Maksu VPOS 3.0

Merchant integration

Redirection version 5, XML/JSON API5

Revision 1.8

16/ Jul/ 2025

Disclaimer

Maksu disclaims all warranties and conditions with regard to the Products described in this document or other Maksu products, including all warranties and conditions of merchantability, whether express, implied or statutory, fitness for a particular purpose, title and non-infringement.

Maksu Development does not warrant that the Product shall satisfy customers' specific requirements, or that copies of the Product other than those provided or authorized by the Maksu shall possess functional integrity. Also, Maksu makes no warranties with respect to the fitness and operability of modifications not made by Maksu

Maksu makes no warranties as to the accuracy of the information in this document. The information is based on the knowledge and information available in the time of issue and is subject to change without any notice.

Liability

In no event shall Maksu be liable for any loss or damage to revenues, profits or other incidental, indirect and consequential damage of any kind, resulting from use of Product or Product's performance.

Contact

Any inquiries relating to this document should be directed to:

Any enquiries relating to this document should be directed to:
Maksu GMBH
Millennium Tower, 23rd and 24th Floor,Handelskai 94-96
AT-1200 Vienna
AUSTRIA
E-mail: support@Maksupay.com

Table of contents

1. Overview

Maksu VPOS is a payment application that is designed for processing merchant payments in e-commerce environment. The inputs to VPOS are requests from merchant shopping solution and from there the payment process is controlled by VPOS until the payment has completed successfully or failed and the information will be sent back to merchant shopping solution.

The payment methods available will depend on area application will be used and which are necessary for the client business model. It could have enabled credit and debit card payments that are also integrated with

Integration with 3D Secure server technology or external payment methods like net payments in shopper local banks and so on. Exact payment methods available should be specified by client and setup.

Maksu VPOS core design enables multiple types of merchant interfaces to be implemented and also the easy to implement secure default interface.

Merchants can easily attach their look and feel to payment pages by supplying their own custom CSS stylesheet.

This document describes newest version 5.0 of interfaces to date based on RSA SHA256 signature security (5.0).

Payment services provide by Maksu include but not limited to following option:

MasterCard, VISA, American Express (ask availability), GooglePay™, ApplePay™,
3D-Secure and many local payment methods by banks and wallets in Europe and elsewhere.

1.1. Security notes

VPOS merchant interface delivers great level of security. Messages are signed and verified with 3072 bit RSA Keys. Client side encryption also uses 3072 bit RSA public key to encrypt cardholder and PAN in user browser.

1.2. Changes recorded:

1.0 26.09.2024 Intitial version of V5 interface documentation
1.1 30.09.2024 – Added 3DS API desciptions
1.2 23.12.2024 – Some content fixes
1.3 28.01.2025 – orderDesc, var1..9 lengths to 4000 supporting items.
1.4 25.03.2025 – Using Google Pay™ with Maksu API, small API naming adjustments.
1.5 11.04.2025 – Document structure changes, Sandbox and production endpoints.
1.6 29.05.2025 – Added details on Google Pay button hosted and self intgration and registration.
1.7 06.06.2025 – Added details on Apple Pay button hosted and self integration.
1.8 16.07.2025 – Added status notification messages

2. Maksu VPOS interface for merchant shop

Communication between Maksu VPOS and Merchant shopping cart software can be implemented via HTTP post protocol following the specification below.

2.1. Merchant shop request to initiate payment with Maksu VPOS

Endpoints:

Production: https://pay.eu.maksupay.com/vpos/shophandler
Sandbox: https://pay.test.maksupay.com/vpos/shophandler

The following table describes the parameters of the POST from the payment page to *VPOS.*

Field (HTTP POST parameter)	Required / Optional	Description
version	R	**Required for interface version 5**
mid	R	Merchant id supplied (integer number) will be supplied to merchant, max length 30
lang	O	Language selection for payment page (ISO 639-1 language code en, fi, sv...)
trType	R	Transaction type, valid values 1 - payment, 2 - pre authorization (applicable only to card payments only), 8 – tokenization only – perform tokenization no payment. Shall be with orderAmount = 0.0
orderid	R	Merchant shop order id provided by merchant shop max length 50 chars (string 1..50)
orderDesc	O	Order description text (string 1..4000) recommended JSON **items**
orderAmount	R	Order amount (decimal number >0.0 or 0.0 for tokenizer) max length 13 with decimal point (up to 12 numbers)
currency	R	Order amount currency (string 3 ISO ISO 4217 alphabetic code (EUR, USD))
payerName	O	Order payer name, highly recommended, required for some payment flows
payerEmail	O	Order payer email address (string 1..64), Optional but Strongly recommended.
payerPhone	O	Order payer phone number, optional but strongly recommended (string..30)
billCountry	O[3DS2]	Billing address country code (string 2 ISO 3166-1-alpha-2 code (US, FI, GB))
billState	O[3DS2]	Billing address state (string..50) recommended Var str 3 3166-2 country subdivision code
billZip	O[3DS2]	Billing address zip code (string..16)
billCity	O[3DS2]	Billing address city (string..64)
billAddress	O[3DS2]	Billing address street (string..100)
shipCountry	O/R	Shipping address country code (string 2 ISO 3166-1-alpha-2 code (US, FI, GB)) Optional, required when parameter weight or dimensions are present.
shipState	O/R	Shipping address state (string..50) Optional, required when parameter weight or dimensions are present. Recommended Var str 3 3166-2 country subdivision code
shipZip	O/R	Shipping address zip code (string..16) Optional, required when parameter weight or dimensions are present. Optional, required when parameter weight or dimensions are present.
shipCity	O/R	Shipping address city (string..64) Optional, required when parameter weight or dimensions are present.
shipAddress	O/R	Shipping address street (string..100) Optional, required when parameter weight or dimensions are present.
weight	O	Order shipping weight (kg) if item is shippable and shipping needs to be calculated by VPOS (decimal number >0) max length 12 with decimal point For electronic deliveries send weight=0.0
dimensions	O	Order shipping dimensions (mm) in format width:height:depth for example a box 200:200:200 (string..25) can be used for shipping calculation if implemented so
addFraudScore	O	Incoming starting risk score (integer) max length 12
maxPayRetries	O	Maximum payment retries allowed before user is sent back to merchant,

		overrides specific profile setting if present (integer) max length 2
reject3dsU	O	Should 3-D Secure return U status, merchant has option of continuing the transaction without liability shift or reject the transaction. >If this value is true, the transaction will not be accepted. (string 1 Y/N)
payMethod	O	For pre selection of payment method. Paymethod id , can be used to preselect payment method at merchant site, so user can not select other payment method later (string..32), exact values will depend of implemented methods on service provider side.
blockScore	O	Optional block score parameter that will be used to block the transaction if transaction riskScore reaches this value or above. (Postive Integer number) max length 9
cssUrl	O	The absolute or relative (to vpos location on server) url of custom CSS stylesheet, to be used to display payment page styles. (string..128) Note: if absolute and payment page is SSL secured make sure the url is also SSL secured as browsers will not show unsecure element object warning. This may subject of approval.
confirmUrl	R	Confirmation url where to send payment confirmation in case payment was successful (string..128)
cancelUrl	R	Cancel url where to send payment feedback in case payment has failed or was canceled by user (string..128)
extInstallmentoffset	O	Optional. In case installments are supported by the processing system then this parameter of installments can be used to indicate initial offset in months when first payment will be submitted (by acquirer). Applicable for card payments only. Integer max length 3
extInstallmentperiod	O/R	Optional, required in case previous parameter is present. In case installments are supported by the processing system then this parameter of installments is to used to indicate number of payments/months the merchant requests for installments. Applicable for card payments only. Value must be >1. Max length 3 **Installment parameters and recurring parameters together are not allowed on same request**
extRecurringfrequency	O	Optional. In case recurring payments are supported by the processing system then this parameter can be used to indicate frequency of recurring payments, defines minimum number of days between any two subsequent payments .The number of days equal to 28 is special value indicating that transactions are to be initiated on monthly basis. Applicable for card payments only. Max length 3
extRecurringenddate	O/R	Optional, required in case previous parameter is present. In case recurring payments are supported by the processing system then this parameter can be used to indicate date after which recurring ends and no more transactions no more transactions are initiated. The format is YYYYMMDD. Applicable for card payments only. **Installment parameters and recurring parameters together are not allowed on same request.**
extXOrderId	O	Optional merchant and acquirer agreed extension for recognizing returning customers with submitting previous successful order id of the merchant recognized customer. If functionality is not enabled for merchant this parameter is silently ignored.
extTokenOptions	O	Optional merchant and acquirer agreed extension for generating a card token on successful payment. If functionality is not enabled for merchant this parameter is silently ignored. (number string ..3) To request new token value (on new card) first char shall be set to 1 To request auto payment (wo need user enter cvv) with existing token second char shall be 1. To request full auto payment (wo need to do 3d) with existing token third char shall be 1 (in order full auto to work both cvv omit and 3d omit shall be set 1 and allowed in merchant settings).
extToken	O	Optional merchant and acquirer agreed extension for recognizing tokens of returning customers with submitting previously generated card token. If functionality is not enabled for merchant this parameter is silently ignored.
var1	O	Optional merchant and acquirer agreed free variable type string ..4000
var2	O	Optional merchant and acquirer agreed free variable type string ..4000

var3	O	Optional merchant and acquirer agreed free variable type string ..4000
var4	O	Optional merchant and acquirer agreed free variable type string ..4000
var5	O	Optional merchant and acquirer agreed free variable type string ..4000
var6	O	Optional merchant and acquirer agreed free variable type string ..4000
var7	O	Optional merchant and acquirer agreed free variable type string ..4000
var8	O	Optional merchant and acquirer agreed free variable type string ..4000
var9	O	Optional merchant and acquirer agreed free variable type string ..4000
signature	R	**Version 5: Message signature to ensure and verify message security and integrity. RSA with SHA2 256 signature of all the field values above concatenated together (see section 2.4).**
publicKeyHash	O	Optional but highly recommended. The SHA-2 256 base64 hash of X509 encoded public key to verify signature, useful if merchant has multiple public keys in file or in transition from one key to another, so correct public key can be selected for validation

[3DS2] – required with some schemes when 3DS2, so strongly recommended to send always.

If extension parameters extInstallmentoffset, extInstallmentperiod are present and valid then the request in considered an installment payment (parent).
If extension parameters extRecurringfrequency, extRecurringenddate are present and valid then the request in considered an recurring payment (parent).

All parameters in the post must be in form default encoding (application/x-www-form-urlencoded) and form must be submitted with utf-8 encoding.
form.action="{supplied vpos service url}"
form.method="POST"
form.enctype="application/x-www-form-urlencoded"
form.accept-charset="UTF-8"

2.2. VPOS return message POST to inform merchant shop about payment success or failure.

The following table describes the parameters of the POST from *VPOS* back to merchant shop.

Field (HTTP POST parameter)	Required / Optional	Description
version	R	**Version 5**
mid	R	Merchant id supplied (integer number) max length 30
orderid	R	Merchant shop order id string max length 50
status	R	Payment status (string..16) see section 2.5 payment statuses
orderAmount	R	Order amount (decimal number >0.0) same as in request
currency	R	Order amount currency (string 3, ISO ISO 4217 alphabetic code (EUR, USD)) same as in request
paymentTotal	R	Order amount plus fees and shipping and additional service charges if applicable (decimal number >0,0) Required when payment was a success, can be omitted when payment was failed or canceled
message	O	Optional message (string..128) can provide optional information about payment or error description.
riskScore	O	Optional information about the possible risk with transaction (integer number)
payMethod	O	Optional information about payment method used to complete transaction (string 20). Is present only when payment was success
txId	O	Optional system assigned transaction reference id (integer number)
paymentRef	O	Optional end payment system reference or approval code. String 1..64
shipCountry	O	Shipping address country code (string 2 ISO 3166-1-alpha-2 code (US, FI, GB)) Optional, may be present if returned from wallet
shipState	O	Shipping address state (string..50) Optional, may be present if returned from wallet
shipZip	O	Shipping address zip code (string..16) Optional, may be present if returned from

		wallet
shipCity	O	Shipping address city (string..64) Optional, may be present if returned from wallet
shipAddress	O	Shipping address street (string..100) Optional, may be present if returned from wallet
shipRecipientName	O	Shipping recipient name (string..100) Optional, may be present if returned from wallet
shipRecipientPhone	O	Shipping recipient phone number (string..35) Optional, may be present if returned from wallet
extToken	O	If merchant requested tokenization and tokenization enabled then on successful payment token value of card used will be returned.
extTokenPanEnd	O	If merchant requested tokenization and tokenization enabled then on successful payment last 4 digits of tokenized pan are retunred.
extTokenExp	O	If merchant requested tokenization and tokenization enabled then on successful token payment token expiration is returned in format YYYYMMDD
extData	O	Optional merchant and acquirer agreed variable type string ..1024 May be encoded and contain subfields in format p1=v1&p2=v2.. (+url encoded value)
var1	O	Optional merchant and acquirer agreed free variable type string ..1024
var2	O	Optional merchant and acquirer agreed free variable type string ..1024
var3	O	Optional merchant and acquirer agreed free variable type string ..1024
var4	O	Optional merchant and acquirer agreed free variable type string ..1024
var5	O	Optional merchant and acquirer agreed free variable type string ..1024
var6	O	Optional merchant and acquirer agreed free variable type string ..1024
var7	O	Optional merchant and acquirer agreed free variable type string ..1024
var8	O	Optional merchant and acquirer agreed free variable type string ..1024
var9	O	Optional merchant and acquirer agreed free variable type string ..1024
signature	R	**Version 5: Message signature to ensure and verify message security and integrity. RSA with SHA2 256 all the field values above concatenated together having ; in end of each values . Signed by processor private key. Merchant shall obtain Certificate of processor from service provider. (see section 2.4)**
publicKeyHash	R	The SHA-2 256 base64 hash of X509 encoded public key to verify signature, useful if processor has multiple public keys set up or in transition from one key to another, so correct public key can be selected for signature validation

Note: When payment is success the message is returned to url provide by merchant **confirmUrl** parameter in request. If payment fails or is canceled the message is returned to **cancelUrl** parameter provided in merchant request.

Note2: There is also available configurable option that server sends in background delayed (5..120 seconds) independent confirmation message (copy of redirection confirmation) to merchant server without user browser interaction as sometimes user browser may fail to reach back to merchant site. So it is recommended that merchant systems are prepared to handle multiple confirmations for same order properly due this feature and also possible user browser reloads, back buttoning etc. Background confirmation http request can be identified by having user-agent header set to value “Maksu VPOS”

2.3. Recurring notification POST

The following table describes the parameters of the direct POST from VP*OS* back to merchant shop (recurring notifications url) in case of scheduled recurring child is processed by VPOS server.

