GOVERNIKUS

GOVERNIKUS DATA BOREUM® Web Edition

# Entwicklungs- und Administrationshandbuch

## Governikus DATA Boreum WebEdition

Governikus DATA Boreum WebEdition, Version 10.9.0

**Dokumentenversion: 10.9.0_0**

# Inhaltsverzeichnis

1 Einführung ... 3

2 WebEdition Lizenzdateien ... 6
2.1 Allgemein ... 6
2.2 Grundlegender Aufbau ... 6
2.3 Inhalt der Lizenzdatei ... 6

3 Funktionsübersicht ... 12
3.1 Aufrufparameter ... 12
3.1.1 Allgemeine Parameter ... 12
3.1.2 Parameter für Signier-Funktion ... 14
3.1.3 Parameter Entschlüsselungs-Funktion ... 15
3.1.4 Parameter Verschlüsselungs-Funktion ... 15
3.2 Aufruf der Web Edition ... 16
3.3 Steuerdatei der WebEdition ... 17
3.3.1 Java ... 17
3.4 Konfigurationsdatei ... 18
3.4.1 Konfiguration für die Funktion Signieren ... 18
3.4.2 Beispiel einer Konfigurationsdatei ... 19
3.5 Return-Codes, Fehlerbehandlung, Dateiuploads ... 21
3.5.1 Standard ReturnCodes ... 21
3.5.2 Upload von Ergebnisdateien ... 24

4 Historie Softwareänderungen ... 25

5 Administration ... 26
5.1 Systemanforderungen ... 26
5.2 URI-Encoding ... 26
5.3 SSL Verbindung ... 26
5.4 Governikus DATA Boreum WebEdition deployen ... 27

6 Empfehlungen für den Betrieb ... 28
6.1 Empfohlene Anforderungen an die Einsatzumgebung ... 28
6.2 Empfehlungen für den sicheren Betrieb ... 28
6.3 Technische Anforderungen ... 29
6.4 Anforderungen an die Konfiguration ... 29

7 Verzeichnisse ... 30

# 1 Einführung

Die Governikus DATA Boreum WebEdition ist eine Web-Anwendung zum Signieren und zum Ver- und Entschlüsseln von Dateien.

**Hinweis**: Im Folgenden wird die Governikus DATA Boreum WebEdition oder kurz **WebEdition** genannt.

**Achtung**: Durch die Umstellung der Auslieferung ist diese Version nicht mehr kompatibel zu Vorgängerversionen!

### Signieren

Die WebEdition stellt eine Oberfläche zum Signieren von Dateien zur Verfügung. Es wird die Erstellung von qualifizierten elektronischen Signaturen sowie von fortgeschrittenen Signaturen unterstützt. Die Erstellung von qualifizierten elektronischen Signaturen erfolgt über eine Signaturkarte in einem lokal angeschlossenen Kartenleser. Für die Erstellung von fortgeschrittenen elektronischen Signaturen kann ein Software-Zertifikat genutzt werden.

### Ver- und Entschlüsseln

Die WebEdition stellt eine Oberfläche zum Verschlüsseln und eine Oberfläche zum Entschlüsseln von Dateien zur Verfügung. Die Ver- und Entschlüsselung kann über eine Signaturkarte in einem lokal angeschlossenen Kartenleser erfolgen. Alternativ ist auch eine Verwendung eines Passwortes oder Software-Zertifikats möglich.

Um die Funktionalitäten der WebEdition nutzen zu können, wird ein Programm als Installationsdatei oder als ZIP-Archiv zur Verfügung gestellt. Der Aufruf erfolgt über eine Fachanwendung. Bei der Verwendung von Kartenlesern muss ein entsprechender Treiber installiert sein.

Die Bestimmung der zu verarbeitenden Datei erfolgt durch die Fachanwendung mit dem Aufruf der Anwendung. Es kann eine lokal vorliegende Datei bzw. ein lokales Verzeichnis bestimmt werden, oder es kann die zu verarbeitende Datei über eine URL referenziert werden. Die signierte Datei kann entweder an die Fachanwendung zurückgegeben oder lokal abgelegt werden.

Darüber hinaus kann die aufrufende Fachanwendung Einfluss auf den zur Verfügung gestellten Funktionsumfang nehmen und die Anwendung dadurch präzise an den jeweiligen Workflow anpassen.

**Achtung**: Die Funktionen der WebEdition können nur durch eine Fachanwendung aufgerufen werden. Die Software kann nicht unabhängig von der Fachanwendung als Programm auf dem lokalen PC aufgerufen werden.

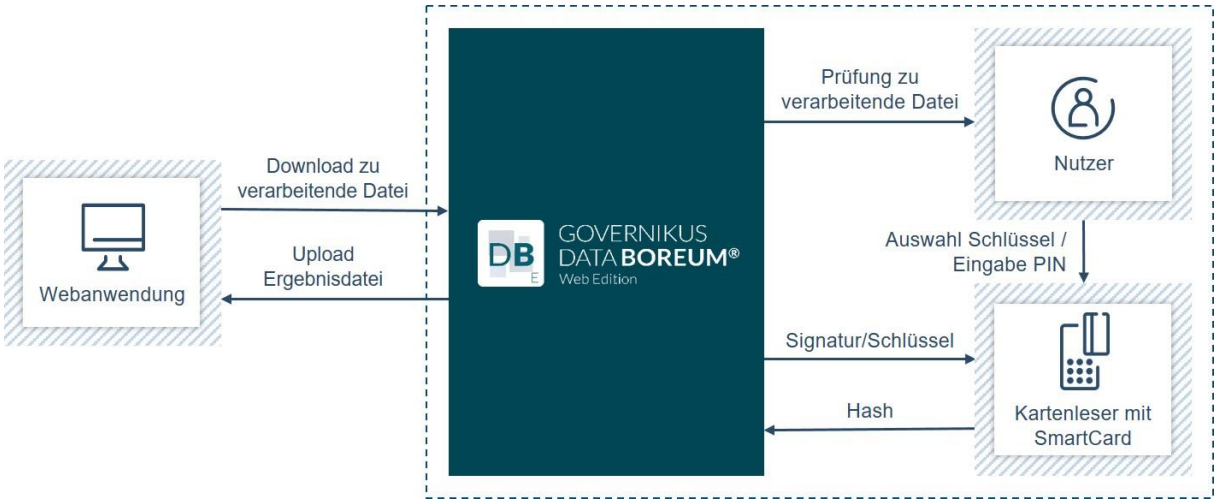


Abbildung 1: Ablaufdiagramm

Die Verarbeitung folgt diesem Ablauf:

- Nach dem Aufruf durch die Fachanwendung öffnet sich die Anwendung WebEdition mit der Anwendungsoberfläche. Die zu verarbeitenden Dateien werden ggf. auf den PC heruntergeladen.
- Wählen Sie von einer Signaturkarte oder aus einer Datei den Signaturschlüssel, Entschlüsselungsschlüssel bzw. ein Verschlüsselungszertifikat aus.

Sofern es die Fachanwendung erlaubt, vergleiche Kapitel 2, können folgende Optionen festgelegt werden:

- Signieren
  - Signaturformat (PKCS#7 detached, PKCS#7 enveloped, PDF-Inline)
  - Verhalten bei vorhandener Signatur (nur bei PDF-Inline-Signatur)
  - Bestimmung des Zielverzeichnisses
  - Eine lokale Kopie erstellen
  - Zeitstempel erzeugen
- Verschlüsseln
  - Entschlüsselungsmethode auswählen (Passwort oder Entschlüsselungsschlüssel von Signaturkarte oder Softwareschlüssel)
  - ZIP-Archive nach dem Entschlüsseln automatisch entpacken
  - Bestimmung des Zielverzeichnisses
  - Eine lokale Kopie erstellen
- Entschlüsseln
  - Verschlüsselungsmethode auswählen (Passwort oder Zertifikat)
  - Dateien automatisch vor dem Verschlüsseln in einem ZIP-Archiv zusammenfassen
  - Bestimmung des Zielverzeichnisses
  - Eine lokale Kopie erstellen
- Vor der Signaturerstellung bzw. vor dem Verschlüsseln besteht die Möglichkeit, die zu signierenden Dateien einzusehen. Beim Entschlüsseln besteht diese Möglichkeit nicht.

- Nach dem Starten des Signatur- bzw. des Entschlüsselungsvorgangs ist für jede verarbeitende Datei die Eingabe der PIN erforderlich, es sei denn, es wird eine der unterstützten Stapelsignaturkarten oder ein Software-Schlüssel verwendet.
- Nach dem Abschluss der Dateiverarbeitung schließt sich die Anwendung und liefert einen `ReturnCode` an die aufrufende Anwendung zurück.

