EMV®

Secure Remote Commerce

Specification – JavaScript SDK

Version 1.2

June 2021

Legal Notice

Revision Log – Version 1.2

The following changes have been made to the document since the publication of version 1.1.

- Editorial changes to Section 1 Introduction
- Addition of Section 1.2 Constraints
- Introduction of the concept of the SRC Specifications, encompassing the suite of SRC documents (Section 1.4.2 Published EMVCo Documents)
- Changes to the descriptions of the SDK Methods in Section 2 Web Client SDK to bring consistency across the SRC Specifications
- Changes to the descriptions in the tables in Section 2 Web Client SDK to bring consistency across the SRC Specifications
- Deletion of the appInstanceId parameter in three SDK methods: Is Recognized (Section 2.3), Checkout (Section 2.9) and Unbind AppInstance (Section 2.11)

Contents

Tables

1 Introduction

Secure Remote Commerce (SRC) is an evolution of remote commerce that provides for secure and interoperable card acceptance established through a standard specification.

This document, the EMV Secure Remote Commerce Specification – JavaScript SDK, (hereafter the "SRC JavaScript SDK Specification"), contains the definition of the SRC JavaScript SDK for the SRC Initiator. The SDK will be provided by each supported SRC System to be incorporated into the SRC Initiator integration software provided for Digital Payment Applications (e.g. SRC Initiator web application).

It is intended to be used in conjunction with the SRC Specifications (see Section 1.4.2 Published EMVCo Documents).

1.1 Scope

The SRC JavaScript SDK Specification defines a set of JavaScript methods that can be used by a web client to perform SRC operations. The specification defines the method names, request parameters, response attributes and possible errors. It does not define how an implementation of the SDK interacts with the SRC System. This is SRC System proprietary and out of scope.

1.2 Constraints

The SRC JavaScript SDK Specification is designed to work within the constraints described in the SRC Core Specification. In particular, the SRC JavaScript SDK Specification or any implementation of the SRC JavaScript SDK Specification is not intended to replace or interfere with any international, regional, national or local laws and regulations; those governing requirements supersede any industry standards.

1.3 Audience

This document is intended for use by SRC Systems and SRC System Participants.

1.4 References

The latest version of any reference, including all published amendments, shall apply unless a publication date is explicitly stated.

1.4.1 Normative References

The standards in Table 1.1 may be associated with SRC JavaScript SDK Specification.

Table 1.1: Normative References

Reference	Publication Name
ISO 3166	Country Codes — ISO 3166
ISO 4217	Currency Codes — ISO 4217
ISO/IEC 7812	Identification cards — Identification of issuers
RFC 7515	JSON Web Signature
RFC 7516	JSON Web Encryption
RFC 7519	JSON Web Token

1.4.2 Published EMVCo Documents

The documents in Table 1.2 are related to or are associated with SRC and are located at www.emvco.com.

Table 1.2: EMVCo References

Reference	Publication Name
SRC Core Specification	EMV® Secure Remote Commerce Specification
SRC Reproduction Requirements	EMV® Secure Remote Commerce (SRC): Click to Pay Icon Reproduction Requirements
SRC UI Guidelines and Requirements	EMV® Secure Remote Commerce Specification – User Interface Guidelines and Requirements
SRC API	EMV® Secure Remote Commerce Specification – API
SRC Data Dictionary	EMV® Secure Remote Commerce Data Dictionary
SRC Version Management	EMV® Secure Remote Commerce Version Management for SRC API and SRC JavaScript SDK Specifications

Collectively, the term SRC Specifications refers to:

- SRC Core Specification
- SRC Reproduction Requirements
- SRC UI Guidelines and Requirements
- SRC API
- SRC JavaScript SDK (this document)
- SRC Data Dictionary
- SRC Version Management

1.5 Definitions

For the definition of the terms used in the SRC JavaScript SDK Specification, refer to Table 1.3: Definitions in the SRC Core Specification. For definitions of data elements refer to the SRC API or the SRC Data Dictionary.

