EMV®

Secure Remote Commerce

Specification – API

Version 1.1

January 2020

Legal Notice

The EMV® Specifications are provided "AS IS" without warranties of any kind, and EMVCo neither assumes nor accepts any liability for any errors or omissions contained in these Specifications. EMVCO DISCLAIMS ALL REPRESENTATIONS AND WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AS TO THESE SPECIFICATIONS.

EMVCo makes no representations or warranties with respect to intellectual property rights of any third parties in or in relation to the Specifications. EMVCo undertakes no responsibility to determine whether any implementation of the EMV® Specifications may violate, infringe, or otherwise exercise the patent, copyright, trademark, trade secret, know-how, or other intellectual property rights of third parties, and thus any person who implements any part of the EMV® Specifications should consult an intellectual property attorney before any such implementation.

Without limiting the foregoing, the Specifications may provide for the use of public key encryption and other technology, which may be the subject matter of patents in several countries. Any party seeking to implement these Specifications is solely responsible for determining whether its activities require a license to any such technology, including for patents on public key encryption technology. EMVCo shall not be liable under any theory for any party's infringement of any intellectual property rights in connection with the EMV® Specifications.

Revision Log – Version 1.1

Version 1.1 of the specification incorporates the changes described in SB-240: EMV® Secure Remote Commerce Specification – API v1.0. These changes correct minor errors in version 1.0 of this specification and improve consistency between this specification and the SRC JavaScript SDK specification.

For full details of the changes, refer to SB-240.

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 – API, (hereafter the "SRC API Specification"), contains server-based APIs which can be used to securely build interfaces between SRC Systems and SRC System Participants. It is intended to be used in conjunction with the EMV Secure Remote Commerce Specification (hereafter the "SRC Core Specification").

1.1 Scope

The SRC API Specification describes APIs to be used for the transmission of data between SRC Systems to SRC System Participants. These APIs are based on the following assumptions:

- The server-based APIs provide a toolkit for SRC System Participants
- They are not intended to provide context for all scenarios or use cases, and individual SRC Systems are responsible for creating implementation instructions for their SRC System Participants
- They do not preclude an SRC System from providing additional technical components to support their implementations

1.2 Constraints

The SRC API Specification is designed to work within the constraints described in the SRC Core Specification. In particular, The SRC API Specification or any implementation of the SRC API 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 the SRC API 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 3447	Public-Key Cryptography Standards (https://tools.ietf.org/html/rfc3447)
RFC 7515	JSON Web Signature (https://tools.ietf.org/html/rfc7515)
RFC 7516	JSON Web Encryption (https://tools.ietf.org/html/rfc7516)
RFC 7517	JSON Web Key (https://tools.ietf.org/html/rfc7517)
RFC 7518	JSON Web Algorithms (https://tools.ietf.org/html/rfc7518)
RFC 7519	JSON Web Token (https://tools.ietf.org/html/rfc7519)

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
Transaction Types	Recommendations for EMV Processing for Industry-Specific Transaction Types

Reference	Publication Name
3-D Secure	EMV® 3-D Secure – Protocol and Core Functions Specification
Payment Tokenisation	EMV® Payment Tokenisation Specification – Technical Framework
SB-240	SB-240: EMV® Secure Remote Commerce Specification – API v1.0
SRC UI Guidelines	EMV® Secure Remote Commerce Specification – User Interface Guidelines and Requirements
SRC Core Specification	EMV® Secure Remote Commerce Specification
SRC JavaScript SDK	EMV® Secure Remote Commerce Specification – JavaScript SDK

1.5 Definitions

For the definition of the terms used in the SRC API Specification, refer to Table 1.3: Definitions in the SRC Core Specification.

1.6 Notational Conventions

1.6.1 Abbreviations

For the definition of the abbreviations used in the SRC API 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 API Specification, refer to section 1.9.2 Terminology and Conventions in the SRC Core Specification.

2 Data Dictionary

2.1 Complex Data Objects

Table 2.1 to Table 2.32 introduce the common data objects used across the API defined in the SRC API Specification. Each table defines a single data object.

The column headed R/C/O in each table refers to whether the data element 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

2.1.1 Address

Table 2.1: Address

Data Element	R/C/O	Constraints	Description
addressId Type: String	O	UUID	Identifier used to point to the address.
name Type: String	O	Max Length = 100	Name of the ordering customer**.**
line1 Type: String	O	Max Length = 75	Address line 1**.**
line2 Type: String	O	Max Length = 75	Address line 2**.**
line3 Type: String	O	Max Length = 75	Address line 3**.**
city Type: String	O	Max Length = 50	Address city**.**
state Type: String	O	Max Length = 30	Address state**.**

Data Element	R/C/O	Constraints	Description
countryCode Type: String	O	ISO 3166 alpha 2 country code	Address country code**.**
zip Type: String	O	Max Length = 16	Address zip/postal code**.**
createTime Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time the address was created.
lastUsedTime Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time the address was last used**.**

2.1.2 AppInstance

Table 2.2: AppInstance

Data Element	R/C/O	Constraints	Description
userAgent Type: String	C	N/A	User agent string of the connecting client application. **Conditionality**: Need to be sent for browsers. For non-browsers, this would be optional
applicationName Type: String	O	Max Length = 255	Name of the connecting client application.
countryCode Type: String	O	ISO 3166 alpha 2 country code	Can be derived from, for example, IP address, to assess where the Consumer is accessing the service from. Can inform various services. For example, risk based decisioning for access to service.

Data Element	R/C/O	Constraints	Description
deviceData Type: DeviceData	O	See DeviceData	Device specific data as associated with the application instance.

2.1.3 AssuranceData

Table 2.3: AssuranceData

Data Element	R/C/O	Constraints	Description
cardVerificationEntity Type: String (Numeric)	O	Length = 2	Entity performing card verification. Valid values are: • 01 SRC Initiator • 02 SRC System • 03 SRCPI • 04 DCF • 05 DPA • 06 - 99 Others
cardVerificationMethod Type: String (Numeric)	O	Length = 2	Card verification check to validate that the PAN is active and valid at the Card Issuer. Valid values are: • 01 $0 authorisation, or single unit of currency authorisation • 02 Card Verification Number validation • 03 Postal code and address verification, where supported • 04 - 20 EMVCo future use • 21 - 99 SRC System specific

Data Element	R/C/O	Constraints	Description
cardVerificationResults Type: String (Numeric)	O	Length = 2	Verification status of the PAN. Valid values are: • 01 Verified • 02 Not Verified • 03 Not performed • 04 - 20 EMVCo future use • 21 - 99 SRC System specific
cardVerificationTimestamp Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time when the card verification was conducted.
cardAssuranceData Type: String	O		Data collected that is associated with the PAN and presented to the SRC System.
cardholderAuthenticationEntity Type: String	O	Max Length = 64	Entity performing Cardholder Authentication.

Data Element	R/C/O	Constraints	Description
cardholderAuthenticationMethod Type: String (Numeric)	O	Length = 2	Card Issuer verification of the Cardholder. Valid values are: • 01 Use of a 3-D Secure ACS • 02 Mobile banking verification of the Cardholder with an authentication code • 03 Federated login systems • 04 A shared secret between the Card Issuer and the Cardholder such as One Time Passcode (OTP), activation code • 05 - 20 EMVCo future use • 21 - 99 SRC System specific
cardholderAuthenticationResults Type: String (Numeric)	O	Length = 2	Indicates whether the Cardholder was verified or not, and what the results are when verified. • 01 Verified • 02 Not Verified • 03 Not performed • 04 - 20 EMVCo future use • 21 - 99 SRC System specific
cardholderAuthenticationTimestamp Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time when the Cardholder Authentication was conducted.

Data Element	R/C/O	Constraints	Description
cardholderAssuranceData Type: String	O		Data collected that is associated with the Cardholder and presented to the SRC System.
consumerVerificationEntity Type: String	O	Max Length = 64	Entity performing Consumer verification.
consumerVerificationMethod Type: String (Numeric)	O	Length = 2	The verification method used to verify Consumer credential. Valid values are: • 01 Static Passcode • 02 SMS One Time Passcode (OTP) • 03 Keyfob or EMV cardreader One Time Passcode (OTP) • 04 Application One Time Passcode (OTP) • 05 One Time Passcode Other (OTP) • 06 KBA • 07 Out of Band Biometrics • 08 Out of Band Login • 09 Out of Band Other • 10 Risk-Based • 11 Other • 12 - 99 EMVCo future use

Data Element	R/C/O	Constraints	Description
consumerVerificationResults Type: String (Numeric)	O	Length = 2	Indicates whether the Consumer was verified or not, and what the results are when verified. Valid values are: • 01 Verified • 02 Not Verified • 03 Not performed • 04 - 20 EMVCo future use • 21 - 99 SRC System specific
consumerVerificationTimestamp Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time when the Consumer verification was conducted
consumerAssuranceData Type: String	O		Data collected that is associated with the Consumer for assurance purposes
deviceVerificationEntity Type: String (Numeric)	O	Length = 2	Entity performing device verification. The valid values are: • 01 SRC Initiator • 02 SRC System • 03 SRCPI • 04 DCF • 05 DPA • 06 - 99 Others

Data Element	R/C/O	Constraints	Description
deviceVerificationMethod Type: String (Numeric)	O	Length = 2	Verification method used to verify Consumer Device information. Valid values are: • 01 - 20 EMVCo future use • 21 - 99 SRC System specific
deviceVerificationResults Type: String (Numeric)	O	Length = 2	Verification method used to verify Consumer Device information
deviceVerificationTimestamp Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time when the device verification was conducted.
deviceAssuranceData Type: String	O		Data collected that is associated with the device for assurance purposes.
relationshipVerificationEntity Type: String (Numeric)	O	Length = 2	Entity performing relationship verification of a combination of data. The valid values are: • 01 SRCI • 02 SRC System • 03 SRCPI • 04 DCF • 05 DPA • 06 - 99 Others
relationshipVerificationMethod Type: String (Numeric)	O	Max Length = 2	Verification method used to verify information associated with the relationship.
relationshipVerificationResults Type: String (Numeric)	O	Max Length = 2	Results of the verification of the relationship of a combination of data.

Data Element	R/C/O	Constraints	Description
relationshipVerificationTimestamp Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time when the relationship verification was conducted.
relationshipAssuranceData Type: String	O		Data collected that is associated with the binding relationship for assurance purposes

2.1.4 Card

Table 2.4: Card

Data Element	R/C/O	Constraints	Description
primaryAccountNumber Type: String (Numeric)	R	Min Length = 9 Max Length = 19	Primary Account Number. A variable length, ISO/IEC 7812-compliant account number that is generated within account ranges associated with a BIN by a Card Issuer
panExpirationMonth Type: String (Numeric)	C	Length = 2	Expiration month of the Card, expressed as a two-digit calendar month **Conditionality**: Supplied when specified for the Card (PAN)
panExpirationYear Type: String (Numeric)	C	Length = 4	Expiration year of the Card, expressed as a four-digit calendar year **Conditionality**: Supplied when specified for the Card (PAN)
cardSecurityCode Type: String (Numeric)	O	Length = 3 or 4	Card security code

Data Element	R/C/O	Constraints	Description
cardholderFullName Type: String	O	Max Length = 100	Cardholder Name
cardholderFirstName Type: String	O	Max Length = 50	Cardholder First Name
cardholderLastName Type: String	O	Max Length = 50	Cardholder Last Name
billingAddress Type: Address	O	See Address	Billing Address
paymentAccountReference Type: String	O	Max Length = 29	Payment Account Reference. A non-financial reference assigned to each unique PAN and used to link a Payment Account represented by that PAN to affiliated Payment Tokens.

2.1.5 CardholderData

Table 2.5: CardholderData

The CardholderData represents set of Card Issuer provided data related to the Cardholder.

Data Element	R/C/O	Constraints	Description
fullName Type: String	O	Max Length = 100	Full name of the Cardholder.
firstName Type: String	O	Max Length = 50	First name of the Cardholder.
lastName Type: String	O	Max Length = 50	Last name of the Cardholder.

