
C_12154_Anlage

Inhaltsverzeichnis

1 Änderungen in gemSpec_Kon.....	2
1.1 4.1.5 Kartendienst.....	2
1.2 4.1.8 Signaturdienst.....	10
1.3 5.5.2 Dokumente der gematik.....	10
2 Änderungen in gemProdT_Kon_PTV6.....	11

1 Änderungen in gemSpec_Kon

1.1 4.1.5 Kartendienst

A_25970-01 - Operation StartCardSession

Der Konnektor MUSS an der Außenschnittstelle eine Operation StartCardSession, wie in Tabelle TAB_KON_273 Operation StartCardSession beschrieben, anbieten.

Tabelle 1: TAB_KON_273 Operation StartCardSession

Name		
Beschreibung	Die Operation nimmt eine operationsübergreifende Reservierung einer Karte vor und erzeugt eine unique Session ID zur Verwendung in Folgeaufrufen von Kartenoperationen.	
Aufrufparameter	Name	Beschreibung
	CCTX:Context	MandantId, CsId, Workplaceld verpflichtend
	CONN:CardHandle	Adressiert die Karte, mit der in Folgeaufrufen Kartenoperationen ausgeführt werden sollen. Die Operation MUSS die den Kartentyp eGK unterstützen. Die Operation DARF NICHT andere Kartentypen unterstützen. Wird die Operation mit einem nicht unterstützten Kartentypen aufgerufen, so MUSS der Konnektor die Bearbeitung mit dem Fehler 4209 abbrechen.
Rückgabe	Name	Beschreibung
	SessionID	UUID gem. [RFC4122]

Der Ablauf der Operation StartCardSession ist in Tabelle TAB_KON_274 Ablauf StartCardSession beschrieben.

Tabelle 2: TAB_KON_274 Ablauf StartCardSession

Nr.	Aufruf Technischer Use Case oder	Beschreibung
-----	----------------------------------	--------------

	Interne Operation	
1.	checkArguments	Die übergebenen Werte werden auf Konsistenz und Gültigkeit überprüft. Treten hierbei Fehler auf, so bricht die Operation mit Fehler 4000 ab.
2.	TUC_KON_000 „Prüfe Zugriffsberechtigung“	Prüfung der Zugriffsberechtigung durch den Aufruf <pre>TUC_KON_000 { mandantId = Context.mandantId; clientSystemId = Context.clientsystemId; workplaceId = Context.workplaceId; userId = Context.userId; CardHandle }</pre> 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 <pre>TUC_KON_026 { mandantId =\$context.mandantId; clientsystemId = \$context.clientsystemId; cardHandle = \$context.cardHandle; userId = \$context.userId }</pre>
4.	TUC_KON_223 „Starte Kartensitzung“	TUC_KON_223 { cardSession = CardSession }

Tabelle 3: TAB_KON_277 Fehlercodes StartCardSession

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen technischen Use Cases können folgende weitere Fehlercodes auftreten:			
4000	Technical	Error	Syntaxfehler
4209	Technical	Error	Kartentyp %CardType% wird durch diese Operation nicht unterstützt.

[<=, Konnektor Highspeed, Konnektor PTV6, Sich.techn. Eignung: CC-Evaluierung, funkt. Eignung: Test Produkt/FA, Sich.techn. Eignung: Prüfung durch CC-Prüfstelle]

A_25822-02 - Operation SecureSendAPDU

Der Konnektor MUSS an der Außenschnittstelle eine Operation SecureSendAPDU, wie in Tabelle TAB_KON_270 Operation SecureSendAPDU beschrieben, anbieten.

