

DZAB Access Service

Archive Service SecDocs

Administration und Bedienung

Benutzerhandbuch

Stand der Beschreibung

DZAB Access Service 22.00.01.00

Archive Service SecDocs V3.1B

Ausgabe November 2021

1 Der DZAB Access Service

Zur rechtskonformen und revisionssicheren Langzeitarchivierung digitaler Objekte (Dateien, Akten, Vorgänge, Dokumente, ...) stellt das Bundesarchiv den Ministerien, Behörden und Gerichten des Bundes das Digitale Zwischenarchiv des Bundes (DZAB) bereit. Das zugrundeliegende eArchiv wird von der Bundesagentur für Arbeit (BA) im Rahmen der IT-Konsolidierung des Bundes betrieben. Es beinhaltet den Access Service als Frontend und die Fujitsu Software SecDocs als Archive Service.

Dieser Band enthält das Benutzerhandbuch zur Administration und Bedienung der Fujitsu Software SecDocs.

FUJITSU Software

SecDocs

Administration und Bedienung

Benutzerhandbuch

Stand der Beschreibung
V3.1B

Ausgabe November 2021

Kritik... Anregungen... Korrekturen...

Die Redaktion ist interessiert an Ihren Kommentaren zu diesem Handbuch. Ihre Rückmeldungen helfen uns, die Dokumentation zu optimieren und auf Ihre Wünsche und Bedürfnisse abzustimmen.

Sie können uns Ihre Kommentare per E-Mail an bs2000services@ts.fujitsu.com senden.

Zertifizierte Dokumentation nach DIN EN ISO 9001:2015

Um eine gleichbleibend hohe Qualität und Anwenderfreundlichkeit zu gewährleisten, wurde diese Dokumentation nach den Vorgaben eines Qualitätsmanagementsystems erstellt, welches die Forderungen der DIN EN ISO 9001:2015 erfüllt.

Copyright und Handelsmarken

Copyright © 2021 Fujitsu Technology Solutions GmbH.

Alle Rechte vorbehalten.

Liefermöglichkeiten und technische Änderungen vorbehalten.

Alle verwendeten Hard- und Softwarenamen sind Handelsnamen und/oder Warenzeichen der jeweiligen Hersteller.

Inhaltsverzeichnis

SecDocs Administration und Bedienung	9
1 Einführung	10
1.1 Konzept und Zielgruppe des Handbuchs	11
1.2 Änderungen gegenüber der vorigen Version	12
1.3 Darstellungsmittel	13
2 Konzepte und Funktionen	14
2.1 Grundlagen	15
2.1.1 TR-03125 und ArchiSig-Konzept	16
2.1.2 Kryptografische Verfahren	17
2.2 Architektur	20
2.2.1 Komponenten von SecDocs	21
2.2.2 Software-Umgebung	23
2.2.3 Storage-Systeme	24
2.2.4 Ablagestruktur im Archiv	25
2.2.4.1 Ablagestruktur für externe Datenobjekte	27
2.3 Multi-Node-Betrieb	28
2.3.1 Architektur	29
2.3.2 Konfiguration des Verbunds	30
2.3.2.1 Die Konfigurationsdatei secdocs.properties	31
2.3.2.2 Konfigurationsparameter des CryptoModule	32
2.4 Archivierung	33
2.4.1 Prüfung elektronischer Signaturen	34
2.4.2 Gruppieren und Versiegeln der Datenobjekte gemäß ArchiSig-Konzept	35
2.4.3 Erzeugen des Evidence Records	36
2.4.4 Beschreibung eines Datenobjekts	38
2.4.4.1 Submission Data Object	39
2.4.4.2 Client Object Identifier (COID)	40
2.4.4.3 Versionieren von Dokumenten	41
2.4.4.4 Überschreiben von SDOs	42
2.4.4.5 Validierung	43
2.4.4.6 Informationen zur Adressierung von Daten im SDO	44
2.5 Recherche	45
2.5.1 Recherche mittels Navigation	46
2.6 Web-Services	47
2.6.1 Mandantenfähigkeit	48
2.6.2 Berechtigungskonzept	49
2.6.2.1 Vorgegebene Rolle Archivar	50

2.6.2.2 Organisationspezifisches Rollenkonzept	51
2.6.2.3 Vorgehen beim Umstellen auf organisationspezifische Rollen	53
2.6.3 Testmandant	54
2.6.3.1 Anlegen eines Testmandanten	55
2.6.3.2 Vom Testmandanten zum produktiven Mandanten	56
2.6.3.3 Vollständiges Löschen aller SDOs eines Testmandanten	57
2.6.3.4 Löschen eines Testmandanten	58
2.6.4 Web-Service für die Archivierung	59
2.6.5 Web-Services für die Administration	60
2.6.6 Hinweise zur Migration von Web-Service-Clients	61
2.6.6.1 Die Endpoint-URLs, Namespaces und Schemata der V3.1	62
2.7 Unterstützung von Policies	65
2.8 Sicherer Betrieb	67
2.8.1 Anforderungen für den sicheren Betrieb	68
2.8.2 Logging	69
3 Web-Service für die Client-Anwendung	70
3.1 SOAP-Nachrichten	71
3.1.1 Request und Response	72
3.1.2 Fault-Message	74
3.1.3 Zugangsprüfung	80
3.2 Request-Header	82
3.3 Request- und Response-Body	89
3.3.1 Operation getAOID	90
3.3.2 Operation getAOIDWithRef	92
3.3.3 Operation submitSDO	99
3.3.4 Operation replaceSDO	103
3.3.5 Operation retrieveSDO	106
3.3.6 Operation deleteSDO	109
3.3.7 Operation forceDeleteSDO	111
3.3.8 Operation statusSDO	113
3.3.9 Operation listSDOVersions	124
3.3.10 Operation retrieveMetaData	127
3.3.11 Operation metaDataSDO	130
3.3.12 Operation requestForEvidence	135
3.3.13 Operation navigate	141
3.3.14 Operation getAccountingData	151
3.3.15 Operation moveSDO	154
3.3.16 Operation setExpirationDateTimeSDO	157
3.3.17 Operation forceDeferredSDO	161
3.3.18 Operation getSchemaDataSDO	164
3.3.19 Operation getVersion	168

3.4 Tools	170
3.4.1 Hash-Wert erzeugen (Skript mksha)	171
4 Beschreibung der Datenobjekte	173
4.1 SDO-Struktur	174
4.2 Adressierung von Daten im SDO	180
4.2.1 Filtersyntax	181
4.2.1.1 Element <filter>	183
4.2.1.2 Element <namespace>	185
4.2.1.3 Element <element>	188
4.2.1.4 Element <node>	194
4.2.1.5 Element <alias>	197
4.2.2 Verwendung von Schlüsseln	198
4.2.2.1 Definition von Schlüsselwerten	201
4.2.2.2 Verwendung von Schlüsselwerten	202
4.2.2.3 \$DateFormat	204
4.2.2.4 \$ExpirationDate	205
4.2.2.5 \$RetentionPeriod	207
4.2.2.6 Kombination von \$ExpirationDate und \$RetentionPeriod	209
4.2.2.7 \$Content	213
4.2.2.8 \$ContentRef	214
4.2.2.9 \$ExternalRef	215
4.2.2.10 \$ContentCode	216
4.2.2.11 \$ContentType	217
4.2.2.12 \$DataNode	218
4.2.2.13 \$Signature	220
4.2.2.14 \$SignatureEmbedded	221
4.2.2.15 \$SignatureDetached	222
4.2.2.16 \$SignatureQualityLevel	223
4.2.2.17 \$SignatureType	224
4.2.2.18 \$SignatureVerification	225
4.2.2.19 \$Index	226
4.2.2.20 \$Subject	229
4.2.2.21 \$SDOPath	230
4.2.2.22 \$SoftWorm	234
4.2.3 Beispiel für eine Filterdefinition	235
4.3 Datenobjekt für die Archivierung	237
5 Archiv- und Mandantenadministration	240
5.1 Web-Service ArchiveAdminService	241
5.1.1 Zugangsprüfung	242
5.1.2 Request-Header	243
5.1.3 Request- und Response-Body	247

5.1.3.1 Operation createTSP	248
5.1.3.2 Operation createMandant	256
5.1.3.3 Operation setCredentials	270
5.1.3.4 Operation getTSPs	274
5.1.3.5 Operation updateTSP	276
5.1.3.6 Operation getMandants	281
5.1.3.7 Operation getHashAlgorithms	284
5.1.3.8 Operation getSignatureAlgorithms	288
5.1.3.9 Operation getVersion	290
5.1.3.10 Operation getAccountingData	292
5.1.3.11 Operation getStatisticalData	296
5.1.3.12 Operation deleteTestMandant	302
5.1.3.13 Operation performAction	304
5.1.3.14 Operation getArchiveInfo	308
5.2 Web-Service MandantAdminService	309
5.2.1 Zugangsprüfung	310
5.2.2 Request-Header	311
5.2.3 Request- und Response-Body	317
5.2.3.1 Operation createSDOType	318
5.2.3.2 Operation modifySDOType	324
5.2.3.3 Operation updateMandant	328
5.2.3.4 Operation createOrganisation	331
5.2.3.5 Operation updateOrganisation	335
5.2.3.6 Operation setCredentials	337
5.2.3.7 Operation renewHashAlgorithm	341
5.2.3.8 Operation renewTSPSignature	343
5.2.3.9 Operation getSDOTypes	345
5.2.3.10 Operation getMandantProperties	347
5.2.3.11 Operation getOrganisations	349
5.2.3.12 Operation getTSPs	352
5.2.3.13 Operation getHashAlgorithms	354
5.2.3.14 Operation getSignatureAlgorithms	356
5.2.3.15 Operation getVersion	358
5.2.3.16 Operation getAccountingData	360
5.2.3.17 Operation createPrivilege	363
5.2.3.18 Operation updatePrivilege	365
5.2.3.19 Operation deletePrivileges	367
5.2.3.20 Operation getPrivileges	369
5.2.3.21 Operation createRole	371
5.2.3.22 Operation updateRole	376
5.2.3.23 Operation deleteRoles	378

5.2.3.24 Operation getRoles	380
5.2.3.25 Operation deleteSDOType	385
5.2.3.26 Operation performAction	387
5.2.3.27 Operation getArchiveInfo	393
5.2.3.28 Operation clearAOIDs	401
5.2.3.29 Operation convertTestMandant	402
5.2.3.30 Operation getArchivingOperations	404
5.2.3.31 Operation getAuditLogFileNames	407
5.2.3.32 Operation getAuditLogFile	409
5.3 Tools	412
5.3.1 Recovery (Skript recoverFromStorage)	413
5.3.2 Daten gelöschter AOIDs entfernen (Skript purgeData)	420
6 Externe Datenobjekte	430
6.1 Arbeiten mit externen Datenobjekten	432
6.1.1 Archivieren eines externen Datenobjekts	433
6.1.2 Lesen eines archivierten externen Datenobjekts	436
6.1.3 Löschen eines archivierten externen Datenobjekts	437
6.2 Integrierter SFTP-Server	438
6.2.1 Installation und Konfiguration	439
6.2.2 Authentisierung	440
6.2.3 Öffnen einer SFTP-Sitzung	441
6.2.4 Datenübertragung	442
6.2.5 Kommandos für den SFTP-Server	443
6.2.6 Error-Codes an der Programmschnittstelle	444
6.3 Schritt-für-Schritt-Anleitung	445
6.3.1 Vorbereitungen	446
6.3.2 Archivieren eines externen Datenobjekts	447
6.3.3 Lesen eines archivierten externen Datenobjekts	451
7 Schritt-für-Schritt-Anleitungen	453
7.1 Von den Daten zum SDO-Typ	454
7.1.1 Datenumfang erfassen und strukturieren	455
7.1.2 Struktur des SDOs als XML-Schema definieren	456
7.1.3 Filterdefinition erstellen	457
7.2 SDO-Typ-spezifische Operationen des Web-Service ArchivingSRService ..	460
7.3 Archiv einrichten	466
7.3.1 TSP bekannt machen und Mandanten einrichten	467
7.3.2 SDO-Typ registrieren und Organisationseinheit anlegen	469
7.4 Dokumente archivieren	471
8 Inbetriebnahme und Überwachung	473
8.1 Installation und Konfiguration	474
8.1.1 Installation	475

8.1.2 Anforderungen für den sicheren Betrieb	476
8.1.3 Konfigurationsdatei secdocs.properties	477
8.1.4 Mandantenspezifische Konfigurationsparameter	492
8.2 Storage-System anbinden	494
8.3 Accounting	496
8.4 Erheben von Statistikdaten	498
8.4.1 Messwerte	499
8.5 Logging und Fehlerbehandlung	500
8.5.1 Audit-Logging	501
8.5.1.1 Die Audit-Log-Datei	502
8.5.1.2 Header eines Audit-Logging-Eintrags	503
8.5.1.3 Strukturierte Daten eines Audit-Logging-Eintrags	504
8.5.1.4 Beispiele für Logging-Einträge in der Audit-Log-Datei	507
8.5.1.5 Wechsel der Audit-Log-Dateien	508
8.5.1.6 Überwachung des Storage-Systems	509
8.5.1.7 Evidence Records für mandantenspezifische Audit-Log-Dateien	510
8.5.2 SecDocs-Logging	512
8.5.3 Zusätzliches Monitoring	513
8.5.4 Fehlermeldungen	514
8.5.5 Internationalisierung	515
8.5.5.1 Meldungskataloge	516
8.5.5.2 Einbringen eigener Meldungskataloge	517
8.5.5.3 Sprachauswahl	518
9 Fachwörter	519
10 Abkürzungen	525
11 Literatur	527

1 Einführung

Immer mehr Geschäftsprozesse werden elektronisch abgebildet, weil so die Kosten gesenkt und die Verwaltungsvorgänge nachhaltig beschleunigt werden können. Entsprechend verdrängen immer mehr elektronische Dokumente die in der Handhabung teuren Papierbelege.

Elektronische Dokumente müssen über die gleiche dauerhafte Beweiskraft verfügen und ebenso vertrauenswürdig sein wie die Papierdokumente, um Geschäftsprozesse rechtlich abzusichern. Außerdem muss der Nachweis über ihre Integrität und Authentizität jederzeit – teilweise über 100 Jahre – erbracht werden können.

SecDocs ist eine Archiv-Middleware für elektronische Dokumente und bietet:

- Langzeitarchivierung
- Konzepte der BSI Technischen Richtlinie TR-VELS / TR-ESOR
- Erhaltung der Beweiswerte elektronisch signierter Dokumente, d.h. SecDocs übernimmt die Aufgaben der Übersignatur autonom
- Hohe gerichtsverwertbare Beweiskraft bei dauerhaft niedrigen Betriebskosten
- Einfache Integration in eine Vielzahl von IT-Umgebungen und Fachverfahren
- Nachweis der Integrität des Dokuments mindestens seit dem Zeitpunkt der Archivierung
- Unterstützung unterschiedlicher Betriebsmodelle (stand-alone, public cloud, community cloud, private cloud)

SecDocs basiert auf offenen Standards. SecDocs ist eine Service Oriented Architecture-Lösung (SOA-Lösung) und befreit die öffentliche Verwaltung und Unternehmen mit hohen Compliance-Vorgaben vom Umgang mit komplexen elektronischen Signaturen zur Beweissicherung.

1.1 Konzept und Zielgruppe des Handbuchs

Dieses Handbuch wendet sich an Programmierer und Systemverwalter, die

- das Anwendungskonzept für SecDocs entwickeln,
- die Fachanwendung an SecDocs anbinden sowie die Datenobjekte und die dazu gehörenden Filterbeschreibungen erstellen,
- das SecDocs-Archiv und die Mandanten administrieren,
- SecDocs betreiben und das System administrieren.

Für das Verständnis des Handbuchs werden folgende Kenntnisse vorausgesetzt:

- Linux-Betriebssystem und WildFly Application Server
- allgemeine XML-Kenntnisse
- Administration des jeweiligen Speichersystems
- benötigte Datenbank-Software
- Grundlagen der Langzeitarchivierung

Das [Kapitel „Konzepte und Funktionen“](#) gibt einen kurzen Einstieg in die Grundlagen der elektronischen Langzeitarchivierung und in die Funktionen von SecDocs, die dieses Konzept realisieren.

Das [Kapitel „Web-Service für die Client-Anwendung“](#) beschreibt die Arbeit mit dem Web-Service ArchivingService. Dieser Web-Service stellt die Operationen für die Archivierung von Dokumenten zur Verfügung.

Das [Kapitel „Beschreibung der Datenobjekte“](#) beschreibt, wie Sie ein zu archivierendes Datenobjekt als XML-Dokument aufbauen und die Metadaten definieren.

Das [Kapitel „Archiv- und Mandantenadministration“](#) beschreibt die Administration des Archivs in seiner Gesamtheit und die Administration der spezifischen Archivbereiche für die Mandanten.

Das [Kapitel „Externe Datenobjekte“](#) beschreibt, wie große Dokumente als externe Datenobjekte archiviert werden können.

Das [Kapitel „Schritt-für-Schritt-Anleitungen“](#) beschreibt, wie zentrale Arbeitsabläufe in SecDocs aufeinander aufbauen.

Das [Kapitel „Inbetriebnahme und Überwachung“](#) beschreibt die Konfiguration und Inbetriebnahme von SecDocs sowie die verschiedenen Logging-Möglichkeiten, die SecDocs zur Verfügung stellt.



Die Installation von SecDocs sowie die zugehörigen Software-Voraussetzungen sind abhängig vom aktuellen Release. Fujitsu liefert mit jedem Release von SecDocs eine ausführliche Installationsbeschreibung. Dieses Dokument beschreibt die Software-Voraussetzungen für die Installation von SecDocs, den Lieferumfang, das Bereitstellen der Ablaufumgebung und die Konfiguration von SecDocs.

1.2 Änderungen gegenüber der vorigen Version

Dieser Abschnitt beschreibt die Änderungen in SecDocs V3.1 gegenüber der Handbuch-Ausgabe von Mai 2015.

1. In SecDocs erfolgt die Prüfung der Signaturen gemäß der europäischen eIDAS-Verordnung. Zu diesem Zweck wurde der OpenLimit V3-Server durch das OpenLimit CryptoModule ersetzt.
2. Mit der Unterstützung der eIDAS-Richtlinie hat sich die Prüfung von Signaturen und Zeitstempeln geändert. Insbesondere sieht die Richtlinie vor, dass die Bewertung der Prüfung von Zeitstempeln und Signaturen durch Policies gesteuert werden kann. SecDocs unterstützt mit der Version 3.1 die Verwendung solcher Policies.
3. Standardmäßig bettet SecDocs beim Versiegeln die RevocationInfo und Zertifikate der Zeitstempelanbieters nicht in den Evidence-Record ein. Dies geschieht erst bei der Signaturerneuerung (Renew). Künftig kann man bei der Erstellung eines TSPs festlegen, dass die Einbettung der RevocationInfo bereits beim Versiegeln vorgenommen wird ([siehe Operation createTSP](#)).
4. Softworm wird beim XAIP-Mandanten nicht mehr unterstützt. Das Verwenden eines Softworm-Speichers war bei XAIP auch in der Version 3.0 nicht sinnvoll, da es die Versiegelung von Dokumenten verhindert, da der Evidence-Record in das vorhandene XAIP eingebettet werden muss. In der 3.1. wird das Element SoftWorm bei der Funktion createMandantXAIP nicht mehr unterstützt.



Eine detaillierte Liste von Änderungen und Verbesserungen entnehmen Sie bitte der jeweils aktuellen Freigabemitteilung.

1.3 Darstellungsmittel

In diesem Handbuch wird folgende formale Darstellung verwendet:

Schreibmaschinenschrift

 feste Angaben, die genau in dieser Form ein- oder ausgegeben werden, wie z.B. Schlüsselwörter, URLs oder Dateinamen.

Kursive Schrift

 variable Teile, für die Sie konkrete Angaben einsetzen müssen.

alternative1 | *alternative2*

 Alternativen; die senkrechten Striche dürfen nicht angegeben werden.



Verweis auf detaillierte Informationen zum jeweiligen Thema.



Hinweistexte.



Warnhinweise.

2 Konzepte und Funktionen

Elektronische Dokumente erfordern über ihren gesamten Lebenszyklus andere Verfahren als Papierdokumente. Nur wenn alle Verarbeitungsschritte mit mindestens der gleichen Qualität auf digitalem Weg erledigt werden können, ist eine Ablösung papiergebundener Prozesse durch elektronische Dokumente erfolversprechend.

Schon das Lesen und Bearbeiten elektronischer Dokumente erfordert technische Hilfsmittel, die heute allerdings vom PC bis zum Smartphone allgegenwärtig sind. Auch zum Nachweis der Integrität und Authentizität elektronischer Dokumente existiert in Form digitaler Signaturen eine Lösung, die als gleichwertig mit der traditionellen Unterschrift angesehen werden kann.

Der fortlaufende technische Fortschritt seit der Einführung elektronischer Dokumente bringt aber weitere Herausforderungen mit sich: Während Papierdokumente auch nach langer Zeit noch genauso gelesen werden können wie unmittelbar nach ihrer Erstellung, können Dateien oft schon nach der ersten technologischen Migration nicht mehr geöffnet oder gelesen werden. Auch der Wert einer digitalen Signatur nimmt im Lauf der Jahre ab, weil die stets steigende CPU-Leistung die Sicherheit der kryptografischen Algorithmen kontinuierlich verringert. Genau diesen beiden Herausforderungen kann mit SecDocs begegnet werden.

Das folgende Kapitel bietet Ihnen einen kurzen Einstieg in die Grundlagen der elektronischen Langzeitarchivierung und in die Funktionen von SecDocs, die dieses Konzept realisieren.

2.1 Grundlagen

Fujitsu SecDocs basiert auf der Technischen Richtlinie „Vertrauenswürdige elektronische Langzeitspeicherung“ (TR-03125) des Bundesamtes für Sicherheit in der Informationstechnologie (BSI). Die Integrität der Dokumente wird mit kryptografischen Verfahren nachgewiesen.

2.1.1 TR-03125 und ArchiSig-Konzept

Die Implementierung von SecDocs orientiert sich an der TR-03125 (V1.2.1) des BSI und dem „Protection Profile for an ArchiSafe Compliant Middleware for Enabling the Long-Term Preservation of Electronic Documents“ (ACM_PP). Die TR-03125 enthält Anforderungen und beschreibt ein Referenzmodell mit einer modularen Struktur für die dauerhafte Ablage elektronischer Daten und Dokumente im Rahmen der gesetzlichen Aufbewahrungsfristen.

ArchiSig war ein Verbundprojekt, das vom damaligen Bundesministerium für Wirtschaft und Technologie (BMWi) gefördert wurde. Im Rahmen von ArchiSig wurden aus den allgemeinen gesetzlichen Regelungen konkrete rechtliche Anforderungen abgeleitet. Diese Anforderungen an Systeme zur langfristigen Aufbewahrung elektronisch signierter Dokumente wurden prototypisch implementiert. Es konnte erstmalig gezeigt werden, dass die Langzeitaufbewahrung elektronisch signierter Dokumente gesetzeskonform, performant und akzeptabel umgesetzt werden kann.

SecDocs verwendet das ArchiSig-Konzept, um eine größere Anzahl von Archivobjekten mit einem gemeinsamen Zeitstempel zu versehen und sichert diese mit Hilfe der daraus abgeleiteten reduzierten Hash-Bäume (siehe [Abschnitt „Kryptografische Verfahren“](#) unten). Das ArchiSig-Konzept beschreibt ein Verfahren für die sichere und beweiskräftige langfristige Archivierung von elektronischen Dokumenten. Das Konzept und die Umsetzung stehen im Einklang mit der im internationalen Vergleich führenden deutschen Gesetzgebung.

2.1.2 Kryptografische Verfahren

SecDocs ermöglicht den Nachweis der Integrität (Unverändertheit) eines Dokuments über die gesamte Lebensdauer des Dokuments im Archiv. Dazu verwendet SecDocs digital signierte Zeitstempel (digitally signed timestamps).

Dabei bewahrt SecDocs auch den Wert von Authentizitätsnachweisen (Ursprung und Echtheit), die als digitale Signaturen einem Dokument beigelegt sind: Zum Zeitpunkt der Archivierung eines Dokuments werden die Signaturen geprüft und die resultierenden Prüfberichte zusammen mit dem Dokument entsprechend dem ArchiSig-Konzept versiegelt.

Das ArchiSig-Konzept ermöglicht es, eine Vielzahl von Dokumenten mit einem einzigen Zeitstempel zu versiegeln, wobei weiterhin die Integrität jedes einzelnen Dokuments verifiziert werden kann.



Die Beschreibung dieses Verfahrens finden Sie im [Abschnitt „Gruppieren und Versiegeln der Datenobjekte gemäß ArchiSig-Konzept“](#).

Signatur

Eine Signatur bezieht sich auf die Richtigkeit des Inhalts für ein Dokument. Man spricht daher auch von einer Willenserklärung. Elektronische Signaturen werden dazu verwendet, die ordnungsgemäße Durchführung einer Aktivität zu bestätigen, z.B. das ordnungsgemäße Scannen von Dokumenten durch den signierenden Sachbearbeiter.

Signaturarten des Gesetzgebers

Die EU hat in der eIDAS-VO ([Verordnung \(EU\) Nr. 910/2014](#)) verschiedene Signaturarten mit folgenden Sicherheitsstufen definiert:

- Elektronische Signaturen:
„Elektronische Signatur“ sind Daten in elektronischer Form, die anderen elektronischen Daten beigelegt oder logisch mit ihnen verbunden werden und die der Unterzeichner zum Unterzeichnen verwendet.
- Fortgeschrittene elektronische Signaturen:
Eine fortgeschrittene elektronische Signatur erfüllt alle folgenden Anforderungen:
 1. Sie ist eindeutig dem Unterzeichner zugeordnet.
 2. Sie ermöglicht die Identifizierung des Unterzeichners.
 3. Sie wird unter Verwendung elektronischer Signaturerstellungsdaten erstellt, die der Unterzeichner mit einem hohen Maß an Vertrauen unter seiner alleinigen Kontrolle verwenden kann.
 4. Sie ist so mit den auf diese Weise unterzeichneten Daten verbunden, dass eine nachträgliche Veränderung der Daten erkannt werden kann.
- Qualifizierte elektronische Signaturen (QES):
„Qualifizierte elektronische Signatur“ ist eine fortgeschrittene elektronische Signatur, die von einer qualifizierten elektronischen Signaturerstellungseinheit erstellt wurde und auf einem qualifizierten Zertifikat für elektronische Signaturen beruht.

Zeitstempel

Ein Zeitstempel (Timestamp) wird technisch wie eine elektronische Signatur erstellt. Der Zeitstempel ist jedoch keine personengebundene Signatur (wie z.B. eine Willenserklärung), sondern der Nachweis, dass ein bestimmtes elektronisches Dokument zu einem bestimmten Zeitpunkt vorgelegen hat. Zeitstempel werden entweder online von Zeitstempeldiensten erstellt oder von entsprechenden Servern, die im Sinne einer Black Box ins Netz gestellt werden. Zeitstempel werden im Allgemeinen automatisiert wie Massensignaturen erstellt.

Die Datenstruktur eines Zeitstempels hat unter anderem folgende wesentliche Inhalte:

- Das Erstellungsdatum und die Uhrzeit des Zeitstempels.
Die Dienste, die die Zeitstempel anbieten, gewährleisten auch die aktuelle Uhrzeit.
- Den Hash-Wert (Prüfsumme des „gestempelten“ Dokumenteninhalts).
- Den Algorithmus und die Parameter des verwendeten Hash-Verfahrens.
- Das Zertifikat des Zeitstempelanbieters (optional).

Qualifizierter Elektronischer Zeitstempel

Eine mögliche Form der Qualifizierten Elektronischen Signatur stellt der Qualifizierte Elektronische Zeitstempel dar. Er ist eine Qualifizierte Elektronische Signatur des Zeitstempeldienstes von einem Zeitstempelanbieter (Timestamp Provider, TSP) und bestätigt, dass ein bestimmtes Dokument zu einem bestimmten Zeitpunkt vorgelegen hat. Qualifizierte Zeitstempel können durch zertifizierte Unternehmen, z.B. Trust Center, oder durch entsprechende zertifizierte Geräte erstellt werden.

Hash-Baum

SecDocs fordert zur Versiegelung nicht für jedes einzelne Datenobjekt einen eigenen Qualifizierten Zeitstempel an. Da Zeitstempelanbieter Kosten geltend machen, fordert SecDocs gemäß dem ArchiSig-Konzept erst für eine bestimmte Menge von Datenobjekten einen gemeinsamen Zeitstempel an. Dazu werden die Hash-Werte der einzelnen Datenobjekte gebildet und in einem Hash-Baum gemäß ArchiSig-Konzept eingetragen.

Ein Hash-Baum ist eine Datenstruktur, mit der eine größere Anzahl von Hash-Werten zu einem einzigen gemeinsamen Hash-Wert (Wurzel-Hash-Wert) zusammengefasst wird. Hash-Bäume bilden die Basis des ArchiSig-Konzeptes und der daraus abgeleiteten Evidence Records.

Evidence Record

Um den Nachweis für die Integrität eines Dokuments zu erbringen, das in einem Langzeitarchiv gespeichert ist, muss das Archivsystem einen Evidence Record für das Dokument erstellen (gemäß IETF RFC 4998). Dieser Evidence Record muss eine nahtlose Kette von gültigen Zeitstempeln enthalten (falls diese erneuert wurden), rückwirkend bis zum Zeitpunkt der Übernahme des Dokuments in das Archivsystem.

Bei der Übernahme von signierten Dokumenten in das Archivsystem werden Prüfprotokolle für die Signaturen eingeholt und diese ebenfalls dem Integritätsnachweis unterzogen. Dadurch wird implizit der Nachweis für die Authentizität eines Dokuments erbracht.

Signaturerneuerung gemäß ArchiSig-Konzept

Die Stärke der Algorithmen und Verfahren, auf denen die elektronische Signatur beruht, kann durch technischen Fortschritt und neue wissenschaftliche Erkenntnisse bei der Kryptoanalyse abnehmen. Man muss damit rechnen, dass bestimmte Algorithmen und/oder Parameter mittelfristig nicht mehr für die elektronische Signatur geeignet sind und durch andere Algorithmen bzw. Parameter ersetzt werden müssen. Die Bundesnetzagentur veröffentlicht in Zusammenarbeit mit dem BSI jährlich eine Übersicht über geeignete Algorithmen und Parameter sowie ein Datum, ab wann die Algorithmen oder Parameter nicht mehr verwendet werden sollen.

Gemäß der eIDAS-Verordnung müssen die signierten Daten mit einer neuen Qualifizierten Elektronischen Signatur versehen werden, bevor die eingesetzten Algorithmen und Parameter ablaufen (dieser Vorgang wird als Übersignatur oder Signaturerneuerung bezeichnet). Die neue Signatur muss auf geeigneten Algorithmen und Parametern beruhen und die Daten, die alten Signaturen und deren Prüfprotokolle, sowie die alten Zeitstempel einschließen.

Beim Erneuern des Zeitstempels muss nicht jedes einzelne Dokument mit einem eigenen neuen Zeitstempel versehen werden. Dem ArchiSig-Konzept entsprechend dürfen mehrere Dokumente zusammengefasst und mit einem gemeinsamen Zeitstempel versehen werden.

Aus dem gemeinsam erstellten Zeitstempel kann nach dem ArchiSig-Konzept mit dem Hash-Baum für jedes einzelne repräsentierte Objekt ein sogenannter reduzierter Hash-Baum erstellt werden, der sich im Evidence Record widerspiegelt (siehe [Abschnitt „Gruppieren und Versiegeln der Datenobjekte gemäß ArchiSig-Konzept“](#)).

2.2 Architektur

Die Architektur von SecDocs orientiert sich an den Kernkomponenten, die im Referenzmodell der TR-03125 (V1.2.1) des BSI und dem ArchiSafe-Schutzprofil (ACM_PP) beschrieben sind (siehe auch [Abschnitt „TR-03125 und ArchiSig-Konzept“](#)). Die in der TR-03125 geforderten Basisfunktionen wurden in SecDocs funktional erweitert, um einen breiten Einsatzbereich auch außerhalb der öffentlichen Verwaltung zu ermöglichen.

2.2.1 Komponenten von SecDocs

SecDocs besteht aus folgenden Komponenten:

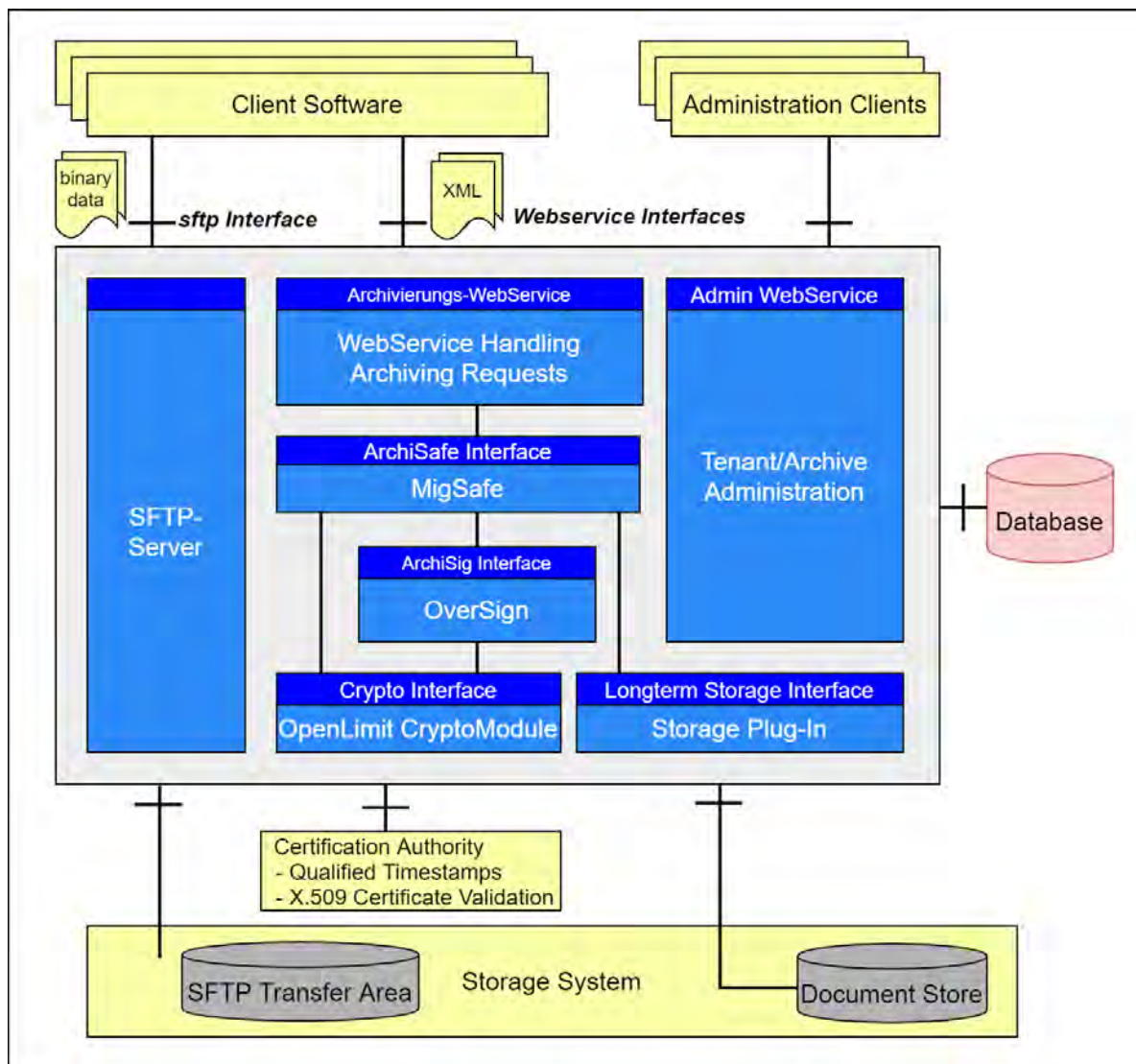


Bild 1: Komponenten von SecDocs (blau dargestellt) und Einbettung in die kundenspezifische Umgebung

WebService Handling, Archiving Requests

Web-Service als Schnittstelle für die Anbindung der Fachanwendungen (auch Client-Anwendungen genannt) wie DMS-, ERP- oder BPM-Systeme sowie fach- oder kundenspezifische Lösungen. Dieser Web-Service bietet Basisfunktionen, die laut TR-03125 gefordert sind, sowie erweiterte Archivfunktionen.



Die ausführliche Beschreibung dieser Schnittstelle finden Sie im [Kapitel „Web-Service für die Client-Anwendung“](#).

Admin Web Services

Web-Services als Schnittstelle für die Anbindung von Administrations-Clients.

Die Aufgaben der Administration sind das Verwalten des Gesamtarchivs (Web-Service `ArchiveAdminService`) und das Verwalten der einzelnen mandantenspezifischen Archivbereiche (Web-Service `MandantAdminService`).



Die ausführliche Beschreibung dieser Schnittstelle finden Sie im [Kapitel „Archiv- und Mandantenadministration“](#).

MigSafe, OverSign, CryptoModule

Krypto-Komponenten zum Nachweis der Integrität und Authentizität des archivierten Schriftguts.

- Überprüfen elektronischer Signaturen, die ggf. im Schriftgut enthalten sind und Erstellen der zugehörigen Prüfprotokolle,
- Bilden und Prüfen der Hash-Werte,
- Zusammenfassen der Hash-Werte von mehreren Archivobjekten zu einem gemeinsamen Hash-Wert (ArchiSig),
- Einholen von Zeitstempeln von ausgewählten Zeitstempelanbietern,
- Erzeugen von Evidence Records als Integritätsnachweis für die einzelnen Archivobjekte,
- Prüfen von Evidence Records und Erstellen eines Prüfberichts,
- Erneuern von Hash-Werten und Zeitstempeln sowie Erzeugen der zugehörigen erweiterten Evidence Records,
- Anbindung an entsprechende Zertifizierungsdiensteanbieter, die qualifizierte Zertifikate und Zeitstempel ausstellen.

Storage Plug-In

Anbindung unterschiedlicher Storage-Systeme.

Die aktuelle Version von SecDocs wird mit einem Plug-In für ein NAS-Filesystem (Network Attached Storage) ausgeliefert.

SFTP-Server

Schnittstelle, um zu archivierende Datenobjekte durch das Secure-File-Transfer-Protokoll unabhängig vom SOAP-Request auf den SecDocs-Server zu übertragen. Diese Schnittstelle steht nur zur Verfügung, wenn Sie ihre Nutzung in der Datei `secdocs.properties` explizit vereinbaren, siehe [Abschnitt „Integrierter SFTP-Server“](#). Sie benötigen sie nur, wenn Sie externe Datenobjekte archivieren wollen, siehe [Kapitel „Externe Datenobjekte“](#).

2.2.2 Software-Umgebung

SecDocs wird als Anwendung in einem Application Server betrieben und nutzt die Komponente CryptoModule

Im Lieferumfang von SecDocs sind die folgenden Bestandteile enthalten:

- SecDocs Enterprise Application
- WildFly-Server
- Java SDK
- CryptoModule

Software-Voraussetzungen

Der Betrieb von SecDocs V3.1 setzt die folgende Software-Umgebung voraus:

- Betriebssystem:
 - SuSE Linux Enterprise Server SLES 15 ab SP2 64bit (AMD64/x64)
- Datenbank (zur Ablage der Verwaltungsdaten):
 - Oracle Database 19c Enterprise Edition für Linux 64bit
- Storage-System
Für die Ablage der zu archivierenden Dokumente setzt SecDocs technisch gesehen eine NAS-Filesystem-Schnittstelle voraus.

Qualitätsgesichert ist die Anbindung von:

- NetApp® -Filer mit einem Betriebssystem ab ONTAP V7 (NetApp, Inc.). Dabei kann auch SoftWORM-Funktionalität genutzt werden. SecDocs hat erfolgreich den Zertifizierungsprozess zur Nutzung von NetApp® SnapLock durchlaufen.
- FUJITSU Storage ETERNUS CS8000 mit ViNS.



Die Software-Voraussetzungen für SecDocs sind jeweils abhängig vom aktuellen Release. Fujitsu liefert mit jedem Release von SecDocs eine ausführliche Installationsbeschreibung ("[SecDocs Installations- und Einsatz-Anleitung für SecDocs](#)" ([SD3] in Abschnitt "[Literatur](#)"). Dieses Dokument beschreibt die Software-Voraussetzungen für die Installation von SecDocs, den Lieferumfang, das Bereitstellen der Ablaufumgebung und die Konfiguration von SecDocs.

2.2.3 Storage-Systeme

Die physische Speicherung der archivierten Dokumente ist keine Komponente von SecDocs. Für diese Funktion werden am Markt etablierte Storage-Systeme eingesetzt.

SecDocs enthält Storage Plug-Ins, die für die jeweiligen Storage-Systeme entwickelt und zur Verfügung gestellt werden.

i Zu Testzwecken kann für die Ablage auch das lokale Filesystem verwendet werden.

2.2.4 Ablagestruktur im Archiv

Laut TR-03125 werden die Archivobjekte mit einer eindeutigen ID adressiert (Archived Object Identifier, AOID). Der Aufbau der ID ist unstrukturiert; anhand der ID ist also nicht erkennbar, wo im Archiv sich welcher Inhalt befindet. Um eine logische Zusammengehörigkeit von Objekten im Archiv erkennen zu können, bietet SecDocs die Möglichkeit, diese entsprechend fachlich und organisatorisch sinnvoller Strukturen zu archivieren.

SecDocs verwendet als Ablagestruktur für elektronisches Schriftgut eine baumartige Verzeichnisstruktur. Das Wurzelverzeichnis dieser Struktur im Dateisystem (die Archivwurzel) ist ein gemountetes Verzeichnis auf dem Storage-System (siehe [Abschnitt „Storage-System anbinden“](#)).

Unterhalb dieses Wurzelverzeichnisses werden die Pfade für die Mandanten erstellt.

Die Gesamtstruktur im Archiv wird logisch in drei Ebenen unterteilt:

- Mandant:

Der Pfad für den Teilbereich eines Mandanten im Archiv wird beim Einrichten des Mandanten mit der Operation `createMandant` festgelegt.



Die Operation `createMandant` stellt der Web-Service `ArchiveAdminService` zur Verfügung, mit dem das Archiv in seiner Gesamtheit administriert wird. Die Beschreibung finden Sie im [Abschnitt „Operation createMandant“](#).

Sollen die Daten der Mandanten auch auf dem Storage-System getrennt gehalten werden, muss dort für jeden Mandanten ein eigenes Volume oder Volume-Set eingerichtet werden. Für jeden Mandanten wird so ein Knoten definiert, unterhalb dessen alle seine Archivobjekte abgelegt werden. Der Administrator muss vorab die entsprechenden Volumes mounten und die Verzeichnisse einrichten. Die Beschreibung finden Sie im [Abschnitt „Storage-System anbinden“](#).

- Organisation:

Den einzelnen Organisationseinheiten werden entsprechende Pfade unterhalb der Ablagestruktur des Mandanten zugewiesen. Der Pfad für den Teilbereich der Organisationseinheit wird beim Einrichten mit der Operation `createOrganisation` festgelegt.



Die Operation `createOrganisation` stellt der Web-Service `MandantAdminService` zur Verfügung, mit dem der Teilbereich eines Mandanten im Archiv administriert wird. Die Beschreibung finden Sie im [Abschnitt „Operation updateMandant“](#).

Die Organisationsstruktur ist eine Unterstruktur eines Mandanten, liegt also unterhalb des Mandantenknotens. Für jede Organisation des Mandanten wird so ein Knoten definiert, unterhalb dessen alle seine Archivobjekte abgelegt werden.

- Schriftgut:

Unterhalb des Knotens für eine Organisationseinheit können weitere Knoten gebildet werden, unter denen fachlich und/oder organisatorisch zusammengehöriges Archivgut abgelegt werden kann (z.B. Akten, Vorgänge). Die Bezeichnungen für die Knoten in der Ablagestruktur der Dokumente kann fest vorgegeben werden, oder sie werden aus den Metadaten der Dokumente entnommen. Welche Metadaten ggf. für die Definition der Ablagestruktur verwendet werden sollen, wird beim Zusammenstellen der zu archivierenden Datenobjekte festgelegt.



Die Informationen zur Strukturierung des Schriftguts finden Sie im [Kapitel „Beschreibung der Datenobjekte“](#).

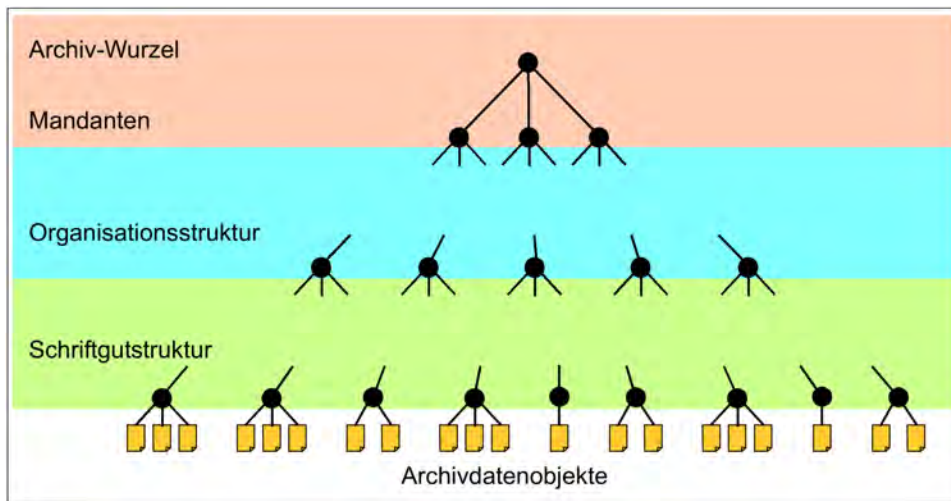


Bild 2: Ablagestruktur im Archiv

2.2.4.1 Ablagestruktur für externe Datenobjekte

Externe Datenobjekte (Näheres hierzu siehe [Kapitel „Externe Datenobjekte“](#)) werden in einem eigenen Bereich des SecDocs-Archivs abgelegt. Die Ablagestruktur in diesem Archivbereich ist die gleiche baumartige Verzeichnisstruktur wie im SDO-Archiv. Das Wurzelverzeichnis dieser Struktur (Archiv-Wurzel für externe Datenobjekte) ist wie die Archiv-Wurzel des SDO-Archivs ein gemountetes Verzeichnis auf dem Storage-System.

Der Anwender legt die Archiv-Wurzel für externe Datenobjekte mit dem Konfigurationsparameter `archiveRootExternalFiles` fest.

Die Trennung des Archivbereichs für externe Datenobjekte von dem des SecDocs-Archivs erlaubt eine unterschiedliche Konfiguration dieser Bereiche auf dem Storage-System.

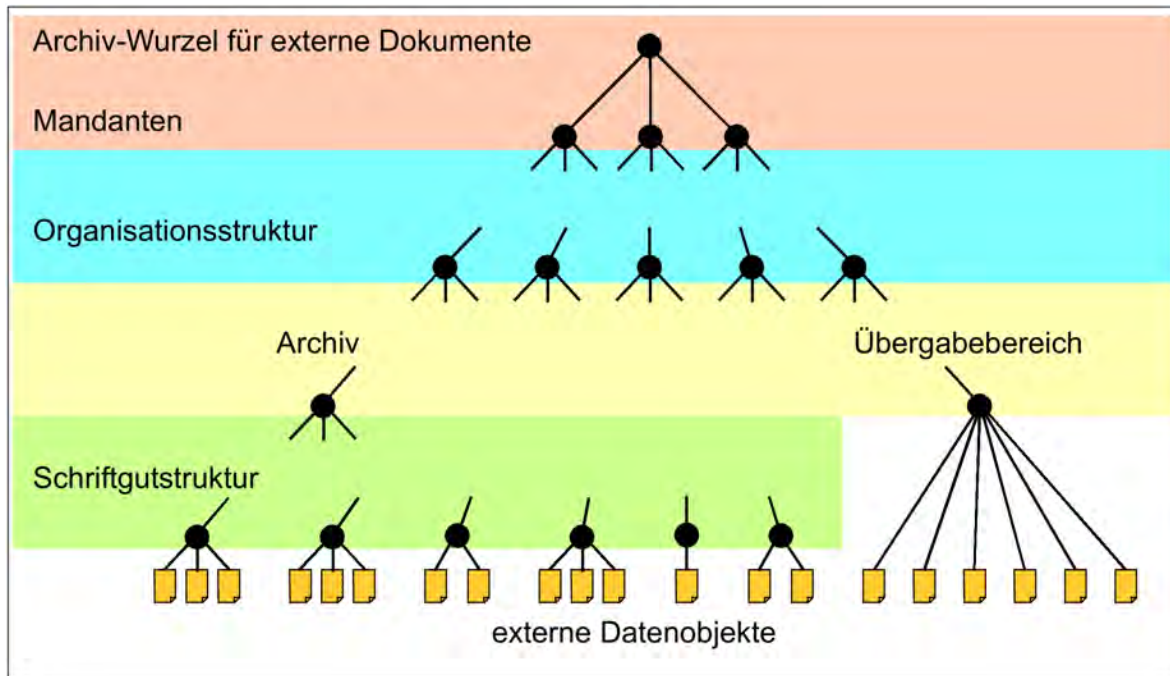


Bild 3: Ablagestruktur des Archivbereichs für externe Datenobjekte

Im Archivbereich für externe Datenobjekte existiert eine zusätzliche Ebene zwischen den Ebenen Organisation und Schriftgut. Diese enthält zwei Knoten: einen für das Archiv und einen für den Übergabebereich.

- **Archiv**
Das Archiv ist der physikalische Ablageort für die archivierten Datenobjekte
- **Übergabebereich**
Im Übergabebereich legt der SFTP-Server (siehe [Abschnitt „Integrierter SFTP-Server“](#)) die von der Client-Anwendung übertragenen externen Dokumente zur weiteren Verarbeitung ab und stellt dort die im Zuge eines `retrieveSDO` angeforderten externen Dokumente für die Rückübertragung auf den Client bereit. Das Verschieben der Datenobjekte zwischen Übergabebereich und Archiv wird SecDocs-intern durchgeführt.
Der SFTP-Server sorgt dafür, dass eine Client-Anwendung nur Sicht und Zugriff auf ihr organisationsspezifisches Unterverzeichnis im Übergabebereich erhält. Dieses erscheint für die Client-Anwendung als virtuelles root-Verzeichnis und lässt nur bestimmte Schreib- und Lese-Operationen innerhalb dieses Systems zu.

2.3 Multi-Node-Betrieb

SecDocs bietet alle seine Archivierungs- und Administrationsdienste über Webservice-Schnittstellen an.

SecDocs kann auf zwei Arten betrieben werden:

- Im Single-Node-Betrieb erfolgt die Bedienung der Webservice-Schnittstellen für ein Archiv über eine einzige SecDocs-Instanz.
- Um erhöhten Anforderungen bzgl. Hochverfügbarkeit und Lastverteilung gerecht zu werden, besteht die Möglichkeit, mehrere SecDocs-Instanzen im Verbund zu nutzen, d.h. einen Parallelbetrieb mehrerer SecDocs-Instanzen für ein Archiv zu erlauben ("Multi-Node-Betrieb").

Eine SecDocs-Instanz kann zum Verbund dynamisch hinzugefügt oder von diesem weggenommen werden.

2.3.1 Architektur

Folgende Grafik zeigt den Aufbau einer Multi-Node-Konfiguration von SecDocs. Details zu den blau dargestellten Komponenten von SecDocs finden Sie im [Abschnitt „Architektur“](#).

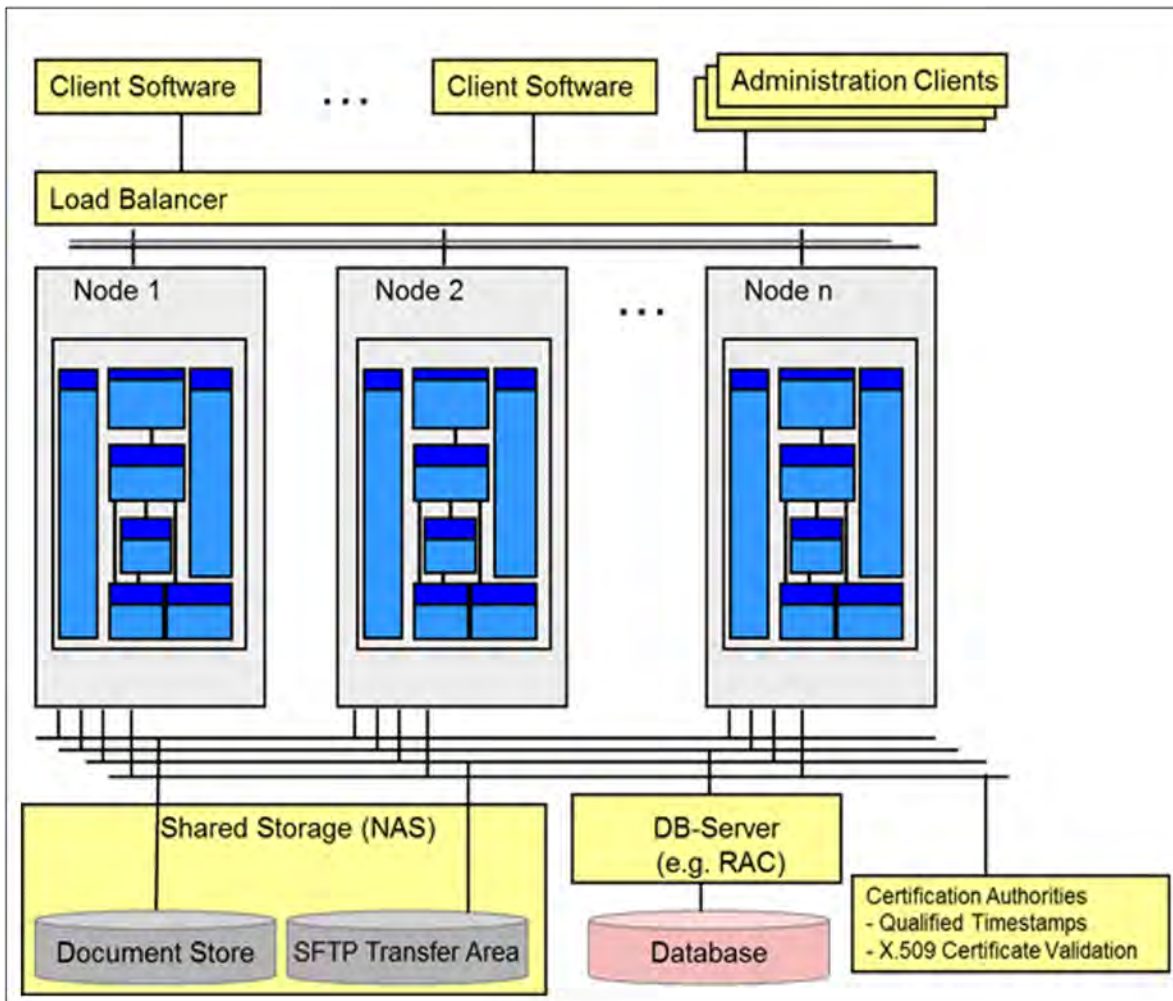


Bild 4: Architektur einer Multi-Node-Konfiguration von SecDocs

2.3.2 Konfiguration des Verbunds

i Die Performance in einem SecDocs-Verbund wird maßgeblich durch die Anzahl der Knoten beeinflusst. Bei der Abschätzung der Größe eines Verbunds (Sizing) sollte daher die Festlegung der Knotenanzahl mit Ihrem technischen Betreuer von Fujitsu abgestimmt werden.

Die allgemeinen Konfigurationsparameter für SecDocs sind in der Datei `secdocs.properties` abgelegt.

SecDocs erwartet die Datei unter folgendem Installationspfad:

`installation-dir/jboss/secdocs/configuration/secdocs/secdocs.properties`

Zur vereinfachten Administration von Multi-Node-Konfigurationen gibt es die Möglichkeit, für alle Verbundteilnehmer eine gemeinsame Konfigurationsdatei zu verwenden. Dazu wird aus der **primären** Konfigurationsdatei `secdocs.properties` im Installationsverzeichnis auf eine **sekundäre** Konfigurationsdatei verwiesen. Diese befindet sich typischerweise auf einem von den Verbundteilnehmern gemeinsam genutzten Speicher und enthält die für alle Verbundteilnehmer identischen Konfigurationsparameter.

Siehe hierzu auch `globalPropertiesFile` auf "[Konfigurationsdatei secdocs.properties](#)".

2.3.2.1 Die Konfigurationsdatei `secdocs.properties`

Die Beschreibung der Konfigurationsdatei `secdocs.properties` ist im [Abschnitt „Konfigurationsdatei `secdocs.properties`“](#) zu finden.

Für den Multi-Node-Betrieb von SecDocs müssen die folgenden Konfigurationsparameter in der Datei `secdocs.properties` versorgt sein:

```
multiNode = true
```

(siehe `multiNode` auf "[Konfigurationsdatei `secdocs.properties`](#)")

```
multiNode.hazelcastConfigFile = Path to Hazelcast configuration file
```

(siehe `multiNode.hazelcastConfigFile` auf "[Konfigurationsdatei `secdocs.properties`](#)")

Zur Handhabung der Konfigurationsdatei `secdocs.properties` wird für den Multi-Node-Betrieb die folgende Vorgehensweise empfohlen:

- Richten Sie eine verbund-globale (sekundäre) Konfigurationsdatei `secdocs.properties` ein, die für alle Server im Verbund zugreifbar ist, z.B. auf dem Shared Storage.

In dieser Datei werden die Konfigurationsparameter festgelegt, die global für alle Knoten des Verbunds gelten.

- Hinterlegen Sie in den einzelnen server-lokalen (primären) Konfigurationsdateien mit dem Konfigurationsparameter `globalPropertiesFile` einen Verweis auf die verbundglobale (sekundäre) Konfigurationsdatei:

```
globalPropertiesFile = <Path to a secondary property file>
```

(siehe `globalPropertiesFile` auf "[Konfigurationsdatei `secdocs.properties`](#)")

Diese primären (lokalen) Konfigurationsdateien liegen im lokalen Installationsverzeichnis einer jeden SecDocs-Instanz.

i Beachten Sie: Eine primäre Konfigurationsdatei darf nicht durch einen Link auf die sekundäre Konfigurationsdatei realisiert werden.

- Damit im Verbund alle Konfigurationsparameter die gleichen Einstellungen haben, empfiehlt sich folgende Vorgehensweise:
 - Legen Sie alle Konfigurationsparameter verbund-global in der sekundären Konfigurationsdatei fest.
 - Fügen Sie in die primären Konfigurationsdateien lediglich einen Verweis auf die sekundäre Konfigurationsdatei ein.
 - Wenn auf einer SecDocs-Instanz spezielle lokale Einstellungen erforderlich sind, hinterlegen Sie diese in der primären Konfigurationsdatei (ein Eintrag in der primären Konfigurationsdatei überschreibt einen Eintrag in der sekundären Datei).

2.3.2.2 Konfigurationsparameter des CryptoModule

Wenn Sie Änderungen an den Konfigurationsparametern des CryptoModule vornehmen, stellen Sie sicher, dass die Änderungen auf allen Knoten im Verbund konsistent durchgeführt werden.

Die Konfigurationsparameter des CryptoModule sind im "[CryptoModule Runtime Umgebung für Fujitsu SecDocs V3.1](#)" ([SD5] in Abschnitt "[Literatur](#)") beschrieben.

2.4 Archivierung

SecDocs führt beim Archivieren von Dokumenten folgende Aktionen durch:

- Schema-Validierung für das übergebene Datenobjekt (siehe [Abschnitt „Beschreibung eines Datenobjekts“](#)).
- Falls im Datenobjekt Signaturen enthalten sind:
Verifizieren der Signaturen und Erstellen der zugehörigen Prüfprotokolle (siehe [Abschnitt „Verifizierung der Signaturen“](#) unten).
- Setzen der Aufbewahrungsfrist, innerhalb derer das Datenobjekt nicht gelöscht werden darf.
- Zusammenstellen des zu speichernden Objekts (Datenobjekt und Prüfprotokolle für die spätere Versiegelung mit einem Zeitstempel) und Übergabe an das Storage-System mit Hilfe des Storage Plug-Ins (siehe [Abschnitt „Erzeugen des Evidence Records“](#)).
- Speichern von Datenobjekt und Prüfprotokollen gemäß der vorgegebenen Ablagestruktur (siehe [Abschnitt „Ablagestruktur im Archiv“](#)).
-

2.4.1 Prüfung elektronischer Signaturen

Beim Archivieren werden eventuell im Dokument mit übergebene Signaturen geprüft. Diese Prüfung übernimmt das OpenLimit CryptoModule. Unterstützt werden die Signaturformate PAdES Baseline-B/-T (ETSI EN 319 142 part 1-2 v1.1.1) sowie CAdES Baseline-B/-T (ETSI EN 319 122 part 1-2 v1.1.1).

Das CryptoModule führt eine AdES-Signaturprüfung nach [ETSI EN 319 102-1 v1.1.1](#) durch. Zur Sperrprüfung werden OCSP (Online Certificate Status Protocol - RFC6960) und Sperrlisten (CRL - Certificate Revocation Lists) verwendet. Die Ermittlung der Signaturqualität orientiert sich an ETSI TS 119 172-4.

2.4.2 Gruppieren und Versiegeln der Datenobjekte gemäß ArchiSig-Konzept

Die Menge der Datenobjekte in einem Hash-Baum ist konfigurierbar (siehe Abschnitt „Hash-Baum“ in [Kryptografische Verfahren](#)). Der Archivadministrator definiert beim Anlegen eines Mandanten

- die Zahl der Datenobjekte, die maximal mit einem Zeitstempel versiegelt werden dürfen,
- die Zeit zwischen zwei Versiegelungen.

Ist die vorgegebene Zahl der Datenobjekte erreicht oder die Wartezeit zwischen zwei Versiegelungen verstrichen,

- bildet SecDocs die Hash-Werte der gespeicherten Datenobjekte,
- bildet SecDocs aus diesen Hash-Werten einen Hash-Baum gemäß ArchiSig-Konzept,
- übergibt SecDocs den so entstandenen Wurzel-Hash-Wert an den Zeitstempelanbieter zur Versiegelung mit einem Zeitstempel,
- versiegelt der Zeitstempelanbieter den Wurzel-Hash-Wert und damit implizit den Hash-Baum: der Hash-Wert wird um die aktuelle Uhrzeit ergänzt und mit dem privaten Schlüssel des Zeitstempelanbieters signiert,
- gibt der Zeitstempelanbieter den versiegelten Hash-Wert zusammen mit seinem Zertifikat an SecDocs zurück,
- erzeugt und schreibt SecDocs einen Evidence Record für jedes Datenobjekt (siehe [Abschnitt „Erzeugen des Evidence Records“](#)).

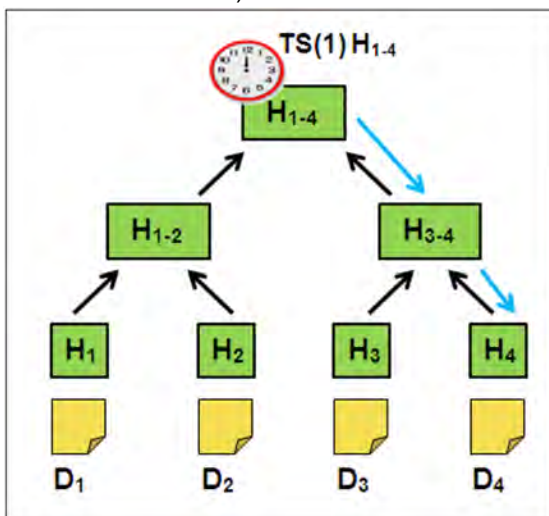


Bild 5: Erzeugen eines Hash-Baums mit Zeitstempel

TS (1)	Zeitstempel (1) des Zeitstempelanbieters 1
$H_{1-2}, H_{3-4}, H_{1-4}$	Zusammengefasste Hash-Werte
H_{1-4}	Wurzel-Hash-Wert
H_1, H_2, H_3, H_4	Hash-Werte der einzelnen Dokumente
D_1, D_2, D_3, D_4	Dokumente

2.4.3 Erzeugen des Evidence Records

SecDocs erzeugt für jedes Datenobjekt im Hash-Baum einen standardisierten Evidence Record mit reduziertem Hash-Baum. Der Evidence Record basiert auf den genutzten kryptografischen Verfahren, dem Hash-Algorithmus und dem Signatur-Algorithmus.

Der Evidence Record kann (zusammen mit dem gespeicherten Datenobjekt) ausgelesen und für einen externen Nachweis der Integrität des Datenobjekts genutzt werden. Die Integritätsprüfung kann ebenfalls durch SecDocs vorgenommen werden.

Der Evidence Record enthält

- den Hash-Algorithmus, der für den Hash-Baum verwendet wurde,
- den Hash-Wert des Dokuments,
- die Hash-Werte aus dem reduzierten Hash-Baum:
Hash-Werte der Seitenäste entlang des Weges von der Wurzel zum Dokument,
- den Wurzel-Hash-Wert des Hash-Baums, der an den Zeitstempelanbieter übergeben wurde.
- den Zeitstempel des Zeitstempelanbieters,
- das Zertifikat des Zeitstempelanbieters.

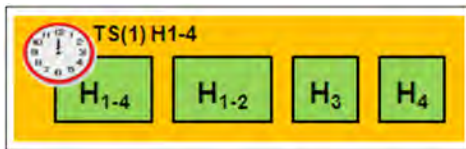


Bild 6: Evidence Record für Dokument D_4 (siehe Bild 5 in "[Gruppieren und Versiegeln der Datenobjekte gemäß ArchiSig-Konzept](#)")

TS (1) Zeitstempel (1) des Zeitstempelanbieters 1

H_{1-2} , H_{1-4} Zusammengefasste Hash-Werte

H_3 , H_4 Hash-Werte der einzelnen Dokumente

Abschließend speichert SecDocs den Evidence Record gemäß der vorgegebenen Ablagestruktur im gleichen Verzeichnis wie das Datenobjekt (siehe [Abschnitt „Ablagestruktur im Archiv“](#)).

Zeitstempel-Signaturerneuerung

Wenn die Algorithmen, die bei der Archivierung von Dokumenten verwendet wurden, oder ihre Parameter von der Bundesnetzagentur für ungeeignet erklärt werden, müssen Sie mit SecDocs die Zeitstempel erneuern (siehe [Abschnitt „Signaturerneuerung gemäß ArchiSig-Konzept“](#) (in "[Kryptografische Verfahren](#)")).

Die Signaturerneuerung dient der Beweiswerterhaltung und muss mit Hilfe der Mandantenadministration angestoßen werden (siehe [Abschnitt „Web-Services für die Administration“](#)).

SecDocs führt diese Erneuerung von Hash-Werten, Signaturen und Zeitstempeln konform zum SigG und zur SigV durch. Die Signaturerneuerung erfolgt nach der Aktivierung automatisiert und auch bei großen Datenmengen schnell und kostenoptimiert. Anwender und Administratoren von SecDocs benötigen dafür kein weiteres Signaturwissen.

2.4.4 Beschreibung eines Datenobjekts

Abhängig von den Bedürfnissen des Anwenders werden in einem Langzeitarchiv Dokumente unterschiedlicher Typen archiviert, z.B. Rechnungen, Urkunden, Verträge, usw.

Bevor Sie Daten archivieren, sind unter anderem folgende Punkte zu klären:

- Welche Dokumente wollen Sie wann und wozu im Geschäftsprozess archivieren?
- Welche Arten von Dokumenten wollen Sie archivieren?
- Welche Metadaten müssen zusammen mit den Dokumenten archiviert werden?
- Was sind die Formate der zu archivierenden Dokumente?

i Die archivierten Dokumente sollen auch noch nach längerer Aufbewahrungszeit in der ursprünglichen Form darstellbar sein. Daher wird derzeit empfohlen, die Dokumente als PDF/A, im TIFF6- oder im ASCII-Format zu archivieren. Diese Formate sind international standardisiert und gelten daher als langfristig reproduzierbar. Weitere empfohlene Formate werden in Zukunft folgen.

- Sind die Dokumente elektronisch signiert und müssen diese Signaturen mit archiviert bzw. beim Archivieren geprüft werden?
- Können mehrere Schriftstücke zusammengehörig als ein Objekt archiviert werden, z.B. ein signierter Vertrag mit seinen Anhängen?
- Wie lange sind die archivierten Dokumente aufzubewahren?
- Werden mehrere Versionen der archivierten Dokumente benötigt?
- Sollen die archivierten Dokumente überschreibbar sein?

Der Dokumententyp, also Art, Struktur und beschreibende Metadaten des zu archivierenden Schriftguts, ist jeweils unterschiedlich. In strukturierten Geschäftsprozessen fallen allerdings immer wieder strukturell gleichartige Datenobjekte zur Archivierung an, z.B. Datenobjekte vom Typ „Rechnungen“ oder „Angebote“.

Beispiel

Der Typ „Angebote“ soll folgende Eigenschaften besitzen:

- Ein Angebot besteht immer aus einem Hauptdokument und mehreren Anhängen.
- Das Hauptdokument ist ggf. mehrfach signiert.
- Die minimale Aufbewahrungszeit ist 10 Jahre.

Solche strukturell gleichartigen Datenobjekte können Sie zu einem Dokumententyp zusammenfassen. Jeden Dokumententyp müssen Sie vor der Archivierung bei SecDocs bekanntmachen, damit SecDocs eine Validierung durchführen kann (siehe Abschnitt „[Hinweise für den Zugriff auf Dokumentversionen](#)“ (in „[Versionieren von Dokumenten](#)“)).



Wie Sie einen Dokumententyp beschreiben und bei SecDocs bekanntmachen, ist ausführlich im [Abschnitt „SDO-Struktur“](#) beschrieben.

2.4.4.1 Submission Data Object

Die Datenobjekte eines bestimmten Dokumententyps werden SecDocs dann in Form eines XML-Containers übergeben. Dieser Vorgang wird mit „Submit“ bezeichnet, der übergebene XML-Container als „Submission Data Object“ (SDO), der Dokumententyp entsprechend als „SDO-Typ“.

Jedes SDO enthält die Dokumente selbst (die Primärdokumente) und ggf. zugehörige Signaturen sowie Metadaten.



Ein Beispiel für den Aufbau eines solchen XML-Containers (eines SDO), finden Sie im [Abschnitt „Datenobjekt für die Archivierung“](#).

2.4.4.2 Client Object Identifier (COID)

Die COID kann anstelle der AOID zur Identifikation eines Archivobjekts genutzt werden. Im Gegensatz zur AOID kann die COID jedoch vom Client frei vergeben werden. Sie muss jedoch innerhalb eines Mandanten eindeutig sein (Einschränkung siehe im folgenden [Abschnitt „Versionieren von Dokumenten“](#)).

Die COID bietet insbesondere die Möglichkeit, bei der Migration eines bestehenden Archivs in ein SecDocs-Archiv die bisherige ID zusammen mit dem SDO in das SecDocs-Archiv zu übertragen. Dadurch können Sie spätere Operationen (z.B. Recherche- und Retrieve-Operationen) weiter mit der bisher bekannten ID durchführen.

Die COID kann zusätzlich zur Versionierung von Dokumenten in SecDocs genutzt werden.

Die COID wird mit der Operation `getAOID` ("[Operation getAOID](#)") oder `submitSDO` ("[Operation submitSDO](#)") an SecDocs übergeben. Wurde eine gültige COID angegeben, ordnet die jeweilige Operation dem SDO eine AOID zu und liefert diese in der Antwortnachricht zurück. Für den späteren Zugriff auf das archivierte Objekt kann wahlweise die AOID oder die COID verwendet werden. Es können auch beide IDs spezifiziert werden, sie müssen sich dann jedoch auf dasselbe Objekt beziehen.

2.4.4.3 Versionieren von Dokumenten

Dokumente können versioniert archiviert werden. Dies wird dadurch ermöglicht, dass mehrere Versionen eines Dokuments unter einer gemeinsamen COID abgelegt werden. Für jede Version eines versionierten Dokuments müssen Sie allerdings mit `getAOID` ("[Operation getAOID](#)") oder `getAOIDWithRef` ("[Operation getAOIDWithRef](#)") eine eigene AOID anfordern. Wenn bei der Anforderung einer AOID eine COID angegeben ist, vergibt SecDocs automatisch eine Versionsnummer. Bei der ersten Anforderung einer AOID für eine bestimmte COID ist diese Versionsnummer 0. Bei jeder weiteren Anforderung einer AOID für dieselbe COID wird sie um 1 erhöht. Die Versionsnummer wird ausschließlich durch die Reihenfolge der Anforderungen von AOIDs festgelegt.

Diese Versionsnummer wird bei der Operation `statusSDO` ("[Operation statusSDO](#)") zurückgeliefert und ist im Filter über den voreingestellten Schlüssel `$VersionNumber` zugänglich, siehe "[Verwendung von Schlüsseln](#)".



Beachten Sie:

- Es können nur Dokumente versioniert werden, denen eine COID zugeordnet ist. Für bereits archivierte Dokumente kann nachträglich keine COID vergeben werden, sie können daher auch nicht versioniert werden. Für archivierte Dokumente, die bereits eine COID besitzen, können weitere Versionen archiviert werden.
- Unterschiedliche Versionen eines Dokuments können mit unterschiedlichen SDO-Typen archiviert werden. Somit ist es insbesondere auch möglich, dass einzelne Versionen des Dokuments überschreibbar sind und andere nicht, siehe [Abschnitt „Überschreiben von SDOs“](#).
- Die Idempotenz des Archivierungsvorgangs bleibt auch für versionierte Dokumente erhalten: Wenn die Archivierung einer Version wiederholt wird, entsteht keine weitere Version im Archiv und der Anwender erhält eine entsprechende Nachricht.
- SDOs mit externen Referenz-Ids können grundsätzlich nicht versioniert werden.

Hinweise für den Zugriff auf Dokumentversionen

Operationen mit Angabe einer COID, aber ohne AOID-Angabe werden für das Dokument mit der höchsten Versionsnummer durchgeführt. Für den Zugriff auf eine bestimmte Version eines Dokuments ist immer die Angabe der AOID erforderlich. Diese kann ggf. mit der Operation `listSDOVersions` ("[Operation listSDOVersions](#)") ermittelt werden. Werden sowohl AOID als auch COID für eine Operation angegeben, prüft SecDocs, ob sich beide Angaben auf dasselbe Dokument beziehen, und gibt ggf. eine Fehlermeldung aus.

2.4.4.4 Überschreiben von SDOs

Sie können mit SecDocs Dokumente auch so archivieren, dass sie überschrieben bzw. ersetzt werden können.

Die Eigenschaft, ob ein Dokument überschreibbar ist, erhält es über seinen SDO-Typ. Beim Anlegen eines SDO-Typs kann die Eigenschaft festgelegt werden, dass SDOs dieses Typs überschreibbar sind. Standardmäßig wird für einen neuen SDO-Typ „nicht überschreibbar“ festgelegt. Diese Eigenschaft kann für bestehende SDO-Typen nachträglich nicht mehr geändert werden.

Die Eigenschaft, ob für einen SDO-Typ die Überschreibbarkeit vereinbart ist, können Sie mit der Operation `getSDOTypes` ermitteln, siehe "[Operation getSDOTypes](#)". Die Operationen `statusSDO` ("[Operation statusSDO](#)") bzw. `getSchemaDataSDO` ("[Operation getSchemaDataSDO](#)") liefern die Information, ob der Typ eines bestimmten SDOs erlaubt, es zu überschreiben.

Ein als überschreibbar archiviertes SDO kann erst dann überschrieben werden, wenn seine Aufbewahrungsfrist abgelaufen ist.



SDOs mit externen Referenz-Ids können grundsätzlich nicht überschrieben werden.

2.4.4.5 Validierung

Bei der Archivierung prüft SecDocs die Validität des zu archivierenden Datenobjekts. Dazu gehört z.B.:

- Prüfen der Vollständigkeit: Sind alle Vorgaben für diesen SDO-Typ eingehalten (Schema-Validierung)?
- Prüfen der Gültigkeit der Signaturen (Signatur-Validierung). Das Prüfergebnis wird zusammen mit dem Primärdokument archiviert.

2.4.4.6 Informationen zur Adressierung von Daten im SDO

Für die Validierung, ebenso wie für die Ablage und Verwaltung der archivierten Datenobjekte benötigt SecDocs Hinweise, wo die einzelnen Daten in der Struktur zu finden sind. Dazu müssen Sie zu jeder Datenobjektbeschreibung (XML-Schema) auch einen Filter definieren. Datenobjektbeschreibung und Filter müssen Sie dann zusammen bei SecDocs als SDO-Typ registrieren.

In den Filterdefinitionen können außerdem Parameter für den SDO-Typ hinterlegt werden, die dann für alle SDOs dieses Typs gelten, z.B.

- die Adressierung der Primärdokumente und ihrer zugehörigen Signaturen,
- die Aufbewahrungsfrist,
- das Format der Primärdokumente, die in diesem SDO-Typ enthalten sind (z.B. PDF/A, mit oder ohne eingebetteter Signatur),
- die Festlegung der Ablagestruktur im Archiv (z.B. Akte/Vorgang/Dokument oder Jahr/Monat/Tag),
- die Auswahl von Elementen aus den Metadaten, die zur Protokollierung der Archivoperationen übernommen werden sollen.



Die ausführliche Beschreibung der Filterdefinition finden Sie im [Abschnitt „Adressierung von Daten im SDO“](#).

2.5 Recherche

Um auf archivierte Objekte zugreifen zu können (z.B. mit der Operation `retrieveSDO`), ist die Kenntnis der AOID oder der COID des Objekts nötig. Die AOID wurde beim Archivierungsvorgang zurückgegeben.

In der Praxis wird es aber oft so sein, dass nur bestimmte Metadaten zu einem archivierten SDO bekannt sind (z.B. ein Aktenzeichen, der Verfasser oder auch nur das ungefähre Archivierungsdatum). Zum Ermitteln der AOIDs bzw. der COIDs von SDOs stehen folgende Möglichkeiten zur Verfügung:

- Mit der Recherche anhand von Metadaten können alle AOIDs bzw. COIDs ermittelt werden, die eine bestimmte Metadatenkombination erfüllen.
- Mit der Recherche anhand der Ablagestruktur können alle AOIDs bzw. COIDs ermittelt werden, die unter einem bestimmten Pfad im Archiv abgelegt wurden.

2.5.1 Recherche mittels Navigation

Mit Hilfe der Operation `navigate` können Sie durch die Ablagestrukturen des Archivs navigieren, um alle AOIDs Ihrer in SecDocs abgelegten Dokumente zu finden.

Der Ablageort der SDOs wird durch den Systemschlüssel `$SDOPath` gesteuert. Der Wert dieses Schlüssels wird zum Zeitpunkt der Archivierung (also bei `submitSDO` oder `replaceSDO`) ermittelt und beeinflusst den Ablageort des SDOs (unterhalb des durch den `MandantAdmin` definierten organisationsspezifischen Knotens).

Mit Hilfe der Operation `navigate` können, ausgehend von dieser organisationspezifischen Wurzel, alle weiteren Unterverzeichnisse und, falls vorhanden, die darin befindlichen AOIDs aufgelistet werden. Wurde der Systemschlüssel `$Subject` bei der Archivierung versorgt, wird der Wert dieses Schlüssels zu jeder AOID ebenfalls zurückgegeben. Ebenso wird zu jeder AOID die COID zurückgegeben, falls diese bei der Archivierung angegeben wurde.

2.6 Web-Services

SecDocs bietet verschiedene Web-Services als Schnittstellen zu den Client-Anwendungen und als Administrationsschnittstelle an. Die Web-Services stellen die Operationen zum Archivieren und Administrieren zur Verfügung.

2.6.1 Mandantenfähigkeit

SecDocs verwaltet das Schriftgut der verschiedenen Kunden, Auftraggeber oder Organisationseinheiten strikt voneinander getrennt. Ein Mandant kann nicht auf die Daten, Dokumente oder Parameter eines anderen Mandanten zugreifen.

Die Mandantenfähigkeit in SecDocs bedeutet unter anderem:

- SecDocs legt die Dokumente logisch getrennt in mandantenspezifischen Verzeichnissen ab. Zusätzlich besteht die Möglichkeit, diese Verzeichnisse auf dem Storage-System in mandantenspezifische Volumes zu legen.
- SecDocs verwaltet die zugehörigen Rollen und Zugriffsrechte.
- SecDocs bietet ein mandantenspezifisches Audit-Logging: Alle Archiv- und Verwaltungsaufträge eines Mandantenadministrators bzw. aller weiteren Rollen dieses Mandanten (siehe [Abschnitt „Berechtigungskonzept“](#)) werden getrennt vom Audit-Logging anderer Mandanten aufgezeichnet.

2.6.2 Berechtigungskonzept

Eine Client-Anwendung räumt einem Nutzer abhängig von seiner Rolle unterschiedliche Rechte ein. Als Nutzer von SecDocs muss sich eine solche Client-Anwendung ihrerseits bei SecDocs authentisieren, d.h. sie meldet sich bei SecDocs in einer bestimmten Rolle an und wird dann von SecDocs autorisiert.

Die Client-Anwendung kann dann nur die Funktionen nutzen, die der Rolle zugewiesen sind und die vom entsprechenden Web-Service zur Verfügung gestellt werden.

Die rollenbasierte Authentisierung erfolgt in SecDocs durch die Angabe des Mandanten, der Organisationseinheit und der Rolle.

SecDocs kennt im Auslieferungszustand nur die folgende Rolle für die Anmeldung am Web-Service:

ArchiveAdmin (Archivadministrator)

Die Anwendung nutzt die Funktionen des Web-Service `ArchiveAdminService` für die Administration des gesamten Archivs (siehe Abschnitt „[Web-Services für die Administration](#)“).

Weitere Rollen werden automatisch erzeugt, wenn Sie einen neuen Mandanten mit `createMandant` (siehe "[Operation createMandant](#)") oder `createMandantXAIP` (siehe Handbuch „[SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR](#)" ([SD2] in Abschnitt "[Literatur](#)")“) anlegen:

MandantAdmin (Mandantenadministrator)

Die Anwendung nutzt die Funktionen des Web-Service `MandantAdminService` für die Administration des Archivbereichs der einzelnen Mandanten (siehe Abschnitt „[Web-Services für die Administration](#)“).

SecDocs_MandantAuditor

Die Rolle ermöglicht den Zugriff auf mandantenspezifische Audit-Log-Dateien mit den Operationen `getAuditLogFileNames` (siehe "[Operation getAuditLogFileNames](#)") und `getAuditLogFile` (siehe "[Operation getAuditLogFile](#)") des Web-Service `MandantAdminService`.

Diese Rolle ist nach dem Einrichten zunächst gesperrt. Zum Freischalten des Zugangs ist eine Aktion des `MandantAdmin` nötig: Er muss mit der Operation `setCredentials` ein Passwort für die Rolle `SecDocs_MandantAuditor` vergeben. Auch das Ändern des Passworts ist nur dem `MandantAdmin` erlaubt.

Archivar

Die Client-Anwendung nutzt entweder die Funktionen des Web-Service `ArchivingService` für die Archivierung von SDOs (siehe Abschnitt „[Web-Service für die Archivierung](#)“) oder die Funktionen des Web-Service S4 für die Archivierung von XAIP-Datenobjekten (siehe Handbuch „[SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR](#)" ([SD2] in Abschnitt "[Literatur](#)")“).

Die Rolle Archivar kann vom `MandantAdmin` durch individuell definierte Rollen ersetzt werden.



Näheres zum Berechtigungskonzept in SecDocs entnehmen Sie dem Abschnitt „[Zugangsprüfung](#)“ bzw. dem Handbuch „[SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR](#)" ([SD2] in Abschnitt "[Literatur](#)")“.

2.6.2.1 Vorgegebene Rolle Archivar

Beim Einrichten eines neuen Mandanten mit der Operation `createMandant` durch den `ArchiveAdmin` wird neben dem mandantenspezifischen Administrationszugang der Zugang für die Archivierung eingerichtet, d.h. es wird die mandantenglobale Rolle `Archivar` eingerichtet. Mit dieser Rolle können alle Operationen des Web-Service `ArchivingService` für alle Organisationen des Mandanten ausgeführt werden. Die Zugangsberechtigung für diese Rolle wird zunächst beim Erstellen des Mandanten gesetzt und kann später mit der Operation `setCredentials` (durch den `MandantAdmin`) geändert werden.

Beim Einrichten eines neuen Mandanten mit der Operation `createMandantXAIP` durch den `ArchiveAdmin` werden zunächst nur die administrativen Rollen (`MandantAdmin` und `SecDocs_MandantAuditor`) eingerichtet; eine mandantenglobale Rolle `Archivar` gibt es für einen solchen Mandanten nicht. Beim Anlegen der Organisationen für diesen Mandanten wird dann für jede Organisation eine eigene Rolle `Archivar` eingerichtet. Für jede dieser organisationsspezifischen Archivar-Rollen muss dann mit `setCredentials` (durch den `MandantAdmin`) die Zugangsberechtigung gesetzt werden.

Für eine feiner granulare Zugangssteuerung kann der `MandantAdmin` organisationsspezifische Rollen definieren und die Menge der erlaubten Operationen einschränken. Werden organisationsspezifische Rollen definiert, so wird der Zugang über die mandantenglobale Rolle `Archivar` gesperrt.

2.6.2.2 Organisationspezifisches Rollenkonzept

Zu jeder Rolle gehören eine Zugangsberechtigung (Credential) und ein Privileg. Ein Privileg besteht aus einer Liste von Operationen, die diese Rolle ausführen kann. Die Privilegien sind durch den `MandantAdmin` definierbar.

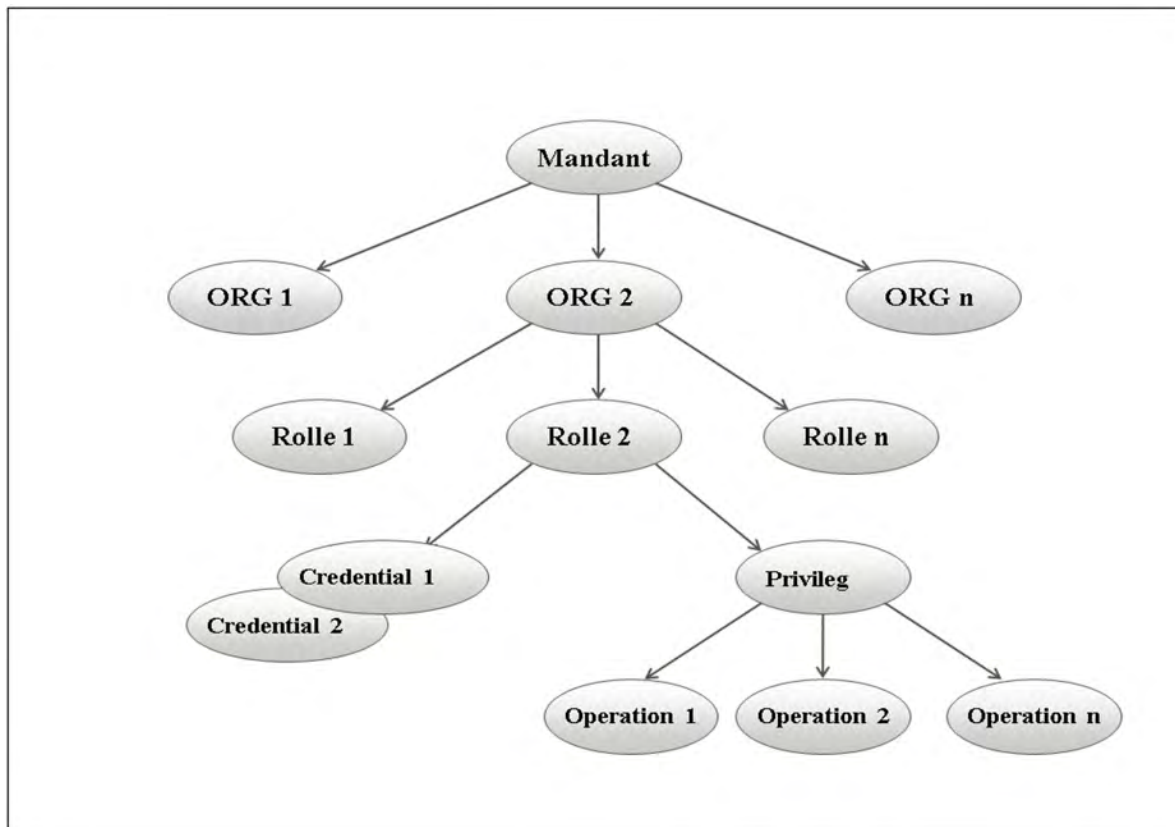


Bild 7: Organisationsspezifische Rollen

Für einen mit der Operation `createMandant` eingerichteten Mandanten gilt:

Sobald ein Administrator neue organisationsspezifische Rollen definiert, ist der mandantenglobale Zugang nicht mehr möglich, das heißt, die von SecDocs bereitgestellte Rolle `Archivar` wird deaktiviert. Durch das Löschen der letzten organisationsspezifischen Rolle wird der Zugang über die mandantenglobale Rolle wieder möglich. Zum Reaktivieren des Zugangs zur mandantenglobalen Rolle `Archivar` muss durch den `MandantAdmin` mit `setCredentials` eine neue Zugangsberechtigung gesetzt werden.

Operationen zum Verwalten organisationsspezifischer Rollen

Zum Verwalten der organisationsspezifischen Rollen gibt es folgende Operationen für den Web-Service `MandantAdminService`:

Operation	Beschreibung
-----------	--------------

createRole	<p>Mit <code>createRole</code> werden organisationsspezifische Rollen definiert und Zugangsberechtigungen für diese Rollen gesetzt.</p> <p>Die Zugangsberechtigung kann durch den <code>MandantAdmin</code> mit <code>setCredentials</code> geändert werden. Die alte Zugangsberechtigung bleibt wirksam, bis mit <code>setCredentials</code> die Zugangsberechtigung ein weiteres Mal gesetzt wird. So können Änderungen der Zugangsberechtigung auf dem Server und dem Client zeitlich entzerrt voneinander erfolgen. Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation createRole“ bzw. in Abschnitt „Operation setCredentials“.</p>
deleteRoles	<p>Mit <code>deleteRoles</code> werden organisationsspezifische Rollen gelöscht.</p> <p>Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation deleteRoles“.</p>
updateRole	<p>Mit <code>updateRole</code> werden die Operationen geändert, die eine organisationsspezifische Rolle ausführen darf.</p> <p>Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation updateRole“.</p>
getRoles	<p><code>getRoles</code> liefert eine Liste der für diesen Mandanten definierten Rollen (mit Eigenschaften), oder eine Liste der Rollen bestimmter Organisationen.</p> <p>Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation getRoles“.</p>

Operationen zum Verwalten von Privilegien

Um das Formulieren gleichartiger Rollen zu vereinfachen, werden vom `MandantAdmin` Mengen von Operationen in Privilegien zusammengefasst. Dazu werden im Web-Service `MandantAdminService` folgende Funktionen angeboten:

Operation	Beschreibung
createPrivilege	<p><code>createPrivilege</code> fasst eine Menge von Operationen unter einem Namen zusammen. Der Name des Privilegs wird dann bei <code>createRole</code> angegeben.</p> <p>Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation createPrivilege“.</p>
updatePrivilege	<p><code>updatePrivilege</code> erlaubt es, die Menge der Operationen eines Privilegs wieder zu ändern. Diese Änderung hat Auswirkungen auf alle Rollen, die dieses Privileg verwenden.</p> <p>Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation updatePrivilege“.</p>
deletePrivileges	<p><code>deletePrivileges</code> löscht bestehende Privilegien. Diese Operation ist nur möglich, wenn keine Rolle diese Privilegien verwendet.</p> <p>Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation deletePrivileges“.</p>
getPrivileges	<p><code>getPrivileges</code> listet diese Privilegien eines Mandanten auf.</p> <p>Eine detaillierte Beschreibung finden Sie in Abschnitt „Operation getPrivileges“.</p>

2.6.2.3 Vorgehen beim Umstellen auf organisationspezifische Rollen

Für einen mit der Operation `createMandant` eingerichteten Mandanten gilt:

Durch das Einrichten der ersten organisationsspezifischen Rolle wird der Zugang über die mandantenglobale Rolle `Archivar` gesperrt. Wenn bereits mit der `Archivar`-Rolle gearbeitet wurde, müssen Sie die Archivierungsclients an die neu definierten Rollen für alle Organisationseinheiten anpassen. Um diese Migration ohne Störung des laufenden Betriebs durchzuführen, empfiehlt sich folgendes Vorgehen:

- Im ersten Schritt richtet der `MandantAdmin` für alle seine Organisationseinheiten die organisationsspezifische Rolle `Archivar` ein (diese Rolle erlaubt, wie die mandantenglobale Rolle, alle Operationen des Web-Service `ArchivingService`) und setzt dieselbe Zugangsberechtigung, die bisher für die globale `Archivar`-Rolle verwendet wurde. Jetzt können alle Archivierungsclients unverändert weiterarbeiten.
- Im zweiten Schritt definiert der `MandantAdmin` die organisationsspezifischen Rollen mit den von ihm festgelegten Operationen.
- Im dritten Schritt werden die Archivierungsclients auf den Zugang über die eben definierten Rollen umgestellt. Die Umstellung kann für die verschiedenen Organisationen zeitlich unabhängig erfolgen.
- Sind alle Archivierungsclients, die für eine Organisationseinheit verwendet werden, angepasst, kann für diese Organisationseinheit der `Archivar`-Zugang gesperrt werden (z.B. Rolle löschen). Jetzt ist für diese Organisation nur noch der Zugang über die Rollen mit selbstdefiniertem Funktionsumfang möglich.

2.6.3 Testmandant

Im produktiven Betrieb kann und darf ein Mandant nicht mehr gelöscht werden – die Langlebigkeit der SDOs und die Fixierung der SDOs mit WORM-Technologie sind hierfür wesentliche Gründe.

Dem produktiven Betrieb vorgeschaltet sind aber üblicherweise Testphasen, in denen z.B. für einen neuen Mandanten neue SDOTypes entwickelt werden. Um nach Abschluss der Testphase das Testmaterial einerseits löschen zu können, andererseits aber die aufgebauten Strukturen in den Produktivbetrieb übernehmen zu können, kann das Einrichten eines Testmandanten hilfreich sein.

2.6.3.1 Anlegen eines Testmandanten

Einen Testmandanten legen Sie mit der Operation `createMandant` (siehe [Abschnitt „Operation createMandant“](#)) oder der Operation `createMandantXAIP` (siehe Handbuch „["SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR" \(ISD2\) in Abschnitt "Literatur"](#)“) des Web-Service `ArchiveAdminService` an. Beim Erstellen eines Mandanten (und nur beim Neuanlegen) kann ein Mandant als Testmandant attribuiert werden. Ob ein Mandant das Attribut „Testmandant“ trägt, können Sie mit den Operationen `getMandantProperties` bzw. `getMandants` abfragen (siehe [Abschnitt „Operation getMandantProperties“](#) bzw. [Abschnitt „Operation getMandants“](#)).

Ein Testmandant hat folgende Eigenschaften:

- Die `SoftWorm`-Eigenschaft wird ignoriert, das heißt, Sie können die Dokumente jederzeit wieder von der Platte löschen.
- Alle weiteren Angaben werden genauso behandelt wie bei einer Archivierung im Produktivbetrieb.
- SDOs lassen sich mit `deleteSDO` löschen, auch wenn das Ablaufdatum noch nicht erreicht ist.
- Die Versiegelung wird genau wie bei produktiven Mandanten ausgeführt.
- Ein Testmandant lässt sich vollständig löschen. Lediglich die Anzahl der Versiegelungsaufträge, die durch inzwischen gelöschte Testmandanten durchgeführt wurden, wird (für alle gelöschten Testmandanten aufaddiert) weiter vermerkt.

2.6.3.2 Vom Testmandanten zum produktiven Mandanten

Für das Überführen eines Testmandanten in einen Produktivmandanten steht die Operation `convertTestMandant` des Web-Service `MandantAdminService` zur Verfügung (siehe [Abschnitt „Operation `convertTestMandant`“](#)).

Alle aktuell im Archiv befindlichen SDOs des zu konvertierenden Testmandanten (also alle während der Testphase archivierten und nicht gelöschten SDOs) werden „officialisiert“. Das heißt, bei der Ausführung von `convertTestMandant` werden folgende Aktionen ausgeführt:

- Für alle SDOs wird bei Bedarf die SoftWORM Funktionalität aktiviert (diese Aktion wird ja zum Zeitpunkt der Archivierung unter einem Testmandanten nicht durchgeführt).
- Die Aufbewahrungsfristen werden gültig, d.h. die SDOs können nicht mehr ohne Weiteres innerhalb der Aufbewahrungsfrist gelöscht werden.

Das Aktivieren von SoftWORM für alle SDOs kann unter Umständen länger dauern, deshalb wird diese Funktion asynchron ausgeführt. Sie läuft bei Neustart von SecDocs automatisch wieder an, wenn der Vorgang noch nicht abgeschlossen war.

Während der Laufzeit dieser Funktion sind keine weiteren Archivierungsoperationen für diesen Mandanten möglich.

Den aktuellen Status der Umwandlung können Sie über `getMandants` (Web-Service `ArchiveAdminService`, siehe ["Operation `getMandants`"](#)) bzw. `getMandantProperties` (Web-Service `MandantAdminService`, siehe ["Operation `getMandantProperties`"](#)) ermitteln. Solange das Element `State` in der Antwort den Wert `convertingToProductive` hat, ist die Operation noch nicht abgeschlossen.

Nach Abschluss der asynchronen Operation ist der Mandant im Status `productive`.

2.6.3.3 Vollständiges Löschen aller SDOs eines Testmandanten

Um alle Testdaten eines Testmandanten zu löschen, können Sie die Operation `clearAOIDs` (Web-Service `MandantAdminService`) verwenden. Dies ist zum z.B. notwendig, bevor Sie die Operation `convertTestMandant` ausführen, um die bereits registrierten SDOTypen anschließend produktiv zu nutzen, oder wenn Sie einen neuen Testlauf aufsetzen wollen.

Die Operation `clearAOIDs` (siehe "[Operation clearAOIDs](#)") kann nur für Testmandanten ausgeführt werden. Sie können sie also nur vor der Aktion `convertTestMandant` aufrufen. Die Operation `clearAOIDs` für einen Mandanten umfasst folgende Aktionen:

- Alle archivierten SDOs des Testmandanten werden gelöscht ohne Berücksichtigung der Aufbewahrungsfrist.
- Der Inhalt des Mandantenverzeichnisses und seine Unterverzeichnisse werden im Filesystem des Speichers vollständig gelöscht.
- AOIDs des Mandanten werden aus der Datenbank gelöscht. Anfragen mit dieser AOID werden mit einer Fehlermeldung abgewiesen, die darauf hinweist, dass die AOID unbekannt ist.

Da diese Aktionen länger dauern können, wird das Leeren asynchron gestartet und läuft bei Bedarf auch wieder an. Der Web-Service kehrt sofort wieder zurück.

Während der Laufzeit des asynchronen Löschvorgangs können für den Mandanten keine Archivierungsoperationen durchgeführt werden.

Den Status der Löschoperation kann man über `getMandants` (Web-Service `ArchiveAdminService`, siehe "[Operation getMandants](#)") bzw. `getMandantProperties` (Web-Service `MandantAdminService`, siehe "[Operation getMandantProperties](#)") ermitteln. Während die Operation läuft, wird im Element `State` der Wert `clearing` zurückgeliefert.

Nach Abschluss der asynchronen Operation ist der Mandant im Status `test`.

2.6.3.4 Löschen eines Testmandanten

Für das Löschen eines Testmandanten steht die Operation `deleteTestMandant` des Web-Service `ArchiveAdminService` zur Verfügung (siehe [Abschnitt „Operation deleteTestMandant“](#)). Zunächst werden alle AOIDs des Mandanten und danach der Mandant, seine Organisationsstruktur und alle seine Daten vollständig gelöscht.

