A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

Secure-FS

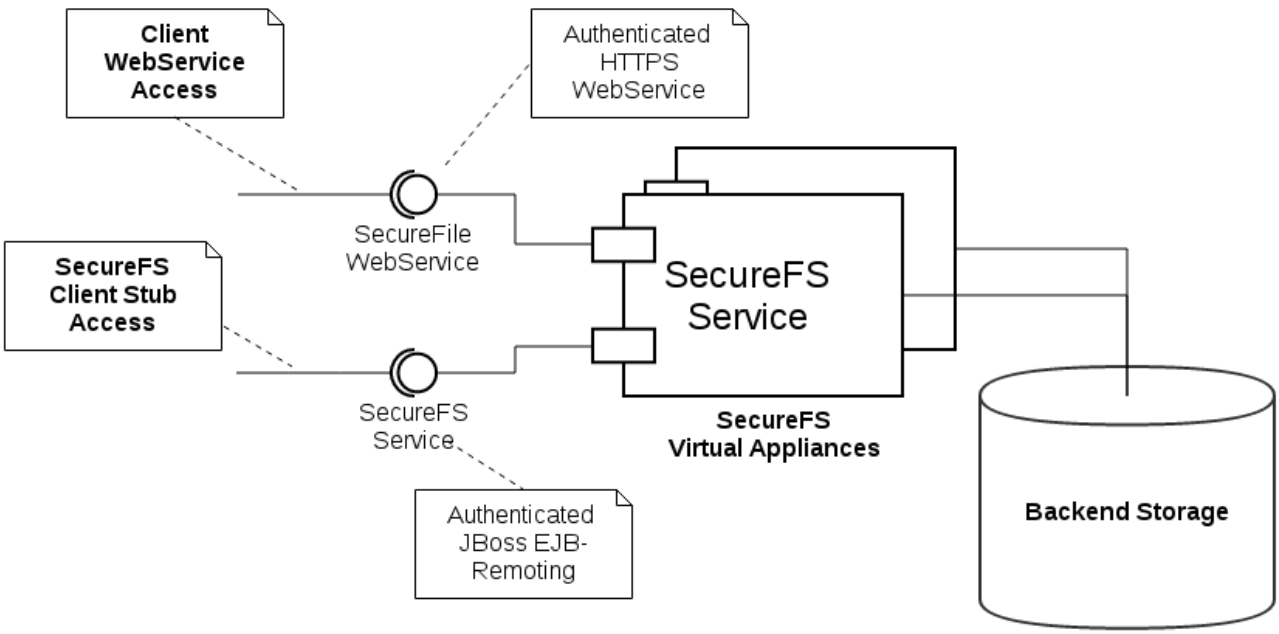
Virtuelle Appliance zur verschlüsselten Dateiablage

Inhaltsverzeichnis

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

Autor	Anmerkung	Datum	Status
Thomas Frühbeck	Arbeitsversion Erstausgabe	03.11.2015	DRAFT

ıd zuletzt geändert von Frühbeck Thomas vor 5 Minuten



A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

1 Einleitung

Dieses Dokument beschreibt die Funktion einer virtuellen Appliance, die medizinische und andere sensible Dokumente verschlüsselt in einem Filesystem ablegt und zum Abruf verfügbar macht.

Neben den HW-Varianten mit Safenet/Cryptas stellt diese Variante technsich äquivalente sichere Verschlüsselung bereit, jedoch nicht mit einer HW Variante, sondern mit einer Java-SW-Variante einer "virtuellen Appliance", die entweder von A1 intern oder von tiani entwickelt werden könnte.

Es gibt die notwendigen Libraries in einer Open-Source-Lizenz (Apache/Eclipse-License). Libs wurden positiv funktionsgetestet.