Tabelle 4: TAB_KON_270 Operation SecureSendAPDU

Name	SecureSendAPDU	
Beschreibung	<p>Die Operation sendet eine Liste von Kommando-APDUs an eine Karte und liefert die Liste der Rückgabe-APDUs. Die Operation MUSS nur eGK unterstützen.</p> <p>Die Zuordnung der Kommando-APDUs und der Rückgabe-APDUs ergibt sich aus der Reihenfolge in den Listen.</p> <p>In der Liste der Kommando-APDUs kann vor jedem Kommando-APDU eine Liste mit erwarteten StatusCodes zu dem jeweiligen Kommando-APDU mitgeschickt werden.</p> <p>Die Liste der Rückgabe-APDUs enthält ausschließlich Rückgabe-APDUs.</p> <p>Die Operation nimmt eine Liste von Kommando-APDUs entgegen, prüft die Signatur, sendet die Kommando-APDUs an eine Karte und liefert eine Liste von Antwort-APDUs zurück.</p> <p>Der Aufrufparameter SignedScenario TransactionData enthält unter anderem eine Liste. Jedes Element dieser Liste enthält eine Kommando-APDU und eine Menge von akzeptablen, erwarteten Statuswörtern.</p> <p>Der Rückgabeparameterwert SignedScenarioResponse TransactionResult enthält eine Liste von Antwort-APDUs. Die erste Antwort-APDU korrespondiert mit der ersten Kommando-APDU aus TransactionData. Die nächste Antwort-APDU korrespondiert mit der nächsten Kommando-APDU, usw.</p> <p>Die Operation MUSS den Kartentyp eGK unterstützen. Die Operation DARF NICHT andere Kartentypen unterstützen.</p>	
Aufrufparameter	Name	Beschreibung
	TransactionData SignedScenario	<p>TransactionData enthält ein Scenario wie in [api-popp] beschrieben (base64-codiert). Ein Scenario enthält eine Unterstruktur Scenario7816 und diese wiederum eine Liste bestehend aus Elementen (hier: Elements).</p> <p>Der Parameter SignedScenario TransactionData ist base64-codiert. Er enthält ein JWT gemäß [RFC-7519] wie in I_PoPP_Token_Generation.yaml (siehe [api-popp]) definiert. Dieses JWT ist identisch zu der Property signedScenario der Nachricht ConnectorScenarioMessage. Der Header dieses JWT folgt dem Format ConnectorScenarioHeaders. Die Payload dieses JWT ist eine StandardScenarioMessage.</p>

		<p>Hinweis: Das JWT folgt JWS Compact Serialization gemäß [RFC-7515]. Es enthält drei durch "." getrennte Abschnitte: <base64URL-kodierter Header>.<base64URL-kodierter Payload>.<base64URL-kodierte Signatur> Die Signaturerstellung erfolgt nach [RFC-7515].</p>
Rückgabe	Name	Beschreibung
	CONN:Status	Enthält den Ausführungsstatus der Operation
	TransactionResult SignedScenarioResponse	<p>Der Parameter SignedScenarioResponse TransactionResult ist base64 codiert. Er enthält die base64 codierte Liste der Rückgabe-APDUs (hier ResultList) wie in I_PoPP_Token_Generation.yaml ([api-popp]) definiert. Das Format der Liste ist in [api-popp] beschrieben entspricht der Property steps der Nachricht ScenarioResponseMessage.</p>
	TimeSpan	<p>Zeitspanne, in der der nächste Aufruf von SecureSendAPDU mit dem nächsten Szenario der Sequenz erfolgen muss TimeSpan = 0 zeigt das letzte Szenario der Sequenz an.</p>
Vorbedingung	keine	
Nachbedingung	keine	

Der Ablauf der Operation SecureSendAPDU ist in Tabelle TAB_KON_271 Ablauf SecureSendAPDU beschrieben.

Tabelle 5: TAB_KON_271 Ablauf SecureSendAPDU

Nr.	Aufruf Technischer Use Case oder Interne Operation	Beschreibung
1.	checkArguments	Die übergebenen Werte werden auf Konsistenz und Gültigkeit überprüft. Treten hierbei Fehler auf, so bricht die Operation mit Fehler 4000 ab.
2.	TUC_KON_161 „nonQES-Dokumentensignatur“	Die Signatur des JWT in SignedScenario TransactionData wird

	prüfen“ Signatur des JWT prüfen	<p>mathematisch geprüft. Tritt hierbei ein Fehler auf, bricht die Operation mit Fehlercode 4701 ab. Die dazugehörige Prüfung des Signaturzertifikats erfolgt durch Aufruf von TUC_KON_037 „Zertifikat prüfen“</p> <pre> certificate = ConnectorScenarioHeaders/x5c; qualifiedCheck = not_required; offlineAllowNoCheck = true; policyList = oid_zd_sig; intendedKeyUsage = intendedKeyUsage(C.ZD.SIG) intendedExtendedKeyUsage = empty; validationMode = OCSP; ocspResponse = ConnectorScenarioHeaders/stpl} </pre> <p>Die technische Rolle gemäß [gemSpec_OID#GS-A_4446] muss einem zulässigen Wert entsprechen. Die aktuell zulässigen Werte sind:</p> <ul style="list-style-type: none"> oid_popp <p>Tritt bei der Signaturprüfung ein Fehler auf, bricht die Operation ab. Tritt bei der Zertifikatsprüfung ein Fehler wegen falscher technischer Rolle auf, bricht die Operation mit Fehlercode 4700 ab.</p> <p>Die nonQES wird geprüft mittels TUC_KON_161 { certificate = X509Certificate; signature = SignatureObject; signedDocument = TransactionData; roleToMatch = oid_popp }.</p>
3.	TUC_KON_208 „Sende gesicherte APDU“	Die Kommando-APDUs werden an die Karte gesendet und das Ergebnis zurückgegeben mittels TUC_KON_208 { transactionData }