Den Status der Löschoperation kann man über `getMandants` (Web-Service `ArchiveAdminService`, siehe ["Operation getMandants"](#)) bzw. `getMandantProperties` (Web-Service `MandantAdminService`, siehe ["Operation getMandantProperties"](#)) ermitteln. Während die Operation läuft, wird im Element `State` der Wert `deleting` zurückgeliefert.

Ist die Löschoperation erfolgreich abgeschlossen, liefert `getMandantProperties` - wie für einen noch nie eingerichteten Mandanten - die Fehlermeldung, dass die Zugangsdaten ungültig sind.

2.6.4 Web-Service für die Archivierung

Die Operationen für die Archivierung, die Basisfunktionen von SecDocs, orientieren sich an den Funktionen zur Ablage von elektronischen Dokumenten, wie sie von der TR-03125 des BSI und dem ArchiSafe-Schutzprofil (ACM_PP) gefordert werden. Diese Funktionen müssen die rechtliche Zulässigkeit der archivierten Dokumente gewährleisten (siehe [Abschnitt „Grundlagen“](#)).

SecDocs unterstützt die Archivierung von SDOs und von XAIP-Datenobjekten. Dieses Handbuch befasst sich im Wesentlichen mit der Archivierung von SDOs. Die Beschreibung der Archivierung von XAIP-Datenobjekten finden sie in einem eigenen Handbuch: „[SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR](#)“ ([SD2] in [Abschnitt "Literatur"](#)).

Die Operationen für die Archivierung von SDOs werden über den Web-Service `Archiving-Service` bereitgestellt. Dazu gehören unter anderem Operationen zum Anfordern einer eindeutigen Archivobjekt-ID (AOID), zur Übergabe eines SDOs an SecDocs (Archivierung), zur Recherche, sowie zum Lesen oder zum Löschen eines archivierten SDOs.

Die durch die TR vorgegebenen Basisfunktionen sind bezüglich der Archivierung mit Sec-Docs erweitert, insbesondere um Funktionen, mit denen archivierte Objekte gefunden und gelesen werden können, ohne dass die Anwendung die Objekt-ID kennen muss. Das heißt, zum Auffinden und Lesen archivierter Objekte ist eine führende Anwendung nicht mehr unbedingt notwendig. Man spricht daher auch von einem selbsttragenden Archiv.



Eine ausführliche Beschreibung dieser Schnittstelle finden Sie im [Kapitel „Web-Service für die Client-Anwendung“](#).

Web-Service ArchivingSRService

Der Web-Service für die Archivierung von SDOs ist in zwei Teile aufgespalten. Der `ArchivingService` enthält die vordefinierten Operationen, bei denen keine Anpassungen in der zugehörigen WSDL Datei erfolgen müssen. Der Web Service `ArchivingSRService` enthält die SDOType-spezifischen `submitSDO`-, `replaceSDO`- und `retrieveSDO`-Operationen, die Sie für jeden neuen SDOType in der zugehörigen WSDL-Datei hinzufügen müssen. Im Folgenden wird meistens nur noch der `ArchivingService` genannt. In der Regel ist damit aber auch der `ArchivingSRService` eingeschlossen.

Die angepasste .wsdl-Definitionsdatei wird von SecDocs nicht benötigt und ist nur für die Client-Anwendung von Bedeutung.

Näheres hierzu ist im [Abschnitt „SDO-Typ-spezifische Operationen des Web-Service ArchivingSRService“](#) zu finden.

2.6.5 Web-Services für die Administration

SecDocs bietet für die Administration ein zweistufiges Konzept. Die Administrationsschnittstelle unterscheidet zwischen der Administration des gesamten Archivs und der Mandantenadministration.

- Administration des gesamten Archivs:

Die Operationen für die Administration des Archivs in seiner Gesamtheit werden mit Hilfe des Web-Service `ArchiveAdminService` bereitgestellt. Dazu gehören unter anderem Operationen zum Einrichten der verwendbaren Zeitstempelanbieter sowie das Einrichten und Verwalten von Mandanten. Über die Archivadministration ist kein Einblick in die fachlichen und organisatorischen Gegebenheiten der einzelnen Mandanten möglich.



Eine ausführliche Beschreibung dieser Schnittstelle finden Sie im [Abschnitt „Web-Service ArchiveAdminService“](#).

- Mandantenadministration:

Die Operationen, die sich auf den jeweiligen Archivbereich eines Mandanten beziehen, werden mit Hilfe des Web-Service `MandantAdminService` bereitgestellt. Dazu gehören unter anderem Operationen zum Registrieren von SDO-Typen sowie zum Einrichten von spezifischen Zugängen zum Archiv für die Client-Software.



Eine ausführliche Beschreibung dieser Schnittstelle finden Sie im [Abschnitt „Web-Service MandantAdminService“](#).

2.6.6 Hinweise zur Migration von Web-Service-Clients

Typischerweise werden zum Erstellen von Web-Service-Clients Tools benutzt, die Wrapper-Code zur Nutzung der Web-Service-Schnittstellen erzeugen (Beispiel: *wsimport* für Java-Clients, *wsdl.exe* für .NET).

Zur Nutzung der V3.1 Web-Service-Schnittstellen müssen Sie den Wrapper-Code auf Basis der neuen V3.1 Daten (WSDLs und Schemas) erzeugen und den alten Code mit diesem neuen Wrapper-Code kompilieren.

Eine Umstellung ist jedoch nur dann notwendig, wenn Sie die neuen Funktionen der V3.1 nutzen wollen, da die Schnittstellen der Vorgängerversionen kompatibel weiter unterstützt werden.

Da in den neuen Web-Services neue XML-Namespace-Namen verwendet werden, müssen z.B. in Java basierten Clients in den Import-Anweisungen die Package-Namen angepasst werden. Die Namen der generierten Wrapper-Klassen selbst bleiben in der Regel unverändert.

Zur Vereinfachung wird der fertige Wrapper-Code für Java mitausgeliefert. Nach Installation von SecDocs finden sich diese in dem Verzeichnis `jaxws/lib`.

2.6.6.1 Die Endpoint-URLs, Namespaces und Schemata der V3.1

Im Folgenden werden die für die Web-Services der SecDocs V3.1 jeweils gültigen End-point-URLs und Target-Namespace sowie Schema-Dateien (und deren Target-Namespace) aufgelistet.

Dabei bezeichnet *secdocsHost* jeweils den Namen des Servers, auf dem die SecDocs-Anwendung läuft und *secdocsPort* den Port (Defaultport für WildFly: 8080), auf dem die SecDocs Anwendung lauscht.

Nach der Installation finden sich die genannten Schema- bzw. WSDL-Dateien in folgenden Verzeichnissen:

unter `http://secdocsHost:secdocsPort/archiver/schemas/3.1/`
im Verzeichnis `schemas/3.1/` der SecDocs Installation.
Default-Verzeichnisname: `/home/secdocs/schemas/3.1/`

Web-Service ArchiveAdminService

(Operationen zur Archiv-Administration)

WSDL: **ArchiveAdmin.wsdl**
Endpoint-URL: `http://secdocsHost:secdocsPort/archiver/ws/3.1/archiveAdmin`
Target-Namespace: `http://ts.fujitsu.com/secdocs/ws/v3_1/archiveAdmin`

In WSDL verwendete Schemadateien Target-Namespace

`secdocs.xsd` `http://ts.fujitsu.com/secdocs/v3_1/secdocs`
`AdminData.xsd` `http://ts.fujitsu.com/secdocs/v3_1/adminData`
`AdminUpdateData.xsd` `http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData`

Web-Service MandantAdminService

(Operationen für die mandantenspezifische Administration)

WSDL: **MandantAdmin.wsdl**
Endpoint-URL: `http://secdocsHost:secdocsPort/archiver/ws/3.1/mandantAdmin`
Target-Namespace: `http://ts.fujitsu.com/secdocs/ws/v3_1/mandantAdmin`

In WSDL verwendete Schemadateien Target-Namespace

`secdocs.xsd` `http://ts.fujitsu.com/secdocs/v3_1/secdocs`
`AdminData.xsd` `http://ts.fujitsu.com/secdocs/v3_1/adminData`
`AdminUpdateData.xsd` `http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData`

Web-Service ArchivingService

(SDOTyp-unabhängige Operationen für den ArchivierungsClient)

WSDL: **Archiving.wsdl**

Endpoint-URL: `http://secdocsHost:secdocsPort/archiver/ws/3.1/archiving`

Target-Namespace: `http://ts.fujitsu.com/secdocs/ws/v3_1/archiving`

In WSDL verwendete Schemadateien

	Namespace
secdocs.xsd	<code>http://ts.fujitsu.com/secdocs/v3_1/secdocs</code>
ArchivingData.xsd	<code>http://ts.fujitsu.com/secdocs/v3_1/archiving</code>
query/sparql-protocoltypes.xsd	<code>http://www.w3.org/2007/SPARQL/protocol-types#</code>
query/rdf.xsd	<code>http://www.w3.org/1999/02/22-rdf-syntax-ns#</code>
query/result.xsd	<code>http://www.w3.org/2007/SPARQL/results#</code>
query/xml.xsd	<code>http://www.w3.org/XML/1998/namespace</code>

Web-Service ArchivingSRService

(SDOTyp-spezifische Operationen Submit/Retrieve)

WSDL: **ArchivingSR.wsdl**

Endpoint-URL: `http://secdocsHost:secdocsPort/archiver/ws/3.1/archiving`

Target-Namespace: `http://ts.fujitsu.com/secdocs/ws/v3_1/archiving`

In WSDL verwendete Schemadateien

	Namespace
secdocs.xsd	<code>http://ts.fujitsu.com/secdocs/v3_1/secdocs</code>
ArchivingData.xsd	<code>http://ts.fujitsu.com/secdocs/v3_1/archiving</code>

Als Beispiel für ein SDO-Type-Schema findet sich

im Verzeichnis `schemas/3.1/samples`

die Datei: `MultiDocument.xsd`

Namespace: `http://ts.fujitsu.com/secdocs/sdosamples/v1_0/multidocument`

Web-Service S4

(Operationen für den ArchivierungsClient)

Siehe Handbuch "[SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR](#)" ([SD2] in Abschnitt "[Literatur](#)")

2.7 Unterstützung von Policies

Mit SecDocs 3.1. wird ein neues OpenLimit CryptoModule ausgeliefert, das die Regeln der europäischen eIDAS-Verordnung implementiert. Teil des neuen Konzepts ist es, dass Signaturprüfungen durch sogenannte Policies gesteuert werden können. Policies können in SecDocs an folgenden Stellen hinterlegt werden:

- pro SDOType
- pro Mandant im Falle eines Mandanten, der die S4-Schnittstelle zur Archivierung verwendet , da hier nur ein vordefinierter SDOTyp unterstützt wird
- pro TSP

In SecDocs 3.1. wurden die folgenden Funktionen angepasst, um solche Policies zu hinterlegen, beziehungsweise die hinterlegte Policy abzufragen:

- `createMandantXAIP (Web-Service ArchiveAdminService)`
- `createTSP(Web-Service ArchiveAdminService)`
- `getTSP(Web-Service ArchiveAdminService)`
- `updateTSP(Web-Service ArchiveAdminService)`
- `createSDOType (Web-Service MandantAdminService)`
- `modifySDOType (Web-Service MandantAdminService)`
- `modifyXAIP(Web-Service MandantAdminService)`
- `getSchemaDataSDO (Web-Service ArchivingService)`



Um nachvollziehen zu können, welche Policy beim Archivieren eines SDOs verwendet wurde, empfiehlt es sich das Attribut Name und das Element Description im Root-Element der Policy mit einer aussagekräftigen Beschreibung zu füllen.

Diese Information findet sich dann im Verification-Report im SimpleReport-Element und da im Policy-Element wieder.

Beispiel:

Policy

```
<ConstraintsParameters Name="My special policy"
  xmlns="http://dss.esig.openlimit.com/validation/policy">
  <Description>
    Diese Policy passt die Signaturprüfung in folgenden Punkten an...
  </Description>
```

Verification-Report:

```
<ns6:report xmlns:ns6="com.openlimit.esig.dss.report"
  xmlns:ns5="http://verification.dss.esig.openlimit.com/"
  xmlns:ns2="http://dss.esig.openlimit.com/validation/detailed-report"
  xmlns:ns4="http://dss.esig.openlimit.com/validation/simple-report"
  xmlns:ns3="http://dss.esig.openlimit.com/validation/diagnostic">
  <detailedReport>
    ...
  </detailedReport>
  ...
  <simpleReport>
    <ns4:Policy>
      <ns4:PolicyName> My special policy </ns4:PolicyName>
      <ns4:PolicyDescription> Diese Policy passt die
        Signaturprüfung in
          folgenden Punkten an... </ns4:PolicyDescription>
    </ns4:Policy>
  </simpleReport>
</ns6:report>
```

2.8 Sicherer Betrieb

Dieses Kapitel enthält folgende Abschnitte:

- Anforderungen für den sicheren Betrieb
- Logging

2.8.1 Anforderungen für den sicheren Betrieb

Die Ablaufumgebung muss vor Angriffen von außen (zum Beispiel durch Schadsoftware, nicht zugelassene Netzwerkverbindungen oder Viren) geschützt sein. Dazu muss ein leistungsfähiger Virens Scanner und eine sicher eingestellte Firewall eingesetzt werden. Der Virens Scanner muss regelmäßig aktualisiert werden. Für das Betriebssystem müssen regelmäßig die verfügbaren Sicherheitsupdates eingespielt werden.

SecDocs sowie die dem Produkt unterliegende IT-Plattform müssen Sie in einer sicheren Umgebung installieren. Auf den Servern, auf denen SecDocs abläuft bzw. auf denen sich Daten befinden, auf die SecDocs zugreifen soll, darf keine weitere Software installiert sein, die nicht für den Betrieb des Integrationsproduktes benötigt wird.

Die Server müssen gegen unautorisierten, physischen und logischen Zugriff sowie gegen Veränderung geschützt sein.

Alle TCP/IP-Verbindungen, die von SecDocs genutzt werden, müssen gegen unautorisierte Zugriffe und Abhörmöglichkeiten physisch (z.B. durch Isolierung des verwendeten lokalen Netzes) oder logisch (z.B. mit Verschlüsselungssoftware und Firewalls) geschützt sein.

Das als Basis für SecDocs dienende Betriebssystem muss vom Administrator des Produkts stets aktuell gehalten werden, d.h. die vom Hersteller des Betriebssystems veröffentlichten, sicherheitskritischen Updates sind einzuspielen. Ebenso muss der Administrator ein adäquates Virenschutzprogramm mit aktuellen Virensignaturen verwenden. Die Systemzeit der Server muss mit einer vertrauensvollen Zeitquelle synchronisiert sein (z.B. durch eine Funkuhr oder durch Verbindung mit einem NTP-Server).

Der von der SecDocs Security-Komponente "OpenLimit CryptoModule" angebotene Web-Service darf beim Betrieb des Produkts ausschließlich lokal angeboten werden, er muss also bei der Installation des Produkts auf das sogenannte "localhost"-Interface gebunden werden.

Die IT-Infrastruktur muss durch Komponenten (von Drittherstellern) vor Viren und Schadsoftware geschützt sein. Weiterhin muss die IT-Infrastruktur vor Netzwerk-basierten Angriffen geschützt sein. Potenzielle Angriffe über das Internet, ein angeschlossenes Intranet, einen manuellen Zugriff Unbefugter oder Datenaustausch per Datenträger müssen durch die bestehenden Sicherheitsvorkehrungen in der Einsatzumgebung mit hoher Sicherheit abgewehrt werden.

Die SecDocs-Komponenten und die von ihnen genutzten Daten müssen durch geeignete Schutzmechanismen davor geschützt werden, dass sie nicht autorisiert verändert werden.

Der Administrator muss vertrauenswürdig sein und die im Handbuch beschriebenen Installations- und Bedienungsanweisungen sorgfältig befolgen. Weiterhin hat er alle ihm zugänglichen Informationen zum Produkt vertraulich zu behandeln. Die benutzten Passwörter sind von ihm sicher zu verwahren. Insbesondere soll der Archivadministrator das Initialpasswort nach der Erstinstallation umgehend ändern.

Für den den nach TR-ESOR zertifizierten Betrieb muss SecDocs so konfiguriert werden, dass die Web-Services für die Administration nur über eine HTTPS-Verbindung erreichbar sind. Darüber hinaus müssen die Properties `checkTspProductionTime` und `tresor.certified` im Betrieb beide auf `true` gesetzt sein.

2.8.2 Logging

SecDocs protokolliert den Zugriff auf die TR-ESOR-Middleware bzw. auf den Langzeitspeicher zu Zwecken der Ablage, des Änderns, des Abrufs der Daten oder des Abrufs von Beweisdaten oder auch des Löschens abgelegter Dokumente und Daten.

Dazu bietet SecDocs die folgenden Möglichkeiten der Protokollierung an:

- Das Audit-Logging zeichnet jeden Aufruf einer Webservice-Operation auf und ermöglicht es, jede Aktion einem Verantwortlichen zuzuordnen.
- Das SecDocs-Logging protokolliert die Aktionen in SecDocs und im WildFly Application Server.

Zusätzlich sollten Sie das Umfeld von SecDocs überwachen.

Zugriff auf die Log-Dateien

Der Zugriff auf die Audit-Log Dateien eines Mandanten ist über den Webservice `MandantAdmin` möglich. Ein direkter Zugriff auf das SecDocs-Logging über die SecDocs-Web-Services ist dagegen nicht möglich. Nur der Systemadministrator des Produkts kann mit den Mitteln des Dateiverwaltungssystems direkt auf die Log-Dateien zugreifen.

Der Systemadministrator muss daher vertrauenswürdig sein und die Log-Dateien müssen durch geeignete Schutzmechanismen vor unautorisierter Veränderung geschützt werden.

Für Zugriffe auf die Log-Dateien können Sie z.B. eine eigene Benutzerkennung einrichten und diese Kennung der Gruppe `secdocs` hinzufügen.

Das Logging in SecDocs ist ausführlich im Abschnitt "[Logging und Fehlerbehandlung](#)" beschrieben.

Im Handbuch "[SecDocs Installations- und Einsatz-Anleitung für SecDocs und V3 Server](#) ([SD3] im Abschnitt "[Literatur](#)") finden Sie Hinweise, wie sie die Logging-Konfiguration anpassen können.

3 Web-Service für die Client-Anwendung

Dieses Kapitel beschreibt die Arbeit mit dem Web-Service `ArchivingService`. Dieser Web-Service stellt die Operationen für die Archivierung zur Verfügung. Die Client-Software nutzt dabei den mandanten- und organisationseinheitsspezifischen Zugang, der mit Hilfe des Web-Service `MandantAdminService` definiert wurde (siehe [Kapitel "Archiv- und Mandantenadministration"](#)). Der Web-Service `ArchivingService` ist unter folgender URL zu erreichen:

```
http://secdocsHost:secdocsPort/archiver/ws/3.1/archiving
```

i Ein Zugriff auf eine AOID über Operationen des Web-Service `ArchivingService` ist nur möglich, wenn die Organisation, die im Header-Element `orgID` angegeben ist, der Organisation entspricht, die die AOID angelegt hat.

Im ersten Teil des Kapitels finden Sie einen kurzen Einstieg in den Aufbau der SOAP-Nachrichten, die in SecDocs verwendet werden.

i Alle Beispiele, die bei der Beschreibung der einzelnen Seiten angegeben sind, sind Auzüge aus SOAP-Nachrichten und nicht ohne Modifikationen ablauffähig.

3.1 SOAP-Nachrichten

- Request und Response
- Fault-Message
- Zugangsprüfung

3.1.1 Request und Response

Eine Client-Anwendung kommuniziert mit einem Web-Service über SOAP-Nachrichten (SOAP-Request und SOAP-Response) nach SOAP 1.1. Jeder SOAP-Request und jede SOAP-Response ist eine XML-Nachricht mit einem SOAP-Envelope als Wurzelement, in den die beiden Sub-Elemente SOAP-Header und SOAP-Body eingebettet sind. Die SOAP-Nachrichten in SecDocs enthalten keine Attachments. Als Nachrichtenformat wird "Document literal" verwendet.

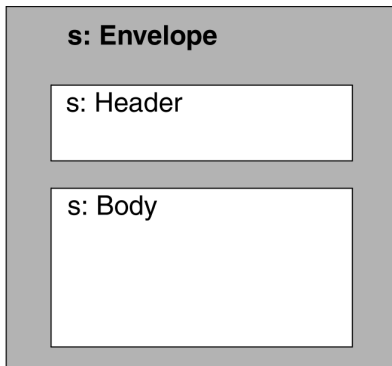


Bild 8: SOAP-Envelope mit Header und Body

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    ...
  </s:Header>
  <s:Body>
    ...
  </s:Body>
</s:Envelope>
```

Die Antwort auf einen SOAP-Request ist bei erfolgreicher Ausführung eine SOAP-Response, die für die Operation spezifisch ist. Im Fehlerfall wird eine Fault-Message zurückgegeben (siehe Abschnitt „[Fault-Message](#)“).

Request-Header

Der Header eines SOAP-Requests an SecDocs enthält die Operation, die ausgeführt werden soll, und die Zugangsdaten für die Anmeldung am Web-Service.



Die ausführliche Beschreibung finden Sie im [Abschnitt "Request-Header"](#).

Response-Header

Der SOAP-Header einer Response, die von SecDocs erzeugt wurde, ist in der Regel mit dem SOAP-Header des SOAP-Requests identisch.

Request-/Response-Body

Der Aufbau des Bodys in SOAP-Request und SOAP-Response ist operationsspezifisch.



Die ausführliche Beschreibung finden Sie im [Abschnitt "Request- und Response-Body"](#).

3.1.2 Fault-Message

SecDocs verwendet die beiden folgenden HTTP Server Response Codes:

200 Erfolgreiche Bearbeitung.

500 Ein Fehler ist aufgetreten; SecDocs sendet eine SOAP Fault-Message als Antwort.

Alle Web-Services senden im Fehlerfall eine Fault-Message. Diese Fault-Message enthält die folgenden SecDocs-Daten.

Definition: `secdocs.xsd`

Element `faultDetails` vom Datentyp `TFaultDetails`

```
<xs:complexType name="TFaultDetails">
  <xs:sequence>
    <xs:element name="nodeName" type="tns:TNonEmptyString"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="requestNumber" type="xs:long"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="operation" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="errorMessage" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="errorCode" type="xs:integer"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="errorDetail" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="additionalData" type="xs:base64Binary"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="migSafeFaultDetails"
      type="tns:TMigSafeFaultDetails"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

`nodeName`

TNonEmptyString:

optional; Name des Knotens, auf dem der Fehler aufgetreten ist.

Dieses Element wird nur ausgegeben, wenn Sie den Konfigurationsparameter

`nodeNameInFaultMessage` in der Datei `secdocs.properties` auf `"true"` gesetzt haben.

`requestNumber`

Nummer des Request.

SecDocs ordnet jedem Request aufsteigend eine Nummer zu, die im Fehlerfall auf der Server-Seite protokolliert wird. Mit Hilfe dieser Nummer kann die Verbindung zwischen Fehlermeldung auf der Client-Seite und Fehlermeldung auf der Server-Seite hergestellt werden.

`operation`

optional; Name der aufgerufenen Operation.

Ist ein Fehler aufgetreten, bevor die Operation ermittelt werden konnte, ist dieses Element leer.

`errorMessage`

Text der Fehlermeldung.



Eine Liste der Fehlermeldungen finden Sie im Handbuch ["SecDocs-Rückgabewerte"](#) ([SD4] im [Abschnitt "Literatur"](#)).

Bezieht sich `errorMessage` auf Return Codes der Komponenten MigSafe / OverSign, so finden Sie dafür detaillierte Informationen im Handbuch ["MigSafe / OverSign - Rückgabewerte"](#) ([SD6] im [Abschnitt "Literatur"](#)).

`errorCode`

Error-Code der Fehlermeldung.

`errorDetail`

optional; Zusatzinformationen zur Fehlermeldung.

`additionalData`

base64Binary:

optional; Weitere Zusatzinformationen zur Fehlermeldung.

Genauere Angaben bzgl. Inhalt und Format dieser Fehlerinformation finden sich bei den einzelnen Operationen.

Derzeit wird `additionalData` ausschließlich bei den Operationen `createTSP` und `updateTSP` verwendet. Siehe hierzu die Abschnitte [„Operation createTSP“](#) und [„Operation updateTSP“](#).

`migSafeFaultDetails`

optional; Information zum Grund einer Ablehnung der Operation `submitSDO` (siehe ["Operation submitSDO"](#)) oder `replaceSDO` (siehe ["Operation replaceSDO"](#)).

Datentyp `TMigSafeFaultDetails`, siehe ["Datentyp TMigSafeFaultDetails"](#).

Dieses Element wird nur ausgegeben, wenn die Verifikation einer oder mehrerer Signaturen fehlgeschlagen ist. In diesem Fall erhalten Sie für jeden Signatur-Container (siehe ["Fachwörter"](#)), der mindestens eine negativ geprüfte Signatur enthält, genau ein Element `migSafeFaultDetail`.

Datentyp `TMigSafeFaultDetails`

```
<xs:complexType name="TMigSafeFaultDetails">
  <xs:sequence>
    <xs:element name="migSafeFaultDetail" type="tns:TMigSafeFaultDetail"
      minOccurs="1" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
```

`TMigSafeFaultDetail`

Datentyp TMigSafeFaultDetail, siehe unten.

Element migSafeFaultDetail *Datentyp* TMigSafeFaultDetail

```
<xs:complexType name="TMigSafeFaultDetail">
  <xs:sequence>
    <xs:element name="info" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="ecode" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="xmlVerificationProtocol"
      type="xs:base64Binary"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="htmlVerificationProtocol"
      type="xs:base64Binary"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

info Text der Fehlermeldung.

ecode Error-Code der Signaturprüfung.

xmlVerificationProtocol

Verifikationsprotokoll des CryptoModule mit allen Detail-Angaben für die Signaturprüfung.

htmlVerificationProtocol

base64Binary:

optional; als XHTML-Datei aufbereitetes Verifikationsprotokoll.

i Die Generierung der XHTML-Datei ist standardmäßig ausgeschaltet. Wenn Sie diese Protokolle erzeugen wollen, müssen Sie den Konfigurationsparameter `doHTMLProtocolsOnRetrieval` in der Datei `secdocs.properties` auf "true" setzen (siehe [Abschnitt „Konfigurationsdatei secdocs.properties“](#)).

Beispiele

Dieses Beispiel zeigt eine vollständige SOAP-Response, die eine Fault-Message mit einer SecDocs-Fehlermeldung enthält. Die Fault-Message ist **magenta markiert**, und die SecDocs-Fehlermeldung entsprechend dem SOAP Standard ist **grün markiert**:

```

<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
    http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Header/>

  <soap:Body>

    <soap:Fault>

      <faultcode>soap:Client</faultcode>
      <faultstring><![CDATA[REQ0009 : SDO not yet expired]]></faultstring>
      <faultactor>/archiver/ws/3.1/archiving</faultactor>

      <detail>

        <tns:faultDetails
          xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">

          <tns:requestNumber>55</tns:requestNumber>

          <tns:operation>deleteSDO</tns:operation>
          <tns:errorMessage>

            <![CDATA[REQ0009 : SDO not yet expired]]>

          </tns:errorMessage>
          <tns:errorCode>409</tns:errorCode>

        </tns:faultDetails>

      </detail>
    </soap:Fault>

  </soap:Body>

</soap:Envelope>

```

Das folgende Beispiel zeigt eine SOAP-Response, die eine Fault-Message mit einer P7S-Signatur enthält. Die farbigen Markierungen haben dabei folgende Bedeutung:

Magenta: Fault-Message

Grün: SecDocs-Fehlermeldung

Blau: Fehlermeldung der Komponenten MigSafe / OverSign

```
<soap:Envelope
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
    http://schemas.xmlsoap.org/soap/envelope/">

<soap:Header/>

<soap:Body>

  <soap:Fault>
    <faultcode>soap:Client</faultcode>

    <faultstring>

      <![CDATA[REQ0025 : Error in OpenLimit function
        migSafe.ol_submitSDO: Signature verification failed!

        (ECODE: c021c018) OpenLimit detail info:

        /{http://ts.fujitsu.com/secdocs/sdosamples/mydocument}
        myDocument[1]/file[1]/signature[1]

        (ECODE: c0212000) ]]>

    </faultstring>
    <faultactor>/archiver/ws/3.1/archiving</faultactor>

    <detail>

      <tns:faultDetails
        xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">

        <tns:requestNumber>3</tns:requestNumber>

        <tns:operation>submitSDO</tns:operation>
        <tns:errorMessage>

          <![CDATA[REQ0025 : Error in OpenLimit function

            migSafe.ol_submitSDO: Signature verification failed!

            ECODE: c021c018) OpenLimit detail info:
            /{http://ts.fujitsu.com/secdocs/sdosamples/mydocument}
            myDocument[1]/file[1]/signature[1]

            (ECODE: c0212000) ]]>

          </tns:errorMessage>
```

```
        <tns:errorCode>425</tns:errorCode>

    </tns:faultDetails>
    <tns:migSafeFaultDetails>

        <tns:migSafeFaultDetail>

            <tns:info>
                /{http://ts.fujitsu.com/secdocs/sdosamples/mydocument}
                myDocument[1]/file[1]/signature[1]
            </tns:info>
            <tns:ecode>c0212000</tns:ecode>

            <tns:xmlVerificationProtocol>

                PD94bWwgdm ... VzWx0Pgo=
            </tns:xmlVverificationProtocol>

        </tns:migSafeFaultDetail>

    </tns:migSafeFaultDetails>
</detail>

</soap:Fault>

</soap:Body>

</soap:Envelope>
```

3.1.3 Zugangsprüfung

Der Zugang zu einem Web-Service in SecDocs ist rollenbasiert.

Die Daten für die Zugangsprüfung müssen im SOAP-Header unter `TSecurity` im Element `principal` angegeben werden (siehe [Abschnitt „Request-Header“](#)).

Bei Zugriff auf SDOs muss die AOID und/oder die COID sowie die Organisation angegeben werden, unter der das SDO archiviert wurde.

Element `principal`

Das Element `principal` enthält folgende Angaben (siehe auch "Datentyp `TPrincipal`" im Abschnitt ["Request-Header"](#)):

role Rolle für die Anmeldung am Web-Service. Die Rolle legt fest, welche Operationen ausgeführt werden können.
Für den Web-Service `ArchivingService` können durch den `MandantAdmin` Rollen und deren Privilegien frei definiert werden (siehe [Abschnitt „Berechtigungskonzept“](#)).

Vordefinierte Rollen:

Web-Service	Rolle
<code>ArchivingService</code>	<code>Archivar</code>
<code>MandantAdminService</code> ¹	<code>MandantAdmin</code>
<code>MandantAdminService</code> ²	<code>SecDocs_MandantAuditor</code>
<code>ArchiveAdminService</code>	<code>ArchiveAdmin</code>

¹ Die Rolle erlaubt die Ausführung aller Operationen außer `getAuditLogFileNames` und `getAuditLogFile`

² Die Rolle erlaubt ausschließlich die Ausführung der Operationen `getAuditLogFileNames` und `getAuditLogFile`

mandant Mandantenname, für den die Operation ausgeführt werden soll. Der Mandantenname legt fest, auf welche Ressourcen des Archivs sich die Operation bezieht.

Die Mandanten, die das Archiv verwenden dürfen, werden mit Hilfe des Web-Service `ArchiveAdminService` eingerichtet.

orgID Name einer Organisationseinheit. `orgID` legt den Ablageort der Archivobjekte für die Organisationseinheit innerhalb des Mandanten fest. Der Ablageort des Archivguts unterschiedlicher Organisationseinheiten wird somit klar voneinander getrennt.

i Es können nur für die archivierten Objekte Operationen ausgeführt werden (z.B. Lesen), die auch unter dieser Organisation archiviert wurden.

Credential

Jedem Element `principal` ist ein Berechtigungsnachweis, ein Credential, zugeordnet. Derzeit ist das Credential in Form eines Passworts realisiert. In Folgeversionen sind weitere Möglichkeiten der Autorisierung vorgesehen.

Das Credential und `principal` sind im Datentyp `TSecurity` zusammengefasst und liefern damit die Basisinformation für die Autorisierung (siehe "Datentyp `TSecurity`" im Abschnitt "[Request-Header](#)").

3.2 Request-Header

Der Header eines SOAP-Requests bezeichnet die Operation, die ausgeführt werden soll, und die Autorisierungsinformationen für die Anmeldung am Web-Service.

Im Schema `secdocs.xsd` ist `elementFormDefault="qualified"` eingestellt.

Daher müssen alle Elemente im SOAP-Header durch einen Namespace qualifiziert angegeben werden.

Definition der Typen im Header: Datentyp `TSoapHeader` in `secdocs.xsd`

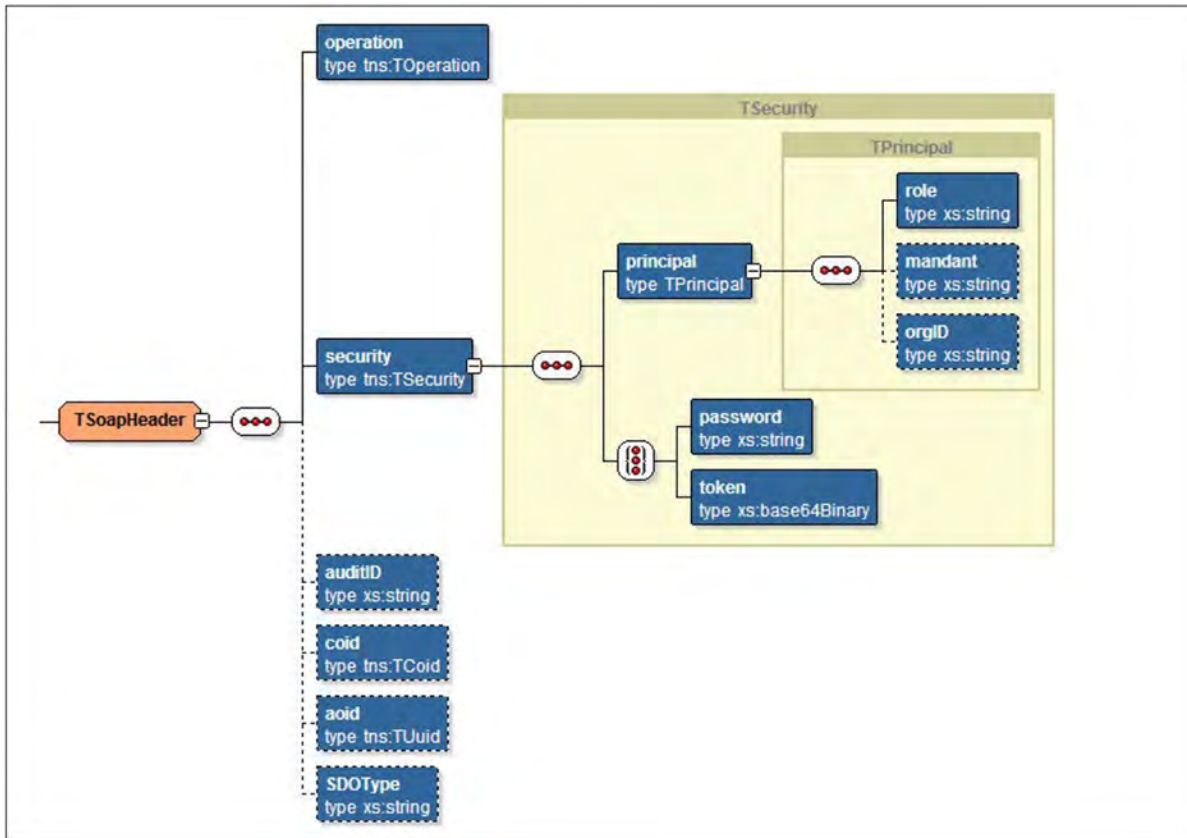


Bild 9: Struktur eines Request-Headers



Sequence: die Elemente müssen genau in der dargestellten Reihenfolge auftreten.



Choice: eines der angegebenen Elemente



Pflichtangabe



Optionale Angabe

Datentyp `TSoapHeader`

```

<xs:complexType name="TSoapHeader">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="operation" type="tns:TOperation"/>
    <xs:element name="security" type="tns:TSecurity"/>
    <xs:element name="auditID" type="xs:string" minOccurs="0"/>
    <xs:element name="aoid" type="tns:TUuid" minOccurs="0"/>
    <xs:element name="SDOType" type="xs:string" minOccurs="0"/>
    <xs:element name="coid" type="tns:TCoid"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>

```

i Welche der im Folgenden beschriebenen Elemente anzugeben sind, hängt von der jeweiligen Operation ab und ist dort im Einzelnen beschrieben.

Insbesondere muss bei folgenden Operationen mindestens eines der Elemente `aoid` oder `coid` angegeben werden:

- `submitSDO`, siehe "[Operation submitSDO](#)"
- `replaceSDO`, siehe "[Operation replaceSDO](#)"
- `retrieveSDO`, siehe "[Operation retrieveSDO](#)"
- `deleteSDO`, siehe "[Operation deleteSDO](#)"
- `forceDeleteSDO`, siehe "[Operation forceDeleteSDO](#)"
- `statusSDO`, siehe "[Operation statusSDO](#)"
- `listSDOVersions`, siehe "[Operation listSDOVersions](#)"
- `retrieveMetaData`, siehe "[Operation retrieveMetaData](#)"
- `metaDataSDO`, siehe "[Operation metaDataSDO](#)"
- `requestForEvidence`, siehe "[Operation requestForEvidence](#)"
- `moveSDO`, siehe "[Operation moveSDO](#)"
- `setExpirationDateTimeSDO`, siehe "[Operation setExpirationDateTimeSDO](#)"
- `forceDeferredSDO`, siehe "[Operation forceDeferredSDO](#)"
- `getSchemaDataSDO`, siehe "[Operation getSchemaDataSDO](#)"

`operation`

Name der Service-Operation, die ausgeführt werden soll:

Datentyp `TOperation`, siehe "[Datentyp TOperation](#)"

`security`

Autorisierungsinformation für die Anmeldung am Web-Service (Zugangsdaten und Passwort):

Datentyp `TSecurity`, siehe "[Datentyp TSecurity](#)"

`auditID`

string:

optional; Diese Zeichenkette wird in das Audit-Protokoll übertragen.

Die Client-Software meldet sich bei SecDocs mit einer Rolle an. Um im Audit-Protokollnachvollziehen zu können, wer der ausführende Benutzer war, kann die Client-Software hier z.B. die Anmeldedaten dieses Benutzers angeben.

aoid

string (Universally Unique Identifier gemäß Standard IETF RFC 4122):

AOID für das SDO, auf das sich die Operation bezieht.

Statt der AOID kann bei den Operationen des Archiving Webservice auch eine COID angegeben werden. Werden beide Elemente mit Werten versorgt, müssen sich COID und AOID auf dasselbe SDO beziehen.

SDOType

string:

Typ eines SDOs

Die Struktur eines SDOs muss vor der Operation `submitSDO` oder `replaceSDO` mit dem Web-Service `MandantAdminService` bei SecDocs registriert worden sein. Die SDO-Typen werden mit dem Web-Service `MandantAdminService` bei SecDocs registriert.

Die Angabe für `SDOType` ist nicht case-sensitiv.

coid

TCoid:

optional; COID für das SDO, auf das sich die Operation bezieht.

Die COID (Client Object Identifier) ist ein vom Client vergebbarer Bezeichner zur Identifizierung eines Archivobjekts. Die COID kann bei den Operationen des Archiving Webservice statt einer AOID angegeben werden. Werden beide Elemente mit Werten versorgt, müssen sich COID und AOID auf dasselbe SDO beziehen.

Die COID kann ein beliebiger nichtleerer String sein. Sie darf jedoch keine Steuerzeichen enthalten und ist auf maximal 256 Zeichen beschränkt.

Die COID ist case-sensitiv, sie muss also genau so verwendet werden, wie sie definiert ist.

Datentyp `TCoid`, siehe "[Datentyp TCoid](#)"

Datentyp `TOperation`

```

<xs:simpleType name="TOperation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="getAOID" />
    <xs:enumeration value="getAOIDWithRef" />
    <xs:enumeration value="submitSDO" />
    <xs:enumeration value="replaceSDO" />
    <xs:enumeration value="retrieveSDO" />
    <xs:enumeration value="deleteSDO" />
    <xs:enumeration value="forceDeleteSDO" />
    <xs:enumeration value="statusSDO" />
    <xs:enumeration value="listSDOVersions" />
    <xs:enumeration value="retrieveMetaData"/>
    <xs:enumeration value="metaDataSDO"/>
    <xs:enumeration value="requestForEvidence"/>
    <xs:enumeration value="sparqlQuery"/>
    <xs:enumeration value="navigate"/>
    <xs:enumeration value="getAccountingData"/>
    <xs:enumeration value="moveSDO" />
    <xs:enumeration value="setExpirationDateTimeSDO"/>
    <xs:enumeration value="forceDeferredSDO"/>
    <xs:enumeration value="getSchemaDataSDO"/>
    <xs:enumeration value="getVersion" />
    ....
  </xs:restriction>
</xs:simpleType>

```

Für den Web-Service ArchivingService sind folgende Operationen möglich:

getAOID

Reservieren einer AOID, siehe ["Operation getAOID"](#)

getAOIDWithRef

Reservieren einer AOID und einer oder mehrerer externer Referenz-IDs, siehe ["Operation getAOIDWithRef"](#)

submitSDO

Archivieren eines SDOs, siehe ["Operation submitSDO"](#)

replaceSDO

Ersetzen eines archivierten SDOs, siehe ["Operation replaceSDO"](#)

retrieveSDO

Lesen eines archivierten SDOs, siehe ["Operation retrieveSDO"](#)

deleteSDO

Löschen eines archivierten SDOs (nach erreichtem Freigabezeitpunkt), siehe ["Operation deleteSDO"](#)

forceDeleteSDO

Löschen eines archivierten SDOs, auch wenn der Freigabezeitpunkt noch nicht erreicht ist, siehe ["Operation forceDeleteSDO"](#)

statusSDO

Anzeigen des Status für ein archiviertes SDO, siehe ["Operation statusSDO"](#)

listSDOVersions

Auflisten aller mit einer bestimmten COID archivierten SDOs (alle archivierten Versionen eines Dokuments), siehe ["Operation listSDOVersions"](#)

retrieveMetaData

Lesen der Metadaten eines archivierten SDOs, siehe ["Operation retrieveMetaData"](#)

metaDataSDO

Lesen der Metadaten eines archivierten SDOs, die bei der Operation submitSDO bzw. replaceSDO vom zugehörigen Filter ermittelt wurden, siehe ["Operation metaDataSDO"](#)

requestForEvidence

Lesen der Daten für ein SDO, die für eine externe Verifikation benötigt werden, siehe ["Operation requestForEvidence"](#)

sparqlQuery

Ermitteln von AOIDs, die eine bestimmte Metadatenkombination erfüllen, mit einer SPARQL-Anfrage an den TripleStore,

! Achtung! Diese Operation wird z.Zt. nicht unterstützt.

navigate

Navigation entlang der Ablagestruktur, siehe ["Operation navigate"](#)

getAccountingData

Abrechnungsinformationen für den Web-Service ArchivingService zur Verfügung stellen, siehe ["Operation getAccountingData"](#)

moveSDO

Festlegen eines neuen Ablageorts für ein bereits archiviertes SDO, siehe ["Operation moveSDO"](#)

setExpirationDateTimeSDO

Setzen bzw. Ändern des Freigabezeitpunkts, siehe ["Operation setExpirationDateTimeSDO"](#)

forceDeferredSDO

Aufheben des Freigabezeitpunkts für eine AOID, siehe ["Operation forceDeferredSDO"](#)

getSchemaDataSDO

Ausgeben des verwendeten SDOTypes sowie der zugehörigen Schemas und des Filters für AOIDs, siehe ["Operation getSchemaDataSDO"](#)

getVersion

Ausgeben der aktuellen SecDocs-Version, siehe ["Operation getVersion"](#)

Datentyp TSecurity

```
<xs:complexType name="TSecurity">
  <xs:sequence>
    <xs:element name="principal" type="TPrincipal"/>
    <xs:choice>
      <xs:element name="password" type="xs:string"/>
      <xs:element name="token" type="xs:base64Binary"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

principal Basisdaten für die Zugangsprüfung:
Datentyp TPrincipal, siehe ["Datentyp TPrincipal"](#)

password string:
Mandanten-, orgID- und rollenspezifisches Passwort.

Das Passwort für die Rolle Archivar wird initial mit dem Web-Service ArchiveAdminService vergeben. Dieses Passwort kann dann mit dem Web-Service MandantAdminService geändert werden (siehe [Abschnitt „Web-ServiceMandantAdminService“](#)).

token Eine Token-Angabe ist derzeit nicht realisiert. Es muss der Passwort-Mechanismus verwendet werden. Das Token-Verfahren ist in einer Folgeversion zur Autorisierung vorgesehen.

Datentyp TPrincipal

```
<xs:complexType name="TPrincipal">
  <xs:sequence>
    <xs:element name="role"
      minOccurs="1" maxOccurs="1" type="xs:string"/>
    <xs:element name="mandant" type="xs:string" minOccurs="0"/>
    <xs:element name="orgID" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

role string:

Rolle für die Anmeldung am Web-Service.
Die Angabe für `role` ist nicht case-sensitiv.

mandant string:

Mandantenname, für den die Operation ausgeführt werden soll.
Die Angabe für `mandant` ist nicht case-sensitiv.

orgID string:
Name einer Organisationseinheit.

Die organisationsspezifischen Archivbereiche werden mit dem Web-Service MandantAdminService strukturiert. Dabei wird auch die orgID zugewiesen.

Die Angabe für orgID ist nicht case-sensitiv.

i Es können nur für die archivierten Objekte Operationen ausgeführt werden (z.B. Lesen), die auch unter dieser Organisation archiviert wurden.

Datentyp TCoid

```
<xs:simpleType name="TCoid">
  <xs:restriction base="xs:string">
    <xs:minLength value="1"></xs:minLength>
    <xs:maxLength value="256"></xs:maxLength>
  </xs:restriction>
</xs:simpleType>
```

Beispiel für einen Request-Header

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xsi:schemaLocation=
    http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <soapHeaderData xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
      <security>
        <principal>
          <role>Archivar</role>
          <mandant>Mandant1</mandant>
          <orgID>Department1</orgID>
        </principal>
        <password>secrets2</password>
      </security>
      <auditID>Archivar1</auditID>
      <operation>getA0ID</operation>
    </soapHeaderData>
  </soap:Header>
  <soap:Body>
    . . . . .
  </soap:Body>
</soap:Envelope>
```

3.3 Request- und Response-Body

Der Aufbau von SOAP-Request und SOAP-Response ist operationsspezifisch.

- Als Vorlage für die Definition der Operationen und Operanden wird die Datei `Archiving.wsdl` mit ausgeliefert.
- Definition der Typen: `ArchivingData.xsd`

3.3.1 Operation getAOID

getAOID fordert für ein zu archivierendes SDO eine eindeutige AOID an. Ein SDO kann nur mit dieser AOID archiviert oder im Archiv angesprochen werden. getAOID fordert die AOID für die mandantenspezifische orgID an, die im SOAP-Header angegeben wurde (siehe ["Request-Header"](#)).

Soll ein SDO mit einer COID archiviert werden, die für den Mandanten noch nicht verwendet wird, kann die Operation getAOID entfallen und die AOID implizit mit submitSDO erzeugt werden, siehe ["Operation submitSDO"](#).

i Falls Sie externe Datenobjekte archivieren wollen, müssen Sie die Operation getAOIDWithRef verwenden, siehe ["Operation getAOIDWithRef"](#).

Dokumentversionen

Falls Sie ein Dokument in mehreren Versionen archivieren wollen, müssen Sie für jede Version dieses Dokuments bei der Anforderung der AOID mit getAOID eine (und dieselbe) COID im SOAP-Header angeben. Die Angabe dieser COID ist bereits bei getAOID für die erste Version dieses Dokuments erforderlich, da einer AOID und insbesondere einem bereits archivierten Dokument nachträglich keine COID mehr zugeordnet werden kann. Näheres zur Versionierung von Dokumenten finden Sie im [Abschnitt „Versionieren von Dokumenten“](#).

Bei der Anforderung einer AOID für eine neue Version eines Dokuments muss die im SOAP-Request angegebene Organisation mit der von den bereits archivierten Version(en) übereinstimmen.

- i** Die Operationen getAOID und submitSDO sind miteinander gekoppelt:
- Nach einer erfolgreichen Operation getAOID muss submitSDO innerhalb eines begrenzten Zeitfensters erfolgen. Das Zeitfenster wird mit dem Parameter aoidKeepReservedPeriod in der Datei secdocs.properties eingestellt (Standardwert: 720 Minuten, siehe ["Konfigurationsdatei secdocs.properties"](#)).
 - Bei den Operationen müssen jeweils die gleichen Zugangsdaten angegeben werden (Datentyp TPrincipal, siehe ["Request-Header"](#)).

Request-Body

Leeres Element vom Datentyp TGetAoidRequest

```
<tns:getAoidRequest  
  
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Datentyp TGetAoidResponse

```

<xs:complexType name="TGetAoidResponse">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="aoid" type="TUuid"/>
  </xs:sequence>
</xs:complexType>

```

aoid string (Universally Unique Identifier gemäß Standard IETF RFC 4122):
Eindeutige AOID

Beispiel

Request

```

<soap:Header>
  <secdocs:soapHeaderData xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>getAOID</secdocs:operation>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:getAoidRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>

```

Response

```

<soap:Body>
  <tns:getAoidResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</tns:aoid>
  </tns:getAoidResponse>
</soap:Body>

```

3.3.2 Operation `getAIDWithRef`

`getAIDWithRef` fordert für ein zu archivierendes SDO eine eindeutige AID sowie eine oder mehrere externe Referenz-IDs an. Ein SDO kann nur mit dieser AID archiviert oder im Archiv angesprochen werden.

Externe Datenobjekte können nur mit diesen externen Referenz-IDs übertragen, archiviert oder im Archiv angesprochen werden.

i Die Funktion `getAIDWithRef` ist speziell für die Archivierung externer Datenobjekte vorgesehen. Falls Sie kein externes Datenobjekt archivieren wollen, müssen Sie stattdessen die Operation `getAID` einsetzen, siehe "[Operation `getAID`](#)".

`getAIDWithRef` fordert sowohl die AID als auch die externen Referenz-IDs für die mandantenspezifische `orgID` an, die im SOAP-Header angegeben wurde (siehe "[Request-Header](#)").

Jede externe Referenz-ID ist innerhalb des Mandanten eindeutig.

Für jedes zu archivierende externe Datenobjekt müssen Sie mit `getAIDWithRef` eine eigene externe Referenz-ID anfordern. Zu diesem Zweck geben Sie für jedes dieser Datenobjekte bei `getAIDWithRef` einen Hash-Wert oder eine frei definierbare Kurzbeschreibung (oder beides) an.

Den Hash-Wert müssen Sie selbst mit einem geeigneten Hash-Algorithmus erzeugen. Mit Hilfe dieses Hash-Werts kann SecDocs zu einem späteren Zeitpunkt eine Prüfung der Unversehrtheit der übertragenen Daten durchführen.

Wenn ein solcher Hash-Wert für ein externes Datenobjekt nicht erforderlich ist, z.B. weil die Datei eine Signatur enthält, können Sie stattdessen einen frei gewählten Beschreibungstext zur Anforderung einer Referenz-ID angeben.

Eine solchen Beschreibungstext können Sie aber auch zusätzlich zu einem Hash-Wert angeben, um die Zuordnung von externem Datenobjekt zu Referenz-ID zu erleichtern.

Pro AID können Sie mit `getAIDWithRef` maximal 10 externe Referenz-IDs anfordern.

Wenn Sie für ein Datenobjekt einen Hash-Wert angegeben haben, können Sie die dafür von `getAIDWithRef` zurückgelieferte externe Referenz-ID nur genau für dieses Datenobjekt verwenden.

Die Übertragung der externen Datenobjekte müssen Sie mit den zurückgelieferten externen Referenz-IDs durchführen. Außerdem müssen Sie jede dieser externen Referenz-IDs im SDO an einer Stelle eintragen, auf die mit den Systemschlüsseln `$ContentRef` oder `$ExternalRef` verwiesen wird.

Danach müssen Sie `submitSDO` für dieses SDO mit der zurückerhaltenen AID durchführen.

Eine implizite Erzeugung der AID mit `submitSDO` (siehe "[Operation `submitSDO`](#)") ist bei der Archivierung externer Datenobjekte nicht möglich.

Achten Sie auf eine korrekte Zuordnung von externen Datenobjekten, externen Referenz-IDs und Hash-Werten.

Zur Erleichterung dieser Zuordnung gibt `getAIDWithRef` zu jeder externen Referenz-ID auch den zugeordneten Hash-Wert und/oder den Beschreibungstext zurück.

Näheres zur Übertragung externer Objekte finden Sie im [Kapitel „Externe Datenobjekte“](#).

i Um die Operation `getAOIDWithRef` ausführen zu können, müssen Sie den Konfigurationsparameter `createSftpServer` ("[Konfigurationsdatei secdocs.properties](#)") auf den Wert `true` und den Konfigurationsparameter `archiveRootExternalFiles` ("[Konfigurationsdatei secdocs.properties](#)") auf einen gültigen Wert setzen. Andernfalls wird die Operation abgewiesen.

Das Archivieren mehrerer Versionen eines Dokuments wird für SDOs mit externen Referenz-IDs nicht unterstützt. Die Operation `getAOIDWithRef` wird abgewiesen, wenn eine im SOAP-Header angegebene COID für den Mandanten und die Organisation bereits existiert.

Die Operationen `getAOIDWithRef` und `submitSDO` sind miteinander gekoppelt:

- Nach einer erfolgreichen Operation `getAOIDWithRef` müssen die Übertragung der externen Daten in den Übergabebereich und der anschließende `submitSDO` mit den erhaltenen externen Referenz-IDs innerhalb eines begrenzten Zeitfensters erfolgen. Andernfalls werden die externen Referenz-IDs (analog zu den unbenutzten AOIDs) zusammen mit eventuell bereits übertragenen Datenobjekten im mandanten- und organisationsspezifischen Übergabebereich gelöscht.

Das Zeitfenster wird mit dem Parameter `aoidWithRefKeepReservedPeriod` in der Datei `secdocs.properties` eingestellt (Standardwert: 2880 Minuten = 2 Tage, Minimum 120 Minuten, siehe "[Konfigurationsdatei secdocs.properties](#)").

- Bei den Operationen müssen jeweils die gleichen Zugangsdaten angegeben werden (Datentyp `TPrincipal`, siehe "[Request-Header](#)").

Request-Body

Element `getAoidWithRefRequest` vom Datentyp `TGetAoidWithRefRequest`

```
<xs:complexType name="TGetAoidWithRefRequest">
  <xs:sequence>
    <xs:element name="externalObjectInData"
                 type="TExternalObjectInData"
                 minOccurs="1"
                 maxOccurs="unbounded">

    </xs:element>
  </xs:sequence>
</xs:complexType>
```

`externalObjectInData`

Angaben zur Anforderung einer externen Referenz-ID für ein externes Dokument. Datentyp `TExternalObjectInData`, siehe unten.

Element `externalObjectInData` vom Datentyp `TExternalObjectInData`

```

<xs:complexType name="TExternalObjectInData">
  <xs:sequence>
    <xs:element name="hashValue" type="xs:base64Binary"
      minOccurs="0"
      maxOccurs="1"/>
    <xs:element name="hashAlgorithmName" type="xs:string"
      minOccurs="0"
      maxOccurs="1"/>
    <xs:element name="descriptionText" type="xs:string"
      minOccurs="0"
      maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>

```

hashValue

base64Binary:

optional; Berechneter Hash-Wert für die externe Datei.

Mit Hilfe dieses Hash-Werts wird SecDocs bei submitSDO eine Prüfung der Unversehrtheit der übertragenen Daten durchgeführt. Zur Ermittlung des korrekten Hash-Wertes, wie er in SecDocs verwendet wird, können Sie im Betriebssystem Linux das Shell-Skript `mksha` verwenden (siehe [Abschnitt „Hash-Wert erzeugen \(Skript mksha\)“](#)).

Wenn Sie `hashValue` angeben, müssen Sie zwingend auch den Operanden `hashAlgorithmName` angeben.

Wenn Sie `hashValue` nicht angeben, müssen Sie zwingend den Operanden `descriptionText` angeben.

hashAlgorithmName

string:

optional; Algorithmus, der verwendet wurde, um den Hash-Wert für die externe Datei zu bilden.

Wenn Sie `hashAlgorithmName` angeben, müssen Sie zwingend auch den Operanden `hashValue` angeben.

i Der angegebene Hash-Algorithmus muss verfügbar sein (siehe [Abschnitt „Operation getHashAlgorithms“](#)).

descriptionText

string:

optional; Frei definierbarer Text, der von SecDocs nicht ausgewertet wird.

Der Text darf maximal 1000 Zeichen enthalten, andernfalls wird die Operation `getAOIDWithRef` abgewiesen.

Wenn Sie `descriptionText` nicht angeben, müssen Sie zwingend die Operanden `hashValue` und `hashAlgorithmName` angeben.

Response-Body

Element `getAoidWithRefResponse` vom *Datentyp* `TGetAoidWithRefResponse`

```
<xs:complexType name="TGetAoidWithRefResponse">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="aoid" type="TUuid" minOccurs="1"
               maxOccurs="1"/>
    <xs:element name="externalObjectOutData"
               type="TExternalObjectOutData"
               minOccurs="1"
               maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

`aoid`

string (Universally Unique Identifier gemäß Standard IETF RFC 4122):
Eindeutige AOID

`externalObjectOutData`

Ausgabedaten für eine externe Datei
Datentyp `TExternalObjectOutData`, siehe unten.

Element `externalObjectOutData` vom *Datentyp* `TExternalObjectOutData`

```
<xs:complexType name="TExternalObjectOutData">
  <xs:sequence>
    <xs:element name="externalReferenceID" type="xs:string"
               minOccurs="1"
               maxOccurs="1"/>
    <xs:element name="hashValue" type="xs:base64Binary"
               minOccurs="0"
               maxOccurs="1"/>
    <xs:element name="descriptionText" type="xs:string"
               minOccurs="0"
               maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

`externalReferenceID`

string:
Externe Referenz-ID
Verwenden Sie an allen Stellen, wo Sie diese externe Referenz-ID angeben müssen, genau den von `getAOIDWithRef` zurückgelieferten Wert.

i Der Aufbau der Referenz-ID ist keine garantierte Schnittstelle.

hashValue

base64Binary:

optional; Hash-Wert derjenigen Datei, für die die externe Referenz-ID zurückgeliefert wird.

Der Hash-Wert wird sofern vorhanden aus dem `getAOIDWithRef`-Request in die Antwortnachricht übernommen.

descriptionText

string:

optional; Beschreibungstext derjenigen Datei, für die die externe Referenz-ID zurückgeliefert wird.

Der Beschreibungstext wird sofern vorhanden aus dem `getAOIDWithRef`-Request in die Antwortnachricht übernommen.

Beispiel:

Request-Body

```
<soap:Body>

  <tns:getAoidWithRefRequest

    xmlns:tns="http://ts.fujitsu.com/secdocs/v2_3/archiving">
    <tns:externalObjectInData>

      <tns:hashValue>0528a5f5e...9a3aacbe6cc3</tns:hashValue>

      <tns:hashAlgorithmName>SHA-256</tns:hashAlgorithmName>
      <tns:descriptionText>Picture1</tns:descriptionText>

    </tns:externalObjectInData>

    <tns:externalObjectInData>
      <tns:hashValue>aec2ec728...8388e97bdb20</tns:hashValue>

      <tns:hashAlgorithmName>SHA-256</tns:hashAlgorithmName>

      <tns:descriptionText>Legend1</tns:descriptionText>
    </tns:externalObjectInData>

    <tns:externalObjectInData>
      <tns:hashValue>a625e94cf...b5d24ed3a6ac</tns:hashValue>
      <tns:hashAlgorithmName>SHA-256</tns:hashAlgorithmName>

    </tns:externalObjectInData>

    <tns:externalObjectInData>
      <tns:descriptionText>SignedDocument1</tns:descriptionText>
    </tns:externalObjectInData>

  </tns:getAoidWithRefRequest>

</soap:Body>
```

Response-Body

```
<soap:Body>

  <tns:getAoidWithRefResponse

    xmlns:tns="http://ts.fujitsu.com/secdocs/v2_3/archiving">
    <tns:aoid>26f24b12-f126-4114-a088-de73644c6084</tns:aoid>

    <tns:externalObjectOutData>

      <tns:externalReferenceID>_26f24b12-f126-4114-a088-de73644c6084/1
      </tns:externalReferenceID>

      <tns:hashValue>0528a5f5e...9a3aacbe6cc3</tns:hashValue>

      <tns:descriptionText>Picture1</tns:descriptionText>
    </tns:externalObjectOutData>

    <tns:externalObjectOutData>

      <tns:externalReferenceID>_26f24b12-f126-4114-a088-de73644c6084/2
      </tns:externalReferenceID>

      <tns:hashValue>aec2ec728...8388e97bdb20</tns:hashValue>

      <tns:descriptionText>Legend1</tns:descriptionText>
    </tns:externalObjectOutData>

    <tns:externalObjectOutData>

      <tns:externalReferenceID>_26f24b12-f126-4114-a088-de73644c6084/3
      </tns:externalReferenceID>

      <tns:hashValue>a625e94cf...b5d24ed3a6ac</tns:hashValue>
    </tns:externalObjectOutData>

    <tns:externalObjectOutData>
      <tns:externalReferenceID>_26f24b12-f126-4114-a088-de73644c6084/4
      </tns:externalReferenceID>

      <tns:descriptionText>SignedDocument1</tns:descriptionText>
    </tns:externalObjectOutData>

  </tns:getAoidWithRefResponse>
</soap:Body>
```

3.3.3 Operation submitSDO

submitSDO archiviert ein SDO. Das SDO wird unter einer archivweit eindeutigen AOID archiviert.

Die AOID kann auf zwei Arten erzeugt werden:

- Explizite Erzeugung:
Hierbei sind wiederum zwei Fälle zu unterscheiden:
 - Falls das SDO keine Referenzen auf externe Datenobjekte (externe Referenz-IDs) enthält, fordert die Client-Anwendung explizit mit der Operation `getAOID` eine AOID an (siehe [Abschnitt „Operation getAOID“](#)).
 - Andernfalls fordert die Client-Anwendung mit der Operation `getAOIDWithRef` eine AOID sowie eine oder mehrere externe Referenz-IDs an (siehe [Abschnitt „Operation getAOIDWithRef“](#)).

Die AOID wird dann im SOAP-Header der Operation `submitSDO` angegeben (siehe "Datentyp TSoapHeader" im Abschnitt ["Request-Header"](#)).

Sie müssen die externen Referenz-IDs, falls vorhanden, an einer Stelle im SDO eintragen, auf die mit dem Systemschlüssel `$ContentRef` oder `$ExternalRef` verwiesen wird. Die Operation `submitSDO` muss dabei innerhalb eines begrenzten Zeitfensters nach der Operation `getAOID` bzw. `getAOIDWithRef` erfolgen. `submitSDO` kann nur mit derselben `orgID` durchgeführt werden, mit der die AOID angefordert wurde.

- Implizite Erzeugung:
Die implizite Erzeugung ist nur möglich, wenn die folgende Bedingungen erfüllt sind:
 - Es ist keine AOID angegeben, aber eine COID, die für den Mandanten noch nicht verwendet wird.
 - Das SDO enthält keine Referenzen auf externe Datenobjekte (externe Referenz-IDs)Der Aufruf der Operation `getAOID` entfällt. Geben Sie statt einer AOID im SOAP-Header der Operation `submitSDO` eine COID an. SecDocs erzeugt zu der COID eine AOID und liefert diese im SOAP-Body der `submitSDO`-Antwortnachricht zurück.

Voraussetzungen

- Die Struktur des SDOs und die zugehörige Filterdefinition müssen registriert sein, Operation `createSDOType` (siehe Abschnitt ["Operation createSDOType"](#)).
- Falls das SDO externe Referenz-IDs enthält, gilt zusätzlich:
 - Die Übertragung aller zu dieser AOID gehörenden externen Datenobjekte vom lokalen Rechner der Client-Anwendung auf den SecDocs-Rechner muss vollständig abgeschlossen sein. Näheres hierzu siehe Abschnitt ["Externe Datenobjekte mit SFTP übertragen"](#) in ["Archivieren eines externen Datenobjekts"](#).
 - Die bei der Operation `getAOIDWithRef` erhaltenen externen Referenz-IDs müssen ins SDO eingetragen sein.
 - Die Datenobjekte, auf die die externen Referenz-IDs verweisen, dürfen nicht leer sein.

Anzahl der archivierten SDOs

Die Anzahl der SDOs, die unter einem Knoten archiviert werden können, ist durch die Anzahl der möglichen Unterverzeichnisse unter einem Knoten limitiert. Diese Grenze ist abhängig vom verwendeten Storage- bzw. Dateisystem.

Wählen Sie z.B. als Unterstruktur `/$Year/$Month/$Day`, dann kann jeden Tag diese Zahl an Dokumenten gespeichert werden.

Unterstrukturen werden mit dem Systemschlüssel `$SDOPath` (siehe ["\\$SDOPath"](#)) definiert.

Request-Body

XML-Dokument:

```
<myXMLDocument>
...
</myXMLDocument>
```

myXMLDocument

Zu archivierendes XML-Dokument. Der zugehörige SDO-Typ muss vor dem Aufruf der Operation `submitSDO` definiert worden sein (Operation `createSDOType`, siehe "[Operation createSDOType](#)").

i Der Body-Inhalt der Request-Message der Operation `submitSDO` wird von der ersten öffnenden spitzen Klammer (<) bis zur letzten schließenden Klammer (>) beweiswerterhaltend archiviert und kann mit der Operation `retrieveSDO 1:1` wieder eingelesen werden. Das heißt, ist das zu archivierende XML-Dokument (SDO) von White Spaces umgeben, so werden diese nicht archiviert. SecDocs ändert nicht das übergebene XML-Dokument, d.h. es wird keine XML Kanonisierung durchgeführt.

Der SOAP-Request, mit dem ein SDO archiviert wird, ist ca. 1,5-mal so groß wie das ursprüngliche Primärdokument, bedingt durch die Umwandlung in Base64-Kodierung. Die maximale Größe eines SOAP-Requests, die SecDocs verarbeiten kann, wird mit dem Parameter `maxSoapRequestSize` in der Datei `secdocs.properties` eingestellt (siehe "[Konfigurationsdatei secdocs.properties](#)").

Response-Body

Datentyp `TSubmitSdoResponse`

```
<xs:complexType name="TSubmitSdoResponse">
  <xs:sequence>
    <xs:element name="status" type="TSubmitSdoResponseStatus"/>
    <xs:element name="aoid" type="TUuid" maxOccurs="1"
                                     minOccurs="0" />
  </xs:sequence>
</xs:complexType>
```

`TSubmitSdoResponseStatus`

Status nach der Operation `submitSDO`:

Datentyp `TSubmitSdoResponseStatus`, siehe unten.

`aoid`

string (Universally Unique Identifier gemäß Standard IETF RFC 4122):
optional; Eindeutige AOID.

Datentyp TSubmitSdoResponseStatus

```
<xs:simpleType name="TSubmitSdoResponseStatus">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SDO submitted successfully"/>
    <xs:enumeration value="SDO already submitted"/>
    <xs:enumeration value="SDO already submitted and sealed"/>
  </xs:restriction>
</xs:simpleType>
```

SDO submitted successfully

Die Operation wurde erfolgreich beendet und das SDO gespeichert.

SDO already submitted

Ein SDO mit dieser AOID oder COID wurde bereits gespeichert. Auf dem Server erfolgte keine weitere Aktion.

SDO already submitted and sealed

Ein SDO mit dieser AOID oder COID wurde bereits gespeichert und versiegelt. Auf dem Server erfolgte keine weitere Aktion.

i Die Operation `submitSDO` wird abgewiesen, wenn die Verifikation einer oder mehrerer Signaturen fehlschlägt. In diesem Fall wird in der Fault-Message zusätzlich zu den SecDocs-spezifischen Fehlerinformationen die Fehlerinformation der OpenLimit-Komponenten ausgegeben, die u.a. die Verifikationsprotokolle aller geprüften Signaturen enthält (siehe [Abschnitt „Fault-Message“](#)). Es wird empfohlen, diese Verifikationsprotokolle für eine nachfolgende Analyse als Dateien abzuspeichern.

i Wurde eine Operation `submitSDO` ohne Fehler bearbeitet, d.h. enthält die Antwortnachricht keine Fault-Message, sollten Sie den Status der Operation in der Antwortnachricht (siehe ["TSubmitSdoResponseStatus"](#)) auswerten, um zu prüfen, ob das SDO tatsächlich von SecDocs gespeichert wurde.

Beispiel

Request

```

<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>submitSDO</secdocs:operation>
    <secdocs:aid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aid>
    <secdocs:SDOType>DocumentNone</secdocs:SDOType>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
<tns:myDocument
  xmlns:tns="http://ts.fujitsu.com/secdocs/sdosamples/myDocument"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <tns:id>AKZ-08/15</tns:id>
  <tns:retentionPeriod>P5Y</tns:retentionPeriod>
  <tns:metaData>
    <tns:author>Author1</tns:author>
    <tns:department>Department1</tns:department>
    <tns:company>MyCompany</tns:company>
    <tns:copyright>MyCompany, 2012</tns:copyright>
    <tns:description>PDF file without any type of
      signature</tns:description>
    <tns:keywords>pdf</tns:keywords>
    <tns:creationDate>2012-07-23</tns:creationDate>
    <tns:lastChangedDate>2012-07-23</tns:lastChangedDate>
  </tns:metaData>
  <tns:file>
    <tns:name>testdoc0001.pdf</tns:name>
    <tns:type>PDF</tns:type>
    <tns:content>JVBERi0xLjQgE5pQG5tJMyAwIG9iai...</tns:content>
  </tns:file>
</tns:myDocument>
</soap:Body>

```

Response

```

<soap:Body>
  <tns:submitSdoResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:status>SDO submitted successfully</tns:status>
  </tns:submitSdoResponse>
</soap:Body>

```


3.3.4 Operation replaceSDO

`replaceSDO` ersetzt ein bereits archiviertes SDO. Das SDO wird durch die Angabe der AOID oder COID im SOAP-Header identifiziert, unter der das SDO archiviert wurde.

Wenn Sie nur eine COID, aber keine AOID angeben und mehrere Versionen des Dokuments mit dieser COID existieren, wird auf das SDO mit der höchsten Version zugegriffen. Für den Zugriff auf andere Versionen des Dokuments muss die AOID angegeben werden. Diese kann ggf. mit der Operation `listSDOVersions` ermittelt werden, siehe "[Operation listSDOVersions](#)".

Bei der Durchführung der Operation `replaceSDO` werden folgende Aktionen ausgeführt:

- Löschen aller physikalischen Daten des zu ersetzenden SDOs im Speicher
- Löschen aller Einträge des zu ersetzenden SDOs aus dem TripleStore

i Die Anbindung an einen Triple-Store wird in der aktuellen Version nicht unterstützt.

- Archivieren des aktuellen SDOs. Dabei werden die Metadaten und Einträge im TripleStore aus dem aktuellen SDO ermittelt. Insbesondere legt SecDocs das neue SDO unter dem aktuellen SDO-Pfad ab. Falls sich dieser vom SDO-Pfad des zu ersetzenden SDOs unterscheidet, löscht SecDocs neben den alten Daten auch das nicht mehr benötigte alte Verzeichnis.

i Falls während der Verarbeitung der Operation `replaceSDO` ein Fehler auftritt, werden alle bis zu diesem Zeitpunkt durchgeführten Aktionen rückgängig gemacht und der ursprüngliche Zustand auf dem Storage und im TripleStore wiederhergestellt.

Die Versionsnummer eines SDOs ändert sich durch das Ersetzen nicht. Insbesondere ist ein SDO eines versionierten Dokuments nach dem Ersetzen nur dann die aktuelle Version, wenn das ersetzte SDO schon die aktuelle Version war.

i Die Operation `replaceSDO` wird mit Fehler `REQ0059` abgewiesen, wenn für die angegebene AOID gleichzeitig ein Versiegelungsvorgang durchgeführt wird.

Voraussetzungen

- SDO-Typ:
 - Der SDO-Typ des zu ersetzenden SDOs muss das Überschreiben erlauben.
 - Der SDO-Typ kann beim Ersetzen geändert werden.
 - Es ist nicht zwingend erforderlich, dass der neue SDO-Typ ebenfalls das Überschreiben erlaubt. Falls er das Überschreiben jedoch nicht erlaubt, kann diese Eigenschaft für das entsprechende SDO nach dem Ersetzen nicht mehr geändert werden.
- AOID:
 - Die angegebene (oder aus der COID ermittelte) AOID muss sich auf ein archiviertes SDO beziehen, d.h.:
 - Es muss bereits eine Operation `submitSDO` mit dieser AOID ausgeführt worden sein.
 - Sie darf nicht gelöscht sein.

- Freigabezeitpunkt (des zu ersetzenden SDOs):
 - Der Freigabezeitpunkt muss bereits überschritten sein.
 - Er darf nicht auf `DEFERRED` gesetzt sein.
- Organisation:
Die Organisation beim Aufruf von `replaceSDO` muss mit der des zu ersetzenden SDOs übereinstimmen.
- SoftWORM-Funktionalität:
Das SDO darf nicht mit `$Softworm=YES` archiviert sein.

i SDOs, die auf einem Storage-System mit SoftWORM-Funktionalität abgelegt sind, können grundsätzlich nicht mit `replaceSDO` ersetzt werden.

- Externe Referenz-IDs:
Das SDO darf keine externe Referenz-ID enthalten.

i SDOs, die externe Referenz-IDs enthalten, können grundsätzlich nicht mit `replaceSDO` ersetzt werden.

Idempotenz des Archivierungsvorgangs

Unter der Idempotenz des Archivierungsvorgangs versteht man Folgendes: Die Operation `submitSDO` wird beim wiederholten Aufruf mit den gleichen Daten im Header ggf. erfolgreich zu Ende geführt. Dabei werden jedoch Daten, die bereits auf dem Storage liegen, nicht verändert.

Im Gegensatz zu `submitSDO` kann diese Idempotenz des Archivierungsvorgangs bei `replaceSDO` nicht gewährleistet werden.

Wenn nach dem Aufruf der Operation `replaceSDO` ein weiterer Aufruf mit denselben Daten eintrifft, sind folgende Fälle zu unterscheiden:

1. Die vorangehende Operation ist bereits abgeschlossen:
Es wird eine erneute Ersetzung des SDOs angestoßen, da sich der Aufruf nicht von einem regulären, erneuten Aufruf unterscheidet. Die erneute Ersetzung kann jedoch insbesondere dann zu einem Fehler führen, wenn bei der vorangegangenen Ersetzung ein noch nicht erreichtes Ablaufdatum festgelegt wurde.
2. Die vorangehende Operation läuft noch:
Der nachfolgende Aufruf wartet, bis die vorangehende Operation abgeschlossen ist. Danach ist das Verhalten wie im Fall 1.

Request-Body

XML-Dokument:

```
<myXMLDocument>
...
</myXMLDocument>
```

myXMLDocument

Der Body der SOAP-Nachricht besteht – wie bei der Operation `submitSDO` – aus dem neu zu archivierenden XML-Dokument des Anwenders entsprechend dem Schema, das mit dem dazugehörigen SDO-Typ definiert wurde.

Response-Body

Element `replaceSdoResponse` vom *Datentyp* `TReplaceSdoResponse`

```
<xs:complexType name="TReplaceSdoResponse">
  <xs:annotation>
    <xs:documentation>SecDocs SOAP body response data for WS
      operations of type replaceSDO
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="status" type="TReplaceSdoResponseStatus"
      minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

status

Status nach der Operation `replaceSDO`:
Datentyp `TReplaceSdoResponseStatus`, siehe unten.

Datentyp `TReplaceSdoResponseStatus`

```
<xs:simpleType name="TReplaceSdoResponseStatus">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SDO replaced successfully"/>
  </xs:restriction>
</xs:simpleType>
```

SDO replaced successfully

Das SDO wurde erfolgreich ersetzt.

Beispiel

Response-Body

```
<soap:Body>
  <tns:replaceSdoResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:status>SDO replaced successfully</tns:status>
  </tns:replaceSdoResponse>
</soap:Body>
```

3.3.5 Operation retrieveSDO

`retrieveSDO` liest ein archiviertes SDO. Das SDO wird durch die Angabe der AOID oder COID im SOAP-Header identifiziert, unter der das SDO archiviert wurde.

Wenn nur eine COID, aber keine AOID angegeben ist und mehrere Versionen des Dokuments mit dieser COID existieren, wird auf das SDO mit der höchsten Version zugegriffen. Für den Zugriff auf andere Versionen des Dokuments muss die AOID angegeben werden. Diese können Sie ggf. mit der Operation `listSDOVersions` ermitteln, siehe "[Operation listSDOVersions](#)".

Voraussetzungen

Das SDO wurde unter der gleichen Organisation mit der Operation `submitSDO` archiviert (siehe "[Operation submitSDO](#)").

i Im SOAP-Header der `retrieveSDO`-Response wird der SDO-Typ zurückgeliefert, mit dem das Dokument abgespeichert wurde. SecDocs überprüft jedoch nicht, ob dieser mit dem SDO-Typ im Aufruf übereinstimmt.

SDOs mit Referenzen auf externe Datenobjekte

Falls das SDO Referenzen auf externe Datenobjekte (externe Referenz-IDs) enthält, stellt `retrieveSDO` auch die zu dem SDO gehörenden externen Datenobjekte für die Übertragung auf den lokalen Rechner der Client-Anwendung bereit. Diese Übertragung muss die Client-Anwendung nach erfolgreichem `retrieveSDO` innerhalb der Zeitspanne durchführen, die mit dem Konfigurationsparameter `externalFilesRetrieveTimeout` ("[Konfigurationsdatei secdocs.properties](#)") festgelegt ist. Näheres hierzu siehe unter „[Adressieren von Daten im SDO](#)“ (in "[Archivieren eines externen Datenobjekts](#)") und „[Lesen eines archivierten externen Datenobjekts](#)“

i Beachten Sie:
SecDocs richtet zum Lesen der externen Datenobjekte symbolische Links ein. Die Anzahl der symbolischen Links, die eingerichtet werden können, ist durch die Anzahl der möglichen Unterverzeichnisse im mandanten- und organisationsspezifischen Übergabebereich begrenzt. Diese Grenze ist wiederum abhängig vom verwendeten Storage- und/oder Dateisystem.

Wenn Sie für einen Mandanten sehr viele archivierte externe Datenobjekte lesen wollen, haben Sie folgende Möglichkeiten, um einen diesbezüglichen Ressourcenengpass zu vermeiden:

- Verringern Sie für den betroffenen Mandanten die Zeitspanne, nach der ein symbolischer Link von SecDocs automatisch gelöscht wird (siehe Konfigurationsparameter `externalFilesRetrieveTimeout` ("[Konfigurationsdatei secdocs.properties](#)") und [Abschnitt „Mandantenspezifische Konfigurationsparameter“](#)),
- Löschen Sie nach der erfolgreichen Übertragung eines externen Datenobjekts auf den lokalen Rechner den symbolischen Link mit dem `sftp`-Kommando `rm` (siehe [Abschnitt „Kommandos für den SFTP-Server“](#)). Voraussetzung hierfür ist, dass Sie das `sftp`-Kommando `get` nicht wiederholen möchten und keine weitere Anwendung gleichzeitig auf dieses externe Datenobjekt lesend zugreift.

Request-Body

Datentyp `TRetrieveSDORequest`

```
<xsd:complexType name="TRetrieveSDORequest">
  <xsd:attribute name="checkHash" type="xs:boolean" use="optional" />
</xsd:complexType>
```

checkHash

Legt fest, ob SecDocs die Integrität des SDOs und der zugehörigen Signature Verification Information automatisch im Rahmen dieser Operation überprüfen soll. Wenn Sie diesen Parameter nicht angeben, gilt die mit dem Konfigurationsparameter `checkHashAtRetrieve` in der Datei `secdocs.properties` (["Konfigurationsdatei secdocs.properties"](#)) vorgenommene Voreinstellung.

Response-Body

XML-Dokument:

```
<myXMLDocument>
...
</myXMLDocument>
```

myXMLDocument

Archiviertes XML-Dokument.

i *myXMLDocument* ist genau das XML-Dokument, das in der Operation `submitSDO` oder `replaceSDO` archiviert worden ist. `retrieveSDO` liefert 1:1 den Body aus der Request-Message der Operation `submitSDO` oder `replaceSDO`.

Beispiel

Request

```

<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>retrieveSDO</secdocs:operation>
    <secdocs:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aoid>
    <secdocs:SDOType>DocumentNone</secdocs:SDOType>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:retrieveSdoRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"
    checkHash="true"/>
</soap:Body>

```

Response

```

<soap:Body>
<tns:myDocument
  xmlns:tns="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <tns:id>AKZ-08/15</tns:id>
  <tns:retentionPeriod>P5Y</tns:retentionPeriod>
  <tns:metaData>
    <tns:author>Author1</tns:author>
    <tns:department>Department1</tns:department>
    <tns:company>MyCompany</tns:company>
    <tns:copyright>MyCompany, 2012</tns:copyright>
    <tns:description>PDF file without any type of
signature</tns:description>
    <tns:keywords>pdf</tns:keywords>
    <tns:creationDate>2012-07-23</tns:creationDate>
    <tns:lastChangedDate>2012-07-23</tns:lastChangedDate>
  </tns:metaData>
  <tns:file>
    <tns:name>testdoc0001.pdf</tns:name>
    <tns:type>PDF</tns:type>
    <tns:content>JVBERi0xLjQgE5pQG5tJMyAwIG9...</tns:content>
  </tns:file>
</tns:myDocument>
</soap:Body>

```

3.3.6 Operation deleteSDO

`deleteSDO` löscht ein unter der Organisation archiviertes SDO. Das SDO wird durch die Angabe der AOID oder COID im SOAP-Header identifiziert.

Wenn nur eine COID, aber keine AOID angegeben ist und mehrere Versionen des Dokuments mit dieser COID existieren, wird das SDO mit der höchsten Version gelöscht. Soll eine andere Version des Dokuments gelöscht werden, muss die AOID angegeben werden. Diese können Sie ggf. mit der Operation `listSDOVersions` ermitteln, siehe "[Operation listSDOVersions](#)".

Falls das SDO Referenzen auf externe Datenobjekte (externe Referenz-IDs) enthält, löscht `deleteSDO` auch alle mit diesem SDO verknüpften externen Datenobjekte im mandanten- und organisationsspezifischen Übergabebereich und die symbolischen Links.



Beachten Sie:

Nach dem Löschen eines SDOs mit COID ist die zugeordnete COID in SecDocs nicht mehr bekannt, falls das gelöschte SDO die einzige oder letzte existierende Version mit dieser COID war. Im Gegensatz dazu ist die AOID eines gelöschten SDOs in SecDocs weiterhin bekannt, aber als gelöscht gekennzeichnet. Wenn eine gelöschte COID bei `getAOID` oder `submitSDO` angegeben wird, werden eine neue AOID und die Versionsnummer 0 zurückgeliefert. Bei anderen Operationen wird eine Fehlermeldung ausgegeben. Falls das SDO externe Referenz-IDs enthält und sich die Ausführung der Operationen `deleteSDO` und `retrieveSDO` für dasselbe SDO zeitlich überschneidet, kann folgende Situation eintreten: Ein nach dem `retrieveSDO`-Request ausgeführtes `sftp`-Kommando `get` findet keine externen Referenz-IDs und meldet daher einen Fehler, obwohl der `retrieveSDO`-Request noch ein gültiges Ergebnis geliefert hat.



Die Operation `deleteSDO` wird mit Fehler REQ0059 abgewiesen, wenn für die angegebene AOID gleichzeitig ein Versiegelungsvorgang durchgeführt wird.

Voraussetzungen

- Der Freigabezeitpunkt darf nicht auf `DEFERRED` gesetzt sein.
- Der Freigabezeitpunkt ist erreicht, siehe [Abschnitt „\\$ExpirationDate“](#).

Wird diese Operation für einen Testmandanten ausgeführt, gelten diese Voraussetzungen nicht. SDOs, die einem Testmandanten zugeordnet sind, können unabhängig vom Freigabezeitpunkt mit `deleteSDO` gelöscht werden.

Request-Body

Leeres Element vom Datentyp `TDeleteSdoRequest`

```
<tns:deleteSdoRequest  
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Leeres Element vom Datentyp `TDeleteSdoResponse`

```
<tns:deleteSdoResponse
```

```
xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Beispiel

Request

```
<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>deleteSDO</secdocs:operation>
    <secdocs:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aoid>
    <secdocs:SDOType>DocumentNone</secdocs:SDOType>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:deleteSdoRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>
```

Response

```
<soap:Body>
  <tns:deleteSdoResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>
```


3.3.7 Operation forceDeleteSDO

`forceDeleteSDO` löscht ein unter der Organisation archiviertes SDO. Das SDO wird durch die Angabe der AOID oder COID im SOAP-Header identifiziert.

Wenn nur eine COID, aber keine AOID angegeben ist und mehrere Versionen des Dokuments mit dieser COID existieren, wird das SDO mit der höchsten Version gelöscht. Soll eine andere Version des Dokuments gelöscht werden, muss die AOID angegeben werden. Diese können Sie ggf. mit der Operation `listSDVersions` ermitteln, siehe "[Operation listSDVersions](#)".

Falls das SDO Referenzen auf externe Datenobjekte (externe Referenz-IDs) enthält, löscht `forcedeleteSDO` auch alle mit diesem SDO verknüpften externen Datenobjekte im mandanten- und organisationsspezifischen Übergabebereich und die symbolischen Links.

Im Unterschied zur Operation `deleteSDO` können Sie das SDO auch dann löschen, wenn der Freigabezeitpunkt noch nicht erreicht ist, sofern Sie einen Grund für das vorzeitige Löschen angeben (Element `reason`).



Beachten Sie:

Nach dem Löschen eines SDOs mit COID ist die zugeordnete COID in SecDocs nicht mehr bekannt, falls das gelöschte SDO die einzige oder letzte existierende Version mit dieser COID war. Im Gegensatz dazu ist die AOID eines gelöschten SDOs in SecDocs weiterhin bekannt, aber als gelöscht gekennzeichnet. Wenn eine gelöschte COID bei `getAOID` oder `submitSDO` angegeben wird, werden eine neue AOID und die Versionsnummer 0 zurückgeliefert. Bei anderen Operationen wird eine Fehlermeldung ausgegeben. Falls das SDO externe Referenz-IDs enthält und sich die Ausführung der Operationen `forcedeleteSDO` und `retrieveSDO` für dasselbe SDO zeitlich überschneidet, kann folgende Situation eintreten: Ein nach dem `retrieveSDO`-Request ausgeführtes `sftp`-Kommando `get` findet keine externen Referenz-IDs und meldet daher einen Fehler, obwohl der `retrieveSDO`-Request noch ein gültiges Ergebnis geliefert hat.



Die Operation `forceDeleteSDO` wird mit Fehler `REQ0059` abgewiesen, wenn für die angegebene AOID gleichzeitig ein Versiegelungsvorgang durchgeführt wird.

Voraussetzungen

- Das SDO wurde mit der Operation `submitSDO` archiviert (siehe "[Operation submitSDO](#)").
- Es wird eine Begründung für den Löschvorgang angegeben.
Ohne Begründung wird die Operation abgewiesen, auch wenn der Freigabezeitpunkt bereits erreicht ist.

Request-Body

Datentyp `TForceDeleteSdoRequest`

```
<xs:complexType name="TForceDeleteSdoRequest">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="reason" type="TNonEmptyString"/>
  </xs:sequence>
</xs:complexType>
```

reason

NonEmptyString:
Begründung, warum das SDO gelöscht werden soll.
reason darf nicht leer sein.

Response-Body

Leeres Element vom Datentyp TForceDeleteSdoResponse

```
<tns:forceDeleteSdoResponse>
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Beispiel

Request

```
<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>forceDeleteSDO</secdocs:operation>
    <secdocs:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aoid>
    <secdocs:SDOType>DocumentNone</secdocs:SDOType>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:forceDeleteSdoRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:reason>The document had to be deleted because the proceedings
      were closed</tns:reason>
  </tns:forceDeleteSdoRequest>
</soap:Body>
```

Response

```
<soap:Body>
  <tns:forceDeleteSdoResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>
```

3.3.8 Operation statusSDO

statusSDO fragt den aktuellen Status und weitere Informationen zu einem SDO ab, sobald eine AOID für die Archivierung eines SDOs für die Organisation reserviert wurde. Im SOAP-Header muss dabei die AOID oder die COID angegeben werden.

Falls eine COID vergeben wurde, können Sie mit dieser Operation die einer AOID entsprechende COID ermitteln und umgekehrt.

Wenn nur eine COID, aber keine AOID angegeben ist und mehrere Versionen des Dokuments mit dieser COID existieren, wird die Information für das SDO mit der höchsten Version geliefert. Um die Information für andere Versionen des Dokuments zu erhalten, muss die AOID angegeben werden. Diese können Sie ggf. mit der Operation listSDOVersions ermitteln, siehe "[Operation listSDOVersions](#)".

Request-Body

Leeres Element vom Datentyp TStatusSDORequest

```
<tns:statusSdoRequest  
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Element StatusSDOResponse *vom Datentyp* TStatusSDOResponse

```
<xs:complexType name="TStatusSDOResponse">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="aoid" type="TUuid"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="coid" type="TCoid"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="version" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="status" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="mandant" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="organisation" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="sdoPath" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="sdoType" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="sdoTypeReplaceable" type="xs:boolean"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="externalReferences"
      type="TStatusSdoExternalReferences"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="sdoSize" type="xs:long"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="shortSubject" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="lastModificationDateTime"
      type="xs:dateTime"
      minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

```

<xs:choice minOccurs="0" maxOccurs="1">
  <xs:element name="expirationDateTime"
    type="xs:dateTime"
    minOccurs="1" maxOccurs="1"/>
  <xs:element name="expirationDateUnlimited"
    type="TEmptyElement"
    minOccurs="1" maxOccurs="1"/>
  <xs:element name="expirationDateDeferred"
    type="TEmptyElement"
    minOccurs="1" maxOccurs="1"/>
</xs:choice>
<xs:element name="archiveDateTime" type="xs:dateTime"
  minOccurs="0" maxOccurs="1"/>
<xs:element name="versionDateTime" type="xs:dateTime"
  minOccurs="1" maxOccurs="1"/>
<xs:element name="versionNumber" type="xs:long"
  minOccurs="1" maxOccurs="1"/>
<xs:element name="sealedDateTime" type="xs:dateTime"
  minOccurs="0" maxOccurs="1"/>
<xs:element name="tsp" type="xs:string"
  minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

```

aid string (Universally Unique Identifier gemäß Standard IETF RFC 4122):
Eindeutige AOID

coid TCoid:
COID (Client Object Identifier), vom Anwender vergebener Bezeichner zur Identifizierung eines Archivobjekts.
Datentyp TCoid, siehe ["Request-Header"](#)

version string:
SecDocs-Version, mit der die AOID erzeugt wurde.

In der Version 3.0A00 liefert SecDocs hier z.B. den Wert „3.0.1.0“. Dabei stehen die beiden ersten Ziffern „3.0“ für V3.0 und die beiden letzten „1.0“ für „A00“.

status string:
Status des SDOs

Mögliche Werte:

"Reserved"	Diese AOID ist reserviert.
"WaitingforTimestamp"	Das SDO wurde archiviert, ist aber noch nicht versiegelt.
"Sealed"	Das SDO wurde archiviert und mit einem Zeitstempel versiegelt.
"Deleted"	Das SDO wurde gelöscht.
"WaitingForRenewDocHash"	Für das SDO wurde eine Neuversiegelung initiiert. Die Erstellung eines neuen Evidence Records (mit Neuberechnung des Hash-Wertes) ist in Arbeit.
"WaitingForRenewTspSig"	Für das SDO wurde eine Neuversiegelung initiiert. Die Erstellung eines neuen Evidence Records ist in Arbeit.

mandant string:
Name des Mandanten

organisation
string:
Name der Organisationseinheit

sdoPath string:
Ablageort des SDOs im Archiv. Die Angabe ist relativ zu dem Pfad, der den Archivbereich der Organisationseinheit des Mandanten bezeichnet.

sdoType string:
SDO-Typ des SDOs

sdoTypeReplaceable
boolean:
optional; gibt an, ob das SDO überschrieben werden kann.

Mögliche Werte:

"true"	das SDO kann überschrieben werden.
"false"	das SDO kann nicht überschrieben werden.

externalReferences

Datentyp TStatusSdoExternalReferences, siehe ["Operation statusSDO"](#):
optional; gibt an, ob das SDO eines oder mehrere der Elemente \$ContentRef bzw. \$ExternalRef enthält, d.h. ob für die AOID externe Referenz-IDs und damit externe Datenobjekte existieren.
externalReferences wird nicht ausgegeben, wenn das SDO den Status Reserved hat.

sdoSize

long:
Größe des gespeicherten SDOs in Byte

shortSubject

string:
Inhalt des Systemschlüssels \$Subject, falls dessen Länge kleiner oder gleich 200 ist, ansonsten die ersten 200 Zeichen dieses Systemschlüssels

lastModificationDateTime

dateTime:
Zeitpunkt der letzten Statusänderung des SDOs

expirationDateTime

dateTime:
Aus den Angaben \$ExpirationDate bzw. \$RetentionPeriod berechneter Freigabezeitpunkt des SDOs (oder der durch die Operation setExpirationDateTimeSDO gesetzte Wert).

expirationDateUnlimited

Datentyp TEmptyElement, siehe "[Operation statusSDO](#)":
Das leere Element <expirationDateUnlimited/> wird anstelle von
<expirationDateTime> geliefert, wenn der Freigabezeitpunkt auf UNLIMITED gesetzt ist.

expirationDateDeferred

Datentyp TEmptyElement, siehe "[Operation statusSDO](#)":
Das leere Element <expirationDateDeferred/> wird anstelle von
<expirationDateTime> geliefert, wenn der Freigabezeitpunkt auf DEFERRED gesetzt ist.

archiveDateTime

dateTime:
Zeitpunkt der Archivierung des SDOs

i Dies ist der Zeitpunkt, zu dem das SDO im Archivsystem abgelegt wurde, und nicht der Zeitpunkt der Versiegelung des SDOs.

versionDateTime

dateTime:
Reservierungszeitpunkt der AOID

versionNumber

long:
Nummer einer der zur ausgegebenen COID gehörenden Versionen

sealedDateTime

dateTime:
Zeitpunkt der Versiegelung des SDOs.

tsp

string:
Liste der TSP-Namen, die für die Versiegelung verwendet wurden.

Element externalReferences vom Datentyp TStatusSdoExternalReferences

```
<xs:complexType name="TStatusSdoExternalReferences">
  <xs:sequence>
    <xs:element name="externalReference"
      type="TStatusSdoExternalReference"
      minOccurs="1" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
```

externalReference

Datentyp TStatusSdoExternalReference, siehe ["Operation statusSDO"](#):
Information über eine externeReferenz-ID

Element externalReference vom Datentyp TStatusSdoExternalReference

```
<xs:complexType name="TStatusSdoExternalReference">
  <xs:sequence>
    <xs:element name="refId" type="xs:string"
      minOccurs="1" maxOccurs="1" />
    <xs:element name="description" type="xs:string"
      minOccurs="0" maxOccurs="1" />
    <xs:element name="hashValue" type="xs:base64Binary"
      minOccurs="0" maxOccurs="1" />
    <xs:element name="hashAlgorithmName" type="xs:string"
      minOccurs="0" maxOccurs="1" />
  </xs:sequence>
</xs:complexType>
```

refId

string:
Externe Referenz-ID

i Der Aufbau der Referenz-ID ist keine garantierte Schnittstelle.

description

string:
optional; Beschreibungstext der externen Datei, der von der Client-Anwendung beim Anfordern der externen Referenz-ID mit `getAOIDWithRef` angegeben wurde.

hashValue

base64Binary:
optional; Hash-Wert der externen Datei, der von der Client-Anwendung beim Anfordern der externen Referenz-ID mit `getAOIDWithRef` angegeben wurde.

hashAlgorithmName

string:
optional; Hash-Algorithmus der externen Datei, der von der Client-Anwendung beim Anfordern der externen Referenz-ID mit `getAOIDWithRef` angegeben wurde.

Datentyp TEmptyElement

```
<xs:complexType name="TEmptyElement">
  <xs:annotation>
    <xs:documentation>XXXX</xs:documentation>
  </xs:annotation>
</xs:complexType>
```

Elemente im Response-Body

Unabhängig vom Status enthält die Response folgende Elemente:

- aoid
- version
- status
- mandant
- organization
- lastModificationDateTime
- versionDateTime
- versionNumber
- externalReferences, falls das SDO externe Referenzen enthält
- coid, wenn bei SubmitSDO eine COID angegeben wurde

Abhängig vom Status des SDOs werden im Response-Body die folgenden Elemente übermittelt:

- Im Status `Reserved` werden folgende Elemente nicht übermittelt, da sie zu diesem Zeitpunkt keine Werte enthalten:
 - `sdoPath`
 - `sdoType`
 - `sdoSize`
 - `archiveDateTime`
 - `expirationDate`, `expirationDateUnlimited` bzw. `expirationDateDeferred`
- Im Status `WaitingforTimestamp` können schon diejenigen TSPs in der Response übermittelt werden, mit denen das SDO bereits versiegelt wurde. In diesem Fall wird zusätzlich das folgende Element ausgegeben:
 - `tsp`
- Der Status `Sealed` wird erst erreicht, wenn das SDO mit allen zugeordneten TSPs versiegelt wurde. In der Response werden alle TSPs übermittelt, die für die Versiegelung verwendet wurden. Im Status `Sealed` werden zusätzlich die folgenden Elemente ausgegeben:
 - `sealedDateTime`
 - `tsp`
- Im Status `WaitingForRenewDocHash` und `WaitingForRenewTspSig` werden folgende Elemente ausgegeben:
 - `sealedDateTime`
 - `tsp`

Beispiel

Request

```
<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>statusSDO</secdocs:operation>
    <secdocs:aoid>ad5dc55e-4c47-4171-9887-0c2857b89b2b</secdocs:aoid>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:statusSDORequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>
```

Response im Status Reserved

```

<soap:Body>
  <tns:statusSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>ad5dc55e-4c47-4171-9887-0c2857b89b2b</tns:aoid>
    <tns:version>3.0.1.0</tns:version>
    <tns:status>Reserved</tns:status>
    <tns:mandant>Mandant1</tns:mandant>
    <tns:organisation>Department1</tns:organisation>
    <tns:lastModificationDateTime>
      2015-02-10T15:23:17.599+01:00</tns:lastModificationDateTime>
    <tns:versionDateTime>
      2015-02-10T15:23:17.599+01:00</tns:versionDateTime>
    <tns:versionNumber>0</tns:versionNumber>
  </tns:statusSDOResponse>
</soap:Body>

```

Response im Status WaitingForTimestamp

```

<soap:Body>
  <tns:statusSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>ad5dc55e-4c47-4171-9887-0c2857b89b2b</tns:aoid>
    <tns:version>3.0.1.0</tns:version>
    <tns:status>WaitingForTimestamp</tns:status>
    <tns:mandant>Mandant1</tns:mandant>
    <tns:organisation>Department1</tns:organisation>
    <tns:sdoPath><![CDATA[DocumentNone/2015/02/10]]">"</tns:sdoPath>
    <tns:sdoType>DocumentNone</tns:sdoType>
    <tns:sdoTypeReplaceable>false</tns:sdoTypeReplaceable>
    <tns:sdoSize>2072</tns:sdoSize>
    <tns:lastModificationDateTime>
      2015-02-10T15:27:40.466+01:00</tns:lastModificationDateTime>
    <tns:expirationDateTime>
      2019-02-10T15:27:40.000+01:00</tns:expirationDateTime>
    <tns:archiveDateTime>
      2015-02-10T15:27:40.466+01:00</tns:archiveDateTime>
    <tns:versionDateTime>
      2015-02-10T15:23:17.599+01:00</tns:versionDateTime>
    <tns:versionNumber>0</tns:versionNumber>
  </tns:statusSDOResponse>
</soap:Body>

```

Response im Status Sealed

```

<soap:Body>
  <tns:statusSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>ad5dc55e-4c47-4171-9887-0c2857b89b2b</tns:aoid>
    <tns:version>3.0.1.0</tns:version>
    <tns:status>Sealed</tns:status>
    <tns:mandant>Mandant1</tns:mandant>
    <tns:organisation>Department1</tns:organisation>
    <tns:sdoPath><![CDATA[DocumentNone/2015/02/10]]">"</tns:sdoPath>
    <tns:sdoType>DocumentNone</tns:sdoType>
    <tns:sdoTypeReplaceable>false</tns:sdoTypeReplaceable>
    <tns:sdoSize>2072</tns:sdoSize>
    <tns:lastModificationDateTime>
      2015-02-10T15:48:56.474+01:00</tns:lastModificationDateTime>
    <tns:expirationDateTime>
      2019-02-10T15:27:40.000+01:00</tns:expirationDateTime>
    <tns:archiveDateTime>
      2015-02-10T15:27:40.466+01:00</tns:archiveDateTime>
    <tns:versionDateTime>
      2015-02-10T15:23:17.599+01:00</tns:versionDateTime>
    <tns:versionNumber>0</tns:versionNumber>
    <tns:sealedDateTime>
      2015-02-10T15:48:56.474+01:00</tns:sealedDateTime>
    <tns:tsp>TSP-DFN</tns:tsp>
  </tns:statusSDOResponse>lc
</soap:Body>

```

Response im Status WaitingForTimeStamp für eine Dokumenten-Version

```

<soap:Body>
  <tns:statusSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>4468fb37-55fb-499f-a8ff-e504330d1bda</tns:aoid>
    <tns:coid>myCOID1382689045840</tns:coid>
    <tns:version>3.0.1.0</tns:version>
    <tns:status>WaitingForTimestamp</tns:status>
    <tns:mandant>Mandant1</tns:mandant>
    <tns:organisation>Department1</tns:organisation>
    <tns:sdoPath><![CDATA[DocumentNone/2015/02/10]]">"</tns:sdoPath>
    <tns:sdoType>DocumentNone</tns:sdoType>
    <tns:sdoTypeReplaceable>false</tns:sdoTypeReplaceable>
    <tns:sdoSize>2072</tns:sdoSize>
    <tns:lastModificationDateTime>
      2015-02-10T10:46:26.749+01:00</tns:lastModificationDateTime>
    <tns:expirationDateTime>
      2019-02-10T10:46:26.000+01:00</tns:expirationDateTime>
    <tns:archiveDateTime>
      2015-02-10T10:46:26.749+01:00</tns:archiveDateTime>
    <tns:versionDateTime>
      2015-02-10T10:46:26.675+01:00</tns:versionDateTime>
    <tns:versionNumber>10</tns:versionNumber>
  </tns:statusSDOResponse>lc
</soap:Body>

```

Response im Status Sealed für ein SDO mit externer Referenz-ID

```
<soap:Body>
  <tns:statusSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>9821a720-a4e2-479d-b0ab-8a400c81a98f</tns:aoid>
    ...
    <tns:externalReferences>
      <tns:externalReference>
        <tns:refId>_9821a720-a4e2-479d-b0ab-8a400c81a98f/1</tns:refId>
        <tns:description>Picture1</tns:description>
        <tns:hashValue>0528a5f5e...9a3aacbe6cc3</tns:hashValue>
        <tns:hashAlgorithmName>SHA-256</tns:hashAlgorithmName>
      </tns:externalReference>
      <tns:externalReference>
        <tns:refId>_9821a720-a4e2-479d-b0ab-8a400c81a98f/2</tns:refId>
        <tns:hashValue>aec2ec728...8388e97bdb20</tns:hashValue>
        <tns:hashAlgorithmName>SHA-256</tns:hashAlgorithmName>
      </tns:externalReference>
      <tns:externalReference>
        <tns:refId>_9821a720-a4e2-479d-b0ab-8a400c81a98f/3</tns:refId>
        <tns:description>SignedDocumentXY</tns:description>
      </tns:externalReference>
    </tns:externalReferences>
    <tns:sdoSize>821</tns:sdoSize>
    ...
  </tns:statusSDOResponse>
</soap:Body>
```

3.3.9 Operation listSDOVersions

listSDOVersions gibt eine Liste der Versionen eines Dokuments aus.

Für jede Version des angegebenen Dokuments wird ein Element ausgegeben, das die gleichen Informationen enthält, die auch von der Operation statusSDO zurückgegeben werden.

Die Ausgabeelemente sind in absteigender Reihenfolge nach der Versionsnummer (Element `versionNumber`) sortiert, d.h. die Version, für die als letztes eine AOID reserviert wurde, wird als erstes Element ausgegeben.

Das Dokument wird durch die Angabe der COID im SOAP-Header identifiziert.

Die AOID darf bei dieser Operation nicht angegeben werden.

Request-Body

Leeres Element listSdoVersionsRequest *vom Datentyp* TListSdoVersionsRequest

```
<tns:listSdoVersionsRequest
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Element listSdoVersionsResponse *vom Datentyp* TListSdoVersionsResponse

```
<xs:complexType name="TListSdoVersionsResponse">
  <xs:annotation>
    <xs:documentation>SecDocs SOAP body response data for the WS
      operation listSdoVersions</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element ref="statusSDOResponse" minOccurs="0"
      maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
</xsd:complexType>
```

statusSDOResponse

Datentyp TStatusSDOResponse, siehe ["Operation statusSDO"](#).

