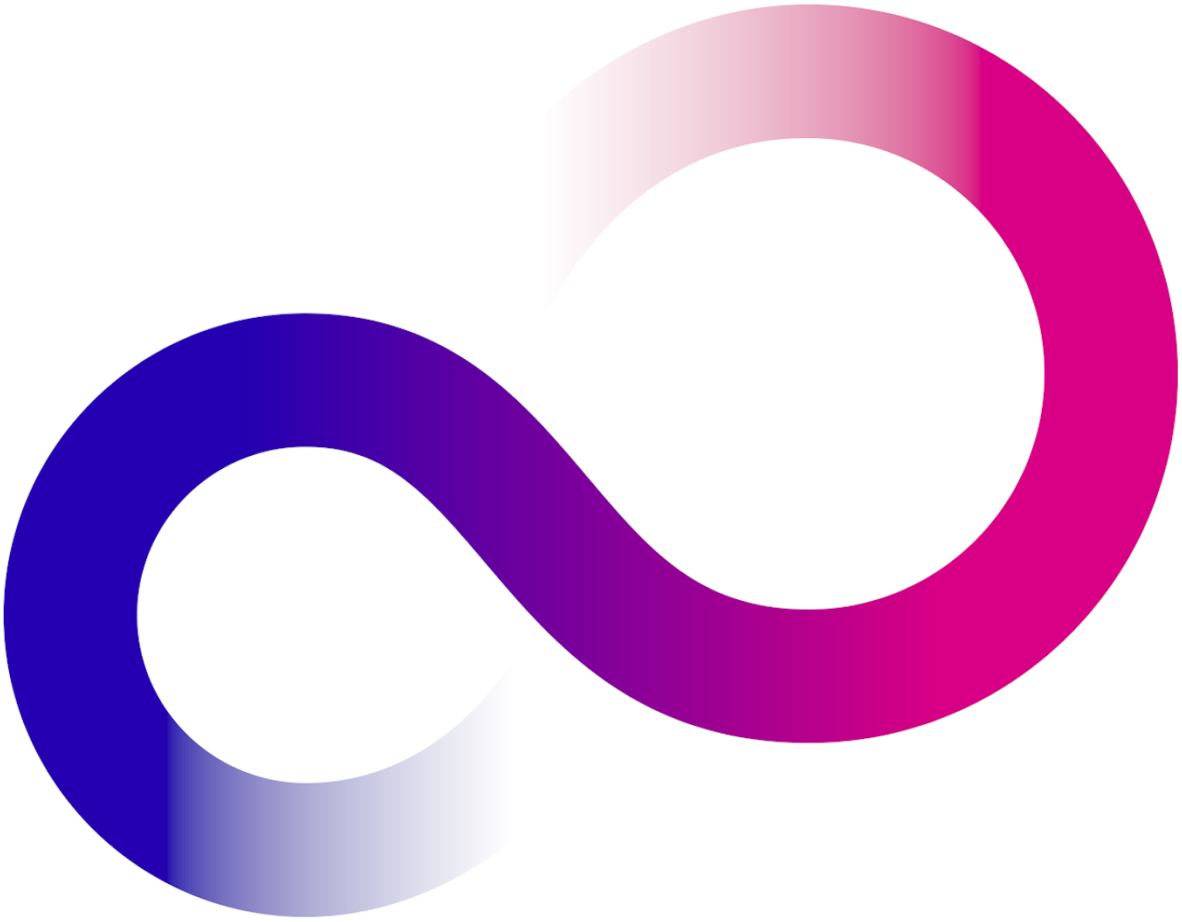
FUJITSU



SecDocs V3.2A00 Installationsanleitung

SecDocs Team

Version SecDocs 3.2A00, Stand: 30.06.2022

Inhaltsverzeichnis

Juni 2022

SecDocs Team (secdocs@ts.fujitsu.com)

1. Softwarevoraussetzungen

Die SecDocs Software ist derzeit für das Betriebssystem **SUSE SLES15 SP3 64bit (AMD64/x64)** oder einer höheren SLES15 SPx Version freigegeben.

Für den Betrieb der SecDocs Software wird die folgende Softwarekomponente benötigt:

- **Oracle Database 19c (19.13 oder höher) für Linux 64bit**
  https://www.oracle.com/database/technologies/oracle-database-software-downloads.html
  Die Datenbank Software kann optional auch auf einem anderen Rechner laufen.
  Eine Beschreibung der Oracle Datenbank Software findet man hier:
  https://docs.oracle.com/en/database/oracle/oracle-database/19/index.html

2. Lieferumfang

- SecDocs
  SecDocs ist eine Java Enterprise Anwendung, die in Java 8 programmiert ist und auf einem WildFly Server läuft. Die Software wird ablauffähig (mit OpenJDK8 64bit Update 332 und WildFly 24.0.1) ausgeliefert.
- CryptoModule Software
  Fujitsu DSEngine 2.0.1
  (wird für den Betrieb der SecDocs Software benötigt)

3. Bereitstellen der SecDocs Ablaufumgebung

3.1. Sprachumgebung

Die SecDocs Software muss mit einem UTF-8 Encoding ablaufen. Es ist deshalb darauf zu achten, dass die Sprachumgebung auf dem Rechner entsprechend gesetzt ist, z.B.:

```
export LANG=en_US.UTF-8
```

oder

```
export LANG=de_DE.UTF-8
```

Ist keine Sprache mit UTF-8 Encoding eingestellt (z.B. LANG=de) wird beim Start der SecDocs Anwendung eine Warnung ausgegeben und es wird

```
LANG=en_US.UTF-8
```

gesetzt.

Damit in allen Ablaufumgebungen (Start als Service oder Start aus der Kennung secdocs) das gleiche Verhalten vorliegt, wird die Sprache, mit der die SecDocs Anwendung abläuft, im Skript /home/secdocs/bin/setSecDocsEnv.sh eingestellt:

```
SECDOCS_LANG=en_US.UTF-8
```

oder

```
SECDOCS_LANG=de_DE.UTF-8
```

Ist die Variable gesetzt (Standardwert: *en_US.UTF-8*), wird die LANG Environmentvariable auf diesen Wert gesetzt.

Die LANG Environmentvariable steuert, in welcher Sprache die SecDocs Anwendung ihre Meldungen ausgibt. Derzeit werden die Sprachen Englisch (Default) und Deutsch unterstützt. Die zugehörigen Meldungsdateien sind im Verzeichnis

```
/home/secdocs/jboss/secdocs/configuration/secdocs/i18n
```

abgelegt. Wird eine andere Sprache eingestellt, erscheinen die Meldungen in Englisch.

3.2. Filesystem Konfiguration

Beim Speichern eines SDOs werden ca. 7 Inodes im Filesystem benötigt. Die maximale Anzahl von Inodes in einem Filesystem ist jedoch in der Regel begrenzt, kann aber per Tuning vergrößert werden.

Wichtig: Vor der Inbetriebnahme des SecDocs Archivs ist zu überprüfen, ob die vorkonfigurierte maximale Anzahl von Inodes ausreichend ist.

3.3. Datenbank Konfiguration

Für alle unterstützten Datenbanksysteme ist zu beachten:

- Der Default Zeichensatz muss UTF-8 sein
- Der verwendete Datenbankbenutzer muss die folgenden Berechtigungen haben:

```
ALTER TABLE, CREATE TABLE, CREATE TEMPORARY TABLES,
DROP TABLE, CREATE INDEX, SELECT, INSERT, UPDATE, DELETE.
```

3.3.1. Oracle Datenbank Konfiguration

Wichtig: In der Regel ist für eine Oracle Datenbankinstanz der XA Support nicht konfiguriert, aber für den Betrieb der SecDocs Anwendung notwendig. Den XA Support kann der Oracle Administrator folgendermaßen aktivieren:

```
$ cd $ORACLE_HOME/rdbms/admin
$ sqlplus /nolog
connect sys/<password> as sysdba
@xaview
exit
```

Zum Betrieb der SecDocs Anwendung wird ein Oracle Datenbankbenutzer (hier ***dbUser*** genannt) benötigt:

CREATE USER "***dbUser***" IDENTIFIED BY "***dbPassword***"
PROFILE "DEFAULT" DEFAULT TABLESPACE "USERS"
QUOTA unlimited on "USERS"
ACCOUNT UNLOCK;

Dieser Datenbankbenutzer muss über die folgenden Rechte verfügen:

GRANT SELECT ON sys.v$xatrans$ TO ***dbUser***;
GRANT SELECT ON sys.dba_pending_transactions TO ***dbUser***;
GRANT SELECT ON sys.pending_trans$ TO ***dbUser***;
GRANT SELECT ON sys.dba_2pc_pending TO ***dbUser***;
GRANT EXECUTE ON sys.dbms_system TO ***dbUser***;

GRANT CONNECT TO ***dbUser***;
GRANT RESOURCE TO ***dbUser***;

Für die später noch durchzuführende Datenbank Konfiguration der SecDocs Software in der SecDocs WildFly Server Instanz sind die folgenden Daten vom Oracle Datenbank Administrator zu erfragen:

- **dbHost**
  Name des Rechners, auf dem der Listener der Oracle Datenbankinstanz läuft.
- **dbPort**
  Portnummer, auf der der Listener der Oracle Datenbankinstanz lauscht.
  (Default: 1521)
- **dbService**
  Name des Oracle Datenbankservice.
- **dbSID**
  SID der Oracle Datenbankinstanze.
- **dbUser**
  Name des zu verwendenden Datenbankbenutzers.
- **dbPassword**
  Passwort des zu verwendenden Datenbankbenutzers.