# 2 WebEdition Lizenzdateien

Dieses Kapitel erklärt allgemeines, den grundlegenden Aufbau und den Inhalt der Lizenzdatei.

## 2.1 Allgemein

Die WebEdition verfügt über zwei Lizenzdateien, die zur Konfiguration der Software genutzt werden können.

- Die Produktlizenzdatei wird mit der WebEdition ausgeliefert und definiert den generellen Funktionsumfang der Software.
- Beim Aufruf der WebEdition kann die aufrufende Anwendung eine weitere Lizenzdatei übergeben. Diese Lizenzdatei für eine Person kann die Produktlizenz nur maskieren, d.h. es können Funktionen deaktiviert werden aber keine über die Produktlizenzdatei deaktivierten Funktionen wieder aktiviert werden.

## 2.2 Grundlegender Aufbau

Die Lizenzdatei ist ein XML-Dokument, das für jeden Unterpunkt einen Lizenztyp und ggf. Unterpunkte definiert. Jeder Lizenztyp hat die drei Eigenschaften "Editierbar", "im Prozess nutzen" und "Anzeigen", um den Funktionsumfang der Software zu konfigurieren.

- **Anzeigen**: Die entsprechenden grafischen Elemente werden auf der Anwendungsoberfläche dargestellt.
- **Editierbar**: Es besteht die Möglichkeit, die Einstellung dieser Funktion über die Anwendungsoberfläche zu ändern.
- **Im Prozess nutzen**: Die jeweilige Funktion steht grundsätzlich zur Nutzung zur Verfügung.

Von den acht möglichen Kombinationen dieser drei Eigenschaften, sind nur fünf Kombinationen in diesem Kontext nutzbar. In der nachfolgenden Tabelle sind die daraus resultierenden fünf Lizenztypen aufgeführt und die jeweils abgedeckten Eigenschaften durch ein "X" gekennzeichnet. Verfügbare Lizenztypen:

| Lizenztyp | Editierbar | Im Prozess nutzen | Anzeigen |
|---|---|---|---|
| `enable` | X | X | X |
| `hide` | | X | |
| `hide_disable` | | | |
| `disable` | | | X |
| `notEditable` | | X | X |

## 2.3 Inhalt der Lizenzdatei

Die Produktlizenzdatei der WebEdition ist wie folgt aufgebaut:

```
<root displayName="Governikus DATA Boreum WebEdition">
  <general>enable
    <actions>
      <settings>
```

```
        <proxyserver>enable</proxyserver>
      </settings>
    </actions>
  </general>
  <process>
    <sign>
      <options>enable
       <visualization>enable</visualization>
        <signformat>enable
          <pkcs7detached>enable</pkcs7detached>
          <pdfinline>enable</pdfinline>
          <pkcs7enveloped>enable</pkcs7enveloped>
        </signformat>
        <extended_pdfsignature>enable
          <reason>enable</reason>enable
         /extended_pdfsignature>
        <timestampserver>enable</timestampserver>
        <timestamp>enable</timestamp>
      </options>
      <sourcefolder>enable
       <addfiles>enable</addfiles>
        <removefiles>enable</removefiles>
      </sourcefolder>enable
     <key>enable
        <software>enable</software>
        <signaturelevel>enable</signaturelevel>
      </key>
      <targetfolder>enable
        <targetfolder>enable</targetfolder>
        <localcopyfolder>enable</localcopyfolder>
      </targetfolder>
    </sign>
  </process>
</root>
```

Listing 1: Inhalt der WebEdition Produktlizenzdatei für das Signieren

Die Produktlizenzdatei für das Ver- und Entschlüsseln ist wie folgt aufgebaut:

```
<root displayName="Encrypt und Decrypt">
  <general>enable
    <actions>
      <settings>
        <proxyserver>enable</proxyserver>
      </settings>
    </actions>
  </general>
  <process>
    <decrypt>enable
      <targetfolder>enable
        <unzip>enable</unzip>
        <localcopyfolder>enable</localcopyfolder>
        <targetfolder>enable</targetfolder>
      </targetfolder>
      <key>enable
        <software>enable</software>
```

```
        <password>enable</password>
      </key>
    </decrypt>
    <encrypt>enable
      <targetfolder>enable
        <targetfolder>enable</targetfolder>
        <createzip>enable</createzip>
        <localcopyfolder>enable</localcopyfolder>
      </targetfolder>
      <key>enable
        <software>enable</software>
        <password>enable</password>
        <encryptmultiaddressee>enable</encryptmultiaddressee>
      </key>
    </encrypt>
  </process>
</root>
```

Listing 2: Inhalt der WebEdition Produktlizenzdatei für das Ver- und Entschlüsseln

### Lizenzdatei für Personen

Die Lizenzdatei für Personen maskiert die Produktlizenzdatei, d.h. sie kann den Funktionsumfang weiter einschränken. Da alle relevanten Einstellungen der Produktlizenz den Lizenztyp `enable` tragen, entsprechen die Inhalte der Lizenzdatei für Personen direkt der "Gesamtlizenz".

Die nachfolgende Tabelle erläutert die einzelnen Elemente und deren möglichen Lizenztypen. Die detaillierte Beschreibung der einzelnen Funktionen entnehmen Sie bitte dem Anwendungshandbuch. Beachten Sie bitte, dass es nachfolgend nur darum geht, inwieweit die einzelnen Funktionen, Konfigurationsmöglichkeiten und Dialogschritte einer Person zur Verfügung stehen. Eine Vorgabe der Konfiguration, z.B. welches Signaturformat oder welches Zielverzeichnis genutzt werden soll, erfolgt über Aufrufparameter.

| **Element** | **Bedeutung** | **Mögliche Lizenztypen** | | | | |
|---|---|---|---|---|---|---|
| | | **enable** | **hide** | **notEditable** | **disable** | **hide_disable** |
| `sign` | Signaturprozess allgemein | X | | | | |
| `sourcefolder` | Dialogschritt "Dateiauswahl" | X | X | X | | |
| `addfiles` | Funktion "Dateien hinzufügen" | X | X | X | X | X |
| `removefiles` | Funktion "Ausgewählte Dateien entfernen" | X | X | X | X | X |
| `options` | Dialogschritt "Optionen" | X | X | X | | |
| `signformat` | Auswahl "Standardsignaturformat wählen" im Dialogschritt "Optionen" | X | X | X | | |

| Element | Bedeutung | Mögliche Lizenztypen | | | | |
|---|---|---|---|---|---|---|
| | | enable | hide | notEditable | disable | hide_disable |
| pkcs7detached | Signaturformat PKCS#7 detached ("Signatur als gesonderte Datei beifügen") | X | X | X | X | X |
| pdfinline | Signaturformat PDF-Signatur ("PDF-Signatur erstellen") | X | X | X | X | X |
| pkcs7enveloped | Signaturformat PKCS#7 enveloped ("Dokument in Signaturdatei einbetten") | X | X | X | X | X |
| extended_pdfsignature | Sichtbare PDF-Signatur ermöglichen ("Sichtbares Signaturfeld") | X | X | X | X | X |
| reason | Angaben zur Signatur von PDF-Dateien (sichtbare und unsichtbare Signatur)<br>• Verwendung von Templates („Vorlagen“)<br>• Signaturgrund bei erweiterter PDF-Signatur ("Grund der Unterschrift")<br>• Angabe Signaturort bei erweiterter PDF-Signatur ("Ort der Unterschrift") | X | X | X | X | X |
| timestampserver | Konfiguration des Zeitstempelservers | X | X | X | X | X |
| timestamp | Externen Zeitstempel anbringen Dialogschritt „Optionen“ | X | X | X | X | X |
| visualization | Mindestanzahl der einzusehenden Dateien prüfen | X | | | X | |
| key | Dialogschritt "Schlüssel wählen" | X | X | X | | |
| software | Funktion "Schlüssel aus Datei laden" (siehe auch nachfolgende Hinweisbox) | X | X | X | X | X |
| signaturelevel | Funktion „Signaturniveau“ | X | X | X | | |
| targetfolder | Dialogschritt "Zielverzeichnis wählen" | X | X | X | | |
| targetfolder | Funktion "Zielverzeichnis wählen" | X | X | X | | |
| localcopyfolder | Funktion "Lokale Kopie erstellen" | X | X | X | | |
| decrypt | Entschlüsselungsprozess allgemein | X | | | | |

| Element | Bedeutung | Mögliche Lizenztypen | | | | |
|---|---|---|---|---|---|---|
| | | enable | hide | notEditable | disable | hide_disable |
| encrypt | Verschlüsselungsprozess allgemein | X | | | | |
| unzip | Automatisches Entpacken von ZIP-Archiven nach dem Entschlüsseln | X | X | X | X | X |
| createzip | Automatisches Erstellen von ZIP-Archiven mit allen Dateien vor dem Verschlüsseln. | X | X | X | X | X |
| encryptmultiaddressee | Verschlüsseln für mehr als ein Zertifikat. | X | X | X | X | X |
| password | Ver-/Entschlüsselung über Passwort | X | X | X | X | X |

Tabelle 1: Übersicht über die Parameter der Lizenzdatei

Darüber hinaus wird beim Einlesen der Lizenzdateien für Personen jeder Unterpunkt mit dessen Knotenpunkt maskiert. Somit kann kein Unterpunkt (z. B. Funktion) sichtbar und dessen Knotenpunkt (z. B. Dialogschritt) unsichtbar sein.

