Beim vorliegenden Dokument handelt es sich um einen Entwurf der gematik in Vorbereitung auf zukünftige normative Festlegungen als Grundlage entsprechender Zulassungs- und Bestätigungsverfahren. Die gematik veröffentlicht diesen Entwurf mit dem Ziel, dass sich Interessierte bereits frühzeitig einen Überblick über die mögliche Weiterentwicklung der Telematikinfrastruktur verschaffen können. Die gematik übernimmt keine Gewähr für die Aktualität, Richtigkeit und Vollständigkeit dieses Entwurfes und behält sich das Recht vor, ohne vorherige Ankündigung Änderungen oder Ergänzungen vorzunehmen oder von den Regelungen insgesamt bzw. teilweise Abstand zu nehmen.

In [gemSpec_Kon], Kapitel 4.1.8 Signaturdienst, sind zwei Dienste enthalten:

- Signaturdienst
- Authentifizierungsdienst

Es gilt beide Dienste ohne fachliche Änderung in separaten Kapiteln darzustellen. Hierzu werden alle Aspekte des Authentifizierungsdienstes aus Kapitel 4.1.8 entfernt und in ein separates Kapitel 4.1.13 ausgelagert.

Änderungsbedarf in gemSpec_Kon

Es wird ein neues Kapitel 4.1.13 ergänzt

4.1.13 Authentifizierungsdienste

Der Authentifizierungsdienst bietet Clientsystemen und Fachmodulen eine Schnittstelle zum Signieren von Binärstrings zum Zweck der externen Authentisierung.

Innerhalb des Authentifizierungsdienstes werden folgende Präfixe für Bezeichner verwendet:

- Events (Topic Ebene 1): *keine Events vorhanden*
- Konfigurationsparameter: *keine Konfigurationsparameter vorhanden*

Eine Prüfung der Signatur bietet der Authentifizierungsdienst nicht an. Sie wird im Rahmen der Operation VerifyDocument des Signaturdienstes angeboten.

4.1.13.1 Funktionsmerkmalweite Aspekte

4.1.13.1.1 Externe Authentisierung

⌦ TIP1-A_5437 Signaturverfahren für externe Authentisierung

Der Signaturdienst MUSS das Signaturverfahren PKCS#1 entsprechend TAB_KON_780 – Signaturverfahren Externe Authentisierung unterstützen. ⌫

Tabelle 1: TAB_KON_780 – Signaturverfahren Externe Authentisierung

Signaturformat	Standard	Dokumentformate	QES/ nonQES	Bemerkung
PKCS#1 (V2.1)	[RFC3447]	Binär	nonQES	Dieses Signaturformat DARF NUR in Verbindung mit dem zur Authentisierung vorgesehenen Schlüssel des HBAx und des SM-B genutzt werden. Die Nutzung ist auf Dokumente (Hash) von maximal 512 bit Länge beschränkt.

⌦ TIP1-A_5149 PKCS#1-Schnittstelle nur für Authentisierung mit HBAx und SM-B nutzen

Der Hersteller des Konnektors MUSS den Anwender (Clientsystem) im Handbuch des Konnektors geeignet und ausreichend darüber informieren, dass das Signaturformat PKCS#1 nur zu Authentisierungszwecken mit dem Authentisierungsschlüssel des HBAx und des SM-B verwendet werden darf.⌫

4.1.13.2 Durch Ereignisse ausgelöste Reaktionen

keine

4.1.13.3 Interne TUCs

keine

4.1.13.4 Operationen an der Außenschnittstelle

⌦ TIP1-A_5665 Basisdienst Authentifizierungsdienst

Der Konnektor MUSS Clientsystemen den Basisdienst Authentifizierungsdienst anbieten.

Tabelle 2: TAB_KON_839 Basisdienst Authentifizierungsdienst

Name	`AuthSignatureService`	
Version (KDV)	Siehe Anhang D (WSDL-Version)	
Namensraum	Siehe Anhang D	
Namensraum-Kürzel	`SIG für Schema und SIGW für WSDL`	
Operationen	**Name**	**Kurzbeschreibung**
	`ExternalAuthenticate`	Binärstring signieren (nonQES)
WSDL	`AuthSignatureService.wsdl`	
Schema	`Kein`	