Field (HTTP POST parameter)	Required / Optional	Description
version	R	**Version 5 value 5**
mid	R	Merchant id supplied by PSP (integer number)
orderid	R	Original Merchant shop order id (unique)

status	R	Payment status or the recurring child (string..16) see section 2.5 payment statuses
orderAmount	R	Original Order amount (decimal number >0.0) same as in parent recurring request
currency	R	Original Order amount currency (string 3, ISO ISO 4217 alphabetic code (EUR, USD)) same as in parent request
paymentTotal	R	Original Order amount plus fees and shipping and additional service charges if applicable (decimal number >0,0) Required when payment was a success, can be omitted when payment was failed or canceled
message	O	Optional message (string..128) can provide optional information about payment or error description.
riskScore	O	Optional information about the possible risk with transaction (integer number)
payMethod	R	Information about payment method used to complete transaction (string 20). Is present only when payment was success
txId	R	Original recurring parent system assigned transaction reference id (integer number)
sequence	R	Sequence number or recurring (parent has sequence number 1, the first recurring child will have sequence number 2, etc)
seqTxId	R	The sequence transaction unique id in system (Integer)
paymentRef	O	Optional end payment system reference or approval code. String 1..64
signature	R	**Version 5: Message signature to ensure and verify message security and integrity. RSA with SHA2 256 all the field values above concatenated together having ; in end of each values . Signed by processor private key. Merchant shall obtain Certificate of processor from service provider. (see section 2.4)**
publicKeyHash	R	The SHA-2 256 base64 hash of X509 encoded public key to verify signature, useful if processor has multiple public keys set up or in transition from one key to another, so correct public key can be selected for signature validation

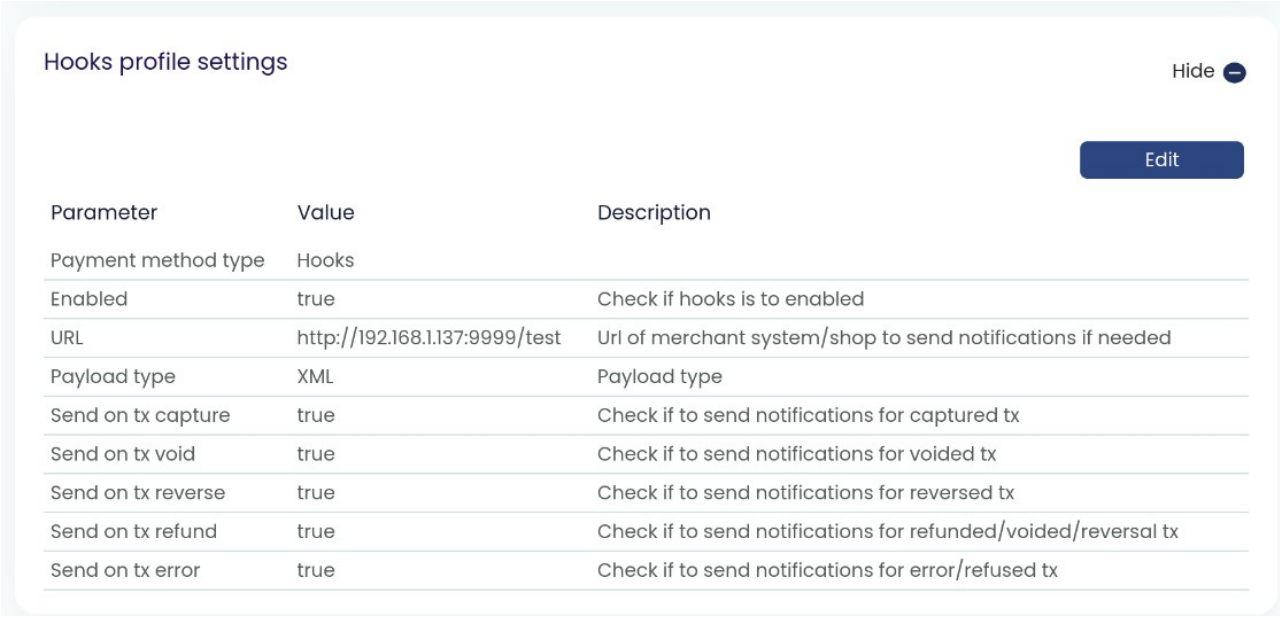
2.4. Status notification POST

The following table describes the parameters of the direct POST from VP*OS* back to merchant shop (hooks url) in case of merchant subscribed to events regarding transaction status changes.

Field (HTTP POST parameter)	Required / Optional	Description
version	R	**Version 5 value 5**
mid	R	Merchant id supplied by PSP (integer number)
orderid	R	Original Merchant shop order id (unique)
statusType	R	Information about event that happened for the transaction. (string) Ex. refund, capture, void...
statusBefore	O	Payment status before the event (string..16) see section 2.5 payment statuses, if available
status	R	Payment status or the recurring child (string..16) see section 2.5 payment statuses
settlStatusBefore	O	Settlement status before the event (string..16) if available
settlStatus	O	Settlement status (string..16) if available
orderAmount	R	Original Order amount (decimal number >0.0) same as in parent recurring request
currency	R	Original Order amount currency (string 3, ISO ISO 4217 alphabetic code (EUR, USD)) same as in parent request
paymentTotal	R	Original Order amount plus fees and shipping and additional service charges if applicable (decimal number >0,0) Required when payment was a success, can be omitted when payment was failed or canceled
payMethod	R	Payment method of transaction
txId	R	Original recurring parent system assigned transaction reference id (integer number)
paymentRef	O	Optional end payment system reference or approval code. String 1..64
signature	R	**Version 5: Message signature to ensure and verify message security and integrity. RSA with SHA2 256 all the field values above**

		concatenated together having ; in end of each values . Signed by processor private key. Merchant shall obtain Certificate of processor from service provider. (see section 2.4)
publicKeyHash	**R**	**The SHA-2 256 base64 hash of X509 encoded public key to verify signature, useful if processor has multiple public keys set up or in transition from one key to another, so correct public key can be selected for signature validation**

Status notification (events, url, payload type) can be configured in the Maksu backoffice for merchants.



Parameter	Value	Description
Payment method type	Hooks	
Enabled	true	Check if hooks is to enabled
URL	http://192.168.1.137:9999/test	Url of merchant system/shop to send notifications if needed
Payload type	XML	Payload type
Send on tx capture	true	Check if to send notifications for captured tx
Send on tx void	true	Check if to send notifications for voided tx
Send on tx reverse	true	Check if to send notifications for reversed tx
Send on tx refund	true	Check if to send notifications for refunded/voided/reversal tx
Send on tx error	true	Check if to send notifications for error/refused tx

2.5. Calculation of the Signature

The signature in the incoming POST and in the return POST is calculated by the following rule.
1. Concatenate all the values of all the possible non empty fields listed in the table, **the same order as parameters are listed in table and having ; at the end of each added field**. If a parameter is omitted or empty nothing is concatenated.

Version 5:
2. Calculate RSA with SHA2-256 signature of step 1 (using of UTF-8 char encoding when converting string to bytes) using Your private key.
3. Return the signature
4. Encode signature bytes with Base64 encoding

signature=base64(RSA with SHA2-256(utf8bytes(value1;value2;...;value n;)))

Note: ';' is separator between values concatenated and at the end of last value.
If optional value is missing or empty do not add any value and no separator as well.
Never implement the signature calculation in browser using javascipt or similar as this way you expose your private key to the world. Only implement it ins server side executed code as (jsp/servlet/asp/php etc).

2.6. Payment statuses in response message

AUTHORIZED, CAPTURED - payment was successful (accept order)
CANCELED - payment failed, user canceled the process (deny order)
REFUSED - payment failed, payment was denied for card or by bank (deny order)

REFUSEDRISK - payment failed, payment was denied for card by risk score (deny order)
ERROR - non recoverable system or other error occurred during payment process (deny order)
COMPLETED - tokenization only completed successfully.

2.7. Examples how to generate merchant keys

2.7.1 With openssl

Its just possible to do all in one line:
openssl req -x509 -newkey rsa:3072 -sha256 -keyout merchantkey.pem -out merchantcert.pem -days 365 -subj "/C=EE/ST=My State/L=my City/O=Company Name/OU=7711223/CN=www.mysite.com"

The output file **merchantcert.pem** need to be sent to service provider to install with Your merchant account so Your messages will be validated with public key in Your certificate.
C – is two letter country code
L – locality eg. city where you are located.
OU - is recommended to fill with Your merchant number with service provider.
O - shall be your company full or public name.
CN – is recommended (not required as with server certificates) to be your website name

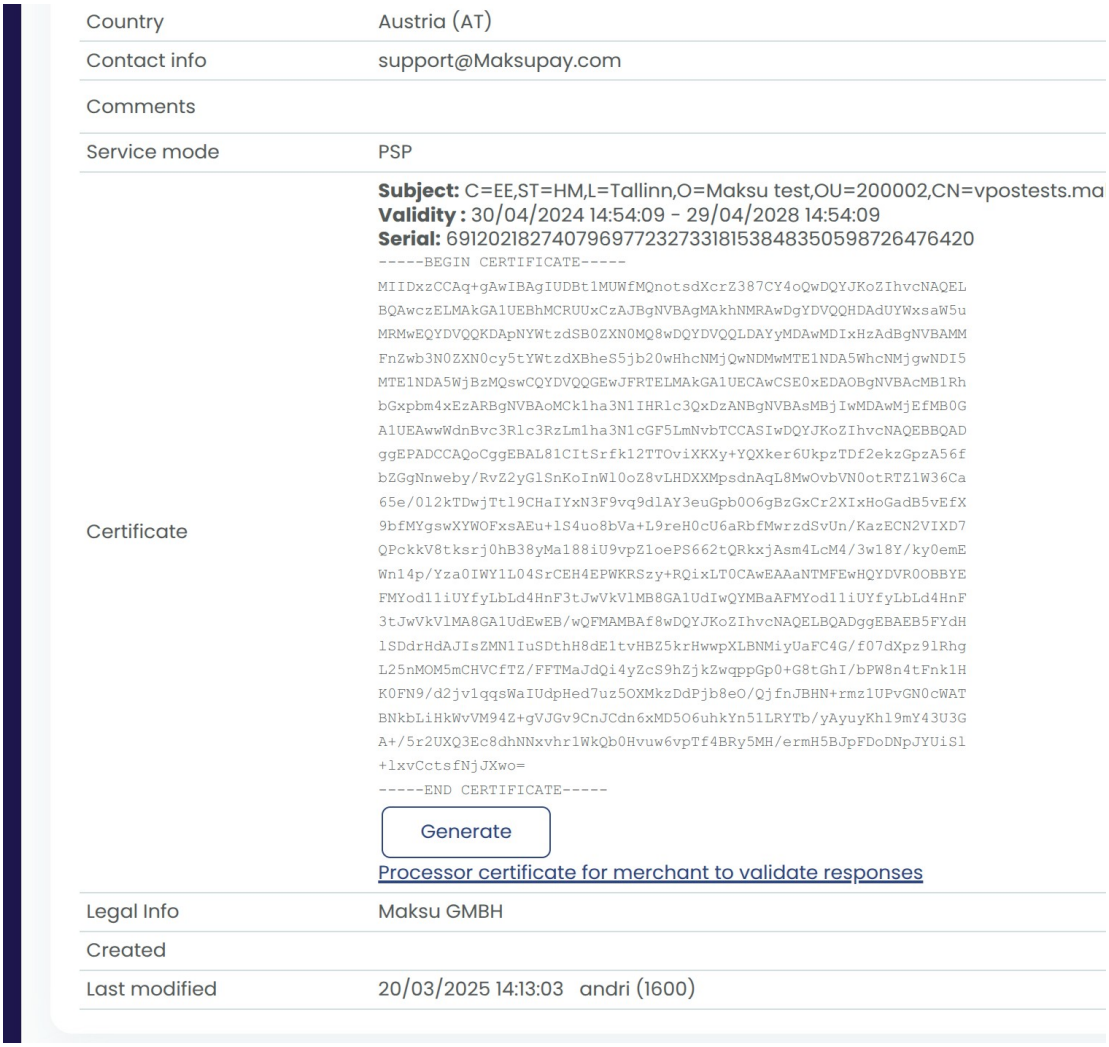
rsa:keysize is recommended to be 2048 or 3072 bits for foreseeable future and validity days up to 1460 days (4 years), ask service provider if it has specific policy or requirements.
Use necessary measures to protect your private key in generated file merchantkey.pem.
Converting private key to PKCS8 format handleable by java:
openssl pkcs8 -topk8 -in merchantkey.pem -inform PEM -outform PEM -out merchantkey-p8.pem -nocrypt

2.7.2 With java keytool

With java keytool private key remains in keystore and cannot be extracted unless special software is used. So Your software shall operate directly with this keystore then.
keytool -genkey -keyalg **RSA** -sigalg **SHA256withRSA** -dname "CN=www.mysite.com,OU=7711223,O=Company Name,L=My City,S=My State,C=EE" -keysize 3072 -validity 1460 -alias mykey2025 -storetype PKCS12 -keystore mykeystore.p12 -keypass strongPassKey -keystore mycerts.p12 -storepass strongPass
Now export Your certificate to a file that can be sent to service provider:
keytool -exportcert -alias mykey2025 -file merchantcert.pem.cer -storetype PKCS12 -keystore mycerts.p12 -storepass strongPass -rfc

2.7.3 Maksu backoffice

Maksu backoffice for merchants also has tool to generate merchant signing keys with few clicks.



In merchant main page click “Generate” button to genrate new signing private key and public key to be stored in Your Maksu account to verify Your signed messages.

2.8. Special data cases

2.8.1 Order items

Some payment methods may require order line item details and tax details. To send such info use orderDesc or var1..va9 fields in format,
then use prefix
“items:”
and values is JSON array of item objects, for example as follows
items:[{“t” :”p”, “n” :”Dennis ball”, “c” : “1234001”, “q” : 2 , “qu” : “pcs”,“up” : 100, “tp” : 200, “tt” : 36, “tr” : 2200},
{“t” :”st”, “n” :”VAT 22%”, “c” : “vat22”, “q” : 1 , “qu” : “%”,“up” : 2200, “tp” : 36, “tt” : 36, “tr” : 2200]

Fields:
t – means type :
p – physical, ds-discount, sf -shipping fee, st – sales tax, d – digital , g – giftcard, sc – store credit, s – surcharge, su – subscription
n - means product name
c – means product code
q – means quantity
qu – means quantity unit eg pcs or m etc
up – unit price eg 100 (in cents, includes tax)
tp – total price eg 200 (in cents includes tax)
td – optional total discount amount (in cents)
tt – total tax (eg vat included)
tr – tax rate eg tax prsentage 2200 means 22%
if type=sc
spi – subscription interval ("DAY" "WEEK" "MONTH" "YEAR")
sbic – subscription interval count

Recurring control:
rcauto:false
if merchant wants self submit executions of recurring payments instead of auto scheduling can submit this flag in var1..var9

Mail telephone order:
moto:true
If merchant wants to initiate mail telephone order instead of ecommerce then can submit this flag in var1..var9

3. XML/JSON API Interface

The XML/JSON API interface plugin makes possible that merchants with their own payment pages hosted in their system to use e-commerce services provided by VPOS using XML or JSON messaging.