Das nachfolgende Beispiel zeigt eine Lizenz und die daraus resultierende Gesamtlizenz. Die Lizenz für Personen erlaubt nur Signaturen im PKCS#7-Format, aber keine PDF-Inline-Signaturen. Die Einstellungsmöglichkeit des Signaturformates im Dialogschritt "Optionen" soll nicht auf der Anwendungsoberfläche dargestellt werden (d.h. welches Format verwendet werden soll, muss als Aufrufparameter übergeben werden). Es können nur Schlüssel von Signaturkarten verwendet werden. Beispiel:

```
<sign>enable
  <options>enable
    <signformat>notEditable
      <pdfinline>disable</pdfinline>
    </signformat>
  </options>
  <key>enable
    <software>hide_disable</software>
  </key>
  <targetfolder>
    <targetfolder>hide</targetfolder>
  </targetfolder>
</sign>
```

Listing 3: Beispiel einer Lizenzdatei für Personen

```
<sign>enable
  <sourcefolder>enable
    <addfiles>enable</addfiles>
    <removefiles>enable</removefiles>
  </sourcefolder>
  <options>enable
```

```
      <signformat>hide
        <pkcs7detached>hide</pkcs7detached>
        <pdfinline>hide_disable</pdfinline>
        <pkcs7enveloped>hide</pkcs7enveloped>
      </signformat>
      <visualization>enable</visualization>
    </options>
    <key>enable
      <software>hide_disable</software>
      <signaturelevel>notEditable</signaturelevel>
    </key>
    <targetfolder>enable
      <targetfolder>hide</targetfolder>
      <localcopyfolder>enable</localcopyfolder>
    </targetfolder>
  </sign>
```

Listing 4: Beispiel einer interpretierten Lizenzdatei ("Gesamtlizenz")

### Standard-Lizenz für Personen

Ohne Angabe einer Lizenzdatei bietet die WebEdition der Person den gesamten Funktionsumfang. I. d. R. sind Optionen wie Signaturformat sowie Zielverzeichnis durch den Einsatzzweck vorgegeben und sollen durch die Personen nicht geändert werden. Die zugehörigen Dialogschritte "Dateiauswahl", "Optionen" und "Zielverzeichnis wählen" können dann entfallen. Die nachfolgend abgebildete Lizenzdatei blendet die o. g. Dialogschritte aus.

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <process>
    <sign>enable
      <sourcefolder>hide</sourcefolder>
      <options>hide</options>
      <targetfolder>hide</targetfolder>
    </sign>
  </process>
</root>
```

Listing 5: Inhalt der enthaltenen Standard-Lizenzdatei für Personen

Diese Standard-Lizenzdatei ist bereits in der Anwendung enthalten. Soll die Lizenzdatei verwendet werden, ist mit dem Aufruf der Anwendung der Parameter `licenceFile` (siehe folgende Kapitel) mit folgender URL anzugeben:

```
http://<server>:<port>/GovernikusDataBoreumWebEditionSample/dynamicSignLicence.xml
```

**Hinweis**: Die mitgelieferte Online-Hilfe beschreibt den vollständigen Funktionsumfang. Es wird auch auf den Fall eingegangen, dass Einstellungsoptionen zwar sichtbar sind, aber durch die Person nicht verändert werden können (Lizenztyp `notEditable`/`disable`).

# 3 Funktionsübersicht

Nachfolgend werden die Funktionen anhand der Aufrufparameter beschrieben. Wie der Aufruf der Anwendung erfolgt, ist im Abschnitten 3.2 dargestellt. Im folgenden Abschnitt 3.1 sind zunächst die allgemeinen Aufrufparameter, die gleichermaßen für alle Funktionen gelten, und anschließen die funktionsspezifischen Parameter erläutert.

## 3.1 Aufrufparameter

Es ist zwischen Pflichtparametern, die bei jedem Aufruf übergeben werden müssen, und optionalen Parametern zu unterscheiden. Bei den optionalen Parametern ist das Zusammenspiel mit der GUI bzw. der dynamischen Lizenz zu beachten. Ist z.B. das Signaturformat über die GUI wählbar, wirkt der entsprechende Parameter nur als Vorbelegung, die von der Person beliebig verändert werden kann. Soll die Person dieses nicht ändern können, so ist diese Option über die Lizenzdatei (vgl. Kapitel 2) entweder zu deaktivieren `notEditable` oder zu verstecken `hide`.

### 3.1.1 Allgemeine Parameter

#### Pflichtparameter:

- **`signer.process=<Prozesstyp>`**: Geben Sie hier an, mit welcher Funktion die WebEdition gestartet werden soll. Ersetzen Sie `<Prozesstyp>` durch einen dieser Einträge `sign`, `encrypt` oder `decrypt`.
- **`signer.rcurl`**: URL (z. B. einer JSP), an die der Return-Code inkl. der Return-Code-Beschreibung über die `GET`-Parameter `returnCode` und `returnCodeDescription`, gesendet werden soll. Beispiel:
  - `http://<server>:<port>/GovernikusDataBoreumWebEditionSample/rcCode?transactionID=12345678`

#### Optionale Parameter

Abhängigkeiten einzelner Parameter untereinander sind zu beachten. Für nicht angegebene Parameter wird, wenn erforderlich, ein Default-Wert verwendet.

- **`signer.source`**: Dateiname oder URL der zur signierenden (Quell-) Datei. Alternativ kann auch ein Verzeichnis angegeben werden, aus dem dann alle vorhandenen Dateien ausgewählt werden. Datei- und Verzeichnisnamen müssen immer mit dem vollständigen Pfad angegeben werden. Im Fall einer URL wird von der zu signierenden Datei zunächst eine lokale Kopie angelegt. Der Name der lokalen Kopie kann der URL optional, durch ein Pipe-Zeichen getrennt, mitgegeben werden. Beispiel:
  - **Einzeldatei lokal**: `/home/governikus/temp/source/test.pdf`
  - **Verzeichnis lokal**: `/home/governikus/temp/source/`
  - **Einzeldatei URL**: Es gibt zwei Möglichkeiten der Übergabe einer Datei:
    - `[URL-zur-Datei|lokale-datei|sha256-hash]` oder
    - `[URL-zur-Datei]`

    Der Aufbau sieht wie folgt aus:
    `https://<server>:<port>/server_datei.pdf|lokale_datei.pdf|[sha256-hash]`

    - **`lokale_datei`**: Bei einer Quelle im Internet kann ein lokaler Dateiname mit angegeben werden, da nicht alle Internetquellen (URL) einen Dateinamen in der URL enthalten. **`sha256-hash`**: Zusätzlich kann ein Sha256-Hashwert der Datei im Aufruf mitgegeben werden. Wenn ein Hashwert mitgegeben wird, überprüft die WebEdition diesen mit der herunter geladenen Datei. Sollten die beiden Hashwerte nicht übereinstimmen, wird ein `ReturnCode 59 (HASHEDFILE_CHANGED)` an den Server gesendet und der Person im Fehlerdialog angezeigt. Mit dem Senden des ReturnCodes wird der lokal errechnete Hashwert mit der Datei verknüpft.

- **Mehrere Dateien URL**: Bei der Übergabe mehrerer Dateien erfolgt die Aufzählung mit dem Trennzeichen #. Es gibt zwei Möglichkeiten der Übergabe mehrerer Dateien:
    - `[URL-zur-Datei|lokale-datei-name|sha256-hash]#[URL-zur-Datei|lokale-datei-name|sha256-hash]…`
    - `[URL-zur-Datei]#[URL-zur-Datei]…`

**Hinweis**: Sobald der Hashwert einer Serverdatei nicht mit dem lokal ermittelten Hashwert übereinstimmt und die Person diesen Vorgang aber trotz der Fehlermeldung nicht abbricht, bekommt er keine weitere Fehlermeldung vor dem Signieren.

- Alle o. g. Quellen können auch mehrfach in beliebiger Kombination angegeben werden.

**Hinweis**: Sobald eine Quelldatei als URL übergeben wurde, sollte als `signer.targetFolderTyp` immer `oneSpecial` übergeben werden, da die Ergebnisdatei einer URL-Quelldatei beim Parameter `sameAsSourceFolder` im Temp-Verzeichnis gespeichert werden würde.

- **`signer.dest`**: Lokales Zielverzeichnis oder Ziel-URL.

  Dieser Parameter wird nicht ausgewertet, wenn für folgenden Parameter gilt:

  - `signer.targetFolderType = sameAsSourceFolder`

- **`signer.targetFolderType`**: Bestimmt die Art des Zielverzeichnisses:
  - **`sameAsSourceFolder`**: (Default) Verzeichnis der jeweiligen Quelldatei verwenden (nicht möglich, wenn der Parameter `signer.source` eine URL enthält). Parameter `signer.dest` wird nicht ausgewertet.
  - **`oneSpecial`**: separat angegebenes Zielverzeichnis (erfordert Parameter `signer.dest`)
- **`signer.processPolicy`**: Steuert, mit welchem Dialogschritt die Anwendung startet. Welche Dialogschritte überhaupt möglich sind, wird über die Lizenz-Datei vorgegeben (vgl. Kapitel 2).
  - **`skipCompletedStep`**: (Default) Alle Dialogschritte, in denen die notwendigen Angaben bereits vorliegen (über Default-Einstellungen oder Parameter) werden übersprungen.
  - **`firstStep`**: Der Dialog beginnt mit dem ersten Schritt.
  - **`lastStep`**: Der Dialog startet mit dem Dialogschritt "Signieren".

- **`signer.configurationFile`**: Voll qualifizierter Dateiname einer lokalen Konfigurationsdatei oder URL zu einer Konfigurationsdatei. Über die Konfigurationsdatei können alle Einstellungen, die in der WebEdition über den Einstellungsdialog eingestellt werden, übergeben werden.
- **`signer.licenceFile`**: Voll qualifizierter Dateiname einer lokalen Lizenzdatei oder URL zu einer Lizenzdatei. Über die Lizenzdatei kann der Funktionsumfang der GUI eingeschränkt werden (vgl. Kapitel 2).

### 3.1.2 Parameter für Signier-Funktion

#### Pflichtparameter:

- **`signer.process = sign`**
- **`signer.rcurl:`** siehe 3.1.1 Allgemeine Parameter

#### Optionale Parameter:

- **`signer.sigFormat`**: Bestimmt das Signaturformat
  - **`detachedPKCS7`**: (Default) PKCS#7 Signaturdatei mit separater Inhaltsdatei. Die Inhaltsdatei wird immer mit an den Zielort übergeben.
  - **`envelopedPKCS7`**: PKCS#7 Signaturdatei mit eingebettetem signierten Inhalt
- **`signer.pdfSignatureType`**: Steuert die Auswahl des Signaturformats für PDF-Dateien:

**Hinweis**: Nach der Erstellung einer PDF-Inline-Signatur enthält der Dateiname der signierten PDF-Datei immer den Zusatz `_signed`". Bsp.: `Rechnung_signed.pdf`

  - **`pdfInline`**: (Default) PDF-Dateien werden, unabhängig vom Parameter `signer.sigFormat`, im Format PDF-Inline, also einer in dem PDF eingebetteten PKCS#7-Signatur, signiert.

**Hinweis**: Die Einstellung, ob ein Signaturfeld verwendet wird, wird nur in der Konfigurationsdatei festgelegt (`signer.configurationFile`, siehe Kapitel 3.1.1). Die Beschreibung zur Konfigurationsdatei finden Sie in Kapitel 3.4.

  - **`noPdfInline`**: PDF-Dateien werden gemäß dem Parameter `signer.sigFormat` signiert.
- **`signer.advancedPdfReason`**: Signaturgrund für die PDF-Signatur. Dieser Text wird in der PDF-Datei in den Unterschriftseigenschaften hinterlegt und in der sichtbaren PDF-Signatur dargestellt, sofern diese erstellt wird. In unsichtbaren Signaturen ist die Angabe ebenfalls enthalten.
- **`signer.advancedPdfLocation`**: Signaturort für die PDF-Signatur. Dieser Text wird in der PDF-Datei in den Unterschriftseigenschaften hinterlegt und in der sichtbaren PDF-Signatur dargestellt, sofern diese erstellt wird. In unsichtbaren Signaturen ist die Angabe ebenfalls enthalten.
- **`signer.xmlSignatureType`**: Steuert die Auswahl des Signaturformats für XML-Dateien:
  - **`noXmlInline`**: (Default) XML-Dateien werden gemäß dem Parameter `signer.sigFormat` signiert.

- **xmlInline**: XML-Dateien werden, unabhängig von dem Parameter signer.sigFormat, im Format XML-detached signiert, also einer Signatur als gesonderte XML-Datei **signer.createTimestamp**: Gibt an, ob ein Zeitstempel erzeugt werden soll.
- **false**: (Default) Es wird kein Zeitstempel erzeugt.
- **true**: Ein Zeitstempel wird erzeugt. Es muss über den Parameter **signer.configurationFile** ein Zeitstempeldienst in der Konfigurationsdatei mit übergeben werden.

### 3.1.3 Parameter Entschlüsselungs-Funktion

#### Pflichtparameter:

- **signer.process = decrypt**
- **signer.rcurl**: siehe 3.1.1 Allgemeine Parameter

#### Optionale Parameter

- **signer.extractziparchiv**: Gibt an, ob ein verschlüsseltes ZIP-Archiv nach der Entschlüsselung automatisch in ein Unterverzeichnis entpackt werden soll. Das Unterverzeichnis bekommt den Namen des ZIP-Archivs.
  - **yes**: (Default) Alle ZIP-Archive werden nach der Entschlüsselung automatisch in ein Unterverzeichnis entpackt.
  - **no**: Es erfolgt keine Prüfung ob es sich um ein ZIP-Archiv handelt. Je verschlüsselte Datei gibt es eine entschlüsselte Datei.
- **signer.usepassword:** Gibt an, ob eine verschlüsselte Datei mit einem Passwort entschlüsselt werden soll.
  - **no**: (Default) Die verschlüsselten Dateien werden mit einem Zertifikat entschlüsselt.
  - **yes**: Passwortverschlüsselte Dateien werden mit Passwort entschlüsselt.

### 3.1.4 Parameter Verschlüsselungs-Funktion

#### Pflichtparameter:

- **signer.process = encrypt**
- **signer.rcurl**: siehe 3.1.1 Allgemeine Parameter

#### Optionale Parameter

- **signer.source**: Dateiname oder URL der (Quell-) Datei. Alternativ kann auch ein Verzeichnis angegeben werden, aus dem alle vorhandenen Dateien (ohne Unterverzeichnisse) ausgewählt werden. Datei- und Verzeichnisnamen müssen immer voll qualifiziert angegeben werden. Im Fall einer URL wird von der zu signierenden Datei zunächst eine lokale Kopie angelegt. Der Name der lokalen Kopie kann der URL optional, durch ein Pipe-Zeichen getrennt, übergegeben werden. Beispiele:
  - Einzeldatei lokal: /home/governikus/temp/source/test.pdf
  - Verzeichnis lokal: /home/governikus/temp/source/
  - Einzeldatei URL:

```
https://<server>:<port>/server_datei.pdf|lokale_datei.pdf|sha256-
hash
```

- **lokaler_datei_name.pdf**: Bei einer Quelle im Internet kann ein lokaler Dateiname mit angegeben werden, da nicht alle Internetquellen einen Dateinamen in der URL enthalten.
- **sha256-hash**: Zusätzlich kann ein Sha256-Hashwert der Datei im Aufruf mitgegeben werden. Wenn ein Hashwert mitgegeben wird, überprüft die WebEdition diesen mit der herunter geladenen Datei. Sollten beide Hashwerte nicht übereinstimmen, wird ein `ReturnCode 59 (HASHEDFILE_CHANGED)` an den Server gesendet und der Person im Fehlerdialog angezeigt. Mit dem Senden des Return-Codes wird der lokal errechnete Hashwert mit der Datei verknüpft.

**Achtung**: Die Übergabe eines Hashwerts ist nur bei einem Aufruf über eine URL möglich.

**Hinweis**: Sobald der Hashwert einer Serverdatei nicht mit dem lokal ermittelten Hashwert übereinstimmt, und die Person diesen Vorgang aber trotz der Fehlermeldung nicht abbricht, wird keine weitere Fehlermeldung vor dem Signieren angezeigt.

**Hinweis**: Sobald eine Quelldatei als URL übergeben wurde, sollte als `targetFolderTyp` immer `oneSpecial` übergeben werden, da die Ergebnisdatei einer URL-Quelldatei beim Parameter `sameAsSource` im Temp-Verzeichnis gespeichert werden würde.

- **signer.usepassword**: Gibt an, ob eine Datei mit einem Passwort verschlüsselt werden soll.
  - **no**: (Default) Die Dateien werden nicht mit einem Passwort verschlüsselt.
  - **yes**: Dateien werden mit Passwort verschlüsselt.
- **signer.createziparchivbeforeencrypt**: Gibt an, ob alle Dateien vor der Verschlüsselung automatisch in einem ZIP-Archiv zusammengefasst werden soll.
  - **no**: (Default) Alle Dateien werden einzeln verschlüsselt.
  - **yes**: Alle Dateien werden vor der Verschlüsselung in einem ZIP-Archiv zusammengefasst. Somit gibt es nur eine verschlüsselte Datei. Diese kann bei der Entschlüsselung auch wieder automatisch entpackt werden.

## 3.2 Aufruf der Web Edition

Durch das Deployen der Datei `GovernikusDataBoreumWebEdition.war` in den Tomcat-Webserver wird für die Aufrufe der WebEdition die Datei `start.jsp` im Tomcat-`webapps`-Verzeichnis erstellt, die folgenden Inhalt hat.