1.1 Ziele:

- Es werden sensible Gesundheitsdatendokumente verschlüsselt in einem Standard-Filesystem abgelegt
- Virtuelle Appliance mit einer NW-Schnittstelle für die Ver-/Entschlüsselung von Dateien
- Eingabeparameter sind: Schlüssel gemäß 4-Augen Prinzip zum Start der Appliance
- Applikation erhält über die Schnittstelle Datein übergeben und legt sie auf einer Standard Storage verschlüsselt ab und stellt die Dateien unverschlüsselt an derselben Schnittstelle zur Verfügung
- Schlüssel-Management erst nach einem Prototyp aufsetzen
- Client-Library, die über Standard-Java IO auf die Dateien zugriff ermöglicht
- Client bindet sich an über SSL (certificate based authentification)
- Soll laufen auf 2 virt. Maschines, machen autom. Fail Over durch FW/LB. Vorschlag/Empfehlung: Jboss
- Aufgabe schreiben-verschlüsseln und wieder lesen-entschlüsseln.

1.2 Nicht-Ziele:

Die Virtuelle Appliance kann Sicherheit der Schlüssel nicht garantieren!

Die Sicherheit der Betriebssytem der Virtuellen Appliance Instanz muss außerhalb des Konzepts hergestellt werden.

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

1.3 Mengengerüst

Bandbreite e-card und ELGA

Vertraglich steht dem e-card System im SV-Kanal folgende Mindestbandbreite zur Verfügung:

- 256Kbit DL (Download) / 128Kbit UL (Upload)
- Durchschnittliche Datei-Größe: ca. 10KB

CDA = Clinical Document Architecture (XML-Dokumenttypen)

Durchschnittliche Größe eines CDA

- Dokumentanzahl gesamt: 337.343
- Durchschnittliche Größe der XML-Dateien: 91,5 Kilobyte
- Maximale Größe der XML-Dateien: 50,6 Megabyte
- Dokumentanzahl > 5 MB: 917 (0,24% der Dokumente)

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

2 Schlüsselmanagement – Shamir Key Sharing

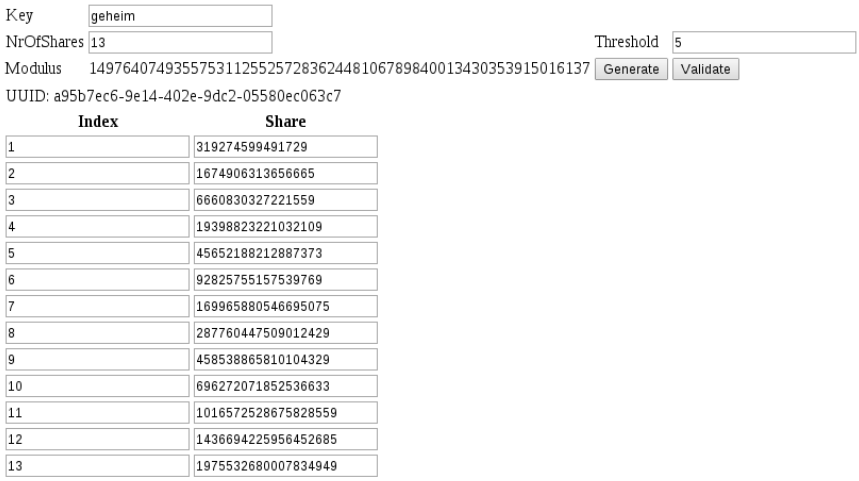
Der Schlüssel wird mittels Shamir Key Sharing (Shamir Key Sharing) in (k/n) Teile geteilt. Damit kann ein 4, 6, 8 oder mehr Augen-Prinzip bei der Freischaltung der Entschüsselung eingehalten werden.

2.1 System-Schlüssel Generierung

Der Systemschlüssel wird im System generiert, die Schlüsselteile können abgelesen und weitergegeben werden.