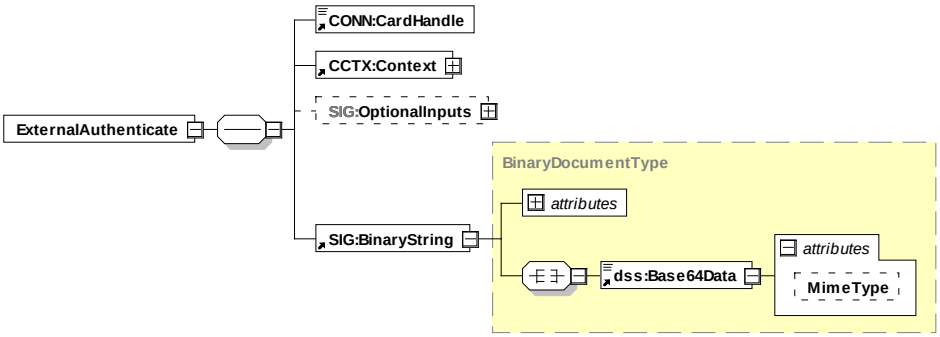
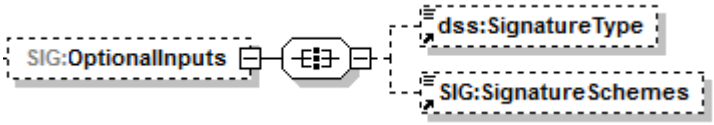
⌫

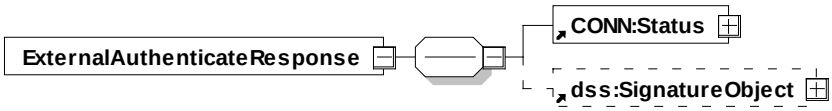
4.1.13.4.1 ExternalAuthenticate

⌦ TIP1-A_5439 Operation ExternalAuthenticate

Der Authentifizierungsdienst des Konnektors MUSS an der Clientschnittstelle eine Operation ExternalAuthenticate anbieten**.**

Tabelle 3: TAB_KON_781 Operation ExternalAuthenticate

Name	`ExternalAuthenticate`	
Beschreibung	Diese Operation versieht einen Binärstring der maximalen Länge 512 Bit mit einer nicht-qualifizierten elektronischen Signatur (nonQES). Dazu wird das Signaturverfahren PKCS#1 verwendet. Das AUT-Zertifikat der SM-B und das AUT-Zertifikat des HBAx werden unterstützt.	
Aufrufparameter		
	Name	Beschreibung
	`CONN:CardHandle`	Identifiziert die zu verwendende Signaturkarte. Die Operation unterstützt HBAx und SM-B.
	`CCTX:Context`	<u>Aufrufkontext für HBAx</u>: MandantId, ClientSystemId, WorkplaceId, UserId verpflichtend <u>Aufrufkontext für SM-B</u>: MandantId, ClientSystemId, WorkplaceId verpflichtend; UserId nicht ausgewertet
	`SIG:OptionalInputs`	Enthält optionale Eingangsparameter: 
	`SIG:BinaryString`	Dieses Element enthält im Kindelement `dss:Base64Data` den zu signierenden Binärstring. Das XML Attribut SIG:BinaryString/dss:Base64Data/@MimeType MUSS den Wert "application/octet-stream" haben. Die maximale Länge des Binärstrings beträgt 512 Bit entsprechend der maximal zu erwartenden Hash-Größe. Aus der Länge des Binärstrings wird auf das verwendete