3.4. Triple Store

Ab der SecDocs Version V3.1 wird die Nutzung einer Triple Store Software nicht mehr unterstützt.

3.5. Mountpunkt für Fujitsu ETERNUS CS High anlegen (Benutzer: root)

Als Root einen NFS3 Mount in der Datei /etc/fstab eintragen, z.B (NetApp Filer).:

```
filerHost:/vol/secdocs /filer nfs rw,nodev,auto,noexec,timeo=600,tcp,vers=3,rsize=32768,wsize=32768,hard,bg,retry=100 0 0
```

Wichtig: der obige Eintrag muss als **eine** Zeile in der /etc/fstab Datei geschrieben werden.

Nach dem ersten Mounten muss der Mountpunkt die Zugriffsrechte des SecDocs Benutzers (Kennung secdocs) erhalten

Die Einstellungen für ETERNUS CS High End sprechen Sie bitte mit Ihrem Storage Betreuer ab.

Wichtig: In Linux hat jeder Benutzer eine UID (User ID) und jede Gruppe eine GID (Group ID). Diese Werte sind mit der Filer Administration abzusprechen.

3.6. Hinweise zu einer Upgrade Installation

Vor einer Upgrade Installation ist die SecDocs Anwendung zu beenden.

Danach kann man die alte Software zuerst deinstallieren und danach die neue Software installieren. Vor der Deinstallation sollte man sich eine Liste der Konfigurationsdateien sichern.

Eine Liste aller als Konfigurationsdateien vermerkten Dateien erhält man mit den folgenden Kommandos:

```
# rpm -qc secdocs
# rpm -qc secdocscm
```

Die Software kann man mit den folgenden Kommandos deinstallieren:

```
# rpm -ev secdocs
# rpm -ev secdocscm
```

Bei diesem Vorgehen bleiben die geänderten Konfigurationsdateien mit dem Suffix .rpmsave an ihrem Installationsort erhalten.

3.7. Fujitsu DSEngine Software

Eine Beschreibung (Installation/Deinstallation/Konfiguration/...) zu dieser Software findet man in der Beschreibung „Fujitsu DSEngine (CryptoModule) für SecDocs 3.2“ (Datei: SecDocs_DSEngine-InstallationGuideDE-2.0.1.1.pdf).

3.7.1. DSEngine Installation (Benutzer: root)

Die Software findet man im Verzeichnis *pkgs*.

Die Fujitsu DSEngine Software wird vom Benutzer root mit Hilfe des RPM Verfahrens installiert:

```
# rpm -ivh secdocscm-2.0.0.1-1.x86_64.rpm
```

Bei der Installation wird (soweit nicht schon auf dem Rechner vorhanden) der Linux Benutzer secdocs und die zugehörige Linux Gruppe secdocs angelegt. Der neu angelegte Benutzer hat das Homeverzeichnis */home/secdocs*. Die gesamte DSEngine Software wird unter dem Verzeichnis */home/secdocs/CryptoModule* abgelegt. Außerdem wird der systemd Service cryptoModule.service eingerichtet.

Weitere Details findet man in der DSEngine Installationsanleitung.

3.8. SecDocs Installation (Benutzer: root)

Die Software findet man im Verzeichnis *pkgs*. Die SecDocs Software wird vom Benutzer root mit Hilfe des RPM Verfahrens installiert:

```
# rpm -ivh secdocs-3.2.1.0-1.x86_64.rpm
```

Ist bereits eine ältere Version installiert, ist diese vor der Installation der neuen Version zu deinstallieren.

Bei der Installation wird (soweit nicht schon auf dem Rechner vorhanden) der Linux Benutzer secdocs und die zugehörige Linux Gruppe secdocs angelegt. Der neu angelegte Benutzer hat das Homeverzeichnis */home/secdocs*.

Nach dem Ausführen der obigen Schritte ist die SecDocs Installation abgeschlossen und man hat im Homeverzeichnis der Kennung secdocs die folgenden Dateiverzeichnisse:

- admin
  Administrations Tools Verzeichnis
  - purge In diesem Verzeichnis befindet sich das Script purgeData (SecDocs Purge Tool)
  - recovery In diesem Verzeichnis befindet sich das Script recoverFromStorage (SecDocs Recovery Tool).
- bin
  SecDocs Start/Stop Script und Diagnosescripts
- docs/licenses
  Dieses Verzeichnis enthält die Lizenztexte der in SecDocs verwendeten Open Source Komponenten.
  In der Datei ThirdPartyLicenseReadme.txt findet man eine Angabe aller verwendeten Komponenten.
- install
  Skripte zum Anlegen und Löschen eines SecDocs SLES Service.
  Optionale Daten für die SecDocs WildFLy AS Instanz.
  migration In diesem Verzeichnis befindet sich das Script startMigration zur Migration der SecDocs Datenbank.
- OpenJDK
  OpenJDK8 64bit Software
- jaxws wsimport generierte Web Service Client Stubklassen
  Im Verzeichnis bin gibt es zusätzlich das Script genArchivingSRWsClientStubs. Dieses Script zeigt, wie man aus der Datei schemas/3.2/ArchivingSR.wsdl die Web Service Client Stubklassen für den Archiving Web Service generieren kann.

javadoc JavaDoc zu den generierten Stubklassen
lib JAR-Dateien mit den Stub Klassen und Sourcen

- jboss
  secdocs SecDocs WildFly Serverinstanz
  secdocs/configuration SecDocs WildFly Server Konfiguration
  secdocs/configuration/secdocs SecDocs Konfigurationsdaten
  secdocs/log Logdateien
  wildfly WildFly Software.
- schemas
  SecDocs Web Services und zugehörige Datentypen
  3.2
  AdminCommon.xsd SecDocs Administrator spezifische Datentypen
  AdminData.xsd SecDocs Administrator spezifische Datentypen
  AdminUpdateData.xsd SecDocs Administrator spezifische Datentypen
  ArchiveAdmin.wsdl Archiv Administrator WSDL
  ArchivingData.xsd Archivierungsspezifische Datentypen
  ArchivingSR.wsdl Archivierungs WSDL Beispiel für die
  kundenspezifischen submit und retrieve
  Operationen
  Archiving.wsdl Archivierungs WSDL
  filter.xsd Schema für die SDO Filter
  MandantAdmin.wsdl Mandanten Administrator WSDL
  policy.xsd DSEngine Policy Schema
  result2.xjb JAXB Mappingdatei für das Tool wsimport
  secdocs.xsd SecDocs spezifische Datentypen
  sparql-protocol-types.xjb JAXB Mappingdatei für das Tool wsimport
  VerificationInfo.xsd Datentyp des Elementes SignatureVerificationInfo
  in der requestForEvidence Response
  VerificationInfo1.xsd Datentyp des Elementes SignatureVerificationInfo
  in der requestForEvidence Response
  XAIPExtensions.xsd SecDocs spezifische Erweiterungen in den
  TR-ESOR 1.2 S4 Antworten

  3.2/query SPARQL Schemata
  rdf.xsd
  result.xsd
  sparql-protocol-types.xsd
  xml.xsd

  3.2/samples
  MultiDocument.xsd Schema für Beispiel Document SDO
  MultiDocumentFilter.xml Beispielfilter

XAIP/1.2 TR-ESOR 1.2 spezifische Schemata und die S4 WSDL

Bei einem Update sollte man überprüfen, ob es Konfigurationsdateien mit der Endung .rpmsave gibt. Solche Dateien gibt es nur, wenn an den ausgelieferten Konfigurationsdateien Änderungen vorgenommen wurden. Es ist in diesem Fall darauf zu achten, dass die Änderungen wieder in die neu installierten Konfigurationsdateien übernommen werden.