Tabelle 6: TAB_KON_272 Fehlercodes SecureSendAPDU

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen technischen Use Cases können folgende weitere Fehlercodes auftreten:			
4000	Technical	Error	Syntaxfehler

4209	Technical	Error	Kartentyp %CardType% wird durch diese Operation nicht unterstützt.
4700	Security	Error	Zertifikat enthält nicht die erwartete Rolle
4701	Security	Error	Signatur ungültig

【<=, Konnektor Highspeed, Konnektor PTV6, Sich.techn. Eignung: CC-Evaluierung, funkt. Eignung: Test Produkt/FA, Sich.techn. Eignung: Prüfung durch CC-Prüfstelle】

A_26069-01 - TUC KON_208 "Sende gesicherte APDU"

Der Konnektor MUSS den technischen Use Case „Sende gesicherte APDU“ gemäß TAB_KON_283 umsetzen.

Tabelle 7: TAB_KON_283 - TUC_KON_208 „Sende gesicherte APDU“

Element	Beschreibung
Name	TUC_KON_208 „Sende gesicherte APDU“
Beschreibung	Der technische Use Case löst Karten-Transaktionen aus. Aus übergebenen integritäts- und authentizitätsgeschützten Transaktionsdaten extrahiert er APDUs, sendet diese zur Ausführung an die Karte und gibt die Ergebnisse an den Aufrufer zurück. Als Kartentyp wird die eGK unterstützt.
Auslöser	Operation SecureSendAPDU
Vorbedingungen	keine
Eingangsdaten	<ul style="list-style-type: none"> signedScenario transactionData
Komponenten	Karte, Kartenterminal, Konnektor
Ausgangsdaten	<ul style="list-style-type: none"> signedScenarioResponse transactionResult timeSpan
Standardablauf	<ol style="list-style-type: none"> 1. Setze die Liste der erwarteten StatusCodes (\$StatusCodeList) durch Leeren der Liste und Einfügen eines Code-Elements mit Wert 0x9000 zurück. 2. Dekodiere transactionData (siehe Eingangsdaten) und Extrahiere das Scenario (\$signedScenario) aus den Eingangsdaten. als Scenario eine Liste von Elements, die jeweils SequenceCounter und SessionID enthalten. Das Format von \$signedScenario ist in [api-popp] beschrieben. 3. Prüfe, ob eine CardSession mit \$clientSessionID existiert (d. h. durch Aufruf von TUC_KON_223 persistiert wurde) 4. Prüfe, dass ein Lock für die durch \$clientSessionID identifizierte CardSession bereits existiert 5. Prüfe ob die laufende Sequenznummer \$sequenceCounter <ol style="list-style-type: none"> a. für den ersten Aufruf = 0 ist entweder die erste