Es kann eingestellt werden:

- der Schlüssel – Freitext
- Modulus
- Anzahl der Schlüsselteile
- Schwellwert Zahl der benötigte Teile



Index	Share
1	319274599491729
2	1674906313656665
3	6660830327221559
4	19398823221032109
5	45652188212887373
6	92825755157539769
7	169965880546695075
8	287760447509012429
9	458538865810104329
10	696272071852536633
11	1016572528675828559
12	1436694225956452685
13	1975532680007834949

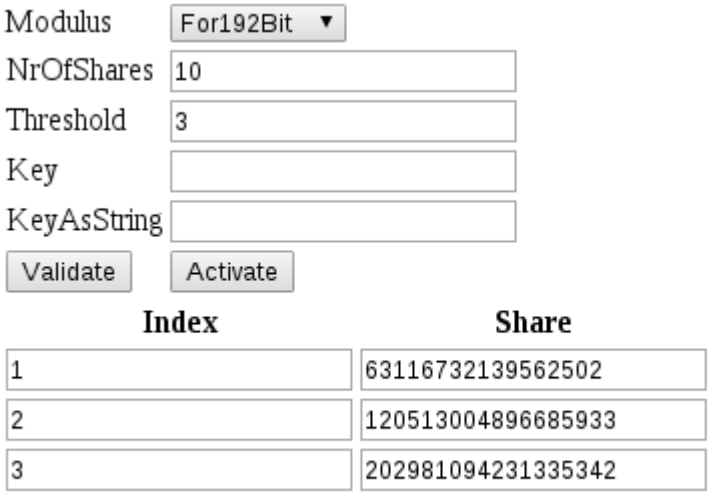
2.2 Schlüssel Zusammenführung Validierung und Aktivierung

Die gemäß Schwellwert benötigten Schlüsselteile

- können eingegeben werden,
- die Gültigkeit des entstehenden Schlüssel verifiziert
- der gültige und verifizierte Schlüssel kann im System aktiviert werden

Damit ist das Secure-FS System einsatzfähig,

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien



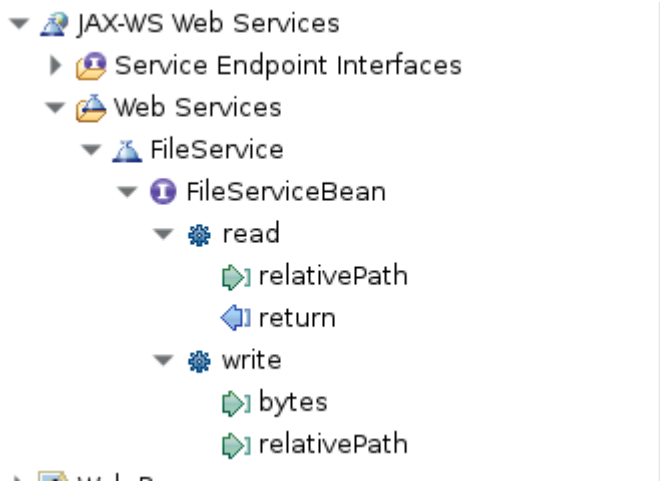
3 Schnittstellen

3.1 WebService File Store

Es steht ein WebService zur Verfügung, Dateien dem Secure-FS System zu übergeben, bzw abzurufen.

Service Operations:

- Read
  - relativer Pfad der Datei
  - Return: Content der Datei als MTOM Attachmert
- Write
  - relativer Pfad der Datei
  - Content der Datei als MTOM Attachmert