Beispiel

Response

```

<tns:listSdoVersionsResponse
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
  <tns:statusSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>26f24b12-f126-4114-a088-de73644c6084</tns:aoid>
    <tns:coid>myCOID1382689045840</tns:coid>
    <tns:version>3.0.1.0</tns:version>
    <tns:status>WaitingForTimestamp</tns:status>
    <tns:mandant>Mandant1</tns:mandant>
    <tns:organisation>Department1</tns:organisation>
    <tns:sdoPath><![CDATA[2013/10/25/10]]">"</tns:sdoPath>
    <tns:sdoType>DocumentNone</tns:sdoType>
    <tns:sdoTypeReplaceable>>false</tns:sdoTypeReplaceable>
    <tns:sdoSize>13298</tns:sdoSize>
    <tns:shortSubject>
      myCOID1382689045840:FileReference=NoAKZ NoOfPages=No of
pages unknown Subject=No Subject
    </tns:shortSubject>
    <tns:lastModificationDateTime>
      2013-10-25T10:17:30.481+02:00
    </tns:lastModificationDateTime>
    <tns:expirationDateTime>
      2013-10-30T09:17:29.000+01:00
    </tns:expirationDateTime>
    <tns:archiveDateTime>
      2013-10-25T10:17:30.481+02:00
    </tns:archiveDateTime>
    <tns:versionDateTime>
      2013-10-25T10:17:29.282+02:00
    </tns:versionDateTime>
    <tns:versionNumber>1</tns:versionNumber>
  </tns:statusSDOResponse>
  <tns:statusSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:aoid>2231af49-6b0b-466d-978d-8a14b0942b6c</tns:aoid>
    <tns:coid>myCOID1382689045840</tns:coid>
    <tns:version>3.0.1.0</tns:version>
    <tns:status>WaitingForTimestamp</tns:status>
    <tns:mandant>Mandant1</tns:mandant>
    <tns:organisation>Department1</tns:organisation>
    <tns:sdoPath> <![CDATA[2013/10/25/10]]">" </tns:sdoPath>
    <tns:sdoType>DocumentNone</tns:sdoType>
    <tns:sdoTypeReplaceable>>false</tns:sdoTypeReplaceable>
    <tns:sdoSize>13298</tns:sdoSize>

```

```
<tns:shortSubject>
  myCOID1382689045840:FileReference=NoAKZ NoOfPages=No of
pages unknown Subject=No Subject
</tns:shortSubject>
<tns:lastModificationDateTime>
  2013-10-25T10:17:28.574+02:00
</tns:lastModificationDateTime>
<tns:expirationDateTime>
  2013-10-30T09:17:28.000+01:00
</tns:expirationDateTime>
<tns:archiveDateTime>
  2013-10-25T10:17:28.574+02:00
</tns:archiveDateTime>
<tns:versionDateTime>
  2013-10-25T10:17:27.460+02:00
</tns:versionDateTime>
<tns:versionNumber>0</tns:versionNumber>
</tns:statusSDOResponse>
</tns:listSdoVersionsResponse>
```

3.3.10 Operation retrieveMetaData

retrieveMetaData gibt eine Untermenge der Informationen aus der Operation statusSDO zurück. Die XML-Elemente in der SOAP-Response sind benannt nach der Vorlage des „Protection Profile for an ArchiSafe Compliant Middleware for Enabling the Long-Term Preservation of Electronic Documents“ für die Operation Read Metadata Information.

Im SOAP-Header müssen dabei die AOID oder die COID des SDOs und die Organisation angegeben werden, unter der das SDO archiviert wurde.

Wenn nur eine COID, aber keine AOID angegeben ist und mehrere Versionen des Dokuments mit dieser COID existieren, werden die Daten für das SDO mit der höchsten Version geliefert. Um die Daten für andere Versionen des Dokuments zu erhalten, muss die AOID angegeben werden. Diese können Sie ggf. mit der Operation listSDOVersions ermitteln, siehe "[Operation listSDOVersions](#)".

Voraussetzungen

Das SDO ist nicht im Zustand Reserved oder Deleted. Dies kann mit der Operation statusSDO überprüft werden, siehe "[Operation statusSDO](#)".

Request-Body

Leeres Element vom Datentyp TRetrieveMetaDataRequest

```
<tns:retrieveMetaDataRequest  
  
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Datentyp TRetrieveMetaDataResponse

```
<xs:complexType name="TRetrieveMetaDataResponse">  
  <xs:annotation>  
    ...  
  </xs:annotation>  
  <xs:sequence>  
    <xs:element name="ObjectId" type="TUuid"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:choice>  
      <xs:element name="RetentionTime" type="xs:dateTime"  
        minOccurs="1" maxOccurs="1"/>  
      <xs:element name="RetentionTimeUnlimited" type="xs:TEEmptyElement"  
        default="true" minOccurs="1" maxOccurs="1"/>  
      <xs:element name="RetentionTimeDeferred" type="xs:TEEmptyElement"  
        minOccurs="1" maxOccurs="1"/>  
    </xs:choice>  
    <xs:element name="ArchiveDate" type="xs:dateTime"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="SDOType" type="xs:string"  
      minOccurs="1" maxOccurs="1"/>  
  </xs:sequence>  
</xs:complexType>
```

ObjectID

string (Universally Unique Identifier gemäß Standard IETF RFC 4122):
Archivweit eindeutige Identifikation für das zu archivierende Objekt.

RetentionTime

dateTime:
Freigabezeitpunkt des SDOs.
Dieses Element wird geliefert, wenn der Freigabezeitpunkt auf ein bestimmtes Datum gesetzt ist.

RetentionTimeUnlimited

Datentyp `EmptyElement`, siehe "[Operation statusSDO](#)":
Dieses Element wird als leeres Element `<RetentionTimeUnlimited/>` anstelle von
`<RetentionTime>` geliefert, wenn der Freigabezeitpunkt auf `UNLIMITED` gesetzt ist.

RetentionTimeDeferred

Datentyp `EmptyElement`, siehe "[Operation statusSDO](#)":
Das leere Element `<RetentionTimeDeferred/>` wird anstelle von `<RetentionTime>` geliefert, wenn
der Freigabezeitpunkt auf `DEFERRED` gesetzt ist.

ArchiveDate

dateTime:
Zeitpunkt der Archivierung des SDOs.

i Dies ist der Zeitpunkt, zu dem das SDO im Archivsystem abgelegt wurde, und nicht der
Zeitpunkt der Versiegelung des SDOs.

SDOType

string:
SDO-Typ des SDOs.

Beispiel

Request

```
<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>retrieveMetaData</secdocs:operation>
    <secdocs:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aoid>
    <secdocs:SDOType>DocumentNone</secdocs:SDOType>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:retrieveMetaDataRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>
```

Response

```
<soap:Body>
  <tns:retrieveMetadataResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:ObjectId>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</tns: ObjectId>
    <tns:RetentionTime>2015-12-01T09:41:07.424+01:00</tns: RetentionTime>
    <tns:ArchiveDate>2012-12-01T09:41:07.565+01:00</tns:ArchiveDate>
    <tns:SDOType>DocumentNone</tns:SDOTyp>
  </tns: retrieveMetadataResponse>
</soap:Body>
```

3.3.11 Operation metaDataSDO

metaDataSDO gibt die im Filter durch den Benutzer definierten Schlüssel und deren Werte aus (siehe Abschnitt „Benutzerdefinierte Schlüssel“ (in ["Verwendung von Schlüsseln"](#))). Zusätzlich werden alle Systemschlüssel außer \$Content, \$ExpirationDate, \$RetentionPeriod, \$SDOPath und \$Signature mit ihren Werten ausgegeben. Das berechnete Verfallsdatum wird unter dem Schlüsselnamen \$ExpirationDateTime zurückgeliefert. Ist der Freigabezeitpunkt noch nicht definiert, wird der Wert DEFERRED ausgegeben. Wird für \$ExpirationDateTime im Element value der Wert UNLIMITED ausgegeben, so verfällt das Dokument nie.

Im SOAP-Header müssen Sie dabei die AOID oder die COID des SDOs und die Organisation angeben, unter der das SDO archiviert wurde.

Wenn nur eine COID, aber keine AOID angegeben ist und mehrere Versionen des Dokuments mit dieser COID existieren, werden die Daten für das SDO mit der höchsten Version geliefert. Um die Daten für andere Versionen des Dokuments zu erhalten, muss die AOID angegeben werden. Diese können Sie ggf. mit der Operation listSDOVersions ermitteln, siehe ["Operation listSDOVersions"](#).

Voraussetzungen

Das SDO ist nicht im Zustand Reserved oder Deleted. Dies kann mit der Operation statusSDO überprüft werden, siehe ["Operation statusSDO"](#).

Request-Body

Leeres Element vom Datentyp TMetaDataSDORequest

```
<tns:metaDataSDORequest  
  
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Datentyp TMetaDataSDOResponse

```
<xs:complexType name="TMetaDataSDOResponse">  
  <xs:annotation>  
    ...  
  </xs:annotation>  
  <xs:sequence>  
    <xs:element name="version" type="xs:string"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="coid" type="TCoid"  
      minOccurs="0" maxOccurs="1"/>  
    <xs:element name="versionNumber"  
      type="xs:nonNegativeInteger"  
      minOccurs="0" maxOccurs="1"/>  
    <xs:element name="metaDataValue" type="TMetaDataValue"  
      minOccurs="0" maxOccurs="unbounded"/>  
    <xs:element name="subMap" type="TSubMapMetaData"  
      minOccurs="0" maxOccurs="unbounded"/>  
  </xs:sequence>  
</xs:complexType>
```

version

SecDocs-interne Version, in der die Metadaten abgelegt wurden.

coid

COID (Client Object Identifier), vom Anwender vergebbarer Bezeichner zur Identifizierung eines Archivobjekts.

Datentyp TCoid, siehe ["Request-Header"](#).

versionNumber

nonNegativeInteger:

Nummer der Dokumentenversion.

metaDataValue

Informationen für einen Metadateneintrag (gefilterte Daten):

kein, ein oder mehrere Datentypen TMetaDataValue, siehe unten.

subMap

optional; Metadaten für ein Element <node> im Filter.

kein, ein oder mehrere Datentypen TSubMapMetaData, siehe ["Datentyp TSubMapMetaData"](#).

Der Datentyp wird für jedes Element <node> im Filter einmal ausgegeben.

Datentyp TMetaDataValue

```
<xs:complexType name="TMetaDataValue">
  <xs:sequence>
    <xs:element name="name" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="value" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="type" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="xpath" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="defaultValue" type="xs:boolean"
      minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

name string:
Name des Metadatums

value string:
optional; Wert des Metadatums

Dieses Element ist nur vorhanden,

- wenn für das Metadatum bei der Filterung ein Wert ermittelt wurde.
- wenn bei der Filterung kein Wert ermittelt wurde, aber im Filter für dieses Metadatum ein `default`-Wert angegeben ist. `value` enthält dann diesen `default`-Wert.

type string:
optional; Typ des Metadatums

xpath string:
optional; XPath-Ausdruck im Filter

Dieses Element ist nur vorhanden, wenn in einem Filterausdruck das Attribut `xpath=` angegeben wurde.

defaultValue

boolean:
Angabe, ob ein `default`-Wert verwendet wurde.

Mögliche Werte:

- "true" Ein `default`-Wert wurde verwendet.
- "false" Beim Filtern wurde ein Wert gefunden oder berechnet, daher wurde kein `default`-Wert verwendet.

Datentyp TSubMapMetaData

```
<xs:complexType name="TSubMapMetaData">
  <xs:sequence>
    <xs:element name="metaDataValue" type="TMetaDataValue"
      minOccurs="0" maxOccurs="unbounded" />
    <xs:element name="subMap" type="TSubMapMetaData"
      minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="xpath" use="optional" type="xs:string"/>
</xs:complexType>
```

metaDataValue

Informationen für einen Metadateneintrag (gefilterte Daten):
kein, ein oder mehrere Datentypen `TMetaDataValue`, siehe "[Datentyp TMetaDataValue](#)".

subMap optional; Dieses Element ist für zukünftige Erweiterungen vorgesehen und wird derzeit nicht ausgegeben.

xpath XPath-Ausdruck, mit dem die Daten im SDO bei `$Content` adressiert werden. Dies schließt auch den XPath-Wert von `$DataNode` ein.

Wenn im SDO keine Daten für `$Content` übergeben wurden, wird hier auch kein Wert für XPath geliefert.

Beispiel

Request

```
<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>metaDataSDO</secdocs:operation>
    <secdocs:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aoid>
    <secdocs:SDOType>DocumentNone</secdocs:SDOType>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:metaDataSDORequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>
```

Response

```
<soap:Body>
  <tns:metaDataSDOResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
  <tns:version>3.0</tns:version>
  <tns:coid>myCOID1382689045840</tns:coid>
  <tns:versionNumber>3</tns:versionNumber>
  <tns:metaDataValue>
    <tns:name>$Day</tns:name>
    <tns:value>15</tns:value>
    <tns:defaultValue>true</tns:defaultValue>
  </tns:metaDataValue>
  <tns:metaDataValue>
    <tns:name>mdKeywords</tns:name>
    <tns:value>pdf</tns:value>
    <tns:xpath>/{http://ts.fujitsu.com/secdocs/sdosamples/mydocument}
      myDocument[1]/metaData[1]/keywords[1]</tns:xpath>
    <tns:defaultValue>false</tns:defaultValue>
  </tns:metaDataValue>
```

```
<tns:metaDataValue>
  <tns:name>mdDescription</tns:name>
  <tns:value>PDF file without any type of signature</tns:value>
  <tns:xpath>/{http://ts.fujitsu.com/secdocs/sdosamples/mydocument}
    myDocument[1]/metaData[1]/description[1]</tns:xpath>
  <tns:defaultValue>false</tns:defaultValue>
</tns:metaDataValue>
...
<tns:metaDataValue>
  <tns:name>$Hour</tns:name>
  <tns:value>10</tns:value>
  <tns:defaultValue>true</tns:defaultValue>
</tns:metaDataValue>
<tns:metaDataValue>
  <tns:name>$ExpirationDateTime</tns:name>
  <tns:value>2017-05-25T06:47:05Z</tns:value>
  <tns:defaultValue>false</tns:defaultValue>
</tns:metaDataValue>
<tns:metaDataValue>
  <tns:name>mdDepartment</tns:name>
  <tns:value>Department1</tns:value>
  <tns:xpath>/{http://ts.fujitsu.com/secdocs/sdosamples/mydocument}
    myDocument[1]/metaData[1]/department[1]</tns:xpath>
  <tns:defaultValue>false</tns:defaultValue>
</tns:metaDataValue>
</tns:metaDataSDOResponse>
</soap:Body>
```


3.3.12 Operation requestForEvidence

`requestForEvidence` liefert für ein archiviertes SDO die Daten, die für die externe Verifikation benötigt werden:

- SDO
- Signature Verification Information
- Alle zugehörigen, vorhandenen Evidence Records

Mit Hilfe dieser Daten kann eine externe Verifikation durchgeführt werden.

Falls das SDO Referenzen auf externe Datenobjekte (externe Referenz-IDs) enthält, gelten die Signatur Verification Information und vorhandene Evidence Records auch für die zu dem SDO gehörenden externen Datenobjekte.

Im SOAP-Header müssen Sie dabei die AOID oder die COID des SDOs und die Organisation angeben, unter der das SDO archiviert wurde.

Wenn nur eine COID, aber keine AOID angegeben ist und mehrere Versionen des Dokuments mit dieser COID existieren, werden die Daten für das SDO mit der höchsten Version geliefert. Um die Daten für andere Versionen des Dokuments zu erhalten, muss die AOID angegeben werden. Diese können Sie ggf. mit der Operation `listSDVersions` ermitteln, siehe "[Operation listSDVersions](#)".

Request-Body

Element `requestForEvidenceRequest`: vom Datentyp `TRequestForEvidenceRequest`

```
<xs:complexType name="TRequestForEvidenceRequest">
  <xs:annotation>
    <xs:documentation>
      SecDocs SOAP body request data for the
      WS operation requestForEvidence
    </xs:documentation>
  </xs:annotation>
  <xs:attribute name="getHtmlProtocols" use="optional" default="false"
    type="xs:boolean" />
</xs:complexType>
```

`getHtmlProtocols`



Die Ausgabe der XHTML-Protokolle wird ab der SecDocs Version 3.1A00 nicht mehr unterstützt!

Bei Angabe des Attributs `getHtmlProtocols` werden die XHTML-Protokolle mit den Ergebnissen der Signaturprüfung und der Prüfung der Evidence Records erzeugt.

Der Standardwert für das Attribut `getHtmlProtocols` ist "false".

Die Erzeugung der XHTML-Protokolle bei der Operation `requestForEvidence` wird ausschließlich durch das Attribut `getHtmlProtocols` gesteuert, nicht jedoch durch die Property `getHTMLProtocolsOnRetrieval` in der Datei `secdocs.properties`.

Response-Body

Element `requestForEvidenceResponse` *Datentyp* `TRequestForEvidenceResponse`

```

<xs:complexType name="TRequestForEvidenceResponse">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="SignatureVerificationInfo"
      type="xs:base64Binary"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="SignatureVerificationHtmlProtocol"
      type="xs:base64Binary"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="SDO" type="xs:base64Binary"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="EvidenceRecord" type="TEvidenceRecord"
      minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xsd:complexType>

```

i Wenn das SDO noch nicht versiegelt ist oder dem SDO-Typ nur ein TSP zugeordnet ist und gerade eine renewTSP-Operation läuft, kann dieser Aufruf keine Evidence Records liefern.

SignatureVerificationInfo

base64Binary:
Daten der zugehörigen Signaturprüfungen

SignatureVerificationHtmlProtocol

i Die Ausgabe der XHTML-Protokolle wird ab der SecDocs Version 3.1A00 nicht mehr unterstützt!

base64Binary:
optional; als XHTML-Datei aufbereitete Daten aus dem Element <SignatureVerificationInfo>.
Dieses Element wird nur ausgegeben, wenn im Request explizit getHtmlProtocols auf "true" gesetzt wurde.

SDO

base64Binary:
Bei der Operation submitSDO oder replaceSDO gespeichertes SDO

EvidenceRecord

Daten zu einem Evidence Record:
ein oder mehrere Datentypen TEvidenceRecord, siehe unten

Element EvidenceRecord vom Datentyp TEvidenceRecord

```

<xs:complexType name="TEvidenceRecord">
  <xs:sequence>
    <xs:element name="TSPName" type="TNonEmptyString"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="ERData" type="xs:base64Binary"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="ERVerificationStatus"
      type="TERVerificationStatus"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="ERHtmlVerificationProtocol"
      type="xs:base64Binary"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="ERXmlVerificationProtocol"
      type="xs:base64Binary"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="HDOHashAlgorithmName"
      type="TNonEmptyString"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="HDOHashAlgorithmOID"
      type="TNonEmptyString"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="HDOHashValue" type="THDOHashValue"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>

```

TSPName

TNonEmptyString:
Name des verwendeten TSPs

ERData

base64Binary:
Daten des Evidence Records

ERVerificationStatus

Ergebnis der Evidence-Record-Verifikation, wie sie im Prüfprotokoll (ERXmlVerificationProtocol) beschrieben ist;
Datentyp TERVerificationStatus, siehe "[Datentyp TERVerificationStatus](#)".

Das Ergebnis der Evidence-Record-Verifikation bezieht sich nur auf das SDO und die Signature Verification Information. Dementsprechend bezieht sich die Erfolgsmeldung im Element ERVerificationStatus nicht auf externe Datenobjekte.

Um die Integrität der externen Objekte zu überprüfen können Sie ein externes Tool verwenden, da sich der Beweiswert der zurückgelieferten Evidence Records auch auf die externen Objekte erstreckt.

ERXmlVerificationProtocol

base64Binary:
zugehöriges Prüfprotokoll im XML-Format.

ERHtmlVerificationProtocol

i Die Ausgabe der XHTML-Protokolle wird ab der SecDocs Version 3.1A00 nicht mehr unterstützt!

base64Binary:
optional; zugehöriges Prüfprotokoll im XHTML-Format.

Dieses Element wird nur ausgegeben, wenn im Request `getHtmlProtocols` auf "true" gesetzt wurde.

HDOHashAlgorithmName

TNonEmptyString:
Name des verwendeten Hash-Algorithmus (z.B. SHA-256)

HDOHashAlgorithmOID

TNonEmptyString:
OID des verwendeten Hash-Algorithmus (z.B. 2.16.840.1.101.3.4.2.1)

HDOHashValue

berechneter Hash-Wert für das Hash Data Object (HDO). Das HDO besteht aus der Konkatenation von `SignatureVerificationInfo` und `SDO` (in dieser Reihenfolge): Datentyp `THDOHashValue`, siehe ["Datentyp THDOHashValue"](#)

Datentyp TERVerificationStatus

```
<xs:simpleType name="TERVerificationStatus">
  <xs:restriction base="xs:string">
    <!-- successful verification -->
    <xs:enumeration value="SUCCESS" />
    <!-- verification with warnings -->
    <xs:enumeration value="WARN" />
    <!-- verification failed -->
    <xs:enumeration value="FAIL" />
    <!-- verification result unknown -->
    <xs:enumeration value="UNDEFINED" />
  </xs:restriction>
</xs:simpleType>
```

i `TERVerificationStatus` zeigt das Ergebnis der Verifikation in Kurzform an. Bei einem Ergebnis `WARN` oder `Fail` finden Sie die Detail-Informationen im zugehörigen Prüfprotokoll.

SUCCESS	Die Verifikation war erfolgreich.
WARN	Die Verifikation war mit Einschränkungen erfolgreich.
FAIL	Die Verifikation ist fehlgeschlagen.
UNDEFINED	Eine Verifikation konnte nicht durchgeführt werden. Es gibt daher kein definiertes Ergebnis.

Datentyp THDOHashValue

```
<xs:complexType name="THDOHashValue">
  <xs:sequence>
    <xs:element name="HashValue" type="xs:base64Binary"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="HashValueHex" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

HashValue

base64Binary:
berechneter Hash-Wert

HashValueHex

string:
berechneter Hash-Wert als abdruckbare Hexadezimal-Zeichen.
Dieser Wert wird beim Lesen der Prüfprotokolle verwendet. In den Prüfprotokollen werden die Hash-Werte oft als Hexadezimal-Zeichenketten dargestellt.

Beispiel

Request

```
<soap:Header>
  <secdocs:soapHeaderData>
    <secdocs:security>
      <secdocs:principal>
        <secdocs:role>Archivar</secdocs:role>
        <secdocs:mandant>Mandant1</secdocs:mandant>
        <secdocs:orgID>Department1</secdocs:orgID>
      </secdocs:principal>
      <secdocs:password>secrets2</secdocs:password>
    </secdocs:security>
    <secdocs:auditID>Archivar1</secdocs:auditID>
    <secdocs:operation>requestForEvidence</secdocs:operation>
    <secdocs:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aoid>
    <secdocs:SDOType>DocumentNone</secdocs:SDOType>
  </secdocs:soapHeaderData>
</soap:Header>
<soap:Body>
  <tns:requestForEvidenceRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
</soap:Body>
```

Response

```
<soap:Body>
  <tns:requestForEvidenceResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:SDO>PD94bWwgd...</tns:SDO>
    <tns:SignatureVerificationInfo>PD94bWwgd...
    </tns:SignatureVerificationInfo>
    <tns:EvidenceRecord>
      <tns:TSPName>TSP-DFN</tns:TSPName>
      <tns:ERData>MIINKwIBAT...</tns:ERData>
      <tns:HDOHashAlgorithmName>SHA-256</tns:HDOHashAlgorithmName>
      <tns:HDOHashAlgorithmOID>2.16.840.1.101.3.4.2.1
      </tns:HDOHashAlgorithmOID>
    </tns:EvidenceRecord>
    <tns:EvidenceRecord>
      <tns:TSPName>TSP-DTRUST</tns:TSPName>
      <tns:ERData>MIIHvgIBAT...</tns:ERData>
      <tns:ERVerificationStatus>SUCCESS</tns:ERVerificationStatus>
      <tns:HDOHashAlgorithmName>SHA-256</tns:HDOHashAlgorithmName>
      <tns:HDOHashAlgorithmOID>2.16.840.1.101.3.4.2.1
      </tns:HDOHashAlgorithmOID>
      <tns:HDOHashValue>
        <tns:HashValue>MIIPvgIxad...</tns:HashValue>
        <tns:HashValueHex>a0flbc...</tns:HashValueHex>
      </tns:HDOHashValue>
    </tns:EvidenceRecord>
  </tns:requestForEvidenceResponse>
</soap:Body>
```

3.3.13 Operation navigate

Mit der Operation `navigate` können Sie durch die Ablagestrukturen Ihres Archivs navigieren, um die AOIDs von in SecDocs abgelegten Dokumenten zu finden. Die Operation erlaubt (durch wiederholte Aufrufe) die rekursive Navigation ausgehend von einer organisationsspezifischen Verzeichniswurzel über beliebig viele Zwischenknoten bis hin zu AOID-spezifischen Blattknoten. Die Suche kann auch von einem Zwischenknoten aus beginnen, wenn dieser Pfad bereits bekannt ist.

Die Navigation ist ähnlich einer Navigation im Filesystem organisiert: ein `path` entspricht dabei einem Verzeichnis und eine `AOID` einer Datei. Im Request ist der jeweils gewünschte `path` absolut anzugeben, d.h. mit einem führenden „/“. Das Wurzelverzeichnis (/) bezieht sich dabei auf das oberste Verzeichnis, das zu einer `OrgID` eines gegebenen Mandanten gehört.

Im SOAP-Body des Requests wird dabei der `path` als Element in der Request Message angegeben:

```
<tns:navigateRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"
>

    <tns:path><![CDATA[ / ]]></tns:path>

</tns:navigateRequest>
```

Im SOAP-Body der Response wird das Element `path` wiederholt. Im Anschluss werden Listen von Elementen ausgegeben, die unter diesem `path` zu finden sind. Es gibt zwei Typen von Listen: Die Liste `subpaths` enthält die Unterverzeichnisse, die Liste `aoids` enthält die gefundenen SDOs:

```
<tns:navigateResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">

    <tns:path><![CDATA[ / ]]></tns:path>
    ...

    <tns:subpaths>

        <tns:subpath><![CDATA[ /MyDocumentNone/ ]]></tns:subpath>
    </tns:subpaths>

    <tns:aoids>

        <tns:aoid id="3ea6f714-f6e9-46bc-b262-0e4a2adeca35">
            <tns:subject>Description of SDO content</tns:
subject>

            </tns:aoid>

            <tns:aoid id="bc3c9f17-9324-4ede-b14e-fbd17ae9d0b1"/>
        </tns:aoids>

</tns:navigateResponse>
```

- i** Ist für das SDO ein `$Subject` hinterlegt, wird dieses im Subelement `<subject>` mitausgegeben. Gibt es keine `$Subject` Informationen für dieses SDO, so fehlt dieses Subelement.

Die Elemente in der Liste sind entweder eine `AOID` oder ein `path`. Ist ein Element ein `path`, so kann dieses Element bei einem neuen Aufruf der Operation `navigateSDO` als `path` verwendet werden.

Beispiel

Im obigen Beispiel gibt es unter dem `path „/“` das Unterverzeichnis `/MyDocumentNone/`. Will man sich den Inhalt dieses Unterverzeichnisses ansehen, gibt man beim Request `/MyDocumentNone` als `path` an:

```
<tns:navigateRequest
xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"
>

    <tns:path><![CDATA[ /MyDocumentNone/ ]]></tns:path>
```

- i** Wurde für das SDO (bei `submitSDO`) eine `COID` vergeben, so wird diese als Attribut mit angezeigt (siehe Datentyp `TNavigateAOIDItem` auf "[Datentyp TNavigateAOIDItem](#)").

- i**
1. Die Operation `navigate` liefert Informationen über den Inhalt des SecDocs-Storage bzw. der Datenbank, wie er zum Zeitpunkt der Abfrage vorliegt. Der Inhalt des SecDocs-Storage bzw. der Datenbank kann sich stets durch parallel ablaufende Operationen wie z.B. `submitSDO` oder `deleteSDO` ändern. Deswegen können Informationen, die einmal mittels `navigate` ermittelt wurden, ihre Gültigkeit verlieren. Sie müssen ggf. neu beschafft werden.
 2. Geliefert werden Informationen ab Organisations-Ebene.
 3. Falls bei `submitSDO` oder `replaceSDO` der Systemschlüssel `$Subject` angegeben war, wird pro `AOID` abhängig von der Angabe bei `useDatabase` folgende Information geliefert:
 - `useDatabase="false"`:
Der Inhalt von `$Subject` wird geliefert
 - `useDatabase="true"`:
Die ersten 200 Zeichen des Inhalts von `$Subject` werden geliefert.

- i** Bei Archiven mit vielen Einträgen oder vielen Unterverzeichnissen in Archivpfaden müssen viele `navigate`-Operationen rekursiv durchgeführt werden, um vollständige Informationen über Archiv-Inhalte zu beschaffen. Dies kann in der Summe zu einer sehr langen Gesamt-Antwortzeit führen.

Request-Body

Element `navigateRequest` vom Datentyp `TNavigateRequest`


```

<xsd:complexType name="TNavigateRequest">
  <xsd:sequence>
    <xsd:element name="path" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xsd:element name="queryOptions" type="TNavigateQueryOptions"
      minOccurs="0" maxOccurs="1"/>
    <xsd:element name="maxResults" type="xs:unsignedInt"
      minOccurs="0" maxOccurs="1" default="0"/>
    <xsd:element name="start" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attribute name="useDatabase" use="optional" default="false"
    type="xs:boolean" />
</xsd:complexType>

```

path	<p>string:</p> <p>Verzeichnis, dessen Inhalte ausgegeben werden sollen (organisationsspezifisch absolut also mit „/“ beginnend).</p> <p>Verwenden Sie in diesem Parameter Werte, die von der <code>navigate</code>-Funktion in vorhergehenden Aufrufen im Element <code>subpath</code> der Response geliefert wurden, da aufgrund interner Zeichenumsetzungen eine SecDocs-spezifische Verzeichnisstruktur vorliegt.</p>
queryOptions	<p>Datentyp <code>TNavigateQueryOptions</code>, siehe "Datentyp TNavigateQueryOptions".</p> <p>optional; Legt fest, welche Ausgaben erfolgen sollen.</p>
maxResults	<p>unsignedInt:</p> <p>Legt fest, wie viele Objekte (Unterverzeichnisse und AOIDs) maximal ausgegeben werden sollen.</p> <p>Bei Angabe von „0“ werden alle Objekte ausgegeben.</p>
start	<p>string:</p> <p>Legt fest, ab welchem Objekt in der Liste die Ausgaben erfolgen sollen.</p> <p>Diese Angabe muss dem Wert von <code>next</code> in der vorhergehenden Navigate-Response entsprechen.</p>

useDatabase **boolean:**
gibt an, ob die zur Durchführung der Operation `navigate` notwendigen Informationen durch Zugriff auf die Datenbank (`useDatabase="true"`) oder durch Zugriff auf das Filesystem (`useDatabase="false"`) ermittelt werden.

Mögliche Werte:

"false" Die Operation `navigate` liefert sämtliche Knoten (Unterverzeichnisse bzw. AOID-Blattknoten), die im Filesystem unter dem im Request angegebenen `path` existieren.

"true" Die Operation `navigate` liefert ausschließlich die nichtleeren Knoten unter dem im Request angegebenen `path`, d.h. Sie benötigen für eine rekursive Navigation durch die Ablagestrukturen Ihres Archivs u.U. eine geringere Zahl an `navigate`-Aufrufen. Mit dem Zugriff auf die Datenbank anstelle auf das Filesystem und einer evtl. geringeren Zahl an Wiederholungsaufrufen können Sie die Gesamt-Antwortzeit verringern.

Standardwert: `false`

Datentyp `TNavigateQueryOptions`

```
<xsd:complexType name="TNavigateQueryOptions">
  <xsd:all>
    <xsd:element name="subPaths" type="xs:boolean"
      minOccurs="0" maxOccurs="1" default="true"/>
    <xsd:element name="aoids" type="tns:boolean"
      minOccurs="0" maxOccurs="1" default="true"/>
    <xsd:element name="subjects" type="xs:boolean"
      minOccurs="0" maxOccurs="1" default="true"/>
  </xsd:all>
</xsd:complexType>
```

subPaths

boolean:
Gibt an, ob Unterverzeichnisse ausgegeben werden sollen.

aoids

boolean:
Gibt an, ob AOIDs ausgegeben werden sollen.

subjects

boolean:
Gibt an, ob `$Subjects` (falls vorhanden) ausgegeben werden sollen.

Response-Body

Datentyp `TNavigateResponse`

```

<xsd:complexType name="TNavigateResponse">
  <xsd:sequence>
    <xsd:element name="path" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
    <xsd:element name="queryOptions" type="TNavigateQueryOptions"
      minOccurs="1" maxOccurs="1"/>
    <xsd:element name="maxResults" type="xs:unsignedInt"
      minOccurs="1" maxOccurs="1"/>
    <xsd:element name="totalResults" type="xs:unsignedInt"
      minOccurs="1" maxOccurs="1"/>
    <xsd:element name="next" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xsd:element name="subpaths" type="TNavigateSubpaths"
      minOccurs="0" maxOccurs="1"/>
    <xsd:element name="aoids" type="TNavigateAOIDs"
      minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>

```

path	string: Verzeichnis, dessen Inhalte ausgegeben werden.
queryOptions	Datentyp TNavigateQueryOptions , siehe " Datentyp TNavigateQueryOptions ": Zeigt an, mit welchen Optionen die Ausgaben erfolgen.
maxResults	unsignedInt: Zeigt an, wie viele Objekte (Unterverzeichnisse + AOIDs) ausgegeben werden sollen.
totalResults	unsignedInt: Zeigt an, wie viele Objekte (Unterverzeichnisse + AOIDs) verfügbar sind.
next	string: Zeigt das Objekt an, das beim nächsten Request für dieses (Unter-)Verzeichnis als start-Element angegeben werden kann. next wird nur ausgegeben, wenn maxResults != 0 und totalResults > maxResults ist.
subpaths	Datentyp TNavigateSubpaths , siehe " Datentyp TNavigateSubpaths ": Enthält die weiteren Unterverzeichnisse des aktuellen Verzeichnisses.
aoids	Datentyp TNavigateAOIDs , siehe " Datentyp TNavigateAOIDs ": Enthält die AOIDs im aktuellen Verzeichnis.

Datentyp [TNavigateSubpaths](#)

```

<xsd:complexType name="TNavigateSubpaths">
  <xsd:sequence>
    <xsd:element name="subpath" type="xs:string"
      minOccurs="1" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

```

subpath

string:

Enthält den Pfad eines Unterverzeichnisses.

Datentyp TNavigateAOIDs

```

<xsd:complexType name="TNavigateAOIDs">
  </xsd:sequence>
  <xsd:element name="aoid" type="TNavigateAOIDItem"
    minOccurs="1" maxOccurs="unbounded" />
</xsd:sequence>
</xsd:complexType>

```

aoid **Datentyp** TNavigateAOIDItem, siehe unten:

Enthält die Beschreibung einer AOID.

Datentyp TNavigateAOIDItem

```

<xsd:complexType name="TNavigateAOIDItem">
  <xsd:sequence>
    <xsd:element name="subject" type="xs:string"
      minOccurs="0" maxOccurs="1">
  </xsd:element>
</xsd:sequence>
<xsd:attribute name="id" use="required" type="TUuid"/>
<xsd:attribute name="coid" use="optional"
  type="TCoid"/>
</xsd:complexType>

```

subject string:

Zeigt abhängig vom Wert des Attributs useDatabase (siehe "[useDatabase](#)") den Inhalt von \$Subject oder einen Teil dieses Inhalts an.

id string (Universally Unique Identifier gemäß Standard IETF RFC 4122):

Zeigt die AOID an.

coid TCoid:
optional; Zeigt die COID an.
Datentyp TCoid, siehe ["Request-Header"](#)

Beispiel

SOAP-Request

```
<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope

  xmlns:xsd="http://www.w3.org/2001/XMLSchema"

  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"

  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/

                        http://schemas.xmlsoap.org/soap/envelope/"

>

  <soap:Header>
    <secdocs:soapHeaderData>

      <secdocs:security>

        <secdocs:principal>
          <secdocs:role>Archivar</secdocs:role>

          <secdocs:mandant>Mandant1</secdocs:mandant>

          <secdocs:orgID>Department1</secdocs:orgID>
        </secdocs:principal>

        <secdocs:password>secrets2</secdocs:password>
      </secdocs:security>
    </secdocs:soapHeaderData>
  </soap:Header>
</soap:Envelope>
```

```
        </secdocs:security>

        <secdocs:operation>navigate</secdocs:operation>

    </secdocs:soapHeaderData>
</soap:Header>

<soap:Body>

    <tns:navigateRequest
        xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"
    >

        <tns:path><![CDATA[ / ]]></tns:path>

    </tns:navigateRequest>
</soap:Body>

</soap:Envelope>
```

SOAP-Response

```

<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope

  xmlns:xsd="http://www.w3.org/2001/XMLSchema"

  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"

  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"

  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Header>

    <secdocs:soapHeaderData>

      <secdocs:security>
        <secdocs:principal>

          <secdocs:role>Archivar</secdocs:role>

          <secdocs:mandant>Mandant1</secdocs:mandant>
          <secdocs:orgID>Department1</secdocs:orgID>

        </secdocs:principal>

        <secdocs:password>secrets2</secdocs:password>
      </secdocs:security>

      <secdocs:operation>navigate</secdocs:operation>

    </secdocs:soapHeaderData>
  </soap:Header>

  <soap:Body>

    <tns:navigateResponse
      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"
    >

      <tns:path><![CDATA[ / ]]></tns:path>

      <tns:queryOptions>
        <tns:subpaths>true</tns:subpaths>

        <tns:aoids>true</tns:aoids>

        <tns:subjects>true</tns:subjects>
      </tns:queryOptions>

      <tns:maxResults>50</tns:maxResults>

      <tns:totalResults>3</tns:totalResults>
      <tns:subpaths>

        <tns:subpath><![CDATA[ /MyDocumentNone/ ]]></tns:subpath>

```

```
</tns:subpaths>

<tns:aoids>
  <tns:aoid id="3ea6f714-f6e9-46bc-b262-0e4a2adeca35">
    <tns:subject>Description of SDO content</tns:
subject>
  </tns:aoid>
  <tns:aoid id="bc3c9f17-9324-4ede-b14e-fbd17ae9d0b1"/>
</tns:aoids>

</tns:navigateResponse>

</soap:Body>
</soap:Envelope>
```

3.3.14 Operation getAccountingData

getAccountingData stellt dem Web-Service ArchivingService Informationen für die Abrechnung zur Verfügung. Folgendes wird ausgegeben:

- Die Anzahl der versiegelten Dokumente zum Zeitpunkt der Abfrage für die im Request-Header angegebene Organisationseinheit
- Die Anzahl der bisher durchgeführten Versiegelungsvorgänge für die im Request-Header angegebene Organisationseinheit

Eine ausführliche Beschreibung des Accounting finden Sie im [Abschnitt „Accounting“](#).

Request-Body

Datentyp TGetAccountingDataRequest

```
<xsd:complexType name="TGetAccountingDataRequest">
  <xs:annotation>
    <xs:documentation>SecDocs SOAP body request data for the
                        WS operation getAccountingData
    </xs:documentation>
  </xs:annotation>
</xsd:complexType>
```

Response-Body

Datentyp GetAccountingDataMandantAdmResponseType

```
<xs:complexType name="TGetAccountingDataResponseType">
  <xs:sequence>
    <xs:element name="NumberOfSealedDocuments" type="xs:long"></xs:element>
    <xs:element name="NumberOfSealingActions" type="xs:long"></xs:element>
  </xs:sequence>
</xsd:complexType>
```

NumberOfSealedDocuments

long:

enthält die Anzahl der versiegelten Dokumente im Archiv für die im Request-Header angegebene Organisationseinheit.

NumberOfSealingActions

long:

enthält die Anzahl der Versiegelungsvorgänge für die im Request-Header angegebene Organisationseinheit.

Beispiel

SOAP-Request

```

<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/07/secext"
  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
    http://schemas.xmlsoap.org/soap/envelope/"
>

  <soap:Header>
    <secdocs:soapHeaderData>

      <secdocs:operation>getAccountingData</secdocs:operation>

      <secdocs:security>
        <secdocs:principal>

          <secdocs:role>Archivar</secdocs:role>

          <secdocs:mandant>Mandant1</secdocs:mandant>
          <secdocs:orgID>Department11</secdocs:orgID>

        </secdocs:principal>

        <secdocs:password>secretAR</secdocs:password>
      </secdocs:security>

      <secdocs:auditID>Administrator XY</secdocs:auditID>

    </secdocs:soapHeaderData>
  </soap:Header>

  <soap:Body>
    <tns:getAccountingDataRequest

      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"

    />

  </soap:Body>
</soap:Envelope>

```

SOAP-Response

```
<?xml version="1.0" encoding="UTF-8"?>

  <soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:sdsh="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
      http://schemas.xmlsoap.org/soap/envelope/">

    <soap:Header>

      <secdocs:soapHeaderData>
        <sdsh:security>
          <sdsh:principal>
            <sdsh:role>Archivar</sdsh:role>
            <sdsh:mandant>Mandant1</sdsh:mandant>
            <sdsh:orgID>Department11</sdsh:orgID>
          </sdsh:principal>
          <sdsh:password>*****</sdsh:password>
        </sdsh:security>
        <sdsh:auditID>Administrator XY</sdsh:auditID>
        <sdsh:operation>getAccountingData</sdsh:operation>
      </sdsh:soapHeaderData>
    </soap:Header>

    <soap:Body>

      <getAccountingDataResponse
        xmlns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
        <NumberOfSealedDocuments
          xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
          626258</NumberOfSealedDocuments>
          <NumberOfSealingActions
            xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
            716251</NumberOfSealingActions>
          </getAccountingDataResponse>
        </soap:Body>
      </soap:Envelope>
```

3.3.15 Operation moveSDO

Mit `moveSDO` kann ein bereits archiviertes SDO zu einem anderen Ablageort in der Verzeichnisstruktur der Organisation verschoben werden. Dabei bleibt der Beweiswert erhalten. Es wird ein neuer `SDOPath` festgelegt.

Die Operation `moveSDO` wird für folgende Arten von SDOs abgewiesen:

- Für SDOs, die mit `$Softworm=YES` archiviert wurden.
- Für SDOs, die externe Referenz-IDs enthalten.

Die Operation `moveSDO` wird mit Fehler `REQ0059` abgewiesen, wenn für die angegebene AOID gleichzeitig ein Versiegelungsvorgang durchgeführt wird.

Request-Body

Datentyp `TMoveSDORequest`

```
<xsd:complexType name="TMoveSDORequest">
  <xsd:sequence>
    <xsd:element name="newPath" type=" tns:string"
      minOccurs="1" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

`newPath`

string:

Pfadname

Der angegebene Pfadname muss mit einem Schrägstrich (/) beginnen.

Sonderzeichen werden in der gleichen Weise umgesetzt, wie sie umgesetzt werden, wenn der Wert von `$SDOPath` durch die Angabe von `"xpath="` in der Filterdefinition direkt aus dem SDO geholt wird (siehe [Abschnitt „\\$SDOPath“](#)).

Response-Body

Leeres Element `TMoveSDOResponse`

```
<TMoveSDOResponse ></TMoveSDOResponse >
```

Beispiel

SOAP-Request

```
<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Header>

    <secdocs:soapHeaderData>
      <secdocs:security>

        <secdocs:principal>

          <secdocs:role>Archivar</secdocs:role>
          <secdocs:mandant>Mandant1</secdocs:mandant>

          <secdocs:orgID>Department1</secdocs:orgID>

        </secdocs:principal>
        <secdocs:password>secrets2</secdocs:password>

      </secdocs:security>

      <secdocs:operation>moveSDO</secdocs:operation>

      <secdocs:aoid>144ed128-cceb-4d19-9459-ba08d6a406e7</secdocs:
aoid>
    </secdocs:soapHeaderData>

  </soap:Header>

  <soap:Body>
    <tns:moveSDORequest

      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">

      <tns:newPath>newSDOPath</tns:newPath>
    </tns:moveSDORequest>

  </soap:Body>

</soap:Envelope>
```

SOAP-Response

```
<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Header>

    <secdocs:soapHeaderData>
      <secdocs:security>

        <secdocs:principal>

          <secdocs:role>Archivar</secdocs:role>
          <secdocs:mandant>Mandant1</secdocs:mandant>

          <secdocs:orgID>Department1</secdocs:orgID>

        </secdocs:principal>
        <secdocs:password>secrets2</secdocs:password>

      </secdocs:security>

      <secdocs:operation>moveSDO</secdocs:operation>

      <secdocs:aoid>144ed128-cceb-4d19-9459-ba08d6a406e7</secdocs:
aoid>
    </secdocs:soapHeaderData>

  </soap:Header>

  <soap:Body>
    <tns:moveSDOResponse

      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">

    </tns:moveSDOResponse>
  </soap:Body>

</soap:Envelope>
```

3.3.16 Operation setExpirationDateTimeSDO

Mit `setExpirationDateTimeSDO` können Sie für eine AOID der Organisation den endgültigen Freigabezeitpunkt festlegen. Falls bereits ein Freigabezeitpunkt ungleich `DEFERRED` festgelegt ist, darf der neue Freigabezeitpunkt nicht kleiner als der aktuelle Freigabezeitpunkt sein.

Mögliche Einsatzfälle

Es gibt folgende Einsatzfälle, wie diese Operation verwendet werden kann:

- Der Freigabezeitpunkt für das SDO ist auf `DEFERRED` gesetzt, entweder zum Archivierungszeitpunkt auf Grund entsprechender Filter-Einstellungen oder durch vorherige Operation `forceDeferredSDO`.

Hier dürfen Sie für den neu zu setzenden Freigabezeitpunkt einen beliebigen Wert festlegen, unabhängig von dem Freigabezeitpunkt, der durch eine vorherige Operation `forceDeferredSDO` aufgehoben wurde.

- Durch `setExpirationDateTimeSDO` ohne explizite Zeitangabe wird der Freigabezeitpunkt auf den aktuellen Wert von `today+$RetentionPeriod` gesetzt.
- Durch `setExpirationDateTimeSDO` mit expliziter Zeitangabe wird der Freigabezeitpunkt auf diesen gewünschten Wert gesetzt.
- Der Freigabezeitpunkt für das SDO ist nicht auf `DEFERRED` gesetzt.

Der neu zu setzende Freigabezeitpunkt muss entweder gleich dem aktuellen Freigabezeitpunkt oder größer sein.

- Durch `setExpirationDateTimeSDO` ohne explizite Zeitangabe wird der Freigabezeitpunkt auf den aktuellen Wert von `today+$RetentionPeriod` gesetzt.
- Durch `setExpirationDateTimeSDO` mit expliziter Zeitangabe wird der Freigabezeitpunkt auf diesen gewünschten Wert gesetzt.

Request-Body

Datentyp `TSetExpirationDateTimeSDORequest`

```
<xs:complexType name="TSetExpirationDateTimeSDORequest">
  <xs:annotation>
    . . .
  </xs:annotation>
  <xs:choice minOccurs="0" maxOccurs="1">
    <xs:element name="expirationDateTime" type="xs:dateTime"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="expirationDateUnlimited" type="TEmptyElement"
      minOccurs="1" maxOccurs="1"/>
  </xs:choice>
</xs:complexType>
```

Falls Sie keinen Wert angeben, wird der endgültige Freigabezeitpunkt auf `today+$RetentionPeriod` gesetzt.

`expirationDateTime`

`dateTime:`

Endgültiger Freigabezeitpunkt des Dokuments (muss im Format `ISO 8601` angegeben werden).

Der Wert muss gleich oder später als `today+$RetentionPeriod` sein, falls der aktuelle Freigabezeitpunkt nicht den Wert `DEFERRED` besitzt.

`expirationDateUnlimited`

Datentyp `TEmptyElement`, siehe "[Operation statusSDO](#)":

Der endgültige Freigabezeitpunkt des Dokuments wird auf den Wert `UNLIMITED` gesetzt.

Response-Body

Leeres Element `TSetExpirationDateTimeSDOResponse`

```
<TSetdateExpirationDateTimeSDOResponse></TSetExpirationDateTimeSDOResponse>
```

Beispiel

SOAP-Request

```
<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Header>

    <secdocs:soapHeaderData>
      <secdocs:security>

        <secdocs:principal>

          <secdocs:role>Archivar</secdocs:role>
          <secdocs:mandant>Mandant1</secdocs:mandant>

          <secdocs:orgID>Department1</secdocs:orgID>

        </secdocs:principal>
        <secdocs:password>secrets2</secdocs:password>

      </secdocs:security>

      <secdocs:operation>setExpirationDateTimesDO</secdocs:operation>

      <secdocs:aoid>144ed128-cceb-4d19-9459-ba08d6a406e7</secdocs:
aoid>
    </secdocs:soapHeaderData>

  </soap:Header>

  <soap:Body>
    <tns:setExpirationDateTimesDORequest

      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">

      <secdocs:expirationDateTime>2020-12-31T23:59:59
      </secdocs:expirationDateTime>

    </tns:setExpirationDateTimesDORequest>

  </soap:Body>
</soap:Envelope>
```

SOAP-Response

```
<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Header>

    <secdocs:soapHeaderData>
      <secdocs:security>

        <secdocs:principal>

          <secdocs:role>Archivar</secdocs:role>
          <secdocs:mandant>Mandant1</secdocs:mandant>

          <secdocs:orgID>Department1</secdocs:orgID>

        </secdocs:principal>
        <secdocs:password>secrets2</secdocs:password>

      </secdocs:security>

      <secdocs:operation>setExpirationDateTimesDO</secdocs:operation>

      <secdocs:aoid>144ed128-cceb-4d19-9459-ba08d6a406e7</secdocs:
aoid>
    </secdocs:soapHeaderData>

  </soap:Header>

  <soap:Body>
    <tns:setExpirationDateTimesSDOResponse

      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">

    </tns:setExpirationDateTimesSDOResponse>
  </soap:Body>

</soap:Envelope>
```

3.3.17 Operation forceDeferredSDO

Mit `forceDeferredSDO` können Sie für eine AOID den endgültigen Freigabezeitpunkt wieder aufheben und den Mechanismus zu seiner fortwährenden Neuberechnung neu anstarten.

Beim entsprechenden SDO muss für `$RetentionPeriod` eine gültige Aufbewahrungszeit eingetragen sein. Werte wie „“, „POD“, „POM“ oder „POY“ dürfen nicht vorkommen.

Wird bei `forceDeferredSDO` ein Wert für `retentionPeriod` angegeben, dann muss für `$RetentionPeriod` kein gültiger Wert vorhanden sein.

i Die Operation `forceDeferredSDO` wird mit Fehler REQ0059 abgewiesen, wenn für die angegebene AOID gleichzeitig ein Versiegelungsvorgang durchgeführt wird.

Request-Body

Datentyp `TForceDeferredSDORequest`

```
<xs:complexType name="TForceDeferredSDORequest">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="reason" type="TNonEmptyString"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="retentionPeriod" type="xs:duration"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

`reason`

`TNonEmptyString`:

Grund für die Aufhebung des endgültigen Freigabezeitpunkts.

`retentionPeriod`

`duration`:

Mit dieser Angabe wird der Aufbewahrungszeitraum geändert bzw. neu gesetzt, falls er bei `submitSDO` oder `replaceSDO` noch nicht gesetzt wurde.

Für `duration` darf nur eine Zeitspanne angegeben werden. Ebenso wie im Filter für `$RetentionPeriod` (siehe "[\\$RetentionPeriod](#)") ist hier nur folgende Syntax erlaubt:

`PyearYmonthMdayD`

Eine bei `forceDeferredSDO` angegebene `retentionPeriod` überschreibt den Wert einer evtl. bereits existierenden `retentionPeriod`.

Response-Body

Leeres Element `TForceDeferredSDOResponse`

```
<TForceDeferredSDOResponse></TForceDeferredSDOResponse>
```

Beispiel

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
    http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Header>
    <secdocs:soapHeaderData>
      <secdocs:security>
        <secdocs:principal>
          <secdocs:role>Archivar</secdocs:role>
          <secdocs:mandant>Mandant1</secdocs:mandant>
          <secdocs:orgID>Department1</secdocs:orgID>
        </secdocs:principal>
        <secdocs:password>*****</secdocs:password>
      </secdocs:security>
      <secdocs:auditID>Administrator XY</secdocs:auditID>
      <secdocs:operation>forceDeferredSDO</secdocs:operation>
      <secdocs:aoid>07d811fe-9d5b-432e-ba20-bdb5f2711ae7</secdocs:aoid>
    </secdocs:soapHeaderData>
  </soap:Header>
  <soap:Body>
    <tns:forceDeferredSDORequest
      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
      <tns:reason>Set ExpirationDate to DEFERRED: 2020-04-18;
        08:00</tns:reason>
    </tns:forceDeferredSDORequest>
  </soap:Body>
</soap:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:sdsh="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
    http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <sdsh:soapHeaderData>
      <sdsh:security>
        <sdsh:principal>
          <sdsh:role>Archivar</sdsh:role>
          <sdsh:mandant>Mandant1</sdsh:mandant>
          <sdsh:orgID>Department1</sdsh:orgID>
        </sdsh:principal>
        <sdsh:password>*****</sdsh:password>
      </sdsh:security>
      <sdsh:auditID>Administrator XY</sdsh:auditID>
      <sdsh:operation>forceDeferredSDO</sdsh:operation>
      <sdsh:aoid>07d811fe-9d5b-432e-ba20-bdb5f2711ae7</sdsh:aoid>
    </sdsh:soapHeaderData>
  </soap:Header>
  <soap:Body>
    <tns:forceDeferredSDOResponse
      xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
  </soap:Body>
</soap:Envelope>
```

3.3.18 Operation getSchemaDataSDO

getSchemaDataSDO liefert für die im Header der SOAP-Message angegebene AOID den Namen des verwendeten SDO-Typs, das zugehörige XML-Schema und den zugehörigen Filter zurück. Verweist das Hauptschema auf weitere Schemas, so werden diese Subschemas ebenfalls ausgegeben. Es werden die Definitionen für Filter und Schema(s) ausgegeben, die zum Zeitpunkt der Archivierung des Dokuments gültig waren. Diese können von den aktuellen Einstellungen für den SDO-Typ abweichen, falls sie mit der Operation modifySDOType modifiziert worden sind.

Request-Body

Leeres Element vom Datentyp TGetSchemaDataSDORequest

```
<tns:getSchemaDataSDORequest  
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Element getSchemaDataSDOResponse *vom Datentyp* TGetSchemaDataSDOResponse

```
<xs:complexType name="TGetSchemaDataSDOResponse">  
  <xs:annotation>  
    <xs:documentation>SecDocs SOAP body response data  
                        for the WS operation getSchemaDataSDO  
    </xs:documentation>  
  </xs:annotation>  
  <xs:sequence>  
    <xs:element name="sdoType" type="TNonEmptyString"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="sdoTypeReplaceable" type="xs:boolean"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="filterData" type="xs:base64Binary"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="mainSchemaData" type="xs:base64Binary"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="subSchemas" type="TSubSchemas"  
      minOccurs="0" maxOccurs="1"/>  
    <xs:element name="policyData" type="xs:base64Binary"  
      minOccurs="0" maxOccurs="1"/>  
    <xs:element name="timestampPolicyData" type="xs:base64Binary"  
      minOccurs="0" maxOccurs="1"/>  
  </xs:sequence>  
</xs:complexType>
```

sdoType

TNonEmptyString:
Name des verwendeten SDO-Typs

sdoTypeReplaceable

boolean:

gibt an, ob SDOs, die mit diesem SDO-Typ erzeugt wurden, überschrieben werden können.

Mögliche Werte:

"true" das SDO kann überschrieben werden

"false" das SDO kann nicht überschrieben werden

filterData

base64Binary:

zugehöriger Filter

mainSchemaData

base64Binary:

zugehöriges Schema

subSchemas

Datentyp `TSubSchemas`, siehe "[Operation getSchemaDataSDO](#)".

optional; weitere Schemas, falls solche von dem unter `mainSchemaData` genannten Hauptschema referenziert werden.

Datentyp `TSubSchemas`

```
<xs:complexType name="TSubSchemas">
  <xs:annotation>
    <xs:documentation>SecDocs schema data for a given AOID
  </xs:documentation>
</xs:annotation>
<xs:sequence>
  <xs:element name="subSchema" type="TSubSchema"
    minOccurs="1" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>
```

subSchema

Datentyp `TSubSchema`, siehe "[Operation getSchemaDataSDO](#)".

Enthält ein Element `subSchema` für jedes Schema, auf das im Hauptschema oder in einem Unterschema mit `import`, `include` oder `redefine` verwiesen wird.

Datentyp `TSubSchema`

```

<xs:complexType name="TSubSchema">
  <xs:annotation>
    <xs:documentation>SecDocs schema data for a given AOID
  </xs:documentation>
</xs:annotation>
  <xs:sequence>
    <xs:element name="location" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="namespace" type="xs:string"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="data" type="xs:base64Binary"
      minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>

```

location

string:

optional; Pfad, der für das Subschema bei der Registrierung angegeben wurde. (siehe Operand SchemaLocation bei der Operation `createSDOType` auf ["Operation createSDOType"](#)).

namespace

string:

optional; Namespace, der für das Subschema bei der Registrierung angegeben wurde (siehe Operand Namespace bei der Operation `createSDOType` auf ["Operation createSDOType"](#)).

data

base64Binary:

Inhalt des Subschemas

policyData

base64Binary:

Datei zur Steuerung der Bewertung des Ergebnisses der Prüfung von eingebetteten oder abgesetzten Signaturen bei ArchiveSubmission.

timestampPolicyData

base64Binary:

Datei zur Bewertung der Prüfung, wenn Zeitstempel als Signaturen mitgegeben werden

Beispiel

SOAP-Request-Body

```
<tns:getSchemaDataSDORequest
```

```
xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

SOAP-Response-Body

```
<tns:getSchemaDataSDOResponse
  xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
<tns:sdoType>MyDocumentNone</tns:sdoType>

<tns:sdoTypeReplaceable>false</tns:sdoTypeReplaceable>

<tns:filterData>PD94bWwgdml...lsdGVyPgo=</tns:filterData>
<tns:mainSchemaData>PD94bWwgdml...NjaGVtYT4K</tns:
mainSchemaData>

</tns:getSchemaDataSDOResponse>
```

3.3.19 Operation getVersion

getVersion liefert die aktuelle SecDocs-Version.

Request-Body

Leeres Element getRequest

```
<tns:getVersionRequest  
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving"/>
```

Response-Body

Element getVersionResponse vom Datentyp TGetVersionResponse

```
<xs:complexType name="TGetVersionResponse">  
  <xs:annotation>  
    ...  
  </xs:annotation>  
  <xs:sequence>  
    <xs:element name="component" type="tns:TComponentVersion"  
      minOccurs="1" maxOccurs="unbounded"/>  
    <xs:element name="nodeName" type="tns:TNonEmptyString"  
      minOccurs="0" maxOccurs="1"/>  
  </xs:sequence>  
</xs:complexType>
```

component

Information über SecDocs

Datentyp TComponentVersion, siehe unten.

nodeName

Name des Knotens, auf dem die Operation ausgeführt wurde.

Das Element wird nur ausgegeben, wenn der Konfigurationsparameter nodeNameInFaultMessage in der Datei secdocs.properties auf den Wert "true" gesetzt ist.

Element component vom Datentyp TComponentVersion

```
<xs:complexType name="TComponentVersion">  
  <xs:sequence>  
    <xs:element name="name" type="xs:string"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="version" type="xs:string"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="major" type="xs:integer"  
      minOccurs="1" maxOccurs="1"/>  
    <xs:element name="minor" type="xs:integer"  
      minOccurs="1" maxOccurs="1"/>  
  </xs:sequence>  
</xs:complexType>
```

name	string: Name der Komponente
version	string: vollständige Version in der Form: Name VMajorNumber.MinorNumber[A-Z]BuildNumber <i>Beispiel:</i> SecDocs V3.0A01
major	integer: Major-Nummer der Version
minor	integer: Minor-Nummer der Version

Beispiel

Request-Body

```
<soap:Body>
  <tns:getVersionRequest
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving" />
</soap:Body>
```

Response-Body

```
<soap:Body>
  <tns:getVersionResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/archiving">
    <tns:component>
      <tns:name>SecDocs</tns:name>
      <tns:version>SecDocs V3.0A01</tns:version>
      <tns:major>3</tns:major>
      <tns:minor>0</tns:minor>
    </tns:component>
    <tns:nodeName>MyNode2</tns:nodeName>
  </tns:getVersionResponse>
</soap:Body>
```

3.4 Tools

- Hash-Wert erzeugen (Skript mksha)

3.4.1 Hash-Wert erzeugen (Skript mksha)

Mit Hilfe des SecDocs-Skripts `mksha` können Sie in der Linux-Shell für eine vorgegebene Datei einen Hash-Wert erzeugen. Der erzeugte Hash-Wert wird sowohl als hexadezimaler Wert als auch BASE64-codiert ausgegeben. Der Hash-Wert kann optional auch als Datei abgespeichert werden.

Aufruf des Skripts `mksha`

Das Skript `mksha` wird folgendermaßen aufgerufen:

```
mksha [-1 | -224 | -256 | -384 | -512] [-d] <file> [<hash data file>]
```

`-1 | -224 | -256 | -384 | -512`

Der Hash-Wert wird mit dem entsprechenden Hash-Algorithmus, z.B. SHA-512, erzeugt. Standardmäßig wird der Hash-Algorithmus SHA-256 verwendet.

`-d`

Es wird eine Ausgabedatei mit dem binären Hash-Wert erzeugt. Wird kein Dateiname für die Ausgabedatei `<hash data file>` angegeben so wird der Ausgabedateiname nach folgendem Schema erstellt: `<file_basename>.<hash_name>.dat`.

`<file>`

Name der Datei, für die ein Hash-Wert erzeugt werden soll.

`<hash data file>`

Name der Datei, die den binären Hash-Wert enthält.

Die Datei wird in dem Verzeichnis abgelegt, in dem der Aufruf von `mksha` erfolgte. Die Angabe wird ignoriert, wenn die Option `-d` nicht angegeben wurde.

Aufrufbeispiele:

`mksha sourcefile1`

SHA-256 hex value

```
e12e115acf4552b2568b55e93cbd39394c4ef81c82447fafc997882a02d23677
```

SHA-256 hex value in BASE64

```
4S4RWs9FUrJWilXpPL05OUxO+ByCRH+vyZeIKgLSNnc=
```

```
mksha -384 -d /home/secdocs/sourcefile2
```

```
SHA-384 hex value
```

```
6f17e23899d2345a156baf69e7c02bbdda3be057367849c02add6a4aecbbd039a660ba815c95f  
2f145883600b7e9133d
```

```
SHA-384 hex value in BASE64
```

```
bxfiOJnSNFoVa69p58Arvdo74Fc2eEnAKt1qSuy70DmmYLqBXJXy8UWINGC36RM9
```

Es wird die Datei `sourcefile2.SHA-384.dat` mit folgendem binären Inhalt erzeugt:

```
6f17e23899d2345a156baf69e7c02bbdda3be057367849c02add6a4aecbbd039a660ba815c95f  
2f145883600b7e9133d.
```

```
mksha -224 -d sourcefile2 hashfile2.224.dat
```

```
SHA-224 hex value
```

```
44a1d724940a36e1b7adee6b5c7bafab2368bb02ddda3c90d74f60b6
```

```
SHA-224 hex value in BASE64
```

```
RKHXJJQKNuG3re5rXHuvqyNouwLd2jyQ109gtg==
```

Es wird die Datei `hashfile2.224.dat` erzeugt.

4 Beschreibung der Datenobjekte

Abhängig von den Bedürfnissen des Anwenders werden in einem Langzeitarchiv Dokumente unterschiedlicher Typen archiviert, z.B. Rechnungen, Urkunden, Verträge, usw. Der Dokumententyp, also Art, Struktur und beschreibende Metadaten des zu archivierenden Schriftguts, muss vor der Archivierung als „SDO-Typ“ bei SecDocs bekanntgemacht werden.



Welche Punkte Sie bei der Strukturierung Ihrer Daten beachten sollten, ist im [Abschnitt „Beschreibung eines Datenobjekts“](#) beschrieben.

Zum Dokumententyp gehören

- die Informationen über die Struktur des zu archivierenden Schriftguts (siehe [Abschnitt „SDO-Struktur“](#),
- die zugehörige Information zur Adressierung für (Nutz-)Daten, Signaturen und Metadaten (siehe [Abschnitt „Adressierung von Daten im SDO“](#)).

Die zu archivierenden Objekte eines bestimmten Dokumententyps werden SecDocs dann in Form eines XML-Containers, dem SDO, zur Archivierung übergeben (Operation `submitSDO`, siehe "[Operation submitSDO](#)" und Operation `replaceSDO`, siehe "[Operation replaceSDO](#)"). Der XML-Container enthält neben den Dokumenten selbst (den Primärdokumenten) ggf. auch Signaturen und Metadaten.

Bevor ein Datenobjekt zur Archivierung an SecDocs übergeben werden kann, müssen folgende Schritte durchgeführt werden:

1. Den Aufbau des SDOs festlegen, siehe [Abschnitt „Beschreibung eines Datenobjekts“](#).
2. Die Struktur des SDOs als XML-Schema beschreiben, siehe [Abschnitt „SDO-Struktur“](#).
3. Die zugehörige Information zur Adressierung von Daten im SDO für (Nutz-)Daten, Signaturen und Metadaten beschreiben, siehe [Abschnitt „Adressierung von Daten im SDO“](#).
4. Den Datentyp bei SecDocs bekanntmachen, d.h. XML-Schema(s) und Adressierungs-Information mit Hilfe des Web-Service `MandantAdminService` registrieren, siehe [Abschnitt „SDO-Typ registrieren und Organisationseinheit anlegen“](#).
5. Ggf. die Schnittstellendefinition des Web-Service für die archivierende Client-Software bereitstellen, siehe [Abschnitt „SDO-Typ-spezifische Operationen des Web-Service ArchivingSRService“](#).

Dieses Kapitel beschreibt die Schritte [2](#) und [3](#), d.h. den Aufbau einer SDO-Struktur als XML-Schema und das Festlegen der Informationen zur Adressierung von Daten im SDO in einem Filter.

4.1 SDO-Struktur

Nachdem festgelegt wurde, welche Primärdokumente und/oder Metadaten zusammen in einem SDO archiviert werden sollen, müssen Sie die Struktur des SDOs in einem XML-Schema beschreiben. Diese Struktur wird beim Registrieren des SDO-Typs an SecDocs übergeben (Operation `createSDOType`, siehe "[Operation createSDOType](#)"). Zum SDO-Typ gehört neben der Struktur des SDOs die Information zur Adressierung von Daten im SDO, die in der zugehörigen Filterdefinition abgelegt ist (siehe [Abschnitt „Adressierung von Daten im SDO“](#)).

Angaben im XML-Schema

Folgende Angaben im Schema sind notwendig. Berücksichtigen Sie dabei auch die Filterfunktionalität zur Adressierung von Daten im SDO (siehe [Abschnitt „Adressierung von Daten im SDO“](#)).

- > Wenn Primärdokumente und Signaturen vorhanden sind und diese geprüft werden sollen, müssen Sie eindeutig beschreiben, welche abgesetzte(n) Signatur(en) zu einem Primärdokument gehören. Dies muss bei der Schema-Definition berücksichtigt werden, insbesondere dann, wenn eine variable Anzahl von Primärdokumenten und Signaturen im SDO vorkommen können.

i Primärdokumente und Signaturen können Sie nicht über XML-Referenzen verknüpfen.

- > Wenn Sie Primärdokumente mit Signaturen archivieren wollen, müssen Sie im Schema Elemente vorsehen, auf die dann im Filter verwiesen wird.

Sowohl Primärdokumente als auch abgesetzte (detached) Signaturen müssen den Datentyp `type="base64Binary"` haben, da sie beliebige Binärwerte enthalten können.

i Diese Angaben werden benötigt, um die Signaturen beim Einbringen in SecDocs prüfen zu können. Wenn Sie auf die Prüfung der Signaturen verzichten, unterbrechen Sie die zur Beweiswerterhaltung notwendigen Verarbeitungsschritte.

Über die Pflichtangaben hinaus sind, je nach Anwendungsfall, zusätzlich folgende Angaben sinnvoll:

- > Wenn der Name des Primärdokuments nicht verloren gehen soll, muss er in einem eigenen Element abgespeichert werden.
- > Wenn die Primärdokumente ein Ordnungskriterium haben, ist es sinnvoll, hierfür ebenfalls ein Element vorzusehen, wie beispielsweise Aktenzeichen und Versicherungsnummer.

Der folgende Text beschreibt anhand eines einfachen Beispiels den möglichen Aufbau einer SDO-Struktur.

Beispiel für ein XML-Schema

Das XML-Schema soll die Struktur für das Archivieren von Dokumenten beschreiben.

Jedes SDO, in dem ein solches Dokument archiviert wird, enthält das Primärdokument, ggf. eine nicht eingebettete Signatur und ggf. Metadaten, die das Primärdokument beschreiben. Für jedes SDO muss eine Aufbewahrungszeit festgelegt werden.

Das XML-Schema wird in der Datei `MyDocument.xsd` gespeichert.

Die folgende Grafik zeigt die Struktur des XML-Schemas MyDocument:

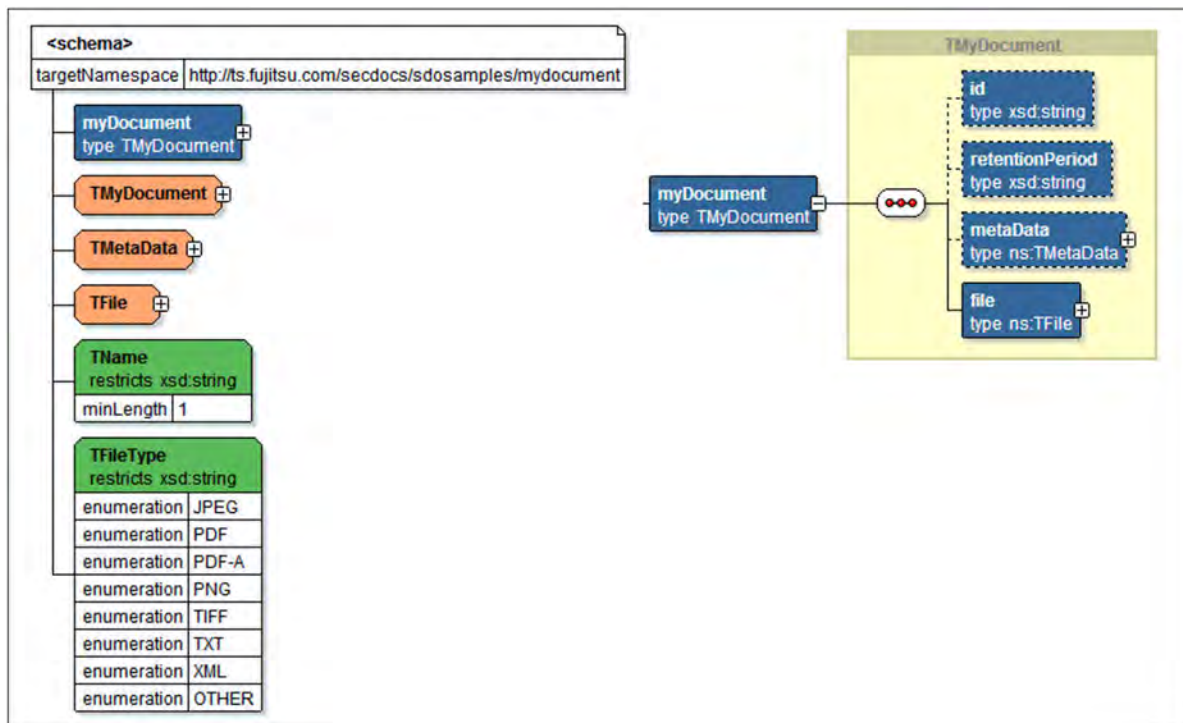


Bild 10: Struktur von MyDocument.xsd

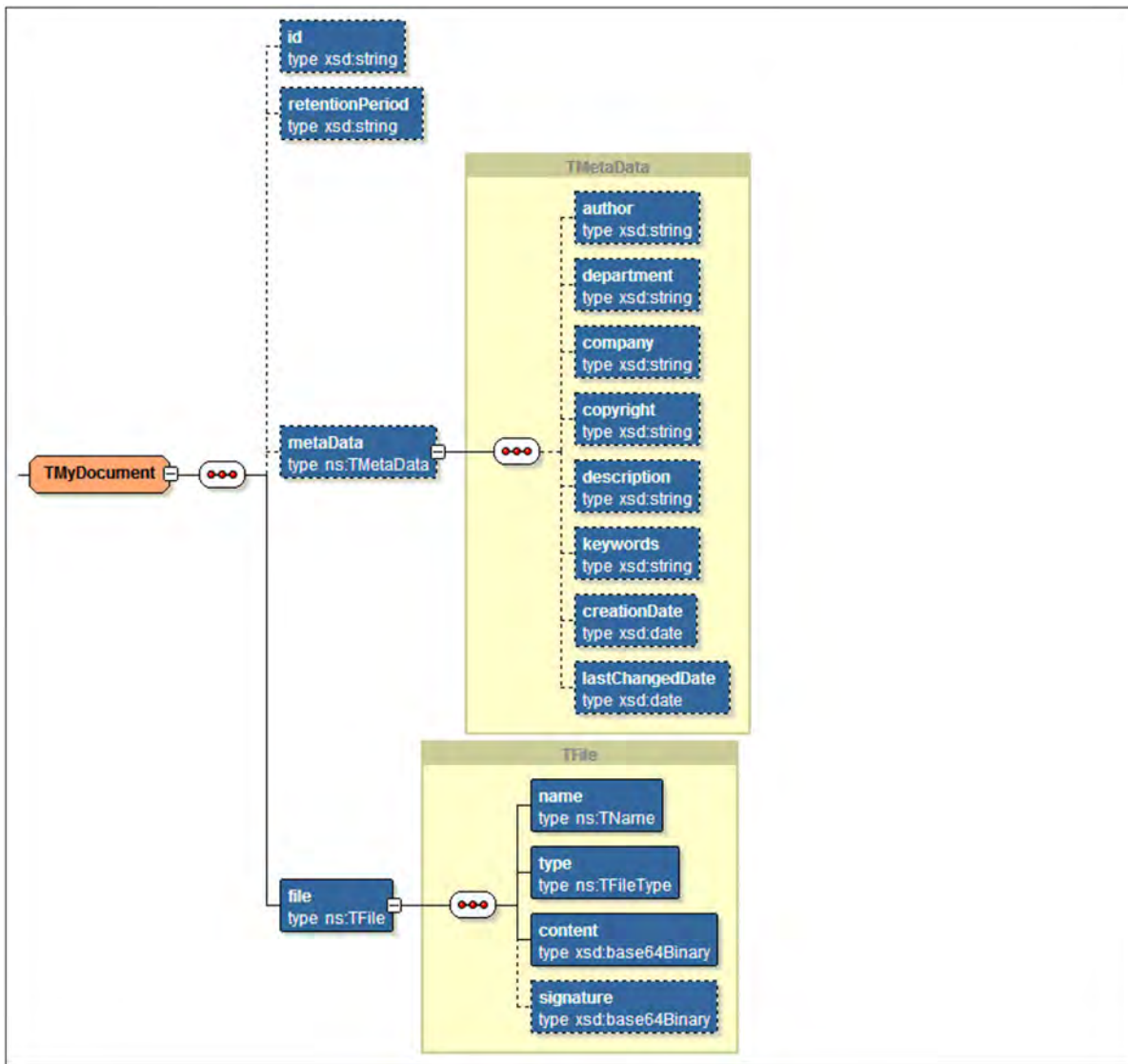


Bild 11: Struktur des Datentyps TMyDocument



Sequence: die Elemente müssen genau in der dargestellten Reihenfolge auftreten.



Choice: eines der angegebenen Elemente



Pflichtangabe



Optionale Angabe

Datei MyDocument.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://ts.fujitsu.com/secdocs/sdosamples/mydocument" ----- (1)
    elementFormDefault="qualified"
    attributeFormDefault="unqualified"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:ns="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"
    targetNamespace="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"
    version="1.1">
...
<xsd:element name="myDocument" type="TMyDocument"> ----- (2)
</xsd:element>
<xsd:complexType name="TMyDocument"> ----- (3)
    <xsd:sequence>
        <xsd:element name="id" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <xsd:element name="retentionPeriod" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <xsd:element name="metaData" type="ns:TMetaData" minOccurs="0"
                                maxOccurs="1"/>
        <xsd:element name="file" type="ns:TFile" minOccurs="1"
                                maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="TMetaData"> ----- (4)
    <xsd:sequence>
        <!-- Creator/Author of the document -->
        <xsd:element name="author" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <!-- Creator's department -->
        <xsd:element name="department" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <!-- Creator's company -->
        <xsd:element name="company" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <!-- Document copyright note -->
        <xsd:element name="copyright" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <!-- Document content description -->
        <xsd:element name="description" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <!-- Document content keywords-->
        <xsd:element name="keywords" type="xsd:string" minOccurs="0"
                                maxOccurs="1"/>
        <!-- Document creation date -->
        <xsd:element name="creationDate" type="xsd:date" minOccurs="0"
                                maxOccurs="1"/>
        <!-- Document date of last change -->
        <xsd:element name="lastChangedDate" type="xsd:date" minOccurs="0"
                                maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>

```

```

<xsd:complexType name="TFile"> ----- (5)
  <xsd:sequence>
    <!-- Name of the document file -->
    <xsd:element name="name" type="ns:TName" minOccurs="1"
      maxOccurs="1"/>

    <!-- Document file type -->
    <xsd:element name="type" type="ns:TFileType" minOccurs="1"
      maxOccurs="1"/>

    <!-- BASE64 coded binary content of the document file -->
    <xsd:element name="content" type="xsd:base64Binary" minOccurs="1"
      maxOccurs="1"/>

    <!-- BASE64 coded binary content of the detached signature file (.p7s file) -->
    <xsd:element name="signature" type="xsd:base64Binary" minOccurs="0"
      maxOccurs="1"/>

  </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="TName"> ----- (6)
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="TFileType"> ----- (7)
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="JPEG"/>
    <xsd:enumeration value="PDF"/>
    <xsd:enumeration value="PDF-A"/>
    <xsd:enumeration value="PNG"/>
    <xsd:enumeration value="TIFF"/>
    <xsd:enumeration value="TXT"/>
    <xsd:enumeration value="XML"/>
    <xsd:enumeration value="OTHER"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:schema>

```

Erläuterung

- (1) Das XML-Schema verwendet den Namensraum mydocument.
- (2) Root-Element des SDOs für Dokumente.
- (3) Elemente im Datentyp TMyDocument; die Hauptelemente eines SDOs für Dokumente sind:
 - optional; Identifier (id)
 - optional; Aufbewahrungszeit (retentionPeriod)
 - optional; Metadaten (metaData). Die Metadaten sind im Datentyp TMetaData abgelegt, siehe (4).
 - Daten der Datei (file). Die Daten der Datei sind im Datentyp TFile abgelegt, siehe (5).
- (4) Metadaten des Primärdokuments, die ggf. zusammen mit dem Primärdokument gespeichert werden sollen. Die Metadaten beschreiben das Primärdokument.

-
- (5) Daten des Primärdokuments; zu diesen Daten gehören:
- Name der Datei (`name`). Der Name ist vom Typ `TName`, siehe (6).
 - Datentyp der Datei (`type`). Mögliche Werte für den Datentyp sind in `TFileType` definiert, siehe (7).
 - Inhalt der Datei (`content`). Der Inhalt ist BASE64-codiert.
 - optional; Signatur (`signature`). Die Signatur ist BASE64-codiert.
- (6) Name der Dokumentdatei; der Name der Datei darf nicht leer sein.
- (7) Liste der unterstützten Dateiformate; andere Formate können mit Hilfe von `OTHER` archiviert werden.



Eine Filterdefinition mit den Informationen zur Adressierung von Daten im SDO für dieses Beispiel finden Sie im [Abschnitt „Beispiel für eine Filterdefinition“](#).

Das zugehörige SDO, das mit der Operation `submitSDO` oder `replaceSDO` archiviert wird, finden Sie im [Abschnitt „Datenobjekt für die Archivierung“](#).

4.2 Adressierung von Daten im SDO

Die Struktur eines SDOs ist in einem oder mehreren verknüpften XML-Schema(s) beschrieben (siehe [Abschnitt „SDO-Struktur“](#)). Diese Struktur wird beim Erstellen des SDO-Typs an SecDocs übergeben (Operation `createSDOType`, siehe "[Operation createSDOType](#)").

Zum SDO-Typ gehören neben der Struktur des SDOs auch die Informationen zum Adressieren von Daten im SDO, die von SecDocs zur Laufzeit benötigt werden. Diese werden in der zugehörigen Filterdefinition abgelegt. In einem Filter wird festgelegt, an welcher Stelle des als XML-Dokument vorliegenden SDOs die für das Archivierungssystem relevanten Daten oder vom Anwender selektierte Metadaten zu finden sind.

Die Filterdefinition liefert

- Information über die Zusammengehörigkeit von Primärdokumenten und Signaturen,
- Adressierungs-Information für die Primärdokumente,
- Adressierungs-Information für Signaturen und Art der Signaturen,
- Adressierungs-Information für Metadaten (z.B. der Freigabezeitpunkt),
- (Default-)Werte für Metadaten, die für diesen SDO-Typ spezifisch sind und beispielsweise für den Ablageort im Archiv oder für die Recherche verwendet werden können.

4.2.1 Filtersyntax

Der Filter muss als XML-Dokument definiert werden, entsprechend dem von SecDocs vorgegebenen XML-Schema `filter.xsd`.

Struktur

Die folgende Grafik zeigt die Struktur eines Filters in SecDocs und die Unterstruktur des Elements `node`:

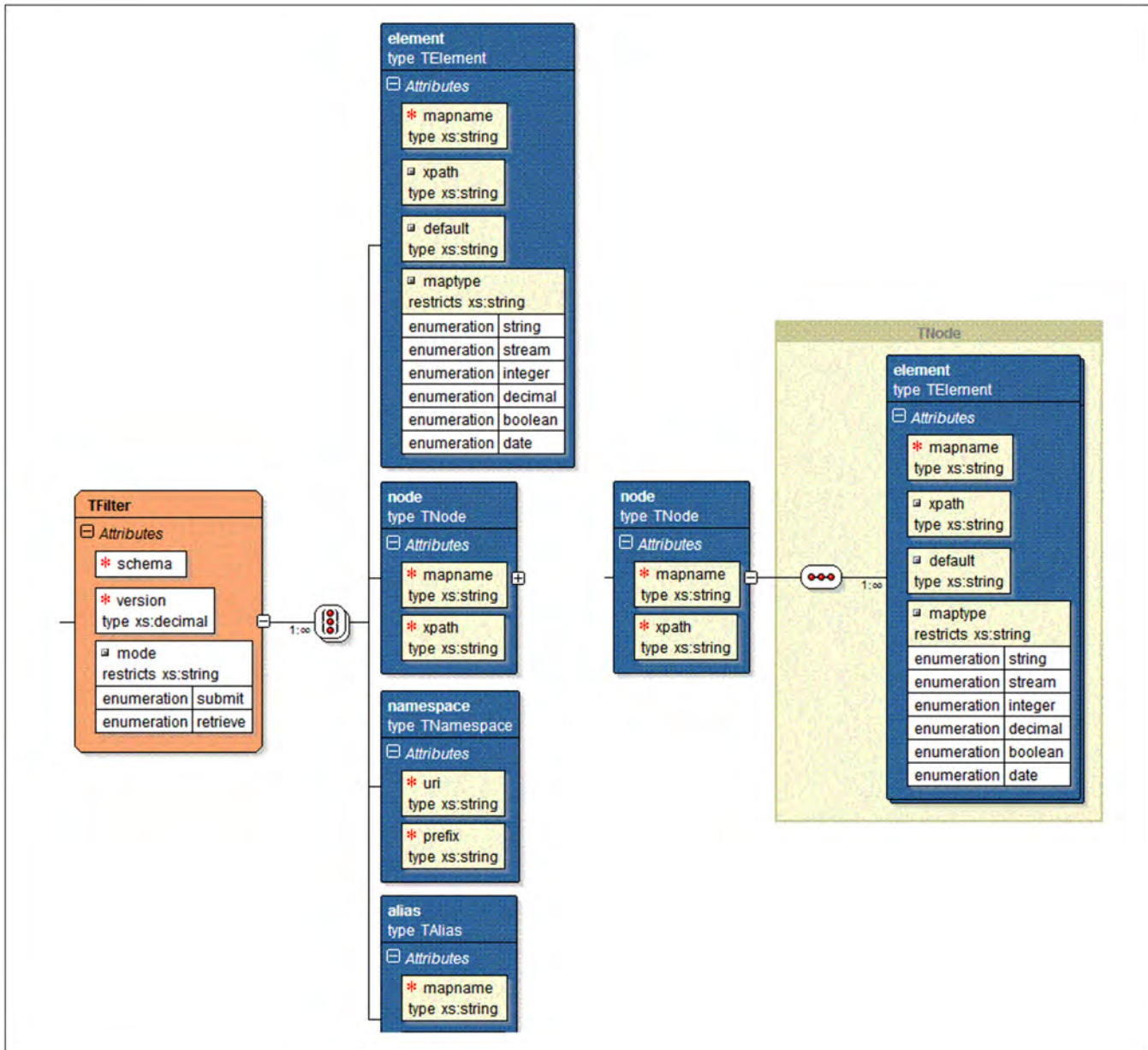


Bild 12: Struktur eines SecDocs-Filters



Sequence: die Elemente müssen genau in der dargestellten Reihenfolge auftreten.



Choice: eines der angegebenen Elemente

In der folgenden Tabelle ist die allgemeine Struktur eines SecDocs-Filters dargestellt. Aufgelistet werden alle Elemente und ihre Pflicht- bzw. optionalen Bestandteile. Die ausführliche Beschreibung der Elemente und Attribute finden Sie in den folgenden Abschnitten.

Element	Pflichtbestandteile	Optionale Bestandteile
<filter>	<ul style="list-style-type: none"> • Attribut <code>schema</code> • Attribut <code>version</code> • Mindestens ein Element <code><element></code> oder <code><node></code> 	<ul style="list-style-type: none"> • Attribut <code>mode</code> • Beliebig viele Elemente <code><namespace></code> • Beliebig viele weitere Elemente <code><element></code> • Beliebig viele weitere Elemente <code><node></code> • Beliebig viele Elemente <code><alias></code>
<namespace>	<ul style="list-style-type: none"> • Attribut <code>prefix</code> • Attribut <code>uri</code> 	
<element>	<ul style="list-style-type: none"> • Attribut <code>mapname</code> 	<ul style="list-style-type: none"> • Attribut <code>maptype</code> • Attribut <code>xpath</code> • Attribut <code>default</code>
<node>	<ul style="list-style-type: none"> • Attribut <code>mapname</code> • Attribut <code>xpath</code> • Mindestens ein Element <code><element></code> 	Beliebig viele weitere Elemente <code><element></code>
<alias>	<ul style="list-style-type: none"> • Attribut <code>mapname</code> • Attribut <code>value</code> • Attribut <code>to</code> 	

Tabelle 1: Filterelemente und ihre Bestandteile

4.2.1.1 Element <filter>

<filter> ist das Wurzelement der Filterdefinition.

```
<xs:element name="filter" type="TFilter"/>
<xs:complexType name="TFilter">
  <xs:choice maxOccurs="unbounded">
    <xs:element name="element" type="TElement"/>
    <xs:element name="node" type="TNode"/>
    <xs:element name="namespace" type="TNamespace"/>
    <xs:element name="alias" type="TAlias"/>
  </xs:choice>
  <xs:attribute name="schema" use="required"/>
  <xs:attribute name="version" type="xs:decimal" use="required"/>
  <xs:attribute name="mode" use="optional" default="submit">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="submit"/>
        <xs:enumeration value="retrieve"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>
```

Elemente

- | | |
|-----------|--|
| element | XML-Element, das eine bestimmte Information zur Adressierung aus dem SDO herausfiltert oder eine Zusatzinformation enthält.
Ein oder mehrere Datentypen <code>TElement</code> , siehe " Element <element> ". |
| node | XML-Element, mit dessen Hilfe andere XML-Elemente oder Zusatzinformationen gruppiert werden können.
Kein, ein oder mehrere Datentypen <code>TNode</code> , siehe " Element <node> ". |
| namespace | optional; Kurzbezeichnung für einen Namespace.
Beliebig viele Datentypen <code>TNamespace</code> , siehe " Element <namespace> ". |
| alias | optional; Wert eines Schlüsselworts im SDO, der zum Zeitpunkt der Operation <code>submitSDO</code> oder <code>replaceSDO</code> auf einen neuen Wert abgebildet werden soll.
Beliebig viele Datentypen <code>TAlias</code> , siehe " Element <alias> ". |

Attribute

- | | |
|--------|--|
| schema | Root-Element desjenigen XML-Schemas, das die Struktur des zugehörigen SDO-Typs beschreibt. |
|--------|--|

`version` SecDocs-interne Versionsnummer des Schemas für die Filterdefinition.

Mögliche Werte: 1.0

i Derzeit kann nur der Wert 1.0 angegeben werden. Ein anderer Wert für `version` wird bei der Registrierung des SDO-Typs (Operation `createSDOType`, siehe "[Operation createSDOType](#)") abgewiesen.

`mode` optional; gibt an, ob der Filter für das Speichern oder das Lesen von Archivdatenobjekten verwendet werden soll.

Mögliche Werte:

"submit" Speichern von Archivdatenobjekten

"retrieve" Lesen von Archivdatenobjekten

Standardwert: submit

i Derzeit kann nur der Wert `submit` angegeben werden. Ein Filter mit der Angabe `mode="retrieve"` wird bei der Registrierung des SDO-Typs (Operation `createSDOType`, siehe "[Operation createSDOType](#)") abgewiesen.

4.2.1.2 Element `<namespace>`

`<namespace>` ordnet einem Namespace eine Kurzbezeichnung zu.

Mit dieser Kurzbezeichnung können alle Bestandteile eines Ausdrucks angegeben werden, der in einem Attribut `xpath` verwendet wird (siehe Abschnitte „[Element `<element>`](#)“ und „[Element `<node>`](#)“).

Regeln

- Die Namespaces im Filter müssen mit den Namespaces im XML-Schema und damit im SDO übereinstimmen.
- Entscheidend ist nicht die Kurzbezeichnung, sondern der zugeordnete Namespace. Daher müssen die Kurzbezeichnungen im Filter nicht mit den Kurzbezeichnungen im SDO übereinstimmen.
- Für einen Namespace können Sie auch mehr als eine Kurzbezeichnung definieren.
- Eine Namespace-Zuordnung gilt ab der Definition im Filter.

```
<xs:complexType name="TNamespace">
  <xs:attribute name="uri" type="xs:string" use="required"/>
  <xs:attribute name="prefix" type="xs:string" use="required"/>
</xs:complexType>
```

Elemente

`<TNamespace>` enthält keine weiteren Elemente.

Attribute

- | | |
|---------------------|---|
| <code>uri</code> | Eindeutiger Namespace, der im Attribut <code>xpath</code> durch die Kurzbezeichnung ersetzt werden soll. Dieser Namespace muss mit dem Namespace im XML-Schema übereinstimmen (siehe Abschnitt „ Beispiel “). |
| <code>prefix</code> | Kurzbezeichnung, die im Attribut <code>xpath</code> den Namespace repräsentieren soll, der mit <code>uri</code> festgelegt wurde. |

Verwendung in `xpath`

Die Kurzbezeichnung für den Namespace kann im Attribut `xpath` wie folgt verwendet werden:

`prefix:localname`

`prefix`

Kurzbezeichnung, wie sie in `<namespace>` definiert wurde.

`localname`

lokaler Name.

Beispiel

```
<namespace prefix="ns"
```

```
uri="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"/>
```

Verwendung in xpath:

```
...  
<node mapname="$DataNode"           xpath="/ns:mydocument/file">  
<element mapname="$Content"         xpath="ns:content"    maptype="stream"/>  
...
```

Vererbung von Namespaces

Für die Vererbung von Namespaces in XPath-Ausdrücken gelten folgende Regeln:

- Wenn für eine Unterstruktur kein Namespace angegeben ist, wird der übergeordnete Namespace grundsätzlich vererbt.

Beispiel

Im XPath-Ausdruck `/abc/ns:def/ghi` hat `abc` keinen Namespace. Aufgrund der Vererbung hat `ghi` den gleichen Namespace `ns` wie `def`.

Die Ausdrücke `/abc/ns:def/ghi` und `/abc/ns:def/ns:ghi` sind also identisch.

- Die Vererbung der Namespaces gilt nicht für Attribute. Ein Attribut ohne Namespace-Angabe hat also nie einen Namespace, auch wenn in der übergeordneten Struktur ein Namespace zugewiesen ist.

Beispiel

Im XPath-Ausdruck `/ns:abc/@aaa` hat das Attribut `aaa` keinen Namespace.

Wenn das Attribut `aaa` den Namespace der übergeordneten Struktur annehmen soll, muss er explizit angegeben werden:

```
/ns:abc/ns:@aaa
```

Beispiel

Das XML-Schema des SDOs `MyDocument` beginnt mit folgenden Zeilen:

```
<xsd:schema xmlns="http://ts.fujitsu.com/secdocs/sdosamples/mydocument "  
    elementFormDefault="qualified"  
    attributeFormDefault="unqualified"  
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
    xmlns:ns="http://ts.fujitsu.com/secdocs/sdosamples/mydocument "  
    targetNamespace="http://ts.fujitsu.com/secdocs/sdosamples/  
        mydocument "  
    version="1.1">
```

Das bedeutet, die Elemente des SDOs `MyDocument` gehören zum Namespace

```
http://ts.fujitsu.com/secdocs/sdosamples/mydocument
```

Diesem Namespace wird im Element `<namespace>` folgende Kurzbezeichnung zugewiesen:

```
<namespace prefix="ns"
```

```
    uri="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"/>
```

Wenn Sie im Filter Elemente vom Typ `MyDocument` adressieren wollen, müssen Sie im XPath-Ausdruck das Element mit dieser Kurzbezeichnung angeben (siehe Attribut „`xpath`“ (in Abschnitt "[Element <element>](#)")):

```
<element mapname="id" xpath="/ns:myDocument/id" default="none"/>
```

4.2.1.3 Element <element>

<element> filtert eine bestimmte Information zur Adressierung aus dem SDO heraus und enthält ggf. Zusatzinformationen.

