandler_V0
sepaParameter.sepaHandlerClass=de.contecon.sepa.SepaHandler_V1

# Service Level / Service Schema
# Konstante SEPA
# default: SEPA
sepaParameter.serviceLevel=SEPA

# Folgende Parameter müssen vorhanden sein, bleiben aber bis auf Weiteres ohne Wert
sepaParameter.XMLFileTemplateForDirectDebit=
sepaParameter.XMLFileTemplateForCreditTransfer=

# Wenn sepaParameter.useFlagBtchBookgDirectDebit auf true gesetzt ist, wird das optionale Feld
gemäß dem Wert von sepaParameter.btchBookgDirectDebit bei Lastschriften gesetzt
# Anderenfalls ist das Feld nicht vorhanden
sepaParameter.useFlagBtchBookgDirectDebit=false

# Auftraggeberbuchung Sammler/Einzelsatz
# Optional, wenn sepaParameter.useFlagBtchBookgDirectDebit=true
# true  = Sammelbuchung
# false = Einzelsatzbuchung
sepaParameter.btchBookgDirectDebit=true

# Wenn sepaParameter.useFlagBtchBookgCreditTransfer auf true gesetzt ist, wird das optionale
Feld in Sepa gemäß dem Wert von sepaParameter.btchBookgCreditTransfer bei Gutschriften gesetzt
# Anderenfalls ist das Feld nicht vorhanden
sepaParameter.useFlagBtchBookgCreditTransfer=false

# Auftraggeberbuchung Sammler/Einzelsatz
# Optional, wenn sepaParameter.useFlagBtchBookgCreditTransfer=true
# true  = Sammelbuchung
# false = Einzelsatzbuchung
sepaParameter.btchBookgCreditTransfer=true

# Im Standardfall erhalten alle erzeugten Dateien die Endung .sepa
# Die Dateiendung kann hier veraendert werden werden (Endung ohne Punkt)
# default: sepa
sepaParameter.sepaFileExtension = xml

# Über den Parameter useReferenceCharHandling gibt es die Möglichkeit die Zeichenumsetzung beim
Verwendungszweck zu konfigurieren
# 0 == ein Leerzeichen bei Umlauten und Sonderzeichen
# 1 == ein Umlaut wird in zwei Vokale, ß in ss umgesetzt
```

```
# 2 == Umlaute und Sonderzeichen werden gemäß Epc best practice transformiert
sepaParameter.useReferenceCharHandling = 2
```

Im zweiten Schritt können die zuvor gesetzten globalen Parameter (SRV_SEPA_DE.properties) für jede einzelne Anlage optional überschrieben werden. Hierzu ist die AnlagenID der jeweiligen Anlage im Dateinamen anstelle von SRV einzusetzen (Beispiel: A1_SEPA_DE.properties).

8 Weitere benötigte Dateien

- properties\groovy\de\contecon\groovy\jobs\SepaFilter.groovy
  Script, um Elemente (XML-Tags) der erzeugten SEPA-Datei zur Laufzeit anpassen zu können (optional)

9 Erzeugte Abrechnungsdateien

Erzeugte Abrechnungsdateien werden, wie auch die bisherigen DTA-Dateien, im Verzeichnis properties\abr abgelegt. Ebenso überschreibt ein zweiter Abrechnungslauf gleichen Datums, wie bisher unter DTA auch, die erzeugten Dateien des zuvor erfolgten Abrechnungslaufes. Der Dateinamen setzt sich aus SEPA_Abr_<Jahr des Abrechnungslaufes>_<Monat des Abrechnungslaufes>_<AnlagenID>_<Fortlaufender Zähler> zusammen. Beispiel: Sepa_Abr_2013_12_A1_0.xml
Die Dateiendung ist über den oben beschriebenen Parameter sepaParameter.sepaFileExtension konfigurierbar.

10 SEPA und DTA parallel betreiben

Es ist nicht möglich, SEPA und DTA parallel zu betreiben. Um, aus welchen Gründen auch immer, SEPA zu deaktivieren und DTA zu reaktivieren, reicht das Auskommentieren der oben beschriebenen Änderung in der Datei properties\parameter.properties aus:

```
# SettlementType=sepa_de
```

Das Zeichen # kommentiert den Parameter aus und nach einem Server-Neustart steht der Standard, in diesem Falle DTA, wieder zur Verfügung.