```
<%@ page
info="WebEdition Config File"
```

```
%><%
String setupfile = request.getParameter("configfile");
response.reset();
response.setContentType("application/octet-stream");
response.setHeader("Content-Type", "application/force-download");
response.setHeader("Content-Transfer-Encoding", "binary");
response.setHeader("Content-Length", Integer.toString(setupfile.length()));
response.setHeader("Content-Disposition", "attachment; filename=\"setup.gov-
ws\"");%><%= setupfile%>
```

Listing 6: Inhalt der Datei `start.jsp`

Damit kann über eine URL die WebEdition mit der gewünschten Funktion aufgerufen werden.

```
http://<server>:<port>/GovernikusDataBoreumWebEdition/start.jsp?configfile=
http://<server>:<port>/GovernikusDataBoreumWebEditionSample/testdata/sign.g
ov-ws
```

## 3.3 Steuerdatei der WebEdition

Für den Aufruf der Steuerdatei auf dem PC des Anwenders wird mit der aufrufenden URL oder über das Web-Portal eine Datei mit der Endung `.gov-ws` übergeben. Auf Windows Betriebssystemen wird die Dateiassoziation der Dateiendung `.gov-ws` mit der Steuerdatei automatisch bei der Installation erstellt.

Die Steuerdatei mit der Endung `.gov-ws`, die mit dem Aufruf der Steuerdatei übergeben wird, enthält die Parameter, die für die angeforderte Funktion (`sign`, `crypt` oder `decrypt`) benötigt werden. Beispiel:

```
signer.process=sign
signer.rcURL =
http://<server>:<port>/GovernikusDataBoreumWebEditionSample/rcCode?transact
ionID=xx.xx.xx.xx
signer.source = <Laufwerk/Serververzeichnis>:/ports.txt
#signer.configurationFile =
http://<server>:<port>/GovernikusDataBoreumWebEditionSample/configuration/c
onfiguration.xml
signer.dest = < Laufwerk/Serververzeichnis >:/
signer.targetFolderType = oneSpecial
signer.sigFormat = detachedPKCS7
signer.processPolicy = skipCompletedStep
signer.pdfSignatureType = noPdfInline
signer.xmlSignatureType = noXmlInline
signer.createtimestamp = false
```

Listing 7: Beispiel für eine `.gov-ws` Datei

### 3.3.1 Java

Dadurch, dass die WebEdition als Programm auf dem Anwender-PC installiert ist, muss im Gegensatz zu früheren Versionen kein Java auf dem PC installiert sein oder aktualisiert werden.

## 3.4 Konfigurationsdatei

Der WebEdition wird eine Konfigurationsdatei übergeben. Diese Konfiguration enthält Einstellungen, die nicht als separate Parameter übergeben werden können:

- Mindestanzahl einzusehender Dateien
- Details zur sichtbaren PDF-Signatur
- Zeitstempelserver

Ein kommentiertes Beispiel der Konfigurationsdatei ist weiter unter enthalten.

**Hinweis**: Die Einstellungen zu den Funktionen "Mindestanzahl einzusehender Dateien", die Einstellungen zur sichtbaren PDF-Signatur und "Zeitstempelserver / Proxy" wirken nicht, wenn diese Funktionen über die Lizenzdatei für Personen deaktiviert sind (`disable` oder `hide_disable`).

### 3.4.1 Konfiguration für die Funktion Signieren

#### Mindestanzahl einzusehender Dateien

Es kann vorgegeben werden, ob das Einsehen von Dateien erforderlich ist und wie groß der Anteil der einzusehenden Dateien ist (Stichprobengröße). Das Signieren wird erst freigegeben, wenn die Person den vorgegebenen Anteil an Dokumenten geöffnet hat.

#### Spezielle Signaturformate

PAdES und CAdES sind die Weiterentwicklungen der bisherigen PDF-Inline bzw. PKCS#7 Signaturformate. Letztere sollte nur ausgewählt werden, wenn bei dem Signaturempfänger Kompatibilitätsprobleme bei mit den neueren Formaten existieren.

#### Details zur sichtbaren PDF-Signatur

Hier kann die sichtbare PDF-Signatur aktiviert und konfiguriert werden. Details zu den Einstellungsmöglichkeiten entnehmen Sie bitte dem Anwendungshandbuch bzw. der kommentierten Beispiel-Konfigurationsdatei. Beachten Sie bitte folgende Einschränkungen:

- Damit eine sichtbare Signatur erzeugt wird, muss über die Einstellung `<visualisationTyp>` eine Visualisierungsart ausgewählt sein. Außerdem muss mindesten ein Textinhalt (Unterzeichnername, Ort, Datum oder Grund) vorhanden sein.
- Wenn die Größe der Textelemente die Größe des Visualisierungsbereichs überschreitet (vgl. Einstellung "Höhe", "Breite", "Grafik/Text" versus Schriftgröße), wird die Signaturerstellung abgebrochen. Aktivieren Sie zur Vermeidung von Abbrüchen bei z.B. langen Namen die Option `<useDynamicTextSize>`.
- Wird eine Grafik eingefügt, muss der Pfad, bzw. die URL, zur Grafik auch vom Client-PC der Person erreichbar sein.

Signatur-Vorlagendateien können mit der Konfiguration nicht übergeben werden.

#### Zeitstempelserver

Als Zeitstempelserver kann nur ein Governikus System verwendet werden. Neben der URL und der Profil-ID zum Zeitstempelserver müssen die Einstellungen zum Authentisierungsdienst für die erforderliche Authentisierung übergeben werden. Diese Einstellungen werden durch den Administrator des Zeitstempelservers bereitgestellt.

### 3.4.2 Beispiel einer Konfigurationsdatei

Das folgende Listing ist ein Beispiel für eine Konfigurationsdatei für die Signaturerstellung. Wenn diese Datei beim Aufruf nicht gefunden wird, werden die Default-Werte benutzt, die im Handbuch `Governikus-DATA-Boreum-WE-Samples-Handbuch.pdf` im Kapitel "Tabelle der Parameter für das Signieren" aufgeführt sind.

Wenn Sie der WebEdition mit einer Konfiguration starten wollen, muss in der Datei `sign.gov-ws` für den Aufruf der WebEdition der Parameter `signer.configurationFile` enthalten sein. Der darin angegebene Pfad muss auf eine gültige Konfigurationsdatei zeigen. Ein Beispiel für die Übergabe einer Konfigurationsdatei finden Sie im Handbuch `Governikus-DATA-Boreum-WE-Samples-Handbuch.pdf` im Kapitel "sign.gov-ws mit allen Parametern".

#### Beispiel einer Konfigurationsdatei

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <process>
  <!-- Konfiguration Signaturerstellung (WebEdition) -->
    <sign>
      <!-- Einsehen von Dateien -->
      <fileVisualization>
        <!-- Einsehen von Dateien erzwingen ("true"|"false") -->
        <useVisualization>false</useVisualization>
        <!-- Anzahl einzusehender Dateien in Prozent ( 0 bis 100); nur
             wenn useVisualization=true -->
        <percent>0</percent>
      </fileVisualization>
      <!-- Einstellungen zur sichtbaren PDF-Signatur -->
      <pdfSignatureExtention>
        <!-- Unterzeichnernamen aus Signaturzertifikat entnehmen
             ("true"|"false") -->
        <useSignatureIssuer>true</useSignatureIssuer>
        <!-- Unterzeichnernamen als Freitext oder leer, wenn kein Name
             angezeigt werden soll (nur wenn useSignatureIssuer=false) -->
        <signatureIssuer/>
        <!-- Ort als Freitext -->
        <signatureLocation/>
        <!-- Datum anzeigen ("true"|"false") und  Datumsformat (z.B.
             "dd.MM.yyyy", "MM/dd/yyyy", "EEEE, dd.MMMM.yyyy;
             EEEE=Wochentag, MMMM=Monat in Textform) -->
        <useDate>false</useDate>
        <dateFormat>dd.MM.yyyy kk:mm 'Uhr'</dateFormat>
        <!-- Dynamische("true") oder statische ("false") sichtbare Signatur
             erzeugen; nur wenn visualisationTyp!=NONE -->
        <useDynamicPdfSignature>true</useDynamicPdfSignature>
        <!-- Art der sichtb. Signatur: Nur Text ("TEXT"), Text und Grafik
             ("BOTH"), nur Grafik ("IMAGE"), keine ("NONE")  -->
        <visualisationTyp>TEXT</visualisationTyp>
        <!-- Position Grafik innerhalb der Signatur
             ("ImageLeft"|"ImageRight"|"ImageTop"|"ImageBottom") -->
        <visibleSignatureLayout>ImageLeft</visibleSignatureLayout>
        <!-- Signatur auf erster Seite ("1") oder letzter Seite ("-1") -->
        <page>1</page>
```

```
        <!-- Breite und Hoehe der Signatur in cm -->
        <visualWidth>5.0</visualWidth>
        <visualHeight>3.0</visualHeight>
        <!-- Signaturfeld für die sichtbare Signatur erstellen, falls dieses
noch nicht vorhanden  -->
        <!-- true | false  -->
        <createSignatureField>true</createSignatureField>
         <!-- Groessenverhaeltnis Grafik zu Text (1/2="0.33", 1/3="0.25",
             1/4="0.2" 2/1="0,66") -->
        <imageRatio>0.25</imageRatio>
        <!-- Position der Signatur, relativ zur Seitengroesse (0.0 bis 1.0,
             ausgehend von linker unterer Seitenecke)  -->
        <positionY>0.2</positionY>
        <positionX>0.4</positionX>
        <!-- PDF/A-Kompatibilitaet soll beibehalten werden ("true"|"false")
             -->
        <pdfACompatibility>true</pdfACompatibility>
        <!-- Schriftart ("COURIER"|"HELVETICA"|"TIMES"); wird ignoriert,
             wenn pdfACompatibility=true -->
        <textFont>TIMES</textFont>
        <!-- Schriftfarbe ("Black"|"Gray"|"Red"|"Green"|"Blue"|"Yellow");
             wird ignoriert, wenn pdfACompatibility=true -->
        <textColor>Black</textColor>
        <!-- Automatisches verkleinern der Schriftgroesse erlauben
             ("true"|"false") true empfohlen-->
        <useDynamicTextSize>true</useDynamicTextSize>
        <!-- Schriftgroesse -->
        <textSize>14</textSize>
        <!-- Textformatierung ("left"|"center"|"right") -->
        <textFormatting>center</textFormatting>
        <!-- Pfad oder URL zur Grafik (interne Grafiken:
             "certificate_128_white.gif"  und  "ballpen_128_white.gif"  -->
<imagefile>jar:file:/C:/Program%20Files%20(x86)/Governikus%20KG/Governikus%
20Signer%20WebEdition/lib/signer-
sign.jar!/resources/certificate_128_white.gif
        </imagefile>
      </pdfSignatureExtention>
      <!-- qualified | indetermined | advanced -->
      <signatureLevel>qualified</signatureLevel>
       <!-- Einstellungen Governikus Dienste -->
      <governikus_sc>
        <authService>[keycloak-server]/auth/</authService>
        <authServiceRealm>[governikus-realm]</authServiceRealm>
        <signService>[govsuite-server]/signservice/rest</signService>
        <authServiceSecret>[boreum-client-secret]</authServiceSecret>
        <authServiceClient>boreum-client</authServiceClient>
        <timestampService>[govsuite-
server]/timestampservice/rest</timestampService>
        <timestampServiceProfileID>[governikus-
realm]<timestampServiceProfileID>
      </governikus_sc>
    </sign>
  </process>
  <proxy>
```

```
    <use_pac>false</use_pac>
    <host>10.10.10.10</host>
    <port>8080</port>
    <user>test</user>
    <!-- Base64 UTF-8 kodiert -->
    <password>MTIzNDU2</password>
    <nonProxyHosts/>
  </proxy>
</root>
```