Data Element	R/C/O	Constraints	Description
issuerIdentity Type: String	O	Max Length = 64	Identity associated with the Cardholder as known by the Card Issuer. This generally enables access to an application, website or other. Examples include username/email address/mobile number.
emailAddress Type: String	O	Max Length = 255	Email address associated with the Cardholder. This value is Cardholder generated and represents contact or notification data.
mobileNumber Type: PhoneNumber	O	See PhoneNumber	Mobile phone number associated with the Cardholder.
billingPhoneNumber Type: PhoneNumber	O	See PhoneNumber	The billing phone number associated with and provided by the Cardholder.

2.1.6 CommunicationsConsent

Table 2.6: CommunicationsConsent

Data Element	R/C/O	Constraints	Description
communicationsOptIn Type: Boolean	O	Boolean	Consumer's "Communications Opt in" preference**.**
affiliateCommunicationsOptIn Type: Boolean	O	Boolean	Consumer's "Affiliate Communications Opt in" preference**.**

Data Element	R/C/O	Constraints	Description
allowEmail Type: Boolean	O	Boolean	Consumer's "Communications" preference for the email channel.
allowText Type: Boolean	O	Boolean	Consumer's "Communications" preference for the SMS channel.
allowCall Type: Boolean	O	Boolean	Consumer's "Communications" preference for the telephony channel.
allowPush Type: Boolean	O	Boolean	Consumer's "Communications" preference for the push notification channel.

2.1.7 ComplianceSettings

Table 2.7: ComplianceSettings

Data Element	R/C/O	Constraints	Description
privacy Type: Consent	O	See Consent	Consumer's privacy consent status.
tnc Type: Consent	O	See Consent	Consumer's "T&Cs" consent status.
cookie Type: Consent	O	See Consent	Consumer's "cookie" consent status.
geoLocation Type: Consent	O	See Consent	Consumer's "geolocation" consent status.

Data Element	R/C/O	Constraints	Description
communications Type: CommunicationsConsent	O	See Communications Consent	Consumer's "communications" consent status.

2.1.8 ConfirmationData

Table 2.8: ConfirmationData

Data Element	R/C/O	Constraints	Description
checkoutEventType Type: String (Numeric)	O	Length = 2	Event type associated with the update. Valid values are: • 01 Authorise • 02 Capture • 03 Refund • 04 Cancel • 05 Fraud • 06 Chargeback • 07 Other
checkoutEventStatus Type: String (Numeric)	O	Length = 2	Event type associated with the order. Valid values are: • 01 Created • 02 Confirmed • 03 Cancelled • 04 Fraud Cancelled • 05 Others • 06 - 50 EMVCo future use • 51 - 99 SRC System specific

Data Element	R/C/O	Constraints	Description
confirmationStatus Type: String (Numeric)	O	Length = 2	Status of the event as provided by the SRC Initiator in the Confirmation message. Valid values are: • 01 Success • 02 Failure • 03 Other
confirmationReason Type: String	O	Max Length = 64	Description of the reason for the event associated with the order.
confirmationTimestamp Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time of the event completion corresponding to the Confirmation event by the SRC Initiator.
networkAuthorizationCode Type: String	O	Max Length = 25	Authorisation code associated with an approved transaction.
networkTransactionIdentifier Type: String	O	Max Length = 25	Unique authorisation related tracing value assigned by a Payment Network and provided in an authorisation response.
paymentNetworkReference Type: String	O	Max Length = 25	Transaction identifier as provided by a Payment Network after authorisation has been complete.
assuranceData Type: AssuranceData	O	See AssuranceData	Assurance data.
transactionAmount Type: TransactionAmount	O	See TransactionAmount	Amount of the transaction.

2.1.9 Consent

Table 2.9: Consent

Data Element	R/C/O	Constraints	Description
acceptedVersion Type: String	O	Max Length = 10	Version accepted by the Consumer.
latestVersion Type: String	O	Max Length = 10	Latest version.
latestVersionUri Type: String	O	Max Length = 1024	URI of the latest version.

2.1.10 Consumer

Table 2.10: Consumer

Data Element	R/C/O	Constraints	Description
consumerIdentity Type: ConsumerIdentity	R	See ConsumerIdentity	Primary verifiable Consumer Identifier within an SRC Profile. For example, an email address or a mobile phone number.
emailAddress Type: String	O	Max Length = 255	Consumer-provided email address.
mobileNumber Type: PhoneNumber	O	See PhoneNumber	Consumer-provided mobile number.
nationalIdentifier Type: String	O	Max Length = 20	Geographic-specific, nationally-provided identity for the Consumer.
countryCode Type: String	O	ISO 3166 alpha 2 country code	Consumer-provided country code.
languageCode Type: String	O	ISO 639-1 Code	Consumer-provided language choice.

Data Element	R/C/O	Constraints	Description
firstName Type: String	O	Max Length = 50	Consumer-provided first name.
lastName Type: String	O	Max Length = 50	Consumer-provided last name.
fullName Type: String	O	Max Length = 100	Consumer-provided full name.

2.1.11 ConsumerIdentity

Table 2.11: ConsumerIdentity

Data Element	R/C/O	Constraints	Description
identityProvider Type: IdentityProvider	O	See IdentityProvider	Entity or organisation that collected and verified the identity
identityType Type: ConsumerIdentityType	R	See ConsumerIdentity Type	Type of Consumer Identity transmitted or collected.
identityValue Type: String	R	Max Length = 255	Consumer Identity value that corresponds to the Consumer Identity Type. Is used to locate information within the SRC Profile. This is not `SRC Consumer Reference Identifier`, but instead a Consumer-provided value.

2.1.12 Dcf

Table 2.12: Dcf

Data Element	R/C/O	Constraints	Description
applicationType Type: ApplicationType	O	See ApplicationType	Type of the environment of the DCF.
uri Type: String	O	Max Length = 1024	DCF URI as provided by DCF.
logoUri Type: String	O	Max Length = 1024	Logo image URI provided by the DCF to support presentation.
name Type: String	O	Max Length = 60	Legal Name of DCF onboarded to the SRC System.

2.1.13 DeviceData

Table 2.13: DeviceData

Data Element	R/C/O	Constraints	Description
type Type: String	O	Max Length = 255	Type of device being used. For example. • Mobile Phone • Tablet • Laptop • Personal Assistant • Connected Auto • Home Appliance • Wearable • Stationary Computer • E-Reader • Handheld Gaming Devices • Other

Data Element	R/C/O	Constraints	Description
manufacturer Type: String	O	Max Length = 255	Manufacturer of the device
brand Type: String	O	Max Length = 255	Brand name of the device
model Type: String	O	Max Length = 255	Specific model of the device

2.1.14 DigitalCardData

Table 2.14: DigitalCardData

Data Element	R/C/O	Constraints	Description
status Type: DigitalCardStatus	R	See DigitalCardStatus	State of the Digital Card at any given time at the SRC System.
presentationName Type: String	O	Max Length = 64	Presentation text created by the Consumer to enable recognition of the PAN entered into the DCF. This value is unique to the DCF and defined by the Consumer. (e.g. Nickname).
descriptorName Type: String	R	Max Length = 64	Presentation text defined by the SRC Programme that describes the PAN presented as a Digital Card. This descriptor is the same across all DCFs.
artUri Type: String	R	Max Length = 1024	URI that houses the Card Art image to be used for presentation purposes. Can be provided by SRCPI

Data Element	R/C/O	Constraints	Description
artHeight Type: String (Numeric)	O		Height of the card art in pixels.
artWidth Type: String (Numeric)	O		Width of the card art in pixels.
pendingEvents Type: List<CardPendingEvent>	C	See CardPendingEvent	Set of events that are pending completion such as address verification or SCA. **Conditionality**: supplied if digitalCardStatus field is set to PENDING.

2.1.15 DigitalCardFeature

Table 2.15: DigitalCardFeature

Data Element	R/C/O	Constraints	Description
content Type: String	R	Max Length = 1024	Content of the digital card feature. The value is specific for the 'contentType'.
contentType Type: DigitalCardFeatureContentType	R	See DigitalCardFeatureContentType	Type of the content of the digital card feature.
style Type: String	O	Max Length = 1024	URL of a CSS style sheet that describes how to present the card feature.
width Type: String (Numeric)	O		Width to be applied to display of card feature.
height Type: String (Numeric)	O		Height to be applied to display of card feature.

2.1.16 DpaData

Table 2.16: DpaData

Data Element	R/C/O	Constraints	Description
dpaPresentationName Type: String	O	Max Length = 60	Merchant company name associated with the DPA to be used for presentation purposes within the user experience.
dpaAddress Type: Address	O	See Address	DPA's business address.
dpaName Type: String	R	Max Length = 60	Legal name of registered DPA.
dpaEmailAddress Type: String	O	Max Length = 255	DPA's contact email address.
dpaPhoneNumber Type: PhoneNumber	O	See PhoneNumber	DPA's contact phone number.
dpaLogoUri Type: String	O	Max Length = 1024	URI of the logo of the DPA.
dpaSupportEmailAddress Type: String	O	Max Length = 255	DPA's support contact email address.
dpaSupportPhoneNumber Type: PhoneNumber	O	See PhoneNumber	DPA's support contact phone number.
dpaSupportUri Type: String	O	Max Length =1024	DPA's support URI.
dpaUri Type: String	O	Max Length = 1024	DPA or Website URI.
applicationType Type: ApplicationType	O	See ApplicationType	Type of DPA.

2.1.17 DpaTransactionOptions

Table 2.17: DpaTransactionOptions

Data Element	R/C/O	Constraints	Description
transactionAmount Type: TransactionAmount	C	See TransactionAmount	The amount of the transaction. **Conditionality**: Must be supplied if 3DS is to be performed by SRC System
transactionType Type: TransactionType	O	See TransactionType	Type of transaction initiated for which the SRC System is being sent a request.
dpaBillingPreference Type: AddressPreference	O	See AddressPreference	Verbosity of billing address required by the DPA.
dpaAcceptedBillingCountries Type: List<String>	O		Billing restrictions. Array of country codes in ISO 3166-1 alpha-2 format - Payments from all the listed billing countries are accepted. For example: ["US","CA","AU"] An empty list means that all countries are accepted.
dpaShippingPreference Type: AddressPreference	O	See AddressPreference	Verbosity of shipping address required by the DPA.

Data Element	R/C/O	Constraints	Description
dpaAcceptedShippingCountries Type: List<String>	O		Shipping restrictions. Array of country codes in ISO 3166-1 alpha-2 format Shipping region country codes that limits the selection of eligible shipping addresses. For example: ["US","CA","AU"] An empty list means that all countries are accepted.
consumerEmailAddressRequested Type: Boolean	O		Whether DPA wants Consumer email address in the payload.
consumerNameRequested Type: Boolean	O		Whether DPA wants Consumer name in the payload.
consumerPhoneNumberRequested Type: Boolean	O		Whether DPA wants Consumer phone number in the payload.
merchantCategoryCode Type: String	O	Length = 4	Describes the merchant's type of business, product or service. The same value is expected in the authorisation request.
merchantCountryCode Type: String	O	ISO 3166 alpha 2 country code	Country code of the merchant.
merchantOrderId Type: String	O	UUID	Digital Payment Application generated order/invoice number corresponding to a Consumer purchase. Typically used for reconciliation purposes by the merchant.

Data Element	R/C/O	Constraints	Description
threeDsPreference Type: ThreeDsPreference	R	See ThreeDsPreference	Merchant's 3DS preferences
threeDsInputData Type: JSON Object	C		Merchant's 3DS input data. **Conditionality**: Must be supplied if 3DS is to be performed by SRC System (i.e. threeDsPreference is ON BEHALF).
srcTokenRequestData Type: JSON Object	O		Data to support a Token Request.
paymentOptions Type: List<PaymentOptions>	O	See PaymentOptions	Specifies the dynamic data requirement for the payload creation.
dpaLocale Type: String	O	ISO language country pair. [ISO 639-1 Code]_[ISO 3166 alpha 2 country code]	Merchant's preferred locale. This can be the same as the locale in the init parameters or can be different. Format: ISO language_country pair (e.g. 'en_US', 'fr_CA').
customInputData Type: JSON Object	O		Extensible container that allows DPA to pass network-specific set of data to SRC System.
orderType Type: String	O	Length = 255	Type of the order.

2.1.18 DynamicData

Table 2.18: DynamicData

Data Element	R/C/O	Constraints	Description
dynamicDataValue Type: String	C		Value of the dynamic data. **Conditionality:** Must be provided when dynamicDataType is not NONE.
dynamicDataType Type: DynamicDataType	R	See DynamicDataType	Type of the dynamic data.
dynamicDataExpiration Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time at which the dynamic data expires.