1.6 Notational Conventions

1.6.1 Abbreviations

For the definition of the abbreviations used in the SRC JavaScript SDK Specification, refer to section 1.9.1 Abbreviations in the SRC Core Specification.

1.6.2 Terminology and Conventions

For the definition of the terminology and conventions used in the SRC JavaScript SDK Specification, refer to section 1.9.2 Terminology and Conventions in the SRC Core Specification.

2 Web Client SDK

2.1 Data Elements

All primitive and composite data elements are as defined in the SRC API or SRC Data Dictionary. The request parameters and response attributes (returned if processing is successful) are provided in a single JSON object.

In the request parameters and response attributes tables for each SDK method, the column headed R/C/O in each table refers to whether the parameter/attribute is required, conditional or optional. The following notation is used:

- R = Required – always present
- C = Conditional – present under certain conditions (as specified in the description)
- O = Optional – can be present

2.2 Initialize SRC SDK

This method initialises each SRC System's SDK in a common state. It must be called for each JavaScript SDK incorporated into the SRC Initiator integration software and before any other methods.

Table 2.2.1: Initialize SRC SDK Method

```
init({

      required String srcInitiatorId;
      conditional String srcDpaId;
      optional String srciTransactionId;
      optional DpaTransactionOptions dpaTransactionOptions;
      conditional DpaData dpaData;
      optional String version;
      optional AcceptanceChannelRelatedData
         acceptanceChannelRelatedData;
})
```

```
// Response - empty
```

2.2.1 Request Parameters

The request parameters are given in Table 2.2.2.

Table 2.2.2: Request Parameters

Name	R/C/O	Description
srcInitiatorId Type: String	R	The reference identifier generated by the SRC System during Onboarding of the SRC Initiator to the SRC System
srcDpaId Type: String	C	The reference identifier that may have been generated either by the SRC Initiator or by the SRC System. During DPA Registration with the SRC System (if it occurs) one, or both of these, value(s) will identify the DPA for subsequent initialisations **Conditionality**: Required when `dpaData` is not supplied. When both the `srcDpaId` and `dpaData` are supplied, then an SRC System can (as an implementation choice) choose how each should be used
srciTransactionId Type: String	O	Transaction-unique identifier assigned by the SRC Initiator
dpaTransactionOptions Type:DpaTransactionOptions	O	These options can be used to override transaction options for the DPA that were configured during the DPA Registration
dpaData Type: DpaData	C	Present when a DPA has not been registered with the SRC System or can be present in order to dynamically update previously registered `DPA Data` in the SRC System (e.g. presentation name) **Conditionality**: Required when `srcDpaId` is not supplied. When both `srcDpaId` and `dpaData` are supplied, then an SRC System can (as an implementation choice) choose how each should be used

Name	R/C/O	Description
version Type: String	O	SDK versioning
acceptanceChannelRelatedData Type: AcceptanceChannelRelatedData	O	Passes specific acceptance channel data. Refer to the SRC API Specification for details

2.2.2 Response Attributes

This method does not have any response attributes.

2.2.3 Application Errors

The application errors are given in Table 2.2.3.

Table 2.2.3: Application Errors

Reason Code	Description
SRCI_ID_MISSING	The `srcInitiatorId` parameter is missing
DPA_ID_OR_DATA_MISSING	The `srcDpaId` and `dpaData` parameters are missing: at least one must be supplied

2.3 Is Recognized

This method uses a Device Identity (derived from a First Party Token) to determine whether it is bound to an SRC Profile and, if so, returns a Federated ID Token. or client application.

If a Federated ID Token is returned, the SRC Initiator integration software may then provide this Federated ID Token in the getSrcProfile() method of the SDKs of the other SRC Systems.

Table 2.3.1: Is Recognized Method

```
isRecognized({

})
```

```
// Response
dictionary {

      required Boolean recognized;
      conditional List <JWT> idTokens;
}
```

2.3.1 Request Parameters

This method does not have any request attributes.