Listing 8: Konfigurationsdatei "Signieren" für die WebEdition

## 3.5 Return-Codes, Fehlerbehandlung, Dateiuploads

Die WebEdition sendet jeden ReturnCode (RC), der während eines Prozessdurchlaufes durch einen Fehler bzw. durch das erfolgreiche verarbeiten einer Datei erstellt wird, an die als Aufrufparameter übergebene RC_URL. Damit kann die Fachanwendung jeden RC des Signiervorganges empfangen und auswerten.

Die Übergabe an die Fachanwendung erfolgt über einen GET und einen POST-Request. Die Übergabeparameter `returnCode`, `returnCodeDescription` und `param0` sind im POST-Request enthalten. Die Übergabe erfolgt zum Zeitpunkt des Auftretens des jeweiligen Ereignisses. Details entnehmen Sie bitte dem Quellcode des Beispiel-RCservlets aus den Samples.

Das Format aller POST-Requests entspricht dem Content-type multipart/form-data. Für jeden übermittelten Parameter wird ein MIME-Abschnitt gesendet, der Name des Parameters ist im `name`-Attribut des Content-Disposition-Headers des jeweiligen Abschnitts angegeben. Der Inhalt des Abschnittes ist der Wert des Parameter.

**Hinweis:** Eine Beispielimplementierung ist im Handbuch „Governikus DATA Boreum Web Edition Samples“ in Kapitel 2.1 „Samples“ im Abschnitt „RCServlet“ beschrieben.

Einige RCs können bei der Auswertung außer Acht gelassen werden. Wird z.B. ein RC 50 `INVALID_PIN` gefolgt von einem RC 3 `FILE_PROCESSED` gesendet, so wurde die PIN beim zweiten Versuch richtig eingegeben und die Datei wurde erfolgreich signiert.

Wird jedoch der RC 50 von einem RC 54 `PIN_INPUT_CANCEL` gefolgt, so hat der Nutzer die erneute PIN-Eingabe abgebrochen und somit wurde die Datei nicht erfolgreich signiert. Es folgt auch kein RC 3 für diese und die folgenden Dateien. Für alle folgenden Dateien wird ein RC 5 `FILE_CANCELLED` gesendet.

Das Ende des gesamten Signiervorgangs (interessant bei der Verarbeitung von mehreren Dateien in einem Vorgang) wird immer durch Return-Code "0" oder "1" gekennzeichnet.

### 3.5.1 Standard ReturnCodes

Die nachfolgende Tabelle enthält alle im Produkt enthaltenen ReturnCodes. Dies schließt auch einige ReturnCodes mit ein, die projektbezogen nicht von Bedeutung sind (z. B. wenn lediglich Softwarezertifikate zur Verwendung kommen).

| Return -Code | Belegung | Bedeutung |
|---|---|---|
| 0 | OK | Der Vorgang {0} wurde erfolgreich durchgeführt |
| 1 | ABORTED | Der Vorgang wurde abgebrochen |
| 3 | FILE_PROCESSED | Die Datei {0} wurde erfolgreich verarbeitet |
| 5 | FILE_CANCELLED | Die Verarbeitung der Datei {0} wurde abgebrochen |
| 10 | MISSING_PARAM | Fehlender Parameter {0} |
| 11 | ILLEGAL_PARAM | Parameterfehler {0} |
| 20 | FILE_NOT_FOUND | Die Datei {0} wurde nicht gefunden |
| 21 | FILE_IN_USE | Die Datei {0} kann nicht gespeichert werden, da sie von einer anderen Anwendung benutzt wird |
| 22 | SOURCE_DIR_EMPTY | Das Verzeichnis {0} ist leer |
| 23 | OUTPUT_DIR_NOT_FOUND | In das Verzeichnis {0} kann nicht geschrieben werden. |
| 24 | CONNECTION_ERROR | Fehler beim Verbinden zur Adresse: {0} |
| 25 | CANNOT_SAVE_OUTPUT | Die Ausgabedatei {0} kann nicht gespeichert werden unter {1} |
| 26 | CANNOT_WRITE_TO_FOLDER | Die Datei {0} kann nicht im Zielverzeichnis gespeichert werden. Bitte überprüfen Sie ihre Sicherheitseinstellungen! |
| 27 | CANNOT_UNZIP | Das ZIP-Archiv konnte nicht entpackt werden |
| 28 | CANNOT_CREATE_ZIP | Das ZIP-Archiv konnte nicht erstellt werden |
| 29 | FILE_IS_EMPTY | Die Datei {0} ist leer |
| 30 | CANNOT_READ_FOLDER | Die Ordner {0} hat keine Leseberechtigung |
| 31 | CANNOT_OPEN_SOURCE | Die Datei {0} konnte weder geöffnet noch gelesen werden |
| 33 | CANNOT_CLOSE_INPUT | Die Eingabedatei {0} kann nicht geschlossen werden. |
| 40 | KEY_NOT_FOUND | Signaturschlüssel (SW-Zertifikat) {0} nicht vorhanden |
| 41 | KEY_EXPIRED | Der Gültigkeitszeitraum des ausgewählten Signaturzertifikats {0} ist überschritten. Das Signieren von Dateien ist nur mit einem gültigen Zertifikat möglich. Bitte wählen Sie ein gültiges Zertifikat aus. |
| 42 | KEY_NOT_YET_VALID | Der gewählte Signaturschlüssel {0} ist noch nicht gültig<br>Der Gültigkeitszeitraum des ausgewählten Signaturzertifikats {0} ist noch nicht erreicht Das Signieren von Dateien ist nur mit einem gültigen Zertifikat möglich. Bitte wählen Sie ein gültiges Zertifikat aus. |
| 43 | CANNOT_LOAD_KEYSTORE | Der Signaturschlüssel konnte nicht geladen werden. Bitte prüfen Sie Ihre Einstellungen. |