		Hashverfahren geschlossen. Es werden folgende Längen unterstützt: • 256 Bit: SHA-256 (OID 2.16.840.1.101.3.4.2.1) • 384 Bit: SHA-384 (OID 2.16.840.1.101.3.4.2.2) • 512 Bit: SHA-512 (OID 2.16.840.1.101.3.4.2.3) Im Falle des Signaturverfahrens RSASSA-PKCS1-v1_5 werden SHA-256, SHA-384 und SHA-512 unterstützt. Im Falle des Signaturverfahrens RSASSA-PSS wird SHA-256 unterstützt. Für die Signaturerstellung gilt: • Im Falle des Signaturverfahrens RSASSA-PKCS1-v1_5 beginnt der Konnektor die Ausführung der Methode EMSA-PKCS1-v1_5-ENCODE nach [RFC3447], Abschnitt 9.2, mit Schritt 2, Erstellung des DigestInfo-Datenfeldes. • Im Falle des Signaturverfahrens RSASSA-PSS beginnt der Konnektor die Ausführung der Methode EMSA-PSS-ENCODE nach [RFC3447], Abschnitt 9.1.1, mit Schritt 3.
	`dss:SignatureType`	Durch dieses in [OASIS-DSS] (Abschnitt 3.5.1) beschriebene Element wird der Typ der zu erzeugenden Signatur bestimmt. Als Signaturtyp wird unterstützt : • **PKCS#1-Signatur** Durch Übergabe der URI urn:ietf:rfc:3447 wird eine PKCS#1 (Version 2.1) Signatur gemäß [RFC3447] erzeugt, die als `dss:Base64Signature` mit der oben genannten URI zurückgeliefert wird. Andere `SignatureType`-Angaben führen zu einer Fehlermeldung 4111 (Ungültiger Signaturtyp oder Signaturvariante). Fehlt dieses Element, so wird ebenfalls der Signaturtyp PKCS#1-Signatur verwendet.
	`SIG:SignatureSchemes`	Durch dieses Element wird für PKCS#1-Signaturen zwischen den folgenden SignatureScheme-Optionen unterschieden: • RSASSA-PSS • RSASSA-PKCS1-v1_5 Fehlt dieses Element, so wird als Default-SignatureScheme RSASSA-PSS gewählt.
Rückgabe		
	`CONN:Status`	Enthält den Status der ausgeführten Operation.

	`dss:SignatureObject`	Enthält im Erfolgsfall die erzeugte Signatur in Form eines `dss:SignatureObject`-Elements gemäß [OASIS-DSS] (Abschnitt 3.2).
		Der Signaturwert wird im XML-Element `dss:SignatureObject/dss:Base64Signature` übergeben. Das XML-Attribut `dss:SignatureObject/dss:Base64Signature/@Type` kennzeichnet durch den Wert urn:ietf:rfc:3447 den Signatur-Typ.
		Die XML-Elemente
		`dss:SignatureObject/ds:Signature`
		`dss:SignatureObject/dss:Timestamp`
		`dss:SignatureObject/dss:SignaturePtr`
		`dss:SignatureObject/dss:Other`
		werden nicht verwendet.

Vorbedingungen	Keine	
Nachbedingungen	Keine	

Der Ablauf der Operation ExternalAuthenticate ist in Tabelle TAB_KON_782 beschrieben:

Tabelle 4: TAB_KON_782 Ablauf Operation ExternalAuthenticate

Nr.	Aufruf Technischer Use Case oder Interne Operation	Beschreibung
1.	checkArguments	Alle übergebenen Parameterwerte werden auf Konsistenz und Gültigkeit überprüft. Treten hierbei Fehler auf, so bricht die Operation mit Fehler 4000 ab.
2.	TUC_KON_000 „Prüfe Zugriffs-berechtigung“	Die Prüfung erfolgt durch den Aufruf TUC_KON_000 {$context.mandantId; $context.clientsystemId; $context.workplaceId; $context.userId; $cardHandle} Tritt bei der Prüfung ein Fehler auf, bricht die Operation mit Fehlercode aus TUC_KON_000 ab.
3.	TUC_KON_026 „Liefere CardSession“	Ermittle CardSession über TUC_KON_026 { MandantId, CsId, CardHandle, UserId }

4.	TUC_KON_218 „Signiere“	Signaturberechnung durch Aufruf des TUC_KON_218 {
		PinRef = PIN.CH bzw. PIN.SMC;
		KeyRef = PrK.HP.AUT bzw. PrK.HCI.AUT;
		AlgorithmusID = signPKCS1_V1_5 oder signPSS;
		DTBS = Binärstring
		}

Tabelle 5: TAB_KON_783 Übersicht Fehler Operation ExternalAuthenticate

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen TUCs können folgende weiteren Fehlercodes auftreten:			
4000	Technical	Error	Syntaxfehler

Fehlercode	ErrorType	Severity	Fehlertext
4058	Security	Error	Aufruf nicht zulässig

Die folgende Tabelle führt die zulässigen privaten Schlüssel für die Operation ExternalAuthenticate auf:

Tabelle 6: TAB_KON_784 Privater Schlüssel je Karte für ExternalAuthenticate

Karte	Schlüssel
SM-B	PrK.HCI.AUT in DF.ESIGN
HBAx	PrK.HP.AUT in DF.ESIGN

⌫

4.1.13.5 Betriebsaspekte

Keine

Aus Kapitel 4.1.8 werden die Bestandteile des Authentifizierungsdienstes entfernt.