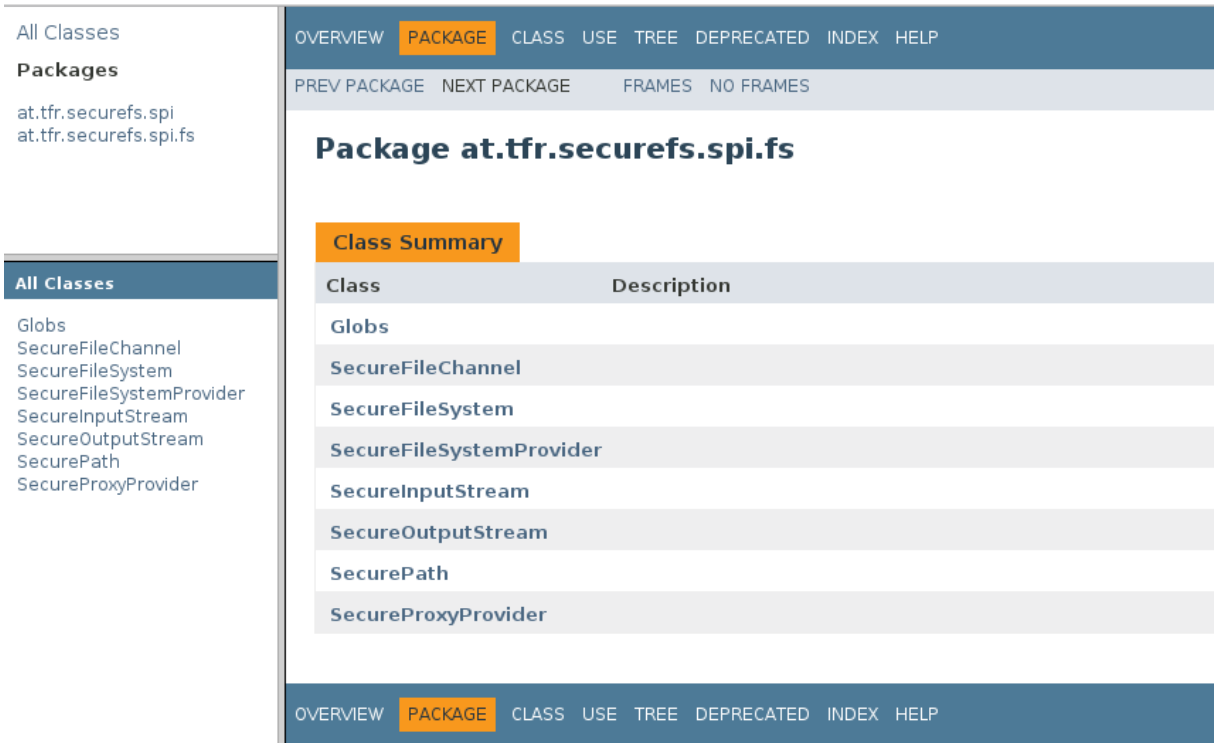


3.2 Secure-FS Java Client Filesystem

Ein Client-seitiges Filesystem gemäß Java JDK8 Package *java.nio.file*: Java NIO2 File

Einbindung des Filesystems gemäß Java Spezifikation für alternative Filesysteme.

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien



3.3 Einbindung der Client Library

3.3.1 Instanziierung des FileSystem Client-Stubs

```
try (FileSystem fs = FileSystems.newFileSystem(new URI(baseDir), null)) {
        // here comes File Transfer
}
```

3.3.2 Schreiben von Dateien / java.io.OutputStream

```
Path sec = fs.getPath(“./relative/path/to/file.txt”);
final OutputStream secOs = Files.newOutputStream(sec);
try {
        IOUtils.copy(Files.newInputStream(path), secOs);
} finally {
        secOs.close();
}
```

3.3.3 Abruf von Dateien / java.io.InputStream

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

```
Path out = Paths.get(“./some/storage.txt”);
final InputStream secIs = Files.newInputStream(sec);
try {
        IOUtils.copy(secIs, Files.newOutputStream(out));
} finally {
        secIs.close();
}
```

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

4 Konfiguration

4.1 Aktivierung Wildfly Remoting u. Authentication

Siehe: https://docs.jboss.org/author/display/WFLY8/Security+Realms

4.2 SecureFS Client Credentials

Konfiguration gemäß Wildfly Remoting
(https://docs.jboss.org/author/display/WFLY8/Remote+EJB+invocations+via+JNDI+-+EJB+client+API+or+remote-naming+project)