	<p>(\$sequenceCounter = 0) bzw.</p> <p>b. oder für alle weiteren Aufrufe das Inkrement des vorhergehenden Aufrufes ist</p> <p>6. Für jedes \$item \$Element aus \$steps:</p> <p>a. Verarbeite erwartete Rückgabewerte</p> <ul style="list-style-type: none"> \$StatusCodeList = \$item.expectedStatusWords Falls \$Element eine Liste von erwarteten StatusCodes (ExpectedStatusWords) ist \$StatusCodeList = "\$Element" <p>b. Verarbeite Kommando-APDU: Falls \$Element eine Kommando-APDU (CommandAPDU) ist</p> <ul style="list-style-type: none"> Ermittle \$responseAPDU für das \$item.commandApdu\$Element mittels Aufruf von TUC_KON_200 { <ul style="list-style-type: none"> cardSession = "\$cardSession"; ctId nicht übergeben; commandAPDU = "\$item.commandApdu \$Element" } Füge Hänge \$responseAPDU an das Ende von signedScenarioResponse.ResponseApduList transactionResult an der ResultList Falls Status der \$responseAPDU nicht in \$StatusCodeList <ul style="list-style-type: none"> nimm die Warnung 4284 in die Antwort auf verlasse die Schleife <p>c. Falls \$Element eine Logging-Information (LoggingInformation) ist</p> <ul style="list-style-type: none"> führe für das \$Element keine Aktionen durch <p>d. Falls \$item \$Element ein anderes Subelement als die in a)-b) genannten enthält aufgezählten ist</p> <ul style="list-style-type: none"> führe für das Subelement keine Aktionen durch <p>7. Falls \$signedScenario.timeSpan = 0 (letztes Szenario): stoppe Kartensitzung durch TUC_KON_224 { sessionId = \$clientSessionID" } Andernfalls: Setze APDU-Szenario-Timer auf Wert von \$signedScenario.timeSpan</p> <p>8. Setze signedScenarioResponse.timeSpan = \$signedScenario.timeSpan</p>
Varianten/ Alternativen	Keine
Fehlerfälle	<p>(->2) Die dekodierten Eingabeparameter sind nicht nach [api-popp] validierbar: Fehlercode 4286</p> <p>(->3)\$clientSessionID existiert nicht: Fehlercode 4288</p> <p>(->4) Es ist kein Lock für\$clientSessionID gesetzt: Fehlercode 4289</p> <p>(->5) Die laufende Sequenznummer \$sequenceCounter ist für den ersten Aufruf != 0 bzw. die erste und nicht das Inkrement des vorhergehenden Aufrufes: Fehlercode 4285</p>

	In Fehlerfällen ab Schritt 5): TUC_KON_224 { sessionID = \$clientSessionID }
--	--

Tabelle 8: TAB_KON_280 - Fehlercodes TUC_KON_223 „Sende gesicherte APDU“

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen technischen Use Cases können folgende weitere Fehlercodes auftreten:			
4284	Technical	Warning	APDU konnte nicht verarbeitet werden.
4285	Technical	Error	Unerwartetes Sequence-Element
4286	Technical	Error	Inhalt von TransactionData nicht valide
4288	Technical	Error	Unbekannte Session ID
4289	Technical	Error	Karte nicht reserviert

【<=, Konnektor Highspeed, Konnektor PTV6, Sich.techn. Eignung: CC-Evaluierung, funkt. Eignung: Test Produkt/FA, Sich.techn. Eignung: Prüfung durch CC-Prüfstelle】

1.2 4.1.8 Signaturdienst

Die Erweiterung von TUC_KON_161 für die Rollenprüfung entfällt.

Die Anforderung TIP1-A_4654-06 wird storniert. Die Anforderung TIP1-A_4654-05 wird verbindlich.

1.3 5.5.2 Dokumente der gematik

[Quelle]	Herausgeber: Titel
[api-popp]	https://github.com/gematik/api-popp Die aktuell gültige Version wird im Produktsteckbrief referenziert.
[api-telematik]	https://github.com/gematik/api-telematik Die aktuell gültige Version wird im Produktsteckbrief referenziert.

2 Änderungen in gemProdT_Kon_PTV6

Tabelle 9: Mitgeltende Dokumente und Web-Inhalte

Quelle	Herausgeber: Bezeichnung / URL	Version/Branch/Tag
api-telematik	<p><u>Hinweis</u>: Die Version für die Vorveröffentlichung von [api-telematik] befindet sich zur Zeit unter diesem Link: https://github.com/gematik/api-telematik/pull/35</p> <p>Der folgende Link wird ab der Veröffentlichung gültig sein: https://github.com/gematik/api-telematik/releases/tag/6.0.1 https://github.com/gematik/api-telematik/releases/tag/6.0.0-2-(gültig-ab-Ende-KW-32/2024)</p>	6.0.1 6.0.0-2
api-popp	<p><u>Hinweis</u>: Die Version für die Vorveröffentlichung von [api-popp] befindet sich zur Zeit unter diesem Link: https://github.com/gematik/api-popp/pull/6</p> <p>Der folgende Link wird ab der Veröffentlichung gültig sein: https://github.com/gematik/api-popp/releases/tag/3.0.0</p>	3.0.0