2.3.2 Response Attributes

The response attributes are given in Table 2.3.2.

Table 2.3.2: Response Attributes

Name	R/C/O	Description
recognized Type: Boolean	R	Flag indicating whether the Consumer Device (e.g. browser or client application) is recognised by the SRC System
idTokens Type: List<JWT>	C	List of Federated ID Tokens identifying the primary Consumer Identity bound to the recognised SRC Profiles **Conditionality**: Required when the value of `recognized` is set to `true` (i.e. one or more SRC Profiles are recognised)

2.3.3 Application Errors

The application errors are given in Table 2.3.3.

Table 2.3.3: Application Errors

Reason Code	Description
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)

2.4 Get SRC Profile

This method takes a list of Federated ID Tokens and returns SRC Profile data to enable card selection. The SRC Initiator aggregates Federated ID Tokens received from multiple SRC Systems and provides these to individual SRC Systems in order to fetch a complete card list.

Note: The `authorization` data element will not be present in the `profiles` returned to the SRC Initiator.

Table 2.4.1: Get SRC Profile Method

```
getSrcProfile({

      optional List<JWT> idTokens;
})
```

```
// Response
dictionary {
      required List<SrcProfile> profiles;
      optional String srcCorrelationId;
}
```

2.4.1 Request Parameters

The request parameters are given in Table 2.4.2.

Table 2.4.2: Request Parameters

Name	R/C/O	Description
idTokens Type: List<JWT>	O	List of the Federated ID Tokens received from one or more SRC Systems

2.4.2 Response Attributes

The response attributes are given in Table 2.4.3.

Table 2.4.3: Response Attributes

Name	R/C/O	Description
profiles Type: List<SrcProfile>	R	List of SRC Profile(s) associated with each recognised Consumer Identity. If no SRC Profiles are recognised, then an empty list is returned
srcCorrelationId Type: String	O	Reference identifier returned by the SRC System.

2.4.3 Application Errors

The application errors are given in Table 2.4.4.

Table 2.4.4: Application Errors

Reason Code	Description
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)

2.5 Identity Lookup

This method checks whether a specified Consumer Identity (email address or mobile phone number) is known to the SRC System.

Table 2.5.1: Identity Lookup Method

```
identityLookup({

      required ConsumerIdentity consumerIdentity;

})
```

```
// Response
dictionary {
      required Boolean consumerPresent;
      optional List<IdentityValidationChannel>
        supportedValidationChannels
}
```

2.5.1 Request Parameters

The request parameters are given in Table 2.5.2.

Table 2.5.2: Request Parameters

Name	R/C/O	Description
consumerIdentity Type: ConsumerIdentity	R	An email or phone number that will be used to determine whether an SRC Profile with this primary Consumer Identity is present

2.5.2 Response Attributes

The response attributes are given in Table 2.5.3.

Table 2.5.3: Response Attributes

Name	R/C/O	Description
consumerPresent Type: Boolean	R	Flag indicating whether an SRC Profile with the provided Consumer Identity is present
supportedValidationChannels Type: List<IdentityValidationChannel>	O	List of channels that can be used to perform identity validation. If returned by the SRC System, these choices could be presented to the Consumer

2.5.3 Application Errors

The application errors are given in Table 2.5.4.

Table 2.5.4: Application Errors

Reason Code	Description
CONSUMER_ID_MISSING	The `consumerIdentity` parameter was missing
ID_FORMAT_UNSUPPORTED	Unsupported Consumer Identity type
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)

2.6 Initiate Identity Validation

This method initiates a process to validate that the Consumer is in possession of, or has access to, the Consumer Identity claimed.