2.1.19 Error

Table 2.19: Error

Data Element	R/C/O	Constraints	Description
status Type: Numeric	R	Length = 3	http status code to categorise the errors.
reason Type: String	R	Max Length = 32	Error reason as associated with the status code.
message Type: String	R	Max Length = 255	Error message as associated with the status code.
errorDetail Type: List<ErrorDetail>	O	See ErrorDetail	Error details.

2.1.20 ErrorDetail

Table 2.20: ErrorDetail

Data Element	R/C/O	Constraints	Description
reason Type: String	O	Max Length = 32	Error reason.
source Type: String	O	Max Length = 255	Name of the source which generated this error.
message Type: String	O	Max Length = 255	Error message.
sourceType Type: String	O	Max Length = 32	Type of the source.

2.1.21 EventHistory

Table 2.21: EventHistory

Data Element	R/C/O	Constraints	Description
ageOfSrcPanEnrolmentSinceCreated Type: String (Numeric)	O	Max Length = 5	Age, in days, of the SRC Profile as it exists in the SRC System since the time it was created.
srcAgeSinceLastSuccessfulTransaction Type: String (Numeric)	O	Max Length = 5	Age, in days, of the SRC Profile as it exists in the SRC System from the time of the last successful transaction.
ageOfSrcRelationship Type: String (Numeric)	O	Max Length = 5	Age, in days, of the SRC Profile in the SRC System
ageOfConsumerRelationship Type: String (Numeric)	O	Max Length = 5	Age, in days, since the Consumer profile binding event occurred at the SRC Profile

Data Element	R/C/O	Constraints	Description
billingAndShippingRelationship Type: String	O	Length = 2	Relationship between the Cardholder billing and shipping information. Valid values are: • 01 Same as Cardholder’s billing address • 02 Consumer's preferred shipping address • 03 Consumer other address
shippingAddressUsageNew Type: String (Numeric)	O	UTC time in Unix epoch format	Date when the shipping address used for this transaction was first used with the SRC Initiator.
ageOfShippingAddressUsage Type: String (Numeric)	O	Max Length = 5	Age, in days, since shipping address used for this transaction was first used

2.1.22 IdentityValidationChannel

Table 2.22: IdentityValidationChannel

Data Element	R/C/O	Constraints	Description
validationChannelId Type: String	R	Max Length = 36	Id of the validation channel.
identityProvider Type: IdentityProvider	O	See IdentityProvider	Entity or organisation that can validate the identity
identityType Type: IdentityValidationChannelType	R	See IdentityValidationChannelType	Type of the identity validation channel (e.g. email, SMS).
maskedValidationChannel Type: String	O	Max Length = 255	Masked ID validation channel (e.g. masked email, masked mobile number).

2.1.23 MaskedAddress

Table 2.23: MaskedAddress

Data Element	R/C/O	Constraints	Description
addressId Type: String	R	UUID	Identifier used to point to the address.
name Type: String	O	Max Length = 100	Name of the individual receiving the delivered goods or service. Only applicable for the shipping address.
line1 Type: String	O	Max Length = 75	Address line 1.
line2 Type: String	O	Max Length = 75	Address line 2.
line3 Type: String	O	Max Length = 75	Address line 3.
city Type: String	O	Max Length = 50	Address city.
state Type: String	O	Max Length = 30	Address state.
countryCode Type: String	O	ISO 3166-1 alpha 2 country code	Address country code**.**
zip Type: String	O	Max Length = 16	Address zip/postal code.
createTime Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time the address was created.
lastUsedTime Type: String (Numeric)	O	UTC time in Unix epoch format	Date and time the address was last used.

2.1.24 MaskedCard

Table 2.24: MaskedCard

Data Element	R/C/O	Constraints	Description
srcDigitalCardId Type: String	C	Max Length=36	Unique Identifier of the Card. Reference representing the PAN or Payment Token that enables a non-SRCPI entity to identify the underlying PAN. A single PAN can have one or more SRC Digital Card Reference Identifiers. Digital Card information can be accompanied with SRC Digital Card Reference Identifier. It is associated with the SRC Profile to which the Digital Card belongs and is unique within an SRC System. **Conditionality**: Supplied when returned to SRCI or DCF; not required when returned to an SRCPI

Data Element	R/C/O	Constraints	Description
srcPaymentCardId Type: String	C	Max Length = 36	Reference representing the PAN that enables the SRC System to communicate with the SRCPI without transmitting the actual PAN. It is associated with the SRC Profile to which the Payment Card belongs and is unique within an SRC System. **Conditionality**: Supplied when returned to the SRCPI.
panBin Type: String (Numeric)	R	Max Length=PAN Length-10	First significant digits of the PAN included in an unmasked form.
panLastFour Type: String (Numeric)	R	Length = 4	Attribute of the Payment Card that represents the Last four digits of the PAN included in an unmasked form.
tokenBinRange Type: String (Numeric)	C	Max Length=Payment Token Length-10	Specific BIN range or subset of the BIN Range that has been designated only for the purpose of issuing Payment Tokens included in an unmasked form. **Conditionality**: must be supplied if Payment Token is used.

Data Element	R/C/O	Constraints	Description
tokenLastFour Type: String (Numeric)	C	Length = 4	Last four digits of the Payment Token included in an unmasked form. **Conditionality**: must be supplied if Payment Token is used
digitalCardData Type: DigitalCardData	R	See DigitalCardData	Digital Card Data contains digital card information that is used in the acceptance environment and in the user interface. Its purpose is to provide reference to the actual PAN or Payment Token without actually disclosing either. Digital Card Data is grouped based on the following: • PAN Authorisation Digital Card Information: data used in Request and Response Messages • UI/UX Presentation Data: data used user interfaces to provide the user with a recognisable descriptor • Digital Card Art: image that accompanies Digital Card information for user interface purposes
panExpirationMonth Type: String (Numeric)	C	Length = 2	Expiration month of the Payment Card expressed as a two-digit calendar month used for presentation purposes. **Conditionality**: Supplied when specified for the card (PAN)

Data Element	R/C/O	Constraints	Description
panExpirationYear Type: String (Numeric)	C	Length = 4	Expiration year of the Payment Card expressed as four-digit calendar year, used for presentation purposes. **Conditionality**: Supplied when specified for the card (PAN).
paymentCardDescriptor Type: String	O	Max Length = 32	Conveys the card brand, and will be a free-form string, to be defined within an SRC Programme.
paymentCardType Type: String	O	Max Length = 32	Conveys the card type.
digitalCardFeatures Type: List<DigitalCardFeature>	O	See DigitalCardFeature	Set of Digital Card attributes related to Digital Card features that should be displayed to the Consumer.
countryCode Type: String	O	ISO 3166-1 alpha 2 country code	Country code of issuance associated with the Card Issuer's BIN license
maskedBillingAddress Type: MaskedAddress	O	See MaskedAddress	Masked billing address associated with the card
dcf Type: Dcf	O	See Dcf	Digital Card Facilitator (DCF) associated with the card.
serviceId Type: String	O	Max Length = 255	Service identifier associated to a specific configuration, configured during SRC System.

Data Element	R/C/O	Constraints	Description
paymentAccountReference Type: String	O	Max Length = 29	Payment Account Reference. A non-financial reference assigned to each unique PAN and used to link a Payment Account represented by that PAN to affiliated Payment Tokens.
dateOfCardCreated Type: String (Numeric)	R	UTC time in Unix epoch format	Date when card was enrolled into the SRC System.
dateOfCardLastUsed Type: String (Numeric)	O	UTC time in Unix epoch format	Date when card was last used for an SRC transaction.

2.1.25 MaskedConsumer

Table 2.25: MaskedConsumer

Data Element	R/C/O	Constraints	Description
srcConsumerId Type: String	O	UUID	SRC Consumer Reference Identifier as generated by the SRC System.
maskedConsumerIdentity Type: MaskedConsumerIdentity	R	See MaskedConsumerIdentity	Masked value of the primary verifiable Consumer Identifier within an SRC Profile. For example, an email address or a mobile phone number.
maskedEmailAddress Type: String	O	Max Length = 255	Masked Consumer email address.
maskedMobileNumber Type: PhoneNumber	O	See PhoneNumber	Masked Consumer mobile phone number.

Data Element	R/C/O	Constraints	Description
maskedNationalIdentifier Type: String	O	Max Length = 20	Masked Consumer national identifier.
complianceSettings Type: ComplianceSettings	O	See ComplianceSettings	Consumer compliance settings**.**
countryCode Type: String	O	ISO 3166 alpha 2 country code	Consumer-provided country code**.**
languageCode Type: String	O	ISO 639-1 Code	Consumer-provided language choice**.**
status Type: ConsumerStatus	R	See ConsumerStatus	Signifies the state of the Consumer at any given time at the SRC System.
maskedFirstName Type: String	O	Max Length = 50	Masked first name of the Consumer.
maskedLastName Type: String	O	Max Length = 50	Masked last name of the Consumer.
maskedFullname Type: String	O	Max Length = 100	Masked full name of the Consumer.
dateConsumerAdded Type: String (Numeric)	R	UTC time in Unix epoch format	Date Consumer was added to the SRC System.
dateConsumerLastUsed Type: String (Numeric)	O	UTC time in Unix epoch format	Date Consumer last transacted in the SRC System.

2.1.26 MaskedConsumerIdentity

Table 2.26: MaskedConsumerIdentity

Data Element	R/C/O	Constraints	Description
identityProvider Type: IdentityProvider	O	See IdentityProvider	Entity or organisation that collected and verifies the identity.
identityType Type: ConsumerIdentityType	R	See ConsumerIdentity Type	Type of Consumer Identity transmitted or collected.
maskedIdentityValue Type: String	R	Max Length = 255	Masked Consumer Identifier value. For example, masked email address or masked mobile phone number.

2.1.27 Payload

Table 2.27: Payload

Data Element	R/C/O	Constraints	Description
card Type: Card	C	See Card	Card data associated with the PAN used for the purchase. **Conditionality**: Must be supplied if the indicated payload type is "FULL" or "PAYMENT" AND the SRC System determines that a PAN-based payload must be returned. Either a Card or a Payment Token credential is returned, never both.

Data Element	R/C/O	Constraints	Description
token Type: PaymentToken	C	See PaymentToken	Payment Token data associated with the PAN used for the purchase. **Conditionality**: Must be supplied if the indicated payload type is "FULL" or "PAYMENT" AND the SRC System determines that a Payment Token-based payload must be returned. Either a Card or a Payment Token credential is returned, never both.
shippingAddress Type: Address	C	See Address	Shipping Address as required for the delivery of the goods/services being purchased. **Conditionality**: Must be supplied if: • Identified Shipping Address is available in the SRC Profile AND • Shipping address is requested (based on dpaShippingPreference) AND • PayloadTypeIndicator is "FULL" or "NON_PAYMENT"

Data Element	R/C/O	Constraints	Description
consumerEmailAddress Type: String	C	Max Length = 255	Consumer-provided email address. **Conditionality**: Must be provided if: • Consumer Email Address is available in the SRC Profile AND • Email address is requested (consumerEmailAddressRequested is true) AND • PayloadTypeIndicator is “FULL” or “NON_PAYMENT”
consumerFirstName Type: String	C	Max Length = 50	Consumer-provided first name. **Conditionality**: Must be provided if: • Consumer First Name is available in the SRC Profile AND • Consumer Name is requested (consumerNameRequested is true) AND • PayloadTypeIndicator is “FULL” or “NON_PAYMENT”

Data Element	R/C/O	Constraints	Description
consumerLastName Type: String	C	Max Length = 50	Consumer-provided last name. **Conditionality**: Must be provided if: • Consumer Last Name is available in the SRC Profile AND • Consumer Name is requested (consumerNameRequested is true) AND • PayloadTypeIndicator is “FULL” or “NON_PAYMENT”
consumerFullName Type: String	C	Max Length = 100	Consumer-provided full name. **Conditionality**: Must be provided if: • Consumer Full Name is available in the SRC Profile AND • Consumer Name is requested (consumerNameRequested is true) AND • PayloadTypeIndicator is “FULL” or “NON_PAYMENT”

Data Element	R/C/O	Constraints	Description
consumerMobileNumber Type: PhoneNumber	C	See Phonenumber	Consumer-provided mobile number. **Conditionality**: Must be provided if: • Consumer Mobile Name is available in the SRC profile AND • Consumer Mobile number is requested (consumerPhoneNumber Requested is true) AND • PayloadTypeIndicator is “FULL” or “NON_PAYMENT”
srcTokenResultsData Type: JSON Object	C		**Conditionality**: Must be provided if Token results are provided
dynamicData Type: List<DynamicData>	R	See DynamicData	Dynamic data, as generated using the dynamic data preference. Implementation specific

Data Element	R/C/O	Constraints	Description
billingAddress Type: Address	C	See Address	Billing address associated with the card used for the purchase. **Conditionality**: Must be provided if: • Billing address is available in the SRC Profile AND • Billing address is requested (based on dpaBillingPreference) AND • PayloadTypeIndicator is “FULL” or “NON_PAYMENT”
threeDsOutputData Type: JSONObject	C		Result of 3DS authorisation of the payment. **Conditionality:** Must be provided if SRC System has performed 3DS 2.0 authorisation. Specifically, when threeDsInputData is either dynamically supplied in the DpaTransactionOptions structure or statically derived using the default DPA configuration.