4. SecDocs Konfiguration

SecDocs verwendet 2 Konfigurationsdateien:

- /home/secdocs/jboss/secdocs/configuration/standalone.xml
  Konfiguration des SecDocs WildFly Servers.
- /home/secdocs/jboss/secdocs/configuration/secdocs/secdocs.properties
  SecDocs spezifische Konfiguration.

Die SecDocs WildFly Konfiguration ist in der Datei

```
/home/secdocs/jboss/secdocs/configuration/standalone.xml
```

abgelegt.

Nach der Installation ist die SecDocs Anwendung für die Nutzung einer Oracle Datenbank vorkonfiguriert. Die folgenden Namen in dieser Datei müssen durch die realen Werte der installierten Oracle Datenbankumgebung ersetzt werden:

- dbHost
  Name des Rechners, auf dem der Listener der Oracle Datenbankinstanz läuft.
- dbPort
  Portnummer, auf der der Listener der Oracle Datenbankinstanz lauscht.
  (Default: 1521)
- dbService
  Name des Oracle Datenbankservice.
- dbUser
  Name des zu verwendenden Datenbankbenutzers.
- dbPassword
  Passwort des zu verwendenden Datenbankbenutzers.

Wichtig: In der Datei *standalone.xml* sind **zwei** Datasources (ArchiveDS und ArchiveDSXA) definiert.

Die SecDocs spezifische Konfiguration ist in der Datei

```
/home/secdocs/jboss/secdocs/configuration/secdocs/secdocs.properties
```

abgelegt.

Hier sind die beiden folgenden Einträge anzupassen:

```
# path to the root directory of the SecDocs archive data
archiveRoot=/filer

# Specify a List of Files separated by ;
# These Files are needed by the certified crypto components to
# write audit log files. If none of these files can be written,
# the crypto components will cease work and thus any requests to
# the Archiving Web Service will be rejected.
# To ensure that the crypto components work reliably please state
# at least to files located in different file systems or volumes
# on the filer.
# SecDocs will not start unless at least one of these audit log
# files is writable
cryptoAuditFiles=<path1>/cryptoLog1.log;<path2>/cryptoLog2.log
```

Hinweis zum Parameter cryptoAuditFiles
In dieser Datei (diesen Dateien) werden die Audit Einträge der DSEngine Komponenten, die in der SecDocs Anwendung genutzt werden, abgelegt. Hier muss mindestens eine Datei angegeben werden. Mehrere Dateien sind durch ein Semikolon (';') voneinander zu trennen. Da die DSEngine Komponenten nicht mehr genutzt werden können, wenn die angegebenen Dateien nicht mehr geschrieben werden können, sollte man hier mindestens 2 Dateien angeben, die in verschiedenen Filesystemen liegen!

Die Bedeutung der weiteren Parameter findet man im SecDocs Handbuch im Kapitel "Konfigurationsdatei secdocs.properties".

5. SecDocs Multi-Node Konfiguration

Optional kann man SecDocs auch in einer Multi-Node Konfiguration betreiben.
Multi-Node bedeutet, dass mehrere SecDocs Instanzen mit einer identischen Konfiguration (Datenbank und Storage) zusammenarbeiten. Damit eine solche Konfiguration unterstützt wird, sind in der Datei secdocs.properties die folgenden Einträge hinzuzufügen:

```
# For multi node support set this property to true
# Default false
multiNode=true
# Full path to the Hazelcast configuration file
# This property is mandatory in multi node mode and will be
# ignored in single node mode
multiNode.hazelcastConfigFile=<full-path-to>/hazelcast.xml
```

5.1. Hazelcast

Die Synchronization der einzelnen Multi-Node Instanzen erfolgt mit Hilfe der Hazelcast Software (http://www.hazelcast.com/). Die Hazelcast Schicht wird mit Hilfe der Datei hazelcast.xml konfiguriert. Eine Vorlage für die eigene Konfiguration findet man unter /home/secdocs/jboss/secdocs/configuration/secdocs/hazelcast.xml. Eine Beschreibung aller Konfigurationsoptionen findet man in der Hazelcast Dokumentation (*/home/secdocs/docs/hazelcast-documentation-3.5.4.pdf*).

Im Folgenden werden die Teile der Datei hazelcast.xml beschrieben, die man gegebenenfalls anpassen sollte:

<group>
<name>**SecDocs**</name>
<password>secdocs</password>
</group>

Bei gleicher Netzwerkkonfiguration (siehe multicast-group) bilden alle Instanzen mit dem gleichen Namen einen Hazelcast Cluster. Diesen Namen kann man nutzen, um verschiedene SecDocs Mult-Node Konfiguration mit jeweils einem eigenen Namen auszuzeichnen.

```
<port auto-increment="true" port-count="100">5701</port>
```

Es wird der Port 5701 als lokaler Listener Port verwendet. Ist dieser Port beim Start belegt, wird automatisch so lange (Schrittweite 100) hochgezählt, bis ein freier Port gefunden wurde.

```
<multicast enabled="true">
<multicast-group>224.2.2.3</multicast-group>
<multicast-port>54327</multicast-port>
</multicast>
```

Die Kommunikation im Hazelcast Cluster erfolgt via Multicast. Die Multicast Adresse und der Multicast

Port sind eventuell anzupassen.

```
<interfaces enabled="false">
<interface>192.168.1.*</interface>
</interfaces>
```

Für die Netzwerkkommunikation verwendet Hazelcast das erste gefundene Netzwerk Interface. Will man (bei mehreren Netzwerk Interfaces) für die Kommunikation ein bestimmtes Netzwerk Interface verwenden, kann man das Attribute enabled auf true setzen und im Element interface das gewünschte Netzwerk Interface angeben.

5.2. Multi-Node Konfigurationsempfehlung

Damit mehrere SecDocs Instanzen eine konsistente Multi-Node Konfiguration bilden, sollte man die Dateien secdocs.properties und hazelcast.xml auf einem Storage ablegen, der von allen SecDocs Instanzen aus erreichbar ist. In den einzelnen Instanzen kann man dann eine secdocs.properties Datei mit dem folgenden Inhalt verwenden (s. Datei secdocs.properties.multinode)

```
globalPropertiesFile=<full path>/secdocs.properties
```

Alle SecDocs Instanzen verwenden dann diese zentrale, sogenannte Secondary Properties Datei.

5.3. Hinweis zum ersten Start einer Multi-Node Konfiguration

Beim ersten Start nach einer SecDocs Installation sollte man zuerst nur eine Instanz der Multi-Node Konfiguration starten. Ist diese Instanz betriebsbereit, können die anderen Multi-Node Instanzen problemlos gestartet werden. Startet man in diesem Fall mehrere Multi-Node Instanzen gleichzeitig, kann es zu Abbrüchen führen, für die man im Logging Fehlermeldungen der Form:

```
ORA-01408: such column list already indexed
```

findet. Die fehlerhaft beendeten Instanzen kann man in diesem Fall jedoch auch problemlos neu starten.

6. SecDocs Logging

Alle Logginginformationen der SecDocs Anwendung werden im Verzeichnis:

```
/home/secdocs/jboss/secdocs/log
```

abgelegt.

- WildFly Consolemeldungen: console.log
  Alle WildFly Loggingausgaben werden auf dem Bildschirm ausgegeben. Diese Ausgaben werden in die Datei console.log umgeleitet.
- WildFly Loggingdatei server.log
  Alle WildFly und SecDocs Loggingausgaben werden in diese Datei geschrieben.

Die SecDocs WildFly Anwendung benutzt das WildFly Logging Framework für die Loggingausgaben. Die Loggingkonfiguration ist in der Datei

```
/home/secdocs/jboss/secdocs/configuration/standalone.xml
```

definiert und ist Teil des WildFly Logging Subsystems

```
<subsystem xmlns="urn:jboss:domain:logging:8.0">
...
</subsystem>
```

Mit Hilfe des Tools bin/logAdmin kann man in einer laufenden SecDocs WildFly Server Instanz jederzeit die Logging Konfiguration anpassen.

7. SecDocs Anwendung starten/beenden

Nach der RPM Installation ist der systemd Service secdocs.service eingerichtet. D.h. bei einem Neustart der Maschine wird die SecDocs Anwendung automatisch gestartet. Mit Hilfe dieses Service kann der Administrator (Kennung root) die SecDocs Anwendung auch einfach manuell starten und beenden.