Table 2.6.1: Initiate Identity Validation Method

```
initiateIdentityValidation({

      optional String requestedValidationChannelId;
})
```

```
// Response
dictionary {

      required String maskedValidationChannel;
      optional String validationMessage;
      optional List<IdentityValidationChannel>
         supportedValidationChannels;
}
```

2.6.1 Request Parameters

The request parameters are given in Table 2.6.2.

Table 2.6.2: Request Parameters

Name	R/C/O	Description
requestedValidationChannelId Type: String	O	Identifier of the channel over which the identity validation should be initiated

2.6.2 Response Attributes

The response attributes are given in Table 2.6.3.

Table 2.6.3: Response Attributes

Name	R/C/O	Description
maskedValidationChannel Type: String	R	Masked value of the channel (e.g. email/phone) that the SRC System used to deliver the validation data (e.g. OTP)
validationMessage Type: String	O	Message returned by the SRC System to provide a locale-specific advisory to the Consumer about the identity validation process
supportedValidationChannels Type: List<IdentityValidationChannel>	O	List of additional channels that are supported and can be used to perform identity validation. If returned by the SRC System, these choices may be presented to the Consumer

2.6.3 Application Errors

The application errors are given in Table 2.6.4.

Table 2.6.4: Application Errors

Reason Code	Description
OTP_SEND_FAILED	The validation data could not be sent to the recipient
RETRIES_EXCEEDED	The limit for the number of retries for validation data generation was exceeded

Reason Code	Description
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)

2.7 Complete Identity Validation

This method determines whether data, provided by the Consumer as part of a second step of an identity validation process, is valid. It can also be used to check whether an out-of-band service was successful.

Table 2.7.1: Complete Identity Validation Method

```
completeIdentityValidation({

      conditional String validationData;
})
```

```
// Response
dictionary {

      required JWT idToken;
}
```

2.7.1 Request Parameters

The request parameters are given in Table 2.7.2.

Table 2.7.2: Request Parameters

Name	R/C/O	Description
validationData Type: String	C	The validation data (e.g. the OTP value) entered by the user **Conditionality**: Required when the content of the requested `identityValidationChannelType` was set to a value other than OUT_OF_BAND

2.7.2 Response Attributes

The response attributes are given in Table 2.7.3.

Table 2.7.3: Response Attributes

Name	R/C/O	Description
idToken Type: JWT	R	The Federated ID Token returned following successful validation of the `validationData`

2.7.3 Application Errors

The application errors are given in Table 2.7.4.

Table 2.7.4: Application Errors

Reason Code	Description
VALDATA_MISSING	The `validationData` parameter was missing
CODE_INVALID	The supplied `validationData` was invalid
CODE_EXPIRED	The `validationData` is expired
RETRIES_EXCEEDED	The limit for the number of retries for `validationData` generation has been reached
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)
VALIDATION_IN_PROGRESS	The requested `identityValidationChannelType` was set to OUT_OF_BAND and no result is available

2.8 Enrol Card

This method enrols a new PAN to the SRC System during checkout. The PAN may be enrolled to an existing / identified SRC Profile, or to a newly-created SRC Profile, or (in the case of guest checkout) may not be added to an SRC Profile at all.

Table 2.8.1: Enrol Method

```
enrollCard({
      optional String srciTransactionId;
      required JWE encryptedCard;
      optional JSONObject threeDsInputData;
      conditional JWT idToken;
      optional JWE<Consumer> encryptedConsumer;
      optional JSONObject srcTokenRequestData;
      optional AssuranceData assuranceData;
})
```

```
// Response
dictionary {
      required MaskedCard maskedCard;
      optional String srcCorrelationId;
}
```

2.8.1 Request Parameters

The request parameters are given in Table 2.8.2.