2.1.28 PaymentOptions

Table 2.28: PaymentOptions

Data Element	R/C/O	Constraints	Description
dpaDynamicDataTtlMinutes Type: String (Numeric)	O		Requested “Time to Live” (expiry period) of the dynamic data, specified in minutes.
dynamicDataType Type: DynamicDataType	O	See DynamicDataType	Type of dynamic data required in the payload.

2.1.29 PaymentToken

Table 2.29: PaymentToken

The Payment Token represents set of data related to the tokenised card.

Data Element	R/C/O	Constraints	Description
paymentToken Type: String	R	ISO/IEC 7812 format	Payment Token.
tokenExpirationMonth Type: String (Numeric)	C	Length = 2	Two-digit expiry month (MM) **Conditionality**: Supplied when specified for the Payment Token.
tokenExpirationYear Type: String (Numeric)	C	Length = 4	Four-digit expiry year (YYYY) **Conditionality**: Supplied when specified for the Payment Token
cardholderFullName Type: String	O	Max Length = 100	Cardholder Name

Data Element	R/C/O	Constraints	Description
cardholderFirstName Type: String	O	Max Length = 50	Cardholder First Name
cardholderLastName Type: String	O	Max Length = 50	Cardholder Last Name
paymentAccountReference Type: String	O	Max Length = 29	Payment Account Reference. A non-financial reference assigned to each unique PAN and used to link a Payment Account represented by that PAN to affiliated Payment Tokens.

2.1.30 PhoneNumber

Table 2.30: PhoneNumber

Data Element	R/C/O	Constraints	Description
countryCode Type: String	R	Min Length = 1 Max Length = 4 (International calling code format)	Phone number country code.
phoneNumber Type: String	R	Min Length = 4 Max Length = 14	Phone number without country code.

2.1.31 SrcProfile

Table 2.31: SrcProfile

Data Element	R/C/O	Constraints	Description
maskedCards Type: List<MaskedCard>	O	See MaskedCard	Masked card dataassociated with the SRC Profile.

Data Element	R/C/O	Constraints	Description
maskedShippingAddresses Type: List<MaskedAddress>	O	See MaskedAddress	Masked shipping address data associated with the SRC Profile.
maskedConsumer Type: MaskedConsumer	C	See MaskedConsume r	Masked consumer data associated with the SRC Profile. **Conditionality**: Must be supplied for a non device bound SRC Profile only
authorization Type: String	R	See Section 3.1 Authorisation Token	First party Consumer authorisation token, provided by the SRC System.

2.1.32 TransactionAmount

Table 2.32: TransactionAmount

Data Element	R/C/O	Constraints	Description
transactionAmount Type: Number	R	Max Length = 18	Amount of the transaction represented as a floating point number.
transactionCurrencyCode Type: String (Numeric)	R	ISO 4217 three-digit currency code	Currency in which the transaction amount is expressed.

2.2 Individual Attributes

Table 2.33: Individual Attributes

Name	Constraints	Description
appInstanceId Type: String		Long-lived first party token representing an app bound to the SRC Profile. The token is intended to be used instead of cookies for non-browser implementations. The "appInstanceId" is generated by SRC System when AppInstance object is provided in Card Enrolment and Add Consumer Identifiers APIs. When client calls Unbind App Instance the association with the provided "appInstanceId" should be removed.
consumerPresent Type: Boolean		Indicates if the "Identity lookup" (Identity Lookup API) was successful or not
customOutputData Type: JSON Object		Network-specific output data.
idLookupSessionId Type: String	UUID	Session identifier returned by SRC System following an Identity Lookup service call. Can be used in call to Initiate Identity Validation service.
idValidationSessionId Type: String	UUID	Session identifier returned by SRC System following an Initiate Identity Validation service call. Used in subsequent call to Complete Identity Validation service.

Name	Constraints	Description
payloadTypeIndicatorCheckout Type: PayloadTypeIndicator	See PayloadTypeIndicator	Type of encrypted payload to be returned in the Checkout API response.
payloadTypeIndicatorPayload Type: PayloadTypeIndicator	See PayloadTypeIndicator	Type of encrypted payload to be created for the retrieval by the Payload API.
recipientIdCheckout Type: String	Max Length = 36	Recipient of the encrypted payload known to the SRC System (as provided in the Checkout API response) for the intended recipient.
recipientIdPayload Type: String	Max Length = 36	Recipient of the encrypted payload known to the SRC System (as retrieved by the Payload API) for the intended recipient.
setAsShippingAddress Type: Boolean		Used by the “Add billing Address” API to specify if the shipping address needs to be created. If this Boolean is set to true, the shipping address created and is set same as the billing address.
scrCorrelationID Type: String	Max Length = 256	Unique identifier generated by an SRC System. If an srcCorrelationId is generated and returned to a participant, it must be included in all subsequent messages sent to the SRC System within the same transaction context
threeDsOutputData Type: JSON Object		Output data following 3DS processing.

Name	Constraints	Description
validationData Type: String	Max Length = 255	Validation data (e.g. OTP) as entered by the consumer as a part of the step up authentication.
validationMessage Type: String	Max Length = 255	Validation message that needs to be presented to the consumer for step up authentication.

2.3 Enumerations

Note: the enumeration values set out below are not exhaustive. Other values may be added in future versions of this SRC API Specification, and/or other values may be defined within the scope of a specific implementation.

Table 2.34: Enumerations

Name	Valid Values
AddressPreference	• NONE • FULL • POSTAL_COUNTRY
ApplicationType	• WEB_BROWSER • MOBILE_APP • IOT_DEVICE • OTHER
CardPendingEvent	• PENDING_AVS • PENDING_SCA • PENDING_CONSUMER_IDV
ConsumerIdentityType	• EMAIL_ADDRESS • MOBILE_PHONE_NUMBER
ConsumerStatus	• ACTIVE • SUSPENDED • LOCKED

Name	Valid Values
DigitalCardFeatureContentType	• TEXT_STRING • IMAGE_URL • CONTENT_URL • LINK_URL
DigitalCardStatus	• ACTIVE • SUSPENDED • EXPIRED • PENDING • CANCELLED
DynamicDataType	• CARD_APPLICATION_CRYPTOGRAM_SHORT_FORM • CARD_APPLICATION_CRYPTOGRAM_LONG_FORM • DYNAMIC_CARD_SECURITY_CODE • CARDHOLDER_AUTHENTICATION_CRYPTOGRAM • NONE
IdentityProvider	• SRC
IdentityValidationChannelType	• EMAIL • SMS
PayloadTypeIndicator	• SUMMARY • FULL • PAYMENT • NON_PAYMENT
ThreeDsPreference	• NONE • SELF • ONBEHALF
TransactionType	• PURCHASE • BILL_PAYMENT • MONEY TRANSFER • DISBURSEMENT • P2P

2.4 Signed Checkout Objects

2.4.1 Checkout Request JWS

The Checkout Request JWS is signed object with protection of a nonce (jti) and expiry (exp) generated by the SRC System for the SRCI front-end to pass to DCF front-end. The SRC System can recognise/verify this JWS when passed by the DCF front-end in the Checkout Request API.

Table 2.35: Checkout Request JOSE Header

Parameter Name	R/C/O	Description
alg	R	Algorithm used to digitally sign the payload according to RFC 7518 section 3.1: • 'None' is not supported. • 'PS256' is preferred to 'RS256' following the recommendation in RFC 3447
kid	R	Id of the cryptographic public key of SRC System signing the checkout request. Relying party SHOULD use the ID to select appropriate key to verify the signature. Type of the public key identified by the ID MUST match type of the signing algorithm.

Table 2.36: Checkout Request Claim Set

Claim Name	Cardinality	Notes
iss	1	Value has to be URI or other ID of the SRC System that generated this JWS. The format of the ID is specific to SRC Programme. Sample value: https://srcsytem1.com
exp	1	Expiration time in UTC and unix/epoch format. This is useful for the cases where the transaction is abandoned and the JWS can be used for one-time attack, where jti cannot help.

Claim Name	Cardinality	Notes
iat	1	Issuance time in UTC and unix/epoch format Time at which the JWS was issued. This should not be before the current date/time.
jti	0..1	The "jti" (JWS ID) claim provides a unique identifier for the JWS. The value is a case-sensitive string. This helps against replay attacks.
jti_IDToken	0..1	Populated from the IDToken_JWT.jti, if the authorization is the IDToken.
srcInitiatorId	1	SRCI ID obtained during SRC onboarding
maskedCard	1	Masked digital card information Type: MaskedCard
maskedConsumer	0..1	Masked consumer information Type: MaskedConsumer
maskedShippingAddresses	0..n	Array of masked shipping addresses Type: List<MaskedAddress>
authorization	0..1	Opaque first party authorisation token.
srcCorrelationId	1	SRC Correlation Id. A new one is generated by the SRC System if there is none provided in the input.
srciTransactionId	0..1	Populated if there is one provided in the input
srcDpaId	0..1	Populated if there is one provided in the input
dpaData	0..1	Complex JSON object, DpaData
dpaTransactionOptions	1	Complex JSON object, DpaTransactionOptions
assuranceData	0..1	Populated if there is one provided in the input. This is a complex JSON object, AssuranceData
checkoutRequestUri	1	This will be the URI that SRCI will use to invoke the DCF.

Claim Name	Cardinality	Notes
		This can be same as or derived from the checkoutRequestURI in the request body.
checkoutResponseUri	1	URI that DCF will use to redirect back after the transaction is completed or cancelled or failed. Provided by SRCI during the onboarding.
serviceId	0..1	Service identifier Type: String
payloadTypeIndicatorCheckout	0..1	Type of encrypted payload to be returned in the Checkout API response Type: PayloadTypeIndicator
payloadTypeIndicatorPayload	0..1	Type of encrypted payload to be created for the retrieval by the Payload API Type: PayloadTypeIndicator
recipientIdCheckout	0..1	Recipient of the encrypted payload known to the SRC System (as provided in the Checkout API response) for the intended recipient. Type: String
recipientIdPayload	0..1	Recipient of the encrypted payload known to the SRC System (as retrieved by the Payload API) for the intended recipient. Type: String

2.4.2 Checkout Payload Response

Table 2.37 defines a data type (CheckoutPayloadResponse) returned by both the Checkout API (see Section 5.5.2 Checkout) and the Get Payload API (see Section 5.5.3 Get Payload). The CheckoutPayloadResponse returned by the API is signed by the SRC System.