XML/JSON Messaging is using request real time and response messages in the same request/response cycle. In request message merchant provides payment and order info and in response messages VPOS indicates the result of the action performed. By default the merchant should receive the response message within 30 seconds maximum.
Root element of request and response messages is *VPOS*

Current version of XML/JSON API is 5.0
The request message general structure:

```
<VPOS xmlns="http://www.modirum.com/schemas/vposxmlapi5"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
	<Message id="M1727355916647" senderId="200002" ts="2024-09-26T13:05:16.647Z"
version="5.0">
		<SaleReq merchantId="200002">
			<OrderInfo orderId="01727355899011">
				<OrderDesc />
				<OrderAmount>1.25</OrderAmount>
				<Currency>EUR</Currency>
			</OrderInfo>
			<PaymentInfo>
				<PayMethod>visa</PayMethod>
				<Card>
					<Pan>4016360000000010</Pan>
					<ExpDate>2712</ExpDate>
					<CVV2>756</CVV2>
					<HolderName>John Smith</HolderName>
				</Card>
				<PayerEmail />
				<PayerIpAddress>172.29.0.1</PayerIpAddress>
			</PaymentInfo>
		</SaleReq>
	</Message>
	<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
		<ds:SignedInfo>
			<ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-
xml-c14n-20010315" />
			<ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-
more#rsa-sha256" />
			<ds:Reference URI="#M1727355916647">
				<ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256" />

	<ds:DigestValue>NuDrebbMO7lzCWN5pypq5pLde9huVxRYUrOzu0UGNJE=</ds:DigestValue>
			</ds:Reference>
		</ds:SignedInfo>
		<ds:SignatureValue>
rCgl9cFMYDsbro2acdbTCme0n4OK2cDTQxbug7ZHI5R1Nveyu1mb+h8aOZAm6Jhn2rc3OJeD3Qky
QMQ7QRKzCD4HQIPwpaBoI4Kk6PFwwk3aD/s9+WWfeJQVpZnRqcDZOlKdeoMvrHj6PYSBqGUIKtT2
/tKw6ygwcvnX5SBiVO5gZo4DReY8f+GOOBlLfqNFHtyVG+HTUATagpKzEiQBLlDuVsWksGkN5jk+
oFwoYlev8qB2f4niQEKFcT/KN16WszVlkVnRVcJvl8u7IGKhTv7wfzqnb/AXbf/Qq7n1uAioLbLF
RE25mHVt6z1138xkX6rlBbeQ4D/HU8SZSiXaDg==
		</ds:SignatureValue>
		<ds:KeyInfo>
			<ds:X509Data>
				<ds:X509Certificate>
MIIDxzCCAq+gAwIBAgIUDBt1MUWfMQnotsdXcrZ387CY4oQwDQYJKoZIhvcNAQELBQAwczELMAkG
A1UEBhMCRUUxCzAJBgNVBAgMAkhNMRAwDgYDVQQHDAdUYWxsaW5uMRMwEQYDVQQKDApNYWtzdSB0
ZXN0MQ8wDQYDVQQLDAYyMDAwMDIxHzAdBgNVBAMMFnZwb3N0ZXN0cy5tYWtzdXBheS5jb20wHhcN
```

```
MjQwNDMwMTE1NDA5WhcNMjgwNDI5MTE1NDA5WjBzMQswCQYDVQQGEwJFRTELMAkGA1UECAwCSE0x
EDAOBgNVBAcMB1RhbGxpbm4xEzARBgNVBAoMCk1ha3N1IHRlc3QxDzANBgNVBAsMBjIwMDAwMjEf
MB0GA1UEAwwWdnBvc3Rlc3RzLm1ha3N1cGF5LmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAL81CItSrfk12TTOviXKXy+YQXker6UkpzTDf2ekzGpzA56fbZGgNnweby/RvZ2yGlSn
KoInWl0oZ8vLHDXXMpsdnAqL8MwOvbVN0otRTZ1W36Ca65e/0l2kTDwjTtl9CHaIYxN3F9vq9dlA
Y3euGpb0O6gBzGxCr2XIxHoGadB5vEfX9bfMYgswXYWOFxsAEu+lS4uo8bVa+L9reH0cU6aRbfMw
rzdSvUn/KazECN2VIXD7QPckkV8tksrj0hB38yMa188iU9vpZ1oePS662tQRkxjAsm4LcM4/3w18
Y/ky0emEWn14p/Yza0IWY1L04SrCEH4EPWKRSzy+RQixLT0CAwEAAaNTMFEwHQYDVR0OBBYEFMYo
d11iUYfyLbLd4HnF3tJwVkVlMB8GA1UdIwQYMBaAFMYod11iUYfyLbLd4HnF3tJwVkVlMA8GA1Ud
EwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAEB5FYdHlSDdrHdAJIsZMN1IuSDthH8dE1tv
HBZ5krHwwpXLBNMiyUaFC4G/f07dXpz9lRhgL25nMOM5mCHVCfTZ/FFTMaJdQi4yZcS9hZjkZwqp
pGp0+G8tGhI/bPW8n4tFnk1HK0FN9/d2jv1qqsWaIUdpHed7uz5OXMkzDdPjb8eO/QjfnJBHN+rm
z1UPvGN0cWATBNkbLiHkWvVM94Z+gVJGv9CnJCdn6xMD5O6uhkYn51LRYTb/yAyuyKhl9mY43U3G
A+/5r2UXQ3Ec8dhNNxvhr1WkQb0Hvuw6vpTf4BRy5MH/ermH5BJpFDoDNpJYUiSl+lxvCctsfNjJ
Xwo=
                        </ds:X509Certificate>
                  </ds:X509Data>
            </ds:KeyInfo>
      </ds:Signature>
</VPOS>
```

The response message general structure:

```
<<VPOS xmlns="http://www.modirum.com/schemas/vposxmlapi5"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
      <Message id="M1727355916647" senderId="C=EE,ST=HM,L=Tallinn,O=ChaosPay,OU=E-COM
Signer,CN=ChaosPay E-COM Signer 2024-01"
            ts="2024-09-26T13:05:17.101Z" version="5.0">
            <SaleRes errorCode="SE" orderId="01727355899011" status="ERROR" txId="0">
            <Description>Unexpected system error id 1727355917065</Description>
            </SaleRes>
      </Message>
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:SignedInfo>
                  <ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-
xml-c14n-20010315" />
                  <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-
more#rsa-sha256" />
                  <ds:Reference URI="#M1727355916647">
                        <ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256" />

      <ds:DigestValue>Dbg+Kja0oQcSwTlThv9UeZzpTrzA9FmGsjik+EHh0Fs=</ds:DigestValue>
                  </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>
rFhOMahqaMNDKcoAb5xf4QmXMDkUKvk2oeUmWI3A2NKnEIjafs3kA51/38i92GhT1+KRtR8qJKmd
qrytSqdNCr3CwYdhlGB2RyZrUE/zhSHUxWXFqrpFVH+Ikb4EYJ2ytakDrG7XzMdyJu9YKMsnakHB
W6xXyRJkC/P3b90R6myaQJRFJemGXGu75wJ/R+qmF6S5V/eRS2l18YuNX/3qMtINWQNALPfftxLQ
0ZKKNwT44XIU1GspK8jevPXmV52JkvSE+3uGu3a5t7RD8/7jgOgHa4yNeT7Zbe5OrCLlZsoW6Ypg
yyVmipG8UEH+h9lhSWIDn/+oeRO6BckuDKZL8FWxuFrA/GUdfWqbnznNrrA30jQEFhRrxVgGhN6g
PFnEgXPwUSvQLuYcNXryTR3wT3Ndi+Wsc3hvlfl4Ah6HbZMmu5S9m/+5AIqb6lDxsB4AGxDBi7UG
3jVswAq1uK0zGZPiPPtQGJ/J8wQ1tmJCx1Bb9ssESEQneYfRC1txM01f
            </ds:SignatureValue>
            <ds:KeyInfo>
                  <ds:X509Data>
                        <ds:X509Certificate>
MIIEczCCAtsCBGWfmh4wDQYJKoZIhvcNAQELBQAwfjEmMCQGA1UEAxMdQ2hhb3NQYXkgRS1DT00g
U2lnbmVyIDIwMjQtMDExFTATBgNVBAsTDEUtQ09NIFNpZ25lcjERMA8GA1UEChMIQ2hhb3NQYXkx
EDAOBgNVBAcTB1RhbGxpbm4xCzAJBgNVBAgTAkhNMQswCQYDVQQGEwJFRTAeFw0yNDAxMTEwNzM0
NTRaFw0yODAxMTAwNzM0NTRaMH4xJjAkBgNVBAMTHUNoYW9zUGF5IEUtQ09NIFNpZ25lciAyMDI0
LTAxMRUwEwYDVQQLEwxFLUNPTSBTaWduZXIxETAPBgNVBAoTCENoYW9zUGF5MRAwDgYDVQQHEwdU
YWxsaW5uMQswCQYDVQQIEwJITTELMAkGA1UEBhMCRUUwggGiMA0GCSqGSIb3DQEBAQUAA4IBjwAw
ggGKAoIBgQDI/pOZUHHlatYS/MV0i+xEMQE96tVFdRdbBesVa1P4Y1eiXITdvz5+8c2LEiBbYY6J
If4AHst0PuFlJSCYPBalEWNJTQCFOw09fHY9D3qhPoIixOM7WD46PGxbwU+Ld9y5ZfNthg+HBQ+F
```

```
OUCVcmXZ/id51FoyldValPlyjzea/5ydNhukfxIKG4zsDOa5Nn/EP2sS17MSMtTeDQbNT75asLTP
CURueVXDgAZDwTD4J3uGER4hxMAw/ybd32X0PA+MCMmNAUuLkWI1Y6SpRGELr8yFTnIYrvsFuoPM
qToLklq+E7YoJlWrDgdDXc3rGW5fu+Rk1NPRw8G6+tulkA416u2lOX5LaQkcA8ZQ3ZFHVegw7XUf
NmMycw2ofKTS5HA3nMwG4Ajr441PefTel0tKnX3qIJkBeBBTsUN+tPH7eZwz++Diru/oi435UFLg
zKdxfMc/KvjoEFw5nI1P2PZmKXnLfZbhqWxanKADzL1MRhrjCWJNo//B6puJasGDgoUCAwEAATAN
BgkqhkiG9w0BAQsFAAOCAYEAPTN5kuS/I6nsE5jBIr7zaxV7UM+ooSEdzFme6A83VvBIxJztyLMr
7kYJX6NQFbCSjIWdFs9uppKqW19VdutAhPrtLZHohdrWKHtEQOVppuarU8pNp2+MR5CAW9FmbJww
QAvh5BwJfmgIOUC+WZ5CCR5KEdPcizJwQV3j8iddysEyrdAUYetE/unZv291JtvpmTJpGu9nnb/r
cCLqbfCT/3mdxQ+idYvGCSKbEkPYfnlWm3rK9Z+BnRqsMHxmteDYlB6EHdNLzOzf/N3IKkwxrtld
Cm5VLx2XbvpP40vBs48yny++I+4aa2dhMAJNlg4YMtB1fRbJI7A9IBxWRuNQwX1wAX6mnyOF3loO
+KlCV5jezX7TjA97GTrSR0O4Zr0w3S1D4/dDctnlnOfQlE3/sFyTmzjwNoMzgBxhlhcRU6Tohb27
HJDV0yJRb27dX9SowapkUIKYhJ+zAxkWaIU3uHPAOiWp73e7p5yPCvRoYjZZDlKyGi1l2iOh+iGk
B3rI
                        </ds:X509Certificate>
                    </ds:X509Data>
                </ds:KeyInfo>
            </ds:Signature>
</VPOS>
```

The general error message structure (returned in case request: message was unparseable or unvalidatable)

```
<VPOS>
   <Message version="5.0" id="12345">
      <ErrorMessage>
            <ErrorCode></ErrorCode>
            <Description></Description>
            <OriginalXML></OriginalXML>
      </ErrorMessage>
   </Message>
</VPOS>
```

The exact xml bindings are defined in xsd schema.
https://pay.test.maksupay.com/vpos/xsd/VPOS5.xsd

Description of request and response message elements and fields and their usage:
Note JSON messages are identical to XML except element attributes are then JSON objects and properties respectively. Also JSON messages missing Signature object as this is sent in http headers and calculated over message body being posted.

3.1 Payments API

Endpoints for both XML or JSON:

Production: https://pay.eu.maksupay.com/vpos/vposapi
Sandbox: https://pay.test.maksupay.com/vpos/vposapi

Field/request	Type	Description
Request		
VPOS	element	XML root element / JSON root { }
Message	element type Message	Message contents element / JSON Object "message"
version	attribute, xsi:string	Message version default value "5.0"
id	attribute, xsi:ID	Message unique identifier (values in request and reply messages this must match, also used for lookup signature reference object when validating signature) (“M1234567”)
lang	attribute, xsi:string(2)	Message attribute to specify context language (Optional) (ISO 639-1 language code en, fi, sv, el, etc..)

ts	Attribute xsi:dateTime	Approximate time when message was created
senderId	attribute, xsi:string	In Requestes merchant service account id (merchant number)
Signature	element ds:SignatureType	Required if version = 5.0 (only supported version) The xml signature as defined https://www.w3.org/TR/xmldsig-core/ Canonicalization http://www.w3.org/TR/2001/REC-xml-c14n-20010315 SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256" Requests are signed by merchant private ket and validated with merchant Certificate and responses are signed by processor private key and validated with Processor certificate
Payment API messages SaleReq, AuthorisationReq, CaptureReq, OriginalCreditReq RefundReq, CancelReq RecurringOperationReq, StatusReq, TokenizationReq, DeTokenizationReq, WalletDataReq	element	RequestMessage element depending on request type Equivalent JSON objects: saleReq,authorisationReq, originalCreditReq, refundReq, cancelReq recurringOperationReq, statusReq, tokenizationReq, deTokenizationReq, walletDataReq
merchantId	attribute	Merchant number/identification in VPOS=Message senderId
OrderInfo		Orderinfo element of request Message / JSON Object orderInfo
OrderId	xsi:string AN1..50	Merchant defined unique order id / JSON string orderId
OrderDesc	xsi:string AN1..4000	Order description defined by Merchant / JSON string items
OrderAmount	xsi:decimal (max 9+3 or 10+2)	Order amount (decimal number >0.0 and max 12 digits + decimal point) /JSON number orderAmount
Currency	xsi:string A3	ISO4217 alphabetic currency code (USD, EUR) / JSON string currency
PayerName	xsi:string AN1..64	Order payer full name (string..64) /JSON string payerName
PayerEmail	xsi:string AN1..64	Order payer email address (string..64) /JSON string payerEmail
PayerPhone	xsi:string N1..30	Order payer phone number, optional but strongly recommended (string..30) /JSON string payerPhone

	Elements Var1..Var9 Var1, Var2, Var3, Var4, Var5, Var6, Var7, Var8, Var9	xsi:string AN1..4000	Free variable defined by merchant or items. /JSON strings var1,var2,var3,var4,var5,var6,var7,var8,var9
	MOTO	xsi:integer N1	Indicating whether it is a MOTO transaction (1 indicates MOTO) /JSON number moto
	Weight	xsi:decimal	Order shipping weight (kg) if item is shippable and shipping needs to be calculated by VPOS (decimal number >0) and it is supported /JSON number waight
	Dimensions	xsi:string AN1..25	Order shipping dimensions (mm) in format width:height:depth for example a box 200:200:200 (string..25) can be used for shipping calculation if implemented so /JSON string dimensions
	BillingAddress	element address	Element of OrderInfo / JSON Object **billingAddress**
	Country	xsi:string AN2	Billing address country code (string 2 ISO 3166-1-alpha-2 code (US, FI, GB)) /JSON string country
	State	xsi:string AN1..50	Billing address state (string..50) /JSON string state
	Zip	xsi:string AN1..16	Billing address zip code (string..16) /JSON string zip
	City	xsi:string AN1..64	Billing address city (string..64) /JSON string city
	Address	xsi:string AN1..100	Billing address street (string..100) /JSON string address
	ShippingAddress	element:address	Element of OrderInfo / JSON Object /JSON object **shippingAddress**
	Country	xsi:string AN2	Shipping address country code (string 2 ISO 3166-1-alpha-2 code (US, FI, GB)) Optional, required when parameter weight or dimensions are present. /JSON string country
	State	xsi:string AN1..50	Shipping address state (string..50) Optional, required when parameter weight or dimensions are present. /JSON string state
	Zip	xsi:string AN1..16	Shipping address zip code (string..16) Optional, required when parameter weight or dimensions are present. Optional, required when parameter weight or dimensions are present. /JSON string zip
	City	xsi:string AN1..64	Shipping address city (string..64) Optional, required when parameter weight or dimensions are present. /JSON string city
	Address	xsi:string AN1..100	Shipping address street (string..100) Optional, required when parameter weight or dimensions are present. /JSON string address
PaymentInfo			Payment info element of request / JSON Object **paymentInfo**
	PayMethod	xsi:string AN1..20	Card brand VISA, MasterCard others are defined on site valid values are visa for VISA cards, mastercard for MasterCard /JSON string payeMethod
	Card	Element CardData	Object type CardData
	PAN	xsi:string N11..19	Card number /JSON string cardPan
	expDate	xsi:string N4	Card expiration date in format YYMM /JSON string cardExpDate
	holderName	xsi:string AN1..24	Card holder name /JSON string cardHolderName