Table 2.8.2: Request Parameters

Name	R/C/O	Description
srciTransactionId Type: String	O	Transaction-unique identifier assigned by the SRC Initiator
encryptedCard Type: JWE<Card>	R	The card being enrolled with the SRC System. Encrypted using a public key of the SRC System to which the card is being enrolled
threeDsInputData Type: JSONObject	O	Merchant provided 3-D Secure data

Name	R/C/O	Description
idToken Type: JWT	C	Federated ID Token used to check whether an SRC Profile exists and, if not, used by the DCF to provide the Consumer with hints to use a Consumer Identity that is consistent across SRC Systems **Conditionality:** Required when it is determined that the Consumer has previously enrolled with, and been recognised by, another SRC System
encryptedConsumer Type: JWE<Consumer>	O	Consumer information related to the card being enrolled. Encrypted using the public key of the SRC System to which the card is being enrolled
srcTokenRequestData Type: JSONObject	O	SRC System-specific data (provided by the merchant) to support a Token Request
assuranceData Type: AssuranceData	O	Assurance data related to the enrolment

2.8.2 Response Attributes

The response attributes are given in Table 2.8.3.

Table 2.8.3: Response Attributes

Name	R/C/O	Description
maskedCard Type: MaskedCard	R	Masked data related to the enrolled Digital Card
srcCorrelationId Type: String	O	A transaction-unique identifier that may be returned by the SRC System

2.8.3 Application Errors

The application errors are given in Table 2.8.4.

Table 2.8.4: Application Errors

Reason Code	Description
CARD_MISSING	The `encryptedCard` parameter is missing
CARD_ADD_FAILED	Unable to add the provided card
CARD_SECURITY_CODE_MISSING	Card security code must be supplied in the `encryptedCard` parameter
CARD_INVALID	Invalid `primaryAccountNumber`
CARD_EXP_INVALID	Invalid card expiry date
AUTH_INVALID	Invalid `idToken`
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)

2.9 Checkout

This method performs checkout using the specified Digital Card or PAN. If successful, the response contains summary checkout information and, conditionally, an encrypted payload signed by the SRC System containing PCI and/or PII data.

Table 2.9.1: Checkout Method

```
checkout({

      optional String srciTransactionId;
      conditional String srcCorrelationId;
      conditional String srcDigitalCardId;
      conditional JWE<Card> encryptedCard;
      optional JWE<Consumer> encryptedConsumer;
      conditional JWT idToken;
      conditional DpaTransactionOptions dpaTransactionOptions;
      optional PayloadTypeIndicator payloadTypeIndicatorCheckout;
      optional String recipientIdCheckout;
      optional PayloadTypeIndicator payloadTypeIndicatorPayload;
      optional String recipientIdPayload;
      optional AssuranceData assuranceData;
      optional SrciActionCode srciActionCode
      optional Window windowRef;
      optional AcceptanceChannelRelatedData
        acceptanceChannelRelatedData;
})
```

```
// Response
dictionary {
      required DcfActionCode dcfActionCode;
      conditional JWT idToken;
      conditional JWS<CheckoutPayloadResponse> checkoutResponse;
      optional Boolean unbindAppInstance;
}
```

2.9.1 Request Parameters

The request parameters are given in Table 2.9.2.

Table 2.9.2: Request Parameters

Name	R/C/O	Description
srciTransactionId Type: String	O	A transaction-unique identifier assigned by the SRC Initiator
srcCorrelationId Type: String	C	A reference identifier previously provided by the SRC System to which the card is being enrolled and/or with which checkout is occurring **Conditionality:** Required when the SRC System returned it in prior calls within the same transaction

Name	R/C/O	Description
srcDigitalCardId Type: String	C	A reference identifier of the card to be used for checkout **Conditionality:** Required for checkout when a Digital Card is selected from a Candidate List
encryptedCard Type: JWE<Card>	C	The card being enrolled with the SRC System. Encrypted using a public key of SRC System to which the card is being enrolled **Conditionality**: Required for a combined flow where this card is being enrolled during checkout
encryptedConsumer Type: JWE<Consumer>	O	Consumer information related to the card being enrolled. Encrypted using the public key of the SRC System to which the card is being enrolled
idToken Type: JWT	C	Federated ID Token used to check whether an SRC Profile exists and, if not, used by the DCF to provide the Consumer with hints to use a Consumer Identity that is consistent across SRC Systems **Conditionality:** Required when it is determined that the Consumer has previously enrolled with, and been recognised by, another SRC System
dpaTransactionOptions Type: DpaTransactionOptions	C	These options can be used to override transaction options for the DPA that were configured during the DPA Registration **Conditionality**: Required when not provided earlier in the init() method call and the DPA has not been Registered with the SRC System