Table 2.37: CheckoutPayloadResponse

Data Element	R/C/O	Constraints	Description
srcCorrelationId Type: String	C		SRC Correlation Id corresponding to this SRC checkout transaction. **Conditionality**: Required for Checkout API and Optional for Get Payload API
srciTransactionId Type: String	C		Transaction-unique identifier assigned by the SRCI**.** **Conditionality**: Required if received in the request for the Checkout API and Optional for the Get Payload API
maskedConsumer Type: MaskedConsumer	C	See MaskedConsumer	Masked consumer data associated with the SRC Profile. **Conditionality**: present if the associated SRC Profile contains Consumer data
maskedCard Type: MaskedCard	R	See MaskedCard	Masked card data

Data Element	R/C/O	Constraints	Description
shippingAddressZip Type: String	C	Max Length = 16	Zip or postal code of selected shipping address. **Conditionality**: present depending on the dpaShippingPreference option in the dpaTransactionOptions structure and either a shippingAddressId or shippingAddress object was present in the Checkout request data
shippingAddressCountryCode Type: String	C	ISO 3166 alpha 2 country code	Country code of selected shipping address. **Conditionality**: present depending on the dpaShippingPreference option in the dpaTransactionOptions structure and either a shippingAddressId or shippingAddress object was present in the Checkout request data
customOutputData Type: JSONObject	O		Optional set of network-specific data transmitted by SRC System to DPA as a result of checkout.
assuranceData Type: AssuranceData	O	See AssuranceData	Assurance data related to the checkout flow.
eventHistory Type: EventHistory	O	See EventHistory	Event history related to the checkout flow.

Data Element	R/C/O	Constraints	Description
payload Type: JWE<JWS<Payload>>	C	See Payload	Payload object. Signed by the SRC System prior to being encrypted for the specific recipient. **Conditionality**: refer to the response definitions for the Checkout API (see Section 5.5.2 Checkout) and the Get Payload API (see Section 5.5.3 Get Payload)

2.4.3 JWS JOSE Header

The JWS structure for the signed CheckoutPayloadResponse and signed Payload contains the protected JOSE header as specified in Table 2.38.

Table 2.38: JWS JOSE Header

Parameter Name	R/C/O	Description
alg	R	Algorithm used to digitally sign the payload according to RFC 7518 section 3.1: • 'None' is not supported. • 'PS256' is preferred to 'RS256' following the recommendation in RFC 3447
kid	R	Id of the cryptographic public key of SRC System signing the checkout request. Relying party SHOULD use the ID to select appropriate key to verify the signature. Type of the public key identified by the ID MUST match type of the signing algorithm.
iss	R	Issuer identifier. The value is a case sensitive URL using the https scheme that contains scheme and full qualified domain name of the host only. Sample value: https://srcsystem1.com

jti	R	A pseudo-random value used as nonce. The value is a case-sensitive string.
iat	R	Issuance timestamp in UTC and Unix/epoch format

2.5 Masking Rule

All masked objects would follow the rule below:

30% will be non-masked, where first character would always be non-masked and remaining non-masked would be the last characters to make 30% otherwise only first character shown.

For masked email address, the masking rule above only applies to the username portion of the email address which is before the '@' sign. The domain name portion will always be shown in clear.

3 Federated Identity

Federated Identity enables collaborating SRC Systems to reduce friction in the Consumer experience, by sharing the results of Consumer recognition. This section describes federated token that support the notion of federated digital identity and authorisation. An ID token is issued by the SRC System and will serve as both an attestation that the requestor is authorised to retrieve or modify that data as well as the identification of which data is to be retrieved or modified.

3.1 Authorisation Token

By default, the digital authorisation has to be a JSON Web Token (JWT) in line with RFC 7519 and compatible with OpenID Connect ID Token.

Each token needs to be digitally signed by the SRC System that issued the token. Relying parties (e.g. other SRC Systems) need to be able to validate this token using the issuing-SRC's public key. Signature has to be compliant with JSON Web Signature (JWS) specification RFC 7515.

3.1.1 Token Header

The header of JWT has to be compliant with JOSE Header as specified by RFC 7519. Table 3.1 describes the JOSE Header, in accordance with RFC.

Table 3.1: JOSE Header

Parameter Name	R/C/O	Description
alg	R	Algorithm used to digitally sign the payload according to RFC 7518 section 3.1: • ‘None’ is not supported • ‘PS256’ is preferred to ‘RS256’ following the recommendation in RFC 3447
kid	R	Id of the cryptographic public key of SRC System signing the checkout request. Relying party SHOULD use the ID to select appropriate key to verify the signature.

Parameter Name	R/C/O	Description
		Type of the public key identified by the ID MUST match type of the signing algorithm.
typ	R	Media type of the token. For JWT tokens the value should be **JWT+ext.id_token**

3.1.2 Token Claims

The Federated ID Token represents digitally signed attestation that a Consumer has been identified by an SRC System. The token contains Consumer Identifiers that allow other SRC Systems to identify the corresponding SRC Profile.

Table 3.2: Federated ID Token Claim Set

Claim Name	Cardinality	Notes
Public Claims		
iss	1	Issuer identifier for the Issuer of the response. Identifiers MUST BE in the form of case sensitive URL using the https scheme that contains scheme and full qualified domain name of the host only. *Sample value*: `https://srcsytem1.com`
sub	1	Subject Identifier. A locally unique and never reassigned identifier within the Issuer for the End-User (Consumer), which is intended to be consumed by the Client, e.g., `24400320` or `AItOawyewNvutrJUqsvl6qs7A4`. It MUST NOT exceed 255 ASCII characters in length. The sub value is a case sensitive string. SRC specific primary identifier of the Consumer that MAY BE used to locate Consumer's SRC Profile.
aud	1..n	JSON Array of the audience(s) that this ID Token is intended for.

Claim Name	Cardinality	Notes
		It MUST contain the identifier of the requestor (SRCI or DCF) as the first element of the array. It MUST also contain identifiers for participating SRC Systems as audiences. Identifiers MUST BE in the form of case sensitive URLs using the https scheme that contains scheme and full qualified domain name of the host only. Sample value: ["https://srci.com", ”https://srcsystem1.com”, “https:// srcsystem2.com”, “https:// srcsystem3.com”]
exp	1	Expiration time on or after which the ID Token SHOULD NOT be accepted for processing. The processing of this parameter requires that the current date/time MUST be before the expiration date/time listed in the value. Implementers MAY provide for some small leeway, usually no more than a few minutes, to account for clock skew. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. See RFC 3339 [RFC3339] for details regarding date/times in general and UTC in particular. Minimum expiration timestamp SHOULD BE 15 minutes from the issued-at timestamp.
iat	1	Time at which the ID Token was issued. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. This MAY BE before current date/time, i.e., an SRC may cache tokens up to close to the expiration time of the token.

Claim Name	Cardinality	Notes
jti	0..1	The "jti" (JWT ID) claim provides a unique identifier for the JWT. The value is a case-sensitive string.
auth_time	0..1	Time when the End-User authentication occurred. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. Value of the claim reflects the time when the user actually provided the credentials for authentication on this specific browser or app instance: A validated token MUST be issued IF and ONLY IF a channel was validated on this specific browser or app instance.
amr	0..n	List of methods End-User was authenticated with. JSON array of strings that are identifiers for authentication methods used in the authentication. For instance, values might indicate that both password and OTP authentication methods were used. The amr value is an array of case sensitive strings. The authentication method used when the user actually provided the credentials for authentication on this specific browser or app instance: A validated token MUST be issued IF and ONLY IF a channel was validated on this specific browser or app instance. For the specific details of each of the values, see: https://tools.ietf.org/html/draft-ietf-oauth-amr-values-04#page-3 Additional values for SRC are: sms_otp email_otp

Claim Name	Cardinality	Notes
Standard ID Token Claims		
phone_number	0..1	Obfuscated End-User's preferred mobile phone number in E.164 [E.164] format without extensions, special characters or white space. Used by Relying Party to help to identify a matching Customer profile. The Relying Party MUST NOT rely upon this value being unique.
phone_number_verified	0..1	True if the End-User’s phone number has been verified; otherwise false. When this Claim Value is true, this means that the OP took affirmative steps to ensure that this phone number was controlled by the End-User at the time the verification was performed. The means by which a phone number is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating. The Relying Party MUST NOT rely upon this value being unique. For SRC, this value MUST be true ONLY IF the OP can deterministically confirm that the phone number was verified by the user authenticated on this specific browser or app instance.
email	0..1	Obfuscated End-User's preferred e-mail address. Underlying email address value MUST conform to the RFC 5322 [RFC5322] addr-spec syntax simplified to all lowercase characters. Used by Relying Party to help to identify a matching customer profile. The Relying Party MUST NOT rely upon this value being unique.
email_verified	0..1	True if the End-User's e-mail address has been verified; otherwise false.

Claim Name	Cardinality	Notes
		When this Claim Value is true, this means that the OP took affirmative steps to ensure that this e-mail address was controlled by the End-User at the time the verification was performed. The means by which an e-mail address is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating. For SRC, this value MUST be true ONLY IF the OP can deterministically confirm that the email address was verified by the user authenticated on this specific browser or app instance.
Private Claims		
src_phone_number_mask	0..1	Masked consumer mobile phone number. This MUST use E.164 format with SRC-specific masking rules. Used by Relying Party to properly render UI and allow frictionless onboarding User Experience.
src_email_masked	0..1	Masked consumer e-mail address in RFC 5322 format with SRC specific masking rules. Used by Relying Party to properly render UI and allow frictionless onboarding User Experience.

3.1.3 Notes on Authentication

Note that:

- The `auth_time` claim is not correlated with the `iat` claim
- The `amr` claim is optional. If value is not specified, the End-User cannot be assumed as authenticated
- The `auth_time` claim SHOULD BE present ONLY IF `amr` claim is present
- The `auth_time` claim MUST always represent the time at which a consumer-interactive authentication method was performed (e.g. `email_otp` or `sms_otp`)
- The `auth_time` claim MUST NOT represent a consumer-transparent authentication method (e.g. swk or rbd)

- The `amr` claim array MUST present authentication methods in order from oldest to most recent
- When the `amr` claim contains a list of different authentication methods the `auth_time` claim shall correspond to the most recent interactive authentication method from the `amr` list

4 SRCI – DCF Interaction

As part of a checkout flow, an SRCI may be required to invoke the DCF to support necessary aspects of the checkout user experience. Upon completion of these steps, the DCF can return control back to the SRCI.

SRCI and DCF will not know each other directly, however. The URIs will be provided by the SRC System to SRCI and DCF to invoke each other for native or browser environments.

The following are the use cases to be addressed here:

- Recognized User (with an IDToken), card list is presented by SRCI and the user chooses a card. The appropriate DCF for that card is invoked by the SRCI using the URI provided by the SRC System.
- Recognized User (with an IDToken), card list is presented by SRCI and the user chooses to add a new card. The appropriate default DCF for that SRC System is invoked by the SRCI using the URI provided by the SRC System.
- Unrecognized User (no IDToken) adds a new card.

All the above use cases should be addressed for the following:

- Browser and native (iOS/Android) use cases.
- Support the following action/result scenarios from DCF to SRCI:
  - Change Consumer
  - Change Card
  - Add Card
  - Cancel Checkout
  - Successful Checkout
  - Error

4.1 Interaction Mechanisms

There can be various possible technical implementation approaches to support these interactions. The example flow is illustrative only to help the reader understand the concept, however actual implementations will differ due to security principles and policies.

The sequence of calls are as follows:

- SRCI front end (e.g. JavaScript from the SRCI that executes in the consumer's browser) calls SRC System back end to create the "checkout request data" and get the DCF URI

- SRCI front end launches DCF front end (e.g. JavaScript from the DCF that executes in the consumer's browser) using the "checkout request URI"
- After the transaction is completed, the DCF front end sends control to the SRCI using the "checkout response URI" obtained from the "checkout request data".

4.2 Launch DCF

This is the mechanism for the SRCI to launch a DCF from the given DCF URI using the signed checkout request data.

```
{checkoutRequestUri}?action={action_code}&IDToken={JWT}#{checkoutRequestJws}
```

The action code, if passed from SRCI to DCF, is expected to be one of the following:

- "NEW_CONSUMER": if specified, will advise the DCF that the Consumer entered the flow to create new Consumer profile
- "AUTH_FAILED": if specified, will advise the DCF that the Consumer failed identity validation with no attempts remaining
- "AUTH_SKIPPED": if specified, will advise the DCF that the Consumer chose to skip identity validation

The DCF application would need to read the JWS from the URI fragment using document.location (in case of the browser) or using native code (in case of a native mobile app). Note that the IDToken might not be present in scenarios like unrecognized user adding a new card.

The usage of fragment has the benefit of not having to pass the contents of JWS through the network.

There can be more suitable methods like Android Intents to launch the DCFs for certain native environments like Android. In those cases, the implementer can choose to use those platform-specific methods.