Mit <element> können Sie

- eigene Schlüssel definieren und ihnen Werte aus dem SDO zuweisen, siehe [Abschnitt „Verwendung von Schlüsseln“](#),
- den Systemschlüsseln Werte aus dem SDO zuweisen,
- den eigenen oder den Systemschlüsseln feste Werte als Standardwert zuweisen.

```
<xs:complexType name="TElement">
  <xs:attribute name="mapname" type="xs:string" use="required"/>
  <xs:attribute name="xpath" type="xs:string" use="optional"/>
  <xs:attribute name="default" type="xs:string" use="optional"/>
  <xsd:attribute name="maptype">
    <xsd:simpleType>
      <xsd:restriction base="xs:string">
        <xsd:enumeration value="string"/>
        <xsd:enumeration value="stream"/>
        <xsd:enumeration value="integer"/>
        <xsd:enumeration value="decimal"/>
        <xsd:enumeration value="boolean"/>
        <xsd:enumeration value="date"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xs:complexType>
```

Elemente

<TElement> enthält keine weiteren Elemente.

Attribute

mapname Schlüssel, dem eine im SDO adressierte Information und/oder ein Standardwert zugewiesen wird. Im Filter können Elemente mit identischem **mapname** verwendet werden.

Benutzerdefinierte Schlüssel

mapname definiert einen Schlüssel, dem mit **xpath** und/oder **default** ein Wert zugewiesen wird.

Soll der Schlüssel zur Indexierung verwendet werden, soll also der Name auch dem Systemschlüssel **\$Index** zugewiesen werden, so dürfen Sie im Namen keine Sonderzeichen (außer Unterstrich) angeben (den genauen Unicode-Zeichensatz finden Sie im [Abschnitt „\\$Index“](#)).

Der Name des benutzerdefinierten Schlüssels darf nicht mit dem Zeichen „\$“ beginnen.

Benutzerdefinierte Schlüssel können typisiert werden. Per Default werden die Werte der benutzerdefinierten Schlüssel als String behandelt. Bei der Recherche mit der Operation **sparqlQuery** können Sie in der SPARQL-Filter-Angabe Vergleichsoperationen definieren, die bei diesen expliziten Typangaben berücksichtigt werden.

Beispiel

```
<element mapname="id" xpath="/ns:document/id" default="none" />
```

Systemschlüssel

mapname enthält einen von SecDocs vorgegebenen Schlüssel. Mit **xpath** und/oder **default** wird diesem Systemschlüssel ein Wert zugewiesen.

Der Name des Schlüssels beginnt mit dem Zeichen „\$“.

Beispiel

```
<element mapname="$ContentType" default="PDF-A" />
```

Die SecDocs-Systemschlüssel und ihre Verwendung sind ausführlich im [Abschnitt „Verwendung von Schlüsseln“](#) beschrieben.

xpath optional; XPath-Ausdruck, mit dem das Element im SDO adressiert wird. Kommt ein Element mehrfach vor, werden alle diese Elemente mit einem XPath-Ausdruck adressiert.

In SecDocs wird ein Subset des standardisierten XPath verwendet.

- i**
- Wenn das mit **xpath** adressierte Element im SDO nicht existiert, wird, falls ein **default** angegeben ist, die Archivierung unter Verwendung dieses Wertes durchgeführt. Ist auch kein **default** definiert, wird die Archivierung abgewiesen.
 - Wenn das mit **xpath** adressierte Element im SDO existiert, der Inhalt aber ungültig ist, wird die Archivierung abgewiesen.



- Zur Vererbung von Namespaces in XPATH-Ausdrücken siehe Abschnitt „[Verwendung in xpath](#)“.
- Zur Kombination der Attribute `default` und `xpath` siehe Abschnitt „[Attribute xpath und default](#)“.
- Zur Verwendung von identischen Map-Namen (Attribut `mapname`) und/oder Pfaden (Attribut `xpath`) siehe Abschnitt „[Attribute xpath und mapname](#)“.
- Zu den Besonderheiten bei der Verwendung von Systemschlüsseln, siehe [Abschnitt „Verwendung von Schlüsseln“](#).

Wenn `<element>` auf der obersten Ebene liegt, gilt:

Der XPath-Ausdruck muss immer mit dem Document-Root „/“ beginnen.

Zulässige Ausdrücke:

<code>/step1/step2/.../element</code>	Adressiert den Inhalt des Elements <code>element</code> im SDO.
<code>/step1/step2/.../element</code> <code>/@attribute</code>	Adressiert den Inhalt des Attributs <code>attribute</code> im Element <code>element</code> im SDO.

`step` bezeichnet hier ein Knotenelement im SDO:

- `step` darf nicht leer sein, d.h. die Angabe „/“ ist nicht erlaubt.
- In `step` dürfen keine Wildcards verwendet werden.

Wenn `<element>` innerhalb eines `<$DataNode>` liegt, gilt:

Der XPath-Ausdruck darf nicht mit dem Document-Root „/“ beginnen, sondern muss den entsprechenden relativen Pfadbestandteil enthalten.

Beispiel 1

```
<namespace prefix="ns"
  uri="http://ts.fujitsu.com/secdocs/sdosamples/mydocument" />
<element mapname="id" xpath="/ns:document/id" default="none" />
```

bezeichnet folgendes Element im SDO:

```
<ns3:document
  xmlns:ns3="http://ts.fujitsu.com/secdocs/sdosamples/mydocument" ...>
  ...
  <ns3:id>1234</id>
  ...
</ns3:document>
```


Beispiel 2

```
<namespace prefix="ns"
  uri="http://ts.fujitsu.com/secdocs/sdosamples/mydocument" />
<element mapname="id" xpath="/ns:document/metadaten/@id"
  default="none" />
```

bezeichnet folgendes Element im SDO:

```
<ns3:document
  xmlns:ns3="http://ts.fujitsu.com/secdocs/sdosamples/mydocument" ...>
  ...
  <ns3:metadaten id="1234"></ns3:metadaten>
  ...
</ns3:document>
```

default optional; SecDocs verwendet diesen Wert, wenn das Attribut `xpath` fehlt oder das Feld nicht im SDO enthalten ist, das mit `xpath` adressiert wird. Existiert jedoch das Feld im SDO, überschreibt sein Wert den Wert, der in `default` angegeben ist.

Wenn in einem Filter Elemente mit identischem Schlüsselnamen und unterschiedlichen Werten für `default` existieren, gilt jeweils der zuletzt gesetzte `default`.

Die möglichen Werte sind abhängig vom Datentyp der Information, die mit `mapname` adressiert ist.



- Zur Kombination der Attribute `default` und `xpath` siehe Abschnitt ["Attribute xpath und default"](#).
- Zu den Besonderheiten bei der Verwendung von Systemschlüsseln, siehe [Abschnitt „Verwendung von Schlüsseln“](#).

maptype Bezeichnet den Datentyp der adressierten Information aus dem SDO.

Mögliche Werte:

"string"	String-Objekt
"stream"	Stream-Objekt
"integer"	Integer-Objekt
"decimal"	Decimal-Objekt
"boolean"	Boolean-Objekt Erlaubte Werte sind hier nur "true" und "false", wobei die Groß-/Kleinschreibung nicht relevant ist.
"date"	DateTime-Objekt nach ISO8601 in der Form YYYY-MM-DD. Durch explizite Definition (siehe Abschnitt „\$DateFormat“) kann auch ein anderes Format vereinbart werden.

Standardwert: `string`

- i**
- Stellen Sie sicher, dass im SDO für die Schlüssel ein dem angegebenen `maptype` entsprechender Wert vorkommt. Andernfalls wird die Operation `submitSDO` bzw. `replaceSDO` abgewiesen.
 - Der Wert `stream` wird für Schlüssel (`$Content` und `$Signature`) verwendet, bei denen auf größere Datenmengen verwiesen wird, um intern eine performantere Verarbeitung zu ermöglichen. Für benutzerdefinierte Schlüssel darf dieser Typ nicht verwendet werden.
 - Bei `maptype="stream"` darf mit `xpath` nicht auf ein Attribut verwiesen werden.
 - Bei `maptype="date"` sind zwei Fälle zu unterscheiden:
 1. Wenn `xpath` auf ein Attribut oder ein Element des Typs `dateTime` verweist, muss der Schlüssel `$DateFormat` im Format `YYYY-MM-DD` definiert werden.
 2. Wenn `xpath` auf ein Attribut oder ein Element des Typs `string` verweist, kann der Schlüssel `$DateFormat` mit einem anderen Format als `YYYY-MM-DD` definiert werden. Bei der Operation `submitSDO` kann dann als Wert entweder nur ein Datum oder auch ein Datum mit Zeitangabe angegeben werden.



Zu den Besonderheiten bei der Verwendung von Systemschlüsseln, siehe [Abschnitt „Verwendung von Schlüsseln“](#).

Attribute `xpath` und `default`

- Wenn in einem Element `<element>` beide Attribute `xpath` und `default` angegeben sind, überschreibt `xpath` den Wert in `default`.
- Wenn in einem Element `<element>` weder `xpath` noch `default` angegeben ist, wird der Information, die mit `mapname` bezeichnet ist, kein Wert zugeordnet (der Schlüssel existiert nicht).

- Wenn `default` angegeben ist, `xpath` jedoch nicht, wird der Information, die mit `mapname` bezeichnet ist, immer dieser feste Wert zugeordnet. Auf diese Weise können Sie Konstanten definieren, z.B. eine feste Aufbewahrungszeit für Rechnungen von 10 Jahren.
- Wenn `default` angegeben ist, aber das Feld, das mit `xpath` adressiert wurde, nicht im SDO enthalten ist, wird der Information, die mit `mapname` bezeichnet ist, ebenfalls dieser feste Wert zugeordnet.

Attribute `xpath` und `mapname`

Die Elemente in einer Filterdefinition dürfen identische Map-Namen (Attribut `mapname`) und/oder Ausdrücke im Attribut `xpath` enthalten.

Identische Map-Namen und unterschiedliche XPath-Ausdrücke

Dieser Fall tritt in folgenden Fällen auf:

- Wenn im Filter für ein und denselben Map-Namen zwei oder mehr XPath-Ausdrücke angegeben werden.

Beispiel

```
<element mapname="name1" xpath="/ns:path1" />
```

```
<element mapname="name1" xpath="/ns:path2" />
```

- Wenn `xpath` auf ein Element im SDO verweist, das mehrfach vorhanden ist. Dies ist der Fall, wenn das Element im XML-Schema für das SDO mit `maxOccurs>1` definiert ist und im SDO mehrere Element enthalten sind.

Die Elemente aus dem SDO werden hier mit den unterschiedlichen XPath-Ausdrücken unter ein- und demselben Map-Namen abgelegt.

Dieses Vorgehen ist beispielsweise sinnvoll, wenn für `mapname` die Systemschlüssel `$RetentionPeriod`, `$ExpirationDate` oder `$Signature` verwendet werden.

Beispiel

In einem SDO sind für ein Primärdokument mehrere Signaturen zu prüfen, diese liegen aber an unterschiedlichen Pfaden.

Unterschiedliche Map-Namen und identische XPath-Ausdrücke

Dieser Fall tritt auf, wenn im Filter für mehrere Map-Namen ein- und derselbe XPath-Ausdruck angegeben wird.

Beispiel

```
<element mapname="name1" xpath="/ns:path1" default="value1" />
```

```
<element mapname="name2" xpath="/ns:path1" default="value2" />
```

Der Inhalt der Elemente mit dem Pfad `xpath` wird unter verschiedenen Map-Namen abgelegt. Dabei kann für jeden Map-Namen ein eigener Wert im Attribut `default` definiert werden.

Mit dieser Vorgehensweise kann ein bereits zugewiesener Wert zusätzlich in einem weiteren benutzerspezifischen Schlüssel hinterlegt werden.

4.2.1.4 Element <node>

<node> gruppiert XML-Elemente im SDO und ggf. weitere Zusatzinformationen in Form von Unterelementen. Auf diese Weise können Primärdokumente und entsprechende Signaturen als zusammengehörig gekennzeichnet werden.

<node> wird verwendet, um mit Hilfe des Schlüssels \$DataNode die Informationen für die Signaturprüfung zu definieren.

Der XPath-Ausdruck beschreibt eine Struktur im SDO, in der die zugehörigen Unterelemente von <node> liegen.

```
<xs:complexType name="TNode">
  <xs:sequence>
    <xs:element name="element" type="TElement" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="mapname" type="xs:string" use="required"/>
  <xs:attribute name="xpath" type="xs:string" use="required"/>
</xs:complexType>
```

Elemente

element

XML-Element, das eine bestimmte Information zur Adressierung aus dem SDO herausfiltert oder eine Zusatzinformation enthält.

Eine oder mehrere Datentypen TElement, siehe "[Element <element>](#)".

Für <element> gilt hier:

Die XPath-Ausdrücke in den Unterelementen sind relativ zum Pfad im darüberliegenden <node> (siehe "[xpath](#)" im Element <node>). Mit dieser Konstruktion können Sie z.B. Primärdokumente und ihre Signaturen einander zuordnen.

Beispiel

```
<node mapname="name4" xpath="/ns:path4">
  <element mapname="name41" xpath="path41"/>
  <element mapname="name42" xpath="path42"/>
</node>
```

Das Element <node> mit dem Map-Namen name4 hat die Unterelemente name41 und name42. Die Pfade für die Unterelemente im SDO sind path41 und path42. Die vollständigen Pfade der beiden Unterelemente ergeben sich damit zu /path4/path41 und /path4/path42.

Attribute

mapname Schlüssel, dem eine im SDO adressierte Information und/oder ein Standardwert zugewiesen wird. Im Filter können Sie Elemente mit identischem `mapname` verwenden.

Im Element `<node>` ist nur der Systemschlüssel `$DataNode` zulässig (siehe [Abschnitt „\\$DataNode“](#)).

Beispiel

```
<node mapname="$DataNode" xpath="/ns:myDocument/file">
```

i Bei `mapname="$DataNode"` muss der adressierte Knoten im SDO mindestens ein Element für einen der Systemschlüssel `$Content`, `$ContentRef` oder `$ExternalRef` enthalten (siehe [Abschnitt „\\$DataNode“](#)):

Falls `$Content` oder `$ContentRef` angegeben ist, müssen folgende weiteren Schlüssel ebenfalls angegeben werden:

```
$ContentCode  
$ContentType  
$SignatureEmbedded  
$SignatureQualityLevel  
$SignatureVerification
```

xpath XPath-Ausdruck, mit dem der Knoten im SDO adressiert wird. In SecDocs wird ein Subset des standardisierten XPath verwendet.

Der XPath-Ausdruck muss immer mit dem Document-Root „/“ beginnen.

Zulässige Ausdrücke:

```
/step1/step2/.../element
```

 Adressiert den Inhalt des Elements `element` im SDO.

`step` bezeichnet hier ein Knotenelement im SDO:

- `step` darf nicht leer sein, d.h. die Angabe „/“ ist nicht erlaubt.
- In `step` dürfen keine Wildcards verwendet werden.



Zu den Besonderheiten bei der Verwendung von Systemschlüsseln, siehe [Abschnitt „Verwendung von Schlüsseln“](#).

Beispiel

`<node mapname="$DataNode" xpath="/ns:Big/myDocument" />`

bezeichnet folgendes Element im XML-Schema:

```
<xsd:element name="Big">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="filereference" type="xsd:string"/>
      <xsd:element name="myDocument" type="TDokument"
        minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

4.2.1.5 Element <alias>

Mit <alias> lassen sich Ersetzungsregeln für Werte eines Schlüssels definieren, die zum Zeitpunkt der Archivierung aus dem SDO (Attribut `xpath`) ermittelt werden (Operation `submitSDO`, siehe "[Operation submitSDO](#)" und Operation `replaceSDO`, siehe "[Operation replaceSDO](#)"). Ein solches Vorgehen ist z.B. sinnvoll, wenn das SDO Werte enthält, die nicht den Vorgaben von SecDocs entsprechen.

i Der Wert wird nur intern in SecDocs ersetzt. Das SDO selbst bleibt unverändert.

```
<xs:complexType name="TAlias">
  <xs:attribute name="mapname" type="xs:string" use="required"/>
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="to" type="xs:string" use="required"/>
</xs:complexType>
```

Elemente

<TAlias> enthält keine weiteren Elemente.

Attribute

- | | |
|----------------------|--|
| <code>mapname</code> | Benutzerdefinierter Schlüssel oder Systemschlüssel, dem mit den Attributen <code>value</code> und <code>to</code> ein neuer Wert zugewiesen wird (siehe Abschnitt „Verwendung von Schlüsseln“). |
| <code>value</code> | Wert des Schlüssels in <code>mapname</code> , der bei der Archivierung aus dem SDO ermittelt wird und auf einen neuen Wert abgebildet werden soll. |
| <code>to</code> | Neuer Wert des Schlüssels in <code>mapname</code> , auf den der Wert in <code>value</code> abgebildet werden soll. |

i Die Angabe im Attribut <default> beim Element <element> wird durch <alias> nicht ersetzt.
Die Aliase sind case-sensitiv, müssen also genau so verwendet werden, wie sie definiert sind.

Beispiel

SecDocs akzeptiert für den Systemschlüssel `$SignatureEmbedded` nur die Werte YES, NO oder AUTO.

Wenn der Wert von `$SignatureEmbedded` durch die Angabe von `xpath` aus dem SDO entnommen werden soll, muss dort YES, NO oder AUTO angegeben sein.

Wenn stattdessen im SDO z.B. die Werte JA oder NEIN verwendet werden, müssen sie mit dem Element <alias> abgebildet werden:

```
<alias mapname="$SignatureEmbedded" value="JA" to="YES" />
<alias mapname="$SignatureEmbedded" value="NEIN" to="NO" />
```

Diese Abbildung legt fest, dass anstelle der Werte JA/NEIN jetzt die vorgeschriebenen Werte YES/NO verwendet werden sollen.

4.2.2 Verwendung von Schlüsseln

Sie können in einer Filterdefinition für SecDocs verschiedene Schlüssel verwenden:

Benutzerdefinierte Schlüssel

In einer Filterdefinition können Sie mit `<element mapname="..." />` Schlüssel frei definieren, um sie dann im Ablagepfad und/oder im Index zu verwenden. Den Namen eines Schlüssels legen Sie im Attribut `mapname` fest.

Die Namen der benutzerdefinierten Schlüssel dürfen nicht mit „\$“ beginnen.

Beispiel

```
<element mapname="id" xpath="..." />
```

Voreingestellte Systemschlüssel

Die folgenden Systemschlüssel haben voreingestellte Werte. Diese Schlüssel werden nicht im Filter definiert. Sie sind für jedes Primärdokument vorhanden und werden von SecDocs mit Werten versorgt.

Schlüssel	Wert
\$Year	Aktuelles Jahr, vierstellig
\$Month	Aktueller Monat im Jahr Mögliche Werte: 01 - 12
\$Day	Aktueller Tag im Monat Mögliche Werte: 01 - 31
\$Dayofyear	Aktueller Tag im Jahr Mögliche Werte: 001 - 366
\$Hour	Aktuelle Stunde im Tag Mögliche Werte: 00 - 23
\$Minute	Aktuelle Minute in der Stunde Mögliche Werte: 00 - 59
\$COID	COID (Client Object Identifier) des SDOs. Die COID ist ein vom Client vergebbarer Bezeichner zur Identifizierung eines Archivobjekts. \$COID darf im Filter nur bei der Definition von \$Index und \$Subject genutzt werden.
\$MandantName	Mandantenname unter dem die Operation <code>submitSDO</code> oder <code>replaceSDO</code> ausgeführt wird bzw. wurde. \$MandantName darf im Filter nur bei der Definition von \$Index genutzt werden.
\$OrgName	Organisationsname unter dem die Operation <code>submitSDO</code> oder <code>replaceSDO</code> ausgeführt wird bzw. wurde. \$OrgName darf im Filter nur bei der Definition von \$Index genutzt werden.
\$Role	Rollenname unter dem die Operation <code>submitSDO</code> oder <code>replaceSDO</code> ausgeführt wird bzw. wurde. \$Role darf im Filter nur bei der Definition von \$Index genutzt werden.

\$SdoType	Der Name des SDOTypes des archivierten SDOs. \$SdoType darf im Filter nur bei der Definition von \$Index genutzt werden.
\$Version	SecDocs-Version mit der archiviert wird/wurde. Formatbeschreibung siehe Operation statusSDO, ab "Operation statusSDO". \$Version darf im Filter nur bei der Definition von \$Index genutzt werden.
\$VersionNumber	Fortlaufende Nummer zur Identifikation einer SDO-Version. Sie wird von SecDocs bei der Reservierung einer AOID für eine SDO-Version vergeben. \$VersionNumber darf im Filter nur bei der Definition von \$Subject, \$Index und \$SDOPath verwendet werden.

Tabelle 2: Vordefinierte Systemschlüssel

Systemschlüssel, denen Sie einen Wert in der Filterdefinition zuweisen können

Sie bestimmen die Werte dieser Schlüssel durch die Angaben im Filter. Die Namen der Systemschlüssel beginnen mit dem Zeichen „\$“.

Folgende Systemschlüssel stehen in SecDocs zur Verfügung:

Schlüsselgruppe	Schlüssel	Beschreibung
Datumsangaben	\$DateFormat	Datumsformat für den Datumsteil von \$ExpirationDate, siehe "\$DateFormat"
	\$ExpirationDate	Freigabezeitpunkt, siehe "\$ExpirationDate"
	\$RetentionPeriod	Aufbewahrungszeit, siehe "\$RetentionPeriod"
Primärdokumente und Signaturen	\$Content	xpath-Angabe für den Knoten, der das Primärdokument enthält, siehe "\$Content"
	\$ContentRef	xpath-Angabe für den Knoten, der die externe Referenz-ID eines Primärdokuments mit Signatur enthält, siehe "\$ContentRef"
	\$ExternalRef	xpath-Angabe für den Knoten, der die externe Referenz-ID eines Primärdokuments ohne Signatur enthält, siehe "\$ExternalRef"
	\$ContentCode	Codierung des Primärdokuments, siehe "\$ContentCode"
	\$ContentType	Dateiformat des Primärdokuments und ggf. der Signatur, siehe "\$ContentType"
	\$DataNode	xpath-Angabe für den Knoten, der das Primärdokument und ggf. die Signaturen enthält, siehe "\$DataNode"
	\$Signature	xpath-Angabe für den Knoten, der die nicht eingebetteten Signaturen enthält, siehe "\$Signature"

	\$SignatureEmbedded	Gibt an, ob das Primärdokument eine eingebettete Signatur hat, siehe "\$SignatureEmbedded"
	\$SignatureDetached	Gibt an, ob SecDocs beim Archivieren eine abgesetzte Signatur zu einem Primärdokument erwartet, siehe "\$SignatureDetached"
	\$SignatureType	Art der Signatur, siehe "\$SignatureType"
	\$SignatureQualityLevel	Anforderungen an die elektronische Signatur bzw. an den Zertifikatsanbieter, die für eine Archivierung durch SecDocs notwendig sind, siehe "\$SignatureQualityLevel"
	\$SignatureVerification	Legt fest, welches Ergebnis die Verifikation der Signaturen mindestens erreichen muss, damit eine Archivierung durchgeführt wird, siehe "\$SignatureVerification"
Recherche im Archiv	\$Index	Schlüsselwörter, die in den Index aufgenommen werden sollen, siehe "\$Index"
	\$Subject	Frei definierbarer String (z.B. Erläuterungstext zum SDO), siehe "\$Subject"
Pfadangaben	\$SDOPath	Ablageort des SDOs im Archiv, relativ zum Pfad bei createOrganisation, siehe "\$SDOPath"

Tabelle 3: SecDocs-Systemschlüssel

4.2.2.1 Definition von Schlüsselwerten

Benutzerdefinierten Schlüsseln und nicht voreingestellten Systemschlüsseln weisen Sie Werte über die Attribute `<xpath>` und `<default>` im Element `<element>` zu. Für welchen Schlüssel Sie Werte definieren wollen, bestimmen Sie mit dem Attribut `<mapname>` (siehe [Element <element>](#)).

Die folgenden Angaben sind möglich:

```
<element mapname="key_name" default="value" />
<element mapname="key_name" xpath="path" />
<element mapname="key_name" xpath="path" default="value" />
```



Bei der Angabe von festen Werten für Systemschlüssel ist Groß- und Kleinschreibung erlaubt.

Beispiel

```
<element mapname="$ContentCode" default="BASE64" />
<element mapname="$ContentCode" default="base64" />
<element mapname="$ContentCode" default="Base64" />
```

4.2.2.2 Verwendung von Schlüsselwerten

Sie können diese Schlüssel mit ihren Werten als Variable im Rahmen der Filterdefinition verwenden.

\$SDOPath

Der wichtigste Anwendungsfall ist der Aufbau des Ablagepfades mit Werten aus dem SDO. Bei der Verwendung in \$SDOPath muss der Schlüsselname dazu in zwei Sterne („“) eingeschlossen werden.

Beispiel

```
<element mapname="$SDOPath"
    default="myDocument/*$Year*/*$Month*/*$Day*/*id*" />
```

ergibt z.B. "myDocument/2012/12/24/345-6789"

Das Beispiel verwendet beide möglichen Schlüsselarten:

- voreingestellten Schlüssel: \$Year, \$Month, \$Day
- Benutzerdefinierte Schlüssel: id

\$Subject

Freier String, der dem SDO zugeordnet wird. Dieser String wird bei verschiedenen Lese-Operationen zusätzlich zur AOID als additive Benutzerinformation zur Verfügung gestellt.

Bei der Formulierung der Beschreibung kann durch Angabe von **Schlüsselname** im Attribut `default=` auch der Inhalt eines Schlüssels verwendet werden. Dabei ist Schlüsselname entweder ein benutzerdefinierter Schlüssel oder der Name eines der folgenden Systemschlüssel: \$Day, \$Dayofyear, \$Hour, \$Minute, \$Month, \$Year, \$COID, \$VersionNumber.

i

- Die Filterdefinition wird beim Registrieren abgewiesen, wenn die Liste im Attribut `default` benutzerdefinierte Schlüssel enthält, die nicht mit `element mapname="..."` definiert sind.

SecDocs nimmt bei `$Subject` eine Ersetzung von Zeilenendezeichen vor:

	wird ersetzt durch
CarriageReturn	Linefeed
CarriageReturnLinefeed	Linefeed

Weitere Sonderzeichen werden nicht ersetzt. Wird `$Subject` direkt mit `"xpath="` ein Wert aus dem SDO zugewiesen und enthält dieser Wert die Zeichenfolge **Schlüsselname**, so wird dies nicht als Angabe eines Schlüssels interpretiert.

- Der `submitSDO` bzw. `replaceSDO` wird abgewiesen, falls einem **Schlüsselname**, der in der Default-Definition von `$Subject` verwendet wurde, kein Wert zugewiesen werden kann. Im Beispiel unten wäre das der Fall, wenn der für **AKZ** definierte `xpath` im zu archivierenden SDO nicht vorkommt.
- Es wird empfohlen, den Schlüssel `$Subject` nur einmal im Filter zu definieren. Bei mehrmaligem Vorkommen wird nur die zuerst aufgeführte Definition verwendet.
- Werden in einem SDO für einen Schlüssel (der in der `$Subject`-Definition genutzt wird) mehrere Werte definiert, so wird nur der erste gefundene Wert in den `$Subject` übernommen.

Beispiel

Im Systemschlüssel `$Subject` wird eine AOID-Beschreibung, zusammengesetzt aus einem AKZ (extrahiert aus dem archivierten Dokument) und dem Archivierungsdatum (ermittelt aus Systemschlüsseln), hinterlegt:

Zuerst wird der Schlüssel `AKZ` zum Ermitteln des Aktenzeichens aus dem SDO definiert.

```
<element mapname="AKZ" xpath=...>
```

Danach wird die Beschreibung aus `AKZ`, `$Day`, `$Month` und `$Year` zusammengesetzt:

```
<element mapname="$Subject" default="filereference=*AKZ* archived on *$Day*.*$Month*.*$Year*">
```

4.2.2.3 \$DateFormat

\$DateFormat legt das Datumsformat für den Datumsteil der folgenden Schlüssel fest:

- Systemschlüssel \$ExpirationDate
- benutzerdefinierte Schlüssel, deren maptype-Attribut mit date definiert wurde

Alle Wertzuweisungen an Schlüssel (also Zuweisungen per default- bzw. per xpath-Attribut) müssen entsprechend diesem Format erfolgen. Alle Zeitangaben innerhalb eines SDO müssen einheitlich sein.

Sinnvollerweise wird \$DateFormat zu Beginn des Filters einmal definiert oder die Angaben erfolgen im Default-Format "YYYY-MM-DD". Bei Mehrfach-Angabe von \$DateFormat in einem Filter wird die erste Definition verwendet, die weiteren Angaben werden ignoriert.

Mögliche Formate:

```
DD.MM.YYYY  
MM.DD.YYYY  
DD-MM-YYYY  
MM-DD-YYYY  
YYYY.DD.MM  
YYYY.MM.DD  
YYYY-DD-MM  
YYYY-MM-DD
```

Standardwert: YYYY-MM-DD

SecDocs akzeptiert nur numerische Datumsformate. D, M und Y entsprechen hier jeweils einer Ziffer von Tag, Monat und Jahr.


Beispiel

```
<element mapname="$DateFormat" default="DD.MM.YYYY">
```

4.2.2.4 \$ExpirationDate

\$ExpirationDate gibt den Freigabezeitpunkt eines SDOs an.

\$ExpirationDate und damit auch der Freigabezeitpunkt kann in der Vergangenheit liegen. Das bedeutet, dass das SDO jederzeit wieder gelöscht werden kann.

 Für \$ExpirationDate sind Werte ab dem dem 1.1.1970 00:01Uhr erlaubt.



Die Regeln zur Ermittlung des tatsächlichen Freigabezeitpunkts sind im [Abschnitt „Kombination von \\$ExpirationDate und \\$RetentionPeriod“](#) beschrieben.

Mögliche Werte: *Datum* | UNLIMITED | DEFERRED

Datum Datumsangabe im Format ISO 8601. Der Datumsteil ist im Format \$DateFormat anzugeben. Mögliche Werte:

DD	1 - 31 01 - 31
MM	1 - 12 01 - 12
hh	00 - 23 (00 24:00:00 Uhr)
mm	00 - 59 (00 60 Minuten)
ss	00 - 59 (00 60 Sekunden) Schaltsekunden werden nicht berücksichtigt
Zeitzone	±hh:mm Z (Z +00:00)

Die Angabe einer Uhrzeit und einer Zeitzone ist optional.

Standardwert für die Uhrzeit: 00:00:00

Standardwert für die Zeitzone: UTC (Koordinierte Weltzeit)

Die Angabe von Wochen ist nicht erlaubt.

Beispiel

2015-09-22T13:15:00+02:00

13:15:00 Uhr am 22. September 2015 in Deutschland, Österreich und der Schweiz
(Sommerzeit)

UNLIMITED

Unbegrenzte Aufbewahrungszeit

DEFERRED

Der endgültige Freigabezeitpunkt des Dokuments wird erst zu einem späteren Zeitpunkt mit der Operation `setExpirationDateSDO` gesetzt. Falls für \$ExpirationDate der Wert DEFERRED angegeben wird, muss \$RetentionPeriod einen gültigen Wert größer als 0 (also nicht P0D, P0M oder P0Y) enthalten.

Für `$ExpirationDate` sind die Werte `DEFERRED` und `UNLIMITED` als Defaultwerte zulässig. `DEFERRED` wird allerdings nur wirksam, wenn keine weitere `$ExpirationDate`-Definition (mit einem Wert ungleich `DEFERRED`) existiert.



- Die Angabe `<element mapname="$SoftWorm" default="YES" />` darf nicht mit der Angabe `UNLIMITED` bei `$ExpirationDate` kombiniert werden (siehe [Abschnitt „\\$SoftWorm“](#)).

Wenn für `$ExpirationDate` ein ungültiger Wert angegeben ist, wird das SDO bei der Archivierung abgewiesen (Operation `submitSDO`, siehe ["Operation submitSDO"](#), und `replaceSDO`, siehe ["Operation replaceSDO"](#)).

`$ExpirationDate` und `xpath`

- Wenn im Filter bei `mapname="$ExpirationDate"` mehrere Angaben mit `xpath` existieren, wird der späteste dieser Zeitpunkte als Wert für `$ExpirationDate` verwendet.
- `$ExpirationDate` gilt als nicht gesetzt,
 - wenn im Filter keine Angabe für `$ExpirationDate` vorliegt und
 - das Attribut `xpath` nicht aufgelöst werden kann, weil das `xpath`-Ziel nicht vorhanden oder leer ist.

Der tatsächliche Freigabezeitpunkt wird dann durch Auswertung von `$RetentionPeriod` ermittelt. Wenn auch `$RetentionPeriod` nicht angegeben ist, wird das SDO bei der Archivierung abgewiesen (Operation `submitSDO`, siehe ["Operation submitSDO"](#), und `replaceSDO`, siehe ["Operation replaceSDO"](#)).

Beispiel

Das SDO enthält folgendes Element:

```
<myDocument>
  <expirationdate>2020-31-12</expirationdate>
</myDocument>
```

Zugehörige Angabe im Filter:

```
<element mapname="$ExpirationDate" maptype="string"
          xpath="/ns:myDocument/expirationdate" />
```


4.2.2.5 \$RetentionPeriod

\$RetentionPeriod gibt die Zeitspanne an, die ein SDO ab dem Archivierungsdatum mindestens aufzubewahren ist.



- SecDocs ignoriert einen gültigen Wert von \$RetentionPeriod, wenn anhand der Filterdefinition bereits ein Wert für \$ExpirationDate ermittelt wurde.

Wird für \$RetentionPeriod ein ungültiger Wert angegeben, wird das SDO bei der Archivierung abgewiesen, auch wenn bereits ein Wert für \$ExpirationDate ermittelt wurde.

Eine Definition für \$RetentionPeriod ohne eine Angabe von xpath und default ist nicht zulässig.



Die Regeln zur Ermittlung des tatsächlichen Freigabezeitpunkts sind im [Abschnitt „Kombination von \\$ExpirationDate und \\$RetentionPeriod“](#) beschrieben.

year, nicht-negative Integerwerte für Jahr, Monat und Tag (entsprechend der ISONorm 8601 ohne
month, day Zeitangabe). Eine Angabe von Wochen ist nicht erlaubt.

Beispiele:

P30Y	30 Jahre
P6M	6 Monate
P2Y3M5D	2 Jahre, 3 Monate und 5 Tage
P0D	0 Tage (keine Aufbewahrungszeit)

Eine \$RetentionPeriod mit Wert „0“ kann durch alle Angaben dargestellt werden, die eine Zeitspanne von 0 Tagen repräsentieren.

UNLIMITED

Unbegrenzte Aufbewahrungszeit



Die Angabe `<element mapname="$SoftWorm" default="YES"/>` darf nicht mit der Angabe UNLIMITED bei \$RetentionPeriod kombiniert werden (siehe [Abschnitt „\\$SoftWorm“](#)).

\$RetentionPeriod und default

- Es wird empfohlen, in einer Filterdefinition für \$RetentionPeriod nur einen Wert für default zu setzen. Wenn mehrere Einträge für \$RetentionPeriod vorhanden sind, sollte also höchstens einer dieser Einträge ein Attribut default enthalten.
- Wenn doch mehrere Angaben für default vorhanden sind, verwendet SecDocs nur die letzte. Alle anderen Werte werden dann ignoriert.

Beispiel

```
<element mapname="$RetentionPeriod" xpath="/ns:P1" default=" P5Y" />
<element mapname="$RetentionPeriod" xpath="/ns:P2" />
<element mapname="$RetentionPeriod" xpath="/ns:P3" />
```

Sind keine Werte für `xpath` im SDO vorhanden, wird die Aufbewahrungszeit auf 5 Jahre gesetzt.

\$RetentionPeriod und xpath

- Enthält `$RetentionPeriod` mehrere Werte, wird der größte Zeitraum als Wert für `$RetentionPeriod` verwendet (siehe [Abschnitt „\\$ExpirationDate“](#)).
- `$RetentionPeriod` gilt als nicht gesetzt,
 - wenn im Filter keine Angabe für `$RetentionPeriod` im Attribut `default` gemacht ist und
 - das Attribut `xpath` nicht aufgelöst werden kann, weil das `xpath`-Ziel nicht vorhanden oder leer ist.

Wenn anhand der Filterdefinition kein Wert für `$ExpirationDate` ermittelt wurde und `$RetentionPeriod` nicht gesetzt ist, wird das SDO bei der Archivierung abgewiesen (Operation `submitSDO`, siehe "[Operation submitSDO](#)", und `replaceSDO`, siehe "[Operation replaceSDO](#)").

4.2.2.6 Kombination von `$ExpirationDate` und `$RetentionPeriod`

i In den meisten Fällen ist die minimale Aufbewahrungsfrist für einen SDO-Typ z.B. durch gesetzliche Vorschriften vorgegeben. In diesem Fall genügt es, diese Aufbewahrungsfrist durch einmalige Angabe der `$RetentionPeriod` im zugehörigen Filter zu definieren. Die folgende Beschreibung ist nur für Sonderfälle relevant, in denen es nötig wird, die Aufbewahrungsfrist als Kombination von `$ExpirationDate` und `$RetentionPeriod` zu beschreiben.

SecDocs ermittelt den tatsächlichen Freigabezeitpunkt für ein SDO aus den Angaben bei `$ExpirationDate` und `$RetentionPeriod`, siehe Abschnitt „[Freigabezeitpunkt](#)“. Das Ergebnis dieser Berechnung kann mit den Operationen `statusSDO` oder `metaDataSDO` im Schlüssel `$ExpirationDateTime` abgefragt werden (siehe "[Operation statusSDO](#)" bzw. "[Operation metaDataSDO](#)").

Bei der Kombination der beiden Schlüssel `$ExpirationDate` und `$RetentionPeriod` sind verschiedene Fälle möglich. Diese sind im Abschnitt „[Fallübersicht](#)“ beschrieben.

Freigabezeitpunkt

Der Freigabezeitpunkt entspricht Datum und Uhrzeit der Koordinierten Weltzeit (UTC), ab dem das SDO gelöscht werden kann.

i Bei Zeitzonen, die hinter der Koordinierten Weltzeit (UTC) liegen, und fehlender Zeitangabe in `$ExpirationDate` kann diese Berechnung dazu führen, dass das SDO schon eine entsprechende Zeit vor dem angegebenen Freigabezeitpunkt gelöscht werden kann.

Bei Zeitzonen, die der Koordinierten Weltzeit (UTC) voraus sind, und fehlender Zeitangabe in `$ExpirationDate` kann diese Berechnung dazu führen, dass das SDO erst eine entsprechende Zeit nach dem angegebenen Freigabezeitpunkt gelöscht werden kann.

SecDocs berechnet den tatsächlichen Freigabezeitpunkt in folgender Reihenfolge:

1. Es können ein oder mehrere Werte für `$ExpirationDate` ermittelt werden:
 - SecDocs berechnet den Maximalwert aus den absoluten Zeitangaben von `$ExpirationDate` und verwendet diesen als Freigabezeitpunkt.
 - Angaben für `$RetentionPeriod` werden ignoriert.
2. Es können ein oder mehrere Werte für `$RetentionPeriod` ermittelt werden:
 - SecDocs berechnet den Maximalwert aus den relativen Zeitangaben von `$RetentionPeriod` und verwendet diesen für die Berechnung des Freigabezeitpunkts.
 - Angaben im Attribut `default` für `$RetentionPeriod` werden ignoriert.
3. Es kann kein Wert für `$RetentionPeriod` ermittelt werden, in der Filterdefinition existiert aber eine Angabe im Attribut `default` für `$RetentionPeriod`:

SecDocs verwendet diesen Wert zur Berechnung des Freigabezeitpunkts. Bei mehreren existierenden `default` Werten wird der letzte gefundene Wert verwendet.

-
4. Es können keine Werte für `$ExpirationDate` oder `$RetentionPeriod` ermittelt werden und in der Filterdefinition existiert keine Angabe im Attribut `default` für `$RetentionPeriod`:

Die Operation `submitSDO` (siehe "[Operation submitSDO](#)") oder `replaceSDO` (siehe "[Operation replaceSDO](#)") wird abgewiesen.

Fallübersicht

Für `$ExpirationDate` sind die folgenden Fälle möglich:

- In der Filterdefinition existiert kein Element für `$ExpirationDate`.
- In der Filterdefinition existiert ein Element für `$ExpirationDate`, aber im SDO gibt es keinen gültigen Wert: Der bei `xpath` angegebene Pfad ist im SDO einmal oder mehrfach vorhanden und mindestens ein Wert ist ungültig.
- In der Filterdefinition existiert ein Element für `$ExpirationDate`, aber im SDO gibt es keine Angabe: Für den Pfad, der bei `xpath` angegeben ist, existiert kein Element im SDO oder das Element ist leer.
- In der Filterdefinition existiert ein Element für `$ExpirationDate` und im SDO existiert ein gültiger Wert: Der bei `xpath` angegebene Pfad ist im SDO einmal oder mehrfach vorhanden und alle Werte sind gültig.



- Wenn `$ExpirationDate` im Filter mehrfach angegeben ist, wird die Filterdefinition bereits bei einem ungültigen Element abgewiesen (Operation `createSDOType`, siehe "[Operation createSDOType](#)").
Wenn `$ExpirationDate` im SDO mehrfach angegeben ist, wird die Archivierung bei einem ungültigen Wert abgewiesen (Operation `submitSDO`, siehe "[Operation submitSDO](#)", und `replaceSDO`, siehe "[Operation replaceSDO](#)").

Für `$RetentionPeriod` sind die folgenden Fälle möglich:

- In der Filterdefinition existiert kein Element für `$RetentionPeriod`.
- In der Filterdefinition existiert ein Element für `$RetentionPeriod`, aber im SDO gibt es keinen gültigen Wert:
 - In der Filterdefinition existiert keine Angabe für `default` und der Pfad, der bei `xpath` angegeben ist, existiert nicht im SDO.
 - In der Filterdefinition existiert keine Angabe für `default` und der Pfad, der bei `xpath` angegeben ist, existiert im SDO, aber das Element ist leer.
 - In der Filterdefinition existiert keine Angabe für `default` und das Attribut, das bei `xpath` angegeben ist, existiert nicht im SDO.
 - Der bei `xpath` angegebene Pfad ist im SDO einmal oder mehrfach vorhanden und mindestens ein Wert ist ungültig. In diesem Fall ist die Angabe bei `default` irrelevant.
- In der Filterdefinition existiert ein Element für `$RetentionPeriod` und für `default` ist ein gültiger Wert angegeben, aber es existiert kein gültiger Wert für `xpath`:
 - `xpath` ist nicht angegeben.
 - Der Pfad, der bei `xpath` angegeben ist, existiert nicht im SDO.
 - Der Pfad, der bei `xpath` angegeben ist, existiert im SDO, aber das Element ist leer.

- In der Filterdefinition existiert ein Element für `$RetentionPeriod` und im SDO ein gültiger Wert:
Der bei `xpath` angegebene Pfad ist im SDO einmal oder mehrfach vorhanden und alle Werte sind gültig.



- Wenn `$RetentionPeriod` im Filter mehrfach angegeben ist, wird die Filterdefinition bereits bei einem ungültigen Element abgewiesen (Operation `createSDOType`, siehe "[Operation createSDOType](#)").

Wenn `$RetentionPeriod` im SDO mehrfach angegeben ist, wird die Archivierung bereits bei einem ungültigen Wert abgewiesen (Operation `submitSDO`, siehe "[Operation submitSDO](#)" und Operation `replaceSDO`, siehe "[Operation replaceSDO](#)").

Sobald im SDO ein gültiger Wert für eine der Angaben bei `$RetentionPeriod` gefunden wurde, werden die Angaben in `default` nicht mehr berücksichtigt.

Die folgende Tabelle zeigt die Kombination der möglichen Fälle für `$ExpirationDate` und `$RetentionPeriod` und die zugehörige Reaktion von SecDocs:

Bei <code>\$RetentionPeriod</code> ¹ Bei <code>\$ExpirationDate</code> ²	Kein Element in der Filterdefinition	Ungültiger Wert im SDO	Gültiger default- Wert in der Filterdefinition und keine <code>xpath</code> -Angabe in der Filterdefinition oder kein gültiger Wert im SDO	Ein Wert oder mehrere gültige Werte im SDO
Kein Element in der Filterdefinition	<code>createSDOType</code> wird abgewiesen	<code>submitSDO</code> / <code>replaceSDO</code> wird abgewiesen	<code>submitSDO</code> / <code>replaceSDO</code> wird akzeptiert ³ Freigabezeitpunkt = Archivierungsdatum + default-Wert für <code>\$RetentionPeriod</code>	<code>submitSDO</code> / <code>replaceSDO</code> wird akzeptiert ³ Freigabezeitpunkt = Archivierungsdatum + Maximalwert für <code>\$RetentionPeriod</code> im SDO

Keine Angabe im SDO	submitSDO / replaceSDO wird abgewiesen	submitSDO / replaceSDO wird abgewiesen	submitSDO / replaceSDO wird akzeptiert ³ Freigabezeitpunkt = Archivierungsdatum + default-Wert für \$RetentionPeriod	submitSDO / replaceSDO wird akzeptiert ³ Freigabezeitpunkt = Archivierungsdatum + Maximalwert für \$RetentionPeriod im SDO
Ein Wert oder mehrere gültige Werte im SDO	submitSDO / replaceSDO wird akzeptiert ³ Freigabezeitpunkt = Maximalwert für \$ExpirationDate im SDO	submitSDO / replaceSDO wird abgewiesen	submitSDO / replaceSDO wird akzeptiert ³ Freigabezeitpunkt = Maximalwert für \$Expirati- onDate im SDO	submitSDO / replaceSDO wird akzeptiert ³ Freigabezeitpunkt = Maximalwert für \$Expirati- onDate im SDO

Tabelle 4: Kombination von \$ExpirationDate und \$RetentionPeriod

¹Ausführliche Beschreibung der möglichen Fälle siehe "[Fallübersicht - \\$RetentionPeriod](#)".

²Ausführliche Beschreibung der möglichen Fälle siehe "[Fallübersicht - \\$ExpirationDate](#)".

³Voraussetzung für submitSDO: createSDOType ist fehlerfrei ausgeführt worden.

4.2.2.7 \$Content

\$Content gibt den Pfad zu demjenigen Knoten im SDO an, der das Primärdokument enthält.

Die Angabe im Attribut `xpath` bezeichnet hier

- einen **relativen** Pfad im SDO, wenn \$Content unterhalb einer Knotendefinition mit dem Element `<node>` verwendet wird. Dieser Pfad ist relativ zur `xpath`-Angabe im Knoten mit `mapname="$DataNode"` (siehe "[\\$DataNode](#)").
- einen **absoluten** Pfad im SDO, wenn \$Content ohne Angabe von \$DataNode eingesetzt wird. In diesem Fall müssen auch die folgenden Schlüssel angegeben werden:

```
$ContentCode  
$ContentType  
$SignatureEmbedded  
$SignatureQualityLevel  
$SignatureVerification
```

Mögliche Werte: Stream-Objekte



- Die Filterdefinition wird bei der Registrierung abgewiesen, wenn für das Attribut `maptype` nicht `maptype="stream"` angegeben ist (siehe [Abschnitt „Element <element>“](#)).

Der Systemschlüssel `$ContentCode` muss bei Angabe von \$Content immer den Wert `BASE64` enthalten.

Enthält der Filter mehr als eines der Elemente \$Content oder \$ContentRef, dann müssen diese Angaben mittels \$DataNode getrennt werden.

Pro \$DataNode darf es maximal eines der Elemente \$Content oder \$ContentRef geben.

Beispiel

```
<element mapname="$Content" xpath="ns:content" maptype="stream"/>
```

```
<element mapname="$ContentCode" default="BASE64"/>
```

4.2.2.8 \$ContentRef

\$ContentRef gibt den Pfad zu demjenigen Element im SDO an, das die externe Referenz-ID des (als externes Datenobjekt vorliegenden) Primärdokuments enthält.

Die Angabe im Attribut `xpath` bezeichnet hier

- einen **relativen** Pfad im SDO, wenn \$ContentRef unterhalb einer Knotendefinition mit dem Element `<node>` verwendet wird. Dieser Pfad ist relativ zur `xpath`-Angabe im Knoten mit `mapname= "$DataNode"` (siehe "[\\$DataNode](#)").
- einen **absoluten** Pfad im SDO, wenn \$ContentRef ohne Angabe von \$DataNode eingesetzt wird. In diesem Fall müssen auch die folgenden Schlüssel angegeben werden:

```
$ContentCode  
$ContentType  
$SignatureEmbedded  
$SignatureQualityLevel  
$SignatureVerification
```

Mögliche Werte:

Referenz-IDs im Format, wie sie von der Operation `getAOIDWithRef` zurückgeliefert werden.

\$ContentRef darf keinen Leerstring enthalten.



- Die Filterdefinition wird bei der Registrierung abgewiesen, wenn das Attribut `maptype` nicht den Wert "string" enthält (siehe [Abschnitt „Element <element>“](#)).

Der Systemschlüssel \$ContentCode muss bei Angabe von \$ContentRef immer den Wert `BINARY` enthalten.

Enthält der Filter mehr als eines der Elemente \$Content oder \$ContentRef, dann müssen diese Angaben mittels \$DataNode getrennt werden.

Pro \$DataNode darf es maximal eines der Elemente \$Content oder \$ContentRef geben.

Beispiel

```
<element mapname="$ContentRef" xpath="ns:content"/>  
<element mapname="$ContentCode" default="BINARY"/>  
<element mapname="$ContentType" default="OTHERS"/>
```


4.2.2.9 \$ExternalRef

\$ExternalRef gibt den Pfad zu demjenigen Element im SDO an, das die externe Referenz-ID des (als externes Datenobjekt vorliegenden) Primärdokuments enthält.

\$ExternalRef dient zum Archivieren von externen Datenobjekten, die keine eingebettete oder abgesetzte Signatur enthalten. Wenn Sie externe Datenobjekte mit Signatur archivieren wollen, verwenden Sie den Systemschlüssel \$ContentRef.

Die Angabe im Attribut `xpath` bezeichnet hier

- einen **relativen** Pfad im SDO, wenn \$ExternalRef unterhalb einer Knotendefinition mit dem Element `<node>` verwendet wird. Dieser Pfad ist relativ zur `xpath`-Angabe im Knoten mit `mapname="$DataNode"` (siehe ["\\$DataNode"](#)).
- einen **absoluten** Pfad im SDO, wenn \$ExternalRef ohne Angabe von \$DataNode eingesetzt wird.

Mögliche Werte:

Referenz-IDs im Format, wie sie von der Operation `getAOIDWithRef` zurückgeliefert werden.
\$ExternalRef darf keinen Leerstring enthalten.



- Die Filterdefinition wird bei der Registrierung abgewiesen, wenn das Attribut `maptype` nicht den Wert "string" enthält (siehe [Abschnitt „Element <element>“](#)).
- Für \$ExternalRef dürfen die folgenden Systemschlüssel **nicht** angegeben werden:

```
$ContentCode  
$ContentType  
$Signature  
$SignatureDetached  
$SignatureEmbedded  
$SignatureType  
$SignatureQualityLevel  
$SignatureVerification
```

- Pro \$DataNode sind auch mehrere Elemente \$ExternalRef erlaubt. Sie müssen daher nicht mittels \$DataNode getrennt werden.

Beispiel

```
<element mapname="$ExternalRef" xpath="ns:content"/>
```

4.2.2.10 `$ContentCode`

`$ContentCode` gibt die Codierung des Primärdokuments an.

Mögliche Werte: `BASE64` | `BINARY`

`BASE64`

Die mit dem Systemschlüssel `$Content` referenzierten Daten liegen in `BASE64`-Codierung vor.

`BINARY`

Die mit dem Systemschlüssel `$ContentRef` referenzierten Daten liegen nicht codiert vor. Sie werden binär ins SecDocs-Archiv übertragen..



- Falls bei der Archivierung für `$ContentCode` kein Wert ermittelt werden kann, verwendet SecDocs den internen Standardwert `BASE64`.

Die Filterdefinition wird bei der Registrierung in folgenden Fällen abgewiesen:

- Im Filter ist `$Content` oder `$ContentRef` definiert, aber die dazugehörige Angabe von `$ContentCode` fehlt.
- Im Filter ist `$Content` definiert und `$ContentCode` hat den Wert `BINARY`.
- Im Filter ist `$ContentRef` definiert und `$ContentCode` hat den Wert `BASE64`.

Beispiel

```
<element mapname="$ContentCode" default="BASE64"/>
```

4.2.2.11 \$ContentType

\$ContentType gibt das Dateiformat des Primärdokuments und ggf. einer eingebetteten Signatur an.

Mögliche Werte: PDF-A | P7M | OTHERS

PDF-A

Das Primärdokument ist PDF/A-konform.

i SecDocs überprüft nicht, ob das Primärdokument PDF/A-konform ist. Die Angabe dient lediglich als Kennzeichnung.

P7M Das Primärdokument ist konform zu PKCS#7 (Cryptographic Message Syntax Standard).

OTHERS Das Primärdokument liegt in einem anderen Dateiformat vor.

- i**
- Falls bei der Archivierung für \$ContentType kein Wert ermittelt werden kann, verwendet SecDocs den internen Standardwert OTHERS.
 - Die Filterdefinition wird bei der Registrierung abgewiesen, wenn \$Content oder \$ContentRef im Filter definiert ist, aber die dazugehörige Angabe für \$ContentType fehlt.
 - Mit der Angabe von `xpath` können Sie SDO-spezifisch einstellen, welches Dateiformat übergeben wird.

Beispiel

```
<element mapname="$ContentType" default="PDF-A" />
```

4.2.2.12 \$DataNode

\$DataNode gibt den Pfad zu demjenigen Knoten im SDO an, der das Primärdokument und ggf. die Signaturen enthält. Die Angabe im Attribut `xpath` bezeichnet hier einen **absoluten** Pfad im SDO.

Die Angabe von \$DataNode ist nicht nötig, wenn im SDO nur eines der Elemente `$Content` oder `$ContentRef` vorhanden ist.

Für die Angabe von `$ExternalRef` ist die Angabe von \$DataNode nicht erforderlich.

Struktur im SDO

Die Struktur im SDO bestimmt die möglichen Werte für die Attribute `xpath` und `default`. Primärdokumente und zugehörige Signaturen können im SDO in unterschiedlichen XML-Strukturen vorkommen.

Für einen Knoten im SDO, der mit \$DataNode adressiert wird, gelten folgende Regeln:

- Er muss einen der Schlüssel `$Content`, `$ContentRef` oder `$ExternalRef` enthalten.
- Die Schlüssel `$Content` und `$ContentRef` dürfen pro Knoten nur einmal angegeben werden und schließen sich gegenseitig aus.
- Der Schlüssel `$ExternalRef` kann auch mehrfach angegeben werden.
- Falls einer der Schlüssel `$Content` oder `$ContentRef` angegeben ist, müssen folgende Schlüssel ebenfalls angegeben werden:

```
$ContentCode  
$ContentType  
$SignatureEmbedded  
$SignatureQualityLevel  
$SignatureVerification
```

- Falls einer der Schlüssel `$Content` oder `$ContentRef` angegeben ist, darf zusätzlich der Schlüssel `$Signature` angegeben werden. In diesem Fall muss zusätzlich `$SignatureType` angegeben werden.
- Falls keiner der Schlüssel `$Content` oder `$ContentRef` angegeben ist, muss der Schlüssel `$ExternalRef` mindestens einmal angegeben werden und die folgenden Schlüssel dürfen nicht angegeben werden:

```
$ContentCode  
$ContentType  
$Signature  
$SignatureDetached  
$SignatureEmbedded  
$SignatureType  
$SignatureQualityLevel  
$SignatureVerification
```

- Bei allen Schlüsseln, die innerhalb eines \$DataNode definiert sind, muss die Angabe im Attribut `xpath` einen relativen Pfad enthalten.

Weitere Systemschlüssel innerhalb eines Knotens, der mit \$DataNode adressiert ist, werden ignoriert.

Schlüssel, die innerhalb eines `$DataNode` definiert sind, werden nur ausgewertet (z.B. mit einem Default-Wert versehen), wenn der zugehörige Knoten, der mit `$DataNode` adressiert ist, im SDO existiert.

Beispiele

```
<node mapname="$DataNode" xpath="/ns:myDocument/file">
  <element mapname="$Content" xpath="ns:content" maptype="stream"/>
  <element mapname="$ContentCode" default="BASE64"/>
  <element mapname="$ContentType" default="PDF-A"/>
  <element mapname="$SignatureEmbedded" default="AUTO"/>
  <element mapname="$Signature" xpath="ns:signature" maptype="stream"/>
  <element mapname="$SignatureType" default="PKCS7"/>
  <element mapname="$SignatureQualityLevel" default="ADVANCED"/>
  <element mapname="$SignatureVerification" default="SUCCESS"/>
</node>

<node mapname="$DataNode" xpath="/ns:myDocument/file">
  <element mapname="$ContentRef" xpath="ns:contentRef1" maptype="string"/>
  <element mapname="$ContentCode" default="BINARY"/>
  <element mapname="$ContentType" default="PDF-A"/>
  <element mapname="$SignatureEmbedded" default="YES"/>
  <element mapname="$Signature" xpath="ns:signature" maptype="stream"/>
  <element mapname="$SignatureType" default="PKCS7"/>
  <element mapname="$SignatureQualityLevel" default="ADVANCED"/>
  <element mapname="$SignatureVerification" default="SUCCESS"/>
</node>

<node mapname="$DataNode" xpath="/ns:myDocument/file">
  <element mapname="$ExternalRef" xpath="ns:externRef1" maptype="string"/>
</node>

<node mapname="$DataNode" xpath="/ns:myDocument/file">
  <element mapname="$ContentRef" xpath="ns:contentRef2" maptype="string"/>
  <element mapname="$ContentCode" default="BINARY"/>
  <element mapname="$ContentType" default="PDF-A"/>
  <element mapname="$SignatureEmbedded" default="YES"/>
  <element mapname="$Signature" xpath="ns:signature" maptype="stream"/>
  <element mapname="$SignatureType" default="PKCS7"/>
  <element mapname="$SignatureQualityLevel" default="ADVANCED"/>
  <element mapname="$SignatureVerification" default="SUCCESS"/>
  <element mapname="$ExternalRef" xpath="ns:externRef2" maptype="string"/>
  <element mapname="$ExternalRef" xpath="ns:externRef3" maptype="string"/>
  <element mapname="$ExternalRef" xpath="ns:externRef4" maptype="string"/>
</node>
```

4.2.2.13 \$Signature

\$Signature gibt den Pfad zu demjenigen Knoten im SDO an, der einen Signatur-Container (siehe "[Fachwörter](#)") mit den nicht eingebetteten (detached) Signaturen enthält.

Die Angabe im Attribut `xpath` bezeichnet hier

- einen **relativen** Pfad im SDO, wenn \$Signature unterhalb einer Knotendefinition mit dem Element `<node>` verwendet wird. Dieser Pfad ist relativ zur `xpath`-Angabe im Knoten mit `mapname= "$DataNode"` (siehe "[\\$DataNode](#)").
- einen **absoluten** Pfad im SDO, wenn \$Signature ohne Angabe von \$DataNode eingesetzt wird. In diesem Fall müssen auch die folgenden Schlüssel angegeben werden:

```
- $ContentCode
- $ContentType
- $SignatureEmbedded
- $SignatureQualityLevel
- $SignatureType
```

- \$SignatureVerification



- Wenn die Angabe von \$Signature in der Filterdefinition fehlt, wird eine evtl. vorhandene Signatur nicht erkannt und somit nicht geprüft.
- Die Angabe von `maptype="stream"` ist zwingend.
- Wenn der im Attribut `xpath` angegebene Pfad im SDO nicht gefunden wird, so erfolgt eine Archivierung ohne Signaturprüfung.
- \$Signature erfordert zwingend eine Angabe für \$SignatureType (siehe "[\\$SignatureType](#)"). Wenn \$Signature angegeben ist und \$SignatureType fehlt, wird die Filterdefinition bei der Registrierung abgewiesen.

Beispiel

```
<element mapname="$Signature" xpath="ns:signature" maptype="stream" />
```

4.2.2.14 \$SignatureEmbedded

\$SignatureEmbedded gibt an, ob das Primärdokument eine eingebettete Signatur enthält.

Mögliche Werte: YES | NO | AUTO

- YES Das Primärdokument enthält eine oder mehrere eingebettete Signaturen.
Im Systemschlüssel \$ContentType muss angegeben werden, ob es sich um eine PDF/A-konforme oder eine PKCS#7-Signatur handelt (siehe [Abschnitt „\\$ContentType“](#)).
- NO SecDocs soll keine Signaturprüfung für eine evtl. vorhandene eingebettete Signatur durchführen.
- AUTO Das Primärdokument kann eine oder mehrere eingebettete Signaturen enthalten.
Der Systemschlüssel \$ContentType muss angeben, ob es sich um eine PDF/A-konforme oder eine PKCS#7-Signatur handelt.

Enthält das Primärdokument eine oder mehrere Signaturen, werden diese geprüft.



- Falls bei der Archivierung für \$SignatureEmbedded kein Wert ermittelt werden kann, verwendet SecDocs den internen Standardwert NO.
- Das SDO wird bei der Archivierung abgewiesen, wenn default="YES" angegeben ist, das Primärdokument jedoch keine Signatur enthält.
- Mit der Angabe von xpath können Sie SDO-spezifisch einstellen, ob eine eingebettete Signatur vorhanden ist oder nicht.
- Die Filterdefinition wird bei der Registrierung abgewiesen, wenn \$Content oder \$ContentRef im Filter definiert ist, aber die dazugehörige Angabe für \$SignatureEmbedded fehlt.

Beispiel

```
<element mapname="$SignatureEmbedded" default="AUTO" />
```

4.2.2.15 \$SignatureDetached

\$SignatureDetached gibt an, ob SecDocs beim Archivieren eine abgesetzte Signatur zu einem Primärdokument erwartet.

Mögliche Werte: YES | NO | AUTO

- YES Im Primärdokument werden eine oder mehrere abgesetzte Signaturen erwartet.
Der Systemschlüssel \$Signature muss in diesem Fall angegeben werden.
SecDocs führt eine Signaturprüfung für die abgesetzten Signaturen durch.
- NO SecDocs erwartet keine abgesetzten Signaturen und führt auch keine Signaturprüfung für evtl. vorhandene abgesetzte Signaturen durch.
- AUTO Standardwert
SecDocs wertet den Systemschlüssel \$Signature aus, um die abgesetzten Signaturen zu einem Primärdokument zu ermitteln.

Wenn im Primärdokument eine oder mehrere abgesetzte Signaturen gefunden werden, dann werden diese geprüft. Andernfalls erfolgt eine Archivierung ohne Signaturprüfung.

i Wenn \$SignatureDetached den Wert "YES" hat und das Primärdokument keine abgesetzte Signatur enthält, wird das SDO bei der Archivierung abgewiesen.

Mit der Angabe von `xpath` können Sie SDO-spezifisch einstellen, ob SecDocs eine abgesetzte Signatur erwartet oder nicht.

Der Systemschlüssel \$SignatureDetached kann optional pro \$Content oder \$ContentRef angegeben werden.

Ohne Angabe von \$SignatureDetached erfolgt eine Signaturprüfung in Abhängigkeit von Angaben bzgl. des Systemschlüssels \$Signature bzw. von Angaben in der Schema-Definition.

Beispiel

```
<element mapname="$SignatureDetached" default="AUTO" />
```

4.2.2.16 \$SignatureQualityLevel

\$SignatureQualityLevel legt die Anforderungen an die elektronische Signatur bzw. an den Zertifizierungsdiensteanbieter fest, die für eine Archivierung mit SecDocs notwendig sind. Sollen elektronisch signierte Primärdokumente in SecDocs archiviert werden, werden die Signaturen geprüft (Operation `submitSDO`, siehe "[Operation submitSDO](#)", und `replaceSDO`, siehe "[Operation replaceSDO](#)"). Das SDO und das Prüfprotokoll werden nur bei genügend erfolgreicher Verifikation archiviert, andernfalls wird die Archivierung abgewiesen.

Mögliche Werte: `ADVANCED` | `QUALIFIED`

`ADVANCED`

fortgeschrittene elektronische Signatur; diese Signatur ermöglicht eine Identifizierung des Unterzeichners.

`QUALIFIED`

fortgeschrittene elektronische Signatur, die auf einem qualifizierten Zertifikat beruht. Jede `QUALIFIED`-Signatur ist auch `ADVANCED`.



Die Filterdefinition wird bei der Registrierung abgewiesen, wenn `$Content` oder `$ContentRef` im Filter definiert ist, aber die dazugehörige Angabe für `$SignatureQualityLevel` fehlt.

4.2.2.17 \$SignatureType

\$SignatureType gibt die Art der Signatur an, ob es sich also um eine elektronische Unterschrift oder einen Zeitstempel handelt.

Mögliche Werte: PKCS7 | TIMESTAMP

PKCS7

Die Signatur ist PKCS#7-konform.

TIMESTAMP

Der Zeitstempel ist RFC3161-konform.

i Die Angabe von \$Signature in der Filterdefinition erfordert zwingend auch eine Angabe für \$SignatureType. Wenn \$Signature angegeben ist und \$SignatureType fehlt, wird die Filterdefinition bei der Registrierung abgewiesen.

Beispiel

```
<element mapname="$SignatureType" default="PKCS7" />
```

4.2.2.18 \$SignatureVerification

\$SignatureVerification legt fest, welches Ergebnis die Auswertung der Signaturen in einem Dokument mindestens erreichen muss, damit eine Archivierung durchgeführt wird. Das Dokument wird nur archiviert, wenn das Ergebnis der Verifikation höher oder gleich dem Wert des Schlüssels ist.

Es wird grundsätzlich empfohlen, ein SDO nur dann zu archivieren, wenn alle Signaturen vollständig positiv geprüft werden können, also den Wert von \$SignatureVerification auf SUCCESS zu setzen. Wenn die Signaturprüfung auf Sperrlisten basiert, gibt es jedoch Situationen, die als bestes mögliches Ergebnis INFORMATION ergeben:

- Es gibt keine aktuelle CRL, d.h. keine CRL, die nach dem Signaturzeitpunkt ausgestellt wurde, da CRLs nur selten vom Herausgeber aktualisiert werden.
- Die Signatur enthält keinen Signaturzeitpunkt, d.h. Prüfzeitpunkt = Archivierungszeitpunkt. Es kann also nie eine CRL geben, die aktueller ist als der Prüfzeitpunkt.

Ob die Signaturprüfung auf Sperrlisten (CRLs) beruht oder OCSP-Antworten (Online Certificate Status Protocols) verwendet werden, ist abhängig von der Public Key Infrastructure (PKI) des Zertifizierungsdiensteanbieters (ZDA) (siehe Abschnitt „[Verifizierung der Signaturen](#)“).

Mögliche Werte: SUCCESS | INFORMATION | CERTIFICATE

SUCCESS

Alle Prüfungen müssen möglich und erfolgreich sein, d.h. die Verifikation muss ohne Fehler verlaufen, inklusive der Prüfung, ob ein Zertifikat gesperrt ist.

INFORMATION

Alle Prüfungen müssen möglich und erfolgreich sein. Für die Prüfung, ob ein Zertifikat gesperrt wurde, dürfen aber Sperrlisten herangezogen werden, deren Ausgabezeitpunkt vor dem Signatur-Erstellungszeitpunkt liegt. Der Wert ist also nur relevant, wenn zur Signaturprüfung CRLs und keine OCSP-Antworten verwendet werden (siehe Abschnitt „[Verifizierung der Signaturen](#)“).

CERTIFICATE

Die Zertifikatskette muss mindestens ein vertrauenswürdigen Zertifikat enthalten.

Die Einstellung CERTIFICATE dient hauptsächlich zu Testzwecken, da eine Prüfung auf Sperrung des Zertifikates unterbleibt.



- Das SDO wird bei der Archivierung abgewiesen, wenn das Ergebnis der Verifikation kleiner als der Wert des Schlüssels ist.
- Die Archivierung wird mit Fehler abgebrochen, wenn eines der verwendeten Zertifikate gesperrt ist.
- Die Filterdefinition wird bei der Registrierung abgewiesen, wenn \$Content oder \$ContentRef im Filter definiert ist, aber die dazugehörige Angabe für \$SignatureVerification fehlt.

4.2.2.19 \$Index

i In der aktuellen Version wird eine Anbindung an einen Triple-Store nicht unterstützt. Eine erneute Unterstützung eines Triple-Stores ist in Planung.

Mit `$Index` werden die Metadaten ausgewählt, die Sie für eine Recherche verwenden wollen. `$Index` enthält eine Liste von Schlüsselwörtern (System- und benutzerdefinierte Schlüssel), wobei jedem Schlüsselwort bei der `submit`-Operation oder per default kein, ein oder mehrere Werte zugeordnet werden. Die Namen der Schlüsselwörter und deren Werte werden in einem so genannten TripleStore abgelegt und zwar in der folgenden Form:

(AOID des Archivobjekts, Name des Schlüssels, Wert des Schlüssels).

Für ein archiviertes Objekt können so mehrere Tripel entstehen, die alle die gleiche AOID enthalten. Die Recherche erfolgt über logische Kombinationen der Tripel entsprechend der standardisierten Abfragesprache SPARQL. Falls einem Schlüssel mehrere Werte zugeordnet sind, entsteht für jeden Wert ein Tripel mit identischer AOID und Schlüsselbezeichnung.

Für die Wahl der Schlüsselbezeichnungen sollte der Anwender einen für ihn relevanten Namensraum definieren. Beispielsweise sollte die Schlüsselbezeichnung `"Nachname"` in den Filtern aller SDO-Typen verwendet werden, unabhängig davon, wie die Elemente in entsprechenden XML-Schemas heißen. Man schafft sich dadurch ein einheitliches Klassifikationsschema. Wird nun nach Dokumenten gesucht, die eine bestimmte Person betreffen, so erhält man bei der Suche nach `Nachname="Nachname einer bestimmten Person"` alle Dokumente dieser Person und zwar für alle SDO-Typen, bei denen das Metadatum `"Nachname"` indexiert ist.

Sie können eine Liste von benutzerdefinierten Schlüsseln und bestimmten Systemschlüsseln, durch Leerzeichen getrennt, angeben. Diese Liste muss im Attribut `default` von `$Index` angegeben werden (die Verwendung des Attributs `xpath` ist nicht möglich).

Den Systemschlüsseln `$DateFormat`, `$Subject` oder `$SoftWorm` müssen Sie mit `element mapname="..."` explizit einen Wert zuweisen, falls Sie sie verwenden wollen. Die anderen Systemschlüssel werden automatisch durch SecDocs mit Werten versorgt.

SecDocs übernimmt alle Werte eines Schlüssels in den Index, insbesondere mehrfache Werte oder innerhalb einer node-Definition stehende Werte. Bei `$Subject` wird nur der erste gefundene Wert verwendet.

i Der Wert eines Schlüssels, der in den Index übernommen wird, darf maximal 1000 Zeichen enthalten. Andernfalls wird die Operation `submitSDO` bzw. `replaceSDO` abgewiesen. Stellen Sie beim Aufbau des XML-Schemas sicher, dass es zu keiner Überschreitung dieser Länge kommen kann.

Mögliche Werte:

- Liste von benutzerdefinierten Schlüsseln, durch Leerzeichen getrennt

Sie dürfen hier nur Schlüsselwörter angeben, die mit `element mapname="..."` definiert sind und deren Name sich auf folgenden Unicode-Zeichen beschränkt:

```
[A-Z] | [a-z] | [#x00C0-#x00D6] | [#x00D8-#x00F6]
| [#x00F8-#x02FF] | [#x0370-#x037D] | [#x037F-#x1FFF] | [#x200C-#x200D] | [#x2070-#x218F] |
[#x2C00-#x2FEF] | [#x3001-#xD7FF] | [#xF900-#xFDCF] | [#xFDF0-#xFFFD] | [0-9] | '_' | #x00B7 |
[#x0300-#x036F] | [#x203F-#x2040]
```

Im Wesentlichen heißt das, es sind alle Ziffern, Klein- und Großbuchstaben erlaubt, aber keine ASCII-Sonderzeichen außer Unterstrich.

- Zusätzlich können Sie in `$Index` folgende Systemschlüssel angeben (der angegebene Typ kann bei einer Recherche (Operation `sparqlQuery`) genutzt werden):

Systemschlüssel	Typ	Beschreibung
<code>\$ArchiveDate</code>	<code>dateTime</code>	Datum der Ausführung der Operation <code>submitSDO</code> oder <code>replaceSDO</code>
<code>\$COID</code>	<code>string</code>	Client Object Identifier des SDOs
<code>\$DateFormat</code>	<code>string</code>	Format für die Datumsangabe bei Schlüsseln
<code>\$Day</code>	<code>integer</code>	(Monats-)Tag der Archivierung
<code>\$Dayofyear</code>	<code>integer</code>	(Jahres-)Tag der Archivierung
<code>\$ExpirationDateTime</code>	<code>dateTime</code> <code>string</code>	Aktuell gültiger frühest möglicher Löszeitpunkt falls nach einem bestimmten Datum gesucht wird falls nach der Angabe von <code>DEFERRED</code> oder <code>UNLIMITED</code> gesucht wird
<code>\$Hour</code>	<code>integer</code>	Stunde der Archivierung
<code>\$MandantName</code>	<code>string</code>	Mandantenname unter dem die Operation <code>submitSDO</code> oder <code>replaceSDO</code> ausgeführt wird bzw. wurde.
<code>\$Minute</code>	<code>integer</code>	Minute der Archivierung
<code>\$Month</code>	<code>integer</code>	Monat der Archivierung
<code>\$OrgName</code>	<code>string</code>	Organisationsname unter dem die Operation <code>submitSDO</code> oder <code>replaceSDO</code> ausgeführt wird bzw. wurde.
<code>\$Role</code>	<code>string</code>	Rollenname unter dem die Operation <code>submitSDO</code> oder <code>replaceSDO</code> ausgeführt wird bzw. wurde.
<code>\$SdoLength</code>	<code>integer</code>	Die Größe des SDOs in Byte
<code>\$SdoType</code>	<code>string</code>	Der Name des SDOTypes des archivierten SDOs
<code>\$SoftWorm</code>	<code>string</code>	Angabe, ob SoftWORM aktiviert werden soll
<code>\$Subject</code>	<code>string</code>	Frei definierbarer String (z.B. Erläuterungstext zum SDO)

\$Version	string	SecDocs-Version mit der archiviert wird/wurde. Formatbeschreibung siehe Operation statusSDO, ab "Operation statusSDO"
\$VersionNumber	integer	Fortlaufende Nummer zur Identifikation einer SDO-Version.
\$Year	integer	Jahr der Archivierung

Beispiel

Zunächst werden die Elemente definiert, die zur Indexierung verwendet werden sollen:

```
<element mapname="Surname" xpath="..." />
<element mapname="Forename" xpath="..." />
<element mapname="DateOfBirth" xpath="..." />
<element mapname="$Subject" xpath="..." />
...
```

Wenn alle Elemente definiert sind, wird der Index festgelegt:

```
<element mapname="$Index" default=" Surname Forename DateOfBirth $ArchiveDate $Year $Subject" />
```

\$ArchiveDate: Archiving date (Systemschlüssel, wird von SecDocs automatisch versorgt)

\$Year: Year of archiving (Systemschlüssel, wird von SecDocs automatisch versorgt)

\$Subject: Brief description of the SDO (\$Subject muss explizit von Filter/SDO bereitgestellt werden)

4.2.2.20 \$Subject

Frei definierbarer String, der dem SDO zugeordnet wird. Dieser String wird bei verschiedenen Lese-Operationen zusätzlich zur AOID als additive Benutzerinformation zur Verfügung gestellt. Hinweise zur Verwendung von \$Subject finden Sie ab "[Verwendung von Schlüsselwerten](#)".

Wenn \$Subject bei der Default-Angabe Verweise auf andere Schlüssel enthält, so müssen diese Schlüssel zum submitSDO- oder replaceSDO-Zeitpunkt Werte enthalten. Ist dies nicht der Fall, wird die entsprechende submitSDO- bzw. replaceSDO-Operation abgewiesen. Aus diesem Grund wird empfohlen, für diese Schlüssel einen Default-Wert vorzugeben.

4.2.2.21 \$SDOPath

\$SDOPath gibt den Ablageort des SDOs im Archiv an. Der Pfad ist relativ zu dem Pfad, der mit der Operation `createOrganisation` festgelegt wurde (siehe „[Ändern von mandantenspezifischen Einstellungen](#)“).

Mögliche Werte:

Pfadname

Der angegebene Pfadname darf nicht mit einem Schrägstrich (/) oder mit einem Unterstrich (_) beginnen.

Er darf keinen der Strings „./“ oder „/_“ enthalten.

Verweise, die in „*“ eingeschlossen sind, dürfen nur benutzerdefinierte Schlüssel enthalten, die mit `element mapname=“...“` definiert wurden, oder den Namen eines der folgenden Systemschlüssel: `$Day`, `$Dayofyear`, `$Hour`, `$Minute`, `$Month`, `$Year`, `$VersionNumber`, oder eine Kombination aus beiden. Andernfalls wird die Filterdefinition bei der Registrierung abgewiesen.

In einem dateibasierten Storage-System sind in Pfadnamen nicht alle Zeichen zulässig. Daher nimmt SecDocs bei \$SDOPath folgende Ersetzungen vor:

- Wenn der Wert von `$SDOPath` durch die Angabe von `"xpath="` im Filter direkt aus dem SDO geholt wird, so wird folgendermaßen ersetzt:

	ersetzt durch
\	%5c
:	%3a
*	%2a
?	%3f
"	%22
<	%3c
>	%3e
	%7c
%	%25
–	%5f
.	%2e
&	%26
'	%27
!	%21
Leerzeichen	%20
Tabulator	%09
Linefeed	%10
CarriageReturn	%13

- In Pfadbestandteilen, die durch Verweise in begrenzenden „*“ gebildet werden (**Schlüsselname**), werden alle im vorhergehenden Abschnitt aufgeführten Sonderzeichen ersetzt und zusätzlich wird folgendermaßen ersetzt:

	ersetzt durch
/	%2f

- Keine Ersetzung von Sonderzeichen erfolgt für diejenigen Teile von `$SDOPath`, bei denen der Wert mit `default="` bereits direkt im Filter angegeben ist und nicht über eine andere Variable geholt wird. Hier wird die Filterdefinition bei der Registrierung in folgenden Fällen abgewiesen:
 - der Pfadname beginnt mit „/“ oder „_“
 - der Pfadname enthält einen der Strings „../“ oder „/_“
 - der Pfadname enthält eines der folgenden Zeichen:
`\ : * ? " < > | & ' !` Leerzeichen, Tabulator, Linefeed, CarriageReturn

i Das Zeichen „*“ ist nur als Verweis auf einen Schlüssel erlaubt (**Schlüsselname**). Das bedeutet, wenn das Zeichen „*“ in gerader Anzahl im Pfadnamen auftritt, so wird dies als Verweis auf einen oder mehrere Schlüssel interpretiert (z.B. gilt **Key1*Key2*Key3** als Verweis auf die benutzerdefinierten Schlüssel `Key1` und `Key3`). Tritt das Zeichen „*“ hingegen in ungerader Anzahl im Pfadnamen auf, gilt dies als Auftreten eines nicht erlaubten Zeichens, was zur Abweisung der Filterdefinition führt.

i Soll der Wert von `$SDOPath` ein oder mehrere Unterverzeichnis(se) enthalten und enthält somit der Pfadname ein oder mehrere Directory-Trenner, so muss in SecDocs immer der Schrägstrich (/) als Directory-Trenner verwendet werden, unabhängig davon, welches Storage-System tatsächlich im Einsatz ist.

Die Definition von `$SDOPath` im Filter ist obligatorisch.

i Ist ein Benutzerschlüssel angegeben, der beim `submitSDO` bzw. `replaceSDO` nicht existiert, da kein Wert im SDO vorhanden ist, so wird die Operation `submitSDO` bzw. `replaceSDO` abgelehnt. Wird zum `submitSDO`- bzw. `replaceSDO`-Zeitpunkt ein `$SDOPath` mit einem Leerstring versorgt, dann wird das SDO direkt unter dem Pfad hinterlegt, der bei `createOrganisation` angegeben wurde.

Anzahl der archivierten SDOs

Die Anzahl der SDOs, die archiviert werden können, ist durch die Anzahl der möglichen Unterverzeichnisse unter einem Knoten limitiert. Diese Grenze ist abhängig vom verwendeten Storage- und/oder Dateisystem.

Für die Definition des Ablageortes eines SDO wird die `orgID` herangezogen, die im SOAP-Header angegeben ist (siehe [Abschnitt „Operation submitSDO“](#)).

Wenn in der Filterdefinition ein statischer SDO-spezifischer Pfad angegeben ist, gilt die Grenze für den gesamten SDO-spezifischen Knoten. Dieses Problem kann mit Hilfe einer dynamischen Ablagestruktur umgangen werden: Wählen Sie z.B. als Unterstruktur `/$Year/$Month/$Day`, kann für die entsprechende Zahl von Jahren jeden Tag diese Zahl an Dokumenten gespeichert werden.

Beispiele

```
<element mapname="$SDOPath" xpath="/ns:Filter/SignatureData/SDOPath" />
```

```
<element mapname="$SDOPath" xpath="/ns:Filter/SignatureData/ns:@Reference" default="
FILTER" />
```

Die folgende Definition

```
<element mapname="InvoiceNumber"  xpath="..." />
<element mapname="OperationID"  xpath="..." />
<element mapname="DocumentID"  xpath="..." />
<element mapname="$SDOPath"
  default="Invoices/*$Year*/*$Month*/*$Day*/*InvoiceNumber*" />
```

ergibt z.B.

"Invoices/2021/03/04/345-67891"

4.2.2.22 \$SoftWorm

\$SoftWorm gibt an, ob das SDO sowie weitere sicherheitsrelevante Daten bis zum Ende der Aufbewahrungsfrist als SoftWORM-Daten auf dem Storage abgelegt werden.

Mögliche Werte: YES | NO

YES Die Daten werden als SoftWORM-Daten auf dem Storage abgelegt und auf readonly gesetzt. Wenn das Storage-System die SoftWORM-Eigenschaft nicht unterstützt, werden die Daten nur auf read-only gesetzt.

Abhängig von den Einstellungen auf dem Storage-System kann der Zeitpunkt, ab dem die Daten tatsächlich gelöscht werden können, nach dem ermittelten Freigabezeitpunkt liegen.

i Die Angabe `default="YES"` dürfen Sie nicht mit `UNLIMITED` bei `$ExpirationDate` oder `$RetentionPeriod` kombinieren.

NO Standardwert

i Mit der Angabe von `xpath` können Sie die SoftWORM-Funktion SDO-spezifisch einstellen.
\$SoftWorm zeigt nur dann eine Wirkung, falls das verwendete Storage System dies auch unterstützt.

Bei SDOs mit externen Referenz-Ids bezieht sich die Einstellung `$SoftWorm=YES` (Ablegen der Daten als SoftWORM-Daten auf dem Storage-System) nur auf das SDO und die Signature Verification Information. Die zum SDO gehörenden externen Datenobjekte unterliegen hingegen nicht diesem Schutz.

Beispiele

```
<element mapname="$SoftWorm" default="YES"/>
```

```
<element mapname="$SoftWorm" xpath="myDocument/metadata/useSoftWorm" default="Yes">
```

4.2.3 Beispiel für eine Filterdefinition

Das folgende Beispiel zeigt eine einfache Filterdefinition für das XML-Schema `myDocument`, das im [Abschnitt „SDO-Struktur“ \(4.1 SDO-Struktur\)](#) beschrieben ist.

```
<?xml version="1.0" encoding="UTF-8"?>
<filter schema="myDocument" version="1.0" ----- (1)
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="../filter.xsd">
<namespace prefix="ns"
    uri="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"/> ----- (2)
<element mapname="id" xpath="/ns:myDocument/id" default="none"/>
<element mapname="$RetentionPeriod" maptype="string"
    xpath="/ns:document/retentionPeriod" default="P10Y"/> ----- (3)
<element mapname="$SDOPath" default="myDocument/*$Year*/*$Month*/*$Day*" /> ----- (4)
<!-- document node -->
<node mapname="$DataNode" xpath="/ns:myDocument/file"> ----- (5)
<element mapname="$Content" xpath="ns:content" maptype="stream"/> - (6)
<element mapname="$ContentCode" default="BASE64"/> ----- (7)
<element mapname="$ContentType" default="PDF-A" /> ----- (8)
<element mapname="$SignatureEmbedded" default="AUTO" /> ----- (9)
<element mapname="$SignatureQualityLevel" default="ADVANCED" />
<element mapname="$SignatureVerification" default="SUCCESS" />
</node>
<!-- meta data -->
<element mapname="mdAuthor" xpath="/ns:myDocument/metaData/author"
    default="none" /> ----- (10)
<element mapname="mdDepartment" xpath="/ns:myDocument/metaData/department"
    default="none" />
<element mapname="mdCompany" xpath="/ns:myDocument/metaData/company"
    default="none" />
<element mapname="mdCopyright" xpath="/ns:myDocument/metaData/copyright"
    default="none" />
<element mapname="mdDescription" xpath="/ns:myDocument/metaData/description"
    default="none" />
<element mapname="mdKeywords" xpath="/ns:myDocument/metaData/keywords"
    default="none" />
<element mapname="mdCreationDate" xpath="/ns:myDocument/metaData/creationDate"
    default="none" />
<element mapname="mdLastChangedDate" xpath="/ns:myDocument/metaData/lastChangedDate"
    default="none" />
</filter>
```

Erläuterung

- (1) Filterdefinition für das Schema `myDocument`
- (2) Namespace-Zuweisung: Die Kurzbezeichnung `ns` ersetzt die Angabe `http://ts.fujitsu.com/secdocs/sdosamples/mydocument`.
- (3) Die Standard-Aufbewahrungsfrist für Dokumente ist 10 Jahre ab dem Zeitpunkt der Archivierung. Dieser Wert gilt nur, wenn im SDO kein Wert für `$RetentionPeriod` angegeben ist.

-
- (4) Das SDO wird im Archiv unter folgendem Pfad abgelegt: `myDocument/Jahr/Monat/Tag`
Der Pfad ist relativ zum Pfad der Organisationseinheit, der mit der Operation `createOrganisation` festgelegt wurde (siehe "[Operation updateMandant](#)").
 - (5) Definition eines Knotens: Die untergeordneten Elemente zeigen, wie sich der Knoten zusammensetzt.
 - (6) Das Primärdokument liegt relativ zum Pfad `/ns:myDocument/file` (xpath in `$DataNode`) im xpath `ns:content`.
 - (7) Das Primärdokument ist BASE64-codiert.
 - (8) Der Dateityp ist PDF/A.
 - (9) Das Primärdokument kann eine oder mehrere eingebettete Signaturen enthalten.
 - (10) Diese und die folgenden Elementdefinitionen geben die Metadaten an, die mit dem Primärdokument gespeichert werden. Alle Metadaten liegen unter dem xpath `/ns:myDocument/metaData/*`, z.B. `/ns:myDocument/metaData/author`.

4.3 Datenobjekt für die Archivierung

Das zu archivierende Datenobjekt wird in Form eines XML-Containers (dem SDO) an SecDocs übergeben.

Die Archivierung erfolgt mit Hilfe der Operation `submitSDO`.



Eine ausführliche Beschreibung dieser Operation finden Sie im [Abschnitt "Operation submitSDO"](#).

Bereits archivierte Datenobjekte können unter bestimmten Voraussetzungen auch mit der Operation `replaceSDO` ersetzt also erneut archiviert werden.



Eine ausführliche Beschreibung dieser Operation finden Sie im ["Abschnitt "Operation replaceSDO"](#).

Der folgende Text zeigt in einem einfachen Beispiel, wie Sie ein SDO im SOAP-Request der Operation `submitSDO` an SecDocs übergeben.



Das zugehörige XML-Schema finden Sie im Abschnitt unter „Beispiel für ein XML-Schema“ im Abschnitt ["SDO-Struktur"](#), die Filterdefinition im Abschnitt ["Beispiel für eine Filterdefinition"](#).

Beispiel für einen submitSDO Web Service Request

```

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:secdocs="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
    http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <secdocs:soapHeaderData>
      <secdocs:security>
        <secdocs:principal>
          <secdocs:role>Archivar</secdocs:role> ----- (1)
          <secdocs:mandant>Mandant1</secdocs:mandant>
          <secdocs:orgID>Depart1</secdocs:orgID>
        </secdocs:principal>
        <secdocs:password>secrets2</secdocs:password>
      </secdocs:security>
      <secdocs:auditID>Archivar1</secdocs:auditID>
      <secdocs:operation>submitSDO</secdocs:operation> ----- (2)
      <secdocs:aoid>0c9ec4b3-154a-4ec0-9c0d-45624d88be00</secdocs:aoid>
      <secdocs:SDOType>DocumentNone</secdocs:SDOType>
    </secdocs:soapHeaderData>
  </soap:Header>
  <soap:Body>
    <tns:myDocument ----- (3)
      xmlns:tns="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://ts.fujitsu.com/secdocs/sdosamples/
        mydocument myDocument.xsd">
      <tns:id>AKZ-08/15</tns:id>
      <tns:retentionPeriod>P5Y</tns:retentionPeriod> ----- (4)
      <tns:metaData ----- (5)
        <tns:author>Author1</tns:author>
        <tns:department>Department1</tns:department>
        <tns:company>ACME</tns:company>
        <tns:copyright>ACME, 2012</tns:copyright>
        <tns:description>PDF file without any type of
          signature</tns:description>
        <tns:keywords>pdf</tns:keywords>
        <tns:creationDate>2012-07-23</tns:creationDate>
        <tns:lastChangedDate>2012-07-23</tns:lastChangedDate>
      </tns:metaData>
      <tns:file ----- (6)
        <tns:name>testdoc0001.pdf</tns:name>
        <tns:type>PDF</tns:type>
        <tns:content>JVBERi0xLjQgE5pQG5tJMyAwIG9iaiAgPDwvTGVuZ3... -- (7)
          ...XSA+PiANc3RhcnR4cmVmDTcxMA01</tns:content>
      </tns:file>
    </tns:myDocument>
  </soap:Body>
</soap:Envelope>

```


-
- (1) Die Anwendung meldet sich mit der Rolle `Archivar` am Web-Service an.
 - (2) Die Operation `submitSDO` soll ausgeführt werden.
 - (3) Root-Element des SDO-Typs im XML-Schema `myDocument`.
 - (4) Die Aufbewahrungszeit für das Dokument beträgt 5 Jahre.
 - (5) Metadaten, die zusammen mit dem Primärdokument gespeichert werden sollen.
 - (6) Daten des Primärdokuments: Name, Dateityp und Inhalt.
 - (7) Übergebener Inhalt.

5 Archiv- und Mandantenadministration

Dieses Kapitel beschreibt die Administration des Archivs in seiner Gesamtheit und die Administration der spezifischen Archivbereiche (Mandanten).

Dazu bietet SecDocs zwei Web-Services, über die die administrativen Operationen ausgeführt werden:

- Über den Web-Service `ArchiveAdminService` werden die Operationen ausgeführt, die das Archiv in seiner Gesamtheit betreffen (Archivadministrator). Dazu gehören die Definition der Zugangsdaten für die verfügbaren TSPs sowie das Anlegen der verschiedenen Mandanten und der zugehörigen Archivbereiche.
- Über den Web-Service `MandantAdminService` werden die Operationen ausgeführt, die sich auf den jeweiligen Archivbereich eines Mandanten beziehen (Mandantenadministrator). Dazu gehören das Registrieren von SDO-Typen sowie das Einrichten von spezifischen Zugängen zum Archiv für die Client-Software.

SecDocs ist als eigenständiges Archiv realisiert. Es ist also möglich, die Zugriffsinformationen für die archivierten Objekte (sie werden für einen performanten Zugriff in der Datenbank gehalten) alleine mit den auf dem Storage-System befindlichen Archivdaten wieder herzustellen. Für diese Aufgabe steht dem Archivadministrator das im Rahmen der Sec-Docs-Installation installierte Skript `recoverFromStorage` zur Verfügung (siehe [Abschnitt „Recovery \(Skript recoverFromStorage\)“](#)).

5.1 Web-Service ArchiveAdminService

Der folgende Abschnitt beschreibt die Arbeit mit dem Web-Service `ArchiveAdminService` (Webservice für den Archivadministrator). Dieser Web-Service stellt die Operationen zur Verfügung, die das Archiv in seiner Gesamtheit betreffen. Dazu gehört die Definition der Zugangsdaten für die verfügbaren TSPs sowie das Anlegen der verschiedenen Mandanten und zugehörigen Archivbereiche. Der Web-Service `ArchiveAdminService` ist unter folgender URL zu erreichen:

`http://secdocsHost:secdocsPort/archiver/ws/3.1/archiveAdmin`

i Die Grundstruktur einer SOAP-Nachricht ist für alle Web-Services in SecDocs gleich. Einen kurzen Einstieg in den Aufbau von SOAP-Request, SOAP-Response und Fault-Message finden Sie im [Abschnitt „SOAP-Nachrichten“](#).

Der folgende Text beschreibt die Optionen, die für den Web-Service `ArchiveAdminService` abweichend von der Grundstruktur anzugeben sind.

i Alle Beispiele, die bei der Beschreibung der einzelnen Seiten angegeben sind, sind Auzüge aus SOAP-Nachrichten und nicht ohne Modifikationen ablauffähig.

5.1.1 Zugangsprüfung

Um eine Operation des Web-Service `ArchiveAdminService` auszuführen, müssen Sie in den Zugangsdaten die Rolle `ArchiveAdmin` angeben.

Die Zugangsdaten zum Web-Service `ArchiveAdminService` bestehen nur aus der Rolle. Die Operationen beziehen sich auf das Gesamtarchiv, deshalb wird **kein** Teilbereich (`mandant`) und keine Organisationseinheit (`orgID`) benötigt.

5.1.2 Request-Header



Die Grundstruktur eines Request-Headers sowie ein Beispiel finden Sie im [Abschnitt „Request-Header“](#) für den Web-Service ArchivingService.

Der folgende Text listet die Besonderheiten auf, die für den Web-Service ArchiveAdminService zu beachten sind:

Datentyp TSoapHeader

```
<xs:complexType name="TSoapHeader">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="operation" type="tns:TOperation"/>
    <xs:element name="security" type="tns:TSecurity"/>
    <xs:element name="auditID" type="xs:string" minOccurs="0"/>
    <xs:element name="aoid" type="tns:TUuid" minOccurs="0"/>
    <xs:element name="SDOType" type="xs:string" minOccurs="0"/>
    <xs:element name="coid" type="tns:TCoid"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

operation	Operation, die ausgeführt werden soll: Datentyp TOperation, siehe " Datentyp TOperation "
security	Autorisierungsinformation für die Anmeldung am Web-Service (Zugangsdaten und Passwort): Datentyp TSecurity, siehe " Datentyp TSecurity "
auditID	string: optional; Diese Zeichenkette wird in das Audit-Protokoll übertragen. Die Client-Software meldet sich bei SecDocs mit einer Rolle an. Um im Audit-Protokoll nachvollziehen zu können, wer der ausführende Benutzer war, kann die Client-Software hier z.B. die Anmeldedaten dieses Benutzers angeben.
aoid	Dieses Element darf für die Operationen des Web-Service ArchiveAdminService nicht angegeben werden.
SDOType	Dieses Element darf für die Operationen des Web-Service ArchiveAdminService nicht angegeben werden.
coid	Dieses Element darf für die Operationen des Web-Service ArchiveAdminService nicht angegeben werden.

Datentyp TOperation

```

<xs:simpleType name="TOperation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="createTSP" />
    <xs:enumeration value="createMandant" />
    <xs:enumeration value="createMandantXAIP" />
    <xs:enumeration value="setCredentials" />
    <xs:enumeration value="getTSPs" />
    <xs:enumeration value="updateTSP" />
    <xs:enumeration value="getMandants" />
    <xs:enumeration value="getHashAlgorithms" />
    <xs:enumeration value="getSignatureAlgorithms" />
    <xs:enumeration value="getVersion" />
    <xs:enumeration value="getAccountingData" />
    <xs:enumeration value="getStatisticalData" />
    <xs:enumeration value="deleteTestmandant" />
    <xs:enumeration value="performAction" />
    <xs:enumeration value="getArchiveInfo" />
  </xs:restriction>
</xs:simpleType>

```

Für den Web-Service ArchiveAdminService sind folgende Operationen möglich:

createTSP	Anlegen eines neuen TSPs, siehe "Operation createTSP"
createMandant	Anlegen eines neuen Mandanten, siehe "Operation createMandant"
createMandantXAIP	Anlegen eines neuen Mandanten für den Web-Service S4, siehe Handbuch "SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR" (SD2) in Abschnitt "Literatur"
setCredentials	Setzen der Zugangsparameter, siehe "Operation setCredentials"
getTSPs	Auflisten der existierenden TSPs, siehe "Operation getTSPs"
updateTSP	Ändern der bestehenden Eigenschaften eines TSP, siehe "Operation updateTSP"
getMandants	Auflisten der existierenden Mandanten, siehe "Operation getMandants"
getHashAlgorithms	Auflisten der verfügbaren Hash-Algorithmen, siehe "Operation getHashAlgorithms"
getSignatureAlgorithms	Auflisten der existierenden Signatur-Algorithmen, siehe "Operation getSignatureAlgorithms"
getVersion	Auflisten von Versionsinformationen, siehe "Operation getVersion"
getAccountingData	Zur Verfügung stellen von Abrechnungsinformationen für den Archivadministrator zur Abrechnung gegenüber seinen Mandanten, siehe "Operation getAccountingData"
getStatisticalData	Ausgeben von Statistikdaten für alle Mandanten und für alle Operationen der Web-Services ArchivingService und S4, siehe "Operation getStatisticalData"
deleteTestmandant	Löschen eines Testmandanten, siehe "Operation deleteTestMandant"
performAction	Auslösen von Aktionen auf dem Archiv, siehe "Operation performAction"

getArchiveInfo

Liefern von Informationen zum aktuellen Zustand der ereignisgesteuerten Aufträge, siehe "[Operation getArchiveInfo](#)"

Datentyp TSecurity

```
<xs:complexType name="TSecurity">
  <xs:sequence>
    <xs:element name="principal" type="TPrincipal"/>
    <xs:choice>
      <xs:element name="password" type="xs:string"/>
      <xs:element name="token" type="xs:base64Binary"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

principal Basisdaten für die Zugangsprüfung:
Datentyp TPrincipal, siehe "[Datentyp TPrincipal](#)"

password string:
ArchiveAdmin-spezifisches Passwort.

Passwort nach der Erstinstallation: AA_initial

! Das voreingestellte Passwort für die Rolle ArchiveAdmin sollte sofort nach der Erstinstallation von SecDocs mit der Operation setCredentials geändert werden (siehe [Abschnitt „Operation setCredentials“](#)).

token base64Binary:
Eine Token-Angabe ist derzeit nicht realisiert. Es muss der Passwort-Mechanismus verwendet werden. Das Token-Verfahren ist in einer Folgeversion zur Autorisierung vorgesehen.

Datentyp TPrincipal

```
<xs:complexType name="TPrincipal">
  <xs:sequence>
    <xs:element name="role" type="xs:string"/>
    <xs:element name="mandant" type="xs:string" minOccurs="0"/>
    <xs:element name="orgID" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

role ArchiveAdmin.

mandant Dieses Element darf für die Operationen des Web-Service ArchiveAdminService nicht angegeben werden.

orgID Dieses Element darf für die Operationen des Web-Service ArchiveAdminService nicht angegeben werden.

5.1.3 Request- und Response-Body

Der Aufbau von SOAP-Request und SOAP-Response ist operationsspezifisch.