CVV2	xsi:string N3..4	CVV2/CVC2 security code from card./JSON string cardCvv2
CENC	Xsi:string ..2048	In case on merchant merchant site user browser RSA card data encryption is used this field contains encrypted card data in form of **Base64**(**RSA**(**UTF8Bytes**("pn={pan}&ey={exp year}&em={exp month}&c2={cvv2}&cn={cardholdername}")) Values are urlencoded and with utf-8 char encoding (with javascript encodeURIComponent). This all is handled by server supplied component, merchant just need to forward value as returned to this field content. If this field is present then fields CardPan, CardExpDate, CardHolderName, CardCvv2 must not be present /JSON string cardEncData
WalletInfo	element	A wallets extension element if merchant inititated the xml api payment with Wallets.
walletId	Attribute, required	Indigting wallet type id **GooglePay** for Google Pay™ , **ApplePay** for Apple Pay™, from walletId of walletData
status	Attribute, required	This shall be value of "success"
data	Element, requred	The "data" value from MaksuPay SDK response walltetData object, this contains full payload or content **as is** returned by Google Pay or Apple Pay SDKs
RecurringFrequency	xsi:string N1..3	A value indicating the number of days between the recurring payments. 28 is a special value indicating a month. /JSON number Recurringferequency
RecurringEndDate	xsi:string N8	Recurring end date Format yyyymmdd /JSON string Recurringenddate
InstallmentParameters	element	Installments parameters element /JSON object **installmentParameters**
InstallmentOffset	xsi:integer N1..2	Defines the number of months between the entering of the transaction, n case installment payment /JSON number Installmentoffset
InstallmentPeriod	xsi:integer N1..2	Defines the number of monthly payments in case installment payment. Valid value must be >1 /JSON number Installmentperiod
ExtXOrderId	xsi:string AN1..50	Optional merchant and acquirer agreed extension for recognizing returning customers with submitting previous successful order id of the merchant recognized customer. If functionality is not enabled for merchant this parameter is silently ignored. And if in such case CardPan is missing or is not valid error condition will be generated. Also used in original credit to locate original payment. /JSON string extXOrderId
ExtTokenOptions	Xsi:string N1	Optional for merchant and acquirer agreed token extension Value 1 if request tokenization and PAN is supplied. /JSON string extTokenOptions

ExtToken	Xsi:string N12..19	Optional merchant and acquirer agreed token extension for recognizing payment tokens from previous successful payments. /JSON string extToken
AddFraudScore	xsi:integer	Incoming starting risk score (integer) /JSON number addFraudScore
BlockScore	xsi:integer	Optional block score parameter that will be used to block the transaction if transaction riskScore reaches this value or above. (Postive Integer number) /JSON number blockScore
ThreeDSecure	element	**In case of merchant is processing 3D secure prior to sending payment reequest this field shall be included, this is also included in AuthRes,AuthResultsRes** JSON object t**hreeDSecure**
CAVV	elem xsi:string AN28	**In case of merchant is processing 3D secure prior to sending this xml message this field should contain 3D secure CAVV if authenticated. Base64 encoded value (28 chars) of CAVV of value of 20 bytes**
tdsTransID	elem xsi:string AN40	Three D Server transaction id
dsTransID	Elem xsi: string AN40	DS Servier transaction id
acsTransID	Elem xsi: string AN40	ACS transaction id
eci	attr xsi:string N2	**message this field can optionally contain ECI value**
protocol	attr xsi:string	Required if not 3DS1, value from MPI responses values 3DS1.0.2, 3DS2.1.0
transStatus	Attr xsi:string	Transaction final authentication status
transStatusReason	Attr xsi:string	Transaction status reason
authMethod	Attr xsi:string	Transaction authentication method
Attribute	elem AttributeType 0..n counts	Extra attributes for 3DS2 add all attibutes with names challengeCancel … And any other 3DS2 defined attribute
TransactionInfo	element	Transaction info element (used in recurring cancel operation present in RecurringOperationRequest only) /JSON Object **transactionInfo**
orderId	Attr xsi:string AN1..50	Merchant defined unique order id (of original payment) /JSON string orderId
txId	Attr Xsi:long	txId applicable in StatusRequest messsgae only /JOSN number txId
Operation	xsi:string AN1..25	Predefined String value, Currently supported operation : **Cancel** (to cancel recurring occurring) **Recurring** to execute recurring in series
WalletDataReq		Wallet data request, message for spcecific wallet data processing activities (eg Apple Pay merchant validation)
action	Attr xsi:string ..20	Value validateMerchantSession for Apple Pay

		merchant validation
walletId	Attr xsi:string ..20	ApplePay
orderId	Attr xsi:string ..50	Payment related order id
Attribute	El attribute	Attribute type element For ApplePay validation name=validationURL and value is the URL
Responses/ Notification		
VPOS	element	XML root element
Message	element type Message	Message contents element
version	attribute, xsi:string	Message version default value "5.0" Required
id	attribute, **xsi:ID**	Message unique identifier (values in request and reply messages this must match, no other purpose)
lang	attribute, xsi:string (2)	Message attribute to specify context language (Optional) (ISO 639-1 language code en, fi, sv, el, etc..)
senderId	attribute	Message sender id (Maksu CN)
ts	Attribute xsi:dateTime	Message timestamp when approximate time of when message was created. Example 2015-04-30T12:21:02.402 +03:00
Signature	element ds:SignatureType	The xml signature as defined https://www.w3.org/TR/xmldsig-core/ Canonicalization http://www.w3.org/TR/2001/REC-xml-c14n-20010315 SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256" On JSON messages signature is missing, but is sent on http header and calculates over exact contents posted.
All **Responses**	element	**XML Element** of response type and named as SaleRes, AuthorisationRes, CaptureRes, OriginalCreditRes, RefundRes, CancelRes, RecurringOperationRes Or **JSON objects** saleRes, authorisationRes, captureRes, originalCreditRes, refundRes, cancelRes, recurringOperationRes
merchantId	attribute	Merchant account id
status	attribute	Result status Transaction status in response or notficiation messages AUTHORIZED, CAPTURED - payment was successful (accept order) REFUSED - payment failed, payment was denied for card or by bank (deny order) REFUSEDRISK - payment failed, payment was denied for card by risk score (deny order) CANCELED - only in recurring operation response

		if supsequent recurrings are set to be canceled
errorCode	attribute	In case of errors
Description	element/string 255	Response/Outcome descrptive text
PaymentResponses/Notifications		
orderId	Attribute xsi:string	Same value as in request message OnrderInfo
txId	xsi:long	txId if transaction is related to action/payment or 0 in case of errors / Server supplied transaction id
Currency	xsi:string	Same value as in request message OnrderInfo
PaymentTotal	xsi:decimal	Actual payment amount normally equals orderAmount or orderAmount + any fees if applicable.
Sequence	Xsi:integer	Used with recurrings
PaymentRef	Xsi:string	Remote payment reference like issue approval code.
RiskScore	xsi:integer	Optional risk score calculated by risk scroring subsystem if available
ExtToken	Xsi:string	Optional payment token if tokenization was requested and performed
ExtTokenPanEnd	Xsi:string	Optional payment token related PAN ending 4 numbers
ExtTokenExp	Xsi:date	Optional payment token expiration. (YYYY-MM-DDZ) example 2018-02-01+02:00
ReqcurringNotification		
OrderAmount	Element xsi:decimal	Same value as in request message OnrderInfo
Currency	Element xsi:string	Same value as in request message OnrderInfo
PaymentTotal	Element xsi:decimal	Actual payment amount normally equals orderAmount or orderAmount + any fees if applicable.
PaymentRef	El Xsi:string	Remote payment reference like issue approval code.
Description	El Xsi:string	Error or result description text
Sequence	Xsi:integer	Recurring sequence number
status	Attribute xsi:string	Transaction status in response or notficiation messages AUTHORIZED, CAPTURED - payment was successful (accept order) REFUSED - payment failed, payment was denied for card or by bank (deny order) CANCELED - only in requrring operation response if supsequent requrrings are set to be canceled ERROR - input, sysrtem or network error (deny order)
seqTxId	attribute Xsi:long	The recurring seequence transaction server supplied id
txId	Attribute Xsi:long	Server supplied transaction id of recurring master that started requrring sequence
errorCode	attr Xsi:string	Error code
Attribute	Complex element many	
StatusNotification		
OrderAmount	Element xsi:decimal	Same value as in request message OnrderInfo

PaymentTotal	Element xsi:decimal	Actual payment amount normally equals orderAmount or orderAmount + any fees if applicable.
Currency	Element xsi:string	Same value as in request message OnrderInfo
PaymentRef	El Xsi:string	Remote payment reference like issue approval code.
Description	El Xsi:string	Error or result description text
status, statusBefore	Attribute xsi:string	Transaction status and status before action status notificiation messages AUTHORIZED, CAPTURED - payment was successful (accept order) REFUSED - payment failed, payment was denied for card or by bank (deny order) CANCELED - only in requrring operation response if supsequent requrrings are set to be canceled ERROR - input, system or network error (deny order)
txId	Attribute Xsi:long	Server supplied transaction id
orderId	Attribute Xsi:string	
statusEvent	Attribute Xsi:string	Status change event name (capture, refund, void, reversal)
settlStatusBefore, settlStatus	Attribute Xsi:string	Settlement status before and after event if available
errorCode	Attribute Xsi:string	
StatusRequest		Query for transaction status
merchantId	Attribute element	Merchant number/identification in VPOS
TransactionInfo	element	
orderId	attr Xsi:string	Use either order id ot txid to query results if order id used then all transactions referenced are included such as captures, refunds associated
txId	attr Xsi:long	Use txId to query by txId, only single transaction data is returned
StatusResponse		Response of transaction status conaining one or many TransactionDetails
merchantId	attribute	
orderId	attribute	
txId	attribute xs:long	Transaction identifier
status	attribute	
errorCode	attribute	
Description	element	
TransactionDetails	element	One or many
orderId	attribute	required
txId	attribute	required
OrderAmount	Element xs:decimal	Merchant submitted order amount
Currency	Element xs:string	Order currency
PaymentTotal	Element xs:decimal	Final payment amount (order +/- adjustments, fees etc)
Sequence	Element xs:integer	In case of recurring
PaymentRef	Element xs:string	Payment reference or approval code if available

RiskScore		Element xs:integer	Risk score if available
TxType		Element xs:string	Transaction type
TxDate		Element xs:dateTime	Transaction execution timestamp
TxStarted		Element xs:dateTime	Transaction started timestamp
TxCompleted		Element xs:dateTime	Transaction completed timestamp
PaymentMethod		Element xs:string	Payment method used.
Attribute		Complex element many	Many, rest of the transaction data. As <Attribute name="MERCHANT NO">0000001</Attribute> <Attribute name="USER IP">195.222.10.3</Attribute> <Attribute name="CHANNEL">Redirection</Attribute> <Attribute name="3D STATUS">1 - Fully authenticated</Attribute> <Attribute name="SETTLEMENT STATUS">NA</Attribute> <Attribute name="BATCH NO">28</Attribute> <Attribute name="ISO response code">15</Attribute> <Attribute name="ORDER DESCRIPTION" /> <Attribute name="CARD MASK PAN">4016#####0002</Attribute> <Attribute name="ECOM-FLG">5</Attribute> <Attribute name="ECI">05</Attribute> <Attribute name="PAYEREMAIL">demo@modirum.com</Attribute> <Attribute name="PAYERPHONE">+372 123 1234</Attribute> <Attribute name="BILLCOUNTRY">FI</Attribute> <Attribute name="BILLSTATE">Harjumaa</Attribute> <Attribute name="BILLZIP">76543</Attribute> <Attribute name="BILLADDRESS">Billto tn 6-9</Attribute> <Attribute name="SHIPCOUNTRY">FI</Attribute> <Attribute name="SHIPSTATE">Harjumaa</Attribute> <Attribute name="SHIPZIP">12345</Attribute> <Attribute name="SHIPADDRESS">Viru tn 6-9</Attribute> <Attribute name="EXTACQUIRERID">026</Attribute>
WalletDataRes			
	walletId	Attr xsi:string ..20	ApplePay
	orderId	Attr xsi:string ..50	Payment related order id
	Attribute	El attribute	Attribute type element For ApplePay validation response name=sessionData and value is the content of session data
ErrorMessage		element	Response type of ErroMessage, normally given if request message validation failed or system error.
errorCode		Attribute Xsi:string	Error code

Description	El Xsi:string	Error description text
OriginalMsg	**El Xsi:string**	**Encoded original XML/JSON received in case the error was in content parsed**

Table of field requirements depending on messages:
R - required, O-optional, C-conditional

Field element/ requests	Sale/ AuthorizationRequest	TokenizatationRequest	DeTokenizationRequest	CaptureRequest	OriginalCreditRequest	RefundRequest	CancelRequest	RecurringOperationRequest	SaleResponse	AuthorisationResponse	CaptureResponse	OriginalCreditResponse	RefundResponse	CancelResponse	RecurringOperationResponse	RecurringNotification	Description
Message																	
version	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	5.0
id	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	Unique value of numbers and or chars **xsi:ID** and matching in request, response messages. max length 128
lang	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	Optional iso language code as el, en, ru, fi, et, sv. This is used to set context language in case emails or any other type actions are triggered with this request.
ts	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	Timestamp Required in V5.0
senderId	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	
OrderInfo	R			R	R	R	R										
orderId	R			R	R	R	R										
OrderDesc	O				O												
OrderAmount	R			R	R	R	R										
Currency	R			R	R	R	R										
Var1	O				O												
Var2	O				O												
Var3	O				O												
Var4	O				O												
Var5	O				O												
Var6	O				O												
Var7	O				O												
Var8	O				O												
Var9	O				O												
MOTO	O																
Weight	O																
DImensions	O																
BillingAddress	O																optional billing address element and sub element, may

																	be required depending on Acquirer
ShippingAddress	O																optional shipping address element and sub element
PaymentInfo	R			O[1]		O[1]	O[1]										O[1] – optional if system set up supports
PayMethod	R[3]			O[1]		O[1]	O[1]										O[1] – optional if system set up supports R[3] – normally required but may be omitted if system is set up so
Card	R																Card object type CardData, required if card payment
PAN	R[2]			O[1]		O[1]	O[1]										O[1] – optional if system set up supports, R[2] – optional if system set up supports, Not present if ENC present.
ExpDate	R																O[1] – optional if system set up supports, Not present if CardEncData present.
CVV2	C																Required if not MOTO and required for card type/brand, Not present if CardEncData present.
HolderName	R																Cardholder name, required if Card element prcent
Or CENC	C																Used if RSA card encryption then Pan, ExpDate, HolderName and CVV2 shal not be present but are in CardEnc encrypted
Or Token	C																If a token is used instead of PAN
Or WalletInfo	C																Depends on wallet case
walletId	C																Required if WalletInfo present
status	C																“success” required, if WalletInfo present
data	C																Required if if WalletInfo present
PayerName* PayerPhone	O																PayerName applicable for non card payments* else in card data Optional but highly recommended. Not present if CardEncData present.
PayerEmail	R																Payer email required
PayerIpAddress	R																Payer ip address
ExtXOrderId ExtTokenOptions	O[2]				R												O[2] – may be present instead of CardPan if system set up supports. Required for original credit to lookup source payment.