SecDocs starten:

```
# systemctl start secdocs.service
```

SecDocs beenden:

```
# systemctl stop secdocs.service
```

SecDocs Service Status:

```
# systemctl status secdocs.service
```

Der SecDocs Administrator kann auch direkt die folgenden Aufrufe ausführen:

```
~/bin/secdocs start
~/bin/secdocs status
~/bin/secdocs stop
```

8. SecDocs: Weitere Konfigurationsschritte

Die weiteren Konfigurationsschritte sind im SecDocs Handbuch "Administration und Bedienung" beschrieben. Eine Übersicht aller erforderlichen Schritte findet man im Kapitel "Schritt-für-Schritt-Anleitungen".

9. SecDocs Migration

Es wird nur eine Migration von der Version SecDocs V3.1B00 auf die aktuelle Version SecDocs V3.2A00 unterstützt.

9.1. SecDocs Datenbank Migration

Wurde eine bereits vorhandene SecDocs Installation auf die aktuelle Version hochgerüstet, sind die SecDocs Datenbanktabellen noch auf den neuen Stand zu bringen. Ohne diese Migration ist die neue SecDocs Version in der Regel nicht ablauffähig.

Die Migration wird mit dem Script startMigration durchgeführt. Dieses Script befindet sich im Verzeichnis install/migration der SecDocs Installation:

```
$ cd /home/secdocs/install/migration
$ ./startMigration
```

9.2. SecDocs SFTP Server

Wenn der SecDocs SFTP Server zum ersten Mal gestartet wird, wird ein SSH Hostkey (RSA Key Pair) erzeugt, der in einer Datei gespeichert wird und ab diesem Zeitpunkt immer benutzt wird.

Ab der Version SecDocs V3.1B00 wurde der Ablageort und das Dateiformat für den SSH Hostkey, den der SFTP Server verwendet, verändert.

In älteren Versionen heißt diese Datei

```
~/.ssh/secdocs_host_key
```

Jetzt heißt diese Datei secdocs_host_key.pem und wird am folgenden Ort abgelegt

```
~/jboss/secdocs/configuration/secdocs/secdocs_host_key.pem
```

Beim Erzeugen eines neuen RSA Key Pair wird jetzt eine RSA Keysize von 4096 – statt wie bisher 2048 – verwendet.

Will man den Hostkey an einem anderen Ort oder unter einem anderen Namen ablegen, kann man in der Datei secdocs.properties die Property *sftp.hostkey.path* auf den gewünschten Namen (volle Pfadangabe) setzen.

Wird der Pfad gesetzt, ist darauf zu achten, dass das Verzeichnis, in dem die Datei abgelegt werden soll bereits existiert und das der Benutzer *secdocs* in diesem Verzeichnis Lese- und Schreibrechte hat. Ist eine dieser Anforderungen nicht erfüllt, wird SecDocs mit einer entsprechenden Fehlermeldung beendet.

Ist eine andere Keysize gewünscht, kann man die Property *sftp.hostkey.keysize* auf den gewünschten Wert setzen.
Hinweis: der angegebene Wert wird nur benutzt, wenn ein neues RSA Key Pair erzeugt wird, d.h. wenn die Datei Hostkey Datei noch nicht existiert oder leer ist.

Es sind nur die folgenden Werte als Keysize erlaubt

- 2048
- 4096
- 8192

Soll eine andere Keysize verwendet werden, kann man das erreichen, indem man selbst ein RSA Key Pair erzeugt.

Will man einen eigenen SSH Hostkey verwenden, muss die PEM Datei das folgende Format haben:

```
-----BEGIN RSA PRIVATE KEY-----
...
-----END RSA PRIVATE KEY-----
```

Dieses Format kann man mit Hilfe des Tools openssl erzeugen:

```
openssl genpkey -algorithm RSA -pkeyopt rsa_keygen_bits:4096 | \
    openssl rsa -out secdocs_host_key.pem
```

Alternativ kann man das Kommando genrsa benutzen:

```
openssl genrsa -out secdocs_host_key.pem 4096
```

Falls der SSH Public Key an die Clients zur Prüfung verteilt werden soll, kann man diesen folgendermaßen aus der SSH Hostkey Datei erzeugen:

```
ssh-keygen -y -f secdocs_host_key.pem >secdocs_host_key_ssh.pub
```

10. SecDocs Recovery Tool (recoverFromStorage)

Das Storage Recovery Tool befindet sich im Verzeichnis admin/recovery in der JAR-Datei StorageRecovery.jar. Mit Hilfe des Scripts recoverFromStorage lässt sich dieses Java Programm einfach starten.

```
$ recoverFromStorage <Optionen>
```

Das Storage Recovery Tool benötigt zum Ablauf eine Properties Datei. Als Template befindet sich im Verzeichnis admin/recovery die Datei recover.properties. In dieser Datei ist gegebenenfalls der folgende Eintrag anzupassen:

asPath=/home/secdocs/jboss/wildfly
(WildFly Home Verzeichnis)

Eine Beschreibung der Funktionsweise des SecDocs Storage Recovery Tools findet man im SecDocs Handbuch (Kapitel: Recovery (Skript: recoverFromStorage)).

11. SecDocs Purge Data Tool (purgeData)

Nach dem Löschen von Daten mit der Operation deleteSDO bleiben Dateien auf dem Storage und Einträge in der Datenbank übrig. Mit Hilfe des Purge Data Tools kann man diese Daten löschen.

Das Purge Data Tool befindet sich im Verzeichnis admin/purge in der JAR-Datei secdocsPurge.jar. Mit Hilfe des Scripts purgeData lässt sich dieses Java Programm einfach starten. Mit dem Aufruf

```
$ ./purgeData --help
```

erhält man eine Ausgabe aller Aufrufoptionen.

12. SecDocs Skripte (Benutzer: root/secdocs)

Die Skripte befinden sich im Verzeichnis bin der Kennung secdocs und sind aufrufbar für den Administrator (Benutzer root) und den Benutzer secdocs.

Die mit * markierten Skripte rufen Webdienste der laufenden SecDocs WildFly Anwendung mit Hilfe der Command Line Tools curl und wget auf. Neben den beschriebenen Rückgabewerten sind deshalb noch 2 weitere Antworten möglich:

1. Keine Antwort (keine Ausgabe)
   In diesem Fall läuft die SecDocs WildFly Anwendung nicht
2. HTML Quelltext Ausgabe.
   In diesem Fall ist die SecDocs Anwendung im WildFly nicht gestartet worden und der WildFly gibt eine HTML Fehlermeldung zurück.

Die meisten Skripte unterstützen die Option –h (oder auch --help), um Nutzungsinformationen auszugeben.

Die als Diagnosetool bezeichneten Skripte sind für den Service gedacht.

- checkAvailability [--silent]
  Prüft die Verfügbarkeit der SecDocs WildFly Services. Dieses Skript gibt, je nach Zustand, eine der folgenden Meldungen aus:

  SecDocs WildFly is not running
  Die SecDocs WildFly Server Instanz läuft nicht.
  SecDocs WildFly services not available
  Die SecDocs WildFly Services sind (noch) nicht verfügbar.
  SecDocs WildFly services available
  Die SecDocs WildFly Services sind verfügbar.

  Wird die Option --silent angegeben, erfolgt keine Textausgabe. In jedem Fall wird der Status durch den Exit Code des Skripts beschrieben:

  0: die Services sind verfügbar
  1: fehlerhafter Aufruf bzw. fehlerhafte Konfiguration
  2: die Services sind nicht verfügbar
  3: die Services sind (noch) nicht verfügbar
  4: die Anwendung läuft nicht
- clearCache
  Löscht den Cache der SecDocs WildFly Server Instanz.
- cli / cli.xml
  Dieses Skript startet das WildFly CLI Administrationstool für die SecDocs WildFly Server Instanz.

Dieses Tool wird in der Regel nur intern verwendet.
Ausnahme: die Konfiguratution eine HTTPS Connectors (s. unten).

- genArchivingSRWsClientStubs
  Dieses Script zeigt, welche Optionen man angeben muss, um mit Hilfe des Java SDK Tools wsimport aus der Datei schemas/3.2/ArchivingSR.wsdl die zugehörigen Web Service Client Stubs zu erzeugen. Lässt man dieses Script ablaufen, werden die Dateien wsStubsArchivingSR-3.2.jar und wsStubsArchivingSourcesSR-3.2.jar im Verzeichnis jaxws/lib erzeugt.
- getADOLockStatus *
  Dieses Skript ist ein Diagnosetool für den SecDocs Servicemitarbeiter.
- getDiagnosticData *
  Dieses Tool dient zum Sammeln von Diagnosedaten für den Service.
- getMultiNodeStatus *
  Ausgabe der laufenden SecDocs Multi Node Instanzen in dieser Multi-Node Konfiguration.
- getSecDocsConfigData *
  Dieses Script liefert für die Diagnose wichtige Informationen, die aus der laufenden SecDocs Anwendung abgerufen werden.
  Läuft die SecDocs Anwendung nicht, werden keine Daten ausgegeben.
- getStatus *
  Gibt aus, ob in der laufenden SecDocs Anwendung die SecDocs Web Services benutzbar sind

  SecDocs web services available

  oder nicht

  SecDocs web services NOT available

  Läuft die SecDocs Anwendung nicht, wird die Meldung

  getStatus: SecDocs WildFly is not running

  ausgegeben.
- getVersion *
  Gibt die aktuelle Version der laufenden SecDocs Anwendung aus.
  Läuft die SecDocs Anwendung nicht, wird die Meldung

  getVersion: SecDocs WildFly is not running

  ausgegeben.
- handleOutOfMemoryEvent
  Dieses Skript sollte nicht aufgerufen werden. Es wird intern im Skript secdocs genutzt, um die

SecDocs WildFly Anwendung im Fall einer OutOfMemoryError Situation automatisch zu beenden.