- Der Web-Service `ArchiveAdminService` ist in der Datei `ArchiveAdmin.wsdl` beschrieben.
- Definition der Typen: `AdminData.xsd`

5.1.3.1 Operation createTSP

`createTSP` legt fest, welcher Zeitstempelanbieter (TSP) und damit, welcher Hash- und welcher Signatur-Algorithmus für das Versiegeln eines SDOs verwendet werden kann.

i Es wird empfohlen, nur TSPs zu verwenden, die offiziell durch Ämter anerkannt sind. Eine Liste von in Deutschland zugelassenen TSPs finden Sie z.B. unter <https://webgate.ec.europa.eu/tl-browser/#/tl/DE>. Nach Möglichkeit sollten Sie mindestens zwei TSPs definieren, die hinsichtlich des verwendeten Hash- und Signatur-Algorithmus verschieden sind. Beim Anlegen eines Mandanten können Sie dann festlegen, welcher der TSPs zum Versiegeln der SDOs verwendet werden soll (Operation `createMandant`, siehe "Operation `createMandant`"). Um zu verhindern, dass archivierte SDOs zeitweise nur schwach versiegelt sind, z.B. weil ein Algorithmus seine Schutzwirkung verloren hat, sollte jedes SDO mit mindestens zwei verschiedenen Verfahren versiegelt werden (siehe Abschnitt "Signaturneuerung gemäß ArchiSig-Konzept" im Kapitel "Kryptografische Verfahren").

Bei Aufruf von `createTSP` wird, um die Korrektheit der Angaben zu prüfen, vom angegebenen Timestamp-Provider ein Zeitstempel geholt und ein Evidence Record erzeugt; es sei denn, der Administrator des Archivs hat den Konfigurationsparameter `checkTSP` (in der Konfigurationsdatei `secdocs.properties`) auf `false` gesetzt.

Wenn die daran anschließende Verifikation des Evidence Records (und damit die Operation `createTSP` insgesamt) fehlschlägt, so wird in der Fault-Message im Element `additionalData` das Verifikationsprotokoll des CryptoModule mit allen Detail-Angaben für die Signaturprüfung mit ausgegeben.

i Für die Erzeugung von Evidence Records werden folgende Hash Algorithmen von SecDocs unterstützt:

- SHA-256
- SHA-384
- SHA-512

Für die Validierung werden folgende Hash Algorithmen von SecDocs unterstützt:

- SHA-1
- SHA-224
- SHA-256
- SHA-384
- SHA-512
- RIPEMD-160

Voraussetzungen

Die Systemadministration muss sicherstellen, dass die Verbindung zum TSP verfügbar ist (z.B. durch entsprechende Einstellungen in der Firewall).

Request-Body

Datentyp TSPTyp

```

<xsd:complexType name="TSPTType">
  <xsd:complexContent>
    <xsd:extension base="tns:BaseType">
      <xsd:sequence>
        <xsd:element name="Name" maxOccurs="1" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:NCName">
              <xsd:maxLength value="50"></xsd:maxLength>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="ProviderName" type="tns:NonEmptyString"
          minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="URL" type="tns:NonEmptyString"
          minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="SigHashAlgorithmName" type="xsd:string"
          minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="HashAlgorithmName" type="xsd:string"
          minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="SignatureAlgorithmName" type="xsd:string"
          minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="KeyLength" maxOccurs="1" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:integer">
              <xsd:maxInclusive value="2147483647">
              </xsd:maxInclusive>
              <xsd:minInclusive value="1">
              </xsd:minInclusive>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="CertificateName" type="xsd:string"
          minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="SignatureQuality" minOccurs="1"
          maxOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="ADVANCED">
              </xsd:enumeration>
              <xsd:enumeration value="QUALIFIED">
              </xsd:enumeration>
              <xsd:enumeration value="ACCREDITED">
              </xsd:enumeration>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="isActive" type="xsd:boolean"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="Policy" maxOccurs="1" minOccurs="0">
          <xsd:simpleType>
            <xsd:restriction base="xsd:base64Binary">
              <xsd:minLength value="1"></xsd:minLength>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<!-- if not present, doEnrich will be set to false -->
<xsd:element name="doEnrich" type="xsd:boolean"
    maxOccurs="1" minOccurs="0">
    </xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

```

Die Beschreibung des Datentyps BaseType finden Sie auf "[Operation createTSP](#)".

Name

NCName:

Name des TSPs in SecDocs. Dieser Name muss im Archiv eindeutig sein. Dabei ist die Groß-/Kleinschreibung nicht relevant. Der Name darf keine Leerzeichen enthalten.

Maximale Länge:

50 Zeichen

Erlaubte Zeichen:

eingeschränkter XML-Name, der mit einem Buchstaben oder einem Unterstrich beginnen muss, auf den Namenszeichen (Buchstaben, Ziffern und andere Zeichen) folgen.

Der Doppelpunkt ist ausgeschlossen.

Die explizite Definition der erlaubten Zeichen finden Sie z.B. unter <http://www.w3.org/TR/2009/REC-xml-names-20091208>.

ProviderName

NonEmptyString:

Name des TSPs (z.B. Firmenname, Name des Instituts oder der Behörde).

ProviderName kann beliebig gewählt werden.

Beispiel

Von einem TSP werden Zeitstempel für SHA-256 und für SHA-384 angefordert.

Dann ist folgende Konfiguration möglich:

Name	HashAlgorithmName	ProviderName
TSP1	SHA-256	ProviderXY
TSP2	SHA-384	ProviderXY

URL

string:

URL des Zeitstempel-Dienstes.

Zertifikatsbasierte Authentifizierung

Wird vom Timestamp-Provider verlangt, dass bei der Zeitstempelanforderung eine signierte Nachricht gesendet wird, dann muss hier das dazu erforderliche Zertifikat mit Passwort übergeben werden (das ist z.B. beim Timestamp-Provider D-Trust erforderlich). Dies geschieht durch Erweitern des URL-Parameters.

Diese zusätzlichen Informationen sind in der folgenden Form in einer einzigen Zeile anzugeben. Die einzelnen Parameter sind getrennt durch #####.

```
https://time.d-trust.net#####  
password=password#####  
certificate=certFileContent
```

password	Base64-kodiert Passwort für den Zugriff auf den Inhalt der PKCS12-Datei, die bei certFileContent angegeben ist.
certFileContent	Base64-kodiert Inhalt der PKCS12-Datei, die das Schlüsselpaar (Private/Public) enthält.

Beispiel

```
<URL>https://testtime.d-trust.  
net#####password=eHl6#####certificate=WmVydGlmaXppZXJlbmdzZGF0ZWk=  
/URL>
```

HTTP Basic Authentifizierung

Wird vom Timestamp-Provider verlangt, dass bei der Zeitstempelanforderung die „HTTP Basic Authentication“ verwendet wird, dann muss hier der dazu erforderliche Username mit Passwort übergeben werden (das ist z.B. bei dem Timestamp-Provider Postsignum erforderlich). Dies geschieht durch Erweitern des URL-Parameters.

Diese zusätzlichen Informationen sind in der folgenden Form in einer einzigen Zeile anzugeben. Die einzelnen Parameter sind getrennt durch #####.

```
https://URLdesZeitstempeldienstes#####  
username=username#####  
userpassword=userpassword
```

username	Base64-codiert Benutzername für HTTP Basic Authentication
userpassword	Base64-codiert Passwort für HTTP Basic Authentication

Beispiel

```
<URL>https://test.timestamp-provider-address  
/#####username=dGVzdHVzZXI=#####userpassword=dGVzdHBhc3M=</URL>
```

SigHashAlgorithmName

string:

Der Hash-Algorithmus, der vom TSP verwendet wird, um den Zeitstempel zu erstellen.
Der Algorithmus wird vom TSP vorgegeben.

i Der angegebene Hash-Algorithmus muss verfügbar sein (siehe [Abschnitt „Operation getHashAlgorithms“](#)).

HashAlgorithmName

string:

Algorithmus, der bei der Versiegelung verwendet wird, um einen Hash-Wert für das übergebene SDO zu bilden.

i Der angegebene Hash-Algorithmus muss vom TSP unterstützt werden (siehe [Abschnitt „Operation getHashAlgorithms“](#)).

SignatureAlgorithmName

string:

Genutzter Signatur-Algorithmus, wie vom TSP vorgegeben.
Muss mit dem verwendeten Algorithmus in der Zertifikatssignatur übereinstimmen.

i Der angegebene Signatur-Algorithmus muss verfügbar sein (siehe [Abschnitt „Operation getSignatureAlgorithms“](#)).

KeyLength

integer:

Länge des Signatur-Algorithmus, wie vom TSP vorgegeben.
Muss mit der Schlüssellänge in der Zertifikatssignatur übereinstimmen.

Mögliche Werte: 1 - 2147483647

CertificateName	<p>string:</p> <p>Name des Erstellerzertifikats, wie vom TSP vorgegeben.</p> <p>Aus dem Zertifikat, das zum Signieren des Zeitstempels vom TSP verwendet wird (beim TSP zu erfragen), ist ein Teil-String des Feldes „Antragssteller“ (Subject) anzugeben. Die Prüfung erfolgt nicht case-sensitiv.</p> <div> <p>i Ein TSP verwendet meist mehrere Signiereinheiten, deshalb ist der Wert des Feldes „Antragssteller“ über alle möglichen Zertifikate eines TSPs nicht konstant. Wählen Sie deshalb nur einen Teilstring, der über alle verwendeten Zertifikate konstant ist.</p> <p>Der Wert des Feldes „Antragssteller“ ist strukturiert in: CN=<string>, OU=<string>, O=<string>, C=<string>. Die Reihenfolge dieser Teilfelder kann variieren. Es wird deshalb empfohlen, nur ein (konstantes) Teilfeld als CertificateName zu verwenden.</p> </div>				
SignatureQuality	<p>string:</p> <p>Qualität der Signatur, die der Zeitstempelanbieter (TSP) verwendet, um seine Zeitstempel zu signieren. Es können verschieden hohe Anforderungen an das Zertifikat gestellt werden, das der TSP für die Signatur der Zeitstempel verwendet. Diese Einstufung ist abhängig vom ausgewählten TSP. Informationen über die einzelnen TSPs sind in einer Liste bei der Bundesnetzagentur erfasst.</p> <p>Mögliche Werte:</p> <table> <tr> <td>"ADVANCED"</td><td>Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur.</td></tr> <tr> <td>"QUALIFIED"</td><td>Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur, die auf einem qualifizierten Zertifikat beruht.</td></tr> </table>	"ADVANCED"	Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur.	"QUALIFIED"	Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur, die auf einem qualifizierten Zertifikat beruht.
"ADVANCED"	Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur.				
"QUALIFIED"	Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur, die auf einem qualifizierten Zertifikat beruht.				
isActive	<p>boolean:</p> <p>Angabe, ob der TSP bei der Versiegelung der archivierten Dokumente verwendet wird.</p> <p>Mögliche Werte:</p> <table> <tr> <td>"true"</td><td>Der TSP wird bei der Versiegelung verwendet</td></tr> <tr> <td>"false"</td><td>Der TSP wird nicht verwendet.</td></tr> </table> <p>Standardwert: true</p> <div> <p>i Derzeit dürfen Sie nur der Wert true angeben. false wird als Fehler abgewiesen.</p> </div>	"true"	Der TSP wird bei der Versiegelung verwendet	"false"	Der TSP wird nicht verwendet.
"true"	Der TSP wird bei der Versiegelung verwendet				
"false"	Der TSP wird nicht verwendet.				

Policy

base64Binary:
Datei zur Steuerung der Bewertung der Signaturprüfung der vom TSP gelieferten
Zeitstempel

! Das Erstellen einer Policy Datei ist eine Service-Leistung der Fujitsu GmbH und sollte mit ihrem technischen Betreuer abgestimmt werden.

doEnrich

boolean:
optional; Gibt an ob Revocation-Info und Zertifikate des Zeitstempelanbieters in den
Evidence-Record eingebettet werden.

Mögliche Werte:

"true" Die Informationen werden eingebettet.

"false" Die Informationen werden erst beim Renew eingebettet

Standardwert: false

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request


```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>createTSP</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>ArchiveAdmin</sdsh:role>
      </sdsh:principal>
      <sdsh:password>SecretPasswordOfArchiveAdmin</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>Administrator1</sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <TSP xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Name>TSP-DFN</Name>
    <ProviderName>Deutsches Forschungs Netz (DFN)</ProviderName>
    <URL>http://zeitstempel.dfn.de/</URL>
    <SigHashAlgorithmName>SHA-1</SigHashAlgorithmName>
    <HashAlgorithmName>SHA-256</HashAlgorithmName>
    <SignatureAlgorithmName>rsaEncryption1
    </SignatureAlgorithmName>
    <KeyLength>2048</KeyLength>
    <CertificateName>DFN-Verein</CertificateName>
    <SignatureQuality>ADVANCED</SignatureQuality>    </TSP>
</soap:Body>
```

Response

```
<soap:Body>
  <result></result>
</soap:Body>
```

5.1.3.2 Operation createMandant

`createMandant` legt einen neuen Mandanten im Archiv an. Dazu gehören:

- Definition des Mandanten: Name und Kontaktinformation
- Definition des zugehörigen Archivbereichs: Dateipfade direkt unterhalb der Archiv-Wurzel und ggf. der Archiv-Wurzel für externe Dokumente, unter denen alle Dateien und Verzeichnisse für diesen Mandanten auf dem Speichersystem abgelegt werden (mandantenspezifischer Archivbereich). Die Archiv-Wurzeln werden mit den Parametern `archiveRoot` und `archiveRootExternalFiles` in der Datei `secdocs.properties` konfiguriert, siehe "[Konfigurationsdatei secdocs.properties](#)".
- Konfigurationsparameter `TreeSize` und `TreeAge`
- Für die Versiegelung zu nutzende TSPs
- Zugang (Credentials) für die Rolle `MandantAdmin`
- Einstellen, ob ein produktiver Mandant oder ein Testmandant erstellt wird
- Erzeugen der Rolle `SecDocs_MandantAuditor`
- Einstellen von mandantenspezifischen Konfigurationsparametern

Voraussetzungen

- Die TSPs müssen in SecDocs angelegt sein, Operation `createTSP` (siehe "[Operation createTSP](#)").
- Die Pfade `archiveRoot/Path` und ggf. `archiveRootExternalFiles/Path` müssen entweder existieren und mit Schreibrecht (für den Linux-Benutzer 'secdocs') zugreifbar sein (eventuell auch als Mount in Absprache mit dem Systemadministrator), oder sie können durch `createMandant` angelegt werden, falls dies im Parameter `createMandantDirLocal` in der Datei `secdocs.properties` eingestellt ist (siehe "[Konfigurationsdatei secdocs.properties](#)").

i Die Operation `createMandant` wird abgewiesen, wenn für einen mandantenspezifischen Konfigurationsparameter ein ungültiger Wert angegeben wird, z.B. ein kleinerer Wert als der festgelegte Minimalwert bzw. ein größerer Wert als der festgelegte Maximalwert.

Request-Body

Die folgende Grafik zeigt die Struktur des Requests `createMandantType` sowie die Datentypen `MandantType` und `CredentialType`.

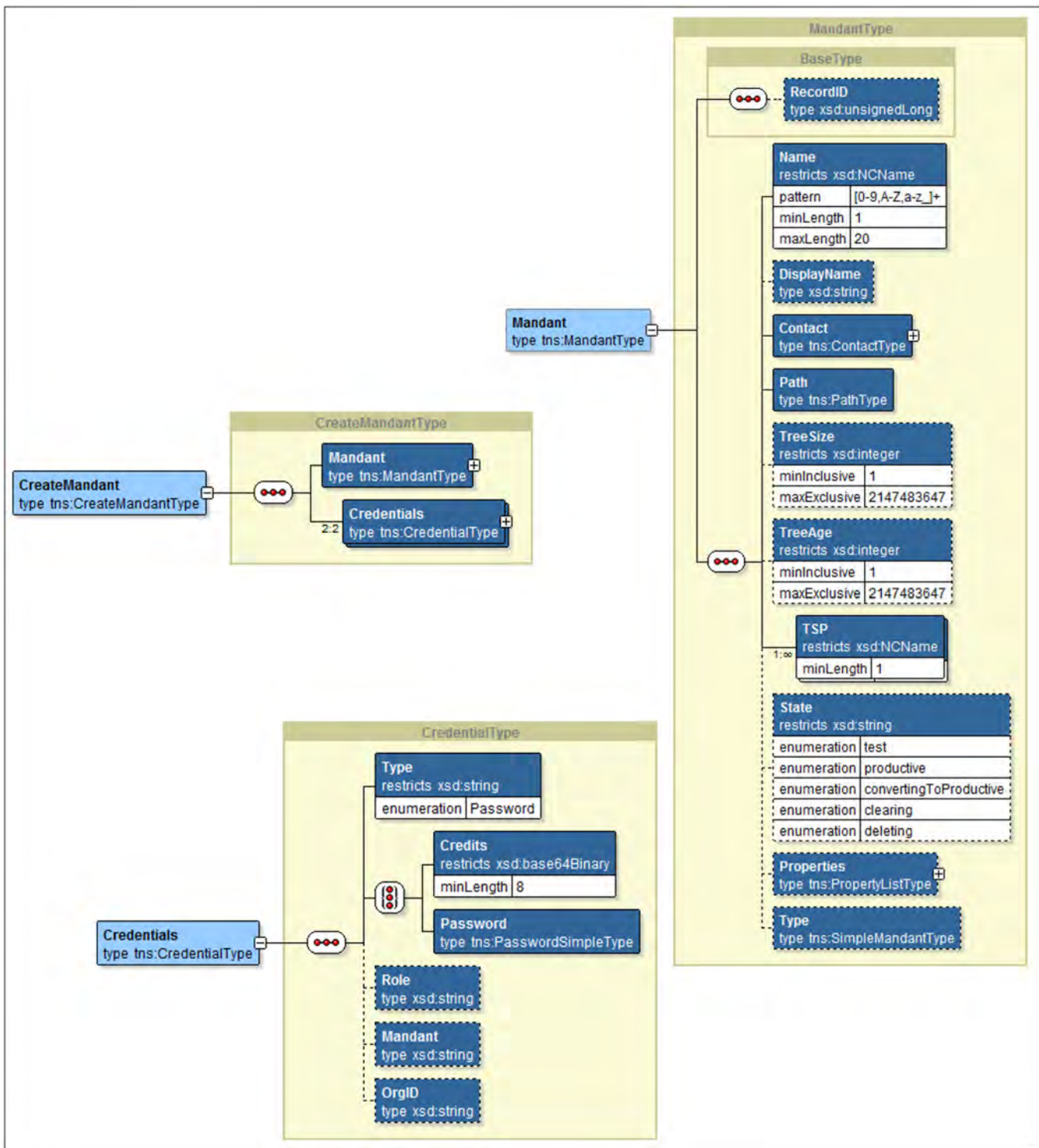


Bild 13: Struktur von `createMandantType`



Sequence: die Elemente müssen genau in der dargestellten Reihenfolge auftreten.



Choice: eines der angegebenen Elemente

— Pflichtangabe

----- Optionale Angabe

Datentyp CreateMandantType

```
<xsd:complexType name="CreateMandantType">
  <xsd:sequence>
    <xsd:element name="Mandant" type="tns:MandantType"
      minOccurs="1" maxOccurs="1">
    </xsd:element>
    <xsd:element name="Credentials" type="tns:CredentialType"
      minOccurs="2" maxOccurs="2">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Mandant Basisdaten für den Mandanten (Name, Kontaktinformation, Archivbereich, Konfigurationsparameter, Liste der TSPs):
Datentyp MandantType, siehe "[Datentyp MandantType](#)"

Credentials Zugangsdaten:
Jeweils ein Datentyp CredentialType für die Rolle MandantAdmin und die Rolle Archivar, siehe "[Datentyp CredentialType](#)"

Datentyp MandantType

```
<xsd:complexType name="MandantType">
  <xsd:complexContent>
    <xsd:extension base="tns:BaseType">
      <xsd:sequence>
        <xsd:element name="Name">
          <xsd:simpleType>
            <xsd:restriction base="xsd:NCName">
              <xsd:pattern value="[0-9,A-Z,a-z_]+">
            </xsd:pattern>
            <xsd:minLength value="1"></xsd:minLength>
            <xsd:maxLength value="20"></xsd:maxLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element name="DisplayName" type="xsd:string"
        minOccurs="0" maxOccurs="1">
      </xsd:element>
      <xsd:element name="Contact" type="tns:ContactType"
        minOccurs="1">
      </xsd:element>
      <xsd:element name="Path" minOccurs="1" type="tns:PathType">
      </xsd:element>
      <xsd:element name="TreeSize" minOccurs="0" maxOccurs="1">

```

```

        <xsd:simpleType>
            <xsd:restriction base="xsd:integer">
                <xsd:minInclusive value="1">
                </xsd:minInclusive>
                <xsd:maxExclusive value="2147483647">
                </xsd:maxExclusive>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element name="TreeAge" minOccurs="0" maxOccurs="1">
        <xsd:simpleType>
            <xsd:restriction base="xsd:integer">
                <xsd:minInclusive value="1">
                </xsd:minInclusive>
                <xsd:maxExclusive value="2147483647">
                </xsd:maxExclusive>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element name="TSP" maxOccurs="unbounded" minOccurs="1">
        <xsd:simpleType>
            <xsd:restriction base="xsd:NCName">
                <xsd:minLength value="1"></xsd:minLength>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element name="State" maxOccurs="1" minOccurs="0">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="test"/>
                <xsd:enumeration value="productive"/>
                <xsd:enumeration value="
                    convertingToProductive"/>
                <xsd:enumeration value="clearing"/>
                <xsd:enumeration value="deleting">
                </xsd:enumeration>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Properties" type="tns:PropertyListType"
        minOccurs="0" maxOccurs="1">

    </xsd:element>,
    <xsd:element name="Type" type="tns:SimpleMandantType"
        minOccurs="0" maxOccurs="1">

    </xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

```

Die Beschreibung des Datentyps BaseType finden Sie auf ["Operation createTSP"](#).

Name	<p>NCName: Name des Mandanten. Der Name darf keine Leerzeichen enthalten.</p> <p>Die Groß-/Kleinschreibung ist für den Namen nicht relevant.</p> <p>Maximale Länge: 20 Zeichen</p> <p>Erlaubte Zeichen: Eingeschränkter XML-Name, der mit einem Buchstaben oder einem Unterstrich beginnen muss, auf den Namenszeichen (Buchstaben, Ziffern und andere Zeichen) folgen. Der Doppelpunkt ist ausgeschlossen. Die explizite Definition der erlaubten Zeichen finden Sie z.B. unter http://www.w3.org/TR/2009/REC-xml-names-20091208. Zusätzlich wird der NCName auf die Zeichen [0-9 , A-Z , a-z _] + eingeschränkt.</p>
DisplayName	<p>string: optional; Name des Mandanten, wie er ggf. in SecDocs angezeigt wird.</p> <p>DisplayName unterliegt nicht den Einschränkungen des Typs NCName</p>
Contact	<p>Kontaktinformation für den Mandanten: Datentyp <code>ContactType</code>, siehe "Datentyp ContactType".</p>
Path	<p>Datentyp <code>PathType</code>:</p> <p>Pfad für den Teilbereich des Mandanten im Archiv. Alle SDOs werden unter diesem Pfad abgelegt.</p> <p>Erlaubte Zeichen: Der Pfadname darf mit keinem der folgenden Zeichen beginnen: Schrägstrich (/), Unterstrich (_) oder Punkt(.). Der Pfadname darf keinen der Strings "/_", ". ./", ". ./" oder "\" enthalten. Der Pfadname darf für keinen anderen Mandanten vergeben worden sein.</p>

integer:

TreeSize

optional; legt fest, nach wie vielen gespeicherten SDOs die Versiegelung angestoßen wird.

Dazu ermittelt SecDocs alle zu versiegelnden Dokumente:

- Wenn die Anzahl der gefundenen Dokumente \leq `maxTreeSize` ist, wird ein Hash-Baum mit allen gefundenen SDOs gebildet und mit einem Zeitstempel versiegelt.
- Wenn die Anzahl der gefundenen Dokumente $>$ `maxTreeSize` ist, werden mehrere Hash-Bäume gebildet mit jeder durch `maxTreeSize` festgelegten Anzahl an SDOs und mit je einem Zeitstempel versiegelt. Abhängig vom Parameter `maxParallelHashtrees` (siehe auch "[Konfigurationsdatei secdocs.properties](#)") werden die Hash-Bäume entweder parallel oder sequenziell versiegelt:
 - Wenn `maxParallelHashtrees=1`, wird sequenziell versiegelt.
 - Wenn `maxParallelHashtrees>1`, wird parallel versiegelt.

Standardwert: `maxTreeSize` (definiert in der Datei `secdocs.properties`, siehe "[Konfigurationsdatei secdocs.properties](#)")

Wertebereich: 1 - 2147483646

Wenn nach einer durchgeführten Versiegelung die `TreeAge`-Zeit abgelaufen ist, wird eine Versiegelung gestartet, auch wenn die `TreeSize`-Anzahl noch nicht erreicht wurde.

TreeAge

integer:

optional; Wartezeit in Minuten bis eine Versiegelung mit einem Zeitstempel gestartet wird. Die Zeitstempel werden nicht pro SDO angefordert. Die SDOs werden gesammelt und dann gemeinsam mit einem Zeitstempel versiegelt (siehe Abschnitt „[Signaturerneuerung gemäß ArchiSig-Konzept](#)“).

Wenn die Anzahl der gefundenen Dokumente `> maxTreeSize` ist, werden mehrere Hash-Bäume gebildet mit je der durch `TreeSize` festgelegten Anzahl an SDOs und mit je einem Zeitstempel parallel oder sequentiell versiegelt (siehe „[TreeSize](#)“).

Standardwert: 720 Minuten

Wertebereich: 1 - 2147483646

Der Monitor zur Überwachung von `TreeAge` (Überwachung der maximalen Wartezeit eines SDOs auf Versiegelung) ist im Multi-Node-Betrieb nach jedem Start von SecDocs deaktiviert. Aus diesem Grund wird empfohlen, im Multi-Node-Betrieb nur den Konfigurationsparameter `TreeSize` zu verwenden.

Sollte in einer betriebsarmen Zeit dann der Fall eintreten, dass der Schwellenwert `TreeSize` für längere Zeit nicht erreicht wird, so gibt es zwei mögliche Vorgehensweisen:

- Explizites Starten des Versiegelungsvorgangs(`Operation performAction` und `Action=sealDocuments`)

Diese Vorgehensweise empfiehlt sich, wenn für längere Zeit kein Auftreten eines `submitSDO` erwartet wird.

Beachten Sie, dass die Aktion `sealDocuments` auf derjenigen SecDocs-Instanz durchgeführt wird, auf der die `Operation performAction` ausgeführt wurde.
- Manuelles Starten des Monitors zur Überwachung von `TreeAge` (`Operation performAction`, `Action=startMonitor` und `MonitorType=TSPTimer`)

Dieser Monitor startet auf demjenigen Server im Verbund, auf dem die `Operation performAction` ausgeführt wurde.

Diese Vorgehensweise empfiehlt sich, wenn auch in einer betriebsarmen Zeit eine periodische Überwachung erwünscht ist, aber ein explizites Anstarten des Versiegelungsvorgangs vermieden werden soll.

Siehe hierzu auch [Abschnitt „Operation performAction“](#) und [Abschnitt „Operation performAction“](#).

TSP

NCName:

Liste von mindestens einem TSP. Die Namen der TSPs wurden mit der `Operation createTSP` festgelegt, siehe "[Operation createTSP](#)".

State	<p>string: optional. Wird State in der Nachricht nicht angegeben, so wird ein produktiver Mandant eingerichtet.</p> <p>Mögliche Werte:</p> <p>test richtet einen Mandanten als Testmandanten ein.</p> <p>productive richtet einen produktiven Mandanten ein.</p> <p>Ein Mandant kann nur beim Anlegen zu einem Testmandanten erklärt werden.</p>
Properties	<p>Datentyp PropertyListType optional; Liste von mandantenspezifischen Konfigurationsparametern, die für diesen Mandanten gesetzt sind (siehe Abschnitt „Mandantenspezifische Konfigurationsparameter“).</p>
Type	<p>Format der Dokumente, die der Mandant archivieren kann. Dieses Format legt auch den Web-Service fest, den der Mandant zum Archivieren nutzen kann.</p> <p>Dieses Element wird von SecDocs automatisch gesetzt und darf beim Aufruf der Operation nicht angegeben werden. Es dient als Ausgabeinformation bei den Operationen <code>getMandant</code>, <code>getMandantProperties</code> und <code>getArchiveInfo</code>.</p>

Element Properties vom Datentyp PropertyListType

```
<xsd:complexType name="PropertyListType">
  <xsd:sequence>
    <xsd:element name="Property" type="tns:PropertyType"
                  minOccurs="0" maxOccurs="unbounded">

    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Property **Datentyp** PropertyType
optional; Angaben zu einem bestimmten mandantenspezifischen Konfigurationsparameter, der für diesen Mandanten gesetzt ist.

Element Property vom Datentyp PropertyType

```
<xsd:complexType name="PropertyType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:string"></xsd:element>
    <xsd:element name="Value" type="xsd:string"></xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Name	<p>string:</p> <p>Name des Konfigurationsparameters.</p> <p>Bei diesem Namen ist die Groß-/Kleinschreibung relevant.</p> <p>Wird der Name eines Konfigurationsparameters angegeben, der prinzipiell nicht mandantenspezifisch eingestellt werden kann, wird die Operation abgewiesen.</p> <p>Eine Liste der Konfigurationsparameter, die mandantenspezifisch eingestellt werden können, finden Sie in der Tabelle „Mandantenspezifisch einstellbare Konfigurationsparameter“.</p>
Value	<p>string:</p> <p>Aktueller Wert des Konfigurationsparameters</p> <p>Genügt der Wert nicht den für diesen Konfigurationsparameter geltenden Bedingungen (z.B. keiner der erlaubten Werte, Überschreiten eines Maximalwerts), so wird die Operation abgewiesen.</p>

Datentyp CredentialType

```

<xsd:complexType name="CredentialType">
  <xsd:sequence>
    <xsd:element name="Type">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="Password">
          </xsd:enumeration>
          <xsd:enumeration value="Certificate">
          </xsd:enumeration>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:choice>
      <xsd:element name="Credits">
        <xsd:simpleType>
          <xsd:restriction base="xsd:base64Binary">
            <xsd:minLength value="8">
            </xsd:minLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element name="Password" type="tns:PasswordSimpleType">
      </xsd:element>
    </xsd:choice>
    <xsd:element name="Role" type="xsd:string"
      minOccurs="0" maxOccurs="1">
    </xsd:element>
    <xsd:element name="Mandant" type="xsd:string" minOccurs="0">
    </xsd:element>
    <xsd:element name="OrgID" type="xsd:string"
      minOccurs="0" maxOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

Type	<p>string: Typ des Credential. Mögliche Werte: Password Certificate Bei dieser Operation wird nur der Typ Password unterstützt. Der Typ Certificate ist nur für den Zugang zum Web-Service S4 vorgesehen und deshalb für diese Operation nicht relevant.</p>
Credits	<p>base64Binary: Möglichkeit der Autorisierung. Dieses Element ist nur für den Zugang zum Web-Service S4 vorgesehen und deshalb für diese Operation nicht relevant.</p>
Password	<p>string: Initiales Passwort für die Rollen MandantAdmin und Archivar (siehe Abschnitt „Zugangsprüfung“). Mindestlänge: 8 Zeichen</p>
Role	<p>string: Rolle für den Zugang zum Web-Service (siehe Abschnitt „Zugangsprüfung“). Der Datentyp CredentialType muss zweimal angegeben werden:</p> <ul style="list-style-type: none">• Für den Zugang zum Web-Service ArchivingService muss im Datentyp CredentialType die Rolle (role) Archivar angegeben werden.• Für den Zugang zum Web-Service MandantAdminService muss im Datentyp CredentialType die Rolle (role) MandantAdmin angegeben werden.
Mandant	<p>string: Mandant darf nicht angegeben werden.</p>
OrgID	<p>string: OrgID darf nicht angegeben werden.</p>

Datentyp ContactType

```

<xsd:complexType name="ContactType">
  <xsd:complexContent>
    <xsd:extension base="tns:BaseType">
      <xsd:sequence>
        <xsd:element name="FirstName" type="xsd:string"
          minOccurs="0">
        </xsd:element>
        <xsd:element name="Surname" type="tns:NonEmptyString"
          minOccurs="1" maxOccurs="1">
        </xsd:element>
        <xsd:element name="Street" type="xsd:string"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="City" type="xsd:string"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="Zip" type="xsd:string"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="Country" type="xsd:string"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="Tel" type="xsd:string" minOccurs="0">
        </xsd:element>
        <xsd:element name="Email" type="xsd:string" minOccurs="0">
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

Die Beschreibung des Datentyps BaseType finden Sie auf ["Operation createTSP"](#).

FirstName	string: optional; Vorname
Surname	string: Nachname
Street	string: optional; Straße
City	string: optional; Stadt
Zip	string: optional; Postleitzahl
Country	string: optional; Land
Tel	string: optional; Telefonnummer

Email	string: optional; Email-Adresse
-------	------------------------------------

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>createMandant</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>ArchiveAdmin</ns2:role>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>ArchiveAdmin operation createMandant sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
```

```

<S:Body>
  <CreateMandant
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Mandant>
      <Name>EDU</Name>
      <DisplayName>EDU</DisplayName>
      <Contact>
        <FirstName>Zaphod</FirstName>
        <Surname>Beeblebrox</Surname>
        <Street>Last Lane 1</Street>
        <City>Milliways</City>
        <Zip>12345</Zip>
        <Country>Universe</Country>
        <Tel>123456789</Tel>
        <Email>Zaphod.Beeblebrox@universe.gov</Email>
      </Contact>
      <Path>archive/EDU</Path>
      <TreeSize>10</TreeSize>
      <TreeAge>2</TreeAge>
      <TSP>TestTSP</TSP>
      <State>test</State>
    </Mandant>
    <Credentials>
      <Type>Password</Type>
      <Password>secrets1</Password>
      <Role>MandantAdmin</Role>
    </Credentials>
    <Credentials>
      <Type>Password</Type>
      <Password>secrets2</Password>
      <Role>Archivar</Role>
    </Credentials>
  </CreateMandant>
</S:Body>

```

Response

```

<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>createMandant</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>ArchiveAdmin</sdsh:role>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>ArchiveAdmin operation createMandant sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>

```

```
<soap:Body>
  <result xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"></result>
</soap:Body>
```

Setzen von mandantenspezifischen Einstellungen

```
<Properties>
  <Property>
    <Name>aoidKeepReservedPeriod</Name>
    <Value>900</Value>
  </Property>
  <Property>
    <Name>doHTMLProtocolsOnRetrieval</Name>
    <Value>true</Value>
  </Property>
  <Property>
    <Name>nodeNameInFaultMessage</Name>
    <Value>true</Value>
  </Property>
</Properties>
```

5.1.3.3 Operation `setCredentials`

`setCredentials` setzt, abhängig von der angegebenen Rolle, das Passwort für den Zugang zum Web-Service `ArchiveAdminService` selbst oder zum Web-Service `MandantAdminService`. Das heißt, es werden die Credentials für den Archivadministrator oder Mandantenadministrator gesetzt.

Bei der Änderung des Credentials für den Web-Service `MandantAdminService` ist im Request-Header das Passwort für den Zugang zum Web-Service `ArchiveAdminService` anzugeben und im Request-Body das neue Passwort für den Web-Service `MandantAdminService`.

i Der Web-Service `MandantAdminService` bietet ebenfalls die Funktion `setCredentials`, siehe "[Operation `setCredentials`](#)". In diesem Fall wird mit `setCredentials` die Zugangsberechtigung zum Web-Service `MandantAdminService` selbst bzw. zum Web-Service `ArchivingService` bzw. S4 für einen Mandanten gesetzt.

Gültigkeit der Passwörter

Das Passwort für den Archivadministrator wird erstmalig bei der Installation gesetzt. Das Passwort für den Mandantenadministrator wird mit `createMandant` gesetzt.

Ein Aufruf von `setCredentials` für die spezifizierte Rolle erlaubt den Zugang mit dem neuen Passwort, das „alte“ Passwort bleibt aber ebenfalls gültig. Durch einen weiteren `setCredentials`-Aufruf verliert das „alte“ Passwort seine Gültigkeit und das Passwort aus dem vorhergehenden `setCredentials`-Aufruf wird zum „alten“ Passwort. Der Zugang ist also mit den beiden zuletzt definierten Passwörtern möglich. Dies erlaubt es, die bei einem Passwortwechsel am Server und bei den Archivierungsclients notwendigen Aktionen zeitlich zu entkoppeln. Soll nur ein Passwort gelten, so muss zweimal nacheinander dasselbe Passwort angegeben werden.

Request-Body

Datentyp `CredentialType`

```

<xsd:complexType name="CredentialType">
  <xsd:sequence>
    <xsd:element name="Type">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="Password">
          </xsd:enumeration>
          <xsd:enumeration value="Certificate">
          </xsd:enumeration>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:choice>
      <xsd:element name="Credits">
        <xsd:simpleType>
          <xsd:restriction base="xsd:base64Binary">
            <xsd:minLength value="8">
            </xsd:minLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element name="Password">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="8">
            </xsd:minLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:choice>
    <xsd:element name="Role" type="xsd:string"
      minOccurs="1" maxOccurs="1">
    </xsd:element>
    <xsd:element name="Mandant" type="xsd:string" minOccurs="0">
    </xsd:element>
    <xsd:element name="OrgID" type="xsd:string"
      minOccurs="0" maxOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

Type	<p>string:</p> <p>Typ des Credential.</p> <p>Mögliche Werte: Password Certificate</p> <p>Für die Web-Services ArchiveAdminService, MandantAdminService und ArchivingService wird nur der Typ Password unterstützt.</p> <p>Der Typ Certificate ist nur für den Zugang zum Web-Service S4 vorgesehen und deshalb für diese Operation nicht relevant.</p>
Credits	<p>base64Binary:</p> <p>Möglichkeit der Autorisierung</p> <p>Dieses Element ist nur für den Zugang zum Web-Service S4 vorgesehen und deshalb für diese Operation nicht relevant.</p>

Password string:
Passwort für den Zugang zum Web-Service ArchiveAdminService bzw. für den Web-Service MandantAdminService.

Mindestlänge: 8 Zeichen

Role string:
Rolle für den Zugang zum Web-Service (siehe [Abschnitt „Zugangsprüfung“](#)).

Zugang zum Web-Service Rolle

ArchiveAdminService ArchiveAdmin

MandantAdminService MandantAdmin

Mandant string:
nur relevant, wenn bei Role MandantAdmin angegeben wurde;
Name des Mandanten, für den das Passwort für die Rolle MandantAdmin geändert werden soll.

OrgID string:
keine Angabe, ist für die Operation setCredentials nicht relevant.

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<soap:Header>
  <sdsh:soapHeaderData xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:sdsh="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
    <sdsh:operation>setCredentials</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>ArchiveAdmin</sdsh:role>
      </sdsh:principal>
      <sdsh:password>Pass1291376297287</sdsh:password>
    </sdsh:security>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <Credentials xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
    <Type>Password</Type>
    <Password>Pass1291376313631</Password>
    <Role>ArchiveAdmin</Role>
  </Credentials>
</soap:Body>
```

Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>ArchiveAdmin</sdsh:role>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:operation>setCredentials</sdsh:operation>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <result></result>
</soap:Body>
```

5.1.3.4 Operation getTSPs

getTSPs gibt eine Liste der TSPs aus, die im Archiv verfügbar sind.

Request-Body

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getTSPs</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>ArchiveAdmin</ns2:role>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>ArchiveAdmin operation getTSPs sample client</ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
```

Leeres Element getRequest

```
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
  </GetRequest>
</S:Body>
```

Response-Body

Datentyp GetTSPsResponseType

```
<xsd:complexType name="GetTSPsResponseType">
  <xsd:sequence>
    <xsd:element name="TSP" type="tns:TSPType"
      minOccurs="0" maxOccurs="unbounded">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

TSP Information über den TSP:
kein, ein oder mehrere Datentypen TSPTyp, siehe ["Operation createTSP"](#).

Beispiel

Response

```
<soap:Body>
  <GetTSPs xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <TSP xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>SecDocsTestTSP</Name>
      <ProviderName>Fujitsu Technology Solutions GmbH</ProviderName>
      <URL>http://172.17.32.94:8280/signserver/
        process?workerName=SECDOCSTESTTSP
      </URL>
      <HashAlgorithmName>SHA-256</HashAlgorithmName>
      <SigHashAlgorithmName>SHA-256</SigHashAlgorithmName>
      <SignatureAlgorithmName>rsaEncryption1</SignatureAlgorithmName>
      <KeyLength>4096</KeyLength>
      <CertificateName>SecDocs</CertificateName>
      <SignatureQuality>ADVANCED</SignatureQuality>
      <isActive>true</isActive>
    </TSP>
    <TSP xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>TSP-DFN</Name>
      <ProviderName>Deutsches Forschungs Netz (DFN)</ProviderName>
      <URL>http://zeitstempel.dfn.de/</URL>
      <HashAlgorithmName>SHA-256</HashAlgorithmName>
      <SigHashAlgorithmName>SHA-1</SigHashAlgorithmName>
      <SignatureAlgorithmName>rsaEncryption1</SignatureAlgorithmName>
      <KeyLength>2048</KeyLength>
      <CertificateName>DFN-Verein</CertificateName>
      <SignatureQuality>ADVANCED</SignatureQuality>
      <isActive>true</isActive>
    </TSP>
  </GetTSPs>
</soap:Body>
```

5.1.3.5 Operation updateTSP

updateTSP ändert bestehende Eigenschaften eines TSP.

Folgende Werte können Sie ändern:

- Eigenschaften, die vom TSP zum Signieren verwendet werden, wie die verwendeten Algorithmen und die Schlüssellänge und Signaturqualität
- Eigenschaften, die den Timestamp-Provider beschreiben, wie Provider-Name, URL, Zertifikatsname
- Die Policy, die beim Überprüfen der Signatur des vom TSP gelieferten Zeitstempels angewendet wird
- Ob beim Erzeugen des Evidence-Records die Zertifikate des Zeitstempelanbieters eingebettet werden sollen

Folgende Werte können Sie nicht ändern:

- der Algorithmus, der zum Bilden des Hash-Wertes des SDOs verwendet wird
- die Eigenschaft `isActive`

i Alle angegebenen Werte werden geändert. Nicht angegebene Werte bleiben unverändert. Einen Wert, den Sie nicht ändern wollen, dürfen Sie nicht angeben.

Bei updateTSP wird, um die Korrektheit der Angaben zu prüfen, vom angegebenen Time-stamp-Provider ein Zeitstempel geholt und ein Evidence Record erzeugt; es sei denn, der Administrator des Archivs hat den Konfigurationsparameter `checkTSP` (in der Konfigurationsdatei `secdocs.properties`) auf `false` gesetzt.

Wenn die daran anschließende Verifikation des Evidence Records (und damit die Operation updateTSP insgesamt) fehlschlägt, so wird in der Fault-Message im Element `additionalData` das Verifikationsprotokoll des CryptoModule mit allen Detail-Angaben für die Signaturprüfung mit ausgegeben.

Request-Body

Element TSP vom Datentyp TSPTyp

```
<xsd:complexType name="TSPTyp">
  <xsd:complexContent>
    <xsd:extension base="tns:BaseType">
      <xsd:sequence>
        <xsd:element name="Name" minOccurs="1" maxOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:NCName">
              <xsd:maxLength value="50"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="ProviderName"
          type="tns:NonEmptyString" minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="URL" type="tns:NonEmptyString"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="SigHashAlgorithmName" type="xsd:string"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
        <xsd:element name="SignatureAlgorithmName" type="xsd:string">

```

```

        minOccurs="0" maxOccurs="1">
</xsd:element>
<xsd:element name="KeyLength" minOccurs="0" maxOccurs="1">
  <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
      <xsd:maxInclusive value="2147483647"/>
      <xsd:minInclusive value="1"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="CertificateName" type="xsd:string"
  minOccurs="0" maxOccurs="1">
</xsd:element>
<xsd:element name="SignatureQuality"
  minOccurs="0" maxOccurs="1">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ADVANCED"/>
      <xsd:enumeration value="QUALIFIED"/>
      <xsd:enumeration value="ACCREDITED"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="isActive" type="xsd:boolean"
  maxOccurs="1" minOccurs="0">
</xsd:element>
<xsd:element name="Policy" maxOccurs="1" minOccurs="0">
  <xsd:simpleType>
    <xsd:restriction base="xsd:base64Binary">
      <xsd:minLength value="1"/></xsd:minLength>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<!-- if not present, doEnrich will be set to false -->
<xsd:element name="doEnrich" type="xsd:boolean"
  maxOccurs="1" minOccurs="0">
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

```

Die Beschreibung des Datentyps BaseType finden Sie auf ["Operation createTSP"](#).

Name	NCName: Name des zu verändernden TSPs in SecDocs.
ProviderName	NonEmptyString: Name des TSPs (z.B. Firmenname, Name des Instituts oder der Behörde). ProviderName kann beliebig gewählt werden.

URL	<p>string: URL des Zeitstempel-Dienstes.</p> <p>Die URL muss hier immer gemeinsam mit den evtl. erforderlichen Erweiterungen angegeben werden. Weitere Information zu den Erweiterungen siehe Abschnitt „Operation createTSP“.</p>
SigHashAlgorithmName	<p>string: Der Hash-Algorithmus, der vom TSP verwendet wird, um den Zeitstempel zu erstellen.</p> <div> <p>i Der angegebene Hash-Algorithmus muss verfügbar sein (siehe Abschnitt „Operation getHashAlgorithms“).</p> </div>
SignatureAlgorithmName	<p>string: Genutzter Signatur-Algorithmus, wie vom TSP vorgegeben. Muss mit dem verwendeten Algorithmus in der Zertifikatssignatur übereinstimmen.</p> <div> <p>i Der angegebene Signatur-Algorithmus muss verfügbar sein (siehe Abschnitt „Operation getSignatureAlgorithms“).</p> </div>
KeyLength	<p>integer: Länge des Signatur-Algorithmus, wie vom TSP vorgegeben. Muss mit der Schlüssellänge in der Zertifikatssignatur übereinstimmen.</p> <p>Mögliche Werte: 1 - 2147483647</p>
CertificateName	<p>string: Name des Erstellerzertifikats, wie vom TSP vorgegeben. Aus dem Zertifikat, das zum Signieren des Zeitstempels vom TSP verwendet wird (beim TSP zu erfragen), ist ein Teil-String des Feldes „Antragssteller“ (Subject) anzugeben. Die Prüfung erfolgt nicht case-sensitiv.</p> <div> <p>i Ein TSP verwendet meist mehrere Signiereinheiten, deshalb ist der Wert des Feldes „Antragssteller“ über alle möglichen Zertifikate eines TSPs nicht konstant. Wählen Sie deshalb nur einen Teilstring, der über alle verwendeten Zertifikate konstant ist. Der Wert des Feldes „Antragssteller“ ist strukturiert in: CN=<string>, OU=<string>, O=<string>, C=<string>. Die Reihenfolge dieser Teilfelder kann variieren. Es wird deshalb empfohlen, nur ein (konstantes) Teilfeld als CertificateName zu verwenden.</p> </div>

SignatureQuality

string:

Qualität der Signatur, die der Zeitstempelanbieter (TSP) verwendet, um seine Zeitstempel zu signieren. Es können verschieden hohe Anforderungen an das Zertifikat gestellt werden, das der TSP für die Signatur der Zeitstempel verwendet. Diese Einstufung ist abhängig vom ausgewählten TSP.

Mögliche Werte:

"ADVANCED" Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur.

"QUALIFIED" Der TSP signiert die Zeitstempel mindestens mit einer fortgeschrittenen elektronischen Signatur, die auf einem qualifizierten Zertifikat beruht.

Policy

base64Binary:

Datei zur Steuerung der Bewertung der Signaturprüfung der vom TSP gelieferten Zeitstempel

! Das Erstellen einer Policy Datei ist eine Service-Leistung der Fujitsu GMBH und sollte mit ihrem technischen Betreuer abgestimmt werden.

doEnrich

boolean:

optional; Gibt an ob Revocation-Info und Zertifikate des Zeitstempelanbieters in den Evidence-Record eingebettet werden.

Mögliche Werte:

"true" Die Informationen werden eingebettet.

"false" Die Informationen werden erst beim Renew eingebettet.

Standardwert: true

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Die Schlüssellänge des TSPs `MYTSP` soll geändert werden.

Request

```
<S:Header>
  <ns3:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
    <ns3:operation>updateTSP</ns3:operation>
    <ns3:security>
      <ns3:principal>
        <ns3:role>ArchiveAdmin</ns3:role>
      </ns3:principal>
      <ns3:password>*****</ns3:password>
    </ns3:security>
  </ns3:soapHeaderData>
</S:Header>
<S:Body>
  <TSP xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
    <Name>MYTSP</Name>
    <KeyLength>2048</KeyLength>
  </TSP>
</S:Body>
```

5.1.3.6 Operation getMandants

getMandants gibt eine Liste aller Mandanten im Archiv aus.

Request-Body

Leeres Element getRequest

```
<GetRequest></GetRequest>
```

Response-Body

Datentyp GetMandantsResponseType

```
<xsd:complexType name="GetMandantsResponseType">
  <xsd:sequence>
    <xsd:element name="Mandant" type="tns:MandantType"
      minOccurs="0" maxOccurs="unbounded">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Mandant Information über den Mandanten:
kein, ein oder mehrere Datentypen `MandantType`, siehe "[Operation createMandant](#)".

Das im Datentyp `MandantType` enthaltene Element `State` kann folgende Werte annehmen:

<code>State</code>	<code>string</code> : Status des Mandanten
	Mögliche Werte:
<code>test</code>	Der Mandant ist ein Testmandant.
<code>productive</code>	Der Mandant ist produktiv.
<code>convertingToProductive</code>	Der Mandant ist ein Testmandant und wird gerade in einen produktiven Mandanten überführt (siehe Abschnitt „Operation convertTestMandant“).
<code>clearing</code>	Der Mandant ist ein Testmandant und es werden gerade alle AOIDs des Testmandanten gelöscht (siehe Abschnitt „Operation clearAOIDs“).
<code>deleting</code>	Der Mandant ist ein Testmandant und er wird gerade gelöscht (siehe Abschnitt „Operation deleteTestMandant“).

Das im Datentyp `MandantType` enthaltene Element `Type` hat folgenden Inhalt:

<code>Type</code>	<code>string</code> : Format der Dokumente, die der Mandant archivieren kann.
	Mögliche Werte:
<code>SDO</code>	Der Mandant kann Datenobjekte in Form eines XML-Containers als Submission Data Objects (SDOs) mit dem Web-Service <code>Archiving-Service</code> archivieren.
<code>XAIP</code>	Der Mandant kann Datenobjekte im Format XAIP mit dem Web-Service <code>s4</code> archivieren.

Für einen Mandanten, der mit der Operation `createMandant` erzeugt wurde, wird immer `SDO` ausgegeben.

Beispiel

Response

```

<soap:Body>
  <GetMandants
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Mandant xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>FujitsuXAIP</Name>
      <Contact>
        <FirstName>Zaphod</FirstName>
        <Surname>Beeblebrox</Surname>
        <Street>Last Lane 1</Street>
        <City>Milliways</City>
        <Zip>12345</Zip>
        <Country>Universe</Country>
        <Tel>123456789</Tel>
        <Email>Zaphod.Beeblebrox@universe.gov</Email>
      </Contact>
      <Path>archive/FujitsuXAIP</Path>
      <TreeSize>10</TreeSize>
      <TreeAge>2</TreeAge>
      <TSP>TestTSP</TSP>
      <State>test</State>
      <Type>XAIP</Type>
      <Properties>
        <Property
          xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
          <Name>$SignatureEmbedded</Name>
          <Value>AUTO</Value>
        </Property>
        <Property
          xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
          <Name>$SignatureQualityLevel</Name>
          <Value>ADVANCED</Value>
        </Property>
        <Property
          xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
          <Name>$SDOPath</Name>
          <Value>*$Year*/*$Month*/*$Day*</Value>
        </Property>
        <Property
          xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
          <Name>$SignatureVerification</Name>
          <Value>SUCCESS</Value>
        </Property>
        <Property
          xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
          <Name>supportsXAIP</Name>
          <Value>true</Value>
        </Property>
      </Properties>
    </Mandant>
  </GetMandants>
</soap:Body>

```

5.1.3.7 Operation getHashAlgorithms

getHashAlgorithms gibt eine Liste der verfügbaren Hash-Algorithmen aus.

Request-Body

Leeres Element getRequest

```
<GetRequest
  xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
  xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
</GetRequest>
```

Response-Body

Datentyp GetHashAlgorithmsResponseType

```
<xsd:complexType name="GetHashAlgorithmsResponseType">
  <xsd:sequence>
    <xsd:element name="Algorithm" type="tns:AlgorithmType"
      minOccurs="0" maxOccurs="unbounded">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Algorithm Information über den Hash-Algorithmus:

kein, ein oder mehrere Datentypen AlgorithmType, siehe "[Datentyp AlgorithmType](#)".

Datentyp AlgorithmType

```

<xsd:complexType name="AlgorithmType">
  <xsd:complexContent>
    <xsd:extension base="tns:BaseType">
      <xsd:sequence>
        <xsd:element name="Name" type="xsd:NCName">
        </xsd:element>
        <xsd:element name="Type">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:pattern value="Hash|Signature" />
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="OID">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:pattern value="[0-9]([0-9]+)*">
            </xsd:pattern>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="URI" type="xsd:string" minOccurs="0"
          maxOccurs="1">
        </xsd:element>
        <xsd:element name="ExpirationDate" type="xsd:dateTime"
          minOccurs="0" maxOccurs="1">
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

Die Beschreibung des Datentyps BaseType finden Sie auf ["Operation createTSP"](#).

Name	NCName: Name des Algorithmus
Type	string: Typ des Algorithmus
OID	string: Object Identifier
URI	string: Uniform Resource Identifier
ExpirationDate	dateTime: Expiration Date des Algorithmus

Beispiel

Response


```
<soap:Body>
  <GetHashAlgorithms
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>SHA-1</Name>
      <Type>Hash</Type>
      <OID>1.3.14.3.2.26</OID>
      <URI>http://www.w3.org/2000/09/xmldsig#sha1</URI>
      <ExpirationDate>2010-12-31T23:59:59Z</ExpirationDate>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>SHA-384</Name>
      <Type>Hash</Type>
      <OID>2.16.840.1.101.3.4.2.2</OID>
      <URI>http://www.w3.org/2001/04/xmlenc#sha384</URI>
      <ExpirationDate>2022-12-31T09:30:47Z</ExpirationDate>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>SHA-224</Name>
      <Type>Hash</Type>
      <OID>2.16.840.1.101.3.4.2.4</OID>
      <URI>http://www.w3.org/2001/04/xmlenc#sha224</URI>
      <ExpirationDate>2015-12-31T09:30:47Z</ExpirationDate>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>SHA-256</Name>
      <Type>Hash</Type>
      <OID>2.16.840.1.101.3.4.2.1</OID>
      <URI>http://www.w3.org/2001/04/xmlenc#sha256</URI>
      <ExpirationDate>2022-12-31T09:30:47Z</ExpirationDate>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>SHA-512</Name>
      <Type>Hash</Type>
      <OID>2.16.840.1.101.3.4.2.3</OID>
      <URI>http://www.w3.org/2001/04/xmlenc#sha512</URI>
      <ExpirationDate>2022-12-31T09:30:47Z</ExpirationDate>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>RIPEMD-160</Name>
      <Type>Hash</Type>
      <OID>1.3.36.3.2.1</OID>
      <URI>http://www.w3.org/2001/04/xmlenc#ripemd160</URI>
      <ExpirationDate>2015-12-31T09:30:47Z</ExpirationDate>
    </Algorithm>
  </GetHashAlgorithms>
</soap:Body>
```

5.1.3.8 Operation `getSignatureAlgorithms`

`getSignatureAlgorithms` gibt eine Liste der verfügbaren Signatur-Algorithmen aus.

Request-Body

Leeres Element `getRequest`

```
<GetRequest
  xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
  xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
</GetRequest>
```

Response-Body

Datentyp `GetSignatureAlgorithmsResponseType`

```
<xsd:complexType name="GetSignatureAlgorithmsResponseType">
  <xsd:sequence>
    <xsd:element name="Algorithm" type="tns:AlgorithmType"
      minOccurs="0" maxOccurs="unbounded">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Algorithm Information über den Signatur-Algorithmus:
kein, ein oder mehrere Datentypen `AlgorithmType`, siehe ["Operation `getHashAlgorithms`"](#).

Beispiel

Response

```
<soap:Body>
  <GetSignatureAlgorithms
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Algorithm xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>ecdsaPlainSHA384</Name>
      <Type>Signature</Type>
      <OID>0.4.0.127.0.7.1.1.4.1.4</OID>
    </Algorithm>
    <Algorithm xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>ECDSASignature</Name>
      <Type>Signature</Type>
      <OID>1.2.840.10045.4</OID>
    </Algorithm>
    <Algorithm xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>rsaEncryption1</Name>
      <Type>Signature</Type>
      <OID>1.2.840.113549.1.1.1</OID>
    </Algorithm>
    <Algorithm xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>iqS_ISO9796-2rndWithripemd160</Name>
      <Type>Signature</Type>
      <OID>1.3.36.3.4.3.2.2</OID>
    </Algorithm>
    ...
  </GetSignatureAlgorithms>
</soap:Body>
```

5.1.3.9 Operation getVersion

getVersion liefert die aktuelle SecDocs-Version.

Request-Body

Leeres Element getRequest

```
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"></GetRequest>
</S:Body>
```

Response-Body

Datentyp VersionType

```
<xsd:complexType name="VersionType">
  <xsd:sequence>
    <xsd:element name="Component" type="tns:ComponentType"
      minOccurs="1" maxOccurs="unbounded">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Component Information über SecDocs und dessen Subkomponenten:
ein oder mehrere Datentypen ComponentType, siehe unten.

Das erste Element enthält Informationen über SecDocs als Ganzes. Alle weiteren Elemente
enthalten Informationen über Subkomponenten.

Datentyp ComponentType

```
<xsd:complexType name="ComponentType">
  <xsd:sequence>
    <xsd:element name="VersionString" type="xsd:string"></xsd:element>
    <xsd:element name="Name" type="xsd:string"></xsd:element>
    <xsd:element name="Major" type="xsd:integer"></xsd:element>
    <xsd:element name="Minor" type="xsd:integer"></xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

VersionString	string: vollständige Version in der Form: Name VMajorNumber.MinorNumber[A-Z]BuildNumber <i>Beispiel:</i> SecDocs V3.1A01
Name	string: Name der Komponente
Major	integer: Major-Nummer der Version
Minor	integer: Minor-Nummer der Version

Beispiel

Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getVersion</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>ArchiveAdmin</sdsh:role>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>ArchiveAdmin operation getVersion sample client</sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <GetVersion
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Component>
      <VersionString>SecDocs V3.1A01</VersionString>
      <Name>SecDocs</Name>
      <Major>3</Major>
      <Minor>1</Minor>
    </Component>
  </GetVersion>
</soap:Body>
```

5.1.3.10 Operation `getAccountingData`

`getAccountingData` stellt dem Archivadministrator Informationen für die Abrechnung gegenüber seinen Mandanten zur Verfügung.

Folgende Informationen werden ausgegeben:

- die Anzahl der insgesamt im Archiv existierenden versiegelten Dokumente zum Zeitpunkt der Abfrage (Archivgröße) für einen bestimmten Mandanten oder für alle Mandanten zusammen
- die Anzahl der bisher durchgeführten Versiegelungsvorgänge für einen bestimmten Mandanten oder für alle Mandanten zusammen
- die Anzahl der bisher bei einem bestimmten Zeitstempelanbieter (TSP) verbrauchten Zeitstempel für das gesamte Archiv, d.h. für alle Mandanten zusammen
- die Summe aller Versiegelungsvorgänge, die für sämtliche Testmandanten des Archivs bis zum jeweiligen Zeitpunkt der Löschung des Testmandanten bzw. der Löschung aller AOIDs des Testmandanten durchgeführt wurden



Eine ausführliche Beschreibung des Accounting finden Sie im [Abschnitt „Accounting“](#).

Request-Body

Datentyp `GetAccountingDataArchivAdmRequestType`

```
<xsd:complexType name="GetAccountingDataArchivAdmRequestType">
  <xsd:sequence>
    <xsd:element name="MandantName" type="xsd:NCName"
      minOccurs="0" maxOccurs="1" ></xsd:element>
    <xsd:element name="TSPName" type="xsd:NCName"
      minOccurs="0" maxOccurs="1" ></xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

MandantName **NCName:**

gibt den Mandanten an, für den die Abrechnungsinformationen ausgegeben werden sollen.
Fehlt das Element `MandantName`, so wird für die einzelnen Accounting-Daten jeweils die Summe über alle Mandanten ausgegeben.

TSPName

NCName:

gibt den Zeitstempelanbieter an, für den die Anzahl der verbrauchten Zeitstempel ausgegeben werden soll.
Fehlt das Element `TSPName`, so wird die Anzahl der verbrauchten Zeitstempel nicht mit ausgegeben.

Response-Body

Element `getAccountingDataArchivAdmRequest` *vom Datentyp*

`GetAccountingDataArchivAdmResponseType`

```

<xsd:complexType name="GetAccountingDataArchivAdmResponseType">
  <xsd:sequence>
    <xsd:element name="NumberOfSealedDocuments"
      type="xsd:long"></xsd:element>
    <xsd:element name="NumberOfSealingActions"
      type="xsd:long"></xsd:element>
    <xsd:element name="NumberOfUsedTimeStamps" type="xsd:long"
      minOccurs="0" maxOccurs="1"></xsd:element>
    <xsd:element name="NumberOfSealingActionsDuringTestPhase"
      type="xsd:long"> </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

NumberOfSealedDocuments

long:

Enthält die Anzahl der versiegelten Dokumente im Archiv für den im Request-Body angegebenen Bereich.

NumberOfSealingActions

long:

Enthält die Anzahl der Versiegelungsvorgänge für den im Request-Body angegebenen Bereich.

i Wird die Information für alle Mandanten angefordert, d.h. ist beim Operanden `MandantName` kein Wert angegeben, so ist in der ausgegebenen Anzahl der unter `NumberOfSealingActionsDuringTestPhase` ausgegebene Wert mit enthalten.

NumberOfUsedTimeStamps

long:

Enthält die Anzahl der verbrauchten Zeitstempel des gesamten Archivs bei dem im Operanden `TSPName` angegebenen Zeitstempelanbieter. Dieser Wert wird nicht ausgegeben, wenn beim Operanden `TSPName` kein Zeitstempelanbieter genannt wurde.

i Für den angegebenen TSP wird immer der Wert für das gesamte Archiv, d.h. der Summenwert über alle Mandanten, ausgegeben. Beachten Sie: Die Fehlversuche bei der Versiegelung mit einem Zeitstempel werden nicht mitgezählt! Bei dem von SecDocs gelieferten Wert handelt es sich also um die Anzahl Zeitstempel, die **mindestens** verbraucht wurde. Der Wert kann aber tatsächlich höher liegen.

NumberOfSealingActionsDuringTestPhase long:
enthält die Summe aller Versiegelungsvorgänge, die für sämtliche Testmandanten des Archivs bis zum jeweiligen Zeitpunkt des Löschens des Testmandanten bzw. der Löschung aller AOIDS des Testmandanten aufgelaufen sind.

i Es wird immer der Gesamtwert für das ganze Archiv ausgegeben, unabhängig von der Angabe beim Operanden MandantName.

Beispiel

SOAP-Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getAccountingData</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>ArchiveAdmin</ns2:role>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>ArchiveAdmin operation getAccountingData sample client</ns2:
auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <getAccountingDataArchivAdmRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <MandantName>Fujitsu</MandantName>
    <TSPName>TestTSP</TSPName>
  </getAccountingDataArchivAdmRequest>
</S:Body>
```


SOAP-Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getAccountingData</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>ArchiveAdmin</sdsh:role>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>ArchiveAdmin operation getAccountingData sample client</sdsh:
auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <getAccountingDataArchivAdmResponse
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <NumberOfSealedDocuments>0</NumberOfSealedDocuments>
    <NumberOfSealingActions>0</NumberOfSealingActions>
    <NumberOfUsedTimeStamps>5</NumberOfUsedTimeStamps>
    <NumberOfSealingActionsDuringTestPhase>3<
/NumberOfSealingActionsDuringTestPhase>
  </getAccountingDataArchivAdmResponse>
</soap:Body>
```

5.1.3.11 Operation `getStatisticalData`

`getStatisticalData` gibt für alle Mandanten und für alle Operationen der Web-Services `ArchivingService` und `s4` die Statistikdaten aus, die auf einem SecDocs-Server seit dem Start von SecDocs bzw. seit dem zuletzt erfolgten Rücksetzen der Messwerte ermittelt wurden.

SecDocs sammelt diese Statistikdaten für einen Mandanten, wenn der Konfigurationsparameter `collectOperationStatistics` entweder mandantenspezifisch (siehe "[Mandantenspezifische Konfigurationsparameter](#)") oder generell in der Konfigurationsdatei `secdocs.properties` (siehe "[Konfigurationsdatei secdocs.properties](#)") auf `true` gesetzt ist.

Wenn keine dieser Voraussetzungen zutrifft wird das Element `mandantData` für diesen Mandanten nicht ausgegeben.

Falls vom Anwender angefordert, werden die Messwerte nach dem Auslesen der Statistikdaten auf Null zurückgesetzt.

Im Multi-Node-Betrieb wirkt sich die Operation `getStatisticalData` immer nur auf dem Knoten aus, der adressiert ist.



Beachten Sie, dass die Client-Anwendung den gewünschten Knoten direkt, d.h. unter Umgehung des Load-Balancers, adressieren muss.



Ausführliche Informationen über die Erfassung von Statistikdaten finden Sie im [Abschnitt „Erheben von Statistikdaten“](#).

Request-Body

Element `getStatisticalDataRequest` vom Datentyp `GetStatisticalDataRequestType`

```
<xsd:complexType name="GetStatisticalDataRequestType">
  <xsd:sequence>
    <xsd:element name="resetData" type="xsd:boolean"
      minOccurs="0" maxOccurs="1" default="false">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

`resetData` `boolean`:

optional; gibt an, ob SecDocs die Statistikdaten zurücksetzt.

Mögliche Werte:

"true" SecDocs setzt die Statistikdaten zurück.

"false" SecDocs setzt die Statistikdaten nicht zurück.

Standardwert: `false`

Response-Body

Element getStatisticalDataResponse *vom Datentyp* GetStatisticalDataResponseType

```
<xsd:complexType name="GetStatisticalDataResponseType">
  <xsd:sequence>
    <xsd:element name="archiveData" type="tns:ArchiveDataType"
                  minOccurs="1" maxOccurs="1">

    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

archiveData Statistikdaten für ein SecDocs-Archiv
Datentyp ArchiveDataType, siehe "[Datentyp ArchiveDataType](#)"

Element archiveData *vom Datentyp* ArchiveDataType

```
<xsd:complexType name="ArchiveDataType">
  <xsd:sequence>
    <xsd:element name="lastFlushTime"        type="xsd:long"
                  maxOccurs="1" minOccurs="1">

    </xsd:element>
    <xsd:element name="currentFlushTime" type="xsd:long"
                  maxOccurs="1" minOccurs="1">

    </xsd:element>
    <xsd:element name="mandantData"        type="tns:MandantDataType"
                  maxOccurs="unbounded" minOccurs="0">

    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

lastFlushTime long:
Zeitpunkt der zuletzt erfolgten Rücksetzung der Messwerte für den gewünschten Knoten in Millisekunden seit 1.1.1970

-1 Die Messwerte wurden für den gewünschten Knoten seit dem Start von SecDocs noch nicht mit der Operation getStatisticalData zurückgesetzt.

currentFlushTime long:
Aktueller Zeitpunkt in Millisekunden seit 1.1.1970

mandantData Statistikdaten für einen Mandanten
Datentyp MandantDataType, siehe "[Datentyp MandantDataType](#)"

Element mandantData *vom Datentyp* MandantDataType

```

<xsd:complexType name="MandantDataType">
  <xsd:sequence>
    <xsd:element name="mandantName" type="xsd:NCName"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="unknownOpCount" type="xsd:long"
      maxOccurs="1" minOccurs="0">
    </xsd:element>
    <xsd:element name="operationData" type="tns:OperationDataType"
      maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

mandantName	NCName: Name des Mandanten
unknownOpCount	long: optional; Anzahl der Aufrufe unbekannter Operationen, d.h. Operationen, die im Funktionsumfang der im Request angegebenen Rolle nicht enthalten sind. unknownOpCount wird nur ausgegeben, wenn es einen Wert > 0 hat.
operationData	Statistikdaten für eine Operation Datentyp OperationDataType, siehe "Datentyp OperationDataType"

Element operationData vom Datentyp OperationDataType

```

<xsd:complexType name="OperationDataType">
  <xsd:sequence>
    <xsd:element name="operationName" type="tns:NonEmptyString"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="countData" type="tns:CountDataType"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="timeData" type="tns:TimeDataType"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="sdoSizeData" type="tns:SdoSizeDataType"
      maxOccurs="1" minOccurs="0">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

operationName	NonEmptyString: Name der Operation
countData	Messwerte bzgl. der Anzahl der Aufrufe der Operation Datentyp CountDataType, siehe "Datentyp CountDataType"

timeData	Datentyp TimeDataType, siehe unten Messwerte bzgl. der Dauer der Operation
sdoSizeData	Datentyp SdoSizeDataType, siehe unten optional; Messwerte bzgl. der SDO-Größe bei der Operation submitSDO bzw. replaceSDO oder Messwerte bzgl. der Größe der XAIP-Datenobjekte bei der Operation ArchiveSubmission

Element countData vom Datentyp CountDataType

```
<xsd:complexType name="CountDataType">
  <xsd:sequence>
    <xsd:element name="totCount" type="xsd:long"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="succCount" type="xsd:long"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="failCount" type="xsd:long"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

totCount	long: Gesamtanzahl der Aufrufe einer Operation
succCount	long: Anzahl erfolgreicher Aufrufe einer Operation
failCount	long: Anzahl fehlgeschlagener Aufrufe einer Operation

Element timeData vom Datentyp TimeDataType

```
<xsd:complexType name="TimeDataType">
  <xsd:sequence>
    <xsd:element name="minDuration" type="xsd:double"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="maxDuration" type="xsd:double"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="avgDuration" type="xsd:double"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

minDuration **double:**
Kürzeste Bearbeitungsdauer der Operation in Millisekunden

maxDuration **double:**
Längste Bearbeitungsdauer der Operation in Millisekunden

avgDuration **double:**
Durchschnittliche Dauer der Operation in Millisekunden

i Bei den ausgegebenen Werten wird das Dezimaltrennzeichen unabhängig von der Spracheinstellung immer als Punkt dargestellt. Näheres zur Spracheinstellung finden Sie im [Abschnitt „Internationalisierung“](#).

Element sdoSizeData vom Datentyp SdoSizeDataType

```
<xsd:complexType name="SdoSizeDataType">
  <xsd:sequence>
    <xsd:element name="minSdoSize" type="xsd:double"
                  maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="maxSdoSize" type="xsd:double"
                  maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="avgSdoSize" type="xsd:double"
                  maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="totSdoSize" type="xsd:double"
                  maxOccurs="1" minOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

minSdoSize **double:**
Größe des kleinsten SDOs bzw. XAIP-Datenobjekts in KB

maxSdoSize **double:**
Größe des größten SDOs bzw. XAIP-Datenobjekts in KB

avgSdoSize **double:**
Durchschnittliche Größe eines SDOs bzw. XAIP-Datenobjekts in KB

totSdoSize **double:**
Summe der SDO-Größen aller erfolgreichen Operationen `submitSDO` bzw. `replaceSDO` in KB
oder Summe der Größen aller XAIP-Datenobjekte bei der Operation `ArchiveSubmission`

i Bei den ausgegebenen Werten wird das Dezimaltrennzeichen unabhängig von der Spracheinstellung immer als Punkt dargestellt. Näheres zur Spracheinstellung finden Sie im [Abschnitt „Internationalisierung“](#).

Beispiele

Ausgabe mit 3 Mandanten mit folgenden Voraussetzungen:

- Für Mandant1 gilt collectOperationStatistics=true, jedoch haben sich noch keine Statistikdaten angesammelt.
- Für Mandant2 gilt collectOperationStatistics=false.
- Für Mandant3 gilt collectOperationStatistics=true und es existieren bereits Statistikdaten.

Request-Body

```
<S:Body>
  <getStatisticalDataRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <resetData>false</resetData>
  </getStatisticalDataRequest>
</S:Body>
```

Response-Body

```
<soap:Body>
  <tns:getStatisticalDataResponse
    xmlns:tns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <tns:archiveData>
      <tns:lastFlushTime>-1</tns:lastFlushTime>
      <tns:currentFlushTime>1381318345813</tns:currentFlushTime>
      <tns:mandantData>
        <tns:mandantName>Mandant1</tns:mandantName>
      </tns:mandantData>
      <tns:mandantData>
        <tns:mandantName>Mandant3</tns:mandantName>
        <tns:unknownOpCount>1</tns:unknownOpCount>
        <tns:operationData>
          <tns:operationName>statusSDO</tns:operationName>
          <tns:countData>
            <tns:totCount>4</tns:totCount>
            <tns:succCount>4</tns:succCount>
            <tns:failCount>0</tns:failCount>
          </tns:countData>
          <tns:timeData>
            <tns:minDuration>28</tns:minDuration>
            <tns:maxDuration>112</tns:maxDuration>
            <tns:avgDuration>56.750</tns:avgDuration>
          </tns:timeData>
        </tns:operationData>
      </tns:mandantData>
    </tns:archiveData>
  </tns:getStatisticalDataResponse>
</soap:Body>
```

5.1.3.12 Operation deleteTestMandant

Die Operation `deleteTestMandant` dient zum Löschen eines Testmandanten. Es werden zunächst alle AOIDs des Mandanten und dann der Mandant, seine Organisationsstruktur und all seine Daten vollständig gelöscht.

Müssen während des Löschens eines Testmandanten viele AOIDs entfernt werden, kann diese Operation länger dauern. Deshalb läuft sie asynchron. Sollte der Auftrag z.B. durch Herunterfahren von SecDocs unterbrochen werden, so läuft er beim Starten von SecDocs wieder an.

Den Status der Löschoperation kann man über `getMandants` (Web-Service `ArchiveAdminService`) bzw. `getMandantProperties` (Web-Service `MandantAdminService`) ermitteln. Solange das Element `State` in der Antwort den Wert `deleting` hat, ist die Operation noch nicht abgeschlossen.

Ist die Löschoperation erfolgreich abgeschlossen, liefert `getMandantProperties` - wie bei einem nicht vorhandenen Mandanten - die Fehlermeldung, dass die Zugangsdaten ungültig sind.

Request-Body

Element `DeleteTestMandant` vom Datentyp `DeleteTestMandantType`

```
<xsd:complexType name="DeleteTestMandantType">
  <xsd:sequence>
    <xsd:element name="MandantName" type="xsd:NCName"
      maxOccurs="1" minOccurs="0"></xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

MandantName NCName:
 Name des Testmandanten, der gelöscht werden soll.

Response-Body

Leeres Element `result`

```
<result></result>
```

Beispiel

Request


```

<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>deleteTestMandant</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>ArchiveAdmin</ns2:role>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>ArchiveAdmin operation deleteTestMandant sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <DeleteTestMandant
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <MandantName>Fujitsu</MandantName>
  </DeleteTestMandant>
</S:Body>

```

Response

```

<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>deleteTestMandant</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>ArchiveAdmin</sdsh:role>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>ArchiveAdmin operation deleteTestMandant sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <result xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"></result>
</soap:Body>

```

5.1.3.13 Operation performAction

performAction löst Aktionen auf dem Archiv aus. Angeboten werden die Aktionen stopMonitor und startMonitor. stopMonitor hält Überwachungsprozesse an, startMonitor aktiviert sie (wieder).

Folgende Überwachungsprozesse (Monitore) können mit der Operation stopMonitor angehalten (die durch die Überwachungsprozesse bereits gestarteten Aufträge laufen davon unberührt bis zum Ende weiter) und mit der Operation startMonitor aktiviert werden.

- **SDOCounting:**
Überwachung der Anzahl der für die Versiegelung anstehenden SDOs (TreeSize). Wird der Wert TreeSize erreicht, so werden die anstehenden SDOs versiegelt (also Aufbau eines Hash-Baums, Einholen eines Zeitstempels, Erzeugen eines Evidence Records je SDO).
TreeSize kann mit dem Web-Service MandantAdminService mit createMandant konfiguriert und mit updateMandant geändert werden.
- **TSPTimer:**
Überwachung der maximalen Wartezeit eines SDOs auf Versiegelung (TreeAge). Wird TreeAge erreicht, so werden die anstehenden SDOs versiegelt (also Aufbau eines Hash-Baums, Einholen eines Zeitstempels, Erzeugen eines Evidence Records je SDO).
TreeAge kann mit dem Web-Service MandantAdminService mit createMandant konfiguriert und mit updateMandant geändert werden.
- **AOIDCleanup:**
In periodischen Abständen wird geprüft, ob AOIDs reserviert wurden (Operation getAOID des Web-Service ArchivingService), deren Reservierung inzwischen abgelaufen ist. Die maximale Reservierungsdauer wird festgelegt durch den Konfigurationsparameter aoidKeepReservedPeriod in der der Datei secdocs.properties.
- **AOIDWithRefCleanup:**
In periodischen Abständen wird geprüft, ob AOIDs mit externen Referenz-IDs reserviert wurden (Operation getAOIDWithRef des Web-Service ArchivingService), deren Reservierung inzwischen abgelaufen ist. Die maximale Reservierungsdauer wird festgelegt durch den Konfigurationsparameter aoidWithRefKeepReservedPeriod (Einstellung mandantenspezifisch oder in der Datei secdocs.properties).
- **ExternalFileCleanup:**
In periodischen Abständen wird geprüft, ob sich symbolische Links angesammelt haben, die von SecDocs im Rahmen der Operation retrieveSDO für AOIDs mit externen Referenz-IDs erzeugt wurden. Die Zeitspanne, nach der ein solcher symbolischer Link gelöscht wird, wird festgelegt durch den Konfigurationsparameter externalFilesRetrieveTimeout (Einstellung mandantenspezifisch oder in der Datei secdocs.properties).

Im Single-Node-Betrieb sind alle genannten Monitore nach dem Hochfahren des Systems automatisch aktiviert. Bei jedem Neustart von SecDocs werden sie automatisch wieder aktiviert.

Im Multi-Node-Betrieb gilt dies nur für die Monitore des Typs SDOCounting und AOIDCleanup. Der Monitor vom Typ TSPTimer ist in diesem Fall standardmäßig deaktiviert.

Request-Body

Element PerformAction *vom Datentyp* PerformActionType

```

<xsd:complexType name="PerformActionType">
  <xsd:sequence>
    <xsd:element name="Action" type="tns:ActionSimpleType"></xsd:element>
    <xsd:element name="MandantName" type="xsd:NCName"
      maxOccurs="unbounded" minOccurs="0"></xsd:element>
    <xsd:choice maxOccurs="1" minOccurs="0">
      <xsd:element name="MonitorType" type="tns:MonitorTypeSimpleType"
        maxOccurs="1" minOccurs="1">
        </xsd:element>
      </xsd:choice>
    </xsd:sequence>
  </xsd:complexType>

```

Action Datentyp ActionSimpleType:

Gibt an, welche Operation ausgeführt werden soll.

Mögliche Werte:

startMonitor Aktiviert einen oder mehrere Überwachungsprozesse.

stopMonitor Deaktiviert einen oder mehrere Überwachungsprozesse.

MandantName NCName, optional:

Gibt an, für welchen Mandanten die Aktion ausgeführt werden soll. Fehlt die Angabe, so wird die Aktion für alle Mandanten ausgeführt.

MonitorType Datentyp MonitorTypeSimpleType:

Gibt an, welcher ereignisgesteuerte Auftrag aktiviert bzw. deaktiviert werden soll.

Mögliche Werte:

TSPTimer Aktiviert bzw. deaktiviert die zeitgesteuerte (mandantenspezifische) Überwachung der Versiegelung. Wann die nächste Versiegelung angestartet wird, hängt von der Eigenschaft `treeAge` ab. Im Multi-Node-Betrieb werden die Aktionen `startMonitor` und `stopMonitor` auf der `SecDocs`-Instanz durchgeführt, auf der die Operation `performAction` ausgeführt wurde.

SDOCounting Aktiviert bzw. deaktiviert die mengengesteuerte Überwachung der Versiegelung. Wann der nächste (mandantenspezifische) Versiegelungsauftrag gestartet wird, hängt von der Eigenschaft `treeSize` ab. Im Multi-Node-Betrieb bewirken die Aktionen `startMonitor` und `stopMonitor` eine Aktivierung bzw. Deaktivierung des Monitors für den gesamten Verbund.

AOIDCleanup	<p>Aktiviert bzw. deaktiviert das zeitgesteuerte Löschen abgelaufener reservierter AOIDs.</p> <p>Wann der nächste Auftrag losläuft, hängt von der Einstellung des Konfigurationsparameters <code>aidKeepReservedPeriod</code> ab (Einstellung entweder mandantenspezifisch oder in der Datei <code>secdocs.properties</code>).</p> <p>Im Multi-Node-Betrieb werden die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> auf der SecDocs-Instanz durchgeführt, auf der die Operation <code>performAction</code> ausgeführt wurde.</p>
AOIDWithRefCleanup	<p>Aktiviert oder deaktiviert das zeitgesteuerte Löschen abgelaufener reservierter AOIDs mit externen Referenz-IDs (Operation <code>getAOIDWithRef</code>).</p> <p>Zusammen mit den AOIDs werden auch deren externe Referenz-IDs sowie evtl. schon übertragene, aber nicht archivierte Dateien gelöscht.</p> <p>Wann der nächste Auftrag losläuft, hängt von der Einstellung des Konfigurationsparameters <code>aidWithRefKeepReservedPeriod</code> ab (Einstellung entweder mandantenspezifisch oder in der Datei <code>secdocs.properties</code>).</p> <p>Im Multi-Node-Betrieb werden die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> auf der SecDocs-Instanz durchgeführt, auf der die Operation <code>performAction</code> ausgeführt wurde.</p>
ExternalFileCleanup	<p>Aktiviert oder deaktiviert das zeitgesteuerte Löschen symbolischer Links, die im Rahmen der Operation <code>retrieveSDO</code> von SecDocs für AOIDs mit externen Referenz-IDs erzeugt wurden.</p> <p>Wann der nächste Auftrag losläuft, hängt von der Einstellung des Konfigurationsparameters <code>externalFilesRetrieveTimeout</code> ab (Einstellung entweder mandantenspezifisch oder in der Datei <code>secdocs.properties</code>).</p> <p>Im Multi-Node-Betrieb werden die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> auf der SecDocs-Instanz durchgeführt, auf der die Operation <code>performAction</code> ausgeführt wurde.</p>
ALL	<p>Aktiviert bzw. deaktiviert alle Aufträge vom Typ <code>TSPTimer</code>, <code>SDOCounting</code>, <code>AOIDCleanup</code>, <code>AOIDWithRefCleanup</code>, <code>ExternalFileCleanup</code>.</p> <p>Im Multi-Node-Betrieb erfolgt die Aktivierung bzw. Deaktivierung wie bei den einzelnen Auftragsstypen beschrieben.</p>

Response-Body

Leeres Element `result`

```
<result></result>
```

Beispiel

Die zeitgesteuerte Überwachung der Versiegelung für den Mandanten `Mandant2` soll eingeschaltet werden.

Request

```
<S:Header>
  <ns2:soapHeaderData xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    <ns2:operation>performAction</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>ArchiveAdmin</ns2:role>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <ns3:PerformAction xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <ns3:Action>startMonitor</ns3:Action>
    <ns3:MonitorType>TSPTimer</ns3:MonitorType>
    <ns3:MandantName>Mandant2</ns3:MandantName>
  </ns3:PerformAction>
</S:Body>
```

5.1.3.14 Operation `getArchiveInfo`

Eine Beschreibung der Operation `getArchiveInfo` finden Sie im Abschnitt "[Operation `getArchiveInfo`](#)".

5.2 Web-Service MandantAdminService

Der folgende Abschnitt beschreibt das Arbeiten mit dem Web-Service `MandantAdminService` (Webservice für den Mandantenadministrator). Dieser Web-Service stellt die Operationen zur Verfügung, die sich auf den jeweiligen Archivbereich eines Mandanten beziehen. Dazu gehört das Registrieren von SDO-Typen sowie das Einrichten von spezifischen Zugängen zum Archiv für die Client-Software. Den Web-Service `MandantAdminService` erreichen Sie unter folgender URL:

`http://secdocsHost:secdocsPort/archiver/ws/3.1/mandantAdmin`

i Die Grundstruktur einer SOAP-Nachricht ist für alle Web-Services in SecDocs gleich. Einen kurzen Einstieg in den Aufbau von SOAP-Request, SOAP-Response und Fault-Message finden Sie im [Abschnitt „SOAP-Nachrichten“](#).

Der folgende Text beschreibt die Optionen, die für den Web-Service `MandantAdminService` abweichend von der Grundstruktur anzugeben sind.

i Alle Beispiele, die bei der Beschreibung der einzelnen Seiten angegeben sind, sind Auzüge aus SOAP-Nachrichten und nicht ohne Modifikationen ablauffähig.

5.2.1 Zugangsprüfung

Um eine Operation des Web-Service `MandantAdminService` auszuführen, müssen Sie in den Zugangsdaten Folgendes angeben:

role

- `SecDocs_MandantAuditor`, wenn Sie die Operationen `getAuditLogFileNames` und `getLogFile` ausführen wollen
- `MandantAdmin` für alle anderen Operationen

mandant

Name des Mandanten. Damit legen Sie fest, auf welchen Teilbereich des Archivs sich die Operationen beziehen sollen.

5.2.2 Request-Header



Die Grundstruktur eines Request-Headers sowie ein Beispiel finden Sie im [Abschnitt „Request-Header“](#) für den Web-Service ArchivingService.

Der folgende Text listet die Besonderheiten auf, die für den Web-Service MandantAdminService zu beachten sind:

Datentyp TSoapHeader

```
<xs:complexType name="TSoapHeader">
  <xs:annotation>
    ...
  </xs:annotation>
  <xs:sequence>
    <xs:element name="operation" type="tns:TOperation"/>
    <xs:element name="security" type="tns:TSecurity"/>
    <xs:element name="auditID" type="xs:string" minOccurs="0"/>
    <xs:element name="aoid" type="tns:TUuid" minOccurs="0"/>
    <xs:element name="SDOType" type="xs:string" minOccurs="0"/>
    <xs:element name="coid" type="tns:TCoid"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
```

operation

Operation, die ausgeführt werden soll:

Datentyp TOperation, siehe ["Request-Header"](#)

security

Autorisierungsinformation für die Anmeldung am Web-Service (Zugangsdaten und Passwort):

Datentyp TSecurity, siehe ["Request-Header"](#)

auditID

string:

optional; Diese Zeichenkette wird in das Audit-Protokoll übertragen.

Die Client-Software meldet sich bei SecDocs mit einer Rolle an. Um im Audit-Protokoll nachzuvollziehen zu können, wer der ausführende Benutzer war, kann die Client-Software hier z.B. die Anmeldedaten dieses Benutzers angeben.

aoid Dieses Tag darf für die Operationen des Web-Service MandantAdminService nicht angegeben werden.

SDOType

Dieses Tag darf für die Operationen des Web-Service MandantAdminService nicht angegeben werden.

coid Dieses Tag darf für die Operationen des Web-Service MandantAdminService nicht angegeben werden.

Datentyp TOperation

```
<xs:simpleType name="TOperation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="createSDOType" />
    <xs:enumeration value="modifySDOType" />
    <xs:enumeration value="updateMandant" />
    <xs:enumeration value="createOrganisation" />
    <xs:enumeration value="updateOrganisation" />
    <xs:enumeration value="setCredentials" />
    <xs:enumeration value="renewHashAlgorithm" />
    <xs:enumeration value="renewTSPSignature" />
    <xs:enumeration value="getSDOTypes" />
    <xs:enumeration value="getMandantProperties" />
    <xs:enumeration value="getOrganisations" />
    <xs:enumeration value="getTSPs" />
    <xs:enumeration value="getHashAlgorithms" />
    <xs:enumeration value="getSignatureAlgorithms" />
    <xs:enumeration value="getVersion" />
    <xs:enumeration value="getAccountingData" />
    <xs:enumeration value="createPrivilege" />
    <xs:enumeration value="updatePrivilege" />
    <xs:enumeration value="deletePrivileges" />
    <xs:enumeration value="getPrivileges" />
    <xs:enumeration value="createRole" />
    <xs:enumeration value="updateRole" />
    <xs:enumeration value="deleteRoles" />
    <xs:enumeration value="getRoles" />
    <xs:enumeration value="deleteSDOType" />
    <xs:enumeration value="performAction" />
    <xs:enumeration value="getArchiveInfo" />
    <xs:enumeration value="clearAOIDs" />
    <xs:enumeration value="convertTestMandant" />
    <xs:enumeration value="getArchivingOperations" />
    <xs:enumeration value="getAuditLogFileNames" />
    <xs:enumeration value="getAuditLogFile" />
    <xs:enumeration value="modifyXAIP" />
  </xs:restriction>
</xs:simpleType>
```

Für den Web-Service MandantAdminService sind folgende Operationen möglich:

createSDOType

Registrieren eines neuen SDO-Typs, siehe ["Operation createSDOType"](#)

modifySDOType

Ändern eines bestehenden SDO-Typs, siehe "[Operation modifySDOType](#)"

updateMandant

Ändern der eigenen Eigenschaften, siehe Seite "[Operation updateMandant](#)"

createOrganisation

Anlegen einer neuen Organisationseinheit, siehe "[Operation updateMandant](#)"

updateOrganisation

Ändern der Eigenschaften einer Organisationseinheit, siehe "[Operation updateOrganisation](#)"

setCredentials

Ändern der Zugangsberechtigung, siehe "[Operation setCredentials](#)"

renewHashAlgorithm

Erzeugen neuer Hash-Werte und Evidence Records für SDOs, wenn der Hash-Algorithmus schwach geworden ist, siehe "[Operation renewHashAlgorithm](#)"

renewTSPSignature

Erzeugen eines neuen Evidence Records für SDOs, wenn die Versiegelung des SDOs erneuert werden muss, siehe "[Operation renewTSPSignature](#)"

getSDOTypes

Auflisten der registrierten SDO-Typen zu einem Mandanten, siehe "[Operation getSDOTypes](#)"

getMandantProperties

Auflisten der eigenen Eigenschaften wie Kontaktinformationen und persönliche Einstellungen, siehe "[Operation getMandantProperties](#)"

getOrganisations

Auflisten der für diesen Mandanten angelegten Organisationseinheiten, siehe "[Operation getOrganisations](#)"

getTSPs

Auflisten aller verfügbaren TSPs, siehe "[Operation getTSPs](#)"

getHashAlgorithms

Auflisten aller verfügbaren Hash-Algorithmen, siehe "[Operation getHashAlgorithms](#)"

getSignatureAlgorithms

Auflisten aller verfügbaren Signatur-Algorithmen, siehe "[Operation getSignatureAlgorithms](#)"

getVersion

Auflisten von Versionsinformationen, siehe "[Operation getVersion](#)"

getAccountingData

Zur Verfügung stellen von Abrechnungsinformationen für den Mandantenadministrator zur Abrechnung gegenüber seinen Organisationseinheiten, siehe "[Operation getAccountingData](#)"

`createPrivilege`

Erstellen eines Privilegs, siehe "[Operation createPrivilege](#)"

`updatePrivilege`

Anpassen des Funktionsumfangs eines Privilegs, siehe "[Operation updatePrivilege](#)"

`deletePrivileges`

Löschen eines Privilegs oder mehrerer Privilegien, siehe "[Operation deletePrivileges](#)"

`getPrivileges`

Ausgeben der Namen der für einen Mandanten definierten Privilegien, siehe "[Operation getPrivileges](#)"

`createRole`

Definieren von organisationsspezifischen Rollen, siehe "[Operation createRole](#)"

`updateRole`

Zuordnen eines anderen Privilegs zu einer Rolle , siehe "[Operation updateRole](#)".

`deleteRoles`

Löschen einer oder mehrerer Rollen, siehe "[Operation deleteRoles](#)"

`getRoles`

Ausgeben der Rollen einer Organisation oder eines Mandanten, siehe "[Operation getRoles](#)"

`deleteSDOType`

Löschen eines bestehenden SDOTyps, siehe "[Operation deleteSDOType](#)"

`performAction`

Auslösen von Aktionen auf dem Archiv, siehe "[Operation performAction](#)"

`getArchiveInfo`

Ausgeben von Informationen zum aktuellen Zustand der ereignisgesteuerten Aufträge, siehe "[Operation getArchiveInfo](#)"

`clearAOIDs`

Löschen aller AOIDs eines Testmandanten, siehe "[Operation clearAOIDs](#)"

`convertTestMandant`

Umwandeln eines Testmandanten in einen produktiven Mandanten, siehe "[Operation convertTestMandant](#)"

`getArchivingOperations`

Ausgeben aller Funktionen, die der Web-Service, für den der Mandant angelegt wurde (ArchivingService oder S4), ausführen kann, siehe "[Operation getArchivingOperations](#)"

getAuditLogFileNames

Ausgeben einer Liste aller verfügbaren Audit-Log-Dateien des Mandanten, siehe ["Operation getAuditLogFileNames"](#)

getAuditLogFile

Ausgeben des Inhalts einer verfügbaren mandantenspezifischen Audit-Log-Datei, siehe ["Operation getAuditLogFile"](#)

modifyXAIP

Erweitern des in Anhang F der Richtlinie TR-ESOR V1.2 festgelegten Formats XAIP (XML formatted Archive Information Package) um benutzerspezifische Definitionen.

Diese Operation wird nur für das Archivieren von Dokumenten mit dem Web-Service `s4` benötigt, siehe Handbuch ["SecDocs Archivierung von Dokumentengemäß Richtlinie TR-ESOR" \(\[SD2\] in Abschnitt "Literatur"\)](#)

Datentyp TSecurity

```
<xs:complexType name="TSecurity">
  <xs:sequence>
    <xs:element name="principal" type="TPrincipal"/>
    <xs:choice>
      <xs:element name="password" type="xs:string"/>
      <xs:element name="token" type="xs:base64Binary"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

principal Basisdaten für die Zugangsprüfung:

Datentyp TPrincipal, siehe ["Request-Header"](#)

password string:

Passwort der Rolle MandantAdmin bzw. SecDocs_MandantAuditor.

Das Passwort der Rolle MandantAdmin wird initial vom ArchiveAdmin vergeben.

Das Passwort der Rolle SecDocs_MandantAuditor wird immer vom Web-Service MandantAdminService vergeben.

Die Passwörter für beide Rollen können Sie mit dem Web-Service MandantAdminService ändern (siehe [Abschnitt „Web-ServiceMandantAdminService“](#)).

token Eine Token-Angabe ist derzeit nicht realisiert. Sie müssen den Passwort-Mechanismus verwenden.

Datentyp TPrincipal

```
<xs:complexType name="TPrincipal">
  <xs:sequence>
    <xs:element name="role" type="xs:string"/>
    <xs:element name="mandant" type="xs:string" minOccurs="0"/>
    <xs:element name="orgID" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

role MandantAdmin

für alle Operationen außer `getAuditLogFileNames` und `getAuditLogFile`

`SecDocs_MandantAuditor`

für die Operationen `getAuditLogFileNames` und `getAuditLogFile`

mandant string:

Name des Mandanten.

Die verschiedenen Mandanten, die Zugang zum Archiv erhalten, werden vom `ArchiveAdmin` eingerichtet.

Die Angabe für `mandant` ist nicht case-sensitive.

orgID Dieses Tag darf für die Operationen des Web-Service `MandantAdminService` nicht angegeben werden.

5.2.3 Request- und Response-Body

Der Aufbau von SOAP-Request und SOAP-Response ist operationsspezifisch.

- Der Web-Service `MandantAdminService` ist in der Datei `MandantAdmin.wsdl` beschrieben.
- Definition der Typen: `AdminData.xsd` und `AdminUpdateData.xsd`

5.2.3.1 Operation createSDOType

`createSDOType` richtet einen neuen SDO-Typ mit Namen, Schema und gegebenenfalls allen abhängigen Schemas und zugehörigem Filter ein.

Voraussetzungen

- Die Struktur für den SDO-Typ ist als XML-Schema(s) (als .xsd-Datei(en)) festgelegt.
- Der zugehörige Filter für den SDO-Typ muss in Form einer XML-Datei definiert sein, d.h. die Informationen zur Adressierung von Daten im SDO wurden beschrieben, die unter anderem festlegen, welches Dokument bzw. welche Dokumente zusammengehörig zu archivieren sind.

Request-Body

Element SDOType vom Datentyp SDOType

```
<xsd:complexType name="SDOType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:NCName" maxOccurs="1"
      minOccurs="1">
    </xsd:element>
    <xsd:element name="isActive" type="xsd:boolean"
      minOccurs="0">
    </xsd:element>
    <xsd:element name="Schema" maxOccurs="1" minOccurs="1"
      type="tns:SchemaSimpleType">
    </xsd:element>
    <xsd:element name="Filter" maxOccurs="1" minOccurs="1">
      <xsd:simpleType>
        <xsd:restriction base="xsd:base64Binary">
          <xsd:minLength value="1"></xsd:minLength>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="DependentSchema"
      type="tns:DependentSchemaType" maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
    <xsd:element name="isReplaceable" type="xsd:boolean"
      maxOccurs="1" minOccurs="0">
    </xsd:element>
  </xsd:sequence>
  <xsd:element name="Policy" maxOccurs="1" minOccurs="0">
    <xsd:simpleType>
      <xsd:restriction base="xsd:base64Binary">
        <xsd:minLength value="1"></xsd:minLength>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="TimestampPolicy" maxOccurs="1"
    minOccurs="0">
    <xsd:simpleType>
      <xsd:restriction base="xsd:base64Binary">
        <xsd:minLength value="1"></xsd:minLength>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
</xsd:complexType>
```

Name NCName:

Name des SDO-Typs.

Dieser Name muss im Archiv eindeutig sein. Dabei ist die Groß-/Kleinschreibung nicht relevant. Der Name darf keine Leerzeichen enthalten.

Erlaubte Zeichen: eingeschränkter XML-Name, der mit einem Buchstaben oder einem Unterstrich beginnen muss, auf den Namenszeichen (Buchstaben, Ziffern und andere Zeichen) folgen. Der Doppelpunkt ist ausgeschlossen.
Die explizite Definition der erlaubten Zeichen finden Sie z.B. unter <http://www.w3.org/TR/2009/REC-xml-names-20091208/#NT-NCName>.

isActive

boolean:
optional; gibt an, ob der SDO-Typ zur Archivierung genutzt werden kann.

Mögliche Werte:

- "true" Standardwert; SDO-Typ kann zur Archivierung genutzt werden.
- "false" SDO-Typ ist für eine (weitere) Archivierung gesperrt.
Ein SDO dieses Typs kann mit `retrieveSDO` jedoch immer gelesen werden.

Derzeit kann nur der Wert "true" angegeben werden.
Die Angabe "false" wird als Fehler abgewiesen.

Schema

base64Binary:
XML-Schema zur Beschreibung der Struktur des SDO-Typs.

Filter

base64Binary:
XML-Struktur mit Informationen zur Zusammengehörigkeit von Dokumenten und Signaturen sowie Informationen zur Adressierung von Daten im SDO für Dokumente, Signaturen und Metadaten.

DependentSchema

Beliebig viele Einträge vom Datentyp `DependentSchemaType`, siehe "[Datentyp DependentSchemaType](#)"
. Optional:
Schemas, die vom Hauptschema oder anderen abhängigen Schemas über `import`, `include`- oder `redefine`-Elemente referenziert werden.

isReplaceable

boolean:
optional; gibt an, ob SDOs dieses Typs überschrieben werden können.

Mögliche Werte:

- "true" SDOs dieses Typs können überschrieben werden
- "false" SDOs dieses Typs können nicht überschrieben werden

Standardwert: false

Policy

base64Binary:

Datei zur Steuerung der Bewertung des Ergebnisses der Prüfung von eingebetteten oder abgesetzten Signaturen bei ArchiveSubmission.

TimestampPolicy

base64Binary:

Datei zur Bewertung der Prüfung, wenn Zeitstempel als Signaturen mitgegeben werden.



Das Erstellen einer Policy Datei ist eine Service-Leistung der Fujitsu GMBH und sollte mit ihrem technischen Betreuer abgestimmt werden.

Datentyp DependentSchemaType

```
<xsd:complexType name="DependentSchemaType">
  <xsd:sequence>
    <xsd:element name="Namespace" type="xsd:anyURI"
      maxOccurs="1" minOccurs="0"/>
    <xsd:element name="SchemaLocation" type="xsd:anyURI"
      maxOccurs="1" minOccurs="1"/>
    <xsd:element name="Schema" type="tns:SchemaSimpleType"/>
  </xsd:sequence>
</xsd:complexType>
```

Namespace

anyURI, optional:

Inhalt des Attributes `namespace` des import-Elements, das auf dieses XML-Schema verweist.

SchemaLocation

anyURI, Pflichtparameter:

Inhalt des Attributes `schemaLocation`, des import-, include- oder redefine-Elements, das dieses XML-Schema referenziert.

Schema

base64Binary:

XML-Schema das importiert oder inkludiert wird.



Wenn Schemas, die in SecDocs registriert werden sollen, import-Anweisungen enthalten, müssen diese die Attribute `schemaLocation` **und** `namespace` enthalten.

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Registriert wird ein Schema, das ein weiteres Schema importiert.

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>createSDOType</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Janel</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <sdsh:auditID>MandantAdmin operation createSDOType sample client
    </sdsh:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <SDOType xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Name>UKPT1319201476970</Name>
    <Schema>PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4NCjx...
    </Schema>
    <Filter>
      PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4NCjxmaWx...
    </Filter>
    <DependentSchema>
      <Namespace>http://ts.fujitsu.com/secdocs/sdosamples/metadata
      </Namespace>
      <SchemaLocation>incl/metadata.xsd</SchemaLocation>
      <Schema>

PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4NCjxzY2h...
      </Schema>
    </DependentSchema>
    <isReplaceable>false</isReplaceable>
  </SDOType>
</S:Body>
```

Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>createSDOType</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation createSDOType sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <result xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"></result>
</soap:Body>
```

5.2.3.2 Operation `modifySDOType`

`modifySDOType` ändert das Schema, abhängige Schemas und/oder den zugehörigen Filter eines bestehenden SDO-Typs. Alle nachfolgend aufgerufenen Operationen `submitSDO` bzw. `replaceSDO` verwenden dann diese neue Definition des SDO-Typs.

`modifySDOType` ist speziell für kleinere Änderungen am Filter oder den Schemadefinitionen vorgesehen.

Ab der SecDocs Version 3.1 lassen sich mit `modifySDOType` auch die bei der Signaturprüfung nach eIDAS verwendeten Policies ändern.



SecDocs führt für Filter und Schema nur dieselben Prüfungen durch wie bei der Operation `createSDOType`. Insbesondere führt SecDocs keine semantischen

Prüfungen der neuen Definitionen für Filter und Schema(s) gegen die alten durch. Prüfen Sie deshalb vor der Ausführung der Operation `modifySDOType` Ihre Eingaben sorgfältig auf Korrektheit.

Request-Body

Element ModifySDOType *vom Datentyp* ModifySDOType

```
<xsd:complexType name="ModifySDOType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:NCName" maxOccurs="1"
      minOccurs="1">
    </xsd:element>
    <xsd:element name="Schema" maxOccurs="1" minOccurs="0"
      type="tns:SchemaSimpleType">
    </xsd:element>
    <xsd:element name="Filter" maxOccurs="1" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:base64Binary">
          <xsd:minLength value="1"></xsd:minLength>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="DependentSchema"
      type="tns:DependentSchemaType" maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
    <xsd:element name="Policy" maxOccurs="1" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:base64Binary">
          <xsd:minLength value="1"></xsd:minLength>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="TimestampPolicy" maxOccurs="1"
      minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:base64Binary">
          <xsd:minLength value="1"></xsd:minLength>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Name NCName:
Name des zu ändernden SDO-Typs.