RecurringParameters	C															Required for recurring payment
RecurringFreq	C															Required for recurring payment
RecurringEndDate	C															Required for recurring payment
InstallmentParameters	C															Required for installment payment
InstallmentOffset	C															Required for installment payment
InstallmentPeriod	C															Required for installment payment
TransactionInfo								R								
OrderId								R								
Operation								R								
Signature	R	R		R	R	R	R	R	R	R	R	R	R	R	R	Required for all
Attribute el name="txId"	C															Depends on wallet case
Attribute el name="walletId"	C															Depends on wallet case
Attribute el name="authMethod"	C															Depends on wallet case
Card		R														CardInfo
Token			R													TokenInfo
Responses/Notification																
orderId									R	R	R	R	R	R	R	Order Id supplied by merchant originally
OrderAmount									R	R	R	R	R	R		R	
PaymentTotal									R	R	R	R	R	R		R	
Currency									R	R	R	R	R	R		R	
Status									R	R	R	R	R	R	R	R	Status
TxId									C	C	C	C	C	C		R	In case of transaction processing has started (no rejection due invalid input request), In case of recurring Notificatuion this is master recurring transaction id
Sequence																R	Sequence of recurring in notification
SeqTxId																R	The executed recurring sequence transaction id
PaymentRef									C	C	C	C	C	C		C	Payment reference such as approval code if available
RiskScore									O	O							Optional risk score calculated by risk scoring subsystem if available
errorCode									C	C	C	C	C	C	C	C	Error code in case of Status=ERROR
Description									O	O	O	O	O	O	O	O	Optional error description
Attribute									O	O	O	O	O	O	O	O	Optional attributes, may be custom per implementation.
OriginalMsg																	In general error message only to copy back the erroneus content received for merchant debugging.
Signature									R	R	R	R	R	R	R	R	Required for all XML

StatusRequest/StaturResponse

Field element/ requests	StatusRequest	TokenizationRequest	DeTokenizationRequest				StatusResponse	TokenizationResponse	DeTokenizationResponse					Description
StausRequest														
merchantId	R	R	R											
TransactionInfo	R													
orderId	C													Either OrderId or TxId is required
txId	C													Either OrderId or TxId is required
StatusResponse							R							
TransactionDatails							R							
OrderAmount							R							
Currency							R							
PaymentTotal							R							
Status							R							
TxId							R							
PaymentRef							O							
Description							O							
TxType							R							
TxDate							R							Transaction exec date
TxStarted							R							Transaction started
TxCompleted							O							May bemissing if transaction did not complete due errors.
Attribute							O							List of attributes depending on what information is available. Attribute name can be one of the following: MERCHANT NO - merchant number, REFUNDED AMOUNT - amount refunded if availble, USER IP - use ip if available, CHANNEL - channle originated 3D STATUS - status CAPTURED AMOUNT - captured amt

														SETTLEMENT FILE - settl file name BATCH NO - batch number ISO response code - iso response if available ExtData – additional data from external payment systems if available ORDER DESCRIPTION - order descr CARD MASK PAN - masked pan 5+3 or 4+4 or 6+2 INSTALLMENT SEQUENCE, INSTALLMENT PERIOD, INSTALLMENT OFFSET, RECURRING SEQUENCE, RECURRING END DATE, RECURRING FREQUENCY, ECOM-FLG - ecom flag in auth message, ECI - eci from mpi, VAR1..VAR9, PAYEREMAIL, PAYERPHONE, BILLCOUNTRY, BILLSTATE, BILLZIP, BILLADDRESS SHIPCOUNTRY, CancelRequest RecurringOperationRequest, StatusRequest, TokenizationRequest, DeTokenizationRequestSHIPSTATE SHIPZIP SHIPADDRESS BONUS PARTICIPATION*, BONUS REF* BONUS ADJUSTMENT* BONUS STATUS* BONUS DETAILS* RETURNING USED** RETURNING ORDER ID** * - Only possible if with special bonus loyalty extension. ** - Only possible if with returning customer extesnion.
Card	R	R							R					CardInfo type ref required, if CSE then encData reuired else pan,exp

													reuired
Token	R		R					R					
id	R	R	R					R	R				
status								R	R				
statusMessage								O	O				

O[1] - if supported feature then fields may not need to be present if not supported then the fields are required. Availability of this option shall confirmed with system administrator (Your customer support). If values not sent then whole PaymentInfo element shall be excluded from message.

R[2] and O[2] - If system supports and merchant is set tp participate in returning customer recognition extension then if merchant already has a successful order with a card with this customer and the card is still valid and customer chooses to make this next order with same card and the days and amounts between orders are in certain limits then merchant may send ExtXOrderId instead of CardPan. In such case if validations are passed system automatically uses pan from previous specified order. Recommended maximum period between previous order and next returning customer extension order could be 6 months (180 days).

R[3] if acquirer/processor has set up so then PayMethod may be omitted. Default required and recommended.

Currently supported operations:

AuthorisationReq -make a pre authorization
CaptureReq - capture a pre authorization
RefundReq - make refund
SaleReq - make a payment
CancelReq - make reversal for an unsettled transaction
RecurringOperationReq - with operation Cancel, cancel recurring master scheduling
RecurringNotification – Optional message posted to merchant if a recurring child is executed on server, merchant does not need to send response XML to this on accept merchant server should respond with http status code 200/OK or in case merchant does not recognize the transaction 406/Not Acceptable or 400/Bad Request if the message format is invalid. Server just acknowledges the response code and performs no additional actions based on merchant response code.
StatusReq - query transaction status
TokenizationReq - tokenize a card to token
DeTokenizationReq - de tokenize a token back to card date
StatusNotification - optional status change notifications/hooks if merchant needs this.

Error code values:
Filled in case status is ERROR with following values
M1 – Invalid merchant id
M2 – Authentication failed (wrong password or digest or signature)
SE – System error (message contains error id, system or configuration error to be investigated)
SD – Service disabled (service disabled)
XE – Invalid XML/JSON request not parseable or does not validate
I0 – Invalid or unsupported request
I1 – Message contains invalid data item or missing required item
I2 – Message contains invalid installment parameters
I3 – Message contains invalid recurring parameters
I4 - Message contains invalid or mismatching card data
I5 - Message contains invalid expiration date card data
I6 – Selected payment method does is not supported or not matching the payment card
O1 – Operation is not allowed because logic is violated or wrong amounts
O2 – Original transaction is not found to perform operation.

May be also filled in case of status is REFUSED
with acquirer network supplied ISO response code

3.2 3D-Secure API

Maksu 3DS Secure API facilitates 3DS 2 authentication via its simple API to utilize 3DS Server services.

Endpoints both XML or JSON:
Production: https://pay.eu.maksupay.com/vpos/vposapi
Sandbox: https://pay.test.maksupay.com/vpos/vposapi

Messages structure is identical to Payments API except the functional messages used are different.

Field/request	Type	Description
Request		
VPOS, Message, Signature etc		See description in payments API section
3DS API messages: TDSMethodReq/ TDSMethodRes AuthReq/AuthRes AuthResultsReqAuthResults Res	element	RequestMessage element depending on request type Equvalent JSON objects: tdsmethodReq/tdsmethodRes authReq/authRes, authResultsReqReq/ authResultsReqRes
ThreeDSMethodReq	El, Message object	Message to check if Three DS method applies for card range
PaymentInfo		
Card	Element CardInfo	Object type CardInfo
PAN	xsi:string N11..19	Attribute Card number /JSON string PAN
expDate	xsi:string N4	Attribute Card expiration date in format YYMM /JSON string expDate
holderName	xsi:string AN1..24	Attribute Card holder name /JSON string holderName
or **WalletInfo**		Object type WalletInfo
walletId	Attribute, required	Indigting wallet type id **GooglePay** for GooglePay™ , **ApplePay** for ApplePay™, from walletId of walletData
status	Attribute, required	This shall be value of “success”
data	Element, requred	The "data” value from walltetData returned by Maksu SDK. This contains full payloas of GooglePay or ApplePay SDK returned data, including encrypted token.
orderId	attribute/property	Required
merchantId	attribute/property	Required
ThreeDSMethodRes	El, object	Response to Three DS Method request
ThreeDSMethodContent	El, xsi string	Html content to be rendered in browser
orderId	attribute	Same as in response
refId	Attribute xsi:long	Transaction id, for later requests in relation to that aurthentication transaction with that pan
merchantId	Attribute	required
status	attribute	required
errorCode		Present when error
Description	Element xsi string	Action result description text
AuthenticationReq	element	Authentication request message
OrderInfo	Element object	See desciption at Payment API
PaymentInfo	Element object	See desciption at Payment API
sessionData	Ekement xsi:string	A session identifier to recognize tx in callBackURL

callBackURL	Element Xsi string	Callback url to call when 3DS authentication session is completed ad issuer. Recommended implementation is that this value is set to “javascript” and instead of callback url API service rendered contenr calls a javascript postMessage function to inform parett window / frame about 3ds window ended/closed
		In case of javascript content posted with window.sendMessage function is JSON data:
		{
		“sd” : “sessionData”,
		“status” : “status”
		}
		sessionData is same value send in that request,
		status can have values:
		SUCCESSS – 3ds authentication succes
		UNABLE – 3ds authentication “U” cases
		ERROR – authentication error
		FAIL – authntication failure
		If callBackURL was url service makes http POST with parameters
		sd=sessionData
		status=status
		with same values as in json message in case of javascript

Attribute	Element AttributeType	Many, 3ds2 related data attributes and values
name	Atrribute name	The attribute name eg: TDS2_BrowserIP, TDS2_BrowserAccept, TDS2_Navigator_language, TDS2_Navigator_javaEnabled, TDS2_Navigator_jsEnabled, TDS2_Screen_colorDepth, TDS2_Screen_height, TDS2_Screen_width, TDS2_Screen_pixelDepth, TDS2_TimezoneOffset, TDS2_UserAgent
AuthenticationRes	El object	Response to **AuthenticationReq**
ThreeDSecure	Element object	
CAVV	El, xsi string	Peresent on authentication success or proof of attempt cases
tdsTransID	Attr xsi: string	3DS Server trans id
dsTransID	Attr xsi: string	Directory server trans id
acsTransID	Attr xsi: string	ACS trans id
ECI	Attr xsi string	3DS E commerce indicator
protocol	Attr xsi string	Authentication 3DS Protocol and version indicator
transStatus	Attr xsi string	Authentication status
transStatusReason	Attr xsi string	Authentication status reason
authMethod	Attr xsi string	Authentication method
Attribute	Element AttributeType	0...Many, 3ds2 related data attributes and values, depends on protocol requirements

ThreeDSMethodContent	Xsi: string	Present if **TDSMethodReq call was omitted, but required for card range, then Three DS method content is to be rendered in browser iframe and then continue with AuthenticationReq having set same orderId and same pan**
ThreeDSContent	Xsi: string	Present if authentication challenge flow is required, the content is to be rendered in browser iframe in ralation to main payment page.
orderId	Attribute, Xsi: string	Original merchant orderId
refId	Attribute, xsi:long	Authenticatoin transaction record identifier
merchantId	Attribute	required
status	attribute	required
errorCode	attribute	Present when error
Description	Element xsi string	Action result description text
AuthResultsReq	El object	Authentication results request to query final authentication outcome after challenge flow is completed.
refId	Attribute, xsi:long	Authenticatoin transaction record identifier, required
merchantId	Attribute	required
AuthResultsRes	El object	Authentication results response with authentication data if authentication was successful
orderId	Attribute, Xsi: string	Originale merchant orderId
refId	Attribute, xsi:long	Authenticatoin transaction record identifier
merchantId	Attribute	required
status	attribute	required
errorCode	attribute	Present when error
Description	Element xsi string	Action result description text
ThreeDSecure	Element object	
CAVV	Attr, xsi string	Peresent on authentication success or proof of attempt cases
tdsTransID	Attr xsi: string	3DS Server trans id
dsTransID	Attr xsi: string	Directory server trans id
acsTransID	Attr xsi: string	ACS trans id
ECI	Attr xsi string	3DS E commerce indicator
protocol	Attr xsi string	Authentication 3DS Protocol and version indicator
transStatus	Attr xsi string	Authentication status
transStatusReason	Attr xsi string	Authentication status reason
authMethod	Attr xsi string	Authentication method
Attribute	Element AttributeType	Many, 3ds2 related data attributes and values, depends on protocol requirements

Status values in for authentication request responses:

INITIAL – authentication initialization stage, use ThreeDSMethodContent and include to payment page.

NOTDSMETHOD – no three ds method for card, continue with authentication flow

INTDSMETHOD – internal status, authentication in 3ds method (not returned in message)

INTDSMETHODC – authentication in 3ds method, continue to authentication flow

INAUTH – internal status that authentication in porgress (not returned in message)

REJECTED – authentication failed with reject status “R”

AUTHFAIL - authentication failed with not authenticated “N” status

AUTHWALLET – applicable response when sent wallet data used to check 3ds requirements,

this status meaning wallet had 3ds cryptogram and not 3ds needed.
UNAVAILABLE - authentication failed with unavailable “U” status
CHALLENGE – authentication challenge was requested, use ACS redirect content to redirect user to authentication at issuer ACS.
COMPLETED – authentication completed with fully authenticated “Y” or attempt “A” astatus.
ERROR – error found or occurred in processing or validating data.

3.3. Signature calculation with XML API V5.0

Signatures shall be calculated and verified according to documentation https://www.w3.org/TR/xmldsig-core/
Canonicalization method to be used is http://www.w3.org/TR/2001/REC-xml-c14n-20010315
SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"
DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"
The signed element is Message element referenced with its **ID** attribute named **id**.
ID attribute is an attribute which type in schema is defined as xsd:ID.

Messages sent by merchant are signed by merchant private key and verified with merchant certificate public key.
Messages sent by VPOS processor service are signed by service provider private key and validated with service provider provided certificate public key.

The canonicalization method to be used is
http://www.w3.org/TR/2001/REC-xml-c14n-20010315

Note that the XML documents should be handled with namespace aware xml libraries (parser/serializer). When the Message element is serialized and canonicalized it should contain xmlns namespace attribute. See from next section XML message with digest/signature example.

Note for XML/JSON API with Three D Secure:
This is 2 step processing at first step merchant should implement 3DS api session as described in 4.2 3D Secure API and obtain the Three D Secure authentication results from there and then next step is to fill the corresponding values to XML/JSON API ThreeDSecure element and proceed with XML/JSON api payment request to VPOS.

3.4. Browser client side encryption and wallets for XML/JSON API using PSP SDK

3.4.1 Using VPOS client side encryption for API messages

Merchants who want to use option of browser RSA card data encryption need to import service provider javascript to their payment page.

Client side encryption java script is to be imported from url like:
Production:

```
<script type="text/javascript"
src="https://pay.eu.maksupay.com/vpos/csescript.js?
version=5&mid=1234567&date=201804271539&signature=xxxxx">
```

Sandbox:

```
<script type="text/javascript"
src="https://pay.test.maksupay.com/vpos/csescript.js?
version=5&mid=1234567&date=201804271539&signature=xxxxx">

</script>
```

with following query parameters to authenticate request

Field (HTTP GET parameter)	Required / Optional	Description
version	R	**Version 5**
mid	R	Merchant id supplied (integer number) max length 30
date	R	Current date and time in format YYYYMMDDHHMM
signature	R	Same as in section 2.4. Calculation of the Signature

Make sure url parameters are properly urlencoded in url and not encoded when calculating signature.