- heapdump <pid> [<hprof file name>]
  Diagnosetool: Mit Hilfe dieses Skripts kann man einen JVM Heapdump einer laufenden JVM erzeugen.

  Parameter:
  pid : Process ID der JVM
  hprof file name : Name der Heap Dump Datei
  (Default: jvm_<pid>.hprof)
- jhatRunner
  Diagnosetool: Skript zum Starten des Java SDK Tool jhat.
- jstatdRunner / jstatdRunner.policy
  Diagnosetool: Skript und Konfigurationsdatei zum Starten des Java SDK Tools jstatd.
- jtop
  Diagnosetool: startet die Java Console mit dem JTop Plugin.
- logAdmin
  Diagnosetool: Mit diesem Skript kann man Logger und ihr Logging Level in der laufenden SecDocs WildFly Server Instanz konfigurieren.
- mksha
  Erzeugt einen Hashwert (Default: SHA-256) für eine Datei und gibt diesen Hashwert als Hexstring und im BASE64 Format aus.

  Beispielausgabe:

```
SHA-256 hex value
bf5853dd535fc3d9889b98a0d01eb0256b95ebc27b99d0fb6d6a9591fe2191d4

SHA-256 value in BASE64
v1hT3VNfw9mIm5ig0B6wJWuV68J7mdD7bWqVkf4hkdQ=
```

- olcmStatus
  Ist der DSEngine Server auf demselben Rechner installiert, lässt sich mit diesem Script überprüfen, ob der Server läuft.
  Dieses Skript ruft intern das Skript CryptoModule/bin/cmStatus auf.
- removeLogs
  Löschen der SecDocs WildFly Loggingdateien.
- sdjcmd
  Diagnosetool: Ruft das Java SDK Tool jcmd für den laufenden SecDocs WildFly Java Prozess auf.
- sdjconsole
  Diagnosetool: Startet das Java SDK Tool jconsole.
- sdjinfo

Diagnosetool_ Startet das Java SDK Tool jinfo.

- sdsyslog
  Hilfsskript zur Erzeugung von syslog Meldungen in SecDocs Skripte.
- secdocs
  Identische Funktionalität und Aufrufparameter wie beim Service secdocs
  Darüber hinaus kann man auch noch eine Reihe weiterer Optionen nutzen:

  secdocs start
  Start der SecDocs WildFly Anwendung

  secdocs startDebug
  Diagnose: Start der SecDocs WildFly Anwendung im Debugging Mode

  secdocs startJProfiler
  Diagnose: Start der SecDocs WildFly Anwendung mit Hilfe eines Java Profilers.

  secdocs stop (oder auch secdocs shut)
  Beenden der SecDocs WildFly Anwendung

  secdocs stop force (oder auch secdocs shut force)
  Forciertes beenden der SecDocs WildFly Anwendung. D.h.: wenn die Anwendung sich trotz stop Kommando nicht beendet, kann man mit diesem Aufruf eine Beendigung erzwingen.
  Hinweis: vor einem Neustart, nach diesem Kommando, sollte man das Skript clearCache aufrufen.

  secdocs restart
  Restart (also stoppen und neu starten) der SecDocs WildFly Anwendung.

  secdocs status
  Prüft, ob der SecDocs WildFly Java Prozess gestartet wurde:

  SUCCESS: secdocs: SecDocs WildFly is running
  SUCCESS: secdocs: pid: 3076 , HTTP port: 8080

  Wichtig: dieses Kommando prüft nur, ob der SecDocs WildFly Java Prozess läuft. Mit Hilfe des Skript getStatus kann man prüfen, ob der Prozess auch (bereits) arbeitsfähig ist.

  Falls der Prozess nicht läuft,gibt es die folgende Ausgabe:

SUCCESS: secdocs: SecDocs WildFly not running

secdocs pid
Gibt die PID des laufenden SecDocs WildFly JVM Prozesses aus. Läuft die Anwendung nicht, gibt es keine Ausgabe und das Skript beendet sich mit dem Exit Code 1.

secdocs jstack
Diagnosetool: Erzeugt für den laufenden SecDocs WildFly JVM Prozess einen Java Thread Dump in der Datei /home/secdocs/jstack.secdocs_<pid>.txt

secdocs edit
Dieses und die folgenden edit Kommandos öffnet eine Datei in einem Editor. Default ist der vi. Man kann in der Environmentvariablen VISUAL den Pfad zu einem anderen Editor setzen, z.B.:

export VISUAL="/usr/bin/emacs"

Als Standard wird die SecDocs Konfigurationsdatei geöffnet:

/home/secdocs/jboss/secdocs/configuration/secdocs/secdocs.properties

secdocs edit server (secdocs edit wildfly)
Es wird die WildFly Konfigurationsdatei geöffnet:
/home/secdocs/jboss/secdocs/configuration/standalone.xml

secdocs edit hazelcast (secdocs edit hz)
Es wird die Hazelcast (= MultiNode Kopnfiguration) Konfigurationsdatei geöffnet:
/home/secdocs/jboss/secdocs/configuration/secdocs/hazelcast.xml

secdocs log
Dieses und die folgenden log Kommandos öffnet eine Datei in einem Editor. Default ist der vi. Man kann in der Environmentvariablen VISUAL den Pfad zu einem anderen Editor setzen, z.B.:

export VISUAL="/usr/bin/emacs"

Als Standard wird die SecDocs WildFly Server Loggingdatei geöffnet:
/home/secdocs/jboss/secdocs/log/server.log

secdocs log tail
Es wird die SecDocs WildFly Server Loggingdatei mit dem Kommando tail ausgegeben:
tail –f /home/secdocs/jboss/secdocs/log/server.log

secdocs log console
Es wird die SecDocs WildFly Server Loggingdatei für die Console Ausgaben (= Meldungen auf stdout/stderr) geöffnet:
/home/secdocs/jboss/secdocs/log/console.log

secdocs log server
Es wird die SecDocs WildFly Server Loggingdatei geöffnet:
/home/secdocs/jboss/secdocs/log/server.log

secdocs uptime
Bei einer laufenden Anwendung wird angezeigt, seit wann die Anwendung läuft.
Beispiel:
Mon Feb 10 14:21:50 2020

2414389.857 seconds

027:22:39:49.857 (days:hours:minutes:seconds:milliseconds)

secdocs version
Diagnosetool: Gibt die verwendeten Versionen von WildFly und OpenJDK8 aus.

secdocs env
Diagnosetool: Gibt eine Liste aller zur Laufzeit gesetzten Environmentvariablen aus.

secdocs fdCount
Diagnosetool: Gibt die Anzahl der offenen Filedeskriptoren aus. Läuft die Anwendung nicht, wird 0 ausgegeben.

setSecDocsEnv.sh
In dieser Datei werden die von allen Script benötigten Variablen gesetzt.

sysinfo
Diagnosetool: Dieses Script liefert für die Diagnose wichtige Informationen mit Konfigurationsdaten des Rechners.

13. SecDocs Tuning

Im Folgenden werden einige Parameter der SecDocs WildFly Konfiguration beschrieben, die, je nach Bedarf, angepasst werden können.

13.1. Maximale Anzahl paralleler Web Service Requests

Die maximale Anzahl paralleler Web Service Requests wird über das Attribut
task-max-threads in der Datei

/home/secdocs/jboss/secdocs/configuration/standalone.xml

gesteuert:

```
<subsystem xmlns="urn:jboss:domain:io:3.0">
<worker name="default" *task-max-threads="70"*/>
<buffer-pool name="default"/>
</subsystem>
```

Will man mehr oder auch weniger parallele Request unterstützen, ist dieses Attribut entsprechend anzupassen. Nach der Installation ist dieser Wert auf 70 gesetzt. Ändert man diesen Wert, muss man die maximale Pool Size des Datenbank Verbindungspools (s. „Datenbank Verbindungspool“) entsprechend anpassen.