4.1.8 Signaturdienst

Der Signaturdienst bietet Clientsystemen und Fachmodulen eine Schnittstelle zum Signieren von Dokumenten und Prüfen von Dokumentensignaturen ~~und zum Signieren von Binärstrings zum Zweck der externen Authentisierung~~.

[…]

~~*4.1.8.1.4. Externe Authentisierung*~~

~~⌦ **TIP1-A_5437 Signaturverfahren für externe Authentisierung**~~

~~Der Signaturdienst MUSS das Signaturverfahren PKCS#1 entsprechend TAB_KON_780 – Signaturverfahren Externe Authentisierung unterstützen. ⌫~~

~~**Tabelle 7: TAB_KON_780 – Signaturverfahren Externe Authentisierung**~~

~~**Signaturformat**~~	~~**Standard**~~	~~**Dokumentformate**~~	~~**QES/ nonQES**~~	~~**Bemerkung**~~

~~**PKCS#1 (V2.1)**~~	~~[RFC3447]~~	~~Binär~~	~~nonQES~~	~~Dieses Signaturformat DARF NUR in Verbindung mit dem zur Authentisierung vorgesehenen Schlüssel des HBAx und des SM-B genutzt werden. Die Nutzung ist auf Dokumente (Hash) von maximal 512 bit Länge beschränkt.~~

~~**TIP1-A_5149 PKCS#1-Schnittstelle nur für Authentisierung mit HBAx und SM-B nutzen**~~

~~Der Hersteller des Konnektors MUSS den Anwender (Clientsystem) im Handbuch des Konnektors geeignet und ausreichend darüber informieren, dass das Signaturformat PKCS#1 nur zu Authentisierungszwecken mit dem Authentisierungsschlüssel des HBAx und des SM-B verwendet werden darf.~~

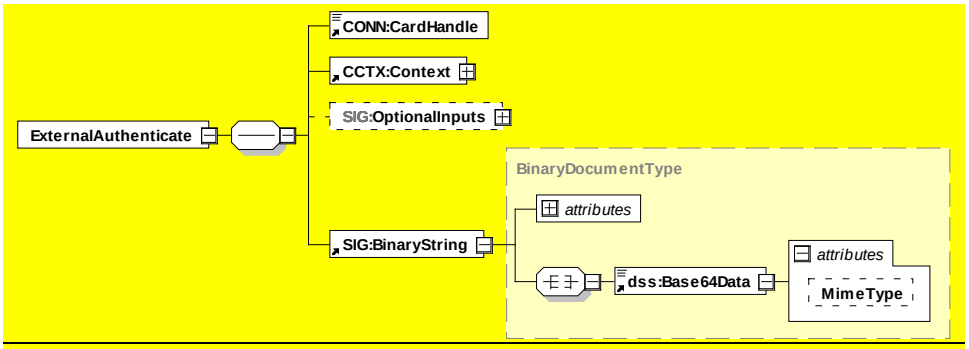
[…]

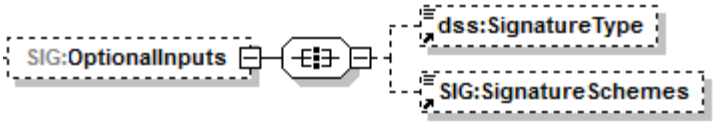
~~*4.1.8.5.5 ExternalAuthenticate*~~

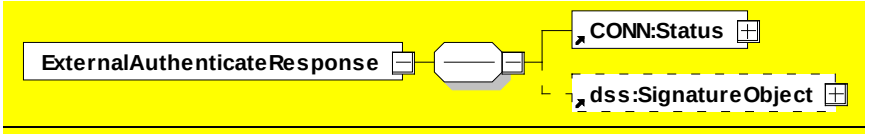
~~**TIP1-A_5439 Operation ExternalAuthenticate**~~

~~Der Authentifizierungsdienst des Konnektors MUSS an der Clientschnittstelle eine Operation ExternalAuthenticate anbieten.~~