To capture user entered card data merchant needs to create a form with following inputs

```
<form name="dummyInputs" action="https://neversumbit.this/neversumbit.this">
  Card number: <input type="text" id ="card.pan" size="20" maxlength="40"/><br/>
  Expiration <select id="card.expiryYear"> with year options from now  to now +20
(2018..2038)</select>
  <select id="card.expiryMonth"> with month options 01..12</select> <br/>
  Name on card: <input type="text" id ="card.holderName" size="20" maxlength="40"/><br/>
  CVV2 <input type="text" id ="card.cvv2" size="5" maxlength="5"/>
</form>
```

Important: inputs and selects must be without name attribute and that form must never submitted to server.
To capture encrypted card data merchant shall call script function

```
var results=getEncryptedCardData(cvvRequired, nameRequired)
```

if parameter cvvRequired or nameRequired is false then input of this value is not required)

in return of function call merchant receives an object with properties.
If exists element 'cardEncData' in result object (`results['cardEncData']` then inputs were considered valid by format and encryption was successful and merchant shall take this value and proceed to XML api payment in server side. Optionally `results['cardType']` contains information about card type detected.

If merchant system captures inputs independently then it can use equivalent method encryptCardData with parameters passed independently of input means

var results=encryptCardData(cardPan, cardExpiryYear, cardExpiryMonth, cardCvv2, cardHolderName, cvvRequired, nameRequired)

Or if its used with combination of existing token from tokenization and only CVV need to be captured can be used function

var results=encryptCVVOnly(cardCvv2);

In case of input or other errors merchant can extract following error from result using keys/properties and display them user if needed

errorPan	(value "Card number must be number with length 12..19")
errorExpiryYear	
errorExpiryMonth	
errorCvv2	
errorHolderName	
errorCardEncData	in case of encryption error.
errorDebug	error details in case of encryption error.

3.4.2 Browser client side card data encryption for API messages with included PSP iframe

In case security is first priority then even more recommended approach is that merchant site icludes iframe with card input fields from service provider. Such iframe content is on browser level protected from data mining by javascripts on merchant site by purpose or by site compromise. In that case merchant may be exmpt of full PCI-DSS requirements (verify with Your PSP or auditor) and integration is performed as follows:

On merchant site payment page where is desirable place for card data entry merchant include client script with iframe

```
<script type="text/javascript"
src="https://pay.test.maksupay.com/vpos/csescript.js">     </script>
<iframe id="vpos-cseiframe" style="display: inline-block;" width="480"
height="110"
src="https://pay.test.maksupay.com/vpos/cseiframe.html?
version=5&mid=1234567&date=201804271539&signature=xxxxx”>       </iframe>
```

Url parameters are exactly the same as when including javascript in section above.
Note: iframe only contains input fields and selects, no buttons. Button must be on merchant site and then call function

```
getEncryptedCardDataFromIframe(cvvRequired, nameRequired,callbackFunction);
```

This callback function is called with parameter results in section 5.0above
For example

```
function myCallBackFunction(results)
{
  if (results['cardEncData']!=null)
       {
              document.getElementById("cardEncData").value=results['cardEncData'];
              document.getElementById("payform").submit();
       }
  else
       { alert("Card data not valid");  }
}
```

3.5. Payments with Google Pay™

Merchants who want to include GooglePay buttons on their self hosted payment pages must ensure they are familiar with GooglePay and adhere **Google Pay Web brand guidelines**, those guides are available at following Google pages:

https://developers.google.com/pay/api/web/overview
https://developers.google.com/pay/api/web/guides/test-and-deploy/integration-checklist
https://developers.google.com/pay/api/web/guides/brand-guidelines

To enable Google Pay functionality, Maksu account manager configures this option on merchant account.

3.5.1. Including Maksu hosted Google Pay button on Your payment page.

Important: In this mode of intergation Your intgration need to be rgistered and get approved in Google Pay console as on Maksu production domain that is **pay.eu.maksupay.com.**

To include GooglePay button to Your own hosted payment page use a iframe tag with src url set as follows:

1. Set http header **Cross-Origin-Opener-Policy** for Your page displaying GooglePay when loaded to value "**unsafe-none**"

Note: If not set iframed Google Pay button is unable to open Google Pay window!

2. Add iframe element with id “vpos-GooglePay” this is where Google Pay button will be rendered

Production:

```
<iframe id="vpos-GooglePay"
      src=”https://pay.eu.maksupay.com/vpos/googlepay.html?
      version=5&mid=1234567&date=202503251539&signature=xxxxx” ></iframe>
```

Sandbox:

```
<iframe id="vpos-GooglePay"
      src=”https://pay.eu.maksupay.com/vpos/googlepay.html?
      version=5&mid=1234567&date=202503251539&signature=xxxxx” ></iframe>
```

Field (HTTP GET parameter)	Required / Optional	Description
version	R	**Version 5**
mid	R	Merchant id supplied (integer number) max length 30
date	R	Current date and time UTC in format YYYYMMDDHHMM
signature	R	Same as in section 2.4. Calculation of the Signature

Make sure url parameters are properly urlencoded in URL and not encoded when calculating signature.

then include VPOS client JS libaray and intialize with order info and callback function that will return You wallet payload once user wallet session is ended successfully.
This payload You need to put to payment API messages into **PaymentInfo.WalletInfo**

```
<script type="text/javascript"
      src="https://pay.eu.maksupay.com/js/vposclient.js"> </script>
<script>
      function myEndWalletSession(data)
      {
            document.getElementById(“myWalletData”).value=data;
       }
```

```
</script>
<script>
  VPOSClientJS.initWalletSession(orderInfo, options, function(data) {
      myEndWalletSession(data);
      }
   );
</script>
```

Order info obect need to be following properties

- orderId – the unique order id
- orderTotal – total price of order
- currency – the currency used for orderTotal.

Options:

- backgroundColor – default white, html colors (iframe background color)
- buttonColor – default black, possible values: black, white
- buttonType – default order, possible values: buy, checkout, donate, order, pay, plain, subscribe
- buttonRadius – value 0...100 default 18 (for rounded corners)

Additionally Maksu defaults (required) are for BillingAddressParameters

format: FULL, phoneNumberRequired: true

Values for gateway and gatewayMerchantID are preset in Maksu iframed button as necessary to render service.

Values for card networks are preset in Maksu iframed button as necessary to render service, currently visa and mastercard

Example how button will render by default :

3DS flow applicability:

To determine if particular GooglePay interactions are already having 3ds cryptogram or needs 3DS performed before payment, send the wallet data in , ThreeDSMethodReq AuthenticationReq, if its PAN_ONLY then authentication api returns infomration needed to continue on normal 3DS flow, if it contained 3ds cryptogram (authenticated when added to wallet) then those requests return status AUTHWALLET and no 3DS flow needed.

Example payment message with GooglePay data:

```
<VPOS xmlns="http://www.modirum.com/schemas/vposxmlapi5"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#"><Message id="M1743011079509"
senderId="200002" ts="2025-03-26T17:44:39.510Z" version="5.0"><SaleReq
merchantId="200002"><OrderInfo
orderId="O1743011041215"><OrderDesc/><OrderAmount>1.25</OrderAmount><Currency>EUR</
Currency></OrderInfo><PaymentInfo payMethod=""><WalletInfo status="success"
walletId="GooglePay"><data>{"apiVersion":2,"apiVersionMinor":0,"email":"a#####@#####.com
","paymentMethodData":{"description":"Mastercard •••• 6199","info":{"assuranceDetails":
{"accountVerified":true,"cardHolderAuthenticated":false},"billingAddress":
{"address1":"Kikerpuu tee
23","address2":"1","address3":"","administrativeArea":"Harjumaa","countryCode":"EE","loc
ality":"R######e","name":"A######
K######","postalCode":"7####6","sortingCode":""},"cardDetails":"6199","cardNetwork":"MAS
TERCARD"},"tokenizationData":
{"token":"{\"signature\":\"MEUCIQDLxhEeZsHJXZys3snPERbbqwRIBPIqg9Za4OLaQ6iQmwIgXLAY28bg5
jyAr3ixz5CFTpOIrrvMSMqd6oyuf5/8vSM\\u003d\",\"intermediateSigningKey\":
{\"signedKey\":\"{\\\"keyValue\\\":\\\"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEgvYC0f4nzc2dY
PztRprz5xp05rO14JERLqJv+s1VC2j3WS8BPhBbKDLzXeyyk/4IfC5//dyjVs7rZqA2gpnl0g\\\\u003d\\\\
u003d\\\",\\\"keyExpiration\\\":\\\"1743701176433\\\"}\",\"signatures\":
[\"MEQCIAMiksS7CGNuMZdMK35bzVxlLQW6sfOrAnbpNGIYEf5PAiALCJKgMSqXwsX01mcshH/
megyVopWNmJj018o9JdIalw\\u003d\\
u003d\"]},\"protocolVersion\":\"ECv2\",\"signedMessage\":\"{\\\"encryptedMessage\\\":\\\
"VagPck6LnMg3P9JgMCy3X9paF+oIK7jQXMhB7F6YdUBCqGqgQsZMR/
HBCH98xI2V36qKoAoVsJ9EwtbmMlNLl8aJ4dCgDVL8eC0RZyQ0/Q3ni2b7hhmqgOp3lhG8BFp/
```

**tAS+1xjCTpHuicp3PYScn0+ThlF11JTXSDvlnOnsHmvgIU0zsjsmqIA2nWoknkymWm+blcoLg8xL1y6mPQH2vozs
FUILD+FEP3Z4G1OvYY1B1nSLh+cuMDbMArrts8hIJ4AkD6SnIDv/geG84Hsq13HuEqmtTLr/FQ9t0oV7l1q17/
QjLKIaDO0CDL8E+s4QkoRo4a2JrQJrvMARdhLXp7e33sqWetheVzLtbbDiZmcdl+SNXb2MZLCZp0oGQ9jR/
MbulmNmKJJhid6zaHlMKZXGn9k4nwGknTyVvD9JFSU1swXbCL6W+/L4IxVJVCT1KLa3G3nCd/
HtjMAHrTIKfDMit6UmeDIeE2Dc5ig82BfUAYtrXaJBCBjEvsH/CxgrZf17R0rP3dvBHo8/IX0o8VPOKN+Z2y/
bUGps5PbboH1Tajy3iQ\\\\u003d\\\\
u003d\\\",\\\"ephemeralPublicKey\\\":\\\"BHDdZhxJuVIiIVQ1IaaLog2A4klvb6TxE+A5mgoZJ9PwF0v
zQRUSngKDUbkMc01RDFpiOqA29nNbzf0/g3FZFSU\\\\
u003d\\\",\\\"tag\\\":\\\"U9YJF3SipZgRroqe8rawj47slyztum5T2bq9MznfsAM\\\\
u003d\\\"}\"}","type":"PAYMENT_GATEWAY"},"type":"CARD"}}</data></
WalletInfo>**<PayerEmail>u####@#####.com</PayerEmail><PayerIpAddress>192.168.8.98</
PayerIpAddress></PaymentInfo></SaleReq></Message><ds:Signature
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<ds:Reference URI="#M1743011079509">
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>yNOUFFamCz7Rt/C8S0XwkHLQuM+PQmb5rRWAX24jexo=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
KRMTTbeHbKOK3v6bPPKyGLkdKQr/GszdTZvIHVszyw2DD1oXssaeaVLsFWpYWxQX985xohU3QniF
fHtT92ux1EkFWTSBedA/D2I0AXZCbMnS6lEN0JYfZg7EgwdPDyzjjCuTRqbiswbCQiLf/LSytyNr
JlCT6nUvhMcmw52Eyiz+JMmiwjiKGUtNwz008elGSUOeZYUVJsWdGX0EXeK4WDpgi1MuofbRGMtV
DSYIWNxHov50fgdjPUyws+uiNW4ALuS1oNH2wbZJUiKaD1+b364+WT6B3Ge6oYobfQfLAA2Voz1t
mru5YUkBe1DBhN/c6Y1keG94JwAT5n5Elp1ijw==
</ds:SignatureValue>
<ds:KeyInfo>
<ds:X509Data>
<ds:X509Certificate>
MIIDxzCCAq+gAwIBAgIUDBt1MUWfMQnotsdXcrZ387CY4oQwDQYJKoZIhvcNAQELBQAwczELMAkG
A1UEBhMCRUUxCzAJBgNVBAgMAkhNMRAwDgYDVQQHDAdUYWxsaW5uMRMwEQYDVQQKDApNYWtzdSB0
ZXN0MQ8wDQYDVQQLDAYyMDAwMDIxHzAdBgNVBAMMFnZwb3N0ZXN0cy5tYWtzdXBheS5jb20wHhcN
MjQwNDMwMTE1NDA5WhcNMjgwNDI5MTE1NDA5WjBzMQswCQYDVQQGEwJFRTELMAkGA1UECAwCSE0x
EDAOBgNVBAcMB1RhbGxpbm4xEzARBgNVBAoMCk1ha3N1IHRlc3QxDzANBgNVBAsMBjIwMDAwMjEf
MB0GA1UEAwwWdnBvc3Rlc3RzLm1ha3N1cGF5LmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAL81CItSrfk12TTOviXKXy+YQXker6UkpzTDf2ekzGpzA56fbZGgNnweby/RvZ2yGlSn
KoInWl0oZ8vLHDXXMpsdnAqL8MwOvbVN0otRTZ1W36Ca65e/0l2kTDwjTtl9CHaIYxN3F9vq9dlA
Y3euGpb0O6gBzGxCr2XIxHoGadB5vEfX9bfMYgswXYWOFxsAEu+lS4uo8bVa+L9reH0cU6aRbfMw
rzdSvUn/KazECN2VIXD7QPckkV8tksrj0hB38yMa188iU9vpZ1oePS662tQRkxjAsm4LcM4/3w18
Y/ky0emEWn14p/Yza0IWY1L04SrCEH4EPWKRSzy+RQixLT0CAwEAAaNTMFEwHQYDVR0OBBYEFMYo
d11iUYfyLbLd4HnF3tJwVkVlMB8GA1UdIwQYMBaAFMYod11iUYfyLbLd4HnF3tJwVkVlMA8GA1Ud
EwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAEB5FYdHlSDdrHdAJIsZMN1IuSDthH8dE1tv
HBZ5krHwwpXLBNMiyUaFC4G/f07dXpz9lRhgL25nMOM5mCHVCfTZ/FFTMaJdQi4yZcS9hZjkZwqp
pGp0+G8tGhI/bPW8n4tFnk1HK0FN9/d2jv1qqsWaIUdpHed7uz5OXMkzDdPjb8eO/QjfnJBHN+rm
z1UPvGN0cWATBNkbLiHkWvVM94Z+gVJGv9CnJCdn6xMD5O6uhkYn51LRYTb/yAyuyKhl9mY43U3G
A+/5r2UXQ3Ec8dhNNxvhr1WkQb0Hvuw6vpTf4BRy5MH/ermH5BJpFDoDNpJYUiSl+lxvCctsfNjJ
Xwo=
</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
</ds:Signature></VPOS>

3.5.2. Including self integrated hosted Google Pay button on Your payment page.

For implementing Your own intgration with Maksu as PAYMENT_GATEWAY of handling and displaying Google Pay button on Your pages follow google implementation guide https://developers.google.com/pay/api/web/overview

When preparing the Google Pay intialization parameters You need to set tokenizationSpecification type as 'PAYMENT_GATEWAY' and for its **parameters** section **gateway** value is is to be set **"maksupayeu"**
and for **gatewayMerchantId** id is to be set Your merchant account unique id on Maksu portal, if unsure ask from maksu support support@maksupay.com

For **merchantInfo** merchant id is to be set merchant id what is associated with Your Google Pay merchant account. That is a value 12 18 characters typically starting with letters "BCR".
In this mode of intergation Your intgration need to be rgistered and get approved to your domain.