Schema

base64Binary:
Optional:
XML-Schema zur Beschreibung der Struktur des SDO-Typs.

Filter

base64Binary:

Optional:

XML-Struktur mit Informationen zur Zusammengehörigkeit von Dokumenten und Signaturen sowie Informationen zur Adressierung von Daten im SDO für Dokumente, Signaturen und Metadaten.

DependentSchema

Beliebig viele Einträge vom Datentyp `DependentSchemaType`, siehe "[Operation createSDOType](#)".

Optional:

Schemas, die vom Hauptschema oder anderen abhängigen Schemas über import, include- oder redefine-Elemente referenziert werden.

Policy

base64Binary:

Datei zur Steuerung der Bewertung des Ergebnisses der Prüfung von eingebetteten oder abgesetzten Signaturen bei `ArchiveSubmission`.

TimestampPolicy

base64Binary:

Datei zur Bewertung der Prüfung, wenn Zeitstempel als Signaturen mitgegeben werden



Das Erstellen einer Policy Datei ist eine Service-Leistung der Fujitsu GMBH und sollte mit ihrem technischen Betreuer abgestimmt werden.

Response-Body

Leeres Element `result`

```
<result></result>
```

Beispiel

Request-Body

```
<soap:Body>
  <tns:ModifySDOType xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <tns:Name>SDOtype1</tns:Name>
    <tns:Schema>PD94bWwgdm...tYT4NCg==</tns:Schema>
    <tns:Filter>77u/PD94bW...sdGVyPg0K</tns:Filter>
  </tns:ModifySDOType>
</soap:Body>
```

Response-Body

```
<soap:Body>
  <tns:result xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
  </tns:result>
</soap:Body>
```

5.2.3.3 Operation updateMandant

updateMandant ändert die bestehenden Eigenschaften des Mandanten. Dazu gehören:

- Kontaktinformation
- Konfigurationsparameter, z.B. `TreeSize`, `TreeAge`
- TSPs, die für die Versiegelung vorgesehen sind
- Einstellungen von mandantenspezifischen Konfigurationsparametern

i Alle angegebenen Elemente werden übernommen. Nicht angegebene Werte bleiben unverändert. Wenn ein Wert nicht geändert werden soll, dürfen Sie das entsprechende Element nicht angeben. Speziell für mandantenspezifische Konfigurationsparameter gilt:

- Es werden nur diejenigen mandantenspezifischen Konfigurationsparameter geändert, die in der Liste `Properties` angegeben sind. Nicht angegebene Konfigurationsparameter bleiben unverändert.
- Wenn Sie den Wert für einen Konfigurationsparameter nicht ändern wollen, dürfen Sie das entsprechende Element nicht in der Liste `Properties` angeben.
- Wenn Sie einen Parameter aus der Liste der mandantenspezifischen Konfigurationsparameter des Mandanten entfernen wollen, sodass für diesen Parameter wieder der in der Datei `secdocs.properties` eingestellte Wert gültig wird, so müssen Sie für diesen Parameter ein leeres Element `Value` angeben.
- Die Operation `updateMandant` wird abgewiesen, wenn für einen mandantenspezifischen Konfigurationsparameter ein ungültiger Wert angegeben wird, z.B. ein kleinerer Wert als der festgelegte Minimalwert bzw. ein größerer Wert als der festgelegte Maximalwert.

Eine Liste der Konfigurationsparameter, die mandantenspezifisch eingestellt werden können, finden Sie in der [Tabelle „Mandantenspezifisch einstellbare Konfigurationsparameter“](#) (8.1.4 Mandantenspezifische Konfigurationsparameter).

! Achtung!

In folgenden Fällen werden alle bisher für diesen Mandanten aufgesammelten Statistikdaten gelöscht:

- Wenn Sie den Konfigurationsparameter `collectOperationStatistics` für den Mandanten auf den Wert `"false"` setzen
oder
- Wenn Sie den Konfigurationsparameter `collectOperationStatistics` aus der Liste der mandantenspezifischen Konfigurationsparameter des Mandanten entfernen und in der Datei `secdocs.properties` für diesen Parameter der Wert `"false"` eingestellt ist.

Request-Body

Datentyp `MandantType`

Die Beschreibung des Datentyps `MandantType` finden Sie ab ["Operation createMandant"](#).



Beachten Sie zusätzlich:

- Die Attribute Name, Path und State können Sie hier nicht ändern.
- Das Attribut TSP kann nur geändert werden, wenn noch keine SDOs vorliegen, die mit diesem TSP versiegelt wurden, der TSP also noch nicht verwendet wurde.
- Die Anzahl der eingetragenen TSPs können Sie nicht ändern.

Response-Body

Leeres Element result

```
<result></result>
```

Beispiele

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>updateMandant</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>ArchiveAdmin operation UpdateMandant sample client</ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <ns3:Mandant
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns3:DisplayName>Fujitsu</ns3:DisplayName>
    <ns3:Contact>
      <ns3:Email>xxx@example.com</ns3:Email>
    </ns3:Contact>
  </ns3:Mandant>
</S:Body>
```

Ändern von mandantenspezifischen Einstellungen

Voraussetzung: Mit createMandant seien folgende Einstellungen vorgenommen worden:

```
<Properties>
  <Property>
    <Name>aoidKeepReservedPeriod</Name>
    <Value>900</Value>
  </Property>
  <Property>
    <Name>doHTMLProtocolsOnRetrieval</Name>
    <Value>true</Value>
  </Property>
  <Property>
    <Name>nodeNameInFaultMessage</Name>
    <Value>true</Value>
  </Property>
</Properties>
```

Nun wird `updateMandant` mit folgenden Einstellungen ausgeführt:

```
<Properties>
  <Property>
    <Name>aoidKeepReservedPeriod</Name>
    <Value>1000</Value>
  </Property>
  <Property>
    <Name>nodeNameInFaultMessage</Name>
    <Value></Value>
  </Property>
</Properties>
```

Dies hat folgende Auswirkungen:

<code>aoidKeepReservedPeriod</code>	Wert wird geändert.
<code>doHTMLProtocolsOnRetrieval</code>	Wert ist bei <code>updateMandant</code> nicht angegeben, bleibt also unverändert.
<code>nodeNameInFaultMessage</code>	Element Value ist leer. <code>nodeNameInFaultMessage</code> wird daher aus der Liste der mandantenspezifischen Konfigurationsparameter entfernt. Somit gilt für den Mandanten wieder der in der Datei <code>secdocs.properties</code> eingestellte Wert.

5.2.3.4 Operation createOrganisation

createOrganisation legt im Archiv eine neue Organisationseinheit innerhalb eines Mandanten an. Dies beinhaltet die Definition der Organisationseinheit und die Definition des für diese Organisationseinheit reservierten Archivbereichs – eine Pfadangabe, unterhalb derer alle SDOs für diese Organisationseinheit archiviert werden.



Die Anzahl der Unterverzeichnisse unter einem Knoten ist limitiert. Die Grenze wird dabei vom verwendeten Storage- bzw. Dateisystem vorgegeben. Entsprechend ist auch die Anzahl der SDOs, die Sie archivieren können, limitiert.

Wenn Sie in der Filterdefinition einen statischen, SDO-spezifischen Pfad angeben, gilt die Begrenzung für den SDO-spezifischen Knoten.

Dieses Problem können Sie umgehen, indem Sie in der Filterdefinition den Ablageort mit Hilfe des Systemschlüssels \$SDOPath dynamisch strukturieren.

Request-Body

Datentyp OrganisationType

```
<xsd:complexType name="OrganisationType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:NCName"></xsd:element>
    <xsd:element name="Path" type="tns:PathType"></xsd:element>
    <xsd:element name="SDOType" type="tns:NonEmptyString"
      maxOccurs="unbounded" minOccurs="0">
      </xsd:element>
    <xsd:element name="Contact" type="tns:ContactType"></xsd:element>
    <xsd:element name="DisplayName" type="xsd:string"
      maxOccurs="1" minOccurs="0">
      </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Name NCName:

Name der Organisationseinheit.

Der Name darf keine Leerzeichen enthalten. Die Groß-/Kleinschreibung ist für den Namen nicht relevant.

Erlaubte
Zeichen:

eingeschränkter XML-Name, der mit einem Buchstaben oder einem Unterstrich beginnen muss, auf den Namenszeichen (Buchstaben, Ziffern und andere Zeichen) folgen. Der Doppelpunkt ist ausgeschlossen. Die explizite Definition der erlaubten Zeichen finden Sie z.B. unter <http://www.w3.org/TR/2009/REC-xml-names-20091208/#NT-NCName>.

Path string:

Typ `PathType`:

Pfad zum Archivbereich der Organisationseinheit des Mandanten.

Der hier angegebene Pfad ist relativ zum mandantenspezifischen Archivbereich (siehe `Path` im Typ `MandantType` auf "[Operation createMandant](#)"). Alle SDOs, die für diese Organisationseinheit archiviert werden, werden unter diesem Pfad abgelegt.

Erlaubte Zeichen: Der Pfadname darf mit keinem der folgenden Zeichen beginnen: Schrägstrich (/), Unterstrich (_) oder Punkt(.).
Der Pfadname darf keinen der Strings `"/_"`, `". . /"`, `". /"` oder `"\"` enthalten.
Der Pfadname darf für keine andere Organisationseinheit des Mandanten vergeben worden sein.

`SDOType`

`NonEmptyString`:
optional; Liste der für diese Organisationseinheit nutzbaren SDO-Typen.
Wenn Sie `SDOType` nicht angeben, stehen der neu eingerichteten Organisationseinheit alle vom `MandantAdmin` registrierten SDO-Typen zur Verfügung.

Eine Angabe von `SDOType` wird derzeit nicht unterstützt.

`Contact`

Kontaktinformation für den `MandantAdmin`, um mit der Organisationseinheit Kontakt aufnehmen zu können.

Datentyp `ContactType`, siehe "[Operation createMandant](#)".

`DisplayName`

`string`:
optional; Name der Organisation, wie er ggf. in `SecDocs` angezeigt wird. `DisplayName` unterliegt nicht den Einschränkungen des Typs `NCName`.

Response-Body

Leeres Element `result`

```
<result></result>
```

Beispiel

Request

```

<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>createOrganisation</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation createOrganisation sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <Organisation
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Name>Demo_Center</Name>
    <Path>Demo_Center</Path>
    <Contact>
      <FirstName>Zaphod</FirstName>
      <Surname>Beeblebrox</Surname>
      <Street>Last Lane 1</Street>
      <City>Milliways</City>
      <Zip>12345</Zip>
      <Country>Universe</Country>
      <Tel>123456789</Tel>
      <Email>Zaphod.Beeblebrox@universe.gov</Email>
    </Contact>
    <DisplayName>Demo_Center</DisplayName>
  </Organisation>
</S:Body>

```

Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>createOrganisation</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation createOrganisation sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <result xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"></result>
</soap:Body>
```

5.2.3.5 Operation updateOrganisation

updateOrganisation ändert die bestehenden Kontaktdaten einer Organisationseinheit.

i Alle angegebenen Elemente werden übernommen. Nicht angegebene Werte bleiben unverändert. Wenn ein Wert nicht geändert werden soll, dürfen Sie das entsprechende Element nicht angeben.

Request-Body

Datentyp OrganisationType

Die Beschreibung des Datentyps OrganisationType finden Sie ab "[Operation createOrganisation](#)".

i Beachten Sie zusätzlich: Die Attribute Path und SDOType können hier nicht geändert werden.

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>updateOrganisation</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation UpdateOrganisation sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <ns3:Organisation
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns3:Name>Demo_Center</ns3:Name>
    <ns3:Contact>
      <ns3:Email>xxx@example.com</ns3:Email>
    </ns3:Contact>
  </ns3:Organisation>
</S:Body>
```

5.2.3.6 Operation `setCredentials`

`setCredentials` ändert die Zugangsberechtigung für die angegebene Rolle:

- Wird als Rolle `MandantAdmin` angegeben, so bezieht sich die Änderung auf den Zugang zum Web-Service `MandantAdminService` selbst.
- Wird als Rolle `SecDocs_MandantAuditor` angegeben, so bezieht sich die Änderung auf den Zugang zu den Audit-Log-File-Funktionen.
- Alle anderen angegebenen Rollen ändern die Berechtigungen für den Zugang zum Web-Service `ArchivingService` oder `S4`.

Im Request-Header ist immer die Rolle `MandantAdmin` und das gerade gültige Passwort für diese Rolle anzugeben.

i Der Web-Service `ArchiveAdminService` bietet ebenfalls die Funktion `setCredentials`, siehe "[Operation `setCredentials`](#)". In diesem Fall wird mit `setCredentials` das Passwort für den Zugang zum Web-Service `ArchiveAdminService` selbst bzw. zum Web-Service `MandantAdminService` gesetzt.

Gültigkeit der Passwörter

Ein Initialpasswort für die Rollen `MandantAdmin` und `Archivar` wird mit der Operation `createMandant` gesetzt. Die Rolle `SecDocs_MandantAuditor` wird mit der Operation `createMandant` erzeugt, aber der Rolle wird kein Passwort zugewiesen. Um die Rolle verwenden zu können müssen Sie ihr erst mit `setCredentials` ein Passwort zuweisen.

Initialpasswörter für alle anderen Rollen können Sie mit der Operation `createRole` setzen, müssen es aber nicht. Ein Aufruf von `setCredentials` für die spezifizierte Rolle erlaubt den Zugang mit dem neuen Passwort, das „alte“ Passwort, falls vorhanden, bleibt aber ebenfalls gültig. Durch einen weiteren `setCredentials`-Aufruf verliert das erste Passwort seine Gültigkeit und das Passwort aus dem vorhergehenden `setCredentials`-Aufruf wird zum „alten“ Passwort. Der Zugang ist also mit den beiden zuletzt definierten Passwörtern möglich. Dies erlaubt es, die bei einem Passwortwechsel am Server und bei den Archivierungsclients notwendigen Aktionen zeitlich zu entkoppeln. Soll nur ein Passwort gelten, so muss zweimal nacheinander dasselbe Passwort angegeben werden.

Zugang mit Zertifikat

Die Zugangsberechtigung für einen Mandanten, der mit der Operation `createMandantXAIP` angelegt wurde, ist im Handbuch „["SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR" \(ISD2\) in Abschnitt "Literatur"](#)“ beschrieben.

Request-Body

Datentyp CredentialType

```
<xsd:complexType name="CredentialType">
  <xsd:sequence>
    <xsd:element name="Type">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="Password">
          </xsd:enumeration>
          <xsd:enumeration value="Certificate">
          </xsd:enumeration>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:choice>
      <xsd:element name="Credits">
        <xsd:simpleType>
          <xsd:restriction base="xsd:base64Binary">
            <xsd:minLength value="8">
            </xsd:minLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element name="Password">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="8">
            </xsd:minLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:choice>
    <xsd:element name="Role" type="xsd:string"
      minOccurs="1" maxOccurs="1">
    </xsd:element>
    <xsd:element name="Mandant" type="xsd:string" minOccurs="0">
    </xsd:element>
    <xsd:element name="OrgID" type="xsd:string"
      minOccurs="0" maxOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Type string:

Typ des Credential.

Mögliche Werte: Password | Certificate

Für die Web-Services ArchiveAdminService, MandantAdminService und ArchivingService wird nur der Typ Password unterstützt.

Der Typ Certificate ist nur für den Zugang zum Web-Service s4 vorgesehen.

Näheres dazu finden Sie im Handbuch ["SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR"](#) ([SD2] in Abschnitt "Literatur").

Credits

base64Binary:

Möglichkeit der Autorisierung

Mindestlänge: 8

Dieses Element ist nur für den Zugang zum Web-Service S4 vorgesehen und daher nur zusammen mit `Type=Certificate` erlaubt.

Password

string:

Neues Passwort für den Zugang zum Web-Service `MandantAdminService` bzw.

für

den Web-Service `ArchivingService`.

Dieses Element ist nur zusammen mit `Type=Password` erlaubt.

Mindestlänge: 8

Role string:

Rolle für den Zugang zum Web-Service (siehe [Abschnitt „Zugangsprüfung“](#)).

Zugang zum Web-Service	Rolle
<code>MandantAdminService</code>	<code>MandantAdmin</code>
<code>MandantAdminService</code>	<code>SecDocs_MandantAuditor</code>
<code>ArchivingService</code> oder S4	<code>Archivar</code>
<code>ArchivingService</code> oder S4	<code>MyRole</code>

Mandant

string:

optional; Falls `Mandant` angegeben wird, muss es der gleiche Mandant sein wie bei `mandant` im Header.

OrgID Soll mit `setCredentials` die Zugangsberechtigung für eine der von `SecDocs` defaultmäßig ausgelieferten Rollen geändert werden, so darf `OrgID` nicht angegeben werden.

Soll die Zugangsberechtigung für eine vom Anwender mit `createRole` angelegte organisationsspezifische Rolle gesetzt werden, so ist hier die Organisation anzugeben

Response-Body

Leeres Element `result`

```
<result></result>
```

Beispiel

Request

```
<soap:Header>
  <sdsh:soapHeaderData
    xmlns:sdsh="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <sdsh:operation>setCredentials</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Mandant1</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>myOldPassword</sdsh:password>
    </sdsh:security>
  </sdsh:soapHeaderData>
</soap:Header>

<soap:Body>
  <Credentials
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Type>Password</Type>
    <Password>*****</Password>
    <Role>MandantAdmin</Role>
  </Credentials>

</soap:Body>
```

5.2.3.7 Operation renewHashAlgorithm

`renewHashAlgorithm` erneuert die Hash-Werte und Evidence Records von SDOs. Diese Funktion müssen Sie für jeden TSP ausführen, bevor der vom TSP verwendete Hash-Algorithmus von der Bundesnetzagentur für ungeeignet erklärt wird. Dazu veröffentlicht die Bundesnetzagentur Vorankündigungen zu Hash-Algorithmen, die schwach werden.

`renewHashAlgorithm` berechnet für alle Dokumente, die mit Hilfe dieses TSPs versiegelt wurden, einen neuen Hash-Wert und versiegelt alle betroffenen SDOs neu. Dabei wird auch ein neuer Evidence Record geschrieben, der alle seine Vorgänger enthält (siehe [Abschnitt „Erzeugen des Evidence Records“ \(2.4.2 Erzeugen des Evidence Records\)](#)). Zur Neuberechnung des Hash-Wertes muss das SDO vom Storage-System gelesen werden.

Darüber hinaus wird der veraltete TSP aus der mandantenspezifischen Liste der TSPs, die zum Versiegeln verwendet werden, entfernt. Der TSP, der ab diesem Zeitpunkt zur Versiegelung verwendet wird, wird in die Liste aufgenommen.

i Im MultiNode-Umfeld wird die erneute Versiegelung immer nur auf dem Knoten gestartet, an den der Request gesendet wurde.

Voraussetzungen

- Ein TSP mit einem noch sicheren Hash-Algorithmus wurde mit dem Web-Service `ArchiveAdminService` angelegt (Operation `createTSP`, siehe ["Operation createTSP"](#)).
- Im Archiv müssen SDOs für diesen Mandanten vorhanden sein, die mit dem alten TSP versiegelt worden sind. Solange bei dem Mandanten noch keine versiegelten SDOs vorhanden sind, können Sie mit der Operation `updateMandant` den zu verwendenden TSP ändern (siehe [Abschnitt „Operation updateMandant“ \(5.2.3.3 Operation updateMandant\)](#)).

Request-Body

Datentyp `RenewTSPTyp`

```
<xsd:complexType name="RenewTSPTyp">
  <xsd:sequence>
    <xsd:element name="OldTSPName" type="xsd:NCName"/></xsd:element>
    <xsd:element name="NewTSPName" type="xsd:NCName"/></xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

`oldTSPName`

NCName:

Name des TSP, dessen Hash-Algorithmus schwach geworden ist. Gegebenenfalls wird dieser TSP aus der TSP-Liste des Mandanten entfernt und nicht mehr für die Versiegelung von SDOs verwendet.

`newTSPName`

NCName:

Name des TSP, dessen Hash-Algorithmus für die Ermittlung der neuen Hash-Werte verwendet werden soll. Dieser TSP wird der TSP-Liste des Mandanten hinzugefügt und für alle zukünftigen Versiegelungen verwendet.

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<soap:Header>
  <sdsh:soapHeaderData
    xmlns:sdsh="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <sdsh:operation>renewHashAlgorithm</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Mandant1</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <RenewHashAlgorithm xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <OldTSPName>TSP-DFN</OldTSPName>
    <NewTSPName>SecDocsTestTSP</NewTSPName>
  </RenewHashAlgorithm>
</soap:Body>
```

5.2.3.8 Operation `renewTSPSignature`

`renewTSPSignature` erneuert die Versiegelung mit einem neuen TSP und erzeugt einen neuen Evidence Record für jedes SDO. Der neue Evidence Record enthält den alten Evidence Record (siehe [Abschnitt „Erzeugen des Evidence Records“ \(2.4.2 Erzeugen des Evidence Records\)](#)). Der neue TSP wird in die mandantenspezifische Liste der TSPs, die zum Versiegeln verwendet werden, aufgenommen, der alte TSP wird aus der Liste entfernt.

Diese Funktion müssen Sie dann ausführen, wenn ein Zeitstempel Gefahr läuft, ungültig zu werden. Dieser Fall kann aus folgenden Gründen eintreten:

- Der Signatur-Algorithmus, der beim Erzeugen des Zeitstempels verwendet wurde, wird schwach.
- Die verwendete Schlüssellänge reicht nicht aus.
- Der Hash-Algorithmus, der zum Erzeugen des Zeitstempels verwendet wurde, wird schwach.

i Im MultiNode-Umfeld wird die erneute Versiegelung immer nur auf dem Knoten gestartet, an den der Request gesendet wurde.

Voraussetzungen

- Ein TSP mit einem besseren Signatur-Algorithmus, größerer Schlüssellänge oder besserem Hash-Algorithmus wurde mit dem Web-Service `ArchiveAdminService` angelegt (Operation `createTSP`, siehe ["Operation createTSP"](#)).
- Alter und neuer TSP müssen für das Versiegeln eines SDOs den gleichen Hash-Algorithmus verwenden.
- Im Archiv müssen Dokumente vorhanden sein, die mit dem alten TSP versiegelt worden sind.

Request-Body

Datentyp `RenewTSPTType`

Die Beschreibung des Datentyps `RenewTSPTType` finden Sie ab ["Operation renewHashAlgorithm"](#).

Response-Body

Leeres Element `result`

```
<result></result>
```

Beispiel

Request

```
<soap:Header>

  <sdsh:soapHeaderData

    xmlns:sdsh="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData">

    <sdsh:operation>renewTSPSignature</sdsh:operation>

    <sdsh:security>
      <sdsh:principal>

        <sdsh:role>MandantAdmin</sdsh:role>

        <sdsh:mandant>Mandant1</sdsh:mandant>
      </sdsh:principal>

      <sdsh:password>*****</sdsh:password>

    </sdsh:security>
  </sdsh:soapHeaderData>

</soap:Header>

<soap:Body>

  <RenewTSPSignature

    xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"

    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminData">

    <OldTSPName>TSP-DFN</OldTSPName>
    <NewTSPName>SecDocsTestTSP</NewTSPName>

  </RenewTSPSignature>

</soap:Body>
```

5.2.3.9 Operation getSDOTypes

getSDOTypes gibt eine Liste aller definierten SDO-Typen zu einem Mandanten aus.

Request-Body

Leeres Element getRequest

```
<GetRequest></GetRequest>
```

Response-Body

Element GetSDOTypes vom Datentyp GetSDOTypesResponseType

```
<xsd:complexType name="GetSDOTypesResponseType">
  <xsd:sequence>
    <xsd:element name="SDOType" type="tns:SDOType"
                  maxOccurs="unbounded" minOccurs="0">

    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

SDOType

Information über den SDO-Typ:

kein, ein oder mehrere Datentypen SDOType, siehe ["Operation createSDOType"](#).

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getSDOTypes</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation getSDOTypes sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"></GetRequest>
</S:Body>
```

Response

```
<soap:Body>
  <GetSDOTypes xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <SDOType xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>Dokument</Name>
      <Schema>PD94bWwgdmVyc2lvbj0iMS4wIi...</Schema>
      <Filter>PD94bWwgdmVyc2dH... </Filter>
      <isActive>true</isActive>
      <isReplaceable>false</isReplaceable>
    </SDOType>
    <SDOType xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>Rechnung</Name>
      <Schema>PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4NCjx4c2Q6c2NoZW1h...</Schema>
      <Filter>PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4NCjxmaWx0ZXIgc2NoZW1h...</Filter>
      <isActive>true</isActive>
      <isReplaceable>false</isReplaceable>
    </SDOType>
  </GetSDOTypes>
</soap:Body>
```

5.2.3.10 Operation `getMandantProperties`

`getMandantProperties` gibt alle Eigenschaften des Mandanten aus.

Request-Body

Leeres Element `getRequest`

```
<getRequest></getRequest>
```

Response-Body

Datentyp `MandantType`

Die Beschreibung des Datentyps `MandantType` finden Sie ab "[Operation createMandant](#)".

Das im Datentyp `MandantType` enthaltene Element `State` zeigt den Status des Mandanten an:

`State`

string:

Status des Mandanten.

Mögliche Werte:

<code>test</code>	Der Mandant ist ein Testmandant.
<code>productive</code>	Der Mandant ist produktiv.
<code>convertingToProductive</code>	Der Mandant ist ein Testmandant und wird gerade in einen produktiven Mandanten überführt (siehe Abschnitt „Operation convertTestMandant“ (5.2.3.29 Operation convertTestMandant)).
<code>clearing</code>	Der Mandant ist ein Testmandant und es werden alle AOIDs des Testmandanten gelöscht (siehe Abschnitt „Operation clearAOIDs“ (5.2.3.28 Operation clearAOIDs)).
<code>deleting</code>	Der Mandant ist ein Testmandant und er wird gerade gelöscht (siehe Abschnitt „Operation deleteTestMandant“ (5.1.3.12 Operation deleteTestMandant)).

Das im Datentyp `MandantType` enthaltene Element `Type` hat folgenden Inhalt:

Type string:

Format der Dokumente, die der Mandant archivieren kann.

Mögliche Werte:

<code>SDO</code>	Der Mandant wurde mit <code>createMandant</code> erstellt und kann Datenobjekte in Form eines XML-Containers als Submission Data Objects (SDOs) mit dem Web-Service <code>ArchivingService</code> archivieren.
<code>XAIP</code>	Der Mandant wurde mit <code>createMandantXAIP</code> erstellt und kann Datenobjekte im Format XAIP mit dem Web-Service <code>S4</code> archivieren.

Beispiel

Response

```
<soap:Envelope
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:sdsh="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/ http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <sdsh:soapHeaderData>
      <sdsh:operation>getMandantProperties</sdsh:operation>
      <sdsh:security>
        <sdsh:principal>
          <sdsh:role>MandantAdmin</sdsh:role>
          <sdsh:mandant>Fujitsu</sdsh:mandant>
        </sdsh:principal>
        <sdsh:password>*****</sdsh:password>
      </sdsh:security>
      <sdsh:auditID>MandantAdmin operation getMandantProperties sample client</sdsh:auditID>
    </sdsh:soapHeaderData>
  </soap:Header>
  <soap:Body>
    <Mandant
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
      <Name>Fujitsu</Name>
      <DisplayName>Fujitsu</DisplayName>
      <Contact>
        <FirstName>Zaphod</FirstName>
        <Surname>Beeblebrox</Surname>
        <Street>Last Lane 1</Street>
        <City>Milliways</City>
        <Zip>12345</Zip>
        <Country>Universe</Country>
        <Tel>123456789</Tel>
        <Email>xxx@example.com</Email>
      </Contact>
      <Path>archive/Fujitsu</Path>
      <TreeSize>10</TreeSize>
      <TreeAge>2</TreeAge>
      <TSP>TestTSP</TSP>
      <State>productive</State>
      <Type>SDO</Type>
      <Properties />
    </Mandant>
  </soap:Body>
</soap:Envelope>
```

5.2.3.11 Operation `getOrganisations`

`getOrganisations` gibt eine Liste aller Organisationseinheiten des Mandanten aus.

Request-Body

Leeres Element `getRequest`

```
<GetRequest></GetRequest>
```

Response-Body

Datentyp `GetOrganisationsResponseType`

```
<xsd:complexType name="GetOrganisationsResponseType">
  <xsd:sequence>
    <xsd:element name="Organisation" type="tns:OrganisationType"
      maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Organisation

Information über die Organisationseinheit:

kein, ein oder mehrere Datentypen `OrganisationType`, siehe "[Operation createOrganisation](#)".

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getOrganisations</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation getVersion sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"></GetRequest>
</S:Body>
```


Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getOrganisations</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation getVersion sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <GetOrganisations
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Organisation
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
        <Name>Demo_Center</Name>
        <Path>Demo_Center</Path>
        <Contact>
          <FirstName>Zaphod</FirstName>
          <Surname>Beeblebrox</Surname>
          <Street>Last Lane 1</Street>
          <City>Milliways</City>
          <Zip>12345</Zip>
          <Country>Universe</Country>
          <Tel>123456789</Tel>
          <Email>Zaphod.Beeblebrox@universe.gov</Email>
        </Contact>
        <DisplayName>Demo_Center</DisplayName>
      </Organisation>
    </GetOrganisations>
  </soap:Body>
```

5.2.3.12 Operation getTSPs

getTSPs gibt eine Liste aller verfügbaren TSPs aus.

Request-Body

Leeres Element getRequest

```
<GetRequest></GetRequest>
```

Response-Body

Datentyp GetTSPsResponseType

```
<xsd:complexType name="GetTSPsResponseType">
  <xsd:sequence>
    <xsd:element name="TSP" type="tns:TSPType" maxOccurs="unbounded"
      minOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

TSP Information über den TSP:
ein oder mehrere Datentypen TSPTyp, siehe ["Operation createTSP"](#).

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getTSPs</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation getTSPs sample client</ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"></GetRequest>
</S:Body>
```

Response

```
<soap:Body>
  <GetTSPs xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <TSP xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>TestTSP</Name>
      <ProviderName>OpenLimit TSP Test Tool</ProviderName>
      <URL>http://127.0.0.1:8081/</URL>
      <SigHashAlgorithmName>SHA-256</SigHashAlgorithmName>
      <HashAlgorithmName>SHA-256</HashAlgorithmName>
      <SignatureAlgorithmName>id-RSASSA-PSS</SignatureAlgorithmName>
      <KeyLength>4096</KeyLength>
      <CertificateName>Test</CertificateName>
      <SignatureQuality>ADVANCED</SignatureQuality>
      <isActive>true</isActive>
      <doEnrich>false</doEnrich>
      <Policy>PD94bWwgdmVyc2lrbj0iMS4wIiBlbmN
        . . . . .
        b25zdHJhaw50c1BhcmFtZXRlcnM+Cg==
      </Policy>
    </TSP>
  </GetTSPs>
</soap:Body>
```

5.2.3.13 Operation `getHashAlgorithms`

`getHashAlgorithms` gibt eine Liste aller verfügbaren Hash-Algorithmen aus.

Request-Body

Leeres Element `getRequest`

```
<GetRequest></GetRequest>
```

Response-Body

Datentyp `GetHashAlgorithmsResponseType`

```
<xsd:complexType name="GetHashAlgorithmsResponseType">
  <xsd:sequence>
    <xsd:element name="Algorithm" type="tns:AlgorithmType"
      maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Algorithm

Information über den Hash-Algorithmus:

kein, ein oder mehrere Datentypen `AlgorithmType`, siehe "[Operation `getHashAlgorithms`](#)".

Beispiel

Response

```
<soap:Body>
  <GetHashAlgorithms
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
        <Name>SHA-1</Name>
        <Type>Hash</Type>
        <OID>1.3.14.3.2.26</OID>
        <URI>http://www.w3.org/2000/09/xmldsig#sha1</URI>
        <ExpirationDate>2010-12-31T23:59:59Z</ExpirationDate>
      </Algorithm>
      <Algorithm
        xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
          <Name>SHA-384</Name>
          <Type>Hash</Type>
          <OID>2.16.840.1.101.3.4.2.2</OID>
          <URI>http://www.w3.org/2001/04/xmenc#sha384</URI>
          <ExpirationDate>2022-12-31T09:30:47Z</ExpirationDate>
        </Algorithm>
        <Algorithm
          xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
            <Name>SHA-224</Name>
            <Type>Hash</Type>
            <OID>2.16.840.1.101.3.4.2.4</OID>
            <URI>http://www.w3.org/2001/04/xmenc#sha224</URI>
            <ExpirationDate>2015-12-31T09:30:47Z</ExpirationDate>
          </Algorithm>
        ...
        ...
      </GetHashAlgorithms>
    </soap:Body>
```

5.2.3.14 Operation `getSignatureAlgorithms`

`getSignatureAlgorithms` gibt eine Liste aller verfügbaren Signatur-Algorithmen aus.

Request-Body

Leeres Element `getRequest`

```
<GetRequest></GetRequest>
```

Response-Body

Datentyp `GetSignatureAlgorithmsResponseType`

```
<xsd:complexType name="GetSignatureAlgorithmsResponseType">
  <xsd:sequence>
    <xsd:element name="Algorithm" type="tns:AlgorithmType"
      maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Algorithm

Information über den Signatur-Algorithmus:

kein, ein oder mehrere Datentypen `AlgorithmType`, siehe "[Operation `getHashAlgorithms`](#)".

Beispiel

Response

```
<soap:Body>
  <GetSignatureAlgorithms
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>sha256WithRSAEncryption</Name>
      <Type>Signature</Type>
      <OID>1.2.840.113549.1.1.11</OID>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>rsaSignature</Name>
      <Type>Signature</Type>
      <OID>1.3.36.3.3.1</OID>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>ecdsaPlainSHA256</Name>
      <Type>Signature</Type>
      <OID>0.4.0.127.0.7.1.1.4.1.3</OID>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>rsaSignature1</Name>
      <Type>Signature</Type>
      <OID>1.2.840.113549.1.1</OID>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>rsaEncryption1</Name>
      <Type>Signature</Type>
      <OID>1.2.840.113549.1.1.1</OID>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>sha384WithECDSA</Name>
      <Type>Signature</Type>
      <OID>1.2.840.10045.4.3.3</OID>
    </Algorithm>
    <Algorithm
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>md4withRSAEncryption</Name>
      <Type>Signature</Type>
      <OID>1.2.840.113549.1.1.3</OID>
    </Algorithm>
    ...
  </GetSignatureAlgorithms>
</soap:Body>
```

5.2.3.15 Operation getVersion

getVersion liefert die aktuelle SecDocs-Version.

Request-Body

Leeres Element getRequest

```
<GetRequest></GetRequest>
```

Response-Body

Datentyp VersionType

Die Beschreibung des Datentyps VersionType finden Sie ab ["Operation getVersion"](#).

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getVersion</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation getVersion sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"></GetRequest>
</S:Body>
```

Response


```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getVersion</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation getVersion sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <GetVersion
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Component>
      <VersionString>SecDocs V3.1A01</VersionString>
      <Name>SecDocs</Name>
      <Major>3</Major>
      <Minor>1</Minor>
    </Component>
  </GetVersion>
</soap:Body>
```

5.2.3.16 Operation `getAccountingData`

`getAccountingData` stellt dem Mandantenadministrator Informationen für die Abrechnung gegenüber seinen Organisationseinheiten zur Verfügung.

Folgende Informationen werden ausgegeben:

- die Anzahl der insgesamt im Archiv existierenden versiegelten Dokumente zum Zeitpunkt der Abfrage (Archivgröße) für eine bestimmte Organisationseinheit des Mandanten oder für alle Organisationseinheiten des Mandanten zusammen
- die Anzahl der bisher durchgeführten Versiegelungsvorgänge für eine bestimmte Organisationseinheit des Mandanten oder für alle Organisationseinheiten des Mandanten zusammen



Eine ausführliche Beschreibung des Accounting finden Sie im [Abschnitt „Accounting“](#).

Request-Body

Element `getAccountingDataMandantAdmRequest` vom Datentyp

`GetAccountingDataMandantAdmRequestType`

```
<xsd:complexType name="GetAccountingDataMandantAdmRequestType">
  <xsd:sequence>
    <xsd:element name="OrganisationName" type="xsd:NCName"
      minOccurs="0" maxOccurs="1" />
  </xsd:sequence>
</xsd:complexType>
```

`OrganisationName`

NCName:

gibt die Organisationseinheit an, für die die Accounting-Information ausgegeben werden soll.

Fehlt das Element `OrganisationName`, so wird für die einzelnen Accounting-Daten jeweils die Summe über alle Organisationseinheiten ausgegeben.

Response-Body

Element `getAccountingDataMandantAdmResponse` vom Datentyp

`GetAccountingDataMandantAdmResponseType`

```
<xsd:complexType name="GetAccountingDataMandantAdmResponseType">
  <xsd:sequence>
    <xsd:element name="NumberOfSealedDocuments" type="xsd:long" />
    <xsd:element name="NumberOfSealingActions" type="xsd:long" />
  </xsd:sequence>
</xsd:complexType>
```

NumberOfSealedDocuments

long:

enthält die Anzahl der versiegelten Dokumente im Archiv für den im Request-Body angegebenen Bereich.

NumberOfSealingActions

long:

enthält die Anzahl der Versiegelungsvorgänge für den im Request-Body angegebenen Bereich.

Beispiel

SOAP-Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getAccountingData</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation getVersion sample client</ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <getAccountingDataMandantAdmRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <OrganisationName>Demo_Center</OrganisationName>
  </getAccountingDataMandantAdmRequest>
</S:Body>
```

SOAP-Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getAccountingData</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation getVersion sample client</sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <getAccountingDataMandantAdmResponse
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <NumberOfSealedDocuments
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
      0
    </NumberOfSealedDocuments>
    <NumberOfSealingActions
      xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs">
      0
    </NumberOfSealingActions>
  </getAccountingDataMandantAdmResponse>
</soap:Body>
```

5.2.3.17 Operation createPrivilege

Mit `createPrivilege` können Sie eine Gruppe von Operationen zu einem Privileg zusammenfassen. Dieses Privileg ist dann bei der Operation `createRole` (siehe "[Operation createRole](#)") anzugeben, um den Funktionsumfang einer Rolle festzulegen.

Ein Privileg können Sie für mehrere Rollen verwenden.

RequestBody

Element Privilege vom Datentyp PrivilegeType

```
<xsd:complexType name="PrivilegeType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:NCName"
      maxOccurs="1" minOccurs="0"/>
    <xsd:element name="Operation" type="tns:NonEmptyString"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

Name NCName:

Name des Privilegs. Es sind beliebige Namen möglich. Lediglich Namen mit dem

Präfix `SecDocs_` sind nicht erlaubt, da sie für `SecDocs`-interne Namen reserviert

sind. Die Groß-/Kleinschreibung ist für den Namen nicht relevant.

Operation

NonEmptyString:

Liste der Operationen, die zu diesem Privileg gehören sollen. Es sind lediglich Operationen des Web-Service erlaubt, für den der Mandant angelegt wurde (`ArchivingService` oder `S4`).

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>createPrivilege</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>
      MandantAdmin operation CreatePrivilege sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <Privilege
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Name>Priv_Submit</Name>
    <Operation>getAOID</Operation>
    <Operation>submitSDO</Operation>
    <Operation>retrieveSDO</Operation>
  </Privilege>
</S:Body>
```

5.2.3.18 Operation updatePrivilege

Mit `updatePrivilege` können Sie den Funktionsumfang eines Privilegs anpassen. Die bisher gültige Liste der Funktionen wird durch die neu angegebene Liste ersetzt. Änderungen an einem Privileg haben unmittelbar Auswirkungen auf alle Rollen, die dieses Privileg verwenden.

RequestBody

Element Privilege vom Datentyp PrivilegeType

```
<xsd:complexType name="PrivilegeType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:NCName"
      maxOccurs="1" minOccurs="0"/>
    <xsd:element name="Operation" type="tns:NonEmptyString"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

Name **NCName:**

Name des Privilegs. Angegeben werden können alle Privilegien, die ein Mandant selbst definiert hat. SecDocs-Interne Privilegien können nicht verändert werden.

Operation

NonEmptyString:

Liste der Operationen, die zu diesem Privileg gehören sollen. Es sind lediglich Operationen des Web-Service erlaubt, für den der Mandant angelegt wurde (ArchivingService oder S4).

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <soapHeaderData xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <operation>updatePrivilege</operation>
    <security>
      <principal>
        <role>MandantAdmin</role>
        <mandant>JaneUpdate1</mandant>
      </principal>
      <password>*****</password>
    </security>
  </soapHeaderData>
</S:Header>
<S:Body>
  <ns3:Privilege xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <ns3:Name>Priv_Submit</ns3:Name>
    <ns3:Operation>getAOID</ns3:Operation>
    <ns3:Operation>submitSDO</ns3:Operation>
    <ns3:Operation>statusSDO</ns3:Operation>
  </ns3:Privilege>
</S:Body>
```

5.2.3.19 Operation deletePrivileges

Mit deletePrivileges können Privilegien gelöscht werden. Voraussetzung ist, dass das Privileg in keiner Rolle mehr Verwendung findet.

Die Privilegien werden in der Reihenfolge gelöscht, in der sie angegeben sind. Tritt ein Fehler während des Löschens eines Privilegs auf, so wird eine Exception erzeugt und das betreffende Privileg sowie alle weiteren Privilegien in der Liste bleiben bestehen.

RequestBody

Element SelectByName vom Datentyp GetByNameType

```
<xsd:complexType name="GetByNameType">
  <xsd:sequence>
    <xsd:element name="Name" type="tns:NonEmptyString"
      maxOccurs="unbounded" minOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Name NCName:

Name des zu löschenden Privilegs. Angegeben werden können alle Privilegien, die ein Mandant selbst definiert hat. SecDocs-interne Privilegien können nicht gelöscht werden.

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>deletePrivileges</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>
      MandantAdmin operation deletePrivileges sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <SelectByName
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Name>Priv_Submit</Name>
  </SelectByName>
</S:Body>
```

5.2.3.20 Operation `getPrivileges`

`getPrivileges` gibt die Liste der aktuell für diesen Mandanten definierten Privilegien zurück.

Request-Body

Leeres Element `GetRequest`

```
<GetRequest></GetRequest>
```

Response-Body

Element `GetPrivileges` vom Datentyp `GetPrivilegeResponseType`

```
<xsd:complexType name="GetPrivilegeResponseType">
  <xsd:sequence>
    <xsd:element name="Privilege" type="tns:PrivilegeType"
      maxOccurs="unbounded" minOccurs="0"></xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

`Privilege`

Information über ein Privileg:

kein, ein oder mehrere Datentypen `PrivilegeType`, siehe "[Operation createPrivilege](#)".

Beispiel

Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>JaneUpdate1</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:operation>getPrivileges</sdsh:operation>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <GetPrivileges xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Privilege xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>Priv_Submit</Name>
      <Operation>getAOID</Operation>
      <Operation>submitSDO</Operation>
      <Operation>retrieveSDO</Operation>
    </Privilege>
  </GetPrivileges>
</soap:Body>
```

5.2.3.21 Operation createRole

Mit der Operation `createRole` können Sie organisationsspezifische Rollen definieren und deren Zugangsberechtigung setzen (eine Änderung der Zugangsberechtigung ist mit der Operation `setCredentials` möglich).

Nach Einrichten eines Mandanten steht automatisch die mandantenglobale Rolle `Archivar` für die Archivierungsfunktionen zur Verfügung. Die Rolle `Archivar` kann alle Operationen des Web-Service, für den der Mandant angelegt wurde (`ArchivingService` oder `S4`), für alle Organisationen ausführen. Durch das Einrichten einer organisationspezifischen Rolle wird die mandantenglobale Rolle `Archivar` deaktiviert und der globale Zugang gesperrt.

Zur Definition einer Rolle gehören:

- Name der Rolle
- Name eines benutzerdefinierten Privilegs (`createPrivilege`)
- Name der Organisation, für die diese Rolle definiert wird (Gültigkeitsbereich)
- Zugang (Credentials) für die neue Rolle

Die organisationsspezifische Archivar-Rolle

Ein Spezialfall ist die organisationsspezifische Archivar-Rolle. Diese Rolle hat folgende Eigenschaften:

- Sie hat den Namen `Archivar`.
- Der Funktionsumfang dieser Rolle umfasst automatisch alle Operationen, die durch den Web-Service, für den der Mandant angelegt wurde (`ArchivingService` oder `S4`), bereitgestellt werden. Dies gilt auch für zusätzliche Operationen, die in einer Folgeversion durch diesen Web-Service bereitgestellt werden. Deshalb kann und darf beim Anlegen dieser Rolle der Funktionsumfang nicht angegeben werden.
- Name der Organisationseinheit, für die diese Rolle definiert wird (Gültigkeitsbereich).
- Der Zugang (Credentials) wird mit einer organisationsspezifischen Zugangsberechtigung geschützt (wie jede andere organisationsspezifische Rolle).

Request-Body

Element `CreateRole` vom Datentyp `CreateRoleType`

```
<xsd:complexType name="CreateRoleType">
  <xsd:sequence>
    <xsd:element name="Role" type="tns:RoleType"/>
    <xsd:element name="Credential" type="tns:CredentialType"
      minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

Role Datentyp `RoleType`, siehe "[Operation createRole](#)".
Daten der Rolle (Name, Gültigkeitsbereich, Privileg).

Credentials Datentyp `CredentialType`, siehe "[Operation setCredentials](#)":
Zugangsdaten für die neue Rolle.
Sind beim Anlegen einer Rolle keine Zugangsdaten angegeben, so wird der Zugriff für diese Rolle von SecDocs solange verweigert, bis mit der Funktion `setCredentials` Zugangsdaten gesetzt sind.

Datentyp `RoleType`

```
<xsd:complexType name="RoleType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:NCName"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="Scope" maxOccurs="1" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:whiteSpace value="collapse"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:choice maxOccurs="1" minOccurs="0">
      <xsd:element name="Privilege" type="tns:PrivilegeType"
        maxOccurs="1" minOccurs="1">
      </xsd:element>
      <xsd:element name="PrivilegeName" type="xsd:NCName"
        maxOccurs="1" minOccurs="1">
      </xsd:element>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>
```

Name

NCName:

Name der Rolle

Der Name darf keine Leerzeichen enthalten. Namen mit dem Präfix `SecDocs_` sind nicht zugelassen, da diese Namen in SecDocs für weitere System-Rollen reserviert sind. Die Groß-/Kleinschreibung ist für den Namen nicht relevant.

Erlaubte Zeichen: eingeschränkter XML-Name, der mit einem Buchstaben oder einem Unterstrich

beginnen muss, auf den Namenszeichen (Buchstaben, Ziffern und andere Zeichen) folgen. Der Doppelpunkt ist ausgeschlossen.

Die explizite Definition der erlaubten Zeichen finden Sie z.B. unter <http://www.w3.org/TR/2009/REC-xml-names-20091208/#NT-NCName>.

Scope	string Gültigkeitsbereich. Name einer existierenden Organisationseinheit, für die die Rolle definiert wird.
PrivilegeName	NCName: Name eines Privilegs, das mit <code>createPrivilege</code> angelegt wurde (siehe " Operation createPrivilege "). Wird als Rollenname <code>Archivar</code> angegeben, können Sie <code>PrivilegeName</code> nicht angeben.
Privilege	PrivilegeType derzeit nicht unterstützt

Response-Body

Leeres Element `result`

```
<result></result>
```

Beispiel

Um mit einer Rolle die Funktionen einzuschränken, die diese Rolle ausführen darf muss zunächst ein Privileg mit dem gewünschten Funktionsumfang erstellt werden. Danach kann dann die Rolle unter Verwendung dieses Privilegs angelegt werden.

Request

Operation createPrivilege:

```
<S:Header>

  <soapHeaderData xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"

    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminData">

    <operation>createPrivilege</operation>

    <security>
      <principal>

        <role>MandantAdmin</role>

        <mandant>JaneUpdate1</mandant>
      </principal>

      <password>*****</password>
    </security>
  </soapHeaderData>

</S:Header>

<S:Body>

  <ns3:Privilege xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"

    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <ns3:Name>MyPriv</ns3:Name>

    <ns3:Operation>submitSDO</ns3:Operation>

    <ns3:Operation>retrieveSDO</ns3:Operation>
    <ns3:Operation>getAOID</ns3:Operation>

  </ns3:Privilege>

</S:Body>
```

Operation createRole:


```

<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>createRole</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation CreateRole sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <CreateRole
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Role>
      <Name>MyRole</Name>
      <Scope>Department1</Scope>
      <PrivilegeName>Priv_Submit</PrivilegeName>
    </Role>
    <Credentials>
      <Type>Password</Type>
      <Password>*****</Password>
    </Credentials>
  </CreateRole>
</S:Body>

```

5.2.3.22 Operation updateRole

Mit updateRole können Sie einer Rolle ein anderes Privileg zuordnen. Die übrigen Elemente dienen der Identifikation der Rolle.

i Die Standardrolle Archivar können Sie mit updateRole nicht ändern.

Request-Body

Element Role vom Datentyp RoleType

```
<xsd:complexType name="RoleType">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:NCName"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
    <xsd:element name="Scope" maxOccurs="1" minOccurs="0">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:whiteSpace value="collapse"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:choice maxOccurs="1" minOccurs="0">
      <xsd:element name="Privilege" type="tns:PrivilegeType"
        maxOccurs="1" minOccurs="1">
      </xsd:element>
      <xsd:element name="PrivilegeName" type="xsd:NCName"
        maxOccurs="1" minOccurs="1">
      </xsd:element>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>
```

Name	NCName: Name der Rolle
Scope	string Gültigkeitsbereich der Rolle
PrivilegeName	NCName: Name eines Privilegs, das mit createPrivilege angelegt wurde (siehe " Operation createPrivilege ").

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>updateRole</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation UpdateRole sample client</ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <Role xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Name>MyRole</Name>
    <Scope>Department1</Scope>
    <PrivilegeName>Priv_Update</PrivilegeName>
  </Role>
</S:Body>
```

5.2.3.23 Operation deleteRoles

deleteRoles löscht eine oder mehrere Rollen. Wird die letzte organisationspezifische Rolle eines Mandanten gelöscht, so wird die mandantenglobale Rolle *Archivar*, die alle Operationen enthält, wieder aktiviert. Der Zugang mit dieser Rolle muss durch explizites Setzen eines Passworts (mit `setCredentials`, siehe "[Operation setCredentials](#)") wieder freigeschaltet werden.

Die Rollen werden in der Reihenfolge gelöscht, in der sie angegeben sind. Tritt ein Fehler während des Löschens einer Rolle auf, so wird eine Exception erzeugt und die betreffende Rolle sowie alle weiteren Rollen in der Liste bleiben bestehen.

Request-Body

Element DeleteRole vom Datentyp DeleteRoleType

```
<xsd:complexType name="DeleteRoleType">
  <xsd:sequence>
    <xsd:element name="Role" type="tns:RoleType" maxOccurs="unbounded"
                  minOccurs="1"></xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Role Datentyp RoleType, siehe "[Operation createRole](#)":

Rolle, die gelöscht werden soll. Zur Identifizierung einer Rolle müssen die Elemente `Name` und `Scope` angegeben werden. Weitere Angaben sind nicht zulässig

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>deleteRoles</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation deleteRole sample client</ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <DeleteRole
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Role>
      <Name>MyRole</Name>
      <Scope>Department1</Scope>
    </Role>
  </DeleteRole>
</S:Body>
```

5.2.3.24 Operation getRoles

getRoles gibt entweder eine Liste der Rollen aus, die für eine Organisation definiert wurden, oder die Liste aller Rollen eines Mandanten. Die Liste aller Rollen eines Mandanten enthält auch die von SecDocs automatisch erzeugten Rollen.

Request-Body

Element GetRolesRequest *vom Datentyp* GetRolesRequestType

```
<xsd:complexType name="GetRoleRequestType">
  <xsd:sequence>
    <xsd:element name="organisation" type="xsd:string"
      maxOccurs="unbounded" minOccurs="0">
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
```

organisation string:

Name der Organisation, deren Rollen ausgegeben werden sollen. Das Element organisation können Sie mehrfach angeben. Falls organisation fehlt, werden alle Rollen des Mandanten aufgezählt.

Response-Body

Element GetRoles *vom Datentyp* GetRolesResponseType

```
<xsd:complexType name="GetRolesResponseType">
  <xsd:sequence>
    <xsd:element name="Role" type="tns:RoleType"
      minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
  </xsd:complexType>
```

Role Kein, ein oder mehrere Datentypen RoleType, siehe "[Operation createRole](#)": Information über die Rolle. RoleType enthält den Namen der Rolle, die Organisation, für die die Rolle definiert ist, sowie das Privileg und dessen Operationen.

Beispiel

Request

```

<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getRoles</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation getRoles sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <GetRolesRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData" />
</S:Body>

```

Response

Vor dem Aufruf von createRole:

```

<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getRoles</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation getRoles sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <GetRolesResponse
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Role xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <Name>MandantAdmin</Name>
      <Scope>SecDocs_Mandant</Scope>
      <Privilege>
        <Operation>createOrganisation</Operation>
        <Operation>createSDOType</Operation>
        <Operation>modifySDOType</Operation>
        <Operation>getTSPs</Operation>
        <Operation>getHashAlgorithms</Operation>
        <Operation>getSignatureAlgorithms</Operation>
      </Privilege>
    </Role>
  </GetRolesResponse>
</soap:Body>

```

```

        <Operation>getSDOTypes</Operation>
        <Operation>getOrganisations</Operation>
        <Operation>setCredentials</Operation>
        <Operation>getVersion</Operation>
        <Operation>renewHashAlgorithm</Operation>
        <Operation>renewTSPSignature</Operation>
        <Operation>updateMandant</Operation>
        <Operation>updateOrganisation</Operation>
        <Operation>getMandantProperties</Operation>
        <Operation>getRoles</Operation>
        <Operation>createRole</Operation>
        <Operation>deleteRoles</Operation>
        <Operation>updateRole</Operation>
        <Operation>deleteSDOType</Operation>
        <Operation>getAccountingData</Operation>
        <Operation>createPrivilege</Operation>
        <Operation>deletePrivilege</Operation>
        <Operation>updatePrivilege</Operation>
        <Operation>deletePrivileges</Operation>
        <Operation>getPrivileges</Operation>
        <Operation>performAction</Operation>
        <Operation>getArchiveInfo</Operation>
        <Operation>clearAOIDs</Operation>
        <Operation>convertTestMandant</Operation>
        <Operation>getArchivingOperations</Operation>
    </Privilege>
</Role>
<Role xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Name>SecDocs_MandantAuditor</Name>
    <Scope>SecDocs_Mandant</Scope>
    <Privilege>
        <Operation>getAuditLogFileNames</Operation>
        <Operation>getAuditLogFile</Operation>
    </Privilege>
</Role>
<Role xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Name>Archivar</Name>
    <Scope>SecDocs_Mandant</Scope>
    <Privilege>
        <Operation>submitSDO</Operation>
        <Operation>getAOID</Operation>
        <Operation>retrieveSDO</Operation>
        <Operation>deleteSDO</Operation>
        <Operation>forceDeleteSDO</Operation>
        <Operation>metaDataSDO</Operation>
        <Operation>statusSDO</Operation>
        <Operation>retrieveMetaData</Operation>
        <Operation>requestForEvidence</Operation>
        <Operation>navigate</Operation>
        <Operation>getSchemaDataSDO</Operation>
        <Operation>moveSDO</Operation>
        <Operation>setExpirationDateTimeSDO</Operation>
        <Operation>forceDeferredSDO</Operation>
        <Operation>getAccountingData</Operation>
        <Operation>sparqlQuery</Operation>
        <Operation>simpleQuery</Operation>
        <Operation>listSDOVersions</Operation>
        <Operation>replaceSDO</Operation>
        <Operation>getVersion</Operation>
    </Privilege>
</Role>

```



```

                <Operation>getAOIDWithRef</Operation>
            </Privilege>
        </Role>
    </GetRolesResponse>
</soap:Body>

```

Nach dem Aufruf von createRole:

```

<soap:Header>
    <sdsh:soapHeaderData>
        <sdsh:operation>getRoles</sdsh:operation>
        <sdsh:security>
            <sdsh:principal>
                <sdsh:role>MandantAdmin</sdsh:role>
                <sdsh:mandant>Fujitsu</sdsh:mandant>
            </sdsh:principal>
            <sdsh:password>*****</sdsh:password>
        </sdsh:security>
        <sdsh:auditID>MandantAdmin operation getRoles sample client</sdsh:auditID>
    </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
    <GetRolesResponse
        xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
        <Role xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
            <Name>MandantAdmin</Name>
            <Scope>SecDocs_Mandant</Scope>
            <Privilege>
                <Operation>createOrganisation</Operation>
                <Operation>createSDOType</Operation>
                <Operation>modifySDOType</Operation>
                <Operation>getTSPs</Operation>
                <Operation>getHashAlgorithms</Operation>
                <Operation>getSignatureAlgorithms</Operation>
                <Operation>getSDOTypes</Operation>
                <Operation>getOrganisations</Operation>
                <Operation>setCredentials</Operation>
                <Operation>getVersion</Operation>
                <Operation>renewHashAlgorithm</Operation>
                <Operation>renewTSPSignature</Operation>
                <Operation>updateMandant</Operation>
                <Operation>updateOrganisation</Operation>
                <Operation>getMandantProperties</Operation>
                <Operation>getRoles</Operation>
                <Operation>createRole</Operation>
                <Operation>deleteRoles</Operation>
                <Operation>updateRole</Operation>
                <Operation>deleteSDOType</Operation>
                <Operation>getAccountingData</Operation>
                <Operation>createPrivilege</Operation>
                <Operation>deletePrivilege</Operation>
                <Operation>updatePrivilege</Operation>
                <Operation>deletePrivileges</Operation>
                <Operation>getPrivileges</Operation>
                <Operation>performAction</Operation>
                <Operation>getArchiveInfo</Operation>
            </Privilege>
        </Role>
    </GetRolesResponse>
</soap:Body>

```

```
                <Operation>clearAOIDs</Operation>
                <Operation>convertTestMandant</Operation>
                <Operation>getArchivingOperations</Operation>
            </Privilege>
        </Role>
        <Role xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
            <Name>SecDocs_MandantAuditor</Name>
            <Scope>SecDocs_Mandant</Scope>
            <Privilege>
                <Operation>getAuditLogFileNames</Operation>
                <Operation>getAuditLogFile</Operation>
            </Privilege>
        </Role>
        <Role xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
            <Name>MyRole</Name>
            <Scope>Department1</Scope>
            <Privilege>
                <Name>Priv_Submit</Name>
                <Operation>getAOID</Operation>
                <Operation>submitSDO</Operation>
                <Operation>retrieveSDO</Operation>
            </Privilege>
        </Role>
    </GetRolesResponse>
</soap:Body>
```

5.2.3.25 Operation deleteSDOType

deleteSDOType löscht einen bestehenden SDOType.

Voraussetzungen

Es sind keine SDOs mit diesem SDOType archiviert.

Request-Body

Element SelectByName vom Datentyp GetByNameType

```
<xsd:complexType name="GetByNameType">
  <xsd:sequence>
    <xsd:element name="Name" type="tns:NonEmptyString"
      maxOccurs="unbounded" minOccurs="1">
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
```

Name NonEmptyString:
Name des zu löschenden SDOTyps.

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```

<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>deleteSDOType</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation deleteSDOType sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <SelectByName
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <Name>FileContainer</Name>
  </SelectByName>
</S:Body>

```

Response

```

<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>deleteSDOType</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation deleteSDOType sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <result xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"></result>
</soap:Body>

```

5.2.3.26 Operation `performAction`

`performAction` löst Aktionen auf dem Archiv aus. Angeboten werden folgende Aktionen:

- `sealDocuments`

Im normalen Betrieb wird die Versiegelung (also das Einholen eines Zeitstempels und Erstellung des Evidence Records) der archivierten SDOs automatisch gestartet, wenn einer der eingestellten Schwellenwerte (`treeSize` bzw. `treeAge`) erreicht und die entsprechende Schwellwert-Überwachung nicht explizit angehalten wurde.

Mit der Operation `sealDocuments` kann die Versiegelung aller aktuell anstehenden SDOs jederzeit gestartet werden, unabhängig davon, ob einer der Schwellenwerte erreicht oder die Schwellwert-Überwachung deaktiviert wurde. Die Operation startet die Versiegelung der wartenden SDOs und kehrt dann zum Aufrufer der Operation zurück. Der eigentliche Versiegelungsvorgang läuft im Hintergrund. Der Fortschritt der Operation lässt sich über die Funktion `getArchiveInfo` verfolgen.

i Im MultiNode-Umfeld wird die erneute Versiegelung immer nur auf dem Knoten gestartet, an den der Request gesendet wurde.

- `stopSealing`, `stopTimestamping`, `stopRenew`

Deaktivieren die aktuell durchgeführte Versiegelung von Dokumenten.

Im Multi-Node-Betrieb werden diese Aktionen auf derjenigen SecDocs-Instanz durchgeführt, auf der die Operation `performAction` ausgeführt wurde.

-
- `startMonitor`, `stopMonitor`

Folgende Überwachungsprozesse (Monitore) können mit der Operation `stopMonitor` angehalten (die durch die Überwachungsprozesse bereits gestarteten Aufträge laufen davon unberührt bis zum Ende weiter) und mit der Operation `startMonitor` aktiviert werden.

- `SDOCounting`:
Überwachung der Anzahl, der für die Versiegelung anstehenden SDOs
Wird der Wert `TreeSize` (konfigurierbar mit der Operation `updateMandant` des Web-Service `MandantAdminService`) erreicht, so werden die anstehenden SDOs versiegelt (also Aufbau eines Hash-Baums, einholen eines Zeitstempels, erzeugen eines Evidence Records je SDO).
- `TSPTimer`:
Überwachung der maximalen Wartezeit eines SDOs auf Versiegelung (`TreeAge`) Wird `TreeAge` (konfigurierbar mit der Operation `updateMandant` des Web-Service `MandantAdminService`) erreicht, so werden die anstehenden SDOs versiegelt (also Aufbau eines Hash-Baums, einholen eines Zeitstempels, erzeugen eines Evidence Records je SDO).
- `AOIDCleanup`:
Prüfung auf reservierte AOIDs
In periodischen Abständen wird geprüft, ob AOIDs reserviert wurden (Operation `getAOID` des Web-Service `ArchivingService`), deren Reservierung inzwischen abgelaufen ist (die maximale Reservierungsdauer können Sie mit dem Konfigurationsparameter `aoidKeepReservedPeriod` in der Datei `secdocs.properties` festlegen).
- `AOIDWithRefCleanup`:
In periodischen Abständen wird geprüft, ob AOIDs mit externen Referenz-IDs reserviert wurden (Operation `getAOIDWithRef` des Web-Service `ArchivingService`), deren Reservierung inzwischen abgelaufen ist. Die maximale Reservierungsdauer wird festgelegt durch den Konfigurationsparameter `aoidWithRefKeepReservedPeriod` (Einstellung mandantenspezifisch oder in der Datei `secdocs.properties`).
- `ExternalFileCleanup`:
In periodischen Abständen wird geprüft, ob sich symbolische Links angesammelt haben, die von SecDocs im Rahmen der Operation `retrieveSDO` für AOIDs mit externen Referenz-IDs erzeugt wurden. Die Zeitspanne, nach der ein solcher symbolischer Link gelöscht wird, wird festgelegt durch den Konfigurationsparameter `externalFilesRetrieveTimeout` (Einstellung mandantenspezifisch oder in der Datei `secdocs.properties`).

Im Single-Node-Betrieb sind alle genannten Monitore nach dem Hochfahren des Systems automatisch aktiviert. Bei jedem Neustart von SecDocs werden sie automatisch wieder aktiviert.

Im Multi-Node-Betrieb gilt dies nur für die Monitore des Typs `SDOCounting` und `AOIDCleanup`. Der Monitor vom Typ `TSPTimer` ist in diesem Fall standardmäßig deaktiviert.

Request-Body

Element `PerformAction` vom Datentyp `PerformActionType`

```

<xsd:complexType name="PerformActionType">
  <xsd:sequence>
    <xsd:element name="Action" type="tns:ActionSimpleType">
    </xsd:element>
    <xsd:element name="MandantName" type="xsd:NCName"
      maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
    <xsd:choice maxOccurs="1" minOccurs="0">
      <xsd:element name="MonitorType"
        type="tns:MonitorTypeSimpleType"
        maxOccurs="1" minOccurs="1">
      </xsd:element>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>

```

Action

Datentyp ActionSimpleType:

Gibt an, welche Operation ausgeführt werden soll.

Mögliche Werte:

sealDocuments	Startet die Versiegelung der Dokumente. Diese Aktion kann nur bei Anfragen an den Web-Service MandantAdminService angegeben werden. Im Multi-Node-Betrieb wird diese Aktion auf derjenigen SecDocs-Instanz durchgeführt, auf der die Operation performAction ausgeführt wurde.
startMonitor	Aktiviert einen oder mehrere Überwachungsprozesse.
stopMonitor	Deaktiviert einen oder mehrere Überwachungsprozesse.

`stopSealing`
`stopTimestamping`
`stopRenew`

Deaktivieren die aktuell durchgeführte Versiegelung von Dokumenten ggf. abhängig vom Auslöser für den Versiegelungsprozess:

- `stopSealing` deaktiviert alle Versiegelungen (unabhängig davon, wodurch sie ausgelöst wurden).
- `stopTimestamping` deaktiviert nur Versiegelungen, die von einem Überwachungsprozess (siehe `MonitorType` auf "[Operation performAction](#)") oder explizit durch die Operation `performAction` mit `Action = sealDocuments` gestartet wurden.
- `stopRenew` deaktiviert nur Versiegelungen, die von einer der Operationen `renewHashAlgorithm` oder `renewTSPSignature` ausgelöst wurden.

Diese Aktionen können nur bei Anfragen an den Web-Service `MandantAdminService` angegeben werden. Versiegelungsaufträge des Mandanten, die zum Zeitpunkt des Aufrufs von `performAction` gerade in Bearbeitung sind, werden noch vollständig bearbeitet, jedoch werden keine neuen Versiegelungsaufträge für die weiteren aktuell zur Versiegelung anstehenden SDOs gestartet. Solange aber noch Überwachungsprozesse für `SDOCounting` oder `TSPTimer` aktiv sind, werden von ihnen auch weiterhin Versiegelungsaufträge gestartet. Soll das Versiegeln vollständig eingestellt werden, dann müssen diese Überwachungsprozesse mit `stopMonitor` deaktiviert werden (außer bei `Action = stopRenew`).

Im Multi-Node-Betrieb werden diese Aktionen auf derjenigen SecDocs-Instanz durchgeführt, auf der die Operation `performAction` ausgeführt wurde.

`MandantName` `NCName`:

Darf nicht angegeben werden (die Aktionen können durch den MandantAdmin nur für den eigenen Archivbereich ausgeführt werden).

`MonitorType`

Datentyp `MonitorTypeSimpleType`:

Darf nur für die Aktionen `startMonitor` und `stopMonitor` angegeben werden und gibt an, welcher ereignisgesteuerte Auftrag aktiviert oder deaktiviert werden soll.

Mögliche Werte:

TSPTimer	<p>Aktiviert oder deaktiviert die zeitgesteuerte Überwachung der Versiegelung. Wann die nächste Versiegelung gestartet wird, hängt von der mandantenspezifischen Eigenschaft <code>treeAge</code> ab.</p> <p>Im Multi-Node-Betrieb werden die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> auf der SecDocs-Instanz durchgeführt, auf der die Operation <code>performAction</code> ausgeführt wurde.</p>
SDOCounting	<p>Aktiviert oder deaktiviert die mengengesteuerte Überwachung der Versiegelung. Wann der nächste Versiegelungsauftrag gestartet wird, hängt von der mandantenspezifischen Eigenschaft <code>treeSize</code> ab.</p> <p>Im Multi-Node-Betrieb bewirken die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> eine Aktivierung bzw. Deaktivierung des Monitors für den gesamten Verbund.</p>
AOIDCleanup	<p>Aktiviert oder deaktiviert das zeitgesteuerte Löschen abgelaufener Reservierungen für AOIDs (Operation <code>getAOID</code>). Wann der nächste Auftrag losläuft, hängt von der Einstellung des Konfigurationsparameters <code>aoidKeepReservedPeriod</code> ab (Einstellung entweder mandantenspezifisch oder in der Datei <code>secdocs.properties</code>).</p> <p>Im Multi-Node-Betrieb werden die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> auf der SecDocs-Instanz durchgeführt, auf der die Operation <code>performAction</code> ausgeführt wurde.</p>
AOIDWithRefCleanup	<p>Aktiviert oder deaktiviert das zeitgesteuerte Löschen abgelaufener Reservierungen für AOIDs mit externen Referenz-IDs (Operation <code>getAOIDWithRef</code>). Zusammen mit den AOIDs werden auch deren externe Referenz-IDs sowie evtl. schon übertragene, aber nicht archivierte Dateien gelöscht. Wann der nächste Auftrag losläuft, hängt von der Einstellung des Konfigurationsparameters <code>aoidWithRefKeepReservedPeriod</code> ab (Einstellung entweder mandantenspezifisch oder in der Datei <code>secdocs.properties</code>).</p> <p>Im Multi-Node-Betrieb werden die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> auf der SecDocs-Instanz durchgeführt, auf der die Operation <code>performAction</code> ausgeführt wurde.</p>
ExternalFileCleanup	<p>Aktiviert oder deaktiviert das zeitgesteuerte Löschen symbolischer Links, die im Rahmen der Operation <code>retrieveSDO</code> von SecDocs für AOIDs mit externen Referenz-IDs erzeugt wurden. Wann der nächste Auftrag losläuft, hängt von der Einstellung des Konfigurationsparameters <code>externalFilesRetrieveTimeout</code> ab (Einstellung entweder mandantenspezifisch oder in der Datei <code>secdocs.properties</code>).</p> <p>Im Multi-Node-Betrieb werden die Aktionen <code>startMonitor</code> und <code>stopMonitor</code> auf der SecDocs-Instanz durchgeführt, auf der die Operation <code>performAction</code> ausgeführt wurde.</p>
ALL	<p>Aktiviert bzw. deaktiviert alle Aufträge vom Typ <code>TSPTimer</code>, <code>SDOCounting</code>, <code>AOIDCleanup</code>, <code>AOIDWithRefCleanup</code>, <code>ExternalFileCleanup</code>.</p> <p>Im Multi-Node-Betrieb erfolgt die Aktivierung bzw. Deaktivierung wie bei den einzelnen Auftragsarten beschrieben.</p>

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <soapHeaderData xmlns="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    <operation>performAction</operation>
    ...
</S:Header>
<S:Body>
  <ns3:PerformAction
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <ns3:Action>sealDocuments</ns3:Action>
  </ns3:PerformAction>
</S:Body>
```

5.2.3.27 Operation getArchiveInfo

getArchiveInfo liefert Informationen zum aktuellen Zustand der ereignisgesteuerten Aufträge. Enthalten sind folgende Informationen:

- Wie viele AOIDs sich in welchem Zustand befinden
- Welche ereignisgesteuerten Aufträge gerade laufen
- Ob die Überwachungsprozesse (Monitore) aktiviert oder deaktiviert sind

Wird die Operation am Web-Service ArchiveAdminService aufgerufen, so werden die Informationen für alle Mandanten ausgegeben (das heißt, es wird pro Mandant eine Datenstruktur ausgegeben).

Wird die Operation am Web-Service MandantAdminService aufgerufen, so wird die Information für den aufrufenden Mandanten ausgegeben.

Request-Body

Leeres Element getRequest

```
<GetRequest></GetRequest>
```

Response-Body

Element GetArchiveInfo *vom Datentyp* ArchiveInfoType , *das pro Mandant ein Element* Info *vom Datentyp* ArchiveInfoType *enthält*

```
<xsd:complexType name="ArchiveInfoType">
  <xsd:sequence>
    <xsd:element name="MandantName" type="xsd:NCName"/>
    <xsd:element name="Mandant" type="tns:MandantType" />
    <xsd:element name="TimeStampingInterval" type="xsd:string"/>
    <xsd:element name="AOIDCleanupInterval" type="xsd:string"/>
    <xsd:element name="AOIDInfo" type="tns:AOIDInfoType"/>
    <xsd:element name="Timer" type="tns:TimerType"
      maxOccurs="unbounded" minOccurs="1"/>
    <xsd:element name="SDOCountingInfo" type="tns:SDOCounterType"/>
    <xsd:element name="RunningJob" type="xsd:tns:RunningJobType"
      maxOccurs="unbounded" minOccurs="0"/>
    <xsd:element name="ScheduledJob" type="tns:ScheduledJobType"
      maxOccurs="unbounded" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

MandantName

NCName:

Name des Mandanten, zu dem die Informationen geliefert werden

Mandant	Datentyp <code>MandantType</code> , siehe " Operation createMandant ": Information über den Mandanten
TimeStampingInterval	string Durch <code>TreeAge</code> eingestelltes Zeitintervall, nachdem die Versiegelung gestartet wird
AOIDCleanupInterval	string Zeitintervall, nachdem die Bereinigung der AOIDs gestartet wird
AOIDInfo	Datentyp <code>AOIDInfoType</code> , siehe " Operation getArchiveInfo ": Informationen über AOIDs
Timer	Datentyp <code>TimerType</code> , siehe " Operation getArchiveInfo ": Informationen über die Überwachungsprozesse (Monitore)
SDOCountingInfo	Datentyp <code>SDOCounterType</code> , siehe " Operation getArchiveInfo ": Informationen über die Anzahl der SDOs
RunningJob	Datentyp <code>RunningJobType</code> , siehe " Operation getArchiveInfo ": Informationen über die laufenden Aufträge
ScheduledJob	Datentyp <code>ScheduledJobType</code> , siehe " Operation getArchiveInfo ": Beschreibt die Aufträge, die als nächstes gestartet werden

Datentyp `AOIDInfoType`

```

<xsd:complexType name="AOIDInfoType">
  <xsd:sequence>
    <xsd:element name="Reserved" type="xsd:integer" />
    <xsd:element name="Cleanup" type="xsd:integer" />
    <xsd:element name="WaitingForTimestamp" type="xsd:integer" />
    <xsd:element name="WaitingForDocRehash" type="xsd:integer" />
    <xsd:element name="WaitingForTSPSig" type="xsd:integer" />
    <xsd:element name="Stored" type="xsd:integer" />
    <xsd:element name="Sealed" type="xsd:integer" />
  </xsd:sequence>

```

Reserved	integer: Anzahl reservierter AOIDs
Cleanup	integer: Anzahl der AOIDs, die beim nächsten Start des AOID-Cleanups gelöscht werden
WaitingForTimestamp	integer: Anzahl der AOIDs, die auf die Versiegelung warten
WaitingForDocsRehash	integer: Anzahl der AOIDs, die auf eine Erneuerung der Hash-Werte und Evidence Records warten
WaitingForTspSig	integer: Anzahl der AOIDs, die auf eine erneute Versiegelung mit einem neuen TSP warten
Stored	integer: Anzahl der archivierten Dokumente
Sealed	integer: Anzahl der versiegelten Dokumente

Datentyp TimerType

```

</xsd:complexType>
  <xsd:complexType name="TimerType">
    <xsd:sequence>
      <xsd:element name="Type" type="xsd:string"/>
      <xsd:element name="State" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

```

Type string:

Typ des Überwachungsprozesses

Mögliche Werte:

TSPTimer	Zeitgesteuerte Versiegelung
SDOCounting	Mengengesteuerte Versiegelung
AOIDCleanup	Zeitgesteuertes Löschen abgelaufener reservierter AOIDs
AOIDWithRefCleanup	Zeitgesteuertes Löschen abgelaufener reservierter reservierter AOIDs mit externen Referenz-IDs
ExternalFileCleanup	Zeitgesteuertes Löschen symbolischer Links, die im Rahmender Operation retrieveSDO von SecDocs für AOIDs mit externen Referenz-IDs erzeugt wurden

State string:

Zustand des Überwachungsprozesses

Mögliche Werte:

Started	Ereignisgesteuerte Überwachung ist aktiviert
Stopped	Ereignisgesteuerte Überwachung ist deaktiviert

Datentyp SDOCounterType

```

<xsd:complexType name="SDOCounterType">
  <xsd:sequence>
    <xsd:element name="TreeSize" type="xsd:integer"/>
    <xsd:element name="SDOsCounted" type="xsd:integer"/>
  </xsd:sequence>
</xsd:complexType>

```

TreeSize integer:

Schwellenwert für die Anzahl der SDOs. Wird dieser Schwellenwert erreicht, läuft die Versiegelung an.

SDOsCounted integer:

Anzahl der bisher gezählten SDOs

Datentyp RunningJobType

```
<xsd:complexType name="RunningJobType">
  <xsd:sequence>
    <xsd:element name="Description" type="xsd:string"/>
    <xsd:element name="Initiated" type="xsd:string"/>
    <xsd:element name="State" type="xsd:string"/>
    <xsd:element name="Type" type="xsd:string"/>
    <xsd:element name="Work" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

Description string:

Beschreibung des laufenden Vorgangs

Initiated string:

Startzeitpunkt in GMT des laufenden Jobs in der Form YYYY-MM-DDThh-mm-ss

State string:

Status des Auftrags

Mögliche Werte:

WAIT Die maximale Anzahl parallel laufender Aufträge ist bereits ausgeschöpft. Der Auftrag wird anlaufen sobald ein gerade laufender Auftrag endet.

RUN Der Auftrag läuft gerade.

Type string:

Typ des Auftrags

Mögliche Werte:

Generate_ER Normaler Versiegelungsauftrag nach der Archivierung

Renew_Doc_Hash Versiegelungsauftrag aufgrund der Erneuerung des Hash-Algorithmus

Renew_Tsp_Signature Auftrag zur Erneuerung der Zeitstempel (siehe Operation [renewTSPSignature](#) ab "[Operation renewTSPSignature](#)")

Work string:

Erklärung, welches Ereignis den Job angestoßen hat

Datentyp ScheduledJobType

```
<xsd:complexType name="ScheduledJobType">
  <xsd:sequence>
    <xsd:element name="Description" type="xsd:string"/>
    <xsd:element name="Interval" type="xsd:string"/>
    <xsd:element name="NextExecutionTime" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

Description

string:
Beschreibung, um welche Art von Auftrag es sich handelt.

Interval

string:
Gibt an, in welchen Zeitintervallen der Auftrag gestartet wird.

NextExecutionTime

string:
Gibt an, zu welchem Zeitpunkt der Auftrag als nächstes gestartet wird, in der Form YYYY-MM-DDThh-mm-ss.

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getArchiveInfo</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>MandantAdmin operation getArchiveInfo sample client</ns2:
auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"></GetRequest>
</S:Body>
```


Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getArchiveInfo</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>MandantAdmin operation getArchiveInfo sample client</sdsh:
auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <GetArchiveInfo
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Info xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
      <MandantName>Fujitsu</MandantName>
      <Mandant>
        <Name>Fujitsu</Name>
        <DisplayName>Fujitsu</DisplayName>
        <Contact>
          <FirstName>Zaphod</FirstName>
          <Surname>Beeblebrox</Surname>
          <Street>Last Lane 1</Street>
          <City>Milliways</City>
          <Zip>12345</Zip>
          <Country>Universe</Country>
          <Tel>123456789</Tel>
          <Email>Zaphod.Beeblebrox@universe.gov</Email>
        </Contact>
        <Path>archive/Fujitsu</Path>
        <TreeSize>10</TreeSize>
        <TreeAge>2</TreeAge>
        <TSP>TestTSP</TSP>
        <State>test</State>
        <Type>SDO</Type>
        <Properties />
      </Mandant>
      <TimeStampingInterval>2</TimeStampingInterval>
      <AOIDCleanupInterval>720</AOIDCleanupInterval>
      <AOIDInfo>
        <Reserved>0</Reserved>
        <Cleanup>0</Cleanup>
        <WaitingForTimestamp>0</WaitingForTimestamp>
        <WaitingForDocRehash>0</WaitingForDocRehash>
        <WaitingForTSPSig>0</WaitingForTSPSig>
        <Stored>0</Stored>
        <Sealed>0</Sealed>
      </AOIDInfo>
      <Timer>
        <Type>AOIDCleanup</Type>
        <State>Started</State>
      </Timer>
    </Info>
  </GetArchiveInfo>
</soap:Body>
```

```

        <Type>SDOCounting</Type>
        <State>Started</State>
    </Timer>
    <Timer>
        <Type>TSPTimer</Type>
        <State>Started</State>
    </Timer>
    <SDOCountingInfo>
        <TreeSize>10</TreeSize>
        <SDOsCounted>0</SDOsCounted>
    </SDOCountingInfo>
    <ScheduledJob>
        <Description>AOID Cleanup</Description>
        <Interval>12h</Interval>
        <NextExecutionTime>2020-12-01T20:03:19.36Z<
/NextExecutionTime>
    </ScheduledJob>
    <ScheduledJob>
        <Description>TimeStamping</Description>
        <Interval>2m</Interval>
        <NextExecutionTime>2020-12-01T16:43:19.34Z<
/NextExecutionTime>
    </ScheduledJob>
</Info>
</GetArchiveInfo>
</soap:Body>

```

5.2.3.28 Operation clearAOIDs

clearAOIDs löscht alle gespeicherten SDOs des Testmandanten vollständig aus SecDocs. Zu dem Zeitpunkt reservierte AOIDs bleiben erhalten.

RequestBody

Leeres Element ClearAOIDs

```
<ClearAOIDs></ClearAOIDs>
```

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

5.2.3.29 Operation convertTestMandant

convertTestMandant kann nur für Testmandanten ausgeführt werden. Alle derzeit für diesen TestMandanten archivierten SDOs werden in einen Zustand versetzt, als wären sie vom Produktivmandanten erzeugt worden. Siehe auch [Abschnitt „Vom Testmandanten zum produktiven Mandanten“](#)

RequestBody

Leeres Element ConvertTestMandant

```
<ConvertTestMandant></ConvertTestMandant>
```

Response-Body

Leeres Element result

```
<result></result>
```

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>convertTestMandant</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>
      MandantAdmin operation convertTestMandant sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <ConvertTestMandant
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData" />
</S:Body>
```

Response

```
<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>convertTestMandant</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>
      MandantAdmin operation convertTestMandant sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <result xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"></result>
</soap:Body>
```

5.2.3.30 Operation getArchivingOperations

getArchivingOperations gibt alle Funktionen aus, die der Web-Service, für den der Mandant angelegt wurde (ArchivingService bzw. S4), ausführen kann.

RequestBody

Leeres Element GetRequest

```
<GetRequest></GetRequest>
```

Response-Body

Element GetOperationsResponse *vom Datentyp* GetOperationsType

```
<xsd:complexType name="GetOperationsType">
  <xsd:sequence>
    <xsd:element name="Operation" type="xsd:string"
      maxOccurs="unbounded" minOccurs="0">
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
```

Operation

string

Name der Operation des Web-Service ArchivingService oder S4.

Beispiel

Request

```
<S:Header>
  <ns2:soapHeaderData
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData">
    <ns2:operation>getArchivingOperations</ns2:operation>
    <ns2:security>
      <ns2:principal>
        <ns2:role>MandantAdmin</ns2:role>
        <ns2:mandant>Fujitsu</ns2:mandant>
      </ns2:principal>
      <ns2:password>*****</ns2:password>
    </ns2:security>
    <ns2:auditID>
      MandantAdmin operation getArchivingOperations sample client
    </ns2:auditID>
  </ns2:soapHeaderData>
</S:Header>
<S:Body>
  <GetRequest
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData"
    xmlns:ns2="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
    xmlns:ns3="http://ts.fujitsu.com/secdocs/v3_1/adminUpdateData"></GetRequest>
</S:Body>
```

Response

```

<soap:Header>
  <sdsh:soapHeaderData>
    <sdsh:operation>getArchivingOperations</sdsh:operation>
    <sdsh:security>
      <sdsh:principal>
        <sdsh:role>MandantAdmin</sdsh:role>
        <sdsh:mandant>Fujitsu</sdsh:mandant>
      </sdsh:principal>
      <sdsh:password>*****</sdsh:password>
    </sdsh:security>
    <sdsh:auditID>
      MandantAdmin operation getArchivingOperations sample client
    </sdsh:auditID>
  </sdsh:soapHeaderData>
</soap:Header>
<soap:Body>
  <GetOperationsResponse
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <Operation>submitSDO</Operation>
    <Operation>getAOID</Operation>
    <Operation>retrieveSDO</Operation>
    <Operation>deleteSDO</Operation>
    <Operation>forceDeleteSDO</Operation>
    <Operation>metaDataSDO</Operation>
    <Operation>statusSDO</Operation>
    <Operation>retrieveMetaData</Operation>
    <Operation>requestForEvidence</Operation>
    <Operation>navigate</Operation>
    <Operation>getSchemaDataSDO</Operation>
    <Operation>moveSDO</Operation>
    <Operation>setExpirationDateTimeSDO</Operation>
    <Operation>forceDeferredSDO</Operation>
    <Operation>getAccountingData</Operation>
    <Operation>sparqlQuery</Operation>
    <Operation>simpleQuery</Operation>
    <Operation>listSDOVersions</Operation>
    <Operation>replaceSDO</Operation>
    <Operation>getVersion</Operation>
    <Operation>getAOIDWithRef</Operation>
  </GetOperationsResponse>
</soap:Body>

```

5.2.3.31 Operation `getAuditLogFileNames`

Die Operation `getAuditLogFileNames` des Web-Service `MandantAdminService` gibt eine Liste aller verfügbaren Audit-Log-Dateien des Mandanten aus.

Bei den verfügbaren Audit-Log-Dateien eines Mandanten handelt es sich um die Dateien `audit_hostname.log` und `audit_hostname.log.datum.suffix`.

Näheres zur Namensbildung der Audit-Log-Dateien finden Sie im [Abschnitt „Wechsel der Audit-Log-Dateien“ \(8.5.1.5 Wechsel der Audit-Log-Dateien\)](#).

Evtl. vorhandene Evidence Records der Audit-Log-Dateien werden nicht mit aufgelistet.

Die Operation `getAuditLogFileNames` können Sie nur mit der Rolle `SecDocs_MandantAuditor` ausführen.

Request-Body

Leeres Element `GetRequest`

```
<GetRequest></GetRequest>
```

Response-Body

Element `GetAuditLogFileNamesResponse` vom Datentyp

`GetAuditLogFileNamesResponseType`

```
<xsd:complexType name="GetAuditLogFileNamesResponseType">
  <xsd:sequence>
    <xsd:element name="AuditLogFileName" type="xsd:string"
      maxOccurs="unbounded" minOccurs="0">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

`AuditLogFileName`

string:

Dateiname einer mandantenspezifischen Audit-Log-Datei.

Beispiel

Response-Body

```
<soap:Body>
  <GetAuditLogFileNamesResponse
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <AuditLogFileName>audit_host1.log</AuditLogFileName>
    <AuditLogFileName>audit_host1.log.20201126.000</AuditLogFileName>
    <AuditLogFileName>audit_host1.log.20201130.000</AuditLogFileName>
    <AuditLogFileName>audit_host1.log.20210121.000</AuditLogFileName>
  </GetAuditLogFileNamesResponse>
</soap:Body>
```

5.2.3.32 Operation `getAuditLogFile`

Die Operation `getAuditLogFile` des Web-Service `MandantAdminService` gibt den Inhalt einer vorliegenden mandantenspezifischen Audit-Log-Datei aus.

Die verfügbaren Audit-Log-Dateien eines Mandanten können Sie mit der Operation `getAuditLogFileNames` (siehe "[Operation `getAuditLogFileNames`](#)") ermitteln.

Falls für die Audit-Log-Datei Evidence Records `AuditLogFileName.TSPName` existieren, wird deren Inhalt ebenfalls ausgegeben.

Die Operation `getAuditLogFile` können Sie nur mit der Rolle `SecDocs_MandantAuditor` ausführen.

Request-Body

Element `GetAuditLogFileRequest` vom Datentyp `GetAuditLogFileRequestType`

```
<xsd:complexType name="GetAuditLogFileRequestType">
  <xsd:sequence>
    <xsd:element name="AuditLogFileName" type="xsd:string"
      maxOccurs="1" minOccurs="1">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

`AuditLogFileName`

string:

Dateiname einer mandantenspezifischen Audit-Log-Datei.

Die Namen der verfügbaren Dateien werden von der Operation

`getAuditLogFileNames` jeweils in einem Element `AuditLogFileName` der Response geliefert.

Response-Body

Element `GetAuditLogFileResponse` vom Datentyp `GetAuditLogFileResponseType`

```
<xsd:complexType name="GetAuditLogFileResponseType">
  <xsd:sequence>
    <xsd:element name="AuditLogFileContent" maxOccurs="1"
      minOccurs="1">
      <xsd:simpleType>
        <xsd:restriction base="xsd:base64Binary">
          <xsd:minLength value="0"/></xsd:minLength>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="AuditLogER" type="AuditLogERType"
      minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

AuditLogFileContent

base64Binary:

Inhalt einer mandantenspezifischen Audit-Log-Datei.

Der ausgegebene String stellt die base64Binary-Codierung des in der angegebenen Audit-Log-Datei enthaltenen Textes dar. (Die Audit-Log-Datei ist eine Text-Datei in UTF-8-Codierung, siehe hierzu [Abschnitt „Die Audit-Log-Datei“ \(8.5.1.1 Die Audit-Log-Datei\)](#).)

AuditLogER

Information über einen Evidence Record einer mandantenspezifischen Audit-Log-Datei.

kein, ein oder mehrere der nachfolgend dargestellten Datentypen AuditLogERType.