4.2.1 Einstellungen in jboss-ejb-client.properties

Siehe: https://github.com/shadogray/securefs/blob/master/securefs-master/securefs-client/src/main/resources/jboss-ejb-client.properties

- remote.connection.default.host
- remote.connection.default.port
- remote.connection.default.username
- remote.connection.default.password

4.2.2 SecureFS Client Naming u. EJB-Remote URL

Siehe: https://github.com/shadogray/securefs/blob/master/securefs-master/securefs-client/src/main/resources/securefs.properties

- java.naming.provider.url = remote://localhost:4447
- securefile.secureFileSystem.url = ejb:/securefs/SecureFileSystemBean!at.tfr.securefs.api.SecureFileSystemItf?stateful

4.3 SecureFS Web-Application

4.3.1 securfs-server.properties

Siehe: https://github.com/shadogray/securefs/blob/master/securefs-master/securefs-common/src/main/resources/securefs-server.properties

Konfiguration:

- securefs.server.keyAlgorithm = AES

- securefs.server.cipherAlgorithm = AES/CBC/PKCS5Padding
- securefs.server.paddingCipherAlgorithm = AES/CBC/PKCS5Padding
- securefs.server.basePath = <Serverseitiges Basisverzeichnis>

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

5 Projektstruktur

5.1 securefs-master

Maven Master Projekt

5.2 securefs-api

SecureFS API Layer

Artefakt: securefs-api-1.0-SNAPSHOT.jar

5.3 securefs-common

SecureFS EJB3 Layer, Crypto-Klassen u. Key-Generierung

Artefakt: securefs-common-1.0-SNAPSHOT.jar

5.4 securefs

SecureFS WebApplication

Artefakt: securefs.war

5.5 securefs-client

SecureFS Client FileSystem Stub Implementation.

Artefakt: securefs-client-1.0-SNAPSHOT.jar

Das Client-seitige FileSystem wird mittel ServiceLoader API registriert.

Service Loader siehe: https://github.com/shadogray/securefs/blob/master/securefs-master/securefs-client/src/main/resources/META-INF/services/java.nio.file.spi.FileSystemProvider

5.6 securefs-client-test

Test Klassen und Last-Test Implementierung.

Artefakt: securefs-client-test-1.0-SNAPSHOT.jar

Der Aufruf erwartet einen laufendem SecureFS-Server.

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

6 SecureFS Service Build

Maven Master Projekt „securefs-master“:

```
~/workspace-sec/securefs/securefs-master> mvn clean install
<SNIP>
Results :

Tests run: 5, Failures: 0, Errors: 0, Skipped: 0
<SNIP>

[INFO] Installing /raid/home/thomas/workspace-sec/securefs/securefs-master/securefs-client-test/pom.xml
to /home/thomas/.m2/repository/at/tfr/securefs/securefs-client-test/1.0-SNAPSHOT/securefs-client-test-
1.0-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] securefs-master ..................................... SUCCESS [  0.290 s]
[INFO] securefs-api ........................................ SUCCESS [  1.434 s]
[INFO] securefs-common ..................................... SUCCESS [  2.534 s]
[INFO] securefs ............................................ SUCCESS [  1.800 s]
[INFO] securefs-client ..................................... SUCCESS [  0.650 s]
[INFO] securefs-json ....................................... SUCCESS [  2.820 s]
[INFO] securefs-client-test ................................ SUCCESS [  1.816 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 11.654 s
[INFO] Finished at: 2015-11-09T22:56:55+01:00
[INFO] Final Memory: 70M/484M
[INFO] ------------------------------------------------------------------------
```

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

7 Serverseitiges Basisverzeichnis

Das Root-Verzeichnis des Servers kann über 2 Methoden eingestellt werden:

- securefs.server.basePath: das Default-Root-Verzeichnis, das bei Basis-relativen Pfaden verwendet wird
  - die URI des FileSystems muss dann relativ sein: „sec:///./“
- Die URI des FileSystems: siehe Files.newFileSystem(<URI>, null)
  - eine absolute URI: „sec:///<RootPath>“ definiert den Server-seitigen Basis-Pfad beim Filezugriff

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

8 Beispiel Implementierung

Projekt: securefs-client-test

Klasse: at.tfr.securefs.client.SecurefsClient

URL: https://github.com/shadogray/securefs/blob/master/securefs-master/securefs-client-test/src/main/java/at/tfr/securefs/client/SecurefsClient.java

8.1 Usage

```
usage: SecureFile
 -a <arg>   Asynchronous tests
 -b <arg>   Base Directory of Server FileSystem
 -f <arg>   Files to run to/from Server, comma separated list
 -t <arg>   Number of concurrent Threads
```

8.2 Aufruf

```
mvn -pl securefs-client-test/ exec:exec -P Client -Dexec.async=false -Dexec.threads=1
-Dexec.files=./dir1/dir2/small.txt
```

8.3 Beispiele / Shell-Scripte

8.3.1 Einfacher Testdurchlauf

Script: https://github.com/shadogray/securefs/blob/master/securefs-master/setupTest.sh

8.3.2 Multithreaded Testdurchlauf

Script: https://github.com/shadogray/securefs/blob/master/securefs-master/setupAsynchronousTest.sh

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

9 Performance Analyse / Last Tests

9.1 Multithreaded File Access

9.1.1 Test-Konfiguration:

- 20 Threads
- Asynchrones Schreiben u. Lesen
- einer großen Datei (ca. 90MB)

9.1.2 Ergebnis

Durchlaufzeit = 108 Sekunden

Durchsatz: 20 * 2 * 90 MB / 108 sec =~ 33 MB / sec

```
Sending file: 2015-11-10T00:11:57.910+01:00 : /tmp/source/bigfile.iso.20
Sending file: 2015-11-10T00:11:57.894+01:00 : /tmp/source/bigfile.iso.17
<SNIP>
Sending file: 2015-11-10T00:11:57.892+01:00 : /tmp/source/bigfile.iso.14
Sending file: 2015-11-10T00:11:57.892+01:00 : /tmp/source/bigfile.iso.9

Reading file: 2015-11-10T00:12:49.468+01:00 : /tmp/source/bigfile.iso.17.out
Reading file: 2015-11-10T00:12:49.652+01:00 : /tmp/source/bigfile.iso.19.out
<SNIP>
Reading file: 2015-11-10T00:12:53.763+01:00 : /tmp/source/bigfile.iso.27.out
Reading file: 2015-11-10T00:12:53.856+01:00 : /tmp/source/bigfile.iso.22.out
Reading file: 2015-11-10T00:12:53.922+01:00 : /tmp/source/bigfile.iso.25.out

Checked Checksums: 2015-11-10T00:13:28.392+01:00 : 4135937012 / 4135937012
Checked Checksums: 2015-11-10T00:13:28.684+01:00 : 4135937012 / 4135937012
Checked Checksums: 2015-11-10T00:13:29.028+01:00 : 4135937012 / 4135937012
<SNIP>
```

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 01:48 min
[INFO] Finished at: 2015-11-10T00:11:07+01:00
[INFO] Final Memory: 10M/379M
[INFO] ------------------------------------------------------------------------

real    1m48.978s
user    0m37.668s
sys     0m16.636s
```

A1 Telekom Austria AG
ITS / SDE / B2B
Antonigasse
A-1170 Wien

10 Security

Die Zugriffe auf die Services erfolgt gesichert über SSL über Standard ApplicationServer-Komponenten.

Es stehen Standard Authentifizierungsverfahren zur Verfügung:

- Username / Passwort
- X-509 Client Certificate
- JBoss Remoting SASL Authenticated