3DS and Pay flow API is parts are identical to utlizing Maksu hosted Google Pay button eg populating WalletInfo object with proper wallet id and token/payment data collected from Google Pay user interaction.

3.6. Payments with Apple Pay™

Merchants who want to include Apple Pay buttons on their self hosted payment pages must ensure they are familiar with Apple Pay and adhere **Apple Pay for web instrictions and brand guidelines**, those guides are available at following Apple pages:

https://developer.apple.com/documentation/applepayontheweb
https://developer.apple.com/documentation/applepayontheweb/payment-request-api
https://developer.apple.com/documentation/applepayontheweb/displaying-apple-pay-buttons-using-javascript

To enable Apple Pay functionality, Maksu account manager configures this option on merchant account with merchant specified domain and informs it about and its merchant identifier that is registred for service.

3.6.1. Including self integrated hosted Apple Pay button on Your payment page.

For implementing Your own intgration with Maksu Payment Gateway of handling and displaying Apple Pay button on Your pages follow Apple implementation guide https://developer.apple.com/documentation/applepayontheweb

When preparing the Applee Pay intialization parameters You need to set **merchantIdentifier** to identifier provided by Maksu when Your account is made enabled and registered with Apple Pay**.**
This Your merchant account unique id on Maksu portal, if unsure ask from maksu support support@maksupay.com

Apple Pay flow also contains a merchant session validation that is executed when Apple payment sheet is opening. Apple Pay is waiting for session info until it allows user to proceed with payment.
This session info and merchant validation can be made via maksu api using WalletDataRequest message.

In javascipt code when **PaymentRequest** is initialized its needed to be impllemented handler
that performs merchant validation fetches session info and calls complete function on event

```
this.paymentRequest = new PaymentRequest(paymentMethodData, shoppingCartDetails,
paymentOptions);

this.paymentRequest.onmerchantvalidation = (event) =>
       {
              this.validateMerchant(event);
       };
```

This merhod validateMerchant shall inform Your backend about the validation url (event.validationURL)
then Your backend needs to send a session validation message to Maksu XML/JSON endpoint and from its response You need to extract merchant session and call

```
              event.complete(sessionData)
```

Example request to get session data:

```
<VPOS xmlns="http://www.modirum.com/schemas/vposxmlapi5"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#"><Message id="M1749126792720"
senderId="200002" ts="2025-06-05T12:33:12.720Z" version="5.0">
<WalletDataReq action="validateMerchantSession" merchantId="200002"
orderId="O1749126792721" walletId="ApplePay">
<Attribute name="validationURL">https://apple-pay-gateway-cert.apple.com/paymentservices/
startSession</Attribute>
</WalletDataReq>
</Message><ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<ds:Reference URI="#M1749126792720">
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>i5PYoRn+gORZVZvRv2l5mvUkkqSXf1yP1FDfJePMEhI=</ds:DigestValue>
```

```
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
uQH9SGvLgyFJGtiaUxSzR9ApbcUQTB4AdukKnlAMBrky6aEQ/aqsv3kSbqrAvk9FK4At7nvvTsJg
GP7731mfAvMIGC2y9A5hRZoEhk221OOTSNtcOOFRb5lIfs1DNll5WGbwKS2W7XK9KaMCE9XjVJ/f
nsXLAmKC8SwI9gZH6vbxzevXPX9JH64W+UQI66VD/rd8sqS5PinMdxCQ/UJwkvGmWkwvim+KnKim
YB/Ea+z3xhNHXbjm5MyryoUHYxIyeFyB45GiBTfynSQjuzWY6I3pHg+hyepcmtKpOZqSwtS5n5ge
8LLqhH8vqAhaqKEtxjpA3NANJ7fu6q8IUDzNtQ==
</ds:SignatureValue>
<ds:KeyInfo>
<ds:X509Data>
<ds:X509Certificate>
MIIDxzCCAq+gAwIBAgIUDBt1MUWfMQnotsdXcrZ387CY4oQwDQYJKoZIhvcNAQELBQAwczELMAkG
A1UEBhMCRUUxCzAJBgNVBAgMAkhNMRAwDgYDVQQHDAdUYWxsaW5uMRMwEQYDVQQKDApNYWtzdSB0
ZXN0MQ8wDQYDVQQLDAYyMDAwMDIxHzAdBgNVBAMMFnZwb3N0ZXN0cy5tYWtzdXBheS5jb20wHhcN
MjQwNDMwMTE1NDA5WhcNMjgwNDI5MTE1NDA5WjBzMQswCQYDVQQGEwJFRTELMAkGA1UECAwCSE0x
EDAOBgNVBAcMB1RhbGxpbm4xEzARBgNVBAoMCk1ha3N1IHRlc3QxDzANBgNVBAsMBjIwMDAwMjEf
MB0GA1UEAwwWdnBvc3Rlc3RzLm1ha3N1cGF5LmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAL81CItSrfk12TTOviXKXy+YQXker6UkpzTDf2ekzGpzA56fbZGgNnweby/RvZ2yGlSn
KoInWl0oZ8vLHDXXMpsdnAqL8MwOvbVN0otRTZ1W36Ca65e/0l2kTDwjTtl9CHaIYxN3F9vq9dlA
Y3euGpb0O6gBzGxCr2XIxHoGadB5vEfX9bfMYgswXYWOFxsAEu+lS4uo8bVa+L9reH0cU6aRbfMw
rzdSvUn/KazECN2VIXD7QPckkV8tksrj0hB38yMa188iU9vpZ1oePS662tQRkxjAsm4LcM4/3w18
Y/ky0emEWn14p/Yza0IWY1L04SrCEH4EPWKRSzy+RQixLT0CAwEAAaNTMFEwHQYDVR0OBBYEFMYo
d11iUYfyLbLd4HnF3tJwVkVlMB8GA1UdIwQYMBaAFMYod11iUYfyLbLd4HnF3tJwVkVlMA8GA1Ud
EwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAEB5FYdHlSDdrHdAJIsZMN1IuSDthH8dE1tv
HBZ5krHwwpXLBNMiyUaFC4G/f07dXpz9lRhgL25nMOM5mCHVCfTZ/FFTMaJdQi4yZcS9hZjkZwqp
pGp0+G8tGhI/bPW8n4tFnk1HK0FN9/d2jv1qqsWaIUdpHed7uz5OXMkzDdPjb8eO/QjfnJBHN+rm
z1UPvGN0cWATBNkbLiHkWvVM94Z+gVJGv9CnJCdn6xMD5O6uhkYn51LRYTb/yAyuyKhl9mY43U3G
A+/5r2UXQ3Ec8dhNNxvhr1WkQb0Hvuw6vpTf4BRy5MH/ermH5BJpFDoDNpJYUiSl+lxvCctsfNjJ
Xwo=
</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
</ds:Signature></VPOS>
```

Response

```
<VPOS xmlns="http://www.modirum.com/schemas/vposxmlapi5"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#"><Message id="M1749126792720"
senderId="C=EE,ST=HM,L=Tallinn,O=ChaosPay,OU=E-COM Signer,CN=ChaosPay E-COM Signer 2024-
01" ts="2025-06-05T12:33:13.497Z" version="5.0">
<WalletDataRes merchantId="200002" orderId="O1749126792721" status="AUTHORIZED"
walletId="ApplePay">
<Description>Merchant session validated</Description>
<Attribute
name="sessionData">{"epochTimestamp":1749126793367,"expiresAt":1749130393367,"merchantSe
ssionIdentifier":"SSHA2BBD4939B58418AA0FF949EBBB8ED21_916523AAED1343F5BC5815E12BEE9250AF
FDC1A17C46B0DE5A943F0F94927C24","nonce":"5ea2fcaf","merchantIdentifier":"038F027A59C9891
B16BA879429FD167A4391977B97FDA9F2529C6C595C024FE3","domainName":"pay.dev.maksupay.com","
displayName":"Maksu
test","signature":"308006092a864886f70d010702a0803080020101310d300b060960864801650304020
1308006092a864886f70d0107010000a080308203e43082038ba003020102020859d8a1bcaaf4e3cd300a060
82a8648ce3d040302307a312e302c06035504030c254170706c65204170706c69636174696f6e20496e74656
7726174696f6e204341202d20473331263024060355040b0c1d4170706c652043657274696669636174696f6
e20417574686f726974793113301106035504a0c0a4170706c6520496e632e310b300906035504061302555
3301e170d323130343230313933373035a170d323630343139313933363539 5a3062312830260603550403 0
c1f6563632d736d702d62726f6b65722d7369676e5f5543342d53414e44424f583114301206035504 0b0c0b6
94f532053797374656d733113301106035504a0c0a4170706c6520496e632e310b300906035504061302555
33059301306072a8648ce3d020106082a8648ce3d03010703420004823 0fdabc39cf75e202c50d99b4512e63
7e2a901dd6cb3e0b1cd4b526798f8cf4ebde81a25a8c21e4c33ddce8e2a96c2f6afa1930345c4e87a4426ce9
51b1295a38202113082020d300c0603551d130101ff04023000301f0603551d2304183016801423f249c44f9
3e4ef27e6c4f6286c3fa2bbfd2e4b304506082b060105050701010439303703506082b06010505073001862
9687474703a2f2f6f6373702e6170706c652e636f6d2f6f63737030342d6170706c656169636133303230820
11d0603551d20048201143082011030820 10c06092a864886f7636405013081fe3081c306082b06010505070
2023081b60c81b352656c69616e6365206f6e207468697320636572746966696361746520627920616e79207
0617274792061737375 6d657320616363657074616e6365206f662074686520746856e206170706c6963616
26c65207374616e6461726420746572 6d7320616e6420636f6e646974696f6e73206f66207573652c2063657
2746966696361746520706f6c69637920616e642063657274696669636174696f6e207072616374696365207
3746174656d656e74732e303606082b060105050702011 62a687474703a2f2f7777772e6170706c652e636f6
d2f6365727469666963617465617574686f726974792f30340603551d1f042d302b3029a027a025862368747
4703a2f2f63726c2e6170706c652e636f6d2f6170706c6561696361332e63726c301d0603551d0e041604140
224300b9aeeed463197a4a65a299e4271821c45300e0603551d0f0101ff040403020780300f06092a864886f
76364061d040205003 00a06082a8648ce3d0403020347003044022074a1b324db4249430dd3274c5074c4808
d9a1f480e3a85c5c1362566325fbca3022069369053abf50b5a52f9f6004dc58aad6c50a7d608683790e0a73
ad01e4ad981308202ee30820275a0030201020208496d2fbf3a98da97300a06082a8648ce3d0403023067311
b301906035504030c124170706c6520526f6f74204341202d204733312630240603550 40b0c1d4170706c652
043657274696669636174696f6e20417574686f726974793113301106035504a0c0a4170706c6520496e632
```

```
e310b30090603550406130255533301e170d3134303530363233343633305a170d323930353036323333436333
05a307a312e302c06035504030c254170706c65204170706c69636174696f6e20496e746567726174696f6e2
04341202d20473331263024060355040b0c1d4170706c652043657274696669636174696f6e20417574686f7
269747931133011060355040a0c0a4170706c6520496e632e310b30090603550406130255533059301306072
a8648ce3d020106082a8648ce3d03010703420004f017118419d76485d51a5e25810776e880a2efde7bae4de
08dfc4b93e13356d5665b35ae22d097760d224e7bba08fd7617ce88cb76bb6670bec8e82984ff5445a381f73
081f4304606082b06010505070101043a3038303606082b06010505073001862a687474703a2f2f6f6373702
e6170706c652e636f6d2f6f63737030342d6170706c65726f6f7463616733301d0603551d0e0416041423f24
9c44f93e4ef27e6c4f6286c3fa2bbfd2e4b300f0603551d130101ff040530030101ff301f0603551d2304183
0168014bbb0dea15833889aa48a99debebdebafdacb24ab30370603551d1f0430302e302ca02aa0288626687
474703a2f2f63726c2e6170706c652e636f6d2f6170706c65726f6f74636167332e63726c300e0603551d0f0
101ff0404030201063010060a2a864886f7636406020e04020500300a06082a8648ce3d04030203670030640
2303acf7283511699b186fb35c356ca62bff417edd90f754da28ebef19c815e42b789f898f79b599f98d5410
d8f9de9c2fe0230322dd54421b0a305776c5df3383b9067fd177c2c216d964fc6726982126f54f87a7d1b99c
b9b0989216106990f09921d00003182018830820184020101308186307a312e302c06035504030c254170706
c65204170706c69636174696f6e20496e746567726174696f6e204341202d20473331263024060355040b0c1
d4170706c652043657274696669636174696f6e20417574686f7269747931133011060355040a0c0a4170706
c6520496e632e310b3009060355040613025553020859d8a1bcaaf4e3cd300b0609608648016503040201a08
193301806092a864886f70d010903310b06092a864886f70d010701301c06092a864886f70d010905310f170
d3235303630353132333331335a302806092a864886f70d010934311b3019300b0609608648016503040201a
10a06082a8648ce3d040302302f06092a864886f70d0109043122042088db61ee964194fbda8680ce4dc3433
54a8f403cb8cad6b61f0d90c1cec7ba67300a06082a8648ce3d04030204473045022010a7753fc9b2d7c6cbe
5b53c0153a58906a7c9bbb774c46c1bbcc35224b12c8d022100a8049b0615b2d9f74c4f801f09635889c1014
7b62e3b76fde5f016d300cfc4b1000000000000","operationalAnalyticsIdentifier":"Maksu
test:038F027A59C9891B16BA879429FD167A4391977B97FDA9F2529C6C595C024FE3","retries":0,"pspI
d":"44F574A73C9C7F497B5CF46DD7AA5C07D587F2DFE3285C4ECA1DC1AA0DD1F706"}</Attribute>
</WalletDataRes>
</Message><ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<ds:Reference URI="#M1749126792720">
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>P9KLc39gVdoySKWp3PkdLvJcOQAYLPfWg+ZkCFqPAyc=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
aHH0ws2rPD7UU4s6aBlCT5DEH1SwfpBAaQnVln3lZCzY6VLHOmN1BQSiy3fafDFz1PN9GZXHye+g
BBL4w3U21a9xtg85usRGQsrGmvKnoYOi3ZactqyxdIkWVbmq0FRP5QzY5Tj7kJYwF5g53PH0tjwM
eulycQQmpPJC+ELh9WAjMIXnqYDX+rFCLpJafLsYRCaHD5wYaPpne3oN669ydAMBD+mOPLonNRnY
+XO4UCbi3Gwtfn7HcoUAsesjqPDvTKXC46ty+iaO+yhDMh5srm52Zc9UTLohVdX1cF6WU47Q7wiN
HgkTJkz+RbWr599N2JWz1pOO+lSsNqzA2d3D1GVaLYkVGR52o0eErOg3VldNe14vZXlduj1mv7sE
G5iCNMkQZXHkCNgdtJc9+M6D2WcczltzUOMEHDEdC0M9bYBFBRjzV/SJqWneGG8VOA7j+23NyjDw
PweY+wtCNNbMLkQMxAm6Fnp9QY9DKOtK0j9+TFLCdsOln7c6TBJ+DxQU
</ds:SignatureValue>
<ds:KeyInfo>
<ds:X509Data>
<ds:X509Certificate>
MIIEczCCAtsCBGWfmh4wDQYJKoZIhvcNAQELBQAwfjEmMCQGA1UEAxMdQ2hhb3NQYXkgRS1DT00g
U2lnbmVyIDIwMjQtMDExFTATBgNVBAsTDEUtQ09NIFNpZ25lcjERMA8GA1UEChMIQ2hhb3NQYXkx
EDAOBgNVBAcTB1RhbGxpbm4xCzAJBgNVBAgTAkhNMQswCQYDVQQGEwJFRTAeFw0yNDAxMTEwNzM0
NTRaFw0yODAxMTAwNzM0NTRaMH4xJjAkBgNVBAMTHUNoYW9zUGF5IEUtQ09NIFNpZ25lciAyMDI0
LTAxMRUwEwYDVQQLEwxFLUNPTSBTaWduZXIxETAPBgNVBAoTCENoYW9zUGF5MRAwDgYDVQQHEwdU
YWxsaW5uMQswCQYDVQQIEwJITTELMAkGA1UEBhMCRUUwggGiMA0GCSqGSIb3DQEBAQUAA4IBjwAw
ggGKAoIBgQDI/pOZUHHlatYS/MV0i+xEMQE96tVFdRdbBesVa1P4Y1eiXITdvz5+8c2LEiBbYY6J
If4AHst0PuFlJSCYPBalEWNJTQCFOw09fHY9D3qhPoIixOM7WD46PGxbwU+Ld9y5ZfNthg+HBQ+F
OUCVcmXZ/id51FoyldValPlyjzea/5ydNhukfxIKG4zsDOa5Nn/EP2sS17MSMtTeDQbNT75asLTP
CURueVXDgAZDwTD4J3uGER4hxMAw/ybd32X0PA+MCMmNAUuLkWI1Y6SpRGELr8yFTnIYrvsFuoPM
qToLklq+E7YoJlWrDgdDXc3rGW5fu+Rk1NPRw8G6+tulkA416u2lOX5LaQkcA8ZQ3ZFHVegw7XUf
NmMycw2ofKTS5HA3nMwG4Ajr441PefTel0tKnX3qIJkBeBBTsUN+tPH7eZwz++Diru/oi435UFLg
zKdxfMc/KvjoEFw5nI1P2PZmKXnLfZbhqWxanKADzL1MRhrjCWJNo//B6puJasGDgoUCAwEAATAN
BgkqhkiG9w0BAQsFAAOCAYEAPTN5kuS/I6nsE5jBIr7zaxV7UM+ooSEdzFme6A83VvBIxJztyLMr
7kYJX6NQFbCSjIWdFs9uppKqW19VdutAhPrtLZHohdrWKHtEQOVppuarU8pNp2+MR5CAW9FmbJww
QAvh5BwJfmgIOUC+WZ5CCR5KEdPcizJwQV3j8iddysEyrdAUYetE/unZv291JtvpmTJpGu9nnb/r
cCLqbfCT/3mdxQ+idYvGCSKbEkPYfnlWm3rK9Z+BnRqsMHxmteDYlB6EHdNLzOzf/N3IKkwxrtld
Cm5VLx2XbvpP40vBs48yny++I+4aa2dhMAJNlg4YMtB1fRbJI7A9IBxWRuNQwX1wAX6mnyOF3loO
+KlCV5jezX7TjA97GTrSR0O4Zr0w3S1D4/dDctnlnOfQlE3/sFyTmzjwNoMzgBxhlhcRU6Tohb27
HJDV0yJRb27dX9SowapkUIKYhJ+zAxkWaIU3uHPAOiWp73e7p5yPCvRoYjZZDlKyGi1l2iOh+iGk
B3rI
</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
</ds:Signature></VPOS>
```