Datentyp AuditLogERType

```
<xsd:complexType name="AuditLogERType">
  <xsd:sequence>
    <xsd:element name="TSPName" type="xsd:NCName"
      minOccurs="1" maxOccurs="1"/>
    <xsd:element name="AuditLogERContent"
      minOccurs="1" maxOccurs="1" >
      <xsd:simpleType>
        <xsd:restriction base="xsd:base64Binary">
          <xsd:minLength value="0"/></xsd:minLength>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

TSPName

NCName:

Name des verwendeten TSPs.

AuditLogERContent

base64Binary:

Inhalt des Evidence Records.

Der ausgegebene String stellt die base64Binary-Codierung des Inhalts des Evidence Records dar.

Der Inhalt eines Evidence Records ist ein String in ASN.1 Syntax, siehe hierzu RFC 4998.

Beispiel

Response-Body

```
<soap:Body>
  <GetAuditLogFileResponse
    xmlns="http://ts.fujitsu.com/secdocs/v3_1/adminData">
    <AuditLogFileContent>
      PDExMD4xIDIwMjEtMDItMDFUMDk6
      .....
      GaWxlIHNhbXBsZSBjbGllbnQiXQo=
    </AuditLogFileContent>
    <AuditLogER>
      <TSPName>TestTSP</TSPName>
      <AuditLogERContent>
        MIILqQIBATAPMA0GCWCGSAFlA
        .....
        LBgiSesTbBP7U1DxgwQG2g==
      </AuditLogERContent>
    </AuditLogER>
  </GetAuditLogFileResponse>
</soap:Body>
```

5.3 Tools

- Recovery (Skript `recoverFromStorage`)
- Daten gelöschter AOIDs entfernen (Skript `purgeData`)

5.3.1 Recovery (Skript `recoverFromStorage`)

Mit Hilfe des Skripts `recoverFromStorage` können Sie das Archiv auf Konsistenz prüfen. Das sind im Einzelnen folgende Prüfungen:

- Konsistenz der Daten auf dem Storage-System
Prüfung, ob sich, z.B. aufgrund von (fehlerhaft) abgebrochenen Aktionen, übrig gebliebene Dateien oder Verzeichnisse auf dem Storage-System befinden. Erkannte Fragmente werden gelöscht, sofern die WORM-Funktion dies nicht verhindert.
- Konsistenz von Storage-System und Datenbank
Maßgebend sind die Archivstrukturen und archivierten SDOs auf dem Storage-System. In der Datenbank wird (für eine performante Ausführung der Archivierungsoperationen) redundante Information geführt:
 - Mandanten/Organisationsstruktur
 - registrierte SDO-Typen, Rollen-Definitionen und Privilegien-Definitionen, ...
 - recherchierbare Metadaten im TripleStore

i Eine Anbindung an einen Triple-Store wird in der aktuellen Version nicht unterstützt.

- Ablageort, sowie bestimmte Metadaten und Stati der archivierten SDOs

Es erfolgt eine Prüfung, ob die Information in der Datenbank – relativ zu den Daten auf dem Storage-System – vollständig ist. Fehlende Informationen werden in die Datenbank eingetragen.

i Für Testmandanten erfolgt kein Nachtrag der AOIDs (und der damit verbundenen Metadaten, Status-Angaben und TripleStore-Einträge) vom Storage-System in die Datenbank. Die Testdaten auf dem Storage-System werden nicht gelöscht. Es wird empfohlen, das Löschen der Testdaten manuell durchzuführen, um bei einer späteren Verwendung des Testmandanten als Produktivmandant (Operation `convertTestMandant` bzw. `createMandant` für einen bereits gelöschten Testmandanten) diese Testdaten nicht dauerhaft zu archivieren.

Der Aufbau des TripleStore erfolgt mittels `recoverFromStorage` mit der Operation `update`.

Die Durchführung einer Recovery über ein gesamtes SecDocs-Archiv bzw. für die gesamte Datenbank kann unter Umständen längere Zeit in Anspruch nehmen. Zur performanteren Bearbeitung wird daher die folgende Funktionalität angeboten:

- Auswahl eines Teilbereichs des Archivs für die Recovery
Die Aufrufoptionen `-mand`, `-org` und `-subPath` ermöglichen es, die Recovery nur für einen Teilbereich des Archivs durchzuführen. Auf diese Weise können Sie die Recovery eines großen Archivs in mehrere Abschnitte aufteilen.

- Beschränkung der Recovery auf den TripleStore

Mit der Property `onlyTripleStore` in der mit `-properties` angegebenen Datei und/oder der Aufrufoption `-onlyTripleStore` kann die Recovery auf den TripleStore beschränkt werden.

Wenn die Property auf `true` gesetzt oder die Aufrufoption angegeben ist, führt das Recovery-Tool die jeweilige Operation nur noch für den TripleStore aus. Andere Datenbankeinträge werden nicht bearbeitet.

Mit dieser Funktionalität kann z.B. der folgende Fall behandelt werden:

Gezielte Nachindizierung von Archivbeständen, in denen Dokumente mit dem Systemschlüssel `$Index` enthalten sind, und die mit ausgeschaltetem TripleStore in ein Sec-Docs-Archiv aufgenommen wurden. Hierzu muss aus den vorhandenen Variablen in `$Index` ein TripleStore neu aufgebaut werden.

Aufruf des Skripts `recoverFromStorage`

Das Skript `recoverFromStorage` wird folgendermaßen aufgerufen:

```
recoverFromStorage -operation scan | check | restore | update
                   -properties <name-of-property-file>
                   [-repair]
                   [-onlyTripleStore]
                   [-mand <name-of-tenant>]
                   [-org <name-of-organisation>]
                   [-subPath <name-of-subpath>]
```


-operation

- | | |
|---------|--|
| scan | Es erfolgt nur die Konsistenzprüfung der Daten auf dem Storage-System (ein Abgleich mit der Datenbank findet nicht statt). Gefundene Inkonsistenzen werden nach <code>stdout</code> protokolliert. Wenn die Aufrufoption <code>-repair</code> angegeben ist oder in der mit <code>-properties</code> angegebenen Datei <code>repair=true</code> gesetzt ist, werden erkannte Inkonsistenzen auf dem Storage-System behoben. |
| check | Es erfolgt eine Konsistenzprüfung der Daten auf dem Storage-System (wie bei <code>scan</code>). Zusätzlich werden die Storage-Daten mit der Datenbank abgeglichen. Gefundene Inkonsistenzen werden nach <code>stdout</code> protokolliert. Wenn die Aufrufoption <code>-repair</code> angegeben ist oder in der mit <code>-properties</code> angegebenen Datei <code>repair=true</code> gesetzt ist, werden erkannte Inkonsistenzen auf dem Storage-System behoben. Es erfolgt keine Änderung in der Datenbank, d.h. die als fehlend erkannten Einträge in der Datenbank werden nicht nachgetragen. |
| restore | Mit dieser Operation wird die SecDocs Datenbank neu aufgebaut (allein beruhend auf den Informationen auf dem Storage-System). Besteht die Datenbank und ist sie nicht leer, wird <code>recoverFromStorage</code> mit Fehler beendet. |

i SecDocs darf während dieser Aktion nicht aktiv sein.

- | | |
|--------|---|
| update | Mit dieser Operation wird für eine bestehende Datenbank geprüft, ob auf dem Storage-System weitere Informationen für die Datenbank verfügbar sind, die in die Datenbank eingepflegt werden müssen. Wenn ja, werden diese in die Datenbank eingepflegt. Implizit wird auch die Storage-Konsistenz geprüft (wie bei <code>scan</code>). Nach einer Storage-Recovery mit der Operation <code>update</code> wird empfohlen, alle Testmandanten zu löschen, wenn diese für eine spätere Umwandlung in produktive Mandanten vorgesehen sind. |
|--------|---|

i Die Operation `update` darf nicht gestartet werden, wenn gerade eine der Operationen `renewHashAlgorithm` oder `renewTSPSignature` bearbeitet wird.

<code>-properties</code>	<p>In der mit <code><name-of-property-file></code> angegebenen Datei werden dem Recovery- Tool für den Ablauf notwendige Informationen bereitgestellt (z.B. Link auf das Archiv). Als Template finden Sie die Datei <code>recover.properties</code> im Unterverzeichnis <code>admin/recovery</code> des Installationsverzeichnisses.</p>
<code>-repair</code>	<p>Bei den Operationen <code>scan</code> und <code>check</code> werden erkannte Inkonsistenzen auf dem Storage-System behoben. Solche Inkonsistenzen können z.B. in Form von unbenutzten Dateien oder Verzeichnissen nach einem Systemabsturz o.ä. auftreten. Bei den Operationen <code>restore</code> und <code>update</code> ist es nicht erforderlich, noch zusätzlich <code>-repair</code> anzugeben.</p> <p>Diese Option hat dieselbe Wirkung wie die Angabe <code>repair=true</code> in der mit der Option <code>-properties</code> angegebenen Datei. Ist die Option angegeben, hat sie Vorrang vor der Angabe in der <code>properties</code>-Datei.</p>
<code>-onlyTripleStore</code>	<div>  Eine Anbindung an einen Triple-Store wird in der aktuellen Version nicht unterstützt </div> <p>Die Operation wird nur für den TripleStore durchgeführt. Andere Einträge in der SecDocs-Datenbank werden nicht bearbeitet.</p> <p>Bei den Operationen <code>scan</code> und <code>restore</code> wird die Angabe von <code>-onlyTripleStore</code> ignoriert.</p> <p>Diese Option hat dieselbe Wirkung wie die Angabe <code>onlyTripleStore=true</code> in der mit der Option <code>-properties</code> angegebenen Datei. Ist die Option angegeben, hat sie Vorrang vor der Angabe in der <code>properties</code>-Datei.</p> <p>Aufrufbeispiel:</p> <pre>recoverFromStorage -operation update -properties myPropertyFile -onlyTripleStore</pre> <p>Der TripleStore wird für alle Mandanten neu aufgebaut, z.B. wenn Dokumente mit dem Systemschlüssel <code>\$Index</code> bei ausgeschaltetem TripleStore archiviert wurden.</p>
<code>-mand</code>	<p>Die Operation wird für einen mandantenspezifischen Teilbereich des Archivs durchgeführt.</p> <p>Aufrufbeispiel:</p> <pre>recoverFromStorage -operation check -properties myPropertyFile -mand Mandant1</pre>

-org

Die Operation wird für einen organisationsspezifischen Teilbereich des Archivs durchgeführt.

Die Option `-org` wird nur ausgewertet, wenn die Option `-mand` angegeben ist. Andernfalls wird der Aufruf des Recovery-Tools abgewiesen.

Aufrufbeispiel:

```
recoverFromStorage -operation check
                  -properties myPropertyFile
                  -mand Mandant1
                  -org Org1
```

-subPath

Die Operation wird für ein mandanten- und organisationsspezifisches Unterverzeichnis durchgeführt.

Der angegebene Pfad ist relativ zum Pfad der angegebenen Organisation im Verzeichnis des angegebenen Mandanten.

Die Option `-subPath` wird nur ausgewertet, wenn gleichzeitig die Optionen `-mand` und `-org` angegeben sind. Andernfalls wird der Aufruf des Recovery-Tools abgewiesen.

Aufrufbeispiel:

```
recoverFromStorage -operation check
                  -properties myPropertyFile
                  -mand Mandant1 -org Org1
                  -subPath SDOsample/2015/03/08
```

Die Operation `check` wird nur für das Unterverzeichnis `SDOsample/2015/03/08` durchgeführt, welches sich im mandantenspezifischen Verzeichnis von Mandant1 und dort im organisationsspezifischen Unterverzeichnis von Org1 befindet.



Beachten Sie: Ein paralleler Ablauf mehrerer Recovery-Vorgänge darf nicht erfolgen, d.h. Sie dürfen das Recovery-Tool erst dann ein weiteres Mal starten, wenn ein vorangegangener Lauf beendet ist. Insbesondere darf das Recovery-Tool auch nicht gleichzeitig auf mehreren SecDocs-Instanzen im Verbund ablaufen.

Beispiel für ein Ausgabeprotokoll

i Der Ablauf von `recoverFromStorage` wird auf STDOUT ausgegeben, Fehler werden auf STDERR ausgegeben. Zusätzlich existiert eine Loggingdatei `recoverFromStorage.log` im Unterverzeichnis `admin/recovery` des Installationsverzeichnisses.

In den folgenden Beispielausgaben wird aus Platzgründen das Datum am Anfang der Zeile unterdrückt.

Zu Beginn des Protokolls werden die Aufrufparameter protokolliert:

```
13:04:08,943: INFO - SecDocs Recovery Tool Version : SecDocs 3.0.1.0 build-1 qabuild1
[14]
13:04:08,944: INFO - Revision      : Build March 10 2015 (*)
                  $LastChangedRevision: 9011 $ - $LastChangedDate: 2015-06-12 12:21:38 +0200
(Fri, 12 Jun 2015) $
13:04:08,951: INFO - *****
13:04:08,951: INFO - *      Storage <-> DB : CHECK      *
13:04:08,952: INFO - *****
13:04:08,952: INFO - maxParThreads    : 2
13:04:08,952: INFO - testMandants     : false
13:04:08,952: INFO - standAlone       : false
13:04:08,952: INFO - packetSize       : 10000
13:04:08,953: INFO - repair           : true
13:04:08,953: INFO - asPath           : /home/secdocs/jboss/wildfly
```

Dann wird je Mandant/Organisation die Recovery angestoßen:

```
13:04:08,953 main SecDocs.Storage: INFO - ===== Start Processing 13:04:08.953
=====
...
```

Je Mandant/Organisation wird die Anzahl der gefundenen AOIDs festgehalten:

```
13:04:31.737 INFO : =====
13:04:31.737 INFO : AOID Recovery : Jane3/org1 at org1/_ORG
13:04:31.993 INFO : AOID(s) to process from storage : 12
13:04:32.337 INFO : AOID(s) wait for timestamping : 2
13:04:32.337 INFO : AOID(s) sealed : 4
13:04:32.337 INFO : AOID(s) in delete state : 6
13:04:32.337 INFO : Searching time per AOID : 14,3 msec
13:04:32.337 INFO : Processing time per AOID : 28,6 msec
13:04:32.337 INFO : AOID Recovery : Jane3 / org1 elapsed time 0,5 sec
13:04:32.337 INFO : =====
...
```

Zum Schluss wird noch eine Zusammenfassung über den gesamten Lauf gegeben:

```
13:04:32,345: INFO - =====
13:04:32,345: INFO - ===== Start Processing 13:04:08.953 =====
13:04:32,346: INFO - *****
13:04:32,346: INFO - *      Storage <-> DB : CHECK      *
13:04:32,346: INFO - *****
13:04:32,346: INFO - Elapsed time 23,4 sec
13:04:32,346: INFO -          DOCUMENTS :                6
13:04:32,346: INFO -          DELETED   :                6
13:04:32,346: INFO -          CHECKS    :                12
13:04:32,346: INFO -
13:04:32,346: INFO - ===== End    Processing 13:04:32.346 =====
```

5.3.2 Daten gelöschter AOIDs entfernen (Skript `purgeData`)

Beim Löschen einer AOID mit `deleteSDO` oder `forceDeleteSDO` werden nicht alle Informationen gelöscht, die diese AOID betreffen. Vielmehr wird die AOID nur als gelöscht markiert, während einige Metadaten zu dieser AOID erhalten bleiben. Dadurch nimmt die Anzahl benutzter Knoten ständig zu.

SecDocs bietet daher ab der Version 3.0 mit dem Skript `purgeData` die Möglichkeit, die Überbleibsel gelöschter AOIDs vollständig zu entfernen. Es werden alle Verzeichnisse und Dateien gelöscht, die zu diesen AOIDs gehören, und die Einträge für diese AOIDs aus der SecDocs-Datenbank entfernt.

Mit dem Skript `purgeData` können Sie folgende Aktionen ausführen:

- Ermitteln der Anzahl der in der SecDocs-Datenbank als gelöscht markierten AOIDs.
- Entfernen aller Daten, die zu AOIDs gehören, die als gelöscht markiert sind. Das bedeutet:
 - Löschen des zugehörigen Verzeichnisses und der Dateien aus dem Archiv
 - Löschen der AOID aus der SecDocs-Datenbank.

Die Menge der AOIDs, für die diese Aktionen ausgeführt werden sollen, können Sie über verschiedene Filter-Kriterien einschränken, z.B. auf:

- eine Liste explizit angegebener AOIDs eines bestimmten Mandanten
- alle als gelöscht markierten AOIDs eines bestimmten Mandanten oder einer bestimmten Organisation eines Mandanten oder auf einem bestimmten Ablageort für eine Organisation eines Mandanten

Voraussetzungen

- Das Skript `purgeData` muss unter der Linux-Benutzerkennung `secdocs` ausgeführt werden.
- Das Skript darf nicht mehrfach parallel ablaufen. Sie dürfen es erst dann ein weiteres Mal starten, wenn ein vorangegangener Lauf beendet ist.
Dies gilt auch, wenn die Skripts für unterschiedliche Mandanten oder Ablageorte ausgeführt werden. Insbesondere darf das Skript auch nicht gleichzeitig auf mehreren SecDocs-Instanzen im Verbund ablaufen.



- Das Skript `purgeData` verarbeitet auch AOIDs, die erst nach dem Start des Skripts mit `deleteSDO`/ `forceDeleteSDO` gelöscht werden.

Leere Verzeichnisse, die dadurch entstehen, dass die letzte darin enthaltene AOID gelöscht wurde, werden vom Skript `purgeData` nicht gelöscht.



Die Wiederherstellung der SecDocs-Datenbank aus einer Sicherungsversion kann zu Inkonsistenzen zwischen Datenbankinhalt und Storage-System führen, wenn in der Zwischenzeit die Operationen `deleteSDO/forceDeleteSDO` und das Skript `purgeData` ausgeführt wurden.

In diesem Fall lässt sich SecDocs nicht mehr starten und es wird eine entsprechende Fehlermeldung in die Protokolldatei `server.log` der WildFly SecDocs-Server-Instanz ausgegeben.

Dieses Problem können Sie auf folgende Weise beheben:

- Führen Sie das Skript `recoverFromStorage` mit der Operation `check` aus, um eine Konsistenzprüfung durchzuführen.
- Führen Sie das Skript `recoverFromStorage` mit der Operation `update` aus, um einen konsistenten Zustand herzustellen.
- Starten Sie SecDocs

Aufruf des Skripts `purgeData`

Das Skript `purgeData` wird folgendermaßen aufgerufen:

```
purgeData [-mode CHECK | PURGE]
          [-asPath <WildFly-home-directory>]
          [-mandant <name-of-tenant>]
          [-org <name-of-organisation>]
          [-aoid <aoid>|<aoid-list>]
          [-aoid-list-file <file>]
          [-path <name-of-SDO-path>]
          [-subpath-filter y|n]
          [-threads <integer-value>]
          [-threshold <integer-value>]
          [-h (--help)]
```

`-mode` Aktion, die ausgeführt werden soll: `CHECK` oder `PURGE`.

Standardwert: `CHECK`

Beide Aktionen erstellen ein Protokoll auf `stdout` und in der Datei `purgeData.log` im Unterverzeichnis `admin/purge` des Installationsverzeichnis. Fehlermeldungen werden nach `stderr` ausgegeben.

`CHECK` Gibt die Anzahl der als gelöscht markierten AOIDs aus, die den angegebenen Filterkriterien entsprechen.

Die Zählung wird in der SecDocs Datenbank durchgeführt.

`PURGE` Löscht alle Daten der AOIDs, die als gelöscht markiert sind und den angegebenen Filter-Kriterien entsprechen, sowohl aus dem Dateisystem als auch aus der SecDocs-Datenbank.

`-` Home-Verzeichnis der WildFly SecDocs-Server-Instanz.

`asPath` Standardwert: `installation-dir/jboss/wildfly`

`-mandant` Name des Mandanten, für dessen AOIDs die angegebene Aktion ausgeführt werden soll.

Falls `-mandant` nicht angegeben ist, wird die Aktion für die AOIDs aller Mandanten ausgeführt.

<code>-org</code>	<p>Name der Organisation, für deren AOIDs die angegebene Aktion ausgeführt werden soll.</p> <p>Wenn <code>-org</code> angegeben ist, muss auch <code>-mandant</code> angegeben sein.</p> <p>Falls <code>-org</code> nicht angegeben ist, wird die Aktion für die AOIDs aller Organisationen des/der angegebenen Mandanten ausgeführt.</p> <p>Die Angabe <code>-org</code> wird ignoriert, falls <code>-aoid</code> oder <code>-aoid-list-file</code> angegeben ist.</p>
<code>-aoid</code>	<p>AOID oder Liste von AOIDs, für die die angegebene Aktion ausgeführt werden soll. In einer Liste müssen die AOIDs durch Leerzeichen getrennt angegeben werden.</p> <p>Für die Angabe einer längeren Liste von AOIDs wird die Option <code>-aoid-list-file</code> empfohlen.</p> <p>Wenn <code>-aoid</code> angegeben ist, muss auch <code>-mandant</code> angegeben sein und es darf weder <code>-aoid-list-file</code> noch <code>-path</code> angegeben sein.</p> <p>Falls keine der Optionen <code>-aoid</code>, <code>-aoid-list-file</code> oder <code>-path</code> angegeben ist, wird die Aktion für alle AOIDs ausgeführt, die den Filterkriterien <code>-mandant</code> und ggf. <code>-org</code> entsprechen.</p> <p>Für AOIDs, die hier angegeben, aber nicht als gelöscht markiert oder nicht dem/den angegebenen Mandanten zugeordnet sind, wird die Aktion nicht ausgeführt. Stattdessen werden entsprechende Fehlermeldungen ausgegeben.</p>
<code>-aoid-list-file</code>	<p>Name einer Textdatei, die eine Liste von AOIDs enthält, für die die angegebene Aktion ausgeführt werden soll. In dieser Datei muss jede AOID in einer eigenen Zeile stehen, z.B.:</p> <div data-bbox="329 1014 855 1136" data-label="Text"> <pre>f3becfa9-9ead-4612-8f8f-3e1b181765bb e853701f-dbf8-47db-8d5b-2253e052312d 460fc3f4-ac3d-4703-92f3-fc37a78f2eec f1ba2a4e-f876-4096-96af-e481546d566b</pre> </div> <p>Wenn <code>-aoid-list-file</code> angegeben ist, muss auch <code>-mandant</code> angegeben sein und es darf weder <code>-aoid</code> noch <code>-path</code> angegeben sein.</p> <p>Falls keine der Optionen <code>-aoid</code>, <code>-aoid-list-file</code> oder <code>-path</code> angegeben ist, wird die Aktion für alle AOIDs ausgeführt, die den Filterkriterien <code>-mandant</code> und ggf. <code>-org</code> entsprechen.</p> <p>Für AOIDs, die in der Datei angegeben, aber nicht als gelöscht markiert oder nicht dem/den angegebenen Mandanten zugeordnet sind, wird die Aktion nicht ausgeführt. Stattdessen werden entsprechende Fehlermeldungen ausgegeben.</p>

`-path`

Ablageort, an dem die Daten von AOIDs bereinigt werden sollen.

Der Pfadname muss mit einem führenden "/" angegeben werden.

Der angegebene Pfad ist relativ zum Pfad der angegebenen Organisation im Verzeichnis des angegebenen Mandanten. Das bedeutet: Das Wurzelverzeichnis ("/") bezeichnet das oberste Verzeichnis, das zu der angegebenen Kombination aus Organisation und Mandant gehört.

i Unterverzeichnisse des angegebenen Pfades werden nur bearbeitet, wenn für die Option `-subpath-filter` der Wert "y" angegeben ist.

Wenn `-path` angegeben ist, müssen auch `-mandant` und `-org` angegeben sein und es darf weder `-aoid` noch `-aoid-list-file` angegeben sein.

Falls keine der Optionen `-aoid`, `-aoid-list-file` oder `-path` angegeben ist, wird die Aktion für alle AOIDs ausgeführt, die den Filterkriterien `-mandant` und ggf. `-org` entsprechen.

`-subpath-filter`

Gibt an, ob auch die Unterverzeichnisse des angegebenen Ablageorts (`-path`) bereinigt werden sollen.

Mögliche Werte:

- y Alle Unterverzeichnisse des mit der Option `-path` angegebenen Verzeichnisses (und dieses Verzeichnis selbst) werden bereinigt.
- n Nur das mit der Option `-path` angegebene Verzeichnis wird bereinigt, die Unterverzeichnisse werden nicht berücksichtigt.

Standardwert: n

Wenn `-subpath-filter` angegeben ist, muss auch die Option `-path` angegeben werden.

`-threads`

Maximale Anzahl von Threads für die Ausführung der angegebenen Aktion.

Standardwert: 4

i Berücksichtigen Sie bei der Angabe von höheren Werten für `-threads` oder `-threshold` die Größe des verfügbaren Speichers.

`-threshold`

Maximale Anzahl von AOIDs, die gleichzeitig in einem Thread verarbeitet werden sollen.

Standardwert: 10000.

i Berücksichtigen Sie bei der Angabe von höheren Werten für `-threads` oder `-threshold` die Größe des verfügbaren Speichers.

`-h (--help)`

Gibt eine kurze Beschreibung der verfügbaren Optionen des Skripts `purgeData` aus.

Beispiele für Ausgabeprotokolle

i In den folgenden Beispielausgaben wird aus Platzgründen das Datum am Anfang der Zeile unterdrückt.

Ein PURGE-Auftrag für alle Mandanten wird gestartet, es gibt jedoch keine gelöschten AOIDs .

```
purgeData -mode PURGE
13:36:27,193: INFO -
*****
13:36:27,194: INFO - *
13:36:27,194: INFO - *
13:36:27,197: INFO - *
PurgeData Tool
build: SecDocs 3.0.1.0 build-1 qabuild1 [PurgeData
build23]
13:36:27,197: INFO - *
13:36:27,197: INFO -
*****
13:36:27,732: INFO - reading properties
13:36:28,752: INFO - Starting parameters:
13:36:28,752: INFO - Server path ----- /home/secdocs/secdocs30/jboss
/wildfly
13:36:28,752: INFO - Run mode ----- PURGE
13:36:28,752: INFO - Max parallel threads ----- 4
13:36:28,752: INFO - Max Aoids per thread ----- 10000
13:36:28,752: INFO - Mandant -----
13:36:28,752: INFO - Organization -----
13:36:28,752: INFO - AOIDs -----
13:36:28,752: INFO - SDO PATH -----
13:36:28,752: INFO - Subpath filtering enabled ----- false
13:36:28,755: INFO - Counted aoids for mandant: Mandant1 0 AOIDs
13:36:28,759: INFO - Counted aoids for mandant: JPurgeData01 0 AOIDs
13:36:28,759: INFO - Counted aoids for mandant: JPurgeData02 0 AOIDs
13:36:28,759: INFO - Counted aoids for mandant: JPurgeData03 0 AOIDs
13:36:28,760: INFO - Counted aoids for mandant: JPurgeData04 0 AOIDs
13:36:28,760: INFO - Counted aoids for all mandant: 0
13:36:28,760: INFO - AOIDs to process: 0
13:36:28,760: INFO - Nothing to do !!!
```

Ein PURGE-Auftrag für gelöschte AOIDs an einem bestimmten Ablageort und in allen Unterverzeichnissen für eine bestimmte Organisation eines Mandanten soll ausgeführt werden.

Zunächst wird eine Prüfung durchgeführt:

```

        purgeData -mode CHECK -mandant JPurgeData02 -org org1
                -path /forPurgeData -subpath-filter y
13:36:30,183: INFO -
*****
13:36:30,184: INFO - *
13:36:30,184: INFO - *
13:36:30,187: INFO - *
13:36:30,187: INFO - *
13:36:30,187: INFO - *
13:36:30,187: INFO - *
13:36:30,187: INFO - *
*****
13:36:30,728: INFO - reading properties
13:36:31,666: INFO - Starting parameters:
13:36:31,666: INFO - Server path ----- /home/secdocs/secdocs30/jboss
/wildfly
13:36:31,666: INFO - Run mode ----- CHECK
13:36:31,666: INFO - Max parallel threads ----- 4
13:36:31,666: INFO - Max Aoids per thread ----- 10000
13:36:31,666: INFO - Mandant ----- JPurgeData02
13:36:31,666: INFO - Organization ----- org1
13:36:31,667: INFO - AOIDS -----
13:36:31,667: INFO - SDO PATH ----- /forPurgeData
13:36:31,667: INFO - Subpath filtering enabled ----- true
13:36:31,673: INFO - Counted aoids for mandant: JPurgeData02 5 AOIDS
13:36:31,675: INFO - AOIDS to process: 5
13:36:31,675: INFO - CHECK mode end.

```

Anschließend wird die Bereinigung durchgeführt:

```

        purgeData -mode PURGE -mandant JPurgeData02 -org org1
                -path /forPurgeData -subpath-filter y
13:38:12,357: INFO -
*****
13:38:12,358: INFO - *
13:38:12,358: INFO - *
13:38:12,361: INFO - *
                        PurgeData Tool
                        build: SecDocs 3.0.1.0 build-1 qabuild1 [PurgeData
build23]
13:38:12,361: INFO - *
13:38:12,361: INFO -
*****
13:38:12,906: INFO - reading properties
13:38:13,839: INFO - Starting parameters:
13:38:13,839: INFO -   Server path ----- /home/secdocs/secdocs30/jboss
/wildfly
13:38:13,839: INFO -   Run mode ----- PURGE
13:38:13,839: INFO -   Max parallel threads ----- 4
13:38:13,839: INFO -   Max Aoids per thread ----- 10000
13:38:13,839: INFO -   Mandant ----- JPurgeData02
13:38:13,839: INFO -   Organization ----- org1
13:38:13,839: INFO -   AOIDS -----
13:38:13,839: INFO -   SDO PATH ----- /forPurgeData
13:38:13,839: INFO -   Subpath filtering enabled ----- true
13:38:13,841: INFO - AOIDS to process: 5
13:38:13,841: INFO - Entering mode purge.
13:38:13,841: INFO - Collecting data.
13:38:13,877: INFO - Process: 5 AOIDS in one thread
13:38:13,877: INFO - Thread: ForkJoinPool-1-worker-1 processing:5
13:38:13,942: INFO - Thread: ForkJoinPool-1-worker-1 has 0 aoids to go.
13:38:13,943: INFO - AOIDS processed: 5
13:38:13,944: INFO - Elapsed time: 103 ms

```

Ein PURGE-Auftrag soll für gelöschte AOIDS einer bestimmten Organisation eines Mandanten ausgeführt werden. Die Liste der AOIDS steht in einer Datei.

```

    purgeData -mode PURGE -mandant JPurgeData02 -org org1
              -aoid-list-file funktionTests.sh.aoidlistfile.txt
13:40:54,433: INFO -
*****
13:40:54,434: INFO - *
13:40:54,434: INFO - *
13:40:54,437: INFO - *
                      PurgeData Tool
                      build: SecDocs 3.0.1.0 build-1 qabuild1 [PurgeData
build23]
13:40:54,437: INFO - *
13:40:54,437: INFO -
*****
13:40:54,998: INFO - reading properties
13:40:57,422: INFO - Starting parameters:
13:40:57,423: INFO -   Server path ----- /home/secdocs/secdocs30/jboss
/wildfly
13:40:57,423: INFO -   Run mode ----- PURGE
13:40:57,423: INFO -   Max parallel threads ----- 4
13:40:57,423: INFO -   Max Aoids per thread ----- 10000
13:40:57,423: INFO -   Mandant ----- JPurgeData02
13:40:57,423: INFO -   Organization ----- org1
13:40:57,423: INFO -   AOIDS ----- (see funktionTests.sh.
aoidlistfile.txt)
13:40:57,423: INFO -   SDO PATH -----
13:40:57,423: INFO -   Subpath filtering enabled ----- false
13:40:57,425: INFO -   AOIDS to process: 2500
13:40:57,425: INFO -   Entering mode purge.
13:40:57,425: INFO -   Collecting data.
13:40:57,472: INFO -   Process: 2500 AOIDS in one thread
13:40:57,472: INFO -   Thread: ForkJoinPool-1-worker-1 processing:2500
13:41:02,488: INFO -   Thread: ForkJoinPool-1-worker-1 has 2000 aoids to go.
13:41:09,622: INFO -   Thread: ForkJoinPool-1-worker-1 has 1000 aoids to go.
13:41:17,542: INFO -   Thread: ForkJoinPool-1-worker-1 has 0 aoids to go.
13:41:17,544: INFO -   AOIDS processed: 2500
13:41:17,544: INFO -   Elapsed time: 20119 ms

```

Ein PURGE-Auftrag für alle Mandanten wird gestartet, und es gibt eine hohe Anzahl gelöschter AOIDS .

```

    purgeData -mode PURGE
16:49:30,042: INFO -
*****
16:49:30,043: INFO - *
16:49:30,043: INFO - *
16:49:30,046: INFO - *
                        PurgeData Tool
                        build: SecDocs 3.0.1.0 build-1 qabuild1 [PurgeData
build23]
16:49:30,046: INFO - *
16:49:30,047: INFO -
*****
16:49:31,058: INFO - reading properties
16:49:31,060: INFO - globalProperties read
16:49:31,060: INFO - reading properties
16:49:37,290: INFO - Starting parameters:
16:49:37,291: INFO -   Server path ----- /home/secdocs/secdocs30/jboss
/wildfly
16:49:37,291: INFO -   Run mode ----- PURGE
16:49:37,291: INFO -   Max parallel threads ----- 30
16:49:37,291: INFO -   Max Aoids per thread ----- 10000
16:49:37,291: INFO -   Mandant -----
16:49:37,291: INFO -   Organization -----
16:49:37,291: INFO -   AOIDS -----
16:49:37,291: INFO -   SDO PATH -----
16:49:37,291: INFO -   Subpath filtering enabled ----- false
16:52:14,234: INFO - Counted aoids for mandant: Mandant1 15404721 AOIDS
16:52:14,314: INFO - Counted aoids for all mandant: 15404721
16:52:14,314: INFO - AOIDS to process: 15404721
16:52:14,314: INFO - Entering mode purge.
16:52:14,314: INFO - Collecting data.

```

```

16:52:59,219: INFO - Dispatching: 300000 AOIDS
16:52:59,257: INFO - Dispatching: 290000 AOIDS
16:52:59,285: INFO - Dispatching: 280000 AOIDS
..
..
16:53:00,233: INFO - Dispatching: 30000 AOIDS
16:53:00,233: INFO - Dispatching: 20000 AOIDS
16:53:00,282: INFO - Process: 10000 AOIDS in one thread
16:53:00,282: INFO - Thread: ForkJoinPool-1-worker-18 processing:10000
16:53:00,371: INFO - Process: 10000 AOIDS in one thread
16:53:00,371: INFO - Thread: ForkJoinPool-1-worker-19 processing:10000
16:53:00,446: INFO - Process: 10000 AOIDS in one thread
..
..
16:53:02,770: INFO - Thread: ForkJoinPool-1-worker-10 processing:10000
16:53:02,857: INFO - Process: 10000 AOIDS in one thread
16:53:02,858: INFO - Thread: ForkJoinPool-1-worker-3 processing:10000
16:53:02,903: INFO - Process: 10000 AOIDS in one thread
16:53:02,903: INFO - Thread: ForkJoinPool-1-worker-2 processing:10000
16:54:13,320: INFO - Thread: ForkJoinPool-1-worker-18 has 9000 aoids to go.
16:54:14,253: INFO - Thread: ForkJoinPool-1-worker-29 has 9000 aoids to go.
16:54:14,877: INFO - Thread: ForkJoinPool-1-worker-22 has 9000 aoids to go.
..
..
17:05:08,491: INFO - Thread: ForkJoinPool-1-worker-23 has 0 aoids to go.
17:05:12,651: INFO - Thread: ForkJoinPool-1-worker-25 has 0 aoids to go.
17:05:12,654: INFO - AOIDS processed: 300000
17:07:54,981: INFO - Counted aoids for mandant: johann1 15104721 AOIDS
17:07:55,119: INFO - Counted aoids for all mandant: 15104721
17:09:50,284: INFO - Dispatching: 300000 AOIDS
17:09:50,286: INFO - Dispatching: 290000 AOIDS
17:09:50,288: INFO - Dispatching: 280000 AOIDS
..
..
2015-09-12 10:42:21,629: INFO - Thread: ForkJoinPool-1-worker-14 has 0 aoids to go.
2015-09-12 10:42:24,278: INFO - Thread: ForkJoinPool-1-worker-7 has 0 aoids to go.
2015-09-12 10:42:24,280: INFO - AOIDS processed: 104721
2015-09-12 10:46:02,366: INFO - Counted aoids for mandant: Mandant1 0 AOIDS
2015-09-12 10:46:02,466: INFO - Counted aoids for all mandant: 0
2015-09-12 10:46:02,466: INFO - Elapsed time: 64428152 ms

```

6 Externe Datenobjekte

Standardmäßig werden Dateien, die mit SecDocs archiviert werden sollen, über eine synchrone Web-Service-Schnittstelle vom Client in das SecDocs-Archivierungssystem übertragen. Alle zu übertragenden Daten werden dabei im SOAP-Envelope selbst verpackt. Diese Vorgehensweise ist jedoch für Dateien in der Größe von einem oder mehreren Gigabyte (z.B. medizinische Bilddaten) nicht geeignet, denn die Größe eines SOAP-Requests ist auf 2 GB beschränkt.

Daher bietet SecDocs die Möglichkeit, solche Dateien getrennt vom SDO als sogenannte „externe Datenobjekte“ zu archivieren.

Ein SDO enthält dann anstelle des Dokuments nur einen Verweis auf ein externes Dokument: die **externe Referenz-ID**. Diese ist eindeutig innerhalb eines Mandanten.

Die zu archivierenden Objektdaten werden unabhängig vom SOAP-Request auf den Server übertragen (**externe Datenobjekte**), d.h. jede Übertragung findet in zwei Schritten statt.

- Archivieren von Dokumenten
 1. Die externen Datenobjekte werden vom client-lokalen Rechner auf den SecDocs-Server übertragen.
 2. Der `submitSDO`-Request wird ausgeführt.
- Lesen von Dokumenten
 1. Der `retrieveSDO`-Request wird ausgeführt.
 2. Die Daten werden vom Server auf den lokalen Rechner des Client übertragen.

Verfahren zur Datenübertragung

Zum Übertragen der Dateien auf den Server wird das Secure-File-Transfer-Protokoll (SFTP) verwendet. Zu diesem Zweck wird ein integrierter SFTP-Server als fester Bestandteil des SecDocs-Servers installiert (siehe hierzu [Abschnitt „Integrierter SFTP-Server“](#)).

Um die Objektdaten zu übertragen, muss die Client-Anwendung eine *sftp*-Sitzung zu diesem SecDocs-SFTP-Server eröffnen und in dieser Sitzung entsprechende *sftp*-Kommandos ausführen.

Besonderheiten bei der Datenübertragung

Die Ablage der externen Datenobjekte im Archiv und der Zugriff darauf sind nur mandanten- und organisationsspezifisch möglich.

Die Verantwortung für die Datenübertragung wird auf die Client-Anwendung verlagert.

SecDocs bietet Mechanismen, die sicherstellen, dass ein SDO nur archiviert/versiegelt wird, wenn sich die externen Daten während oder nach der Übertragung nicht geändert haben.

Voraussetzungen

Für jedes externe Datenobjekt, das eine abgesetzte (detached) Signatur besitzt, gilt eine Maximalgröße von 50 GB. Der Versuch, ein größeres externes Datenobjekt zu archivieren, führt zur Abweisung der Operation `submitSDO`.

Für jedes externe Datenobjekt, das eine eingebettete (embedded) Signatur enthält, gilt eine Maximalgröße von 50 MB. Der Versuch, ein größeres externes Datenobjekt mit eingebetteter Signatur zu archivieren, führt zur Abweisung der Operation `submitSDO`.

Beachten Sie außerdem Folgendes:

- Die Funktionen Ersetzen (Operation `replaceSDO`), Verschieben (Operation `moveSDO`) und Versionieren werden für SDOs mit Referenzen auf externe Datenobjekte nicht unterstützt.
- Die Einstellung `$SoftWorm=YES` (Ablegen der Daten als SoftWORM-Daten auf dem Storage-System) bezieht sich nur auf das SDO und die Signature Verification Information. Die zum SDO gehörenden externen Datenobjekte unterliegen hingegen nicht diesem Schutz, da sie verdrängbar bleiben sollen. Siehe hierzu [Abschnitt „\\$SoftWorm“](#).

6.1 Arbeiten mit externen Datenobjekten

- Archivieren eines externen Datenobjekts
- Lesen eines archivierten externen Datenobjekts
- Löschen eines archivierten externen Datenobjekts

6.1.1 Archivieren eines externen Datenobjekts

Das Archivieren eines externen Datenobjekts läuft in den nachfolgend beschriebenen Schritten ab:

1. [Anfordern einer oder mehrerer externer Referenz-IDs](#)
2. [Übertragen der externen Datenobjekte](#)
3. [Adressieren von Daten im SDO](#)
4. [submitSDO-Request](#) oder [Löschen einer nicht benutzten AOID](#)

Anfordern einer oder mehrerer externer Referenz-IDs

Die Client-Anwendung fordert von SecDocs neben einer AOID die benötigte Anzahl von externen Referenz-IDs für ihre zu archivierenden externen Datenobjekte an. Pro AOID können maximal 10 externe Referenz-IDs angefordert werden (Operation `getAOIDWithRef` in Abschnitt ["Operation getAOIDWithRef"](#)).

SecDocs erzeugt Referenz-IDs, die innerhalb eines Mandanten eindeutig sind, und hinterlegt sie in der SecDocs-Datenbank. Die Client-Anwendung trägt diese Referenz-IDs für die später erfolgende Operation `submitSDO` ins SDO ein.

Überprüfen der Datenintegrität

Da mit wachsender Dateigröße die Fehlerwahrscheinlichkeit bei der Datenübertragung zunimmt, ist eine Kontrolle der Unversehrtheit der übertragenen Daten unbedingt erforderlich. SecDocs bietet zu diesem Zweck eine Hash-Wert-Kontrolle an, die den gesamten Zeitraum vom Absenden der Daten durch die Anwendung bis zur endgültigen Versiegelung durch die Operation `submitSDO` abdeckt. Um die Hash-Wert-Kontrolle in vollem Umfang zu nutzen, muss die Anwendung bei der AOID-Anforderung für jedes externe Datenobjekt einen selbst erzeugten Hash-Wert angeben, den SecDocs bei der Ausführung der Operation `submitSDO` zur Prüfung der Unversehrtheit der übertragenen Daten heranzieht, siehe [„Überprüfen der Datenintegrität“](#).

Übertragen der externen Datenobjekte

Die Client-Anwendung eröffnet eine Sitzung am SecDocs-SFTP-Server, wobei sie sich mit Mandantennamen, Organisation und Rolle authentisiert. SecDocs authentifiziert den Aufrufer anhand der Mandantenkonfiguration von SecDocs (siehe [Abschnitt „Authentisierung“](#)). In dieser Sitzung kopiert die Client-Anwendung ihre externen Datenobjekte unter Angabe der jeweiligen externen Referenz-ID in den mandanten- und organisationsspezifischen Übergabebereich auf dem Server. Auf diese Weise wird eine strikte Mandantentrennung sichergestellt.

SecDocs überprüft die verwendeten externen Referenz-IDs anhand der Datenbankeinträge auf ihre Gültigkeit.

Adressieren von Daten im SDO

Siehe hierzu auch: [Abschnitt „Adressierung von Daten im SDO“](#) und [„Referenz-IDs ins SDO eintragen“](#) (in ["Archivieren eines externen Datenobjekts"](#)).

Bei externen Datenobjekten sind die zu archivierenden Daten nicht Bestandteil des SDOs. Das SDO enthält stattdessen externe Referenz-IDs als Verweise auf die extern zu archivierenden Datenobjekte. Die Client-Anwendung muss für jedes externe Datenobjekt eine eigene externe Referenz-ID anfordern und diese im SDO in die XML-Elemente eintragen, auf die mit den Systemschlüsseln `$ContentRef` bzw. `$ExternalRef` verwiesen wird (siehe Abschnitte [„\\$ContentRef“](#) und [„\\$ExternalRef“](#)). Diese Schlüssel ersetzen bei externen Datenobjekten den Schlüssel `$Content`. Genauerer siehe unter [„Referenz-IDs ins SDO eintragen“](#) (in ["Archivieren eines externen Datenobjekts"](#)).

submitSDO-Request

Um die Archivierung abzuschließen, ruft die Client-Anwendung nach der erfolgreichen Übertragung aller Datenobjekte die Operation `submitSDO` mit Angabe der verwendeten Referenz-IDs im SDO auf (siehe "[Operation submitSDO](#)").

Die Ausführung der Operation `submitSDO` umfasst folgende Aktionen:

- Überprüfen der Datenintegrität

Um die Datenintegrität für die gesamte Phase bis zur Versiegelung sicherzustellen, muss die Client-Anwendung selbst Hash-Werte für die zu übertragenden externen Dokumente bereitstellen.

Zu diesem Zweck hat die Client-Anwendung die Möglichkeit, bei Anfordern einer AOID mit externen Referenz-IDs für jedes externe Datenobjekt einen selbst erzeugten Hash-Wert (zusammen mit dem dabei verwendeten Hash-Algorithmus) anzugeben, siehe „[Anfordern einer oder mehrerer externer Referenz-IDs](#)“. Die Wahl des Hash-Algorithmus bleibt dabei der Anwendung überlassen, jedoch muss der angegebene Hash-Algorithmus in SecDocs verfügbar sein.

Bei der Ausführung der Operation `submitSDO` bildet SecDocs ebenfalls den Hash-Wert des externen Datenobjekts und vergleicht diesen mit dem vom Client angegebenen Wert. Dadurch kann SecDocs eine eventuelle Diskrepanz feststellen, deren Ursache eine fehlerhafte Datenübertragung, unbeabsichtigte Veränderung oder Manipulation sein kann.

Für signierte externe Dokumente, deren Signatur bei der Archivierung geprüft werden soll, ist eine solche zusätzliche Hash-Wert-Kontrolle nicht unbedingt erforderlich. Deshalb wird es der Anwendung freigestellt, sie zu nutzen.

- Signaturprüfung

Das Ergebnis der Signaturprüfung wird in der Signature Verification Information hinterlegt. Siehe hierzu auch „[Verifizierung der Signaturen](#)“ (in "[Archivierung](#)").

i Für jedes externe Datenobjekt, das eine eingebettete (embedded) Signatur enthält, gilt eine Maximalgröße von 50 MB. Der Versuch, ein größeres externes Datenobjekt mit eingebetteter Signatur zu archivieren, führt zur Abweisung der Operation `submitSDO`.

- Endgültiges Archivieren

SecDocs verschiebt die externen Datenobjekte an den endgültigen Ablageort im SecDocs-Archiv (siehe [Abschnitt „Ablagestruktur für externe Datenobjekte“](#)). Damit sind sie externen Zugriffen über SFTP entzogen.

Nach einer erfolgreichen Ausführung der Operation `submitSDO` ist kein erneuter SFTP-Transfer für die Referenz-IDs dieses SDOs mehr möglich (siehe „[Übertragen der externen Datenobjekte](#)“).

- Versiegelung

Siehe hierzu auch: [Abschnitt „Gruppieren und Versiegeln der Datenobjekte gemäß ArchiSig-Konzept“](#) und [Abschnitt „Erzeugen des Evidence Records“](#).

Die Hash-Werte der externen Datenobjekte werden als Einzelobjekte in den Hash-Baum eingetragen. Der damit erzeugte Evidence Record ist somit für den Nachweis der Integrität jedes einzelnen extern referenzierten Objekts ausreichend, d.h. der Beweiswert eines Evidence Records erstreckt sich auch auf die extern referenzierten Datenobjekte.

Löschen einer nicht benutzten AOID

Erfolgt nach dem Anfordern einer AOID mit externen Referenz-IDs (siehe „[Anfordern einer oder mehrerer externer Referenz-IDs](#)“) innerhalb einer festgelegten Zeitspanne kein `submitSDO` für diese AOID, löscht SecDocs diese nicht benutzte AOID. Zusätzlich werden auch alle mit dieser AOID verknüpften externen Referenz-IDs und eventuell bereits übertragene Datenobjekte oder -Fragmente im Übergabebereich gelöscht.

6.1.2 Lesen eines archivierten externen Datenobjekts

Das Lesen eines externen Datenobjekts läuft in den nachfolgend beschriebenen Schritten ab:

1. `retrieveSDO-Request`
2. Übertragen der externen Datenobjekte
3. Löschen der symbolischen Links

retrieveSDO-Request

Mit dem Aufruf der Operation `retrieveSDO` erhält die Client-Anwendung ein SDO aus dem Archiv. Das SDO enthält die Referenz-IDs auf die externen Datenobjekte. SecDocs stellt diese Datenobjekte über symbolische Links im mandanten- und organisationsspezifischen Übergabebereich zur Verfügung.

Übertragen der externen Datenobjekte

Die Client-Anwendung eröffnet eine Sitzung am SecDocs-SFTP-Server, wobei sie sich wie bei der Archivierung mit Mandantennamen, Organisation und Rolle authentisiert.

In dieser Sitzung kopiert die Client-Anwendung die im mandanten- und organisationsspezifischen Übergabebereich auf dem Server bereitgestellten Datenobjekte unter Angabe der jeweiligen, dem SDO entnommenen, externen Referenz-IDs auf den lokalen Rechner.

Überprüfen der Datenintegrität

Bei der Datenübertragung vom SecDocs-Übergabebereich zum lokalen Anwenderrechner führt SecDocs keine eigene Hash-Wert-Kontrolle durch. Die Client-Anwendung kann die Datenintegrität jedoch selbst überprüfen, indem sie mit der Operation `requestForEvidence` die Evidence-Records ermittelt und damit eine externe Verifikation durchführt.

Löschen der symbolischen Links

Die symbolischen Links bleiben auch nach einem erfolgreichen Abschluss des `sftp`-Kommandos `get` erhalten. Auf diese Weise sind nach einem erfolgreichen `retrieveSDO-Request` beliebig viele SFTP-Transfers für die externen Datenobjekte dieses SDOs möglich. Zwei oder mehrere Anwendungen können gleichzeitig auf die externen Dokumente derselben AOID lesend zugreifen, oder eine Anwendung kann das `sftp`-Kommando `get` wiederholen, wenn die vorhergehende Übertragung fehlerhaft war.

Es liegt grundsätzlich in der Verantwortung der Client-Anwendung, die symbolischen Links im Übergabebereich mit dem `sftp`-Kommando `rm` zu löschen, wenn diese nicht mehr benötigt werden, z.B. weil die externen Datenobjekte erfolgreich übertragen wurden.

Um einen Ressourcenengpass im mandanten- und organisationsspezifischen Übergabebereich zu vermeiden, überwacht SecDocs die symbolischen Links, die sich durch die `retrieveSDO-Requests` eines Mandanten angesammelt haben, mit einem internen Monitor (siehe `Monitor ExternalFileCleanup` auf "[Operation performAction](#)"). Nach Ablauf einer durch den Konfigurationsparameter `externalFilesRetrieveTimeout` (siehe "[Konfigurationsdatei secdocs.properties](#)") festgelegten Zeitdauer löscht SecDocs diese symbolischen Links.

6.1.3 Löschen eines archivierten externen Datenobjekts

`deleteSDO` löscht neben dem SDO auch alle mit diesem SDO verknüpften externen Datenobjekte im mandanten- und organisationsspezifischen Übergabebereich und vorhandene symbolische Links.

Beachten Sie:

Sollte sich die Ausführung der Operationen `deleteSDO` und `retrieveSDO` für dasselbe SDO zeitlich überschneiden, wird ein nach einem erfolgreichen `retrieveSDO`-Request ausgeführtes *sftp*-Kommando `get` einen Fehler melden, da es keine externen Referenz-IDs findet.

6.2 Integrierter SFTP-Server

Das zur Übertragung externer Objekte auf den Server verwendete Secure-File-Transfer-Protokoll (SFTP) ist eine auf der Secure Shell (SSH) aufbauende Form des Netzwerkprotokolls FTP.

Das File Transfer Protocol FTP ist ein im Standard IETF RFC 959 von 1985 spezifiziertes Netzwerkprotokoll zur schnellen Übertragung von Dateien über IP-Netzwerke. FTP kann Dateien vom Server zum Client (Herunterladen) oder vom Client zum Server (Hochladen) übertragen (File Exchange Protocol), Verzeichnisse anlegen und auslesen sowie Verzeichnisse und Dateien umbenennen oder löschen.

Das Secure Shell (SSH) Protokoll ermöglicht eine sichere, authentifizierte und verschlüsselte Verbindung zwischen zwei Rechnern über ein unsicheres Netzwerk.

Die Einbettung des SFTP-Servers in SecDocs ist im [Abschnitt „Komponenten von SecDocs“](#) dargestellt.

Der Zugriff auf den SecDocs-SFTP-Server (bzw. auf dort abgelegte Verzeichnisse und Dateien) ist durch Authentisierung mittels Benutzername und Kennwort geschützt.

So kann der Zugriff auf veröffentlichte Ressourcen auf einen bestimmten Benutzerkreis beschränkt werden. Näheres hierzu siehe im [Abschnitt „Authentisierung“](#).

Für die Datenübertragung wird auf der Client-Seite ein *sftp*-Client benötigt. Die Kommunikation mit den Client-Anwendungen wird innerhalb des SFTP-Servers über einen konfigurierbaren TCP/IP-Port abgewickelt (Konfigurationsparameter `sftpTcpPort`). Als Default-Portnummer wird 2121 verwendet.

6.2.1 Installation und Konfiguration

SFTP-Server

Der SFTP-Server ist ein fester Bestandteil von SecDocs und muss nicht eigens installiert werden.

Zum Betrieb des SFTP-Servers müssen Sie jedoch vor dem Start von SecDocs die folgenden Konfigurationsparameter in der Datei `secdocs.properties` mit gültigen Werten versorgen:

`archiveRootExternalFiles`

Pfad des Wurzelverzeichnis, unter dem Dokumente, die mit dem SFTP-Server zu SecDocs transferiert wurden, abgelegt werden (Details siehe "[Konfigurationsdatei secdocs.properties](#)").

`createSftpServer=true`

Stellt ein, dass der integrierte SFTP-Server zusammen mit SecDocs gestartet wird. Nur in diesem Fall ist eine Archivierung externer Dokumente möglich (Details siehe "[Konfigurationsdatei secdocs.properties](#)").

`sftpTcpPort`

TCP/IP-Port, über den die Kommunikation des SFTP-Servers mit den Client-Anwendungen abgewickelt wird. Bei diesem Parameter können Sie den eingestellten Standardwert unverändert übernehmen (Details siehe "[Konfigurationsdatei secdocs.properties](#)").

Beispiel:

```
archiveRootExternalFiles=/home/secdocs/SecDocsExtArchiveRoot
createSftpServer=true
sftpTcpPort=2121
```

Logging:

Logging-Einträge, die den SFTP-Server betreffen, finden sich in der Logdatei der WildFly SecDocs-Server-Instanz `server.log`.

SFTP-Client

Für die Kommunikation mit dem SFTP-Server müssen Sie einen eigenen SFTP-Client verwenden. Hierzu können Sie das in OpenSSH enthaltene Shell-Kommando `sftp` benutzen. Alternativ können Sie auch verschiedene OpenSource C- und Java-Bibliotheken zur Programmierung eines Clients heranziehen, sofern diese Software das SSH2-Protokoll unterstützt.

6.2.2 Authentisierung

Um eine sichere Verbindung zwischen dem SFTP-Client und dem SFTP-Server herzustellen, muss sich die Client-Anwendung vor jeder Sitzung beim SFTP-Server mit einem Benutzernamen anmelden und mit einem Passwort authentisieren.

Anmelden mit Benutzernamen

Die Client-Anwendung meldet sich beim SFTP-Server mit einem Benutzernamen an, der folgendermaßen aufgebaut sein muss:

Mandant:Organisation:Rolle

Damit erhält SecDocs Informationen über den Mandanten, seine Organisation und seine Rolle und kann die Rolle der Client-Anwendung im SFTP-Server eindeutig identifizieren. Auf diese Weise ist die gleiche rollenspezifische Authentifizierung wie bei den SecDocs SOAP-Requests gewährleistet.

Ungültige Mandanten, Organisationen oder Rollen werden abgewiesen und so jeder weitere Zugriff auf den SFTP-Server verweigert.

Authentisieren mit Passwort

Bei der Anmeldung wird der anmeldende Benutzer interaktiv zur Passwort-Eingabe aufgefordert. Das Passwort muss dasselbe rollenspezifische SecDocs-Passwort sein, das auch im SOAP-Request verwendet wird. Das eingegebene Passwort wird verschlüsselt übertragen und der Benutzer kann anschließend durch SecDocs authentifiziert werden.

Um eine ausreichende Sicherheit zu gewährleisten, sind innerhalb einer Sitzung maximal 3 Zugriffsversuche möglich.

6.2.3 Öffnen einer SFTP-Sitzung

Sie eröffnen eine SFTP-Sitzung, z.B. unter Linux, mit dem Shell-Kommando *sftp*.

```
sftp {-oPort=<Port> | -P <Port>} [-o <ssh-option>]  
<Tenant>:<Organisation>:<Role>@<Server-Host>
```

<Port>

Portnummer des TCP/IP-Ports, die mit dem Parameter `sftpTcpPort` (siehe "[Konfigurationsdatei secdocs.properties](#)") in der Konfigurationsdatei `secdocs.properties` eingestellt wurde.

Die zu verwendende Eingabeoption für die Portnummer (`-P <Port>` oder `-oPort=<Port>`) hängt von der eingesetzten OpenSSH-Version ab.

<Tenant>

Name des Mandanten, für den das externe Datenobjekt archiviert werden soll.

<Organisation>

Name der Organisation, für die das externe Datenobjekt archiviert werden soll.

<Role>

Rolle, unter der die Archivierung des externen Datenobjekts durchgeführt werden soll. Das Shell-Kommando *sftp* ist nur für solche Rollen erlaubt, in deren Funktionsumfang die Operation `getAOIDWithRef` enthalten ist. Von den voreingestellten Rollen ist dies nur die Rolle `Archivar` (voreingestellte Rollen siehe [Abschnitt „Berechtigungskonzept“](#)).

<Server-Host>

Hostname des SecDocs-Servers

<ssh-option>

ssh-Optionen im Format, wie sie in der OpenSSH SSH Client Konfigurationsdatei (`ssh_config`) verwendet werden.

Genauerer siehe in http://linux.die.net/man/5/ssh_config.

Der SecDocs-SFTP-Server lässt nur die Angabe der folgenden Ciphers zu: "aes128-cbc", "aes128-ctr", "aes192-cbc", "aes256-cbc", "aes256-ctr".

Datenkompression wird nicht unterstützt.

Es wird empfohlen, die Hostkey-Überprüfung ("Host Key Checking") nicht zu nutzen.

Beispiel:

```
sftp -P 2121 -o User="Mandant1:Org1:Archivar" myServer2
```

Nach erfolgreicher Authentisierung beim SFTP-Server (siehe [Abschnitt „Authentisierung“](#)) können Sie weitere *sftp*-Kommandos eingeben (siehe [Abschnitt „Kommandos für den SFTP-Server“](#)).

Nach Beendigung des Datentransfers melden Sie sich mit `quit`, `bye` oder `exit` ab.

6.2.4 Datenübertragung

Nach erfolgreicher Authentisierung durch den SFTP-Server erhält die Client-Anwendung eine Eingabeaufforderung, nach der sie weitere *sftp*-Kommandos eingeben kann.

Damit kann die Client-Anwendung die gewünschte Datenübertragung unter Verwendung der passenden externen Referenz-ID vornehmen:

- Hochladen des externen Datenobjekts
Hierzu dient das *sftp*-Kommando `put`. Es muss nach einer erfolgreichen Operation `getAOIDWithRef` und vor `submitSDO` ausgeführt werden.

Bis zu einer erfolgreichen Ausführung der Operation `submitSDO` zur Archivierung (siehe unter „[SDO mit submitSDO archivieren](#)“ (in "[Archivieren eines externen Datenobjekts](#)")), kann die Client-Anwendung für dieselbe externe Referenz-ID beliebig viele Datentransfers mit `put` durchführen. Jedoch muss sie in diesem Fall vor jedem `put` das bereits übertragene externe Dokument mit `rm referenceID` löschen.
- Herunterladen des externen Datenobjekts
Hierzu dient das *sftp*-Kommando `get`. Es muss nach einer erfolgreichen Operation `retrieveSDO` ausgeführt werden.

6.2.5 Kommandos für den SFTP-Server

Zur Kommunikation mit dem SFTP-Server können Sie die folgenden *sftp*-Kommandos verwenden:

`dir [remote-directory]`

Anzeigen der im mandanten- und organisations-spezifischen Übergabebereich existierenden Directories und Datenobjekte

`get ref [local-file]`

Kopieren der durch *ref* referenzierten Datei im mandanten- und organisationsspezifischen Übergabebereich auf den lokalen Rechner.

Der Name *ref* muss eine registrierte externe Referenz-ID sein, die durch einen vorausgehenden Aufruf der Operation `retrieveSDO` ermittelt wurde.

`ls [-lafhlmrSt] [path]`

Anzeigen der im mandanten- und organisations-spezifischen Übergabebereich existierenden externen Objekte.

Die einzelnen Optionen sind in <http://linux.die.net/man/1/sftp> beschrieben.

`put local-file ref`

Kopieren einer lokalen Datei in den mandanten- und organisationsspezifischen Übergabebereich.

Die Optionen `-P`, `-p` und `-r` werden nicht unterstützt.

local-file muss angegeben und der Name einer Datei (nicht der eines Verzeichnisses) sein.

ref muss angegeben und eine registrierte externe Referenz-ID sein.

Existiert die Datei im Übergabebereich bereits, wird die Übertragung abgelehnt, um konkurrierende Zugriffe zu vermeiden.

Die im Übergabebereich gespeicherten Datenobjekte bleiben dort so lange stehen, bis sie entweder im Rahmen eines `submitSDO` endgültig archiviert oder aber mit `rm ref` oder vom SecDocs-Monitor `AOIDWithRefCleanup` ("[Operation performAction](#)") gelöscht werden.

`rm ref`

Löschen der durch *ref* referenzierten Datei oder des symbolischen Links im mandanten- und organisationsspezifischen Übergabebereich. Der Name *ref* muss eine registrierte externe Referenz-ID sein.

Beachten Sie folgende Einschränkungen:

- Der SFTP-Server unterstützt ausschließlich die hier angegebenen *sftp*-Kommandos
- Sie können mit dem SFTP-Server keine Shell auf dem SecDocs-Rechner via *ssh*-Kommando starten.
- Sie können den SFTP-Server nicht mit dem Kommando *scp* ansprechen.
- Sie können die Option, eingehende Verbindungen im SFTP-Server auf andere Rechner weiterzuleiten (Port Forwarding), nicht verwenden.

6.2.6 Error-Codes an der Programmschnittstelle

Eine als SFTP-Client eingesetzte Software erhält vom SecDocs-SFTP-Server als Antwort auf einen Request eine standardisierte Ausgabestruktur, die „Status Response“.

Der Aufbau dieser Ausgabestruktur ist z.B. in <http://tools.ietf.org/html/draft-ietf-secsh-filexfer-13#page-48> beschrieben.

Bestandteil der Status Response sind unter anderem

- ein Fehlercode (`error/status code`)
Der SecDocs-SFTP-Server gibt hier einen standardisierten Fehlercode zurück. Mögliche Werte sind z.B. in <http://tools.ietf.org/html/draft-ietf-secsh-filexfer-13#page-49> beschrieben.
- eine Fehlermeldung (`error message`)
Hierbei handelt es sich um eine SecDocs-spezifische SFTP-Meldung, bestehend aus Meldungsschlüssel und Meldungstext.
Eine Liste der möglichen Fehlermeldungen finden Sie im im Handbuch "[SecDocs-Rückgabewerte](#)" ([SD4] im [Abschnitt "Literatur"](#)).

Übersicht über die Zuordnung von Fehlercode und Fehlermeldung:

error/status code		Meldungsschlüssel der möglichen SFTP-Meldungen in error message
symbolischer Name	Wert	
SSH_FX_OK	0	-
SSH_FX_EOF	1	-
SSH_FX_NO_SUCH_FILE	2	FTP0005
SSH_FX_PERMISSION_DENIED	3	FTP0010, FTP0016, FTP0017
SSH_FX_OP_UNSUPPORTED	8	FTP0002
SSH2_FX_INVALID_HANDLE	9	FTP0008
SSH2_FX_NO_SUCH_PATH	10	FTP0006, FTP0007
SSH2_FX_FILE_ALREADY_EXISTS	11	FTP0020
SSH2_FX_QUOTA_EXCEEDED	15	FTP0021
SSH2_FX_INVALID_FILENAME	20	FTP0015
SSH_FX_FAILURE	4	sonstige

6.3 Schritt-für-Schritt-Anleitung

Dieser Abschnitt beschreibt wichtige Arbeitsabläufe beim Einsatz von externen Datenobjekten Schritt für Schritt. Halten Sie die Reihenfolge ein, in der die beschriebenen Arbeitsschritte aufeinanderfolgen, wenn Sie die Anleitungen in Ihrem System nachvollziehen.

Die Installation von SecDocs muss abgeschlossen sein.

6.3.1 Vorbereitungen

Konfigurationsparameter einstellen

Um externe Datenobjekte archivieren zu können, müssen Sie vor dem Start von SecDocs in der Konfigurationsdatei `secdocs.properties` die folgenden Parameter setzen:

- `archiveRootExternalFiles`
Pfad des Wurzelverzeichnisses, unter dem Dokumente, die mit dem SFTP-Server zu SecDocs transferiert wurden, abgelegt werden (Details siehe ["Konfigurationsdatei secdocs.properties"](#)).
- `createSftpServer=true`
Stellt ein, dass der integrierte SFTP-Server zusammen mit SecDocs gestartet wird. Der Start des SFTP-Servers ist Grundvoraussetzung für das Archivieren externer Dokumente (Details siehe ["Konfigurationsdatei secdocs.properties"](#)).
- `sftpTcpPort`
TCP/IP-Port, über den die Kommunikation des SFTP-Servers mit den Client-Anwendungen abgewickelt wird. Diesen Parameter müssen Sie nur angeben, wenn Sie einen anderen Port verwenden wollen als den standardmäßig eingestellten Port 2121.

Beispiel

```
archiveRootExternalFiles=/home/secdocs/SecDocsExtArchiveRoot
createSftpServer=true
sftpTcpPort=2121
```

Einrichten eines SDO-Typs

Die prinzipielle Vorgehensweise beim Einrichten eines SDO-Typs entnehmen Sie den Abschnitten [„Von den Daten zum SDO-Typ“](#) und [„Operation createSDOType“](#).

Wenn Sie ein Dokument als externes Datenobjekt archivieren wollen, müssen Sie beim zugrunde liegenden SDO-Typ statt des Systemschlüssels `$Content` (siehe ["\\$Content"](#)) einen der Systemschlüssel `$ContentRef` (siehe ["\\$ContentRef"](#)) oder `$ExternalRef` (siehe ["\\$ExternalRef"](#)) verwenden.

Diese beiden Systemschlüssel geben wie auch `$Content` den Pfad zu einem Element im SDO an. Im Gegensatz dazu wird dieses Element bei `$ContentRef` bzw. `$ExternalRef` jedoch nicht das Primärdokument selbst sondern einen Verweis darauf enthalten (Näheres siehe unter [„Referenz-IDs ins SDO eintragen“](#) (in ["Archivieren eines externen Datenobjekts"](#))).

Den Schlüssel `$ContentRef` müssen Sie verwenden, wenn zu dem externen Datenobjekt Signaturen gehören. Dies können eingebettete Signaturen sein, die im Datenobjekt enthalten sind, oder abgesetzte Signaturen im SDO, auf die mit `$Signature` verwiesen wird. Der Schlüssel `$ExternalRef` ist für Datenobjekte ohne Signatur vorgesehen.

6.3.2 Archivieren eines externen Datenobjekts

Um ein externes Datenobjekt zu archivieren, sind mehrere Arbeitsschritte erforderlich. Die einzelnen Arbeitsschritte werden in diesem Abschnitt in der erforderlichen Reihenfolge dargestellt.

Siehe hierzu auch [Abschnitt „Archivieren eines externen Datenobjekts“](#).

Hash-Wert des externen Datenobjekts erzeugen

Dieser Schritt ist nicht zwingend erforderlich, ist aber empfehlenswert. Mit Hilfe eines mitübertragenen Hash-Werts kann SecDocs zu einem späteren Zeitpunkt eine Prüfung der Unversehrtheit der übertragenen Daten durchführen und so die Datenintegrität für die gesamte Phase bis zur Versiegelung sicherstellen.

Beachten Sie, dass der zum Erzeugen des Hash-Werts verwendete Hash-Algorithmus in SecDocs verfügbar sein muss (siehe [Abschnitt „Operation getHashAlgorithms“](#)).

Zur Ermittlung des korrekten Hash-Wertes, wie er in SecDocs verwendet wird, steht ein Shell-Skript zur Verfügung (siehe [Abschnitt „Hash-Wert erzeugen \(Skript mksha\)“](#)).

Externe Referenz-ID anfordern

Mit der Operation `getAOIDWithRef` fordern Sie neben einer AOID die benötigte Anzahl von externen Referenz-IDs für Ihre zu archivierenden externen Datenobjekte an (siehe [Abschnitt „Operation getAOIDWithRef“](#)).

Für jedes zu archivierende externe Dokument benötigen Sie dabei eine eigene Referenz-ID. Mit einem Aufruf von `getAOIDWithRef`, d.h. pro AOID, können Sie maximal 10 Referenz-IDs anfordern.

Geben Sie beim `getAOIDWithRef`-Request für jedes externe Datenobjekt den unter [„Hash-Wert des externen Datenobjekts erzeugen“](#) erzeugten Hash-Wert an. Falls Sie für ein Dokument keinen Hash-Wert erzeugt haben, geben Sie stattdessen eine frei definierbare Kurzbeschreibung an.

Achten Sie auf eine korrekte Zuordnung von externen Datenobjekten, externen Referenz-IDs und Hash-Werten.

i Beachten Sie, dass Sie die folgenden Schritte [„Externe Datenobjekte mit SFTPübertragen“](#), [„Referenz-IDs ins SDO eintragen“](#) und [„SDO mit submitSDO archivieren“](#), innerhalb der Zeitspanne ausführen müssen, die mit dem Konfigurationsparameter `aoidWithRefKeepReservedPeriod` (siehe ["Konfigurationsdatei secdocs.properties"](#)) festgelegt ist.

Externe Datenobjekte mit SFTP übertragen

Wenn Sie die relevanten Referenz-IDs erhalten haben, übertragen Sie die zu archivierenden externen Datenobjekte von Ihrem lokalen Rechner in den Übergabebereich.

-
- > Eröffnen Sie eine SFTP-Sitzung mit dem Shell-Kommando

```
sftp -P port -o User="tenant:organisation:role" server-host
```

z.B. `sftp -P 2121 -o User="Mandant1:Org1:Archivar" myServer2`

Näheres finden Sie im [Abschnitt „Öffnen einer SFTP-Sitzung“](#).

Nach dem Aufruf erfolgt eine Passwortabfrage, bei der Sie das SecDocs-spezifische Passwort der beim Aufruf angegebenen Rolle angeben müssen.

Nach erfolgreicher Authentisierung beim SFTP-Server (siehe [Abschnitt „Authentisierung“](#)) erhalten Sie eine Eingabeaufforderung. Danach können Sie weitere `sftp`-Kommandos eingeben (siehe [Abschnitt „Kommandos für den SFTP-Server“](#)).

- > Starten Sie die Übertragung eines externen Dokuments mit dem `sftp`-Kommando

```
put local-file referenceID
```

Bis zu einer erfolgreichen Ausführung der Operation `submitSDO` zur Archivierung (siehe [Abschnitt „Archivieren eines externen Datenobjekts“](#)) können Sie für dieselbe externe Referenz-ID beliebig viele Datentransfers mit `put` durchführen. Jedoch müssen Sie in diesem Fall vor jedem `put` das bereits übertragene externe Dokument mit `rm referenceID` löschen.

- > Nach Beendigung des Datentransfers beenden Sie die SFTP-Sitzung mit einem der `sftp`-Kommandos

`quit`, `bye` oder `exit`

Referenz-IDs ins SDO eintragen

Tragen Sie die Referenz-IDs in die XML-Elemente im SDO ein, die Sie in der Filterdefinition durch die Angabe von `$ContentRef` (siehe "[\\$ContentRef](#)") und/oder `$ExternalRef` (siehe "[\\$ExternalRef](#)") festgelegt haben.

Verwenden Sie dabei exakt die Werte, die von der Operation `getAOIDWithRef` in der Response geliefert wurden (Ausgabeelemente `externalReferenceID`).

Für Datenobjekte mit einer oder mehreren Signaturen müssen Sie dabei den Schlüssel `$ContentRef` verwenden; für Datenobjekte ohne Signatur ist der Schlüssel `$ExternalRef` vorgesehen.

Beispiel:

- Filterdefinition:

```
<node mapname="$DataNode" xpath="/ns:myDocument/file1"
>
  <element mapname="$ContentRef" xpath="ns:Ref1"
          maptype="string"/>
  <element mapname="$ContentCode" default="BINARY"/>
  <element mapname="$ContentType" default="OTHERS" />
  ...
</node>

...
<node mapname="$DataNode" xpath="/ns:myDocument/file2"
>
  <element mapname="$ExternalRef" xpath="ns:Ref2"
          maptype="string"/>
  ...
</node>
```

- SDO

```
<tns:myDocument
  xmlns:tns="http://ts.fujitsu.com/secdocs/sdosamples/mydocument"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ts.fujitsu.com/secdocs/sdosamples/
                      mydocument myDocument.xsd">
  ...
  <tns:file1>
    ...
    <tns:Ref1>_26f24b12-f126-4114-a088-de73644c6084/1
    </tns:Ref1>
  </tns:file1>
  <tns:file2>
    ...
    <tns:Ref2>_26f24b12-f126-4114-a088-de73644c6084/2
    </tns:Ref2>
  </tns:file2>
</tns:myDocument>
```

SDO mit submitSDO archivieren

Nachdem Sie alle referenzierten externen Dateien übertragen und das SDO erstellt haben, archivieren Sie das SDO mit der Operation `submitSDO` (siehe ["Operation submitSDO"](#)).

i Beachten Sie, dass die Zeitspanne zwischen der Operation `getAOIDWithRef` und der erfolgreichen Bearbeitung von `submitSDO` den mit dem Konfigurationsparameter `oidWithRefKeepReservedPeriod` eingestellten Wert nicht überschreiten darf.

Beachten Sie auch die „[Voraussetzungen](#)“ (in Operation `submitSDO`).

6.3.3 Lesen eines archivierten externen Datenobjekts

Um ein externes Datenobjekt zu lesen, sind mehrere Arbeitsschritte erforderlich. Die einzelnen Arbeitsschritte werden in diesem Abschnitt in der erforderlichen Reihenfolge dargestellt.

Siehe hierzu auch [Abschnitt „Lesen eines archivierten externen Datenobjekts“](#).

SDO mit retrieveSDO lesen

Lesen Sie ein archiviertes SDO mit der Operation `retrieveSDO` (siehe "[Operation retrieveSDO](#)").

i Beachten Sie, dass Sie die folgenden Schritte „Referenz-IDs ermitteln“ und „Externe Datenobjekte mit SFTP auf den lokalen Rechner übertragen“, innerhalb der Zeitspanne ausführen müssen, die mit dem Konfigurationsparameter `externalFilesRetrieveTimeout` ("Konfigurationsdatei `secdocs.properties`") festgelegt ist.

Referenz-IDs ermitteln

Sie können die externen Referenz-IDs dem SDO entnehmen, das Sie mit `retrieveSDO` erhalten haben. Die Werte stehen in den XML-Knoten, die in der Filterdefinition mit der Angabe von `$ContentRef` oder `$ExternalRef` festgelegt sind. Alternativ können Sie die Werte der Ausgabe der Operation `statusSDO` entnehmen.

Dieser Schritt kann entfallen, wenn Sie eine eigene Buchführung bzgl. der Zuordnung von externen Referenz-IDs zu einer bestimmten AOID haben.

Beispiele hierzu finden Sie unter „Referenz-IDs ins SDO eintragen“ (in "[Archivieren eines externen Datenobjekts](#)") und „Response im Status Sealed für ein SDO mit externer Referenz-ID“ (in "[Operation statusSDO](#)").

Externe Datenobjekte mit SFTP auf den lokalen Rechner übertragen

Nachdem Sie die relevanten Referenz-IDs ermittelt haben, übertragen Sie die gewünschten externen Datenobjekte auf Ihren lokalen Rechner.

- > Eröffnen Sie eine SFTP-Sitzung mit dem Shell-Kommando

```
sftp -P port tenant:organisation:role@server-host
```

z.B. `sftp -P 2121 Mandant1:Org1:Archivar@myServer2`

Näheres finden Sie im [Abschnitt „Öffnen einer SFTP-Sitzung“](#).

Nach dem Aufruf erfolgt eine Passwortabfrage, bei der Sie das SecDocs-spezifische Passwort der beim Aufruf angegebenen Rolle angeben müssen.

Nach erfolgreicher Authentisierung beim SFTP-Server (siehe [Abschnitt „Authentisierung“](#)) erhalten Sie eine Eingabeaufforderung. Danach können Sie weitere `sftp`-Kommandos eingeben (siehe [Abschnitt „Kommandos für den SFTP-Server“](#)).

- > Starten Sie die Übertragung eines externen Datenobjekts mit dem `sftp`-Kommando

```
get referenceID local-file
```

- > Nach Beendigung des Datentransfers können Sie das von SecDocs im mandanten- und organisationsspezifischen Übergabebereich bereitgestellte externe Datenobjekt bzw. die symbolischen Links löschen. Verwenden Sie dazu das *sftp*-Kommando

```
rm referenceID
```

Das Kommando löscht das externe Datenobjekt nur aus dem Übergabebereich, nicht jedoch aus dem SecDocs-Archiv.

i Stellen Sie hierbei jedoch sicher, dass keine weiteren Client-Anwendungen existieren, die dieses externe Datenobjekt noch benötigen.

- > Abschließend beenden Sie die SFTP-Sitzung mit einem der *sftp*-Kommandos

```
quit, bye oder exit
```

SecDocs richtet zum Lesen der externen Datenobjekte symbolische Links ein. Die

i Beachten Sie:
Anzahl der symbolischen Links, die eingerichtet werden können, ist durch die Anzahl der möglichen Unterverzeichnisse im mandanten- und organisationsspezifischen Übergabebereich begrenzt. Diese Grenze ist wiederum abhängig vom verwendeten Storage- und/oder Dateisystem.

Wenn Sie für einen Mandanten sehr viele archivierte externe Datenobjekte lesen wollen, haben Sie folgende Möglichkeiten, um einen diesbezüglichen Ressourcenengpass zu vermeiden:

- Verringern Sie für den betroffenen Mandanten die Zeitspanne, nach der ein symbolischer Link von SecDocs automatisch gelöscht wird (siehe Konfigurationsparameter `externalFilesRetrieveTimeout` ("[Konfigurationsdatei secdocs.properties](#)") und [Abschnitt „Mandantenspezifische Konfigurationsparameter“](#)),
- Löschen Sie nach der erfolgreichen Übertragung eines externen Datenobjekts auf den lokalen Rechner den symbolischen Link mit dem *sftp*-Kommando `rm` (siehe [Abschnitt „Kommandos für den SFTP-Server“](#)). Voraussetzung ist, dass Sie das *sftp*-Kommando `get` nicht wiederholen möchten und keine weitere Anwendung gleichzeitig auf dieses externe Datenobjekt lesend zugreift.

7 Schritt-für-Schritt-Anleitungen

Dieses Kapitel beschreibt für die Nutzung des Web-Service `ArchivingService` wichtige Arbeitsabläufe Schritt für Schritt. Halten Sie die Reihenfolge ein, in der die beschriebenen Arbeitsschritte aufeinanderfolgen, wenn Sie die Anleitungen in Ihrem System nachvollziehen.

Die Installation von SecDocs muss abgeschlossen und SecDocs gestartet sein.

7.1 Von den Daten zum SDO-Typ

Die Definition eines SDO-Typs enthält das XML-Schema für das SDO und die Informationen zur Adressierung von Daten im SDO, die in der zugehörigen Filterdefinition abgelegt sind.

Um einen SDO-Typ einzurichten, sind mehrere Arbeitsschritte erforderlich. Die einzelnen Arbeitsschritte werden in diesem Abschnitt in der erforderlichen Reihenfolge dargestellt.

Diese Anleitung ist für Sie relevant, wenn Sie eine Fachanwendung an SecDocs anbinden.

7.1.1 Datenumfang erfassen und strukturieren

Der Datenumfang legt fest, welche Primärdokumente, Signaturen und Metadaten zusammen in einem SDO archiviert werden sollen.

Datenumfang erfassen und strukturieren

- > Primärdokumente und die dazugehörigen Signaturen und Metadaten erfassen, die aus fachlicher Sicht zusammengehören und in einem SDO archiviert werden sollen.
- > Gegebenenfalls Datenumfang strukturieren.

Wenn mehrere Primärdokumente mit abgesetzten Signaturen archiviert werden sollen, sollten Sie sie bereits in der XML-Datei in Unterstrukturen zusammenfassen, damit sie später im Filter eindeutig referenziert werden können.



Eine Liste der Fragen, die Sie beantworten sollten, wenn Sie den Datenumfang für einen SDO-Typ erfassen, finden Sie im [Abschnitt „Beschreibung eines Datenobjekts“](#).

7.1.2 Struktur des SDOs als XML-Schema definieren

Das XML-Schema (.xsd-Datei) beschreibt die Struktur des SDOs für die Archivierung. Auf diese Struktur wird später beim Einrichten des SDO-Typs verwiesen.

Mit SecDocs wird ein Beispiel für ein XML-Schema ausgeliefert, das die Struktur eines SDOs beschreibt.



Ein Beispiel-XML-Schema von SecDocs und Erläuterungen dazu finden Sie auf "[SDO-Struktur](#)".

Voraussetzungen

Der Datenumfang für einen neuen SDO-Typ ist aus fachlicher Sicht erfasst und strukturiert.

Struktur des SDOs als XML-Schema definieren



Welche Angaben im Schema notwendig sind, entnehmen Sie dem Abschnitt „[Angaben im XML-Schema](#)“ (in "[SDO-Struktur](#)").

7.1.3 Filterdefinition erstellen

Zum SDO-Typ gehören neben der Struktur des SDOs die Informationen zur Adressierung von Daten im SDO, die in der zugehörigen Filterdefinition abgelegt sind. In einem Filter legen Sie fest, an welcher Stelle des als XML-Dokument vorliegenden SDOs die Daten zu finden sind, die für das Archivierungssystem relevant sind.

Mit SecDocs wird das XML-Schema `filter.xsd` ausgeliefert, das die Struktur für SecDocs-Filterdefinitionen vorgibt.



- Eine Liste der Informationsbestandteile einer Filterdefinition finden Sie im Abschnitt "[Adressierung von Daten im SDO](#)".
- Detaillierte Information sowie ein Schema zur Struktur eines SecDocs-Filters finden Sie im Abschnitt "[Filtersyntax](#)".
- Eine Beispiel-Filterdefinition mit Erläuterungen finden Sie im Abschnitt "[Beispiel für eine Filterdefinition](#)".
- Zusammen mit dem Beispiel-Schema werden mehrere Beispielfilter mitausgeliefert.

Voraussetzungen

Die Struktur des neuen SDOs ist als XML-Schema definiert.

Filterdefinition erstellen

- > XML-Datei für die Filterdefinition anlegen, entsprechend der Struktur des von SecDocs vorgegebenen XML-Schemas `filter.xsd`.
- > Bezüge zu den erforderlichen XML-Schemas definieren

Geben Sie folgende Attribute des Wurzelements `<filter>` an:



Detaillierte Informationen zum Wurzelement `<filter>` und den darin erforderlichen Angaben finden Sie im Abschnitt "[Element](#)".

- `schema`: Name des Root-Elements des korrespondierenden XML-Schemas, das die Struktur des SDOs beschreibt.
- `version`: Version des SecDocs-Filter-Schemas; erlaubter Wert: "1.0".
- Es empfiehlt sich, auf das XML-Schema `filter.xsd` zu verweisen, das die Struktur von SecDocs-Filterdefinitionen beschreibt.

- > Dem Namespace der SDO-Schemadefinition eine Kurzbezeichnung zuordnen

Geben Sie folgende Attribute des Elements `<namespace>` an:



Detaillierte Informationen zum Element `<namespace>` und den darin erforderlichen Angaben finden Sie im Abschnitt "[Element](#)".

- `prefix`: Kurzbezeichnung für den Namespace, der mit `uri` festgelegt wurde.
- `uri`: eindeutiger Namespace, der durch die Kurzbezeichnung ersetzt werden soll.

> Informationen zur Adressierung von Daten aus dem SDO herausfiltern

Geben Sie dazu entsprechend viele Elemente `<element>` mit den folgenden Attributen an:



Detaillierte Informationen zum Element `<element>` und den darin erforderlichen Angaben finden Sie im Abschnitt "[Element](#)".

- `mapname`: Schlüssel, dem ein im SDO adressierte Information und/oder ein Standardwert zugewiesen wird, siehe auch "[Element](#)".
- `maptype`: optional; Datentyp der Information, die mit `mapname` bezeichnet wurde, siehe auch "[Element](#)".
- `xpath`: optional; Ausdruck, mit dem ein Element im SDO adressiert wird, siehe auch "[Element](#)".
- `default`: optional; Standardwert, der dem Schlüssel (`mapname`) zugeordnet wird, wenn `xpath` nicht angegeben oder im SDO nicht vorhanden ist, siehe auch "[Element](#)".

> Pro Element `<element>` im Attribut `mapname` folgende Systemschlüssel in der Filterdefinition zuweisen:



Detaillierte Information zu allen Systemschlüsseln von SecDocs finden Sie im Abschnitt "[Verwendung von Schlüsseln](#)".

- Freigabezeitpunkt: entweder `$ExpirationDate` oder `$RetentionPeriod`
- Primärdokumente und Signaturen: `$Content`, `$ContentRef` oder `$ExternalRef`
Bei `$Content` oder `$ContentRef` zusätzlich:
 - `$ContentCode`
 - `$ContentType`
 - für Primärdokumente im SDO, die abgesetzte (detached) Signaturen enthalten, gegebenenfalls `$Signature`, `$SignatureType`
 - wenn zu einem Primärdokument abgesetzte oder eingebettete Signaturen gehören, `$SignatureQualityLevel`, `$SignatureVerification`

> Zusammengehörende Elemente von Primärdokument und Signatur gruppieren

Geben Sie dazu entsprechend viele Elemente `<node>` mit den folgenden Attributen an:



Detaillierte Informationen zum Element `<node>` und den darin erforderlichen Angaben finden Sie im Abschnitt "[Element](#)".

-
- `mapname`: Schlüssel `$DataNode`, dem eine im SDO adressierte Information und/oder ein Standardwert zugewiesen wird, siehe auch "[Element](#)".
 - `xpath`: Teilpfad, mit dem der Knoten im SDO adressiert wird, siehe auch "[Element](#)".
 - `<element>`: XML-Element, das eine bestimmte Information zur Adressierung von Daten aus dem SDO herausfiltert oder eine Zusatzinformation enthält. `<element>` kann mehrmals vorkommen. Siehe auch "[Element](#)".

> `$SDOPath` definieren



Detaillierte Informationen zu `$SDOPath` finden Sie ab "[\\$SDOPath](#)".

7.2 SDO-Typ-spezifische Operationen des Web-Service ArchivingSRService

Die Archivierung eines SDOs erfolgt mit der Operation `submitSDO` (siehe [Abschnitt „Operation submitSDO“](#)), mit der Operation `replaceSDO` können Sie es ersetzen (siehe [Abschnitt „Operation replaceSDO“](#)) und mit der Operation `retrieveSDO` zurückholen (siehe [Abschnitt „Operation retrieveSDO“](#)). Damit die Client-Software diese Operationen ausführen kann, muss der Web-Service entsprechend dem übergebenen SDO-Typ „typisiert“ werden.

Definieren Sie für jedes SDO-Schema jeweils eine Submit- und eine Retrieve-Operation, z.B. `archiviereRechnung` und `leseRechnung` oder `archiviereBestellung` und `leseBestellung`, usw.

Mit SecDocs wird das Template `ArchivingSR.wsdl` als Beispiel für die Schnittstellendefinition des Web-Service `ArchivingSRService` ausgeliefert.

Die Datei `ArchivingSR.wsdl` enthält nur die SDO-spezifischen Submit- und Retrieve-Operationen, hier `submitMultiDocument` und `retrieveMultiDocument` genannt.

Die SDO-Typ-unabhängigen Operationen (z.B. `statusSDO`, `getAOID`, `requestForEvidence` usw.) sind in der Datei `Archiving.wsdl` zusammengefasst.

Zusätzlich wird im Verzeichnis `/home/secdocs/schemas/3.0/samples` als Beispiel ein Schema für den SDO-Typ `MultiDocument.xsd` ausgeliefert.

Nachfolgend ist der Inhalt des Templates `ArchivingSR.wsdl` dargestellt. Die für den SDO-Typ `MultiDocument` spezifischen Stellen sind **blau markiert**. Diese sind also SDO-Typ-spezifisch anzupassen. Optional können Sie auch den Namespace der WSDL (**grün markiert**) ändern.

Inhalt der Datei `ArchivingSR.wsdl`:

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="ArchivingSR"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:sdheader="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
  xmlns:sdarchiving="http://ts.fujitsu.com/secdocs/v3_1/archiving"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:sdo="http://ts.fujitsu.com/secdocs/sdosamples/v1_0/multidocument"
  xmlns:tns="http://ts.fujitsu.com/secdocs/ws/v3_1/archiving"
  targetNamespace="http://ts.fujitsu.com/secdocs/ws/v3_1/archiving">
  <types>
    <schema xmlns="http://www.w3.org/2001/XMLSchema">
      <import namespace="http://ts.fujitsu.com/secdocs/v3_1/secdocs"
        schemaLocation="secdocs.xsd"/>
    </schema>
    <schema xmlns="http://www.w3.org/2001/XMLSchema">
      <import namespace="http://ts.fujitsu.com/secdocs/v3_1/archiving"
```

```

        schemaLocation="ArchivingData.xsd"/>

</schema>

<!-- SDO specific schema files -->

<schema xmlns="http://www.w3.org/2001/XMLSchema">
    <import
namespace="http://ts.fujitsu.com/secdocs/sdosamples/v1_0/multidocument"
        schemaLocation="samples/MultiDocument.xsd"/>
    </schema>
</types>

<message name="SecDocsSoapHeader">
    <part name="secDocsSoapHeader" element="sdheader:soapHeaderData"/>
</message>

<message name="FaultMessage">
    <part name="secDocsSoapFault" element="sdheader:faultDetails"/>
</message>

<message name="SubmitSDO_ResponseMessage">
    <part name="submitSdoResponse" element="sdarchiving:submitSdoResponse"/>
</message>

<message name="RetrieveSDO_RequestMessage">
    <part name="retrieveSdoRequest" element="sdarchiving:retrieveSdoRequest"/>
</message>

<message name="ReplaceSDO_ResponseMessage">
    <part name="replaceSdoResponse" element="sdarchiving:replaceSdoResponse"/>
</message>

<!-- SDO specific messages -->

<message name="SubmitMultiDocument_RequestMessage">
    <part name="multiDocument" element="sdo:multiDocument"/>
</message>

<message name="RetrieveMultiDocument_ResponseMessage">
    <part name="multiDocument" element="sdo:multiDocument"/>
</message>

<message name="ReplaceMultiDocument_RequestMessage">
    <part name="multiDocument" element="sdo:multiDocument"/>
</message>

```

```

<portType name="ArchivingSR_PortType">
  <!-- SDO specific submit/retrieve operations -->
  <operation name="submitMultiDocument">
    <input message="tns:SubmitMultiDocument_RequestMessage"/>
    <output message="tns:SubmitSDO_ResponseMessage"/>
    <fault name="FaultMessage" message="tns:FaultMessage"/>
  </operation>
  <operation name="retrieveMultiDocument">
    <input message="tns:RetrieveSDO_RequestMessage"/>
    <output message="tns:RetrieveMultiDocument_ResponseMessage"/>
    <fault name="FaultMessage" message="tns:FaultMessage"/>
  </operation>
  <operation name="replaceMultiDocument">
    <input message="tns:ReplaceMultiDocument_RequestMessage"/>
    <output message="tns:ReplaceSDO_ResponseMessage"/>
    <fault name="FaultMessage" message="tns:FaultMessage"/>
  </operation>
</portType>
<binding name="ArchivingSR_Binding" type="tns:ArchivingSR_PortType">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="submitMultiDocument">
    <soap:operation soapAction="" style="document"/>
    <input>
      <soap:header message="tns:SecDocsSoapHeader"
        part="secDocsSoapHeader" use="literal"/>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
    <fault name="FaultMessage">
      <soap:fault name="FaultMessage" use="literal"/>
    </fault>
  </operation>

```

```
<operation name="retrieveMultiDocument">
  <soap:operation soapAction="" style="document"/>

  <input>

    <soap:header message="tns:SecDocsSoapHeader"
      part="secDocsSoapHeader" use="literal"/>

    <soap:body use="literal"/>
  </input>
  <output>

    <soap:body use="literal"/>
  </output>
```

```

    <fault name="FaultMessage">
        <soap:fault name="FaultMessage" use="literal"/>
    </fault>
</operation>

<operation name="replaceMultiDocument">
    <soap:operation soapAction="" style="document"/>
    <input>
        <soap:header message="tns:SecDocsSoapHeader"
            part="secDocsSoapHeader" use="literal"/>
        <soap:body use="literal"/>
    </input>
    <output>
        <soap:body use="literal"/>
    </output>
    <fault name="FaultMessage">
        <soap:fault name="FaultMessage" use="literal"/>
    </fault>
</operation>
</binding>

<service name="ArchivingSR_Service">
    <port name="ArchivingSR_Port" binding="tns:ArchivingSR_Binding">
        <soap:address
            location="http://localhost:8080/archiver/ws/3.1/archiving"/>
    </port>
</service>
</definitions>

```

Hinweis

Um aus einer modifizierten ArchivingSR.wsdl Datei den Wrapper-Code mit Hilfe des Tools `wsimport` zu erzeugen, müssen Sie die beiden folgenden Optionen beim Aufruf von `wsimport` angeben:

```

-XadditionalHeaders
-extension

```

Als Beispiel hierzu wird das Skript `genArchivingSRWsClientStubs` ausgeliefert, das den Aufruf von `wsimport` enthält.

Um den Wrapper-Code für die Datei `Archiving.wsdl` selbst zu erzeugen, sind zusätzlich die folgenden Optionen anzugeben:

```
-b ./sparql-protocol-types.xjb  
-b ./result2.xjb
```

7.3 Archiv einrichten

Die Ablagestruktur für die Archivierung richten Sie über folgende zwei Web-Services ein:

- Über den Web-Service `ArchiveAdminService` legen Sie die TSPs für die Versiegelung eines SDOs fest und richten neue Mandanten im Archiv ein.
- Über den Web-Service `MandantAdminService` registrieren Sie die SDO-Typen und richten die Organisationseinheiten innerhalb der Mandanten ein.

7.3.1 TSP bekannt machen und Mandanten einrichten

Zu den grundlegenden Administrationsaufgaben, die das Archiv in seiner Gesamtheit betreffen, gehören folgende Schritte:

1. TSP zur Versiegelung eines SDOs festlegen.
2. Neuen Mandanten im Archiv anlegen. Es wird empfohlen, diesen zunächst als Testmandanten einzurichten.

Diese Schritte betreffen das gesamte Archiv einer SecDocs-Installation und sind über den Web-Service `ArchiveAdminService` durchzuführen.



Eine Beschreibung aller Operationen, die über den Web-Service `ArchiveAdminService` zur Verfügung stehen, finden Sie in [Abschnitt „Web-ServiceArchiveAdminService“](#).

Diese Anleitung ist für Sie relevant, wenn Sie SecDocs-Administrationseinträge für einen Mandanten erstellen wollen.

Web-Service `ArchiveAdminService` aufrufen

- > Um einen TSP einzurichten und einen Mandanten im Archiv anzulegen, ist der Web-Service `ArchiveAdminService` aufzurufen.



Detaillierte Informationen zum Request-Header für den Web-Service `ArchiveAdminService` und den darin erforderlichen Angaben finden Sie ab ["Request-Header"](#).

TSP-Eintrag erstellen

Voraussetzungen

Die Systemadministration muss die Verbindung zum TSP (z.B. entsprechende Einstellungen in der Firewall) ermöglichen.

TSP einrichten

- > Um einen TSP zum Versiegeln von SDOs festzulegen, führen Sie die Operation `createTSP` aus.



Verwenden Sie keine Hash-Algorithmen, die von der Bundesnetzagentur als nicht mehr sicher oder als in absehbarer Zeit unsicher werdend eingestuft werden.



Detaillierte Informationen zur Operation `createTSP` und zu den darin erforderlichen Angaben finden Sie ab ["Operation createTSP"](#).

Mandanteneintrag erstellen

Voraussetzungen

- Die TSPs müssen in SecDocs angelegt sein, siehe oben.

-
- Der Pfad `archiveRoot/mandantPath` muss mit Schreibrecht für den SecDocs-Benutzer zugreifbar sein. Der Pfad `mandantPath` beschreibt den zur Archiv-Wurzel relativen Pfad für den Teilbereich des Mandanten im Archiv. Die Archiv-Wurzel wird mit dem Parameter `archiveRoot` in der Datei `secdocs.properties` konfiguriert, siehe "[Konfigurationsdatei secdocs.properties](#)".

Falls der Parameter `archiveRootExternalFiles` in der Datei `secdocs.properties` angegeben ist (siehe "[Konfigurationsdatei secdocs.properties](#)"), muss auch der Pfad `archiveRootExternalFiles/mandantPath` mit Schreibrecht für den SecDocs-Benutzer zugreifbar sein.

Ausnahme: Wenn der Parameter `createMandantDirLocal` in der Konfigurationsdatei `secdocs.properties` auf `true` gesetzt ist, wird der Pfad `mandantPath` automatisch von SecDocs erzeugt.

Mandanteneintrag erstellen

- > Einen neuen Mandanten im Archiv richten Sie mit der Operation `createMandant` ein.



Detaillierte Informationen zur Operation `createMandant` und zu den darin erforderlichen Angaben finden Sie ab "[Operation createMandant](#)".

7.3.2 SDO-Typ registrieren und Organisationseinheit anlegen

Zu den grundlegenden Administrationsaufgaben, die den Archivbereich eines Mandanten betreffen, gehören folgende Schritte:

1. Neuen SDO-Typ registrieren
2. Organisationseinheit anlegen

Diese Schritte betreffen den Archivbereich eines Mandanten und sind deshalb über den Web-Service `MandantAdminService` durchzuführen.



Eine Beschreibung aller Operationen, die über den Web-Service `MandantAdminService` zur Verfügung stehen, finden Sie in [Abschnitt „Web-ServiceMandantAdminService“ \(5.2 Web-Service MandantAdminService\)](#).

Diese Anleitung ist für Sie relevant, wenn Sie den Archivbereich eines Mandanten administrieren.

Voraussetzungen

- Es wurde festgelegt, welches Primärdokument bzw. welche Primärdokumente zusammengehörig zu archivieren sind, siehe ["Datenumfang erfassen und strukturieren"](#).
- Die Struktur für das SDO ist als XML-Schema (als .xsd-Datei) festgelegt, siehe ["Struktur des SDOs als XML-Schema definieren"](#).
- Eine Filterdefinition ist erstellt, siehe ["Filterdefinition erstellen"](#).
- Der Archivbereich des Mandanten ist im Archiv angelegt, siehe ["TSP bekannt machen und Mandanten einrichten"](#).

Web-Service MandantAdminService aufrufen

- > Um einen neuen SDO-Typ zu registrieren und eine Organisationseinheit im Archiv anzulegen, ist der Web-Service `MandantAdminService` aufzurufen.



Detaillierte Informationen zum Request-Header für den Web-Service `MandantAdminService` und den darin erforderlichen Angaben finden Sie ab ["Request-Header"](#).

SDO-Typ registrieren

Bevor ein SDO archiviert werden kann, müssen Sie den zugehörigen SDO-Typ mit Hilfe des Web-Service `MandantAdminService` in SecDocs registrieren. D.h., der SDO-Typ wird in SecDocs bekannt gemacht, damit anschließend entsprechende SDOs durch die Client-Software archiviert werden können.

-
- > Um den SDO-Typ zu registrieren, ist die Operation `createSDOType` auszuführen. Diese enthält die Definition des SDO-Typs.

i Es gibt keine Funktion zum Löschen eines für einen produktiven Mandanten registrierten SDO-Typs mit existierenden SDOs, da SecDocs dafür sorgen muss, dass die SDO-Typen zu jedem archivierten SDO jederzeit wieder bereitgestellt werden können. Es wird daher empfohlen, einen neuen SDO-Typ zuerst in einer Testumgebung auszutesten oder ihn zunächst für einen Testmandanten zu registrieren.



Detaillierte Informationen zur Operation `createSDOType` und zu den darin erforderlichen Angaben finden Sie ab "[Operation createSDOType](#)".

Organisationseinheit einrichten

- > Um im Archiv eine neue Organisationseinheit innerhalb eines Mandanten einzurichten, ist die Operation `createOrganisation` auszuführen. Dies beinhaltet die Definition der Organisationseinheit und die Definition des Archivbereichs für diese Organisationseinheit, in dem alle SDOs für diese Organisationseinheit archiviert werden.



Detaillierte Informationen zur Operation `createOrganisation` und zu den darin erforderlichen Angaben finden Sie ab "[Operation updateMandant](#)".

7.4 Dokumente archivieren

Zum Archivieren von Dokumenten nutzen Sie die Funktionen des Web-Service `ArchivingService`.

Diese Anleitung ist für Sie relevant, wenn Sie Web-Service-Aufrufe in die Fachanwendung implementieren.

- > Implementieren Sie die Client-Schnittstelle in der Fachanwendung.
- > Rufen Sie den Web-Service `ArchivingService` auf.
- > Folgende Parameter sind im SOAP-Request-Header für den Web-Service `ArchivingService` anzugeben:



- Detaillierte Informationen zum Request-Header für den Web-Service `ArchivingService` und den darin erforderlichen Angaben finden Sie im Abschnitt ["Request-Header"](#).
 - Ein Beispiel mit konkreten Angaben für alle Parameter finden Sie im Abschnitt ["Request-Header"](#).
-
- `operation`: Operation, die ausgeführt werden soll: z.B. `getAOID`
 - `role`: `Archivar` (mit der Option, später auf ein differenziertes Rollenkonzept zu wechseln)
 - `mandant`: Mandantenname, für den die Operation ausgeführt werden soll
 - `orgID`: Organisationseinheit
 - `password`: Mandanten- und rollenspezifisches Passwort
 - `auditID`: optional; Benutzer-ID, die im Audit-Log hinterlegt wird
 - `aoid`: AOID für das SDO, auf das sich die Operation bezieht; nicht bei `getAOID`
 - `coid`: COID für das SDO, auf das sich die Operation bezieht
 - `SDOType`: SDO-Typ; nur bei `submitSDO` und `replaceSDO`

-
- > Nutzen Sie die nachfolgend aufgelisteten Funktionen des Web-Service `ArchivingService` zur Archivierung von Dokumenten.

Archivierungsfunktion	Operation	Details siehe
AOID anfordern	<code>getAOID</code>	"Operation getAOID"
AOID und eine oder mehrere externe Referenz-IDs anfordern	<code>getAOIDWithRef</code>	"Operation getAOIDWithRef"
SDO archivieren	<code>submitSDO</code>	"Operation submitSDO"
SDO ersetzen	<code>replaceSDO</code>	"Operation replaceSDO"
Archiviertes Dokument holen	<code>retrieveSDO</code>	"Operation retrieveSDO"
Dokument löschen	<code>deleteSDO</code>	"Operation deleteSDO"
	<code>forceDeleteSDO</code>	"Operation forceDeleteSDO"
Status/Versionen/Metadaten anzeigen	<code>statusSDO</code>	"Operation statusSDO"
	<code>listSDOVersions</code>	"Operation statusSDO"
	<code>retrieveMetaData</code>	"Operation retrieveMetaData"
	<code>metaDataSDO</code>	"Operation metaDataSDO"
Gültigkeit nachweisen	<code>requestForEvidence</code>	"Operation requestForEvidence"
Anhand der Ablagestruktur recherchieren	<code>navigate</code>	"Operation navigate"
Informationen für die Abrechnung einholen	<code>getAccountingData</code>	"Operation getAccountingData"
SDO verschieben	<code>moveSDO</code>	"Operation moveSDO"
Aufbewahrungsfrist setzen	<code>setExpirationDateTimeSDO</code>	"Operation setExpirationDateTimeSDO"
Freigabezeitpunkt aufheben	<code>forceDeferredSDO</code>	"Operation forceDeferredSDO"
XML-Schema und Filter ausgeben	<code>getSchemaDataSDO</code>	"Operation getSchemaDataSDO"

8 Inbetriebnahme und Überwachung

Dieses Kapitel beschreibt die Konfiguration und die Voraussetzungen für die Inbetriebnahme von SecDocs sowie die verschiedenen Möglichkeiten zur Überwachung und Fehlerdiagnose von SecDocs.

i Dieses Kapitel gibt Ihnen spezifische Hinweise für den Umgang mit SecDocs. Die allgemeine Administration des Basissystems, wie Administration, Monitoring, Sicherung und Pflege, ist hier nicht beschrieben.



Wie Sie das OpenLimit CryptoModule und SecDocs starten und beenden, entnehmen Sie der Installationsbeschreibung zu SecDocs und dem CryptoModule.

8.1 Installation und Konfiguration

Der folgende Abschnitt beschreibt die allgemeinen Konfigurationsparameter für SecDocs. Er ist für Sie relevant, wenn Sie SecDocs installieren und in Betrieb nehmen wollen.

8.1.1 Installation



Die Installation von SecDocs sowie die zugehörigen Software-Voraussetzungen sind abhängig vom aktuellen Release. Fujitsu liefert mit jedem Release von SecDocs eine ausführliche Installationsbeschreibung ("[SecDocs V3.1A01 Installationsanleitung](#)" ([SD3] in Abschnitt "Literatur")). Dieses Dokument beschreibt die Software-Voraussetzungen für die Installation von SecDocs, den Lieferumfang, das Bereitstellen der Ablaufumgebung und die Konfiguration von SecDocs.