Die Änderung kann auch bei einer laufenden SecDocs Anwendung erfolgen.

Bestimmen des aktuellen Wertes:

```
/home/secdocs/bin/cli \
--command="/subsystem=io/worker=default/:read-attribute(name=task-max-threads)"
```

Setzen eines neuen Wertes (im Beispiel auf 50):

```
/home/secdocs/bin/cli \
--command="/subsystem=io/worker=default/:write-attribute(name=task-max-threads,value=50)"
```

Damit der geänderte Wert gültig wird, muss die SecDocs Anwendung beendet und anschließend neu gestartet werden.

13.2. Vom SecDocs WildFly genutzten Speicher erhöhen

Beim Starten des SecDocs WildFly Anwendung wird der zugrundeliegenden JVM ein maximal nutzbarer Speicher zugewiesen.

```
JAVA_MEM_MX=-Xmx4g
```

Dieser Wert kann in einer Produktionsumgebung zu klein sein. Ist genug RAM in der Maschine vorhanden, kann man den Standardwert (= 4 GB) erhöhen. Die obige Zeile findet man in der Skriptdatei */home/secdocs/bin/setSecDocsEnv.sh*.

Beispiele für mögliche Speicherengpässe bei der Standardkonfiguration:

- Paralleles Speichern von sehr großen bzw. sehr vielen SDOs.
- Paralleles Verarbeiten von SDOs mit sehr vielen Signaturen

13.3. Transaction Timeout

In der Datei /home/secdocs/jboss/secdocs/configuration/standalone.xml steht die folgende Zeile:

```
<coordinator-environment default-timeout="1800"/>
```

Dies bedeutet, dass eine Transaktion maximal **1800** Sekunden dauern kann. Dieser Wert ist bei größeren Datenmengen (= große SDOs und/oder viele Signaturen in einem SDO) gegebenenfalls anzupassen. Diese Anpassung kann auch bei einer laufenden SecDocs Anwendung erfolgen:

Bestimmen des aktuellen Timeout Wertes

```
/home/secdocs/bin/cli \
--command="/subsystem=transactions/:read-attribute(name=default-timeout)"
```

Verändern des Timeout Wertes. Im Beispiel wird der Wert auf 2000 Sekunden gesetzt:

```
/home/secdocs/bin/cli \
--command="/subsystem=transactions/:write-attribute(name=default-timeout,value=2000"
```

Damit der geänderte Wert gültig wird, muss die SecDocs Anwendung beendet und anschließend neu gestartet werden.

13.4. Datenbank Verbindungspool

Die Datenbankverbindungen werden vom WildFly Application Server in einem Verbindungspool verwaltet. In der Datei

/home/secdocs/jboss/secdocs/configuration/standalone.xml

steht **zweimal** (2 Datasources!) die folgende Zeile:

```
<max-pool-size>75</max-pool-size>
```

Dies bedeutet, dass jeweils maximal **75** (also insgesamt **150**!) Verbindungen zur Datenbank aufgebaut werden können. Dieser Wert kann, je nach Bedarf, verkleinert oder aber auch vergrößert werden.

Der aktuelle Wert lässt sich auch zur Laufzeit von SecDocs ermitteln bzw. ändern.

Bestimmen der aktuellen Pool Größe:

```
/home/secdocs/bin/cli \
--command="/subsystem=datasources/data-source=ArchiveDS/:read-attribute(name=max-pool-size)"

/home/secdocs/bin/cli \
--command="/subsystem=datasources/xa-data-source=ArchiveDSXA/:read-attribute(name=max-pool-size)"
```

Verändern der Pool Größe. Im Beispiel wird der Wert auf **100** gesetzt.

```
/home/secdocs/bin/cli \
--command="/subsystem=datasources/data-source=ArchiveDS/:write-attribute(name=max-pool-size,value=100)"

/home/secdocs/bin/cli \
--command="/subsystem=datasources/xa-data-source=ArchiveDSXA/:write-attribute(name=max-pool-size,value=100)"
```

Damit der geänderte Wert gültig wird, muss die SecDocs Anwendung beendet und anschließend neu gestartet werden.

Achtung: in einer Multi-Node SecDocs Konfiguration ist die Zahl der Verbindungen noch mit der Anzahl der verwendeten SecDocs Instanzen zu multiplizieren.

Im folgenden Beispiel für Oracle beziehen sich die Daten auf eine SecDocs Single-Node Konfiguration.

13.4.1. Oracle

Achtung: es kann sein, dass die maximale Anzahl der Oracle Datenbank Prozesse entsprechend angepasst werden muss, denn jede Verbindung zur Datenbank benötigt einen Oracle Prozess.

Es kann deshalb sein, dass der Standardwert für die maximal parallel laufenden Oracle Prozesse zu klein ist. Den aktuellen Wert kann man sich als Datenbankadministrator mit dem folgenden Kommando im Oracle Tool sqlplus anzeigen lassen:

```
show parameter processes;
```

Neben weiteren Konfigurationsparametern der Datenbank erhält man eine Ausgabe der folgenden Form:

```
NAME      TYPE    VALUE
processes integer 150
```

Diesen Wert kann man mit den folgenden SQL Kommandos als Datenbankadministrator verändern (hier wird der Wert auf 250 gesetzt):

```
shutdown immediate;
startup mount;
alter system set processes=250 scope=spfile;
alter database open;
shutdown immediate;
startup;
show parameter processes;
```

13.5. HTTPS Connector Konfiguration (Port: 8443)

Die SecDocs WildFly Server Instanz wird ohne HTTPS Connector Konfiguration ausgeliefert. Eine entsprechende Konfiguration mit Server Zertifikat lässt sich leicht mit Hilfe des WildFly CLI Tools erzeugen. Die hier gezeigten Aufrufe findet man in der Datei install/wildfly/add_https.cli.

```
batch
#
/core-service=management/security-realm=SSLRealm:add()
#
/core-service=management/security-realm=SSLRealm/server-identity=ssl:add(keystore-path="SecDocsServerKeyStore.jks",keystore-
relative-to=jboss.server.config.dir,keystore-password="changeit")
#
/core-service=management/security-realm=SSLRealm/server-identity=ssl:write-attribute(name="protocol",value="TLS")
#
/subsystem=undertow/server=default-server/https-listener=https:add(socket-binding="https",max-post-size=157286400,security-
realm="SSLRealm",enabled-protocols="TLSv1.2",enabled-protocols="TLSv1.2",enabled-cipher-suites="ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-
RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384",verify-client="NOT_REQUESTED")
#
run-batch
```

Damit die Änderung gültig wird, muss die SecDocs Anwendung beendet und anschließend neu gestartet werden.

Im obigen Beispiel wird die Datei **SecDocsServerKeyStore.jks** als Keystore Datei mit dem Passwort **changeit** verwendet. Diese Datei ist unter /home/secdocs/jboss/secdocs/configuration abgelegt. Ein leerer Keystore mit dem Namen SecDocsServerKeyStore.jks wird mit ausgeliefert. Als Backup findet man unter install/wildfly eine Kopie dieser Datei.

Achtung: derzeit kann man im WildFly nur Java Keystores (Format: JKS) verwenden. Typischerweise werden Server Zertifikate jedoch als Dateien im Format PKCS12 ausgeliefert. Diese Dateien kann man mit Hilfe des Java SDK Tool keytool in eine Datei im Format JKS umwandeln (Option –importkeystore).

13.6. HTTP/HTTPS Connector: max-post-size

In der Konfiguration der HTTP/HTTPS Connectoren wird der Parameter *max-post-size* verwendet. Dieser Parameter legt fest, wie viele Bytes maximal mit einem POST Request (= SecDocs Web Service Operation Request Message) gesendet werden können. Als Default wird der folgende Wert verwendet:

max-post-size="1073741824"

D.h.: die maximale Größe eines POST Requests ist auf 1GB festgelegt.

13.7. Maximale Anzahl offener Dateien

Pro Mandant wird eine Audit Log Datei während des gesamten Laufs der SecDocs Anwendung offen gehalten. Desweiteren werden bei den meisten Aktionen (z.B. Dokument speichern, Dokument lesen, Dokument versiegeln) ebenfalls Dateien kurzzeitig geöffnet.