| Return -Code | Belegung | Bedeutung |
|---|---|---|
| 44 | NO_KEYSTORE_FILE | Die angegebenen Datei {0} konnte nicht als PKCS#12 interpretiert werden. |
| 45 | CARD_NOT_YET_INITIALIZED | Der gewählte Schlüssel wurde noch nicht freigeschaltet. |
| 46 | CARD_REMOVED | Während des Vorganges wurde entweder die Signaturkarte aus dem Kartenleser genommen oder der Kartenleser wurde vom Rechner getrennt. |
| 47 | WRONG_KEY | Die Datei konnte mit dem gewählten Schlüssel nicht entschlüsselt werden |
| 50 | INVALID_PIN | Die eingegebene PIN ist falsch. {0}. |
| 51 | TOO_MANY_WRONG_PINS | Der Fehlbedienungszähler der Signaturkarte ist abgelaufen |
| 52 | UNKNOWN_SIG_TYPE | Signaturformat wird nicht unterstützt |
| 53 | UNKNOWN_SIG_ALGORITHM | Nicht unterstützter Signaturalgorithmus {0} |
| 54 | PIN_INPUT_CANCEL | Die PIN-Eingabe wurde abgebrochen |
| 55 | SHOWNFILE_CHANGED | Die Datei {0} wurde nach dem letzten Zugriff in Ihrem Dateisystem verändert |
| 57 | UNSUPPORTED_SIG_ALGORITHM | Der von Ihnen gewählte Algorithmus wird durch die gewählte Signaturkarte nicht unterstützt. Bitte wählen Sie unter 'Einstellungen' einen anderen Algorithmus. |
| 59 | HASHEDFILE_CHANGED | Die Datei {0} wurde nach dem herunterladen verändert |
| 81 | UNSUPPORTED_PDF_DIALECT | Die PDF-Datei {0} enthält Elemente, die von der verwendeten Library "iText" (www.lowagie.com) nicht verarbeitet werden können. |
| 82 | PDF_SIGNATURE_VISUALISATION_TOO_BIG | Die sichtbare PDF-Signatur konnte nicht erzeugt werden. Der Text ist für das vorgegebene Textfeld zu groß. Sie sollten den Text verkürzen, eine kleine Schriftgröße wählen oder die Höhe bzw. Breite der Visualisierung vergrößern. |
| 83 | PDF_SIGNATURE_VISUALISATION_WIDTH_TOO_BIG | Die sichtbare PDF-Signatur konnte nicht erzeugt werden. Die Breite der Signatur soll nicht größer sein, als die Breite der PDF-Seite. Sie sollten eine kleine Signaturgröße wählen. |
| 88 | PDF_SIGNATURE_IMAGE_NOT_AVAILABLE | Grafik der erweiterten PDF-Signatur wurde nicht gefunden. |
| 91 | PDF_SIGNATURE_NO_SIGNATURE_FIELD | Signaturfeld für die erweiterte PDF-Signatur wurde nicht gefunden. |
| 96 | MULTIPLE_SIGNATURES_NOT_SUPPORTED | Mehrfachsignatur bei diesem Dokumententyp nicht möglich. |

| Return -Code | Belegung | Bedeutung |
|---|---|---|
| 99 | `UNKNOWN` | Ein unbekannter Fehler ist aufgetreten. Der Vorgang wurde abgebrochen. Bitte wiederholen Sie den aktuellen Vorgang oder informieren Sie ihren Systemadministrator |
| 201 | `AUTH_SERVICE_NOT_CONFIGURED` | Der Authentisierungs-Dienst ist nicht konfiguriert. |
| 202 | `SIGN_SERVICE_NOT_CONFIGURED` | Der Signatur-Dienst ist nicht konfiguriert. |
| 203 | `AUTH_SERVICE_ERROR` | Fehler im Authentisierungsdienst. Bitte kontaktieren Sie den Betreiber des Authentisierungsdienstes. |
| 204 | `TIMESTAMP_SERVICE_ERROR` | Fehler im Zeitstempeldienst.<br>Bitte kontaktieren Sie den Betreiber des Zeitstempeldienstes. |
| 205 | `SIGN_SERVICE_ERROR` | Fehler im Signaturdienst.<br>Bitte kontaktieren Sie den Betreiber des Signaturdienstes. |

Tabelle 2: Übersicht über die Return-Codes

### 3.5.2 Upload von Ergebnisdateien

Wird im Parameter `signer.dest` eine Ziel-URL angegeben, so wird jede Ergebnisdatei nach erfolgreicher Verarbeitung mit einem eigenen HTTP-POST an diese Adresse hochgeladen. Das Format der Upload-POSTs entspricht wiederum dem oben beschriebenen Content-type multipart/form-data, wobei folgende MIME-Abschnitte übertragen werden:

- Parameter `storage_dir`: Zielordner auf dem adressierten Server
- Parameter `size`: Größe der übertragenen Datei
- Parameter `modified`: Datum der Datei
- Parameter `filename`: Inhalt der Datei; ein zusätzliches Attribut `filename` enthält den Dateinamen. Außerdem ist ggf. ein Content-Type der Datei angegeben (z.B. `application/pdf`)

**Hinweis:** Eine Beispielimplementierung ist im Handbuch „Governikus DATA Boreum Web Edition Samples“ in Kapitel 2.1 „Samples“ im Abschnitt `FileSaveServlet` beschrieben.

# 4 Historie Softwareänderungen

## 4.1 Änderungen der DATA Boreum WE Version 10.0.0 gegenüber Governikus Signer WE

1. In DATA Boreum WE `signer.process=sign` (vgl. Signer WE) werden folgende Parameter nicht mehr ausgewertet/verwendet:
   - `serialnumber`
   - `keyStore`
   - `enableDuplicateSignatures`
   - `enablePseudonymSignatures`
   - `attributeCertificateFilename`
   - Die Parameterwerte `advancedPdfInline` und `automaticPdfInline` werden als `pdfinline` behandelt
2. In DATA Boreum `signer.process=encrypt` werden folgende Parameter nicht mehr ausgewertet/verwendet:
   - `serialnumber`
   - `keyStore`
3. In DATA Boreum `signer.process=decrypt` werden folgende Parameter nicht mehr ausgewertet/verwendet:
   - `Serialnumber`

## 4.2 Konfiguration der Governikus Dienste ab Governikus Suite 4.0.0 und DATA Boreum 10.3.0

Siehe Abschnitt in der Konfiguration `<governikus_sc>…</governikus_sc>`

# 5 Administration

In diesem Kapitel finden Sie Erklärungen zu den Themen SSL-Verbindung, Server SSL-Zertifikate, SSL-Client-Authentication, Online-Hilfe, Anpassung der `provider.properties`, JAR-Datei erstellen und signieren sowie Deployment der WebEdition.

## 5.1 Systemanforderungen

### Application-Server (WebEdition)

Für den Application-Server, der die WebEdition bereitstellt, gelten keine besonderen Auflagen hinsichtlich der Hard- und Software. Die Governikus KG empfiehlt als Application-Server den Einsatz eines Apache Tomcat Webservers in Version 8.5 oder 9.

### Application-Server (Fachanwendung)

Für den Application-Server, der die Kommunikation mit der WebEdition übernimmt, müssen folgende Punkte beachtet werden:

- Das URI-Encoding muss auf "UTF-8" eingestellt werden, falls Dateien mit Umlauten der WebEdition zum Download übergeben werden, siehe dazu Kapitel 5.2 URI-Encoding.
- Das Dateisystem muss mit dem Zeichensatz "ISO-8859-1" eingestellt sein, damit das `FileSaveServlet` aus `GovernikusDataBoreumWebEditionSample.war` die Dateien mit Umlauten richtig speichern kann.
- Für eine SSL-Verbindung mit dem Server siehe Kapitel 5.3.