4.3 Redirect back to SRCI

After the transaction is processed, the control needs to be handed back to the SRCI from DCF.

This is done using the checkoutResponseURI derived from the above-mentioned JWS.

For non-error scenarios:

```
{checkoutRequestJws.checkoutResponseUri}?action={actionCode}&
IDToken={JWT}#{checkoutResponse}
```

The “checkoutResponse” is the signed data element of type “CheckoutPayloadReponse” as returned by the Checkout API. Note that `checkoutResponse` will be present only for the action code, “COMPLETE”.

The `IDToken` JWT is conditional in the response URI fragment and is present only when the user is stepped-up by the DCF and the user chooses to add/change card.

There might be more suitable methods like Android Intents to redirect back to SRCI for certain native environments like Android. In those cases, the implementor can choose to use those platform-specific methods.

The valid values of `actionCode` are as follows:

- “COMPLETE”: DCF processing completed normally
- “CHANGE_CARD”: consumer wishes to select an alternative card
- “ADD_CARD”: consumer wishes to add a new card
- “SWITCH_CONSUMER”: consumer wishes to change account profile / identity
- “CANCEL”: consumer wishes to cancel the flow
- “ERROR”: an error was detected and the DCF processing cannot continue

For error scenarios:

`{checkoutRequestJws.checkoutResponseURI}?action=ERROR&error={errorCode}&errorDescription={errorDescription}`

The error codes and description values are defined in Table 4.1.

Table 4.1: Error Codes

<table>
<tr><th>Name</th><th>R/C/O</th><th>Description</th></tr>
<tr><td><b>errorCode</b>
Type: string</td><td>R</td><td>Code for the error. Used by the API client for error handling.
<table>
<tr><th>error</th><th>Comments</th></tr>
<tr><td>TERMS_AND_CONDITIONS_NOT_ACCEPTED</td><td>Terms and Conditions are not accepted</td></tr>
<tr><td>ACCT_INACCESSIBLE</td><td>User account is disabled or locked out</td></tr>
<tr><td>AUTH_INVALID</td><td>Client is not authorized to make this request</td></tr>
<tr><td>AUTH_ERROR</td><td>Unrecognized client</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td></td><td><table><tr><td>SERVICE_ERROR</td><td>Unexpected server error</td></tr><tr><td>INVALID_REQUEST</td><td>This error can result when the checkoutRequestJws format or contents are invalid (due to invalid signature, etc.)</td></tr></table></td></tr>
<tr><td><b>errorDescription</b>
Type: string</td><td>O</td><td>Description of the error message. Should not be used for display purposes since this message is not localised. However, it could be used for logging and debugging purposes.</td></tr>
</table>

5 Server-Side API

5.1 API Principles

- The server-side API is designed as a set of RPC style web services where each API endpoint represents an operation to be performed.
- All request and response payloads are sent in the JSON (JavaScript Object Notation) data-interchange format derived from Protocol Buffer definitions using canonical mapping.
- Each endpoint in the API specifies the HTTP Method used to perform required operation.
- All strings in request and response objects are UTF-8 encoded.
- Version of the API that the endpoint conforms to should be specified in the URI. The version of API that must be aligned with correct version of the SRC APIs.
- All actionable fields MUST be provided as part of the request parameters (path, query or body). Only meta data must be carried in the headers. This ensures that the SDK and API spec have similar function signatures and that actionable fields can be included as part of cryptographic signatures to control against data tampering as well as repudiation claims.

5.1.1 URI Format

Set of the API endpoints must be accessible in the consistent way under the standardized URI conforming the following specification:

```
scheme://host[:port][/context]/version/path
```

Table 5.1: URI Format

Element	R/C/O	Description	Example
Scheme	R	Must be https as only encrypted HTTP protocol should be supported by the server	https
Host	R	Name of the host	src.system.com
Port	O	Number of the port. If not specified its default 443 for HTTPS	N/A

Element	R/C/O	Description	Example
Context	O	Context path on the host. While optional, this is recommended.	/api
Version	R	Version of the API aligned with the version of SRC Core Specification. The first digit is the major version of the specification, and the second digit is the minor version of the specification.	v1.0
Path	R	Path of each of the endpoints specified below.	/cards

Example URI for the Card Enrolment API as specified in Section 5.2.1 Card Enrolment:

```
https://src.system.com/api/v1.0/cards
```

5.1.2 Common HTTP Status Codes

The following common HTTP status codes are defined:

- 200: OK, the request was successful; details are included in the response body
- 202: Accepted, e.g. card details have been accepted by Enrolment service, but enrolment is outstanding, dependent upon further checks, identified by response data
- 204: No content, the service completed successfully and there is no content to be returned
- 400: Bad request, see error object for details, e.g. identifies a malformed or invalid request
- 401: Unauthorized, see error object for details, e.g. authorization token validation failure
- 403: Forbidden, see error object for details, e.g. client identity (origin) not validated
- 404: Not found, see error object for details, e.g. the SRC profile referenced in the request data was not found
- 409: Conflict, see error object for details, e.g. the submitted identifier(s) are already bound to an established SRC profile
- 500: Internal server error, see error object for details

5.1.3 Error Handling

In case an API service call response contains an HTTP error status code (4xx, 5xx), then the response body contains only an Error object, and the Error object includes details about the error.

5.1.4 Conditionality of Data

Definitions of data conditionality for the “Web Channel-Specific” (HTTP with JSON) APIs are provided based on successful outcomes for those APIs. In case of error outcomes, only an Error object will be returned.

5.1.5 Authorisation

SRC System is able to recognise and match relevant SRC Profile based on the authorisation provided by the API client in HTTP Authorisation header. SRC System should support two categories of authorisations:

- Federated ID Token: JWT token as described in Section 3 Federated Identity. SRC System should respect ID Token issued by other SRC System participating in the SRC Programme. SRC System may decide to step-up consumer by triggering identity verification when necessary
- First Party Token: opaque token issued and recognised by the same SRC System. The specification does not specify the tokens and use of them and leaves that to individual SRC Systems

SRC System should also support device and/or app binding functionality. The SRC API Specification covers two ways of recognising the device and /or app leveraging Is Recognised API:

- Implicit cookie-based recognition: User-Agent may provide HTTP cookie that will allow SRC System to identify a profile and issue Federated ID Token.
- Explicit token-based recognition: Client may provide first party token referencing an “appInstanceId” in HTTP Authorization header. SRC System can issue Federated ID Token based on the AppInstanceId.

5.1.6 API Access Control Levels

Access to all APIs must be protected using an authorization mechanism defined by the SRC System.

5.2 Card Service

Card Service supports operations related to payment card digitisation. It covers operations to enrol a card, delete a card, and to add a billing address card to a previously enrolled card.

5.2.1 Card Enrolment

The Card Enrolment operation enrols a new PAN into an SRC System and can also include consumer identifiers used to establish a new SRC Profile.

If a matching SRC Profile is identified, the card will be enrolled to that specific profile. In case SRC Profile cannot be located the SRC System will either create new SRC Profile based on supplied consumer identifiers, or the card will be enrolled in an unbounded state, which can be bound to an SRC Profile in a subsequent operation, leveraging a first party opaque authorisation token supplied in response.

Table 5.2: Card Enrolment Definition (HTTP with JSON)

HTTP Verb	POST
Path	/cards

Request Body	`{`
	`    required String srcClientId;`
	`    conditional String srcDpaId;`
	`    conditional String srcCorrelationId;`
	`    optional String serviceId;`
	`    optional String srciTransactionId;`
	`    conditional Card card;`
	`    conditional String srcDigitalCardId;`
	`    conditional JWE<Card> encryptedCard;`
	`    optional JSONObject threeDsInputData;`
	`    optional JSONObject srcTokenRequestData;`
	`    optional AssuranceData assuranceData;`
	`    conditional Consumer consumer;`
	`    conditional JWE<Consumer> encryptedConsumer;`
	`    optional AppInstance appInstance;`
	`    optional DigitalCardData digitalCardData;`
	`    conditional CardholderData cardholderData;`
	`    conditional JWE<CardholderData>`
	`encryptedCardholderData;`
	`    optional ComplianceSettings complianceSettings;`
	`}`
	
	Notes on Conditionality:

```
{
    requi
    condi
    condi
    optio
    optio
    condi
    condi
    optio
    optio
    optio
    optio
    optio
}
```

	• Exactly one of “card”, “srcDigitalCardId” or “enctyptedCard” must be provided • Either none or one of “consumer” or “encryptedConsumer” can be provided. • Either none or one of “cardholderData” or “encryptedCardholderData” can be provided. • “srcDpaId”: must be provided except when the calling client is an SRCPI • “srcCorrelationId”: if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated
Response Headers	N/A

Response Body	In case the service is processed successfully:
	`{`
	`    optional String srcCorrelationId;`
	`    required MaskedCard maskedCard;`
	
	`    conditional MaskedConsumer maskedConsumer;`
	`    conditional string authorization;`
	`    conditional string appInstanceId`
	`}`
	Notes on Conditionality:
	If the request is processed successfully and the card is enrolled into the SRC service, the service will respond with an HTTP 200 status code. In this case:
	• The "maskedCard" object has to be present in the response body
	• The "maskedConsumer" object will only be provided if Consumer details were supplied in the request body
	• The “authorization" object will only be provided if no “authorization” object was provided in the request
	If the request is processed successfully, but the card is pending further checks or consumer must be authenticated before enrolment can be completed, then the service will respond with an HTTP 202 status code, with the same response body as per HTTP 200. Specifically, there are three cases to consider:

	• Address Verification Service (AVS): In this case, the consumer should be prompted to provide billing address details to support the pending AVS check for card enrolment • Strong Cardholder Authentication (SCA): In this case, consumer should be redirected to proceed with a strong authentication mechanism i.e. 3DS NPA • Identity & Verification (ID&V): In this case, the consumer should be taken to the flow performing required identification and verification
HTTP Status Codes	• 200: OK, enrolled card details included in the response body • 202: Accepted, card details have been accepted but enrolment is outstanding, dependent upon further checks, e.g. AVS, requiring the subsequent submission of the card billing address or SCA, requiring 3DS NPA flow or required ID&V • 400: Bad request, see error object for details. Identifies a malformed or invalid request, including reporting that a card security code was expected, but not provided in the request, or to convey that the supplied consumer account profile identifier was invalid or not recognised (cookie or authorisation token), or that the supplied srcCorrelationId was invalid or not recognised • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 500: Internal server error, see error object for details

5.2.2 Delete Card

The Delete Card operation will delete the card with the specified SRC digital card identifier. The card identifier may be an SRC Digital Card Identifier or an SRC Payment Card Identifier.

Table 5.3: Delete Card Definition (HTTP with JSON)

HTTP Verb	DELETE
Path	/cards/{cardId}
Parameters	cardId: Value: may be srcDigitalCardId or srcPaymentCardId, Required

Query Parameters	**srcClientId** Type: String	Required	Identifies the connecting client, e.g. SRCI, DCF, SRCPI
	srcDpaId Type: String	Conditional	Must be provided except when the calling client is an SRCPI
	srcCorrelationId Type: String	Conditional	If available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated
	serviceId Type: String	Optional	
	srciTransactionId Type: String	Optional	
Request Body	N/A		
Response Headers	N/A		

Response Body	In case the service is processed successfully:
	{
	optional String srcCorrelationId;
	}		

HTTP Status Codes	• 200: OK, card deletion was successful • 400: Bad request, see error object for details. Identifies a malformed or invalid request, including reporting that the supplied consumer account profile identifier was invalid or not recognised (cookie or authorisation token), or is not linked to the consumer account associated to the referenced cardId, or that the supplied correlationId was invalid or not recognised • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure		

	- 403: Forbidden, see error object for details, e.g. client identity (origin) not validated - 404: Not found, see error object for details, e.g. cardId not recognised - 500: Internal server error, see error object for details

5.2.3 Add Billing Address

The Add Billing Address service adds a billing address to a previously enrolled card.