Name	R/C/O	Description
payloadTypeIndicatorCheckout Type: PayloadTypeIndicator	O	Indicates the scope of the encrypted payload, if any, to be provided in the `checkoutResponse` attribute in the response to this method
recipientIdCheckout Type: String	O	Identifier of the ultimate recipient of the encrypted payload returned in the `checkoutResponse` attribute in the response to this method. Used by the SRC System to determine which key is used for encryption of the payload
payloadTypeIndicatorPayload Type: PayloadTypeIndicator	O	Indicates the type of payload to be provided when the payload is returned in a subsequent interaction (e.g. when the Get Payload operation is called)
recipientIdPayload Type: String	O	Identifier of the recipient that will subsequently request the encrypted payload. Used by the SRC System to identify which key to use for encryption of the payload
assuranceData Type: AssuranceData	O	Assurance data supplied to support risk management
srciActionCode Type: SrciActionCode	C	A code indicating a non-typical behaviour on the SRC Initiator that should be addressed by the DCF **Conditionality**: Required when non-typical behaviour has occurred
windowRef Type: Window	O	A handle to facilitate the DCF opening a custom URI in the popup/iframe window
acceptanceChannelRelatedData Type: AcceptanceChannelRelatedData	O	This field is used to pass along specific acceptance channel related data. Refer to the SRC API Specification for details

2.9.2 Response Attributes

The response attributes are given in Table 2.9.3.

Table 2.9.3: Response Attributes

Name	R/C/O	Description
dcfActionCode Type: DcfActionCode	R	A code indicating the behaviour to be handled by the SRC Initiator
idToken Type: JWT	C	A Federated ID Token related to the current SRC Profile **Conditionality**: Required when: • Requested by the DCF during its processing of the checkout session (e.g. DCF requesting identity validation be performed by the SRC System); *or* • The `unbindAppInstance` attribute returned in this response is set to `true`; *or* • A new / updated `idToken` is generated
checkoutResponse Type: JWS <CheckoutPayloadResponse>	C	Signed structure **Conditionality**: Required when the `dcfActionCode` is set to COMPLETE
unbindAppInstance Type: Boolean	O	Flag indicating whether the Consumer has chosen to be 'un-remembered' from the Consumer Device. The default value is assumed to be `false` if this field is not provided. If this attribute is set to `true`, then the SRC Initiator shall proceed to call the `unbindAppInstance()` method for all available SDKs with the `idToken` attribute returned in this response

2.9.3 Application Errors

The application errors are given in Table 2.9.4.

Table 2.9.4: Application Errors

Reason Code	Description
CARD_MISSING	The `srcDigitalCardId` or `encryptedCard` parameter was required but is missing
CARD_ADD_FAILED	Unable to add the card when combined flow (enrol and checkout) is occurring
CARD_SECURITY_CODE_ MISSING	Card security code must be supplied in the `encryptedCard` parameter when a combined flow (enrol and checkout) is occurring
CARD_INVALID	Invalid `primaryAccountNumber` when combined flow (enrol and checkout) is occurring
CARD_NOT_RECOGNIZED	The provided `srcDigitalCardId` was not recognised
CARD_EXP_INVALID	Invalid card expiry date
MERCHANT_DATA_INVALID	Merchant data is invalid
UNABLE_TO_CONNECT	Unable to connect to / Launch DCF
AUTH_INVALID	Invalid `idToken`
TERMS_AND_CONDITIONS_ NOT_ACCEPTED	Terms and Conditions not accepted
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)

2.10 Delete Card

This method deletes a Digital Card from an SRC Profile.