~~**Tabelle 8: TAB_KON_781 Operation ExternalAuthenticate**~~

~~**Name**~~	~~`ExternalAuthenticate`~~	
~~**Beschreibung**~~	~~Diese Operation versieht einen Binärstring der maximalen Länge 512 Bit mit einer nicht-qualifizierten elektronischen Signatur (nonQES).~~ ~~Dazu wird das Signaturverfahren PKCS#1 verwendet. Das AUT-Zertifikat der SM-B und das AUT-Zertifikat des HBAx werden unterstützt.~~	
~~**Aufrufparameter**~~		
	~~Name~~	~~Beschreibung~~
	~~`CONN:CardHandle`~~	~~Identifiziert die zu verwendende Signaturkarte.~~ ~~Die Operation unterstützt HBAx und SM-B.~~

	~~CCTX:Context~~	~~Aufrufkontext für HBAx:~~ ~~MandantId, ClientSystemId, WorkplaceId, UserId verpflichtend~~ ~~Aufrufkontext für SM-B:~~ ~~MandantId, ClientSystemId, WorkplaceId verpflichtend; UserId nicht ausgewertet~~
	~~SIG:OptionalInputs~~	~~Enthält optionale Eingangsparameter:~~ 
	~~SIG:BinaryString~~	~~Dieses Element enthält im Kindelement dss:Base64Data den zu signierenden Binärstring.~~ ~~Das XML-Attribut SIG:BinaryString/dss:Base64Data/@MimeType MUSS den Wert "application/octet-stream" haben.~~ ~~Die maximale Länge des Binärstrings beträgt 512 Bit entsprechend der maximal zu erwartenden Hash-Größe.~~ ~~Aus der Länge des Binärstrings wird auf das verwendete Hashverfahren geschlossen. Es werden folgende Längen unterstützt:~~ ~~• 256 Bit: SHA-256 (OID 2.16.840.1.101.3.4.2.1)~~ ~~• 384 Bit: SHA-384 (OID 2.16.840.1.101.3.4.2.2)~~ ~~• 512 Bit: SHA-512 (OID 2.16.840.1.101.3.4.2.3)~~ ~~Im Falle des Signaturverfahrens RSASSA-PKCS1-v1_5 werden SHA-256, SHA-384 und SHA-512 unterstützt.~~ ~~Im Falle des Signaturverfahrens RSASSA-PSS wird SHA-256 unterstützt.~~ ~~Für die Signaturerstellung gilt:~~ ~~• Im Falle des Signaturverfahrens RSASSA-PKCS1-v1_5 beginnt der Konnektor die Ausführung der Methode EMSA-PKCS1-v1_5-ENCODE nach [RFC3447], Abschnitt 9.2, mit Schritt 2, Erstellung des DigestInfo-Datenfeldes.~~ ~~• Im Falle des Signaturverfahrens RSASSA-PSS beginnt der Konnektor die Ausführung der Methode EMSA-PSS-ENCODE nach [RFC3447], Abschnitt 9.1.1, mit Schritt 3.~~
	~~dss:SignatureType~~	~~Durch dieses in [OASIS-DSS] (Abschnitt 3.5.1) beschriebene Element wird der Typ der zu erzeugenden Signatur bestimmt. Als Signaturtyp wird unterstützt :~~ ~~• **PKCS#1-Signatur**~~ ~~Durch Übergabe der URI urn:ietf:rfc:3447 wird eine PKCS#1 (Version 2.1) Signatur gemäß [RFC3447] erzeugt, die als dss:Base64Signature mit der oben genannten URI zurückgeliefert wird.~~ ~~Andere SignatureType-Angaben führen zu einer Fehlermeldung 4111 (Ungültiger Signaturtyp oder~~

		~~Signaturvariante).~~ ~~Fehlt dieses Element, so wird ebenfalls der Signaturtyp PKCS#1-Signatur verwendet.~~
	~~`SIG:SignatureSchemes`~~	~~Durch dieses Element wird für PKCS#1-Signaturen zwischen den folgenden SignatureScheme-Optionen unterschieden:~~ ~~• RSASSA-PSS~~ ~~• RSASSA-PKCS1-v1_5~~ ~~Fehlt dieses Element, so wird als Default-SignatureScheme RSASSA-PSS gewählt.~~
~~**Rückgabe**~~		
	~~`CONN:Status`~~	~~Enthält den Status der ausgeführten Operation.~~
	~~`dss:SignatureObject`~~	~~Enthält im Erfolgsfall die erzeugte Signatur in Form eines `dss:SignatureObject`-Elements gemäß [OASIS-DSS] (Abschnitt 3.2).~~ ~~Der Signaturwert wird im XML-Element `dss:SignatureObject/dss:Base64Signature` übergeben. Das XML-Attribut `dss:SignatureObject/dss:Base64Signature/@Type` kennzeichnet durch den Wert urn:ietf:rfc:3447 den Signatur-Typ.~~ ~~Die XML-Elemente~~ ~~`dss:SignatureObject/ds:Signature`~~ ~~`dss:SignatureObject/dss:Timestamp`~~ ~~`dss:SignatureObject/dss:SignaturePtr`~~ ~~`dss:SignatureObject/dss:Other`~~ ~~werden nicht verwendet.~~
~~**Vorbedingungen**~~	~~Keine~~	
~~**Nachbedingungen**~~	~~Keine~~	