Table 5.4: Add Billing Address Definition (HTTP with JSON)

HTTP Verb	POST
Path	/cards/{cardId}/address
Parameters	cardId: Value: srcDigitalCardId or srcPaymentCardId,, Required

Request Body	`{`
	`required String srcClientId;`
	`conditional String srcDpaId;`
	`conditional String srcCorrelationId;`
	`optional String serviceId;`
	`optional String srciTransactionId;`
	`required Address billingAddress;`
	`optional Boolean setAsShippingAddress;`
	`}`
	Notes on Conditionality:
	- “srcDpaId”: must be provided except when the calling client is an SRCPI
	- “srcCorrelationId”: if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated

Response Headers	N/A
Response Body	In case the service is processed successfully:

	`{`
	`    optional String srcCorrelationId;`
	`    required MaskedCard maskedCard;`
	`    conditional MaskedAddress maskedShippingAddress;`
	`}`
	Notes on Conditionality:
	• The “maskedShippingAddress” object will be supplied if the boolean setAsShippingAddress was set to ‘true’ in the request.

HTTP Status Codes	• 200: OK, updated masked card details included in the response body • 400: Bad request, see error object for details. Identifies a malformed or invalid request, including reporting that the supplied consumer account profile identifier was invalid or not recognised (cookie or authorisation token), or that the supplied srcCorrelationId was invalid or not recognised • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 404: Not found, see error object for details, e.g. the digital card referenced in the request data was not found • 500: Internal server error, see error object for details

5.3 Address Service

The Address Service enables shipping addresses to be added to and deleted from an SRC Profile or to a transaction.

5.3.1 Add Shipping Address

The Add Shipping Address service adds a shipping address to an identified SRC Profile or to a transaction.

Table 5.5: Add Shipping Address Definition (HTTP with JSON)

HTTP Verb	POST
Path	/addresses

Request Body	`{`
	`required String srcClientId;`
	`conditional String srcDpaId;`
	`conditional String srcCorrelationId;`
	`optional String serviceId;`
	`optional String srciTransactionId;`
	`required Address shippingAddress;`
	`}`
	Notes on Conditionality:
	• “srcDpaId”: must be provided except when the calling client is an SRCPI
	• “srcCorrelationId”: if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated

Response Headers	N/A

Response Body	In case the service is processed successfully:
	`{`
	`optional String srcCorrelationId;`
	`required MaskedAddress maskedShippingAddress;`
	`}`

HTTP Status Codes	• 200: OK, updated masked card details included in the response body • 400: Bad request, see error object for details. Identifies a malformed or invalid request, including reporting that the supplied consumer account profile identifier was invalid or not recognised (cookie or authorisation token), or that the supplied srcCorrelationId was invalid or not recognised • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 500: Internal server error, see error object for details

5.3.2 Delete Shipping Address

The Delete Shipping Address service deletes a shipping address from an identified SRC Profile or from a transaction.

Table 5.6: Delete Shipping Address Definition (HTTP with JSON)

HTTP Verb	DELETE		
Path	/addresses/{addressId}		
Parameters	addressId: identifier of shipping address to be deleted, Required		
Query Parameters	**srcClientId** Type: String	Required	Identifies the connecting client, e.g. SRCI, DCF, SRCPI
	srcDpaId Type: String	Conditional	Must be provided except when the calling client is an SRCPI
	srcCorrelationId Type: String	Conditional	If available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated
	serviceId Type: String	Optional	
	srciTransactionId Type: String	Optional	
Request Body	N/A		
Response Headers	`N/A`		

Response Body	In case the service is processed successfully:
	`{`
	`    optional String srcCorrelationId;`		

	}
HTTP Status Codes	• 200: OK, the service completed successfully • 400: Bad request, see error object for details. Identifies a malformed or invalid request, including reporting that the supplied consumer account profile identifier was invalid or not recognised (cookie or authorisation token), or that the supplied srcCorrelationId was invalid or not recognised • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 404: Not found, see error object for details, e.g. addressId not recognised • 500: Internal server error, see error object for details

5.4 SRC Profile Service

The SRC Profile Service allows clients to retrieve consumer profile data from SRC System and obtain consumer authorisation for registered and trusted devices (app instances).

5.4.1 Prepare SRC Profile

The Prepare SRC Profile operation prepares the list of eligible consumer profiles and associated data for checkout from the SRC System.

Note: the presence of multiple consumer identifiers in the request or response data is optional, and can be programme or implementation specific.

Table 5.7: Prepare SRC Profile Definition (HTTP with JSON)

HTTP Verb	POST
Path	/profiles/prepare

Request Body	`{`
	`    required String srcClientId;`
	`    conditional String srcDpaId;`
	`    conditional String srcCorrelationId;`
	`    optional String serviceId;`

	`optional String srciTransactionId;`
	`conditional List<JWT> idTokens;`
	`optional DpaTransactionOptions dpaTransactionOptions;`
	`optional DpaData dpaData;`
	`}`
	Notes on Conditionality:
	• “srcDpaId”: must be provided except when the calling client is an SRCPI
	• “srcCorrelationId”: if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated
	• The “idTokens” property carries one or more ID token structured as JSON Web Tokens, generated and signed by SRC Systems. Used to identify associated consumer account profile(s), and attest that the requester is authorised to access this data. It has to be supplied if known / available to the calling client, e.g. SRCI, DCF

Response Headers	N/A

Response Body	In case the service is processed successfully:
	`{`
	`optional String srcCorrelationId;`
	`required List<SrcProfile> profiles;`
	`conditional String srcDpaId;`
	`}`
	The “profiles” list will contain SrcProfile objects if one or more SRC Profiles are found. Otherwise empty list should be returned.
	Notes on Conditionality:
	• The “srcDpaId” value must be supplied if the DPA was registered during the service call

HTTP Status Codes	• 200: OK, consumer account profile details included in the response body • 400: Bad request, see error object for details. Identifies a malformed or invalid request, including reporting that the supplied srcCorrelationId was invalid or not recognised • 401: Unauthorized, see error object for details, e.g. token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated

	• 500: Internal server error, see error object for details

5.4.2 Add Consumer Identifiers

The Add Consumer Identifiers operation binds presented identifiers to the identified SRC Profile or, in case the SRC Profile cannot be located, the SRC System will create a new SRC Profile based on consumer details or device provided in the request.

The Add Consumer Identifiers operation supports identifiers such as e-mail address, phone number and device to support various range of use-cases. In addition, the operation allows to bind a previously enrolled, unbounded card.

In case the type of a supplied consumer identifier is considered to be a primary identifier for an SRC Profile, e.g. an email address, then if the SRC System detects that an SRC Profile already exists with the same primary identifier value, the SRC System should respond to the request by advising that an SRC Profile with that primary identifier already exists.

Whether or not a supplied consumer identifier is used to replace an existing identifier on an existing SRC Profile is an SRC System implementation decision.

Table 5.8: Add Consumer Identifiers Definition (HTTP with JSON)

<table>
<tr><td><b>HTTP Verb</b></td><td>POST</td></tr>
<tr><td><b>Path</b></td><td>/profiles</td></tr>
<tr><td><b>Request Body</b></td><td>

```
{
    required String srcClientId;
    conditional String srcDpaId;
    conditional String srcCorrelationId;
    optional String serviceId;
    optional String srciTransactionId;
    conditional Consumer consumer;
    conditional AppInstance appInstance;
    optional AssuranceData assuranceData;
    optional ComplianceSettings complianceSettings;
    conditional String srcDigitalCardId;
}
```

<b>Notes on Conditionality:</b>

- “srcDpaId”: must be provided except when the calling client is an SRCPI
- “srcCorrelationId”: if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated

</td></tr>
</table>

	• One or both of “consumer” or “appInstance” must be supplied • “srcDigitalCardId” must be supplied if the request is to establish a new SRC Profile and bind the identifier(s) to a previously enrolled, unbound card
Response Headers	In case the service is processed successfully: May include a header to set a cookie, in case of remembering a browser

Response Body	In case the service is processed successfully:
	`{`
	`    optional String srcCorrelationId;`
	`    conditional String authorization;`
	`    conditional MaskedConsumer maskedConsumer;`
	`    conditional String appInstanceId;`
	`}`
	Notes on Conditionality:
	• The “authorization” object must be supplied if a Consumer object was supplied in the request and can be supplied if AppInstance was supplied in the request
	• The “maskedConsumer” object must be supplied if Consumer was supplied in the request
	• The “appInstanceId” must be supplied if AppInstance was supplied in the request (e.g. can be implemented as a cookie)

HTTP Status Codes	• 200: OK, details about the outcome of the call are included in the response body • 400: Bad request, see error object for details. Can be used to report that one or more input parameters in the request body was not valid • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 404: Not found, can be used to report that the consumer account profile referenced by the supplied authorisation token or cookie was not found • 409: Conflict, the submitted consumer identifier(s) are already bound to an established consumer account profile • 500: Internal server error, see error object for details

5.4.3 Unbind App Instance

The Unbind App Instance operation unbinds an application instance from the SRC Profile.

Table 5.9: Unbind App Instance Definition (HTTP with JSON)

HTTP Verb	DELETE		
Path	/profile/appinstances		
Parameters	**srcClientId** Type: String	Required	Identifies the connecting client, e.g. SRCI, DCF, SRCPI
	srcDpaId Type: String	Conditional	Must be provided except when the calling client is an SRCPI
	srcCorrelationId Type: String	Conditional	If available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated
	serviceId Type: String	Optional	
	srciTransactionId Type: String	Optional	
Request Body	N/A		
Response Headers	N/A		

Response Body	In case the service is processed successfully:
	`{`
	`    optional String srcCorrelationId;`
	`}`		

HTTP Status Codes	• 200: OK, the service completed successfully		

	- 401: Unauthorized, see error object for details, e.g. authorisation token validation failure - 403: Forbidden, see error object for details, e.g. client identity (origin) not validated - 404: Not found, can be used to report that the consumer account profile referenced by the supplied authorization token was not found - 500: Internal server error, see error object for details

5.5 Checkout Service

The Checkout Service allows provisioning of transaction credentials during Checkout Process and retrieval of the transaction payload to support wide range of checkout use-cases.

5.5.1 Prepare Checkout Data

This API allows the SRCI to create a checkout request to fetch the DCF Info along with the SRC checkout request JWS for DCF.

The resulting object, checkoutRequestJws is signed by the SRC System and this structure needs to be passed to the SRC System for Checkout request API.

Table 5.10: Prepare Checkout Data Definition (HTTP with JSON)

HTTP Verb	POST
Path	/transaction/preparedata

Request Body	`{`
	`    required String srcClientId;`
	`    conditional String srcDpaId;`
	`    conditional String srcCorrelationId;`
	`    optional String serviceId;`
	`    optional String srciTransactionId;`
	`    required String srcInitiatorId;`
	`    optional PayloadTypeIndicator`
	`payloadTypeIndicatorCheckout;`
	`    optional PayloadTypeIndicator`
	`payloadTypeIndicatorPayload;`
	`    optional String recipientIdCheckout;`
	`    optional String recipientIdPayload;`
	`    optional JSONObject customInputData;`
	`    required String srcDigitalCardId;`

	`conditional String consumerId;`
	`optional List<String> shippingAddressIds;`
	`optional String authorization;`
	`required DpaTransactionOptions dpaTransactionOptions;`
	`optional DpaData dpaData;`
	`optional AssuranceData assuranceData;`
	`optional String checkoutResponseUri;`
	`}`
	Notes on Conditionality:
	• “srcDpaId”: must be provided except when the calling client is an SRCPI
	• “srcCorrelationId”: if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated
	• “consumerId”: if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided

Response Headers	N/A

Response Body	In case the service is processed successfully:
	`{`
	`required String srcCorrelationId;`
	`required JWS<CheckoutRequest> checkoutRequestJws;`
	`}`
	Definition of checkoutRequestJws included in the Section 2.4.1 Checkout Request JWS.

HTTP Status Codes	• 200: OK, details about the outcome of the call are included in the response body • 400: Bad request, see error object for details. Can be used to report that one or more input parameters in the request body was not valid • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 404: Not Found, used to indicate the checkout flow does not require redirection to DCF and checkout operation can be performed instead

	- 500: Internal server error, see error object for details

5.5.2 Checkout

The Checkout operation requests initiation of Checkout in an SRC System.