Table 2.10.1: Delete Card Method

```
deleteCard({
      required String srcDigitalCardId;
})
```

```
// Response
dictionary {
      optional String srcCorrelationId;
}
```

2.10.1 Request Parameters

The request parameters are given in Table 2.10.2.

Table 2.10.2: Request Parameters

Name	R/C/O	Description
srcDigitalCardId Type: String	R	A reference identifier of the card to be deleted

2.10.2 Response Attributes

The response parameters are given in Table 2.10.3.

Table 2.10.3: Response Attributes

Name	R/C/O	Description
srcCorrelationId Type: String	O	A transaction-unique identifier returned by the SRC System if this method is within a particular checkout transaction context

2.10.3 Application Errors

The application errors are given in Table 2.10.4.

Table 2.10.4: Application Errors

Reason Code	Description
CARDID_MISSING	The `srcDigitalCardId` parameter is missing
CARD_NOT_RECOGNIZED	The provided `srcDigitalCardId` was not recognised
AUTH_INVALID	The client does not have authorisation to perform the operation
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)

2.11 Unbind AppInstance

This method unbinds a Device Identity (an application instance) from an SRC Profile.

Table 2.11.1: Unbind AppInstance Method

```
unbindAppInstance({
      required List<JWT> idTokens;
})
```

```
// Response
dictionary {
      optional String srcCorrelationId;
}
```

2.11.1 Request Parameters

The request parameters are given in Table 2.11.2.

Table 2.11.2: Request Parameters

Name	R/C/O	Description
idTokens Type: List<JWT>	R	Each Federated ID Token indicates an SRC Profile(s) from which the Device Identity should be unbound

2.11.2 Response Attributes

The response parameters are given in Table 2.11.3.

Table 2.11.3: Response Attributes

Name	R/C/O	Description
srcCorrelationId Type: String	O	A transaction-unique identifier returned by the SRC System if this method is within a particular checkout transaction context

2.11.3 Application Errors

The application errors are given in Table 2.11.4.

Table 2.11.4: Application Errors

Reason Code	Description
AUTH_INVALID	Invalid `idToken`
ACCT_INACCESSIBLE	The SRC Profile exists but is not currently accessible (e.g. is locked)
AUTH_MISSING	Missing `idTokens` list, or empty `idTokens` list provided

2.12 Standard Errors

SDK method errors can be application errors or standard errors. An `error` object is passed for every error.

- Standard errors can be returned by any SDK method and should be handled in a common way. They are described in Table 2.12.1
- Application errors that are only returned for specific SDK methods are described in the application errors section for the specific SDK method

Table 2.12.1: Standard Errors

Reason Code	Description
UNKNOWN_ERROR	Unknown error
REQUEST_TIMEOUT	Request timeout
INVALID_PARAMETER	The value provided for one or more request parameters is considered invalid. This error is also generated in case of a missing, required, request parameter. Notes: • Whenever possible client-side validation of request parameters should be performed to avoid a round trip to the server. Simple validation constraints are documented as part of the SRC API • If the content of the request parameter is dependent on Consumer input, prompt the user to enter a value or enter an appropriately formatted value
INVALID_REQUEST	The server is not able to adequately parse the request. Usually occurs when some request parameter is expected to be in a particular format but is not. Examples: • base64 decoding failed • The field is not in a particular format. The message field may provide additional clarification of what part/parameter of the request is considered incorrect
AUTH_ERROR	The server cannot perform the authentication necessary to process the request
NOT_FOUND	The requested resource/business entity does not exist. The resource might also be hidden for security reasons

Reason Code	Description
SERVICE_ERROR	Unexpected behaviour on the server caused the error. Either show a generic message or retry the same request again (it might succeed)
UNKNOWN_ERROR	Unknown error

***** END OF DOCUMENT *****