Da im Linux Kernel eine maximale Anzahl von offenen Dateien konfiguriert ist, kann es in einer Produktivumgebung durchaus sein, dass die Anzahl der gleichzeitig geöffneten Dateien diesen Wert überschreitet. Sollte dies der Fall sein, kann der Systemadministrator (Benutzer root) den Kernelwert erhöhen:

1. Schritt: den aktuellen Wert bestimmen
   # sysctl fs.file-max
   Dieses Kommando gibt den aktuellen Wert es Kernelparameters aus.
2. Höheren Wert für den Kernel konfigurieren:
   Hierzu trägt man in der Datei
   /etc/sysctl.conf
   den gewünschten Wert in der Form
   fs.file-max = <Anzahl maximal zu öffnender Dateien>
   ein.
   Beispiel:
   fs.file-max = 10000000
3. Entweder Reboot oder den neuen Wert sofort im System aktivieren:
   Um den neuen Kernelparameterwert für alle neuen Prozesse sofort zu aktivieren, gibt man das folgende Kommando ein:
   # sysctl –p
4. Neuen Wert des Kernelparameter überprüfen:
   # sysctl fs.file-max

Der Wert für die maximale Anzahl offener Datei ist außerdem von der Einstellung für den jeweiligen Benutzer abhängig.

Im 2. Schritt ist deshalb zu prüfen, ob die Einstellungen für den Benutzer secdocs anzupassen sind. Hierzu kann man in der Kennung secdocs das Kommando *ulimit* verwenden:

Ausgabe der Softlimits für die Anzahl der offenen Dateien des Benutzers:

```
$ ulimit -Sn
1024
```

Ausgabe der Hardlimits für die Anzahl der offenen Dateien des Benutzers:

```
$ ulimit -Hn
524288
```

Ausgabe des aktuell gesetzen Wertes für die Anzahl der offenen Dateien des Benutzers:

```
$ ulimit -n
1024
```

Im obigen Beispiel kann der Benutzer secdocs maximal 1024 Dateien öffnen. Mit dem Kommando

```
$ ulimit -n <Anzahl offener Dateien>
```

kann man den eingestellten Wert verändern. Maximal kann der Wert auf das Hardlimit (im Beispiel 524288) gesetzt werden.

```
$ ulimit -n 524288
```

Damit dieser Wert gilt, muss er bei jedem Login gesetzt werden. Außerdem kann der Wert, trotz größerem Kernel Parameterwert (im Beispiel fs.file-max = 10000000) nicht über das Hardlimit hinaus vergrößert werden.

Deshalb sollten die gewünschten Werte als Administrator in der Datei */etc/security/limits.conf* eingetragen werden. Im Beispiel wird die maximale ANzahl offener Datei auf 1000000 gesetzt:

```
@secdocs        soft    nofile          1000000
@secdocs        hard    nofile          1000000
```

Nach einem Reboot sieht man als Benutzer secdocs jetzt die folgenden ulimit Werte:

```
$ ulimit -n
1000000
$ ulimit -Sn
1000000
$ ulimit -Hn
1000000
```

13.8. Maximale Größe einer Audit Logdatei

Die maximale Größe einer Audit Logdatei wird durch die Property maxAuditLogFileSize in der Datei secdocs.properties gesteuert. Die gewählte Größe ist besonders dann von Interesse, wenn man zusätzlich die Property

createAuditLogEvidenceRecord=true

setzt.

13.9. ETSI (Validierungs-) Report deaktivieren

Ab der Version 1.1.9 des DSEngines wird bei der Validierung von Signaturdaten zusätzlich im Validierungsreport ein ETSI Report miterzeugt.

Das hat 2 Konsequenzen:

1. Die Validierungsreports werden größer
2. Die Performance ist etwas geringer.

Die Erzeugung des ETSI Reports kann man im DSEngine (CryptoModule) global abschalten (s. Kapitel "ETSI (Validierungs-) Report deaktivieren" im Handbuch "Fujitsu DSEngine 2.0.0 Runtime Umgebung 1 für Fujitsu SecDocs V3.2").

14. SecDocs XAIP (TR-ESOR 1.2.1)

SecDocs V3.2 unterstützt die BSI TR-ESOR 1.2.1 (BSI TR-03125: Beweiswerterhaltung kryptographisch signierter Dokumente) Technische Richtlinie im Level C1 (TR-ESOR-C.1).

Für einen TR-ESOR konformen Betrieb sind die folgenden Punkte zu beachten und entsprechend zu konfigurieren:

1. SecDocs muss in einem TR-ESOR konformen Betrieb verwendet werden.
2. Überprüfen der Timestamp Erzeugungszeit
3. Für die XAIP (TR-ESOR) Funktionalität in SecDocs wird zwingend eine HTTPS Connector mit Client- und Serverzertifikaten benötigt.

14.1. TR-ESOR konformer Betrieb

Damit SecDocs in einem TR-ESOR konformen Betrieb verwendet werden kann, muss in der Datei *~/jboss/secdocs/configuration/secdocs/secdocs.properties* die Property *tresor.certified* auf *true* gesetzt werden:

```
tresor.certified=true
```

Ruft man das Skript *getVersion* auf, sieht man dann in der Versionsausgabe den Zusatz

```
[TR-ESOR C1 Certified Mode is enabled]
```

In der Datei *~/jboss/secdocs/log/server.log* sieht man den Logsatz

```
2021-09-27 09:17:23,083 INFO  [SecDocs] [host:sles15Manfred] [ServerService Thread Pool -- 67]: SecDocs is running in TR-ESOR C1
certified mode
```

14.1.1. Einschränkungen gegenüber dem Normalbetrieb

Die folgenden Web Service Operationen sind **nicht** nutzbar:

- Archiving Web Service Operation replaceSDO
- Archiving Web Service Operation forceDeferredSDO
- S4 (TR-ESOR) Web Service Operation ArchiveData

Zusätzlich wird die Property *doSecDocsHash* auf *false* gesetzt, d.h. alle Hashwerte werden **immer** im CryptoModule berechnet. Die Property hat zur Laufzeit auch dann den Wert *false*, wenn man sie in der Datei *secdocs.properties* auf *true* gesetzt hat.

14.2. Überprüfen der Timestamp Erzeugungszeit

Um Replay-Attacken zu verhindern, wird ab SecDocs V3.1B00 die Erzeugungszeit des Timestamps mit der aktuellen Maschinenzeit verglichen. Ist die Abweichung zu groß, wird der mit diesem Timestamp erzeugte Evidence Record verworfen und eine Fehlermeldung im Serverlog (*~/jboss/secdocs/log/server.log*) protokolliert.

Dieses Verhalten wird über die beiden folgenden Properties in der Datei *~/jboss/secdocs/configuration/secdocs/secdocs.properties* gesteuert:

```
# default: true
#checkTspProductionTime=true
# max time delta in seconds
# default: 120, minimum: 1
#checkTspProductionTimeDelta=120
```

D.h.: standardmäßig wird die Erzeugungszeit des Timestamps überprüft und darf um maximal 2 Minuten (= 120 Sekunden) von der aktuellen Maschinenzeit abweichen.

Über die Property *checkTspProductionTimeDelta* kann man die maximal erlaubte Zeitdifferenz verändern. Der minimale Wert ist 1 (= 1 Sekunde).

Außerhalb des TR-ESOR-konformen Betriebs kann die Property *checkTspProductionTime* auch auf *false* gesetzt werden. In diesem Fall findet keine Überprüfung der Erzeugungszeit des Zeitstempels statt.

Der Betreiber muss sicherstellen, dass die Systemzeit der Rechner auf denen SecDocs läuft, mit einer vertrauensvollen Zeitquelle synchronisiert ist (z.B. durch eine Funkuhr oder durch Verbindung mit einem NTP-Server).

14.3. HTTPS Connector Konfiguration (Port 8444)

Im folgenden Beispiel wird ein HTTPS Connector mit Client- und Serverzertifikaten im SecDocs WildFly Server mit Hilfe des WildFly CLI Tools eingerichtet. Die hier gezeigten Aufrufe findet man in der Datei *~/install/wildfly/add_https_cs.cli*.

```
batch
#
/core-service=management/security-realm=SSLRealmCS:add()
#
/core-service=management/security-realm=SSLRealmCS/server-identity=ssl:add(keystore-path="SecDocsServerKeyStore.jks",keystore-
relative-to=jboss.server.config.dir,keystore-password="changeit")
#
/core-service=management/security-realm=SSLRealmCS/server-identity=ssl:write-attribute(name="protocol",value="TLS")
#
/core-service=management/security-realm=SSLRealmCS/authentication=truststore:add(keystore-
path="*SecDocsServerTrustStore.jks*",keystore-relative-to=jboss.server.config.dir,keystore-password="*changeit*")
#
/subsystem=undertow/server=default-server/https-listener=https_cs:add(socket-binding="https_cs",max-post-size=157286400,security-
realm="SSLRealmCS",enabled-protocols="TLSv1.2",enabled-protocols="TLSv1.2",enabled-cipher-suites="ECDHE-ECDSA-AES128-GCM-
SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-
RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384",verify-client="REQUIRED")
#
run-batch
```

Nach dem Ausführen der obigen Anweisungen muss der SecDocs WildFly Server beendet werden.