The payload data is encrypted according to JSON Web Encryption (JWE) specification RFC 7516. Algorithm used to encrypt the payload is according to RFC 7518 section 4.1

Table 5.11: Checkout Definition (HTTP with JSON)

HTTP Verb	POST
Path	/transaction/credentials

Request Body	The request body contains the input parameters for the generation of the transaction credentials.
	For requests containing the signed checkoutRequestJws object, the request body shall contain the following:
	`{`
	`    required String srcClientId;`
	`    conditional String srcDpaId;`
	`    conditional String srcCorrelationId;`
	`    optional String serviceId;`
	`    optional String srciTransactionId;`
	`    optional String shippingAddressId;`
	`    optional Address shippingAddress;`
	`    optional ComplianceSettings complianceSettings;`
	`    required JWS<CheckoutRequest> checkoutRequestJws;`
	`}`
	Alternatively, for requests containing only unsigned checkout request data, then the request body shall contain the following:
	`{`
	`    required String srcClientId;`
	`    conditional String srcDpaId;`
	`    conditional String srcCorrelationId;`
	`    optional String serviceId;`
	`    optional String srciTransactionId;`
	`    optional PayloadTypeIndicator`
	`payloadTypeIndicatorCheckout;`
	`    optional PayloadTypeIndicator`
	`payloadTypeIndicatorPayload;`
	`    optional String recipientIdCheckout;`

	`optional String recipientIdPayload;`
	`required String srcDigitalCardId;`
	`optional String shippingAddressId;`
	`optional Address shippingAddress;`
	`conditional DpaTransactionOptions dpaTransactionOptions;`
	`optional AssuranceData assuranceData;`
	`optional ComplianceSettings complianceSettings;`
	`}`
	Notes on Conditionality:
	• "srcDpaId": must be provided except when the calling client is an SRCPI
	• "srcCorrelationId": if available within the present checkout session (e.g. received in an earlier API response during the present session), then it must be provided, otherwise a new checkout session will be initiated
	• "dpaTransactionOptions" object must be supplied if must be supplied if 3DS is to be performed by SRC System, or default configuration values are required to be overridden for a given transaction, or if the calculation of dynamic data is dependent on knowing the transaction amount

Response Headers	N/A

Response Body	In case the service is processed successfully:
	`{`
	`required JWS<CheckoutPayloadResponse> checkoutResponse;`
	`}`
	Additional Notes:
	• Presence of the "Payload" within the CheckoutPayloadResponse object (see Table 2.37) depends on the value of payloadTypeIndicatorCheckout parameter (as dynamically suppled in the request (query) or statically derived using the default DPA configuration) being set to any valid value other than "SUMMARY"

HTTP Status Codes	• 200: OK, transaction credential response details included in the response body • 400: Bad request, see error object for details • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure

	- 403: Forbidden, see error object for details, e.g. client identity (origin) not validated - 500: Internal server error, see error object for details

5.5.3 Get Payload

The Get Payload operation allows the SRC Initiator to retrieve the payload data from the SRC System for authorisation purposes.

The Get Payload operation is server-side API intended for server-based communication.

Table 5.12: Get Payload Definition (HTTP with JSON)

HTTP Verb	GET		
Path	/transaction/credentials		
Parameters	**srcClientId** Type: String	Required	Identifies the connecting client, e.g. SRCI, DCF, SRCPI.
	payloadTypeIndicator Type: PayloadTypeIndicator	Optional	Identifies the type of encrypted payload to be returned. Value “SUMMARY” is illegal for Get Payload operation.
	recipientId Type: String	Optional	Identifies the recipient of the encrypted payload known to the SRC System SRC System decides about the key used for encryption of the payload for the recipient.
	srcDpaId Type: String	Conditional	Must be provided except when the calling client is an SRCPI.
	srcCorrelationId Type: String	Required	Reference to the present checkout session.
	serviceId Type: String	Optional	

	srciTransactionId Type: String	Optional	
Request Body	N/A		
Response Headers	N/A		

Response Body	In case the service is processed successfully:
	`{`
	`     required JWS<CheckoutPayloadResponse>`
	`payloadResponse;`
	`}`
	In case “payloadTypeIndicator” in this request or “payloadTypeIndicatorPayload” in the Checkout API request is set to SUMMARY this request should return error code 400 with error description indicating illegal request.
	Additional Notes:
	• Presence of the “Payload” within the CheckoutPayloadResponse object (see Table 2.37) is always required		

HTTP Status Codes	• 200: OK, transaction credential response details included in the response body • 400: Bad request, see error object for details • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 500: Internal server error, see error object for details		

5.6 Confirmation Service

The Confirmation Service enables SRC Initiator and other SRC Participants to notify about checkout or payment result.

5.6.1 Confirmation

The Confirmation operation allows the SRC Initiator to notify the SRC System about the outcome of a checkout order or payment.

The Confirmation operation is server-side API intended for server-based communication.

Table 5.13: Confirmation Definition (HTTP with JSON)

HTTP Verb	POST
Path	/confirmations

Request Body	The request body must contain both of the following parameters.
	`{`
	`    required String srcClientId;`
	`    conditional String srcDpaId;`
	`    required String srcCorrelationId;`
	`    optional String serviceId;`
	`    optional String srciTransactionId;`
	`    required ConfirmationData confirmationData;`
	`}`
	Notes on Conditionality:
	• “srcDpaId”: must be provided except when the calling client is an SRCPI

Response Headers	N/A
Response Body	N/A
HTTP Status Codes	• 204: No content, the confirmation message was accepted • 400: Bad request, see error object for details • 401: Unauthorized, see error object for details, e.g. authorisation token validation failure • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 500: Internal server error, see error object for details

5.7 Identity Service

The Identity Service allows operations related to validation of identity of the Consumer and generate authorisation asset in form of the Federated ID Token.

Identity validation is two-step process containing initiation and completion to allow challenge/response interaction with the Consumer.

When requested SRC System should perform the validation of the identity regardless the consumer identifiers are known to the SRC System or not.

5.7.1 Identity Lookup

The Identity Lookup operation reports whether or not a consumer is known to an SRC System.

Table 5.14: Identity Lookup Definition (HTTP with JSON)

HTTP Verb	POST
Path	/identities/lookup

Request Body	`{;`
	`    required String srcClientId;`
	`    optional String serviceId;`
	`    required ConsumerIdentity consumerIdentity;`
	`}`

Response Body	In case the service is processed successfully:
	`{`
	`    conditional Boolean consumerPresent;`
	`    conditional ConsumerStatus consumerStatus;`
	`    conditional String idLookupSessionId;`
	`    conditional List <IdentityValidationChannel>`
	`supportedValidationChannels;`
	`}`
	Notes of Conditionality:
	• The "consumerPresent" boolean has to be present if the specified consumer identity was recognised by the SRC System
	• The “consumerStatus” provides status of the consumer if the consumer identity was recognised. This value can be used by the client to decide if identity validation can be initiated

	• If the specified consumer identity was recognised by the SRC System, then "idLookupSessionId" and list of "supportedValidationChannels" will be returned
HTTP Status Codes	• 200: OK, lookup result included in the response body • 400: Bad request, see error object for details • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 500: Internal server error, see error object for details

5.7.2 Initiate Identity Validation

The Initiate Identity Validation operation initiates a process to validate that the consumer possesses a consumer identifier associated to a consumer account profile held by the SRC System.

Table 5.15: Initiate Identity Validation Definition (HTTP with JSON)

HTTP Verb	POST
Path	/identities/validation/initiate

Request Body	`{`
	`    required String srcClientId;`
	`    optional String serviceId;`
	`    conditional ConsumerIdentity consumerIdentity;`
	`    conditional String idLookupSessionId;`
	`    optional IdentityValidationChannel`
	`requestedValidationChannel;`
	`}`
	Notes on Conditionality:
	• Either the "consumerIdentity" object or the "idLookupSessionId" value needs to be provided, but not both

Response Body	In case the service is processed successfully:
	`{`
	`    required String idValidationSessionId;`
	`    required IdentityValidationChannel`
	`maskedValidationChannel;`
	`    optional String validationMessage;`
	`    optional List<IdentityValidationChannel>`

	`  supportedValidationChannels;`
	`}`

HTTP Status Codes	• 200: OK, verification request results included in the response body • 400: Bad request, see error object for details • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 404: Not found, conveys that the supplied identity was not recognised • 500: Internal server error, see error object for details

5.7.3 Complete Identity Validation

The Complete Identity Validation operation validates that the consumer is in possession of credentials associated to a consumer account profile held by the SRC System.

Table 5.16: Complete Identity Validation Definition (HTTP with JSON)

HTTP Verb	POST
Path	/identities/validation/complete

Request Body	The request body must carry both parameters.
	`{`
	`    required String srcClientId;`
	`    optional String serviceId;`
	`    required String idValidationSessionId;`
	`    required String validationData;`
	`}`

Response Body	In case the service is processed successfully:
	`{`
	`    required JWT idToken;`
	`}`

HTTP Status Codes	• 200: OK, lookup result included in the response body • 400: Bad request, see error object for details. Can be used to report that the verification session Id was not recognised, the verification code was incorrect, or the verification process expired. If the verification code is incorrect, the error code will indicate the number of attempts remaining

	• 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 500: Internal server error, see error object for details

5.7.4 Is Recognised

The Is Recognised operation requests the SRC System to provide Federated ID Token if the connecting consumer application instance is recognised by the SRC System.

Table 5.17: Is Recognised Definition (HTTP with JSON)

HTTP Verb	GET		
Path	/identities/recognise		
Parameters	**srcClientId** Type: String	Required	Identifies the connecting client, e.g. SRCI, DCF, SRCPI
	srcDpaId Type: String	Conditional	Must be provided except when the calling client is an SRCPI
	serviceId Type: String	Optional	
	srciTransactionId Type: String	Optional	
Request Body	N/A		

Response Body	In case the service is processed successfully:
	`{`
	`    optional String srcCorrelationId;`
	`    required List<JWT> idTokens`
	`    conditional String appInstanceId;`
	`}`
	Notes on Conditionality:
	• The “appInstanceId” parameter must be supplied if the connecting consumer application instance is recognised by the SRC System		

	• A list of "idTokens" object must be supplied, one for each established SRC Profile associated to the recognised consumer application instance. Each authorisation token should be of the federated JWT identity token type
HTTP Status Codes	• 200: OK, consumer application instance was recognised, with recognition data included in the response body • 400: Bad request, see error object for details. Identifies a malformed or invalid request • 403: Forbidden, see error object for details, e.g. client identity (origin) not validated • 404: Not found, see error object for details, e.g. unable to locate consumer account profile using supplied identifier • 500: Internal server error, see error object for details

5.8 Public Keys Retrieval

SRC System must host cryptographic public keys for retrieval by other SRC Systems and participants to allow signature verification and encryption in the following cases:

- Federated ID Token is signed JWT in the form of JWS
- CheckoutRequest, CheckoutPayloadResponse and Payload are signed in the form of JWS
- Card credentials and consumer details presented during enrolment process can be encrypted in the form of JWE

Each SRC System must publish the cryptographic public keys on the web in well-known location to allow discovery of the keys by the relying party. Each key must be easily identifiable so it can be selected by relying party by the Key ID (kid) specified in the header of JWS.

For signature verification, key retrieval and selection process for SRC follows the steps below:

1. Relying party discovers URI of the signature issuer by examination of the JWS content (i.e. “iss”) or using some other method.
2. Relying party retrieves set of public keys available at well-known path on issuer host as per issuer URI
3. Relying party examines JWS header to discover Key ID (“kid” member) and cryptographic signature algorithm (“alg” member).

4. Relying party selects the corresponding public key that matched Key ID and performs verification of the signature following the algorithm

For encryption, the recipient should fetch the key based on pre-agreed Key ID.

Note: *Symmetric Key Retrieval is not defined in this version of the specification.*

Table 5.18: Public Key Retrieval Definition (HTTP with JSON)

HTTP Verb	GET
Path	/keys
Request Body	N/A
Response Body	JWK Keyset as specified by JSON Web Key standard (RFC 7517). The keyset must specify at last one valid public key. Each key in the keyset must contain the following details: • Key ID ("kid") used for key selection as described in the flow above • Key Type ("kty"). It is also recommended to specify Key Operations ("key_ops") with value "verify" to indicate the public key intended use. • The key is specified as an X.509 certificate chain ("x5c")
HTTP Status Codes	Standard HTTP error codes.

Handling of keys used to encrypt Payload object returned by SRC System is outside of scope for the SRC API Specification. The encryption algorithms and keys should be specified by SRC Programme.

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