8.1.2 Anforderungen für den sicheren Betrieb

Die Ablaufumgebung muss vor Angriffen von außen (zum Beispiel durch Schadsoftware, nicht zugelassene Netzwerkverbindungen oder Viren) geschützt sein. Dazu muss ein leistungsfähiger Virens Scanner und eine sicher eingestellte Firewall eingesetzt werden. Der Virens Scanner muss regelmäßig aktualisiert werden. Für das Betriebssystem müssen regelmäßig die verfügbaren Sicherheitsupdates eingespielt werden.

SecDocs sowie die dem Produkt unterliegende IT-Plattform müssen Sie in einer sicheren Umgebung installieren. Auf den Servern, auf denen SecDocs abläuft bzw. auf denen sich Daten befinden, auf die SecDocs zugreifen soll, darf keine weitere Software installiert sein, die nicht für den Betrieb des Integrationsproduktes benötigt wird.

Die Server müssen gegen unautorisierten, physischen und logischen Zugriff sowie gegen Veränderung geschützt sein.

Alle TCP/IP-Verbindungen, die von SecDocs genutzt werden, müssen gegen unautorisierte Zugriffe und Abhörmöglichkeiten physisch (z.B. durch Isolierung des verwendeten lokalen Netzes) oder logisch (z.B. mit Verschlüsselungssoftware und Firewalls) geschützt sein.

Das als Basis für SecDocs dienende Betriebssystem muss vom Administrator des Produkts stets aktuell gehalten werden, d.h. die vom Hersteller des Betriebssystems veröffentlichten, sicherheitskritischen Updates sind einzuspielen. Ebenso muss der Administrator ein adäquates Virenschutzprogramm mit aktuellen Virensignaturen verwenden.

Der von der SecDocs Security-Komponente "OpenLimit CryptoModule" angebotene Web-Service darf beim Betrieb des Produkts ausschließlich lokal angeboten werden, er muss also bei der Installation des Produkts auf das sogenannte "localhost"-Interface gebunden werden.

Die IT-Infrastruktur muss durch Komponenten (von Drittherstellern) vor Viren und Schadsoftware geschützt sein. Weiterhin muss die IT-Infrastruktur vor Netzwerk-basierten Angriffen geschützt sein. Potenzielle Angriffe über das Internet, ein angeschlossenes Intranet, einen manuellen Zugriff Unbefugter oder Datenaustausch per Datenträger müssen durch die bestehenden Sicherheitsvorkehrungen in der Einsatzumgebung mit hoher Sicherheit abgewehrt werden.

Die SecDocs-Komponenten und die von ihnen genutzten Daten müssen durch geeignete Schutzmechanismen davor geschützt werden, dass sie nichtautorisiert verändert werden.

Der Administrator muss vertrauenswürdig sein und die im Handbuch beschriebenen Installations- und Bedienungsanweisungen sorgfältig befolgen. Weiterhin hat er alle ihm zugänglichen Informationen zum Produkt vertraulich zu behandeln. Die benutzten Passwörter sind von ihm sicher zu verwahren.

8.1.3 Konfigurationsdatei `secdocs.properties`

In der Datei `secdocs.properties` sind die allgemeinen Konfigurationsparameter für SecDocs abgelegt. Die Datei wird mit SecDocs ausgeliefert. Bei der Installation von SecDocs müssen Sie die Parameter entsprechend Ihrer Konfiguration anpassen.

SecDocs erwartet die Datei unter folgendem Pfad:

```
installation-dir/jboss/secdocs/configuration/secdocs/secdocs.properties
```

Zur vereinfachten Administration von Multi-Node-Konfigurationen besteht die Möglichkeit, für alle Verbundteilnehmer eine gemeinsame Konfigurationsdatei zu verwenden. Dies kann mit dem Konzept einer primären und sekundären Konfigurationsdatei erreicht werden.

Näheres hierzu siehe `globalPropertiesFile` unter "[globalPropertiesFile](#)" und [Abschnitt „Die Konfigurationsdatei `secdocs.properties`“](#).

Parameter

i Bei den beiden folgenden Konstanten ist Groß-/Kleinschreibung nicht relevant: `true`, `false`.

`archiveRoot`

Archiv-Wurzel.

Dieser Pfad bezeichnet das Wurzelverzeichnis für das gesamte Archiv. Hier liegen außerdem alle Konfigurationsdateien von SecDocs. Pfade für Teilbereiche von Mandanten im Archiv werden relativ zur Archiv-Wurzel angegeben (Operation `createMandant`, siehe "[Operation `createMandant`](#)").

Der Pfad "Archiv-Wurzel" muss vorhanden sein und auf dem Storage liegen (mounted Volume). Der Benutzer `secdocs` muss mit Schreibrecht auf diesen Pfad zugreifen können.

i `archiveRoot` ist eine Pflichtangabe. Wenn `archiveRoot` nicht gesetzt ist, können Sie SecDocs nicht starten.

`archiveRootExternalFiles`

Archiv-Wurzel für externe Dokumente.

Dieser Pfad bezeichnet für das gesamte Archiv dasjenige Wurzelverzeichnis, unter dem Dokumente abgelegt werden, die mit dem SFTP-Server zu SecDocs transferiert wurden.

Pfade für Teilbereiche von Mandanten im Archiv werden relativ zu dieser Archiv-Wurzel angegeben (Operation `createMandant`, siehe "[Operation `createMandant`](#)").

Der Pfad "Archiv-Wurzel für externe Dokumente" muss vorhanden sein und auf dem Storage liegen (mounted Volume). Der Benutzer `secdocs` muss mit Schreibrecht auf diesen Pfad zugreifen können.

i Wenn `createSftpServer` ("[Konfigurationsdatei secdocs.properties](#)") auf `"true"` gesetzt ist, dann ist `archiveRootExternalFiles` eine Pflichtangabe. Ist in diesem Fall `archiveRootExternalFiles` nicht angegeben, können Sie SecDocs nicht starten.

Mit einem eigenen Wurzelverzeichnis zur Ablage externer Datenobjekte haben Sie die Möglichkeit, die beiden Datenbereiche für das SecDocs-Archiv und die externen Dateien zu trennen. Damit können Sie diese beiden Bereiche auf dem Storage-System unterschiedlich konfigurieren und somit Ihr Filesystem optimal dimensionieren.

So können Sie z.B. das mit `archiveRoot` ("[Konfigurationsdatei secdocs.properties](#)") festgelegte Verzeichnis zum Archivieren vieler kleiner Dokumente verwenden und den mit `archiveRootExternalFiles` bezeichneten Bereich zum Archivieren einer kleinen Anzahl großer Dokumente.

`signCubesHost`

IP-Adresse des Systems, auf dem die SecDocs Security-Komponente OpenLimit CryptoModule installiert ist.

i Derzeit ist hier nur die Angabe von `localhost` erlaubt.

Standardwert: `localhost` (127.0.0.1)

`signCubesPort`

Portnummer, die bei der Installation von OpenLimit CryptoModule angegeben wurde.
Standardwert: 18080

`msosLogThreshold`

Definiert den Schwellenwert für die Protokollierung der MSOS Komponenten. Erlaubte Werte sind: `ERROR,WARN,INFO,DEBUG,TRACE`
Standardwert: `INFO`

`avoidKeepReservedPeriod`

Zeitspanne in Minuten, nach der reservierte AOIDs ungültig werden. Eine AOID wird mit der Operation `getAOID` reserviert (siehe "[Operation getAOID](#)") und dann in der Operation `submitSDO` verwendet (siehe "[Operation submitSDO](#)"). Nach einer erfolgreichen Operation `getAOID` muss `submitSDO` innerhalb der hier angegebenen Zeitspanne aufgerufen werden.

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

Standardwert: 720

Minimum: 120

i Wenn Sie hier einen kleineren Wert als 120 angeben, setzt SecDocs den Wert auf 120 und schreibt eine Warnung in die Log-Datei.

`avoidWithRefKeepReservedPeriod`

Zeitspanne in Minuten, nach der eine mit der Operation `getAOIDWithRef` (siehe "[Operation getAOIDWithRef](#)") reservierte AOID ungültig wird. Nach einer erfolgreichen Operation `getAOIDWithRef` muss die Operation `submitSDO` (siehe "[Operation submitSDO](#)") mit dieser AOID innerhalb der hier angegebenen Zeitspanne aufgerufen und abgeschlossen werden.

Beachten Sie, dass in diesem Zeitrahmen auch die Übertragung aller referenzierten externen Dateien (die erst nach der Operation `getAOIDWithRef` beginnen kann) abgeschlossen sein muss.

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

Standardwert: 2880

Minimum: 120

i Wenn Sie hier einen kleineren Wert als 120 angeben, setzt SecDocs den Wert auf 120 und schreibt eine Warnung in die Log-Datei.

`externalFilesRetrieveTimeout`

Zeitspanne in Stunden, nach der ein durch die Operation `retrieveSDO` (siehe "[Operation retrieveSDO](#)") angelegter symbolischer Link durch SecDocs gelöscht wird.

Mit diesem Konfigurationsparameter können Sie einem Ressourcenengpass auf dem Storage-System (im mandanten- und organisationsspezifischen Übergabebereich) entgegenwirken.

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

Standardwert: 12

Minimum: 1

i Wenn Sie hier einen kleineren Wert als 1 angeben, setzt SecDocs den Wert auf 1 und schreibt eine Warnung in die Log-Datei.

`maxTreeSize`

Maximalwert für die Größe eines Hash-Baums. Dieser Wert legt fest, nach wie vielen unversiegelt gespeicherten SDOs die Versiegelung angestoßen wird. Wenn dieser Wert erreicht ist, beginnt SecDocs auf jeden Fall mit dem Versiegeln.

`maxTreeSize` bestimmt den Standardwert für den Parameter `TreeSize` in den Operationen `createMandant` (siehe "[Operation createMandant](#)") bzw. `updateMandant` ("[Operation updateMandant](#)").

Standardwert: 512

`maxParallelHashtrees`

Gibt an, wie viele Hash-Bäume maximal pro Server/Knoten parallel versiegelt werden können.

Standardwert: 1

i Höhere Werte sollten Sie nur abhängig von den verfügbaren Prozessoren verwenden, da das Versiegeln sehr rechenintensiv ist.
Beachten Sie auch, dass sich während des Versiegelns auch die Antwortzeiten bei den Operationen `submitSDO` bzw. `replaceSDO` erhöhen können.

`maxParallelHashtreesTimeStamping`

Regelt zusammen mit dem Parameter `maxParallelHashtreesRenew` den Anteil der Versiegelungsaufträge, die von einem Überwachungsprozess oder explizit mit der Operation `performAction` gestartet wurden, an der Menge aller parallel ablaufenden Versiegelungsprozesse. Der Anteil der genannten Prozesse an der Gesamtanzahl beträgt ungefähr:

$$\text{maxParallelHashtreesTimeStamping}$$
$$\text{maxParallelHashtreesTimeStamping} + \text{maxParallelHashtreesRenew}$$

Außerdem legt der Parameter fest, wie viele Versiegelungsaufträge, die von einem Überwachungsprozess oder explizit mit der Operation `performAction` gestartet wurden, pro Mandant und Server/Knoten maximal parallel ablaufen können.
Wird der Wert auf 0 gesetzt, werden wartende Versiegelungsaufträge, die von einem Überwachungsprozess oder explizit mit der Operation `performAction` gestartet wurden, nicht gestartet.

Wird `maxParallelHashtreesTimeStamping > maxParallelHashTrees` gesetzt, dann setzt SecDocs den Wert von `maxParallelHashtreesTimeStamping` beim starten gleich dem Wert von `maxParallelHashTrees`.

Standardwert: Wert des Konfigurationsparameters `maxParallelHashtrees`.

`maxParallelHashtreesRenew`

Regelt zusammen mit dem Parameter `maxParallelHashtreesTimeStamping` den Anteil der Versiegelungsaufträge, die von einer der Operationen `renew...` gestartet wurden, an der Menge aller parallel ablaufenden Versiegelungsprozesse. Der Anteil der genannten Prozesse an der Gesamtanzahl beträgt ungefähr:

$\frac{\text{maxParallelHashtreesRenew}}{\text{maxParallelHashtreesTimeStamping} + \text{maxParallelHashtreesRenew}}$

Außerdem legt der Parameter fest, wie viele Versiegelungsaufträge, die von einer der Operationen `renew...` gestartet wurden, pro Mandant und Server/Knoten maximal parallel ablaufen können. Wird der Wert auf 0 gesetzt, werden wartende Versiegelungsaufträge, die von einer der Operation `renew...` gestartet wurden, nicht gestartet.

Wird `maxParallelHashtreesRenew > maxParallelHashTrees` gesetzt, dann setzt SecDocs den Wert von `maxParallelHashtreesRenew` beim starten gleich dem Wert von `maxParallelHashTrees`.

Standardwert: Wert des Konfigurationsparameters `maxParallelHashtrees`.

`maxTransientErrTry`

maximal mögliche Anzahl von Wiederholungen der OpenLimit CryptoModule Server-Aufrufe.
Standardwert: 5

`waitBetweenTry`

Zeit in Millisekunden, die zwischen den Wiederholungen gewartet werden soll.
Standardwert: 500

`msDefaultConnectTimeout`

`osDefaultConnectTimeout`

Maximale Wartezeit für den Aufbau einer internen Socket-Verbindung innerhalb der SecDocs CryptoModule-Komponenten (in Millisekunden).

Sie sollten den Wert für `msDefaultConnectTimeout/osDefaultConnectTimeout` nur dann erhöhen, wenn Sie bei fehlgeschlagenen Operationen `submitSDO` (siehe "[Operation retrieveSDO](#)") in den Logging-Dateien der SecDocs CryptoModule-Komponenten Einträge finden, die auf das Auftreten eines Connection Timeout hinweisen.

Standardwert: 10000

Minimum: 100

i Wenn Sie hier einen kleineren Wert als 100 angeben, setzt SecDocs den Wert auf 100 und schreibt eine Warnung in die Log-Datei.

`msDefaultReceiveTimeout`

`osDefaultReceiveTimeout`

Zeitspanne in Millisekunden, innerhalb der bei einem internen SOAP-Request der SecDocs CryptoModule-Komponenten eine Rückmeldung erfolgen muss.

Sie sollten den Wert für `msDefaultReceiveTimeout/osDefaultReceiveTimeout` nur dann erhöhen, wenn Sie bei fehlgeschlagenen Operationen `submitSDO` (siehe "[Operation retrieveSDO](#)") in den Logging-Dateien der SecDocs CryptoModule-Komponenten Einträge finden, die auf das Auftreten eines Receive Timeout hinweisen.

Standardwert: 60000

Minimum: 0 (= unbegrenzte Wartezeit)

i Wenn Sie hier einen kleineren Wert als 0 angeben, setzt SecDocs den Wert auf 0 und schreibt eine Warnung in die Log-Datei.

`maxSoapRequestSize`

maximale Größe eines SOAP-Requests, die SecDocs verarbeiten kann. Beachten Sie, dass der SOAP-Request, mit dem ein SDO archiviert werden soll, ca. 1,5-mal so groß ist wie das ursprüngliche Primärdokument, siehe auch [Abschnitt „Operation submitSDO“](#).

Mögliche Werte:

gb	Wert in Gigabytes
mb	Wert in Megabytes
kb	Wert in Kilobytes

keine Wert in Bytes
Angabe
einer Einheit

Bei den Einheiten ist die Groß-/Kleinschreibung nicht relevant. Zwischen Wertangabe und Einheit können ein oder mehrere Leerzeichen stehen.

Beispiele: 1 gb, 1GB, 200 MB, 200mb, 300000Kb, 300000 kB, 400000000

Standardwert: 100mb (100 Megabytes)

i Erhöhen Sie diese Einstellung nur nach Rücksprache mit Ihrem technischen Betreuer von Fujitsu. SecDocs erzeugt zur Archivierung intern Objekte, deren Größe nicht nur durch die SDO-Größe, sondern auch durch die Anzahl der darin enthaltenen Dokumentensignaturen wesentlich beeinflusst werden. Für die Ermittlung der Maximalgröße eines SDOs sind auch der Speicherausbau und die Größe dieser internen Objekte zu berücksichtigen.

`createMandantDirLocal`

gibt an, ob SecDocs das Verzeichnis automatisch erzeugen soll, das im Parameter `Path` der Operation `createMandant` angegeben wird (siehe "[Operation createMandant](#)"). Dieser Parameter beschreibt den Pfad für den Teilbereich des Mandanten im Archiv. Es kann auf einen Mount Bezug genommen werden.

Mögliche Werte:

"true" SecDocs erzeugt bei der Operation `createMandant` das Verzeichnis für den Archivbereich des Mandanten automatisch (falls es nicht bereits existiert).

alle Alle anderen Werte werden auf "false" abgebildet:

anderen Die Pfade `archiveRoot/path` und `archiveRootExternalFiles/path` zum
Werte Archivbereich des Mandanten müssen zum Zeitpunkt der Operation `createMandant` mit Schreibrecht zugreifbar sein.
Der Systemadministrator muss vorab die Verzeichnisse einrichten und gegebenenfalls Mount-Einträge für die entsprechenden Volumes erstellen und die Mounts aktivieren (siehe [Abschnitt „Storage-System anbinden“](#)).

Standardwert: `false`;

stellt sicher, dass ein Mandant nicht eingerichtet werden kann, wenn das entsprechende Mandantenverzeichnis nicht eingerichtet ist.

`checkTSP`

gibt an, ob SecDocs bei der Operation `createTSP` automatisch die Verbindung zum angegebenen TSP überprüfen soll (siehe "[Operation createTSP](#)").

Mögliche Werte:

- "true" SecDocs versucht, einen Zeitstempel vom angegebenen TSP zu holen. Wenn kein Zeitstempel bezogen werden kann, schlägt die Operation `createTSP` fehl.
- "false" SecDocs überprüft die Verbindung zum angegebenen TSP nicht.

Standardwert: `true`

`cryptoAuditFiles`

Liste von Dateien, in die die zertifizierte OpenLimit-Komponente von SecDocs Audit-Einträge schreiben kann. Die Liste muss mindestens einen Eintrag enthalten. Die OpenLimit-Komponente schreibt diese Einträge in die erste Datei aus der Liste, die schreibbar ist. Stellt die OpenLimit-Komponente während des Betriebes fest, dass die gerade verwendete Datei nicht mehr schreibbar ist, so wird die nächste schreibbare Datei in der Liste verwendet. Ist keine der Dateien schreibbar, so stellt die OpenLimit-Komponente die Arbeit ein und alle Anfragen an den Web-Service `ArchivingService` bzw. `S4` werden abgewiesen.

Um sicherzustellen, dass die OpenLimit-Komponente diese Logging-Einträge immer schreiben kann, sollten folgende Bedingungen erfüllt sein:

- `cryptoAuditFiles` sollte mindestens zwei, besser drei Dateinamen enthalten.
- Diese Dateien sollten auf verschiedenen Dateisystemen und/oder Volumes im Filer liegen.

Mögliche Werte:

Liste von Dateinamen durch Semikolon getrennt. Werden relative Dateinamen angegeben, so werden die Dateien relativ zum Home-Verzeichnis der WildFly SecDocs-Server-Instanz abgelegt.

i `cryptoAuditFiles` ist eine Pflichtangabe. Wenn Sie `cryptoAuditFiles` nicht setzen oder keine der dort angegebenen Dateien schreibbar ist, startet SecDocs nicht.

Wird SecDocs in einer Multi-Node-Konfiguration betrieben, so ergänzt SecDocs die Dateinamen mit dem Servernamen der lokalen SecDocs-Instanz: `dateiname_hostname.log`. Damit bleiben die Dateien auch dann unterscheidbar, wenn sie für verschiedene Server im selben Verzeichnis des Shared Storage abgelegt werden.

Im Single-Node-Betrieb bleiben die Dateinamen unverändert.

`maxHashDataObjectSize`

Legt die maximal zulässige Größe des Hash Data Object (HDO) fest. Das HDO setzt sich zusammen aus dem SDO und den Verifikationsprotokollen der Signaturen, die in diesem SDO enthalten und geprüft worden sind.

Mögliche Werte:

gb	Wert in Gigabytes
mb	Wert in Megabytes
kb	Wert in Kilobytes
keine Angabe einer Einheit	Wert in Bytes

Bei den Einheiten ist die Groß-/Kleinschreibung nicht relevant. Zwischen Wertangabe und Einheit können ein oder mehrere Leerzeichen stehen.

Beispiele: 1 gb, 1GB, 200 MB, 200mb, 300000Kb, 300000 kB, 4000000000

Standardwert: 1500 MByte (1500 Megabytes)

Diese Voreinstellung können Sie jederzeit verkleinern, erhöhen sollten Sie sie aber nur nach Rücksprache mit Ihrem technischen Betreuer von Fujitsu.

`checkHashAtRetrieve`

Legt fest, ob SecDocs bei der Bearbeitung der Operation `retrieveSDO` (siehe "[Operation retrieveSDO](#)") die Integrität des SDOs und der zugehörigen Signature Verification Information (SVI) automatisch im Rahmen der Operation überprüft. Andernfalls müssten Sie diese Überprüfung mit einem nachfolgenden Aufruf der Operation `requestForEvidence` (siehe "[Operation requestForEvidence](#)") durchführen. Siehe hierzu auch Attribut `checkHash` bei der Operation `retrieveSDO` im [Abschnitt „Operation retrieveSDO“](#).

Unabhängig von der Einstellung von `checkHashAtRetrieve` führt SecDocs bei `retrieveSDO` keine Integritätsüberprüfung für externe Datenobjekte durch.

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

Mögliche Werte (Groß-/Klein-Schreibung ist nicht relevant):

"true"	Die Integrität des SDO und der zugehörigen SVI wird von SecDocs bei der Operation <code>retrieveSDO</code> automatisch überprüft.
"false"	Die Integrität des SDO und der zugehörigen SVI wird von SecDocs bei der Operation <code>retrieveSDO</code> nicht überprüft.

Standardwert: `true`

maxAuditLogFileSize

Legt die maximale Größe für eine Audit-Log-Datei fest. Wenn das Schreiben eines Audit-Logs diese Größe überschreiten würde, wird die Audit-Log-Datei geschlossen und ihr Dateiname mit Datum und Suffix versehen. Der Audit-Log wird dann in eine neue Audit-Log-Datei geschrieben.

`maxAuditLogFileSize` gilt sowohl für mandantenspezifische Audit-Log-Dateien wie für die Archiv-Audit-Log-Datei.

Mögliche Werte:

<code>mb</code>	Wert in Megabytes
<code>kb</code>	Wert in Kilobytes
<code>keine Angabe einer Einheit</code>	Wert in Bytes

Bei den Einheiten ist die Groß-/Kleinschreibung nicht relevant. Zwischen Wertangabe und Einheit können ein oder mehrere Leerzeichen stehen.

Beispiele: 200 MB, 200mb, 300000Kb, 300000 kB, 400000000

Standardwert: 500mb (500 Megabytes)

Minimalwert: 1mb

Maximalwert: 500mb

i Wenn Sie hier einen größeren Wert als 500 Megabytes angeben, setzt SecDocs den Wert auf 500mb und schreibt eine Warnung in die Log-Datei.

Wenn Sie hier einen kleineren Wert als 1 Megabyte angeben, setzt Sec-Docs den Wert auf 1mb und schreibt eine Warnung in die Log-Datei.

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

nodeNameInFaultMessage

Legt fest, ob SecDocs in folgenden Fällen zusätzlich das Element `nodeName` ausgibt, das den Namen des betroffenen Knotens (SecDocs-Instanz) enthält:

- in der Fault-Message (Knoten, auf dem der Fehler aufgetreten ist)
- in der SOAP-Response bei der Operation `getVersion` des Web-Service `ArchivingService` (Knoten, auf dem die Operation bearbeitet wurde).

Dies ist im Multi-Node-Betrieb hilfreich, wenn mit einem Load Balancer gearbeitet wird.

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

Mögliche Werte:

"false" SecDocs gibt das Element `nodeName` nicht aus.

"true" SecDocs gibt zusätzlich das Element `nodeName` aus.

Standardwert: `false`

`multiNode`

Legt fest, ob eine SecDocs-Instanz in einem Multi-Node-Verbund teilnehmen oder als einzelner SecDocs-Server betrieben werden soll.

Mögliche Werte:

"false" SecDocs wird im Single-Node-Modus betrieben

"true" SecDocs wird im Multi-Node-Modus betrieben.

Standardwert: `false`

i Bei `multiNode=true` muss auch `multiNode.hazelcastConfigFile` angegeben werden.

`multiNode.hazelcastConfigFile`

Vollständiger Pfadname der Datei `hazelcast.xml`.

Die Einträge in der Datei `hazelcast.xml` definieren, zu welchem Verbund diese SecDocs-Instanz gehören soll. Auf diese Weise kann z.B. zwischen Test- und Produktiv-Verbund im selben Netzsegment unterschieden werden. Näheres hierzu finden Sie in der Installationsanleitung ("[SecDocs V3.1A01 Installationsanleitung](#)" ([SD3] in Abschnitt "[Literatur](#)").

i Wird SecDocs in einer Single-Node-Konfiguration betrieben, so wird dieser Parameter ignoriert.

jobRestartAllowed

Legt fest, ob SecDocs im Multi-Node-Betrieb die Versiegelung und Signaturerneuerung auch dann fortsetzt, wenn ein Knoten heruntergefahren wird oder abnormal beendet wurde.

i Wird SecDocs in einer Single-Node-Konfiguration betrieben, so wird dieser Parameter ignoriert.

Mögliche Werte:

- | | |
|---------|--|
| "true" | Wenn SecDocs im Multi-Node-Betrieb abläuft und ein Knoten heruntergefahren wird, prüft SecDocs für jeden Mandanten, ob der betreffende Knoten der einzige ist, der Aufträge zur Versiegelung und Signaturerneuerung ausführt. In diesem Fall wählt SecDocs einen anderen Knoten im Verbund und startet die Aufträge auf diesem wieder. |
| "false" | Wenn SecDocs im Multi-Node-Betrieb abläuft und ein Knoten heruntergefahren wird, werden alle Aufträge zur Versiegelung und Signaturerneuerung für den betreffenden Knoten abgebrochen. |

Standardwert: `false`

globalPropertiesFile

Gibt den Pfadnamen der sekundären Konfigurationsdatei `secdocs.properties` an, deren Parameter zur Konfiguration herangezogen werden sollen.

Ist der Zugriff auf diese Datei nicht möglich, wird der Start von SecDocs abgebrochen.

Die Datei `installation-dir/jboss/secdocs/configuration/secdocs/secdocs.properties` wird im Gegensatz zu der in ihr angegebenen sekundären Konfigurationsdatei als primäre Konfigurationsdatei bezeichnet;

Die Angaben in der primären Konfigurationsdatei `secdocs.properties` haben Vorrang vor den Angaben in der sekundären Datei. Ist ein Parameter also in beiden Konfigurationsdateien `secdocs.properties` vorhanden, so überschreibt die Angabe in der primären Datei diejenige in der sekundären Datei.

Der Parameter `globalPropertiesFile` dient insbesondere im Multi-Node-Betrieb dazu, alle Konfigurationsparameter in einer einzigen Konfigurationsdatei für den gesamten Verbund (also eine globale `secdocs.properties`) zu pflegen. Auf diese Weise ist eine einheitliche Konfiguration für alle beteiligten SecDocs-Instanzen gewährleistet.

`createAuditLogEvidenceRecord`

Legt fest, ob beim Anlegen einer neuen mandantenspezifischen Audit-Log-Datei für die alte Audit-Log-Datei zur Beweiswerterhaltung ein Evidence Record erzeugt werden soll.

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

Mögliche Werte:

- "false" Für die mandantenspezifischen Audit-Log-Dateien werden keine Evidence Records erzeugt.
- "true" Für die mandantenspezifischen Audit-Log-Dateien werden Evidence Records erzeugt.

Standardwert: `false`

Wird ein Wert ungleich `"true"` angegeben (wobei die Groß-/Kleinschreibung ignoriert wird), so gilt die Einstellung `"false"`.

`collectOperationStatistics`

Legt fest, ob in SecDocs während des laufenden Betriebs Statistikdaten erhoben werden sollen. Näheres hierzu finden Sie im [Abschnitt „Erheben von Statistikdaten“](#).

Dieser Wert kann auch mandantenspezifisch eingestellt werden (siehe "[Mandantenspezifische Konfigurationsparameter](#)").

Mögliche Werte:

- "false" SecDocs erhebt keine Statistikdaten.
- "true" SecDocs führt eine Erhebung von Statistikdaten durch.

Standardwert: `false`

Wird ein Wert ungleich `"true"` angegeben (wobei die Groß-/Kleinschreibung ignoriert wird), so gilt die Einstellung `"false"`.

i Wird `collectOperationStatistics` für einen Mandanten mit der Operation `updateMandant` ("[Operation updateMandant](#)") auf den Wert `"false"` gesetzt, so werden alle für diesen Mandanten bisher aufgesammelten Daten gelöscht.

`createSftpServer`

Legt fest, ob der integrierte SFTP-Server zusammen mit SecDocs gestartet wird. Nur in diesem Fall ist eine Archivierung externer Dokumente möglich.

Mögliche Werte (Groß-/Kleinschreibung nicht relevant):

"false" Der SFTP-Server wird nicht gestartet. Es erfolgt nur ein Start von SecDocs.

"true" Der SFTP-Server wird zusammen mit SecDocs gestartet.

Standardwert: `false`

i Wenn Sie den Parameter `createSftpServer` auf "true" setzen, so müssen Sie auch den Parameter `archiveRootExternalFiles` ("[Konfigurationsdatei secdocs.properties](#)") angeben. Ist `archiveRootExternalFiles` in diesem Fall nicht gesetzt, können Sie SecDocs nicht starten.

`sftpTcpPort`

TCP/IP-Port, über den die Kommunikation des SFTP-Servers mit den Client-Anwendungen abgewickelt wird.

Standardwert: `2121`

`reverseProxy`

Wird dieser Parameter auf "true" gesetzt werden die weiteren Reverse Proxy Parameter ausgewertet.

"false" Reverse Proxy Nutzung in SecDocs wird deaktiviert.

"true" Reverse Proxy Nutzung in SecDocs wird aktiviert. Alle weiteren "reverseProxy.*" werden ausgewertet.

Standardwert: `false`

`reverseProxy.clientCertificateHeaderVariable`

Der Name des zu verwendenden HTTP Header Elements.

Standardwert: X-SSL-CLIENT-CERT

`reverseProxy.address.[1-10]`

Hier können bis zu 10 Adressen hinterlegt werden. Da es sich um verschiedene Parameter handelt (jede hat ihren eigenen Namen), muss pro Zeile ein Parameter gesetzt werden.

i Die Nummern (1, ..., 10) müssen aufsteigend und lückenlos angegeben werden. Das heißt, werden die Parameter `reverseProxy.address.1` und `reverseProxy.address.7` verwendet, wird nur der Parameter `reverseProxy.address.1` ausgewertet.

8.1.4 Mandantenspezifische Konfigurationsparameter

Seit SecDocs V2.3 können Sie einige der Konfigurationsparameter auch mandantenspezifisch einstellen. Diese mandantenspezifischen Einstellungen werden nicht in der Konfigurationsdatei `secdocs.properties` hinterlegt. Sie werden vom Archivadministrator mit der Operation `createMandant` ("[Operation createMandant](#)") bzw. vom Mandantenadministrator mit der Operation `updateMandant` ("[Operation updateMandant](#)") festgelegt.

Im Unterschied zu den Einstellungen in der Konfigurationsdatei `secdocs.properties`, die ausschließlich beim Start von SecDocs ausgewertet werden, werden mandantenspezifische Einstellungen sofort nach der Änderung wirksam. Die Änderung ist permanent, d.h. sie bleibt bei einem Herunterfahren und Wiederanlaufen von SecDocs erhalten. Läuft SecDocs im Multi-Node-Betrieb, wird die Änderung auf allen Knoten des Verbunds wirksam.

Die Einstellungen in der Konfigurationsdatei `secdocs.properties` gelten für alle Mandanten, für die dieser Parameter nicht mandantenspezifisch eingestellt ist.

Als Wert eines Konfigurationsparameters übernimmt SecDocs bei der Bearbeitung einer Web-Service-Operation den ersten nach folgender Suchreihenfolge gefundenen Wert:

1. Mandantenspezifischer Wert des Parameters für den entsprechenden Mandanten
2. Wert des Parameters in der primären Konfigurationsdatei, d.h. der server-lokalen `secdocs.properties` des Servers, auf dem die Operation bearbeitet wird.
3. Wert des Parameters in der sekundären Konfigurationsdatei, d.h. der server-globalen `secdocs.properties` des Verbunds, falls SecDocs im Multi-Node-Betrieb läuft
4. Standardwert des Parameters

Bei Migration einer älteren SecDocs-Version auf SecDocs V3.1 werden für existierende Mandanten keine mandantenspezifischen Einstellungen vorgenommen.

Die folgenden Konfigurationsparameter können mandantenspezifisch eingestellt werden:

Konfigurationsparameter	Standardwert	Minimal- /Maximalwert	Seite
<code>aoidKeepReservedPeriod</code>	720 Minuten	min. 120 Minuten	" Konfigurationsdatei secdocs.properties "
<code>aoidWithRefKeepReservedPeriod</code>	2880 Minuten	min. 120 Minuten	" Konfigurationsdatei secdocs.properties "
<code>externalFilesRetrieveTimeout</code>	12 Stunden	min. 1 Stunde	" Konfigurationsdatei secdocs.properties "
<code>maxAuditLogFileSize</code>	500 Megabytes	min. 1 Megabyte max. 500 Megabytes	" Konfigurationsdatei secdocs.properties "
<code>nodeNameInFaultMessage</code>	false		" Konfigurationsdatei secdocs.properties "
<code>createAuditLogEvidenceRecord</code>	false		" Konfigurationsdatei secdocs.properties "

collectOperationStatistics	false		"Konfigurationsdatei secdocs. properties"
checkHashAtRetrieve	true		"Konfigurationsdatei secdocs. properties"

Tabelle 5: Mandantenspezifisch einstellbare Konfigurationsparameter

8.2 Storage-System anbinden

Um SecDocs in Betrieb nehmen zu können, müssen Sie das Storage-System anbinden. Hierzu müssen Sie folgende Schritte durchführen:

- Volume auf Storage-System erzeugen
- Volume in Linux einbinden
- Für jeden Mandanten ein eigenes Volume einrichten (bei Bedarf)

Wenn Sie externe Datenobjekte archivieren wollen und dafür eine eigene Archiv-Wurzel vorsehen, müssen Sie voranstehenden Schritte zusätzlich auch für diese Archiv-Wurzel durchführen.

Volume auf Storage-System erzeugen

i Verfahren Sie hier nach den Vorgaben der Storage-spezifischen Dokumentation.

- > Beispiel (ONTAP-Kommando für NetApp®-Filer):

```
vol create SecDocsDemo -l de.UTF-8 aggr1 100g
```

- > Geben Sie bei NFS Exports die Root-Berechtigung für das SecDocs-System frei, um den Eigentümer ändern zu können.

Volume in Linux einbinden (mit Root-Berechtigung)

- > Erzeugen Sie den Mountpoint für die Archiv-Wurzel in Linux.

Beispiel: `mkdir /secdocsArchiveRoot`

- > Tragen Sie einen NFS3-Mount in der Datei `/etc/fstab` ein.

i Der nachfolgende Eintrag muss als eine Zeile geschrieben werden.

```
filename:/vol/SecDocsDemo /secdocsArchiveRoot nfs rw,nodev,auto,noexec,  
timeo=600,tcp,vers=3,rsize=32768,wsiz=32768,hard,bg,retry=100 0 0
```

- > Mounten Sie das Volume bzw. starten Sie den Server neu.

```
# mount /secdocsArchiveRoot
```

- > Stellen Sie Benutzer und Gruppe `secdocs` ein.

```
# chown secdocs:secdocs /secdocsArchiveRoot
```

Für jeden Mandanten ein eigenes Volume einrichten

Wenn für einen Mandanten ein eigenes Volume im Archiv verwendet werden soll, muss der SecDocs-Systemadministrator dies vorbereiten.

- > Erzeugen Sie ein entsprechendes Volume auf dem Storage-System.
- > Richten Sie direkt unterhalb des SecDocs-Wurzelverzeichnisses im Linux einen Mountpoint für das Volume ein (`mkdir /secdocsArchiveRoot/myMandant`).
- > Erzeugen Sie einen entsprechenden Mount-Eintrag in der Datei `/etc/fstab`. Dafür ist Root-Berechtigung erforderlich.
- > Aktivieren Sie mit `mount` den Mount.
- > Stellen Sie Schreibrecht für den SecDocs-Benutzer ein.

Jetzt kann der Pfad bei der Operation `createMandant` verwendet werden. Dabei muss der Parameter `createMandantDirLocal` in der Konfigurationsdatei `secdocs.properties` auf dem Standardwert `false` stehen, siehe auch "[Konfigurationsdatei secdocs.properties](#)".

Beispiel für den Eintrag in der Datei `fstab` (alles in einer Zeile):

```
storage:/vol/myMandantVol /secdocsArchiveRoot/myMandant nfs rw,nodev,auto,noexec,timeo=600,
tcp,vers=3,rsize=32768,wsiz=32768,hard,bg,retry=100 0 0
```

Dabei gilt:

`storage`

Name des Storage-Systems

`/vol/myMandantVol`

Name des Mandanten-spezifischen Volumes

`/secdocsArchiveRoot/myMandant`

Mandanten-spezifischer Mountpoint

8.3 Accounting

Das Accounting stellt Ihnen Informationen für folgende Zwecke bereit:

- Abrechnung von Fujitsu mit dem SecDocs-Betreiber
- Abrechnung des SecDocs-Betreibers mit seinen Mandanten
- Information des SecDocs-Betreibers über den Verbrauch von Zeitstempeln
- Erstellung von Statistiken

Zu diesem Zweck können verschiedene Daten entweder pro Mandant bzw. pro Organisation oder aber für das gesamte Archiv aufsummiert werden.

Für das Accounting steht die Operation `getAccountingData` zur Verfügung (siehe "[Operation getAccountingData](#)" (für Archivadministratoren) und "[Operation getAccountingData](#)" (für Mandantenadministratoren)). Damit sind folgende Daten abrufbar:

- Archivgröße

Die Archivgröße ist die Anzahl der insgesamt im Archiv existierenden versiegelten Dokumente. Diese Anzahl kann im Lauf der Zeit zu- oder abnehmen, abhängig davon, wie viele Dokumente ins Archiv eingebracht bzw. gelöscht werden.

SecDocs liefert die Archivgröße zum Zeitpunkt der Abfrage. Eine Web-Service-Anwendung kann dann selbst die Differenz zwischen zwei beliebigen Zeitpunkten ermitteln.

Die Information über die Archivgröße ist der Rolle `ArchiveAdmin` und in eingeschränktem Maß auch den Rollen `MandantAdmin` und `Archivar` zugänglich.

- Anzahl der bisher durchgeführten Versiegelungsvorgänge

Diese Anzahl wird von SecDocs ab dem ersten Einsatz der Version V2.0 oder einer höheren SecDocs-Version beim Betreiber kontinuierlich aufsummiert, d.h. der Wert nimmt im Lauf der Zeit stetig zu. Dieser Wert bleibt auch bei einem Herunterfahren und Wiederanlaufen von SecDocs und auch nach dem Einsatz einer neuen SecDocs-Version erhalten.

In diese Zahl geht neben der Archivgröße auch die Anzahl der versiegelten Dokumente ein, die wieder aus dem Archiv gelöscht wurden.

SecDocs liefert die Anzahl der bisher durchgeführten Versiegelungsvorgänge zum Zeitpunkt der Abfrage. Eine Web-Service-Anwendung kann dann selbst die Differenz zwischen zwei beliebigen Zeitpunkten ermitteln.

Die Information über die Anzahl der bisher durchgeführten Versiegelungsvorgänge ist der Rolle `ArchiveAdmin` und in eingeschränktem Maß auch den Rollen `MandantAdmin` und `Archivar` zugänglich.

-
- Anzahl der verbrauchten Zeitstempel bei einem Zeitstempelanbieter (TSP)

Diese Größe ermöglicht dem SecDocs-Betreiber eine Überwachung seines Zeitstempel-Kontingents bei einem bestimmten Zeitstempelanbieter (TSP = **T**ime **S**tamp **P**rovi-der).

Diese Anzahl wird von SecDocs ab dem ersten Einsatz der Version V2.0 oder einer höheren SecDocs-Version beim Betreiber kontinuierlich aufsummiert, d.h. der Wert nimmt im Lauf der Zeit stetig zu. Dieser Wert bleibt auch bei einem Herunterfahren und Wiederanlaufen von SecDocs und auch nach dem Einsatz einer neuen SecDocs-Version erhalten.

Zu beachten ist hierbei, dass Fehlversuche bei der Versiegelung mit einem Zeitstempel nicht mitgezählt werden. Bei dem von SecDocs gelieferten Wert handelt es sich also um die Anzahl der Zeitstempel, die mindestens verbraucht wurden. Der Wert kann tatsächlich aber höher liegen.

SecDocs liefert die Anzahl der verbrauchten Zeitstempel zum Zeitpunkt der Abfrage. Eine Web-Service-Anwendung kann dann selbst die Differenz zwischen zwei beliebigen Zeitpunkten ermitteln.

Die Information über die Anzahl der verbrauchten Zeitstempel ist nur der Rolle `ArchiveAdmin` zugänglich.

- Summe aller Versiegelungsvorgänge für sämtliche Testmandanten des Archivs

Wenn für einen Testmandanten alle AOIDs sowie der Testmandant selbst gelöscht werden, wird der bisher aufgelaufene Wert der von diesem Testmandanten durchgeführten Versiegelungsvorgänge aufgehoben.

SecDocs merkt sich den Gesamtwert für das ganze Archiv, d.h. die Summe aller Versiegelungsvorgänge, die für sämtliche Testmandanten des Archivs bis zum jeweiligen Zeitpunkt der Löschung des Testmandanten bzw. der Löschung aller AOIDs des Testmandanten durchgeführt wurden (ausführlichere Informationen über Testmandanten finden Sie im [Abschnitt „Testmandant“](#)).

Die Summe aller Versiegelungsvorgänge für sämtliche Testmandanten des Archivs wird von SecDocs ab dem ersten Einsatz der Version V2.0 oder einer höheren Sec-Docs-Version beim Betreiber kontinuierlich aufsummiert, d.h. der Wert nimmt im Lauf der Zeit stetig zu. Dieser Wert bleibt auch bei einem Herunterfahren und Wiederanlaufen von SecDocs und auch nach dem Einsatz einer neuen SecDocs-Version erhalten.

SecDocs liefert die Summe aller Versiegelungsvorgänge für sämtliche Testmandanten des Archivs zum Zeitpunkt der Abfrage. Eine Web-Service-Anwendung kann dann selbst die Differenz zwischen zwei beliebigen Zeitpunkten ermitteln.

Die Information über die Summe der Versiegelungsvorgänge für die Testmandanten ist nur der Rolle `ArchiveAdmin` zugänglich.

8.4 Erheben von Statistikdaten

Mit dem Erheben von Statistikdaten ermöglicht SecDocs ein Monitoring des laufenden SecDocs-Betriebs.

Diese Funktion kann sowohl global als auch mandantenspezifisch ein- oder ausgeschaltet werden. Die globale Einstellung erfolgt mit dem Parameter `collectOperationStatistics` in der Konfigurationsdatei `secdocs.properties`, siehe "[Konfigurationsdatei secdocs.properties](#)". Mandantenspezifische Einstellungen müssen vom Archivadministrator mit der Operation `createMandant` ("[Operation createMandant](#)") bzw. vom Mandantenadministrator mit der Operation `updateMandant` ("[Operation updateMandant](#)") vorgenommen werden.

Bei Auslieferung ist die Funktion ausgeschaltet.

Für alle Operationen der Web-Services `ArchivingService` und `S4` werden folgende Daten erfasst:

- Operationszähler, wie z.B. Anzahl von Requests, Größe von zu archivierenden Dokumenten, usw.
- Elapsed time von Operationen (Minimum, Maximum, Durchschnittswert)
- Zähler für bestimmte Fehlersituationen (z.B. abgewiesene `submitSDO`)

Alle Daten werden pro Mandant und pro Knoten erhoben. Für eine Gesamtschau über alle Knoten muss der Archivadministrator daher die Auswertungen für die einzelnen Knoten zusammenführen.

Der Archivadministrator kann bei jedem Auslesen der Messwerte festlegen, ob diese nach dem Auslesen zurückgesetzt werden sollen, siehe „[Operation getStatisticalData](#)“. Auf diese Weise werden die Zeitintervalle definiert, auf die sich die Statistikdaten beziehen.

Bei einem Neustart von SecDocs nach einem Herunterfahren oder einem Absturz auf einem Knoten werden die Statistikdaten auf diesem Knoten zurückgesetzt.

Bei einer Storage-Recovery werden die Statistikdaten nicht mit übernommen.

8.4.1 Messwerte

Für jeden Operationsnamen der Web-Services `ArchivingService` und `S4` werden folgende Messgrößen ermittelt:

- Gesamte Anzahl aller Aufrufe
- Anzahl erfolgreicher Aufrufe
- Anzahl fehlgeschlagener Aufrufe
- Dauer der schnellsten Operation
- Dauer der langsamsten Operation
- Durchschnittliche Dauer der Operation
- Bei einer erfolgreichen Bearbeitung der Operation `submitSDO` bzw. `replaceSDO` oder `ArchiveSubmission` zusätzlich:
 - Minimalwert, d.h. Größe des kleinsten SDOs bzw. XAIP-Datenobjekts in KB
 - Maximalwert, d.h. Größe des größten SDOs bzw. XAIP-Datenobjekts in KB
 - Durchschnittliche Größe eines SDOs bzw. XAIP-Datenobjekts in KB
 - Gesamtgröße aller SDOs, d.h. Summe der SDO-Größen aller erfolgreichen Operationen `submitSDO` bzw. `replaceSDO` in KB oder Summe der Größen aller XAIP-Datenobjekte bei der Operation `ArchiveSubmission`

Außerdem wird noch die Anzahl der Operationen ermittelt, die nicht im Funktionsumfang der im Request angegebenen Rolle enthalten sind.

Hinweise:

- Die Dauer einer Operation ist die Zeit vom Beginn bis zum Ende der Bearbeitung in SecDocs auf dem Server, d. h. die Zeit vom Eintreffen der ersten Bytes des Requests bei SecDocs bis zum Senden des letzten Bytes der Response.

Der HTTP-Vorlauf, bei dem schon der HTTP-Requestheader gelesen wurde, und Netzlaufzeiten (Übertragung von Request und Response zum oder vom Server) werden dabei nicht berücksichtigt.

- Bei den Operationen `submitSDO`, `replaceSDO` und `ArchiveSubmission` wird die Versiegelung nicht in die Dauer der Operation einbezogen.
- Die Dauer wird nur für erfolgreich durchgeführte Operationen ermittelt.
- Bei der Erhebung der Statistikdaten werden auch in SecDocs nicht bekannte Mandanten berücksichtigt, wenn sie bei einer Operation angegeben werden.

Die Anzahl erfolgreicher Operationen ist für nicht vorhandene Mandanten natürlich immer 0, da Operationen mit einem unbekannten Mandanten abgewiesen werden.

8.5 Logging und Fehlerbehandlung

Dieser Abschnitt beschreibt die verschiedenen Möglichkeiten zur Protokollierung, die SecDocs bietet:

- Das Audit-Logging zeichnet jeden Aufruf einer WebService-Operation auf und ermöglicht es, jede Aktion einem Verantwortlichen zuzuordnen.
- Das SecDocs-Logging protokolliert die Aktionen in SecDocs und im WildFly Application Server.

Zusätzlich sollten Sie das Umfeld von SecDocs überwachen.

8.5.1 Audit-Logging

Ein Audit-Logging ermöglicht es, jede Aktion in einem IT-System einem Verantwortlichen zuzuordnen. Anders als bei Fehlerprotokollen steht die Aufzeichnung erfolgreicher Aktionen im Vordergrund. Es ermöglicht berechtigten Personen, die Abläufe im System nachträglich zu beurteilen (Revision). Für die Systemüberwachung selbst wird kein Audit-Logging erstellt.

SecDocs verwahrt Einträge im Audit-Logging dauerhaft. Neben dem Audit-Logging müssen Sie auch die Beschreibung des Aufzeichnungsformats (siehe "[Die Audit-Log-Datei](#)") aufbewahren, um auch nach längerer Zeit die Einträge noch korrekt interpretieren zu können.

SecDocs verfügt über

- eine Protokollierung der Aktionen der Archivadministration, die unabhängig von den Mandanten arbeitet. Diese Audit-Log-Datei ist abgelegt unter `archiveRoot/_config/_audit/audit_hostname.log`.
- ein mandantenspezifisches Audit-Logging: Es werden alle Archiv- und Verwaltungsaufträge aufgezeichnet, die von einem Mandantenadministrator angestoßen werden. Diese Aufträge werden im Dokumentenbereich des entsprechenden Mandanten protokolliert. Die Audit-Log-Datei eines Mandanten wird abgelegt unter `archiveRoot/mandantPath/_config/_audit/audit_hostname.log`.

Dabei ist `hostname` der Name des Servers, für den die Audit-Log-Datei angelegt wird. Auf diese Weise erhalten die Audit-Log-Dateien auch im Multi-Node-Betrieb eindeutige Dateinamen. Denn im Multi-Node-Betrieb werden für jeden Knoten im Serververbund eigene Audit-Log-Dateien in einem gemeinsamen Verzeichnis auf dem Shared Storage angelegt.

Darüber hinaus verfügt SecDocs auch über Audit-Log-Dateien der zertifizierten OpenLimit-Komponente. Name und Ablageort dieser Dateien werden in der Konfigurationsdatei `secdocs.properties` über den Parameter `cryptoAuditFiles` eingestellt (siehe "[Konfigurationsdatei secdocs.properties](#)").

Die mit dem Parameter `cryptoAuditFiles` angegebenen Dateinamen werden im Multi-Node-Betrieb mit dem Servernamen der lokalen SecDocs-Instanz ergänzt, damit sie eindeutig sind (`dateiname_hostname.log`). Im Single-Node-Betrieb bleiben diese Dateinamen unverändert.

In diesen Dateien werden neben allen Löschoperationen, bei denen das Ablaufdatum noch nicht erreicht ist, nur Fehler protokolliert, die von dieser Komponente festgestellt werden.

In den folgenden Abschnitten wird das Format der von SecDocs erstellten Einträge im Audit-Logging detailliert beschrieben.

8.5.1.1 Die Audit-Log-Datei

Die Audit-Log-Datei ist eine Text-Datei in UTF-8-Codierung.

Für jede Operation des Web-Services wird Folgendes eingetragen:

- 0 – n Zeilen durch Komponenten von CryptoModule.
- 1 Zeile durch SecDocs; die SecDocs-Zeile ist jeweils die letzte Zeile der Aktion.

Bei SecDocs gibt es folgende drei Arten von Einträgen:

INIT-Eintrag	wird sowohl beim Start des mandantenunabhängigen als auch jeweils des mandantenspezifischen Audit-Loggings von SecDocs geschrieben
STOP-Eintrag	wird beim Beenden des Audit-Loggings von SecDocs geschrieben
<i>operation-</i> Eintrag	wird bei einer Operation eines Archiv- oder Mandantenadministrators oder der Client-Software geschrieben

Bei den CryptoModule-Komponenten produziert jeder Aufruf einer Schnittstellenfunktion zwei Audit-Events:

- Der erste Eintrag gibt an, mit welchen Parametern die Funktion aufgerufen wurde nach Prüfung der Eingabeparameter auf Validität.
- Der zweite Audit-Eintrag gibt an, ob die Funktion erfolgreich durchgeführt oder abgebrochen wurde. Er enthält alle Eingabeparameter aus dem ersten Audit-Eintrag sowie zusätzlich eine Statusangabe.

Aufbau der Einträge

Der Aufbau der einzelnen Einträge in der Audit-Log-Datei orientiert sich an RFC 5424 (Syslog Protocol).

Jeder Logging-Eintrag hat einen Header mit einer Liste von Feldern, die bei jedem Eintrag vorhanden sind, gefolgt von strukturierten Daten (*STRUCTURED-DATA* in RFC 5424), die die protokollierte Aktion im Detail beschreiben. Den nach RFC 5424 zulässigen freien Text (*MSG* in RFC 5424) verwendet SecDocs nur in den INIT- und STOP-Einträgen.



- Eine detaillierte Beschreibung der Syntax-Elemente von RFC 5424 entnehmen Sie dem RFC 5424.
- Beispiele für Logging-Einträge in der Audit-Log-Datei finden Sie im Abschnitt "[Beispiele für Logging-Einträge in der Audit-Log-Datei](#)"

8.5.1.2 Header eines Audit-Logging-Eintrags

Jeder Logging-Eintrag beginnt mit den folgenden, jeweils durch ein Leerzeichen voneinander getrennten Feldern:

Feldinhalt	Bedeutung
<110>1	Konstant bei jeder Zeile der Audit-Log-Datei. Nach RFC 5424 charakterisiert diese Zeichenfolge den Eintrag als Ereignis der Syslog-Facility „log audit“ und der Syslog-Severity „info“, mit der Protokoll-Version 1.
Zeitstempel	in dem Format, das durch RFC 3339 vorgegeben ist
Hostname	Identifikation des sendenden Systems (IP-Adresse)
SecDocs	Konstant in jeder Zeile. Name der Anwendung
PROCID	Prozess-Identifikation des sendenden Prozesses
Msgid	Name der Operation, die protokolliert wird, in abdruckbarer Form; Bei den Version 3 Server-Einträgen handelt es sich um Operationen der OpenLimit-Komponenten.

Beispiel

<110>1	2015-02-12T15:15:09.905+02:00	10.172.105.13	SecDocs	9596	createOrganisation
--------	-------------------------------	---------------	---------	------	--------------------

8.5.1.3 Strukturierte Daten eines Audit-Logging-Eintrags


Vom Header durch ein Leerzeichen getrennt folgen die strukturierten Daten, die das Ereignis beschreiben.

Die strukturierten Daten bestehen aus einer Liste von Elementen (`SD-ELEMENT` in RFC 5424), die jeweils in eckige Klammern (`[]`) eingeschlossen sind.

Jedes Element enthält innerhalb der eckigen Klammern zuerst einen Element-Namen (`SD-NAME` in RFC 5424), gefolgt von einer Liste von Parametern als Paar aus Schlüsselwort und "Wert" (`SD-PARAM` in RFC 5424). Die Werte sind jeweils in doppelte Anführungszeichen (`"`) eingeschlossen.

Die Reihenfolge der Elemente ist nicht festgelegt. Welche Elemente und Werte vorhanden sind, hängt vom jeweiligen Ereignis ab und wird im Folgenden genauer beschrieben.

Die Audit-Logging-Einträge enthalten die folgenden Elemente:

 Das Element mit dem Namen `Sec-Docs:audit@231` ist in jedem Eintrag enthalten, alle anderen Elemente sind optional.

- INIT-Eintrag
 - Element `SecDocs:audit@231`
 - Element `origin`
 - Element `SecDocs:env@231`
 - Freier Text nach den strukturierten Daten
- STOP-Eintrag
 - Element `SecDocs:audit@231`
 - Freier Text nach den strukturierten Daten
- `<operation>`-Eintrag
 - Element `SecDocs:audit@231`
 - Element `SecDocs:operation@231`, d.h. der Eintrag für die Operation `getAOID` enthält das Element `SecDocs:getAOID@231`, der Eintrag für die Operation `submitSDO` das Element `SecDocs:submitSDO@231`, usw.
- Eintrag einer Komponente von OpenLimit Middleware Version 3 Server
 - Element `SecDocs:audit@231`
 - Element `SecDocs:OpenLimit@231`



Beispiele für Logging-Einträge in der Audit-Log-Datei finden Sie auf ["Beispiele für Logging-Einträge in der Audit-Log-Datei"](#).

Element `SecDocs:audit@231`

Jeder Logging-Eintrag enthält ein Element mit dem Namen `SecDocs:audit@231`.

231 ist die bei der Internet Assigned Numbers Authority (IANA) registrierte Kennziffer für Fujitsu Technology Solutions. Somit kennzeichnet der Suffix @231 das Element gemäß RFC 5424 als ein für Fujitsu Technology Solutions reserviertes Element.

Parameter	Bedeutung
result	Gibt an, ob die Operation erfolgreich durchgeführt wurde. In den Einträgen der Version 3 Server-Komponenten ist dieser Parameter nicht enthalten. Mögliche Werte: "success" Die Operation wurde durchgeführt. "failure" Die Operation ist fehlgeschlagen.

Element origin

Das Element `origin` ist in Logging-Einträgen mit der Msgid `INIT` enthalten.

Der Elementname `origin` mit der Bedeutung seiner Parameter ist von der IANA für RFC 5424 registriert und trägt daher nicht den Suffix @231. Das `origin`-Element enthält die Information, welches Produkt welchen Herstellers das Logging erstellt hat.

Parameter	Bedeutung
software	Produktbezeichnung (immer SecDocs)
swVersion	Die zum Zeitpunkt der Erzeugung des Audit-Logging-Eintrags laufende SecDocs-Version.
enterpriseld	bei der IANA registrierte Kennziffer eines Unternehmens; Die Kennziffer von Fujitsu Technology Solutions ist 231.

Element SecDocs:env@231

Dieses Element ist nur in Logging-Einträgen mit der Msgid `INIT` enthalten. Es enthält Informationen über die Ablaufumgebung.

Parameter	Bedeutung
osUser	Benutzerkennung auf Betriebssystem-Ebene, unter der der SecDocs-Service abläuft

Element SecDocs:operation@231

Dieses Element ist genau einmal in Einträgen mit der Msgid `operation` enthalten. Es beschreibt die Details eines `operation`-Requests.



Welche Parameter das Element jeweils enthält, hängt von der Operation und deren Ergebnis ab.

Parameter	Bedeutung
-----------	-----------

adoPath	absoluter Ablagepfad für das SDO
algorithm	Name des Algorithmus
aoid	Eindeutige Objekt-ID für ein zu archivierendes SDO. Die hier protokollierte AOID wurde entweder im Soap-Header des Requests angegeben oder ist die bei erfolgreicher Bearbeitung eines <code>getAOID</code> - bzw. <code>getAOIDWithRef</code> -Requests zugewiesene AOID.
auditId	Wird von der Client-Software versorgt. Die Client-Software kann hier z.B. die Benutzeridentifikation eintragen, mit der sich der Benutzer gegenüber der Client-Software identifiziert hat.
mandant	Mandantenname, der im Soap-Header des Requests angegeben war.
mandantName	Name des Mandanten
operation	Name der Operation in abdruckbarer Form, entspricht der Msgid im Header des Eintrags.
orgId	Organisationsname, der im Soap-Header des Requests angegeben war.
orgName	Name der Organisationseinheit
requestNumber	Fortlaufende Nummer, mit der die einzelnen Requests durchgezählt werden.
role	Rolle, die im Soap-Header des Requests angegeben war. Mögliche Werte: <code>ArchiveAdmin</code> , <code>MandantAdmin</code> , <code>Archivar</code> , <code>SecDocs_MandantAuditor</code> , <code>MyRole</code>
sdoType	Name des SDO-Typs
status	Ursache für die Abweisung eines <code>deleteSDO</code> -Requests
targetMandant	Mandantenname, falls eine Passwortänderung für einen <code>MandantAdmin</code> oder für einen <code>Archivar</code> durchgeführt werden sollte.
targetRole	Rolle, für die eine Passwortänderung durchgeführt werden sollte.
tspName	Name des TSPs

Element SecDocs:OpenLimit@231

Dieses Element ist in allen Logging-Einträgen enthalten, die Funktionen der Komponenten des Version 3 Servers nutzen. Es enthält diejenigen Audit-Daten, die von den OpenLimit-Komponenten in Übereinstimmung mit dem „Common Criteria Protection Profile for an ArchiSafe Compliant Middleware“ zugeliefert werden.

8.5.1.4 Beispiele für Logging-Einträge in der Audit-Log-Datei

1. INIT-Eintrag mit den Elementen `origin`, `SecDocs:audit@231` und `SecDocs:env@231` und freiem Text nach den strukturierten Daten:

```
<110>1 2015-02-10T10:22:46.436+01:00 172.17.32.118 SecDocs 790 INIT
[origin enterpriseId="231" software="SecDocs" swVersion="SecDocs
3.0.1.0"] [SecDocs:audit@231 result="success"] [SecDocs:env@231
osUser="secdocs"] Audit started
```

2. `getAOID`-Eintrag als Beispiel für einen *operation*-Eintrag mit den Elementen `SecDocs:audit@231` und `SecDocs:getAOID@231`:

```
<110>1 2015-02-10T10:24:18.784+01:00 172.17.32.118 SecDocs 790 getAOID
[SecDocs:audit@231 result="success"] [SecDocs:getAOID@231 requestNumber="1"
operation="getAOID" role="Archivar" mandant="Mandant2"
orgId="Department_1" auditId="Archivar2" coid="myCOID1382689045840"
aid="0c3cfa2e-545b-4919-b976-07801412f26f"]
```

3. Eintrag einer Version 3 Server-Komponente mit den Elementen `SecDocs:audit@231` und `SecDocs:OpenLimit@231`:

```
<110>1 2015-02-10T10:24:19.287+01:00 172.17.32.118 SecDocs 790 SUBMIT_SDO
[SecDocs:OpenLimit@231 requestNumber="2" SessionID="db606f16-2ea9-4431adff-d84a503ceeb7"
IdentToken="SecDocs" Aoid="0c3cfa2e-545b-4919-b97607801412f26f"
ProfileIdentToken="Mandant2SDOtype1_RC_KNKi9NwBFSBr1XqeVVPlmD0VYDbKjJvQGc2
YY520Sjg="] [SecDocs:audit@231]
```

4. `STOP`-Eintrag mit dem Element `SecDocs:audit@231` und freiem Text nach den strukturierten Daten:

```
<110>1 2015-02-10T17:04:45.762+01:00 172.17.32.118 SecDocs 790 STOP
[SecDocs:audit@231 result="success"] Audit terminated
```

5. Drei Einträge als Beispiel für die Kombination von Version 3 Server-Einträgen und `SecDocs`-Eintrag bei einer Aktion:

```
<110>1 2015-02-10T10:24:19.287+01:00 172.17.32.118 SecDocs 790 SUBMIT_SDO
[SecDocs:OpenLimit@231 . . . Aoid="0c3cfa2e-545b-4919-b976-07801412f26f" .
. .] [SecDocs:audit@231]
<110>1 2015-02-10T10:24:20.114+01:00 172.17.32.118 SecDocs 790 SUBMIT_SDO
[SecDocs:OpenLimit@231 . . . Aoid="0c3cfa2e-545b-4919-b976-07801412f26f" .
. . MSG="successfully submitted!"] [SecDocs:audit@231]
<110>1 2015-02-10T10:24:20.144+01:00 172.17.32.118 SecDocs 790 submitSDO
[SecDocs:audit@231 result="success"] [SecDocs:submitSDO@231 . . . coid="
myCOID1382689045840" aid="0c3cfa2e-545b-4919-b976-07801412f26f" . . .]
```

8.5.1.5 Wechsel der Audit-Log-Dateien

Die aktuellen Audit-Log-Dateien sind unter dem Namen `audit_hostname.log` abgelegt. Dabei ist *hostname* der Name des Servers, für den die Audit-Log-Datei angelegt wurde. Auf diese Weise erhalten die Audit-Log-Dateien auch im Multi-Node-Betrieb eindeutige Dateinamen. Denn im Multi-Node-Betrieb werden für jeden Knoten im Serververbund eigene Audit-Log-Dateien in einem gemeinsamen Verzeichnis auf dem Shared Storage angelegt.

In folgenden Fällen findet ein Wechsel der Audit-Log-Datei statt:

- täglich beim Schreiben des ersten Logs nach dem Datumswechsel
- wenn die Größe der Audit-Log-Datei den vom Archiv-Administrator festgelegten Maximalwert überschreitet (siehe Konfigurationsparameter `maxAuditLogFileSize` auf "[Konfigurationsdatei secdocs.properties](#)")

Der Wechsel läuft folgendermaßen ab:

1. Die bestehende Audit-Log-Datei `audit_hostname.log` wird umbenannt in `audit_hostname.log.datum.suffix`.

Das Datum wird in der Form *YYYYMMDD* angegeben.

Das Suffix besteht aus mindestens 3 Ziffern mit führenden Nullen und dient zur Unterscheidung der Dateinamen, falls am selben Tag mehr als eine Audit-Log-Datei entsteht. Die Dateien werden chronologisch aufsteigend durchnummeriert, beginnend mit dem Suffix 000.

2. Eine neue aktuelle Audit-Log-Datei `audit_hostname.log` wird angelegt.

SecDocs-Audit-Log-Dateien



Achtung:

Alte Audit-Log-Dateien `audit_hostname.log.datum.suffix` werden nicht automatisch gelöscht. Dies kann dazu führen, dass auf dem gemounteten Volume des Filers ein Speicherengpass auftritt.

Sichern Sie deshalb die alten Audit-Log-Dateien regelmäßig und löschen Sie sie anschließend.

Betroffene Verzeichnisse:

```
archiveRoot/_config/_audit/
```

```
archiveRoot/mandantPath/_config/_audit/
```

Audit-Log-Dateien der OpenLimit-Komponente

In die Audit-Log-Dateien der zertifizierten OpenLimit-Komponente werden nur sehr selten Einträge geschrieben. Trotzdem ist es sinnvoll, die Größe der Dateien zu überwachen und sie gegebenenfalls an einem anderen Ort zu sichern. Eine Möglichkeit der Überwachung auf Linux bietet das Werkzeug `logrotate`. Um die Dateien mit `logrotate` zu überwachen, legt man im Verzeichnis `/etc/logrotate.d` eine neue Datei an, z.B. mit dem Namen `audit_secdocs`, die für jede der im Parameter `cryptoAuditFiles` genannten Dateien einen eigenen Abschnitt enthält.

Beispiel

In dem Beispiel wird die Audit-Log-Datei `/filer/cryptoLog1.log` rotiert und komprimiert, wenn sie die Größe von 40 Mbyte überschreitet. Es entsteht eine Datei mit Namen `cryptoLog1.log YYYYMMDD.gz`, die dann an anderer Stelle gesichert und anschließend gelöscht werden sollte.

```
/filer/cryptoLog1.log {
    rotate 7
    missingok
    compress
    size 40M
    dateext
}
```

8.5.1.7 Evidence Records für mandantenspezifische Audit-Log-Dateien

Mit dem Erzeugen von Evidence Records für mandantenspezifische Audit-Log-Dateien ermöglicht SecDocs die Beweiswerterhaltung für diese Audit-Log-Dateien.

Diese Funktion können Sie mit dem Parameter `createAuditLogEvidenceRecord` in der Konfigurationsdatei `secdocs.properties` global ein- oder ausschalten (siehe "[Konfigurationsdatei secdocs.properties](#)").

Bei Auslieferung ist die Funktion ausgeschaltet.

Wenn die Funktion eingeschaltet ist, werden für jede geschlossene Audit-Log-Datei eines Mandanten, d.h. für die durch Wechsel der Audit-Log-Datei entstandenen Dateien `audit_hostname.log.datum.suffix`, einmalig Evidence Records erzeugt. Dieser Vorgang erfolgt automatisch unmittelbar nach dem Schließen einer Audit-Log-Datei und wird asynchron ausgeführt.

Für jeden registrierten TSP eines Mandanten wird ein eigener Evidence Record erstellt. Dabei wird jeweils ein Zeitstempel eingeholt und ein Evidence Record mit nur einem Hash-Wert erzeugt. Der Aufbau dieser Evidence Records entspricht der Beschreibung im [Abschnitt „Erzeugen des Evidence Records“](#).

In einer Multinode-Umgebung werden auf jedem Knoten nur die jeweils von diesem Knoten erzeugten Audit-Log-Dateien versiegelt.

Es werden nur solche Audit-Log-Dateien versiegelt, bei denen der Rechnername im Dateinamen (hostname) identisch mit dem aktuellen Namen des Knotens ist.

Wurde der Rechnername verändert, dann müssen auch die Audit-Log-Dateien entsprechend umbenannt werden, damit SecDocs diese Dateien versiegelt.

Falls zum Zeitpunkt des SecDocs-Starts unversiegelte geschlossene Audit-Log-Dateien vorliegen, werden diese unmittelbar versiegelt.

Nach der Ausführung der Operation `renewHashAlgorithm` (siehe "[Operation renewHashAlgorithm](#)") oder der Operation `renewTSPSignature` (siehe "[Operation renewTSPSignature](#)") des `MandantAdminService` wird beim Erstellen von Evidence Records für die Audit-Log-Dateien dieses Mandanten der neue TSP verwendet.

Für die Audit-Log-Dateien der Archivadministration werden keine Evidence Records erzeugt.

Evidence Records werden im gleichen Verzeichnis wie die zugehörige Audit-Log-Datei unter dem Namen `<AuditLogFileName>.<TSPName>.der` abgelegt.

```
Also  archiveRoot/mandantPath/_config/_audit/  
      audit_hostname.log.datum.suffix.tspname.der  
  
z.B.  meinArchiv/mandant4/_config/_audit/  
      audit_rechner1.log.20130401.002.TSP-DFN.der
```

i Vorhandene Evidence Records werden bei der Operation `getAuditLogFile` (siehe "[Operation getAuditLogFile](#)") mit ausgegeben.

Bei der Operation `getAuditLogFileNames` (siehe "[Operation getAuditLogFileNames](#)"), die eine Liste aller verfügbaren Audit-Log-Dateien eines Mandanten ausgibt, werden evtl. vorhandene Evidence Records nicht mit aufgelistet.

8.5.2 SecDocs-Logging

Zur Laufzeit der WildFly Application Server SecDocs-Serverinstanz werden folgende Protokolldateien in das Verzeichnis `installation-dir/jboss/secdocs/log` geschrieben:

- Logdatei der WildFly SecDocs-Server-Instanz `boot.log`
wird vom WildFly Application Server beim Starten geschrieben. Hier wird das Starten des WildFly Application Servers **vor** dem Starten der SecDocs-Serverinstanz protokolliert.
- Logdatei der WildFly SecDocs-Server-Instanz `server.log`
wird vom WildFly Application Server geschrieben. Diese Datei enthält alle Logging-Informationen aus dem gesamten Life Cycle der SecDocs-Server-Instanz (Starten, Ablauf, Beenden). Diese Datei enthält sowohl die Loggingsätze, die der WildFly Application Server erzeugt, als auch die, die SecDocs schreibt.

i Die Logdatei der WildFly SecDocs-Server-Instanz `server.log` wird neu angelegt, wenn sie die Größe von 100 Mbyte erreicht. Die alten Logdateien der WildFly SecDocs-Server-Instanz werden umbenannt. Es werden maximal 10 alte Logdateien der WildFly SecDocs-Server-Instanz aufbewahrt.

- Logdateien der WildFly SecDocs-Server-Instanz

`console.log`

Diese Datei enthält alle Ausgaben und Informationen von WildFly, die direkt auf die Konsole (Linux: `stdout`) geschrieben werden und deshalb in der Datei `server.log` nicht enthalten sind.

i Im Multi-Node-Betrieb befinden sich alle Protokolldateien in den jeweiligen serverlokalen Filesystemen. In diesem Fall müssen Sie für eine globale Fehleranalyse die Protokolldateien aller am Verbund beteiligten Knoten zusammenführen. Wenn der Client-Auftrag (`requestNumber` der Fault Message), der zu einem Fehler geführt hat, bekannt ist, können Sie so den entsprechenden Server identifizieren (Suche nach "request number: <requestNumber>"). Die auf diesem Server angelegten Protokolldateien `server.log` und `console.log` können dann gezielt ausgewertet werden.



Eine detaillierte Beschreibung dieses Themas finden Sie auch in "[SecDocsInstallations- und Einsatz-Anleitung für SecDocs und V3 Server](#)" ([SD3] in Abschnitt "Literatur").

8.5.3 Zusätzliches Monitoring

Im Umfeld von SecDocs muss der Systemadministrator zusätzlich Folgendes überwachen:

Filer

Die gemounteten Volumes des Filers dürfen nicht zu 100% voll geschrieben werden.



Achtung:

Ein Vollschieben des Storage-Systems führt zu Störungen im Betrieb von SecDocs und muss unter allen Umständen vermieden werden.

CryptoModule

Das CryptoModule muss gegebenenfalls nachgestartet werden.

8.5.4 Fehlermeldungen



- Eine Liste der Fehlermeldungen, die auftreten können, finden Sie im Handbuch "[SecDocs-Rückgabewerte](#)" ([SD4] im Abschnitt "[Literatur](#)").

Weitere Rückgabewerte für eine detaillierte Analyse der Fehlermeldungen finden Sie im Handbuch "[MigSafe / OverSign - Rückgabewerte](#)" ([SD6] im Abschnitt "[Literatur](#)").

- Eine Beschreibung der Fault-Messages der Web-Services und der darin enthaltenen SecDocs-Daten finden Sie in [Abschnitt „Fault-Message“](#).

8.5.5 Internationalisierung

Die zu erwartende zunehmende internationale Verbreitung von SecDocs sowie die Anwendung durch Nicht-IT-Spezialisten machen es notwendig, den SecDocs-Server über Deutsch und Englisch hinaus in weiteren nationalen Sprachvarianten betreiben zu können. Dabei handelt es sich im Wesentlichen um die nationalisierte Ausgabe von Meldungs- und Exceptiontexten, die in der Server-Log-Datei bzw. an der Webservice-Schnittstelle sichtbar werden.

Alle relevanten auszugebenden Meldungen sind in SecDocs in nationalen Meldungskatalogen ausgelagert, auf die der SecDocs-Server in Abhängigkeit der eingestellten Sprache zugreift. Diese werden direkt im Filesystem installiert, also an einem für den SecDocs-Administrator zugänglichen Ort.

Somit ist der SecDocs-Administrator in der Lage, selbst eigene Sprachvarianten zu erzeugen und diese in SecDocs zu implementieren und zu aktivieren, ohne in die SecDocs-Struktur eingreifen zu müssen.

Die gewünschte Sprachvariante wird in der Konfigurationsdatei `setSecDocsEnv.sh` über die Variable `SECDOCS_LANG` gesteuert. Server-Log-Dateien werden dann in dieser Sprache geschrieben, Webservice-Requests der SecDocs-Clients in dieser Sprache beantwortet.

8.5.5.1 Meldungskataloge

Die Meldungskataloge, die die nationalen Meldungstexte für die Logmeldungen und die Exceptions enthalten, befinden sich in Form von Meldungs-Propertiesdateien auf dem System im Verzeichnis:

```
installation-dir/jboss/secdocs/configuration/secdocs/i18n
```

Dort können auch Sie auch Ihre eigenen Propertiesdateien hinterlegen.

Standardmäßig enthält die SecDocs-Installation jeweils eine Defaultversion der Logmeldungsriante. und der Exceptiontextedatei in Englisch sowie beide Dateien in einer deutschen Va- Auf die Defaultversion greift SecDocs immer dann zu, wenn ein Meldungstext in einer nationalen Variante der Datei nicht gefunden wird.

Die Namen der Propertiesdateien müssen folgendem Muster entsprechen:

- Logmeldungsdatei: `logmessages[_locale].properties`
- Exceptiontextedatei: `exception[_locale].properties`

Hierbei ist *locale* die länderspezifische Lokale, die bei der Defaultversion entfällt.

Die länderspezifische Lokale setzt sich aus einem Sprachenkürzel (ISO 639-2) und einem optionalen Länderkürzel (ISO 3166) zusammen, z.B.

```
de[_DE]
```

Dies entspricht der Syntax für die `LANG`-Variable.

In der aktuellen SecDocs-Version heißen diese Dateien `logmessages_de.properties` bzw. `exception_de.properties`, da die deutsche Version der Dateien keine länderspezifischen Besonderheiten aufweist.

Bei Start des SecDocs-Servers werden alle Propertiesdateien mit gemeinsamem Namensstamm (`logmessages` bzw. `exception`), die zur eingestellten Sprachvariante passen, zu einem Ressourcen-Bundle zusammengefasst. Bei der Einstellung der Sprachvariante `de_DE.UTF-8` können das beispielsweise folgende Dateien sein:

```
logmessages_de_DE.properties
logmessages_de.properties
logmessages.properties
```

Bei der Meldungsabgabe werden die Dateien, sofern sie existieren, in dieser Reihenfolge nach der Meldung durchsucht. Sobald die Meldung in einer der Dateien gefunden wurde, werden keine weiteren Dateien durchsucht. Deshalb müssen in den höher spezialisierten Propertiesdateien nur die Meldungstexte enthalten sein, die sich von den Texten in den weniger spezialisierten Dateien unterscheiden. Der Defaultkatalog stellt schließlich sicher, dass immer ein Meldungstext gefunden wird, wenn dann auch nur in Englisch.

8.5.5.2 Einbringen eigener Meldungskataloge

Wenn Sie SecDocs in einer anderen Sprache als Englisch oder Deutsch betreiben wollen, müssen Sie folgende Schritte durchführen:

- Übersetzen Sie die vorhandenen Default-Propertiesdateien in Ihre gewünschte Sprache.

Beachten Sie:

- Die einzelnen Zeilen einer Propertiesdatei bestehen aus einem Schlüssel am Zeilenanfang sowie einem zugeordneten Text. Der Schlüssel darf weder verändert, noch mit übersetzt werden.
- Meldungstexte können Positionsparameter ({0},{1},...) enthalten, die durch variable Argumente ersetzt werden. Diese dürfen Sie entsprechend den Anforderungen der gewünschten Sprachsyntax variabel positionieren .
- Die Texte in den Meldungs-Propertiesdateien müssen UTF-8-kodiert sein.

Das bedeutet insbesondere, dass Zeichen, die nicht zum 7-Bit-ASCII-Zeichensatz gehören, mit einer Escape-Sequenz für Unicode-Zeichen, d.h. im Format `\uxxxx` dargestellt werden müssen, wobei *x* eine hexadezimale Ziffer ist, also 0...9, A...F (oder a...f) .

Beispiel

é muss als `\u00E9`, Æ als `\u00C3` dargestellt werden.

- Speichern Sie Ihre Übersetzungen in eigenen Propertiesdateien entsprechend der Vorgabe mit dem passenden länderspezifischen Namenszusatz ab.
- Kopieren Sie diese Propertiesdateien im Filesystem in das Verzeichnis
`installation-dir/jboss/secdocs/configuration/secdocs/i18n`
- Ändern Sie ggf. Ihre Einstellung der Sprachvariante und starten Sie SecDocs neu.

8.5.5.3 Sprachauswahl

Die gewünschte Sprachvariante des SecDocs-Servers können Sie mit der Variablen `SECDOCS_LANG` im SecDocs-Konfigurationsskript `setSecDocsEnv.sh` konfigurieren. Beim Starten des SecDocs-Servers setzt SecDocs die Umgebungsvariable `LANG` auf den im Startskript konfigurierten Wert.

Wird SecDocs in einer Sprache gestartet, für die keine sprachspezifischen Meldungs-Propertiesdateien existieren, werden die Meldungstexte immer in Englisch ausgegeben.

Näheres zur Spracheinstellung finden Sie in der Installationsanleitung "[SecDocs Installations- und Einsatz-Anleitung](#)" ([SD3] im Abschnitt "Literatur").

i SecDocs benutzt zahlreiche Fremdprodukte, die ihrerseits selbst Meldungen in den SecDocs-Log-Dateien ausgeben sowie Exceptions werfen können (z.B. Java, MSOS, ...). Meldungen dieser Produkte erscheinen u.U. auch dann in Englisch, wenn im System eine andere Sprache als Englisch eingestellt ist.

9 Fachwörter

Fachwörter, die an anderer Stelle erklärt werden, sind mit ->*kursiver* Schrift ausgezeichnet.

- AOID** (Archive Object Identifier)
Ein eindeutiger Identifikator für ein gespeichertes Submission Data Object (->*SDO*). Mit Hilfe der AOID kann die Anwendung Operationen bzgl. eines SDOs ausführen, z.B. lesen oder löschen.
- ArchiSafe** beschreibt den Mindestumfang der Archivierungs-Funktionen in der ->*TR-03125*.
<http://www.commoncriteriaportal.org/files/ppfiles/pp0049b.pdf>
- ArchiSig** Verbundprojekt, das vom Bundesministerium für Wirtschaft und Technologie (BMWi) im Rahmen des Programms „VERNET - Sichere und verlässliche Transaktionen in offenen Kommunikationsnetzen“ gefördert wird. ArchiSig befasst sich mit der beweiskräftigen und sicheren Langzeitspeicherung elektronisch signierter Dokumente. Im Rahmen von ArchiSig wurden aus den allgemeinen gesetzlichen Regelungen konkrete rechtliche Anforderungen abgeleitet. Diese Anforderungen an Systeme zur langfristigen Aufbewahrung elektronisch signierter Dokumente wurden prototypisch implementiert. Es konnte erstmalig gezeigt werden, dass die Langzeitaufbewahrung elektronisch signierter Dokumente gesetzeskonform, performant und akzeptabel umgesetzt werden kann.
- Authentizität** kennzeichnet die Echtheit eines Unterzeichners. SecDocs verwendet Qualifizierte ->*Elektronische Signaturen* zum Nachweis der Authentizität eines elektronischen Dokumentes.
- Client** Anwendung, die die Dienste eines ->*Web-Service* anfordert und nutzt. Der Begriff ist synonym zum „Service Consumer“ beim ->*SOA-Konzept*.
- COID** (Client Object Identifier)
Ein vom Client vergebbarer Bezeichner zur Identifizierung eines Archivobjekts.
Die COID kann bei den Operationen des *Archiving* Webservice statt einer ->*AOID* angegeben werden. Sie kann auch dazu verwendet werden, mehrere Versionen eines Dokuments über eine gemeinsamen COID zu verwalten.

Digitale Signatur

Klasse von kryptografischen (d.h. mathematischen) Verfahren.
siehe auch ->*elektronische Signatur*

Elektronische Archivierung

unveränderbare, langzeitige Aufbewahrung elektronischer Information.

Elektronische Signatur

mit elektronischen Informationen verknüpfte Daten, mit denen der Unterzeichner (Signaturersteller) identifiziert und die Integrität der signierten elektronischen Informationen geprüft werden kann. In der Regel handelt es sich bei den elektronischen Informationen um elektronische Dokumente. Die elektronische Signatur erfüllt technisch gesehen den gleichen Zweck wie eine eigenhändige Unterschrift auf Papierdokumenten.

Die elektronische Signatur ist ein rechtlicher Begriff, im Gegensatz zur ->*Digitalen Signatur*, die eine Klasse von kryptografischen Verfahren bezeichnet.

Evidence Record

liefert den Beweis für die Integrität eines Dokuments, das in einem Langzeitarchiv gespeichert ist. Der Evidence Record muss eine nahtlose Kette von gültigen ->*Zeitstempeln* enthalten, rückwirkend bis zum Zeitpunkt der Übernahme des Dokuments in das Archivsystem (gemäß IETF RFC 4998).

Externes Datenobjekt

Zu archivierendes Datenobjekt, das unabhängig vom SOAP-Request auf den Server übertragen wird. Datenübertragung und `submitSDO-/retrieveSDO-` Request erfolgen zu verschiedenen Zeitpunkten: Bei der Archivierung erfolgt zuerst die Übertragung der externen Datenobjekte vom client-lokalen Rechner auf den SecDocs-Server und dann zu einem späteren Zeitpunkt der `submitSDO`-Request. Analog dazu wird beim Lesen von Dokumenten erst der `retrieveSDO`-Request abgesetzt und danach erst die Datenübertragung vom Server auf den lokalen Rechner des Client durchgeführt.

Externe Referenz-ID

Eindeutige Referenz auf ein externes Datenobjekt im SDO.

Wird ein Dokument als externes Datenobjekt archiviert, so steht im SDO nur noch eine Referenz auf dieses externe Dokument.

Freigabezeitpunkt

Datum und Uhrzeit, ab der ein ->*SDO* gelöscht werden kann.

Gegenseitige Authentifizierung

In einer Netzwerkumgebung müssen alle Kommunikationspartner ihre Identität nachweisen, bevor untereinander vertrauliche Daten ausgetauscht werden.

Auf diese Art können Client und Server sicher gehen, dass sie es mit legitimen Partnern zu tun haben.

Hash-Algorithmus

(auch Digest-Algorithmus oder Fingerprint-Algorithmus)

Identifikationsmerkmal für einen Datenbereich beliebiger Größe mit einer extrem geringen Wahrscheinlichkeit, dass zwei unterschiedliche Datenbereiche das gleiche Identifikationsmerkmal erhalten. Umgekehrt kann aus dem Identifikationsmerkmal nicht auf den Inhalt der Daten geschlossen werden. Es ist auch nahezu unmöglich, ein Dokument zu erzeugen oder zu ändern, das das gleiche Identifikationsmerkmal bekommt wie ein bestimmtes anderes Dokument.

Ein Hash-Algorithmus ist eine mathematische Einweg-Funktion, die zu einem großen Eingangswert (z.B. einer 700 GB = 700.000.000.000 Byte/Oktett großen Datei) einen kleinen Ausgangswert (16 Byte/Oktett) berechnet.

Im Umfeld der Qualifizierten Elektronischen Signatur gemäß SigV erstellt das BSI regelmäßig im Auftrag Bundesnetzagentur eine Bewertung zur zeitlichen Verwendung bestimmter Hash-Algorithmen für die Signaturerzeugung und Signaturprüfung.

Hash-Baum	Datenstruktur, die die nachprüfbare Zusammenfassung einer beliebigen Anzahl von -> <i>Hash-Werten</i> zu einem einzigen Hash-Wert (Wurzel-Hash-Wert) ermöglicht (1979, Ralph Merkle).
Hash-Wert	(auch Digest, Fingerprint oder Fingerabdruck) Ergebnis (Ausgangswert) der Anwendung eines -> <i>Hash-Algorithmus</i> auf einen Eingangswert.
HDO	(Hash Data Object) besteht aus der Konkatenation von -> <i>SignatureVerificationInfo</i> und -> <i>SDO</i> (in dieser Reihenfolge).
HTTPS	(HyperText Transfer Protocol Secure) Kommunikationsprotokoll im World Wide Web, um Vertraulichkeit und Integrität in der Kommunikation zwischen Webserver und Webbrowser herzustellen. Dies wird u.a. durch Verschlüsselung und Authentifizierung erreicht.

Langzeitarchivierung

Erfassung, langfristige Aufbewahrung und Erhaltung der dauerhaften Verfügbarkeit von Informationen.

Mandantenfähigkeit

Technik, die auf demselben Server oder demselben Software-System mehrere Mandanten (Kunden, Auftraggeber oder Organisationseinheiten) verwalten kann. Das Schriftgut der verschiedenen Mandanten wird strikt voneinander getrennt gehalten. Ein Mandant kann nicht auf die Daten, Dokumente oder Parameter eines anderen Mandanten zugreifen.

Multi-Node-Betrieb, Multi-Node-Konfiguration

Parallelbetrieb mehrerer SecDocs-Instanzen für ein Archiv.
Vgl. ->*Single-Node-Betrieb*, *Single-Node-Konfiguration*

Primärdokumente

Dokumente (Dateien), die BASE64-codiert im ->*SDO* eingebettet sind.

SDO (Submission Data Object)
->*XML*-Container, in dem ein zu archivierendes Datenobjekt an SecDocs übergeben wird.

SDO-Typ Ein SDO-Typ beschreibt das zu archivierende ->*SDO*. Er besteht aus

- den Informationen über die Struktur des zu archivierenden Schriftguts. Diese Struktur wird in einem ->*XML-Schema* festgelegt.
- den Informationen zur Adressierung für (Nutz-)Daten, Signaturen und Metadaten, die in der zugehörigen Filterdefinition abgelegt sind.

Signatur-Algorithmus

Verschlüsselungs-Algorithmus, der zum Erzeugen ->*elektronischer Signaturen* verwendet wird.

SFTP-Server Zu archivierende externe Datenobjekte werden unabhängig vom SOAP-Request per Secure File-Transfer-Protokoll (SFTP) auf den SecDocs-Server übertragen. Zu diesem Zweck ist ein integrierter SFTP-Server als fester Bestandteil des SecDocs-Servers installiert.

Signatur-Container

Datencontainer in einem der Formate PKCS#7, CMS oder XMLDSig, der eine oder mehrere Signaturen enthalten kann. Ein Signatur-Container kann in einem Primärdokument enthalten sein, oder als abgesetzte Signatur zu einem Primärdokument vorliegen.

Ein Primärdokument kann höchstens einen Signatur-Container enthalten, jedoch können mehrere Signatur-Container als abgesetzte Signaturen zu einem Primärdokument vorliegen.

Signature Verification Information

Die zum Zeitpunkt der Archivierung eines SDOs verwendeten vollständigen Zertifikatsketten und Sperrinformationen (Sperrlisten und OCSP-Antworten) werden in der Signature Verification Information archiviert und zusammen mit dem SDO durch einen Evidence Record abgesichert. Mit diesen Informationen – vollständige Zertifikatsketten und Sperrinformationen – ist es möglich, die zum Zeitpunkt der Archivierung durchgeführte Verifikation zu einem beliebigen Zeitpunkt später noch einmal durchzuführen. Darüber hinaus enthält die Signature Verification Information das Protokoll der zum Archivierungszeitpunkt durchgeführten Verifikation. Dieses Verifikationsprotokoll zeigt, welche Signaturen geprüft wurden, mit welchen Zertifikaten und Sperrinformationen die Verifikation durchgeführt wurde und zu welchem Resultat das Verifikationsmodul gelangte. Detaillierte Informationen zum Prüfprotokoll können beim Hersteller OpenLimit angefordert werden.

Single-Node-Betrieb, Single-Node-Konfiguration

Eine einzige SecDocs-Instanz bedient alle Schnittstellen für ein Archiv.
Vgl. -> *Multi-Node-Betrieb, Multi-Node-Konfiguration*

SOA

(**S**ervice-**O**riented **A**rchitecture).

Eine SOA ist ein Konzept für eine Systemarchitektur, in dem Funktionen in Form von wieder verwendbaren, technisch voneinander unabhängigen und fachlich lose gekoppelten *Services* implementiert werden. Services können unabhängig von zugrunde liegenden Implementierungen über Schnittstellen aufgerufen werden, deren Spezifikationen öffentlich und damit vertrauenswürdig sein können. Service-Interaktion findet über eine dafür vorgesehene Kommunikationsinfrastruktur statt.

SOAP

(**S**imple **O**bject **A**ccess **P**rotocol)

Protokoll, mit dessen Hilfe Daten zwischen Systemen ausgetauscht und Remote Procedure Calls durchgeführt werden können. SOAP stützt sich auf die Dienste anderer Standards, ->XML zur Repräsentation der Daten und Internet-Protokolle der Transport- und Anwendungsschicht zur Übertragung der Nachrichten.

Timestamp

siehe -> *Zeitstempel*.

TR-03125

(Technische Richtlinie „Vertrauenswürdige elektronische Langzeitspeicherung“)

Diese vom Bundesamt für Sicherheit in der Informationstechnik veröffentlichte Richtlinie beschreibt ein Architekturmodell zur vertrauenswürdigen elektronischen Langzeitspeicherung unter Verwendung von ArchiSig, ArchiSafe und OpenLimit CryptoModule. Die TR-03125 ist die Entwicklungsbasis von SecDocs.

TSP (Timestamp Provider)
siehe -> *Zeitstempelanbieter*.

Web-Service

Anwendung, die auf einem Web-Server läuft und über eine standardisierte und programmatische Schnittstelle (öffentlich) verfügbar ist. Die Anwendung kann über das -> *SOAP*-Protokoll angesprochen werden. Die Schnittstelle eines Web-Service ist in -> *WSDL* beschrieben.

WSDL (Web Services Description Language)
bietet -> *XML*-Sprachregeln für die Beschreibung von -> *Web-Services*. Ein Web-Service wird dabei durch eine Auswahl von Ports definiert.

XAIP (XML Formatted Archival Information Package)
selbstbeschreibendes und wohlgeformtes *XML*-Dokument, das gegen ein gültiges und autorisiertes *XML*-Schema geprüft werden kann. Ein solches Archivdatenobjekt enthält sämtliche Inhaltsdaten (Primärinformationen) und Metainformationen, die für eine zuverlässige und vollständige Rekonstruktion von Geschäfts- oder Verwaltungsvorgängen bis zum Ablauf der gesetzlich vorgeschriebenen Aufbewahrungsfristen erforderlich sind.

XML (Extensible Markup Language)
definiert eine Sprache zur logischen Strukturierung von Dokumenten mit dem Ziel, diese einfach zwischen verschiedenen Anwendungen auszutauschen.

XML-Schema definiert die zulässigen Elemente und Attribute einer -> *XML*-Beschreibung.
XML-Schemas können verschiedene Formate haben, z.B. DTD (Document Type Definition), *XML Schema* (W3C-Standard) oder XDR (XML Data Reduced).

XPath (XML Path Language)
Abfragesprache, mit der Teile eines -> *XML*-Dokuments adressiert werden können. XPath wurde vom W3-Konsortium entwickelt und ist die Grundlage weiterer Standards wie XSLT, XPointer und XQuery.

Zeitstempel Bestätigung der Vorlage eines Dokuments zu einem bestimmten Zeitpunkt durch den -> *Zeitstempelanbieter*. Der Zeitstempel bezieht sich auf die Existenz des Dokuments und nicht auf dessen Inhalt.

Zeitstempelanbieter

(auch Zeitstempel-Dienst)
nimmt als Server im Internet/Intranet signierte Dateien oder auch nur deren Signaturen entgegen und versieht diese mit einem -> *Zeitstempel*. Ein Zeitstempel-Dienst kann sowohl in einem internen Netz als auch im Internet angeboten und genutzt werden.
Es wird empfohlen, nur Zeitstempelanbieter zu verwenden, die offiziell durch Ämter akkreditiert sind. In Deutschland durch die Bundesnetzagentur. Jeder Zeitstempelanbieter ist auch ein -> *Zertifizierungsdiensteanbieter*.

Zertifikat Ein digitales Zertifikat ist ein digitaler Datensatz, der bestimmte Eigenschaften von Personen oder Objekten bestätigt und dessen Authentizität und Integrität durch kryptografische Verfahren geprüft werden kann. Das digitale Zertifikat enthält insbesondere die zu seiner Prüfung erforderlichen Daten.

Zertifizierungsdiensteanbieter

natürliche oder juristische Personen, die qualifizierte Zertifikate und Zeitstempel ausstellen.

Zertifizierungsdiensteanbieter können sich freiwillig bei der Bundesnetzagentur gemäß §15 Abs. 1 SigG und §11 SigV akkreditieren lassen. Eine Akkreditierung dient als Nachweis für die technische und administrative Sicherheit von qualifizierten Signaturen und wird nach einer Überprüfung des Zertifizierungsdiensteanbieters durch die Bundesnetzagentur ausgesprochen. Akkreditierte Zertifizierungsdiensteanbieter müssen die von ihnen ausgestellten Zertifikate noch 30 Jahre nach Ablauf des Gültigkeitszeitraumes über einen Verzeichnisdienst abrufbar und nachprüfbar halten.

10 Abkürzungen

ACM_PP	Protection Profile for an ArchiSafe Compliant Middleware for Enabling the Long-Term Preservation of Electronic Documents
ADO	Archivdatenobjekt
AOID	Archive Object Identifier, Archivobjekt-ID
BMWi	Bundesministerium für Wirtschaft und Technologie
BPM	Business Process Management
BSI	Bundesamt für Sicherheit in der Informationstechnik
CRL	Certificate Revocation List
COID	Client Object Identifier
DMS	Dokumenten-Management-System
ERP	Enterprise Resource Planning
FTP	File Transfer Protocol
HDO	Hash Data Object
HTTPS	HyperText Transfer Protocol Secure
IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
OID	Object Identifier
OCSP	Online Certificate Status Protocol
PKI	Public Key Infrastructure
RFC	Request for Comments
PKCS	Public Key Cryptography Standards
SDO	Submission Data Object
SFTP	Secure File Transfer Protocol
SHA	Secure Hash Algorithm
SigG	Signaturgesetz
SigV	Signaturverordnung
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SSH	Secure Shell

SVI	Signature Verification Information
TCP/IP	Transmission Control Protocol / Internet Protocol
TLS	Transport Layer Security
TR	Technische Richtlinie
TSP	Zeitstempelanbieter, Timestamp Provider
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTC	Coordinated Universal Time
WSDL	Web Services Description Language
XAIP	XML Formatted Archive Information Package
XML	Extended Markup Language
XPath	XML Path Language
ZDA	Zertifizierungsdiensteanbieter

11 Literatur

Die Handbücher sind online unter <http://manuals.ts.fujitsu.com> und auf der mitgelieferten DVD zu finden. Eine Zuordnung der Handbuchtitel zu den entsprechenden Dateinamen entnehmen Sie bitte der Freigabemitteilung.

- [SD1] **SecDocs Administration und Bedienung**
Benutzerhandbuch
- [SD2] **SecDocs Archivierung von Dokumenten gemäß Richtlinie TR-ESOR**
Benutzerhandbuch
- [SD3] **SecDocs V3.1 Installationsanleitung**
Referenzhandbuch
- [SD4] **SecDocs-Rückgabewerte**
Referenzhandbuch
- [SD5] **CryptoModule Runtime Umgebung für Fujitsu SecDocs V3.1**
Benutzerhandbuch
- [SD6] **MigSafe / OverSign Rückgabewerte**
Benutzerhandbuch

Literatur zur elektronischen Langzeitarchivierung im Internet

- [W1] **BSI Technische Richtlinie 03125:**
Vertrauenswürdige elektronische Langzeitspeicherung
Bundesamt für Sicherheit in der Informationstechnologie (BSI)

https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Technische-Richtlinien/TR-nach-Thema-sortiert/tr03125/TR-03125_node.html

Unter dieser Adresse finden Sie Verweise auf die nachfolgend genannten Dokumente der BSI TR-03125 Version 1.2:

- [W2] **BSI TR-03125**
Vertrauenswürdige elektronische Langzeitspeicherung Version 1.2.1

https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03125/BSI_TR_03125_V1_2_1.pdf?__blob=publicationFile&v=2

-
- [W3] BSI TR-03125 Anlage M.1
ArchiSafe Modul Version 1.2.1
https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03125/BSI_TR_03125_Anlage_M1_V1_2_1.pdf?__blob=publicationFile&v=1
- [W4] BSI TR-03125 Anlage TR-ESOR-S
Schnittstellenspezifikation Version 1.2.1
https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03125/BSI_TR_03125_Anlage_S_V1_2_1.pdf?__blob=publicationFile&v=1
- [W5] BSI TR-03125 Anlage TR-ESOR-F
Formate Version 1.2.1
https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03125/BSI_TR_03125_Anlage_F_V1_2_1.pdf?__blob=publicationFile&v=1
- [W6] **Common Criteria Protection Profile for an ArchiSafe Compliant Middleware for Enabling the Long-Term Preservation of Electronic Documents**
<http://www.commoncriteriaportal.org/files/ppfiles/pp0049a.pdf>

Verwendete Standards

- [S1] **Data elements and interchange formats – Information interchange – Representation of dates and times, ISO8601:**
http://de.wikipedia.org/wiki/ISO_8601
- [S2] **Date and Time on the Internet: Timestamps, RFC 3339**
<http://tools.ietf.org/html/rfc3339>
- [S3] **Evidence Record Syntax (ERS), RFC 4998**
<http://tools.ietf.org/html/rfc4998>
- [S4] **Internet X.509 Public Key Infrastructure , Time-Stamp Protocol (TSP), RFC 3161**
<http://tools.ietf.org/html/rfc3161>
- [S5] **Java™ Platform, Enterprise Edition (Java EE) Specification, v6**
<http://www.oracle.com/technetwork/java/javaee/tech/javaee6technologies-1955512.html>
- [S7] **Simple Object Access Protocol (SOAP) 1.1**
<http://www.w3.org/TR/soap/>
- [S9] **The Syslog Protocol, RFC 5424**
<http://tools.ietf.org/html/rfc5424>

[S10] **Web Services Description Language (WSDL) 1.1**

<http://www.w3.org/TR/wsdl.html>