## 5.2 URI-Encoding

Um das URI-Encoding des Webservers einzustellen, muss in der Datei `conf/server.xml` des Webservers der HTTP- bzw. SSL-Connector um den Parameter `URIEncoding="UTF-8"` erweitert werden.

### Tomcat conf/server.xml

```
<Connector port="8080" maxHttpHeaderSize="8192"  maxThreads="150"
minSpareThreads="25" maxSpareThreads="75" enableLookups="false"
redirectPort="8443" acceptCount="100" connectionTimeout="20000"
disableUploadTimeout="true"  URIEncoding="UTF-8"/>
```

**Hinweis:** Diese Änderung beeinträchtigt den Betrieb eines auf demselben Server betriebenen Governikus Systems nicht. Um Auswirkungen auf andere Web-Anwendungen auszuschließen, kann auch, anstelle der Anpassung des bestehenden Connectors, ein weiterer Connector mit einem anderen Port (und einem anderem URI-Encoding) angelegt werden.

## 5.3 SSL Verbindung

Die WebEdition kann sich auch über SSL-Client-Authentication mit dem Server verbinden.

**Achtung**: Die SSL-Client-Authentication gegenüber dem Server muss mit einem offiziellen Zertifikat gesichert werden, z.B. vom Anbieter Verisign Inc. – es werden nur offizielle Zertifikate, die von Java als vertrauenswürdig anerkannt werden, akzeptiert (siehe dazu Java in der Datei „cacerts“ im Ordner `C:\Program Files (x86)\Governikus KG\DATA Boreum WebEdition\jre\lib\security`).

## 5.4 Governikus DATA Boreum WebEdition deployen

Die WAR-Datei `GovernikusDataBoreumWebEdition.war` muss in das Deploy-Verzeichnis des Webservers kopiert werden. Das Deploy-Verzeichnis ist bei den unterstützten Webservern in den folgenden Unterverzeichnissen zu finden.

### JBoss

`server/standalone/deployments/`

### Tomcat

`webapps/`

Sobald der Webserver gestartet wurde, wird die Governikus DATA Boreum WebEdition unter der Adresse `http://<server>:<port>/GovernikusDataBoreumWebEdition` zur Verfügung gestellt.

Um die Beispiele zu nutzen, muss die folgende WAR-Datei ebenfalls deployed werden:

`GovernikusDataBoreumWebEditionSample.war`

Der Vorgang ist analog zu oben. Unter der folgenden Adresse steht dann die Startseite zur Verfügung: `http://<server>:<port>/ GovernikusDataBoreumWebEditionSample`

**Hinweis**: Wenn Sie eine bereits installierte Version aktualisieren möchten, löschen Sie die alte Version vom Webserver, bevor Sie die neue Version aufspielen. Achten Sie darauf, dass beim Entfernen der .war-Datei die zugehörige Webanwendung komplett vom Webserver gelöscht wurde.

**Achtung**: Die Beispiele, die in dieser Datei beschrieben sind: `GovernikusDataBoreumWebEditionSample.zip`, enthalten als Default-Wert den Server `localhost`. Das Ändern dieses Defaults ist in diesem Handbuch beschrieben:

`Governikus-DATA-Boreum-WE-Samples-Handbuch.pdf`.

# 6 Empfehlungen für den Betrieb

Um qualifizierte elektronische Signaturen und Siegel sicher, korrekt und vertrauenswürdig anbringen zu können, sind besondere Anforderungen an die Software selbst und die Einsatzumgebung zu stellen. Eine notwendige hohe Sicherheit gegenüber potenziellen Bedrohungen muss immer komplett sein, d.h. sie wird immer durch einen "Mix" von Sicherheitsvorkehrungen in der Software selbst und in der Einsatzumgebung komplettiert.

## 6.1 Empfohlene Anforderungen an die Einsatzumgebung

Folgende Empfehlungen bezüglichen der räumlichen und technischen Gegebenheiten bestehen:

- Anbindung an ein Netzwerk:
  - Netzwerkverbindungen sollten so abgesichert werden, dass Angriffe erkannt bzw. unterbunden werden - z. B. durch eine geeignet konfigurierte Firewall und durch die Verwendung geeigneter Anti-Viren-Programme.
- Sicherheit der IT-Plattform und Programme:
  - Von der Hardware, auf der Governikus DATA Boreum betrieben wird, dürfen keine Angriffe ausgehen. Installierte Software darf nicht böswillig manipuliert oder verändert werden. Maßnahmen gegen Viren oder trojanische Pferde sollten regelmäßig geprüft und aktualisiert werden.
- Schutz vor manuellem Zugriff Unbefugter und Datenaustausch per Datenträger. Folgende Empfehlungen bestehen bezüglich der baulichen, personellen und organisatorischen Anforderungen:
  - Unbefugte dürfen keinen Zugriff auf den PC haben, auf dem Governikus DATA Boreum betrieben wird. Dies sollte ausgeschlossen oder zumindest mit hoher Sicherheit erkennbar sein - beispielsweise durch Sperren des Rechners oder Verschließen des Raumes bei Abwesenheit.
  - Beim Übertragen von Daten, die auf Datenträgern vorliegen sollte - z. B. durch die Verwendung geeigneter Anti-Viren-Programme - sichergestellt werden, dass keine Viren oder trojanische Pferde übertragen werden können.

## 6.2 Empfehlungen für den sicheren Betrieb

Passwörter sollten hinreichend komplex sein (z.B. für die Anmeldung am Betriebssystem), d. h. nutzen Sie

- keine Trivialpasswörter (z. B. "BBBBBBBB" oder "12345678"),
- Passwörter mit mindestens einem Zeichen pro Passwort, das kein Buchstabe ist (Sonderzeichen oder Zahl),
- Passwörter, die mindestens 8 Zeichen lang sind.
- Passwörter müssen geheim gehalten werden. Stellen Sie sicher, dass niemand Ihr Passwort kennt.

Das persönliche Verzeichnis (Profil-Verzeichnis) der Benutzenden, die Governikus DATA Boreum WE betreiben, sollte gegen Manipulationen durch Unbefugte geschützt werden - z.B. durch Einschränkung der Zugriffsberechtigung.

Vor der Installation der Software ist die Integrität des Installationspakets über einen Vergleich eines vor Ort erstellten Hashwerts mit dem durch die Governikus KG veröffentlichten Hashwert zu prüfen.

## 6.3 Technische Anforderungen

Die von Governikus DATA Boreum WE unterstützte Hard- und Software ist im Handbuch "Governikus DATA Boreum WE - Systemanforderungen" beschrieben. Zur Ausstattung für die Erstellung von qualifizierten elektronischen Signaturen und Siegeln zählen die folgenden Karten und Kartenleser:

Es können qualifizierte elektronische Signaturerstellungseinheiten sowie qualifizierte Siegeleinheiten verwendet werden, die durch qualifizierte Vertrauensdiensteanbieter aus Deutschland herausgegeben werden und mit denen man eine QES erzeugen kann.

Seit dem 01.07.2016 gilt in Deutschland die eIDAS-Verordnung, die keine Zertifizierung von geeigneten Chipkartenlesern regelt.

## 6.4 Anforderungen an die Konfiguration

Hinsichtlich der Konfiguration müssen Sie folgende Anforderungen berücksichtigen:

**Zeitstempeldienst**: Für das Anbringen von qualifizierte Zeitstempeln ist ein vertrauenswürdiger Zeitstempeldiensteanbieter einzurichten, der die qualifizierten Zeitstempel über diesen Zeitstempeldiensteanbieter erstellen lässt. Die Verbindungsdaten erhalten Sie über Ihren Governikus Suite Administrator.

# 7 Verzeichnisse

## Abbildungen

Abbildung 1: Ablaufdiagramm ....4

## Listings

Listing 1: Inhalt der WebEdition Produktlizenzdatei für das Signieren ....7
Listing 2: Inhalt der WebEdition Produktlizenzdatei für das Ver- und Entschlüsseln ....8
Listing 3: Beispiel einer Lizenzdatei für Personen....10
Listing 4: Beispiel einer interpretierten Lizenzdatei ("Gesamtlizenz”) ....11
Listing 5: Inhalt der enthaltenen Standard-Lizenzdatei für Personen....11
Listing 6: Inhalt der Datei `start.jsp` ....17
Listing 7: Beispiel für eine `.gov-ws` Datei....17
Listing 8: Konfigurationsdatei "Signieren" für die WebEdition ....21

## Tabellen

Tabelle 1: Übersicht über die Parameter der Lizenzdatei ....10
Tabelle 2: Übersicht über die Return-Codes ....24