~~Der Ablauf der Operation ExternalAuthenticate ist in Tabelle TAB_KON_782 beschrieben:~~

~~**Tabelle 9: TAB_KON_782 Ablauf Operation ExternalAuthenticate**~~

~~**Nr.**~~	~~**Aufruf Technischer Use Case oder Interne Operation**~~	~~**Beschreibung**~~
~~5.~~	~~checkArguments~~	~~Alle übergebenen Parameterwerte werden auf Konsistenz und Gültigkeit überprüft. Treten hierbei Fehler auf, so bricht die Operation mit Fehler 4000 ab.~~
~~6.~~	~~TUC_KON_000 „Prüfe Zugriffs-~~	~~Die Prüfung erfolgt durch den Aufruf~~

Nr.	Aufruf Technischer Use Case oder Interne Operation	Beschreibung
	berechtigung“	TUC_KON_000 {$context.mandantId; $context.clientsystemId; $context.workplaceId; $context.userId; $cardHandle} Tritt bei der Prüfung ein Fehler auf, bricht die Operation mit Fehlercode aus TUC_KON_000 ab.
7.	TUC_KON_026 „Liefere CardSession“	Ermittle CardSession über TUC_KON_026 { MandantId, CsId, CardHandle, UserId }

8.	TUC_KON_218 „Signiere“	Signaturberechnung durch Aufruf des TUC_KON_218 {
		PinRef = PIN.CH bzw. PIN.SMC;
		KeyRef = PrK.HP.AUT bzw. PrK.HCI.AUT;
		AlgorithmusID = signPKCS1_V1_5 oder signPSS;
		DTBS = Binärstring
		}

Tabelle 10: TAB_KON_783 Übersicht Fehler Operation ExternalAuthenticate

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen TUCs können folgende weiteren Fehlercodes auftreten:			
4000	Technical	Error	Syntaxfehler
4058	Security	Error	Aufruf nicht zulässig

Die folgende Tabelle führt die zulässigen privaten Schlüssel für die Operation ExternalAuthenticate auf:

Tabelle 11: TAB_KON_784 Privater Schlüssel je Karte für ExternalAuthenticate

Karte	Schlüssel
SM-B	PrK.HCI.AUT in DF.ESIGN
HBAx	PrK.HP.AUT in DF.ESIGN

In Kapitel 3.3 wird Tabelle TAB_KON_504 korrigiert

Tabelle 12 TAB_KON_504 Ausführungserlaubnis für Dienste in kritischen Fehlerzuständen

	EC_Software_Integrity_Check_Failed	EC_Random_Generator_Not_Reliable	EC_Security_Log_Not_Writable	EC_Time_Sync_Pending_Critical	EC_Time_Difference_Intolerable	EC_CRL_Out_Of_Date	EC_TSL_Out_Of_Date_Beyond_Grace	EC_TSL_Trust_Anchor_Out_Of_Date	EC_Firewall_Not_Reliable	EC_Secure_KeyStore_Not_Available
Technische Use Cases (TUCs) der Basisdienste relevant für Fachanwendung und die Kommunikation mit Bestandsnetz und SIS										
Zugriffsberechtigungsdienst										
TUC_KON_000 PrüfeAufrufkontext	-	x	x	x	x	x	x	x	x	x
Dienstverzeichnisdienst										
TUC_KON_041 Einbringen der Endpunktinformationen während der Bootup-Phase	-	-	-	x	x	x	x	x	x	x
Kartenterminaldienst										
TUC_KON_051 Mit Anwender über Kartenterminal interagieren	-	-	-	-	-	x	x	x	-	-
Kartendienst										
TUC_KON_005 Card-to-Card authentisieren	-	-	-	-	-	x	x	x	-	-
TUC_KON_006 Datenzugriffsaudit eGK schreiben	-	-	-	-	-	x	x	x	-	-
TUC_KON_018 eGK-Sperrung prüfen	-	-	-	-	-	x	x	x	-	-
TUC_KON_024 Karte zurücksetzen	-	-	-	-	-	x	x	x	-	-
TUC_KON_026 Liefere CardSession	-	-	-	-	-	x	-	x	-	-
TUC_KON_200 SendeAPDU	-	-	-	-	-	x	x	x	-	-
TUC_KON_202 LeseDatei	-	-	-	-	-	x	x	x	-	-
TUC_KON_203 SchreibeDatei	-	-	-	-	-	x	x	x	-	-
TUC_KON_209 LeseRecord	-	-	-	-	-	x	x	x	-	-
Systeminformationsdienst										
TUC_KON_256 Systemereignis absetzen	-	x	x	x	x	x	x	x	x	x
Verschlüsselungsdienst										
TUC_KON_072 Daten symmetrisch verschlüsseln	-	-	-	x	x	x	x	x	-	-
TUC_KON_073 Daten symmetrisch entschlüsseln	-	-	-	x	x	x	x	x	-	-
Zertifikatsdienst										
TUC_KON_034 Zertifikatsinformationen extrahieren	-	-	-	x	x	x	x	x	-	-
Protokollierungsdienst										