Im obigen Beispiel muss der Java Key Store mit den Server Daten (Zertifikat (= Public Key) und Private Key) in der Datei

```
/home/secdocs/jboss/secdocs/configuration/SecDocsServerKeyStore.jks
```

abgelegt werden. Ein leerer Keystore mit dem Namen SecDocsServerKeyStore.jks wird mit ausgeliefert. Als Backup findet man unter install/wildfly eine Kopie dieser Datei. Hat der Java Key Store ein anderen Namen oder ein anderes Passwort, muss das obige Beispiel entsprechend angepasst werden.

Die Clientzertifikate müssen in die Datei

```
/home/secdocs/jboss/secdocs/configuration/SecDocsServerTrustStore.jks
```

importiert werden. Ein leerer Keystore mit dem Namen SecDocsServerTrustStore.jks wird mit ausgeliefert. Als Backup findet man unter install/wildfly eine Kopie dieser Datei. Hat der Java Key Store ein anderen Namen oder ein anderes Passwort, muss das obige Beispiel entsprechend angepasst werden.

In der Regel sind die in einer Organisation verwendeten Clientzertifikate alle von derselben CA signiert. In diesem Fall reicht es aus, das Zertifikat dieser CA in den Java Key Store zu importieren:

```
keytool -importcert -trustcacerts \
        -keystore SecDocsServerTrustStore.jks \
        -storepass changeit \
        -noprompt \
        -file clientCA.crt \
        -alias secdocsclientca \
        -v
```

Danach muss die SecDocs WildFly Server Instanz beendet und neu gestartet werden. Alle von dieser CA signierten Clientzertifikate werden dann automatisch vom SecDocs WildFly Server akzeptiert.

Gibt es mehrere solche CAs, so ist der Import für jedes CA Zertifikat durchzuführen. Es ist dabei darauf zu achten, dass der Aliasname (Wert des Parameters –alias) bei jedem Import ein eindeutiger Name sein muss.

14.3.1. Reverse Proxy Konfiguration (SecDocs Multi-Node)

In einer SecDocs Multi-Node Konfiguration wird in der Regel ein Load Balancer für den Zugriff auf die einzelnen SecDocs Instanzen verwendet.

Dabei ergibt sich das Problem, dass das in SecDocs benötigte Client Zertifikat im Load Balancer konfiguriert werden muss und nicht auf dem Transport Protokoll Level (TLS) an den SecDocs WildFly Server weitergereicht werden kann.

Für solche Situationen gibt es die Möglichkeit, das Client Zertifikat in einem HTTP Header Element zu

übertragen. Um das zu ermöglichen, sind 2 Änderungen in der SecDocs WildFly Konfiguration nötig.

Anforderung eines Client Zertifikats im WildFly abschalten

Für den Single-Node Einsatz ist der HTTPS Connector (s. Beispiel in der Datei *~/install/wildfly/add_https_cs.cli*) so konfiguriert, dass eine Verbindung nur möglich ist, wenn ein Client Zertifikat gesendet wird. Damit die Übertragung in einem HTTP Header Element möglich ist, muss das HTTPS Connector Attribut *verify-client* von

```
verify-client="REQUIRED"
```

auf

```
verify-client="REQUESTED"
```

abgeändert werden.

Reverse Proxy Konfiguration aktivieren

Damit nicht jeder Request irgendein beliebiges Zertifikat benutzen kann, muss die Reverse Proxy Nutzung in SecDocs aktiviert werden. Dazu sind in der Datei *secdocs.properties* die folgenden Properties zu setzen:

```
reverseProxy=true
reverseProxy.clientCertificateHeaderVariable=X-SSL-CLIENT-CERT
reverseProxy.address.1=nnn:nnn:nnn:nnn
#
# Gibt es mehre Reverse Proxy Adressen müssen hier
# alle angegeben werden.
# Mindestens eine Adresse muss angegeben werden, da sonst
# die Reverse Proxy Funktionalität abgeschaltet wird.
# reverseProxy.address.2=nnn:nnn:nnn:nnn
# ...
# reverseProxy.address.10=nnn:nnn:nnn:nnn
```

- reverseProxy
  Nur wenn diese Property auf true gesetzt wird, werden die weiteren Reverse Proxy Properties gelesen.
  Default: false
- reverseProxy.clientCertificateHeaderVariable
  Hier wird der Name des zu verwendenden HTTP Header Elements angegeben
  Default: X-SSL-CLIENT-CERT
- reverseProxy.address.1, ..., reverseProxy.address.10
  Hier können bis zu 10 Adressen hinterlegt werden.
  Da es sich um verschiedene Properties handelt (jede hat ihren eigenen Name), muss pro Zeile eine solche Property gesetzt werden.
  Nur wenn ein Request von einer dieser Adressen kommt, wird das HTTP Header Element gelesen.

Hinweis: die Nummern (1,...,10) müssen aufsteigend und lückenlos angegeben werden. D.h.: werden die Properties reverseProxy.address.1 und reverseProxy.address.7 verwendet, wird nur die Property reverseProxy.address.1 bewertet.

Durch diese Konfiguration wird der Reverse Proxy zu einer vertrauenswürdigen Instanz, d.h.: es wird in SecDocs angenommen, dass das Zertifikat im HTTP Header Element wirklich zurecht verwendet wird.

Damit das funktioniert, muss das zu verwendende Client Zertifikat im Reverse Proxy konfiguriert sein und der Reverse Proxy muss sicherstellen, dass nur dieses Zertifikat im HTTP Header Element an SecDocs gesendet wird.

Suchreihenfolge für die Client Zertifikate

Ist die Property reverseProxy auf false gesetzt, werden nur Zertifikate verwendet, die im Sever via TLS übergeben werden.

Ist die Property reverseProxy auf true gesetzt, gibt es 2 Fälle:

1. Ist die Client Adresse nicht in den Properties reverseProxy.address... hinterlegt, werden nur Zertifikate verwendet, die im Sever via TLS übergeben werden.
2. Ist die Client Adresse in den Properties reverseProxy.address... hinterlegt, werden nur Zertifikate verwendet, die im HTTP Header übergeben wurden.

14.4. XAIP Meta Data Store

Eine Indizierung ist nicht Teil der BSI TR-ESOR 1.2 Spezifikation und deshalb in SecDocs nicht implementiert. Als Alternative wird eine Schnittstelle bereitgestellt, mit der man für die Indizierung einen externen Dienst (SecDocs Meta Data Store) beauftragen kann. Die Beschreibung hierzu findet man im Dokument „XAIP Metadaten Indizierung in SecDocs“ (*xaipMetaDataIndizierung.pdf*).

15. INFO Meldungen im Server Logging

Derzeit findet man im Server Logging (Datei */home/secdocs/jboss/secdocs/log/server.log*) die folgenden Einträge mit dem Log Level INFO:

```
[com.fujitsu.ts.dsengine.client.operations.soap.WsProviderCXF]:
Failed to get CXF provider implementation. Cause:
ClassNotFoundException: org.apache.cxf.jaxws.spi.ProviderImpl from
[Module "deployment.archiver.ear" from Service Module Loader]
```

und

```
[com.fujitsu.ts.dsengine.client.operations.soap.WsProviderCXF]:
Using service delegate org.jboss.wsf.stack.cxf.client.ProviderImpl$JBossWSServiceImpl
```

Diese Einträge können vom Anwender ignoriert werden.

16. Rücksetzen der SecDocs Umgebung

In einer Testumgebung kann es nötig sein, dass die angelegten Archivdaten gelöscht werden müssen, um mit neuen Tests ohne Altlasten beginnen zu können.

Folgende Daten sind zu löschen.

16.1. Datenbank

Alle Tabellen des verwendeten Benutzers. Entweder "DROP TABLE *tablename*;" für alle Tabellen aufrufen oder einfach den Datenbankbenutzer löschen und wieder neu anlegen

16.2. Filesystem

In den in den Properties

- archiveRoot
- archiveRootExternalFiles

festgelegten Verzeichnissen müssen alle Verzeichnisse/Dateien gelöscht werden.

16.3. Mountpunkte

Wurden mandantenspezifische Mountpunkte angelegt, so sollte man die zugehörigen Verzeichnisse nur löschen, wenn man diese Mountpunkte in der neuen Testumgebung nicht mehr verwenden will. Den Inhalt dieser Verzeichnisse muss man jedoch auf jeden Fall löschen.