	EC_Software_Integrity_Check_Failed	EC_Random_Generator_Not_Reliable	EC_Security_Log_Not_Writable	EC_Time_Sync_Pending_Critical	EC_Time_Difference_Intolerable	EC_CRL_Out_Of_Date	EC_TSL_Out_Of_Date_Beyond_Grace	EC_TSL_Trust_Anchor_Out_Of_Date	EC_Firewall_Not_Reliable	EC_Secure_KeyStore_Not_Available
TUC_KON_271 Schreibe Protokolleintrag	-	x	x	x	x	x	x	x	x	x
TLS-Dienst										
TUC_KON_110 Kartenbasierte TLS-Verbindung aufbauen	-	-	-	-	-	-	-	-	-	-
Verbindung zum VPN-Konzentrator										
TUC_KON_321 Verbindung zu dem VPN-Konzentrator der TI aufbauen	-	-	-	-	-	-	-	-	-	-
TUC_KON_322 Verbindung zum dem VPN-Konzentrator des SIS aufbauen	-	-	-	-	-	-	-	-	-	-
Operationen der Basisdienste										
Kartendienst										
VerifyPin	-	-	-	-	-	x	x	x	-	-
UnblockPin	-	-	-	-	-	x	x	x	-	-
ChangePin	-	-	-	-	-	x	x	x	-	-
GetPinStatus	-	-	-	-	-	x	x	x	-	-
Systeminformationsdienst										
Schnittstelle der Ereignissenke	-	x	x	x	x	x	x	x	x	x
GetCardTerminals	-	x	x	x	x	x	x	x	x	x
GetCards	-	x	x	x	x	x	x	x	x	x
GetResourceInformation	-	x	x	x	x	x	x	x	x	x
Subscribe	-	x	x	x	x	x	x	x	x	x
RenewSubscription	-	x	x	x	x	x	x	x	x	x
Unsubscribe	-	x	x	x	x	x	x	x	x	x
GetSubscription	-	x	x	x	x	x	x	x	x	x
Verschlüsselungsdienst										
EncryptDocument	-	-	-	-	-	x	x	x	-	-
DecryptDocument	-	-	-	-	-	x	x	x	-	-
Signaturdienst										
SignDocument	-	-	-	-	-	x	x	x	-	-
VerifyDocument	-	-	-	-	-	x	x	x	-	-
GetJobNumber	-	-	-	-	-	x	x	x	-	-
StopSignature	-	-	-	-	-	x	x	x	-	-
Authentifizierungsdienst										
ExternalAuthenticate	-	-	-	-	-	x	x	x	-	-
Zertifikatsdienst										
ReadCardCertificate	-	-	-	-	-	x	x	x	x	x
CheckCertificateExpiration	-	-	-	-	-	x	x	x	x	x
VerifyCertificate	-	-	-	-	-	x	x	x	x	x
Zeitdienst										

			EC_Software_Integrity_Check_Failed	EC_Random_Generator_Not_Reliable	EC_Security_Log_Not_Writable	EC_Time_Sync_Pending_Critical	EC_Time_Difference_Intolerable	EC_CRL_Out_Of_Date	EC_TSL_Out_Of_Date_Beyond_Grace	EC_TSL_Trust_Anchor_Out_Of_Date	EC_Firewall_Not_Reliable	EC_Secure_KeyStore_Not_Available
		I_NTP_Time_Information	-	-	-	-	-	x	x	x	-	x
	Konnektormanagement											
		Softwareaktualisierung	x	x	x	x	x	x	x	x	x	x
		Protokolleinsicht	x	x	x	x	x	x	x	x	x	x
		Werksreset	x	x	x	x	x	x	x	x	x	x
		Sonstiges	-	x	x	x	x	x	x	x	x	x

In Kapitel 4.1.5.4.16 wird Tabelle TAB_KON_231 korrigiert

Tabelle 13: TAB_KON_231 – TUC_KON_218 „Signiere“

Element	Beschreibung
Name	TUC_KON_218 „Signiere“
Beschreibung	Dieser Use Case beschreibt das Anwenden eines privaten Schlüssels einer Karte zur Signatur oder Authentisierung.
Auslöser	• Aufruf einer der Operationen SignDocument ~~oder ExternalAuthenticate~~ des Signaturdienstes oder ExternalAuthenticate des Authentifizierungsdienstes durch das Clientsystem. • Aufruf durch Fachmodul

In Anhang H – Mapping von „Architektur der TI-Plattform“ auf Konnektorspezifikation wird Tabelle TAB_KON_712 korrigiert:

Tabelle 14 - TAB_KON_712 Architektur der TI-Plattform, Berechtigt Clientsysteme

Interface	Operation	→	Funktionsmerkmal	Interface / TUC	Operation
I_Crypt_Operations	decrypt_Document	→	Verschlüsselungsdienst	EncryptionService	DecryptDocument
	encrypt_Document	→	Verschlüsselungsdienst	EncryptionService	EncryptDocument
I_DNS_Name_Resolution	get_FQDN	→	Namensdienst und Dienstlokalisierung	GetFQDN	I_DNS_Name_Resolution
	get_IP_Address	→	Namensdienst und Dienstlokalisierung	GetIPAddress	
I_IP_Transport	send_Data_External	→	Anbindung LAN/WAN		AFOs: Routing der IP-Pakete von Client --> VPN_SIS
I_KV_Card_Handling	discard_Card_Usage_Reference	→	---		*--- keine Umsetzung notwendig. Erfolgt implizit*
	get_Card_Usage_Reference	→	---		*--- keine Umsetzung notwendig. Erfolgt implizit*
I_KV_Card_Unlocking					
	change_PIN	→	Kartendienst	CardService	ChangePin
	get_PIN_Status	→	Kartendienst	CardService	GetPinStatus
	initialize_PIN	→	Kartendienst	CardService	ChangePin
	unblock_PIN	→	Kartendienst	CardService	UnblockPin
	verify_PIN	→	Kartendienst	CardService	VerifyPin
I_Poll_System_Information	get_Ressource_Information	→	Systeminformationsdienst	EventService	GetResourceInformation
	get_Ressource_List	→	Systeminformationsdienst	EventService	GetCardTerminals
	get_Ressource_List	→	Systeminformationsdienst	EventService	GetCards

© gematik –öffentlich_Entwurf

Interface	Operation	→	Funktionsmerkmal	Interface / TUC	Operation
I_Reg_Notification	register_for_Notifications	→	Systeminformationsdienst	EventService	Subscribe
		→	Systeminformationsdienst	EventService	Unsubscribe
		→	Systeminformationsdienst	EventService	GetSubscription
I_SAK_Operations	sign_Document_QES	→	Signaturdienst	SignatureService	SignDocument
	verify_Document_QES	→	Signaturdienst	SignatureService	VerifyDocument
I_Sign_Operations	sign_Document	→	Signaturdienst	SignatureService	SignDocument
	external_Authenticate	→	~~Signaturdienst~~ Authentifizierungsdienst	~~SignatureService~~ AuthSignatureService	ExternalAuthenticate
	verify_Document	→	Signaturdienst	SignatureService	VerifyDocument
	get_Certificate	→	Zertifikatsdienst	CertificateService	ReadCardCertificate
I_NTP_Time_Information	sync_Time	→	Zeitdienst		TIP1-A_2331 „I_NTP_Time_Information“
I_Directory_Query	search_Directory	→	LDAP-Proxy	LDAP-Operation	TIP1-A_5521

© gematik –öffentlich_Entwurf