You need to extract this value of attribute named session data and **call event complete with this data AS IS.**

Then user can continue to complete Apple Pay on its device.

Once user approves Apple Pay payment

then executes flow of

```
this.paymentRequest.show().then(
                        function(paymentResponse)
                        {
                               let payload=JSON.stringify(paymentResponse);
                               // this payload needs to go as is to
                               // payment request WalletInfo.data sent to Maksu
                        }
                        ).catch((error) =>
                        {
                               console.log("Apple Pay error",error);

                        });
```

Example of payment reuest with Apple Pay data

```
<VPOS xmlns="http://www.modirum.com/schemas/vposxmlapi5"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#"><Message id="M1749127517070"
senderId="200002" ts="2025-06-05T12:45:17.070Z" version="5.0"><SaleReq
merchantId="200002"><OrderInfo
orderId="O1749126777973"><OrderDesc/><OrderAmount>1.25</OrderAmount><Currency>EUR</
Currency></OrderInfo><PaymentInfo>
<WalletInfo status="success" walletId="ApplePay">
<data>{"requestId":"f79b867c-47d1-4dac-9290-82c12427dffa","methodName":"https://
apple.com/apple-pay","details":{"billingContact":
{"familyName":"Kruus","givenName":"######","countryCode":"EE","phoneticGivenName":"","su
bAdministrativeArea":"","locality":"######","phoneticFamilyName":"","subLocality":"","ad
dressLines":["######
23"],"postalCode":"7###6","country":"Estonia","administrativeArea":""},"shippingContact"
:
{"emailAddress":"villemson@hotmail.com","phoneNumber":"+372#########","givenName":"#####
#","phoneticGivenName":"","phoneticFamilyName":"","familyName":"#########"},"token":
{"paymentData":{"data":"vffawJ+QCPOl3d5pXcQcii4qoGvaJt4MfB6lPnKTiyRYqAG+MWrnSplRkSQ9M/
LIyptW/yDyA82aAxCLF89tsBb6yquGSBi7FcvA6JY9PiPGpW1nm3Mcz0BeixaESsSX0xSCHbRSLwhyTRKto/
r0AgSHHbRzXh955zk601dK/
ZHLw9EDj+k3XriQyZZywcAUk+U2Z3L22j1TDtEsr9rWDVIFPkpeXcj8tgrtSUvjsO9F0DNCAjaneV5kebP8LIeAt
sPVqtmJTAFN66hHzgTHBw6dHRZek5io5hfioBAQeiSJO7IUTYfLGQqShjejg5x9XDNyDtArC5WMfwJnFKDC1ve/
X5wwWHQk28g18H5+/zSkpzrRZ6yfkLxeXVBmMDpcpmiTFViIMk/
lPlex","signature":"MIAGCSqGSIb3DQEHAqCAMIACAQExDTALBglghkgBZQMEAgEwgAYJKoZIhvcNAQcBAACg
gDCCA+QwggOLoAMCAQICCFnYobyq9OPNMAoGCCqGSM49BAMCMHoxLjAsBgNVBAMMJUFwcGxlIEFwcGxpY2F0aW9u
IEludGVncmF0aW9uIENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYD
VQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzAeFw0yMTA0MjAxOTM3MDBaFw0yNjA0MTkxOTM2NTlaMGIxKDAm
BgNVBAMMH2VjYy1zbXAtYnJva2VyLXNpZ25fVUM0LVNBTkRCT1gxFDASBgNVBAsMC2lPUyBTeXN0ZW1zMRMwEQYD
VQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABIIw/
avDnPdeICxQ2ZtFEuY34qkB3Wyz4LHNS1JnmPjPTr3oGiWowh5MM93OjiqWwvavoZMDRcToekQmzpUbEpWjggIRM
IICDTAMBgNVHRMBAf8EAjAAMB8GA1UdIwQYMBaAFCPyScRPk+TvJ+bE9ihsP6K7/
S5LMEUGCCsGAQUFBwEBBDkwNzA1BggrBgEFBQcwAYYpaHR0cDovL29jc3AuYXBwbGUuY29tL29jc3AwNC1hcHBsZ
WFpY2EzMDIwggEdBgNVHSAEggEUMIIBEDCCAQwGCSqGSIb3Y2QFATCB/
jCBwwYIKwYBBQUHAgIwgbYMgbNSZWxpYW5jZSBvbiB0aGlzIGNlcnRpZmljYXRlIGJ5IGFueSBwYXJ0eSBhc3N1b
WVzIGFjY2VwdGFuY2Ugb2YgdGhlIHRoZW4gYXBwbGljYWJsZSBzdGFuZGFyZCB0ZXJtcyBhbmQgY29uZGl0aW9uc
yBvZiB1c2UsIGNlcnRpZmljYXRlIHBvbGljeSBhbmQgY2VydGlmaWNhdGlvbiBwcmFjdGljZSBzdGF0ZW1lbnRzL
jA2BggrBgEFBQcCARYqaHR0cDovL3d3dy5hcHBsZS5jb20vY2VydGlmaWNhdGVhdXRob3JpdHkvMDQGA1UdHwQtM
CswKaAnoCWGI2h0dHA6Ly9jcmwuYXBwbGUuY29tL2FwcGxlYWljYTMuY3JsMB0GA1UdDgQWBBQCJDALmu7tRjGXp
KZaKZ5CcYIcRTAOBgNVHQ8BAf8EBAMCB4AwDwYJKoZIhvdjZAYdBAIFADAKBggqhkjOPQQDAgNHADBEAiB0obMk2
0JJQw3TJ0xQdMSAjZofSA46hcXBNiVmMl+8owIgaTaQU6v1C1pS+fYATcWKrWxQp9YIaDeQ4Kc60B5K2YEwggLuM
IICdaADAgECAghJbS+/
OpjalzAKBggqhkjOPQQDAjBnMRswGQYDVQQDDBJBcHBsZSBSb290IENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENl
cnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzAeFw0xNDA1MDYy
MzQ2MzBaFw0yOTA1MDYyMzQ2MzBaMHoxLjAsBgNVBAMMJUFwcGxlIEFwcGxpY2F0aW9uIEludGVncmF0aW9uIENB
IC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMu
MQswCQYDVQQGEwJVUzBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABPAXEYQZ12SF1RpeJYEHduiAou/
ee65N4I38S5PhM1bVZls1riLQl3YNIk57ugj9dhfOiMt2u2ZwvsjoKYT/
VEWjgfcwgfQwRgYIKwYBBQUHAQEEOjA4MDYGCCsGAQUFBzABhipodHRwOi8vb2NzcC5hcHBsZS5jb20vb2NzcDA0
LWFwcGxlcm9vdGNhZzMwHQYDVR0OBBYEFCPyScRPk+TvJ+bE9ihsP6K7/S5LMA8GA1UdEwEB/
wQFMAMBAf8wHwYDVR0jBBgwFoAUu7DeoVgziJqkipnevr3rr9rLJKswNwYDVR0fBDAwLjAsoCqgKIYmaHR0cDovL
2NybC5hcHBsZS5jb20vYXBwbGVyb290Y2FnMy5jcmwwDgYDVR0PAQH/
BAQDAgEGMBAGCiqGSIb3Y2QGAg4EAgUAMAoGCCqGSM49BAMCA2cAMGQCMDrPcoNRFpmxhvs1w1bKYr/
0F+3ZD3VNoo6+8ZyBXkK3ifiY95tZn5jVQQ2PnenC/gIwMi3VRCGwowV3bF3zODuQZ/
0XfCwhbZZPxnJpghJvVPh6fRuZy5sJiSFhBpkPCZIdAAAxggGIMIIBhAIBATCBhjB6MS4wLAYDVQQDDCVBcHBsZS
BBcHBsaWNhdGlvbiBJbnRlZ3JhdGlvbiBDQSAtIEczMSYwJAYDVQQLDB1BcHBsZSBDZXJ0aWZpY2F0aW9uIEF1dG
hvcml0eTETMBEGA1UECgwKQXBwbGUgSW5jLjELMAkGA1UEBhMCVVMCCFnYobyq9OPNMAsGCWCGSAFlAwQCAaCBkz
AYBgkqhkiG9w0BCQMxCwYJKoZIhvcNAQcBMBwGCSqGSIb3DQEJBTEPFw0yNTA2MDUxMjMzMThaMCgGCSqGSIb3DQ
EJNDEbMBkwCwYJYIZIAWUDBAIBoQoGCCqGSM49BAMCMC8GCSqGSIb3DQEJBDEiBCCj3udSP4zK1OMTcx7MW7rini
kOHQrVDTtArWwW5DDXXDAKBggqhkjOPQQDAgRHMEUCIBV9n3ZGMCSrDx1rm0Ookan6HEGDy1fH992FKh4LkKxTAi
EAm6joQHL3t4e56sYrcu5FRN5AYGFewYi4FPmlhPkHBDoAAAAAAAA=","header":
```

{"publicKeyHash":"B1ZvybfYo8vkekKXzJc+rOops7Xkkbr5zI0AUYvuWe0=","ephemeralPublicKey":"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEgYWFsh0BgmsFSel1PYjR49qKs11vgyZGUfEiBPhnZ3pMeB6+ezizCLfaUOZR0MybVmiiXD9lzjDIUsKskZnTdw==","transactionId":"2598328b3c02015f21fcc0a64cb3f10f924be1f62c65b8ed048454a365239995"},"version":"EC_v1"},"paymentMethod":{"displayName":"MasterCard 0049","network":"MasterCard","type":"credit"},"transactionIdentifier":"2598328b3c02015f21fcc0a64cb3f10f924be1f62c65b8ed048454a365239995"}},"shippingOption":null,"payerName":"########","payerEmail":"################","payerPhone":"+372##########"}</data>
</WalletInfo>
<PayerEmail>user@host.com</PayerEmail><PayerIpAddress>192.168.0.125</PayerIpAddress></PaymentInfo></SaleReq></Message><ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<ds:Reference URI="#M1749127517070">
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>mY3ArnmUF5e7u5gyIonkNHGSqtw+fBpQgOnuXJS8JJs=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
eqbWo8FtjgRKG220GWJmRuuw2DHplTy998vQDVfJyFuxCVr8VWZrYhno0/7BLV8Cwl/015WjxW5Y
npyVT2kMroJVDNLfaQGX00DR32oSRLVjdx0iPcv/sYgdqqr2b9ztReSA+5Tv11VRlHd0TVZ/NjnT
GOAlDD78DX2njfV6O3iliewQ/5psUo2Q86CRcu6wwecheTOLS8HjCMY5k9LOuo4KC50SK9ITDMkn
6c8P3PjLnmQVea240tsthiVPvwdB48hNecAQ2u7Vi2Xojbc+szL8UceEmL5c78U/sW7ZHaU5VVlp
AFeYBevXgo+tDHen+jFHDSlf7QTKVZEshN/gQA==
</ds:SignatureValue>
<ds:KeyInfo>
<ds:X509Data>
<ds:X509Certificate>
MIIDxzCCAq+gAwIBAgIUDBt1MUWfMQnotsdXcrZ387CY4oQwDQYJKoZIhvcNAQELBQAwczELMAkG
A1UEBhMCRUUxCzAJBgNVBAgMAkhNMRAwDgYDVQQHDAdUYWxsaW5uMRMwEQYDVQQKDApNYWtzdSB0
ZXN0MQ8wDQYDVQQLDAYyMDAwMDIxHzAdBgNVBAMMFnZwb3N0ZXN0cy5tYWtzdXBheS5jb20wHhcN
MjQwNDMwMTE1NDA5WhcNMjgwNDI5MTE1NDA5WjBzMQswCQYDVQQGEwJFRTELMAkGA1UECAwCSE0x
EDAOBgNVBAcMB1RhbGxpbm4xEzARBgNVBAoMCk1ha3N1IHRlc3QxDzANBgNVBAsMBjIwMDAwMjEf
MB0GA1UEAwwWdnBvc3Rlc3RzLm1ha3N1cGF5LmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAL81CItSrfk12TTOviXKXy+YQXker6UkpzTDf2ekzGpzA56fbZGgNnweby/RvZ2yGlSn
KoInWl0oZ8vLHDXXMpsdnAqL8MwOvbVN0otRTZ1W36Ca65e/0l2kTDwjTtl9CHaIYxN3F9vq9dlA
Y3euGpb0O6gBzGxCr2XIxHoGadB5vEfX9bfMYgswXYWOFxsAEu+lS4uo8bVa+L9reH0cU6aRbfMw
rzdSvUn/KazECN2VIXD7QPckkV8tksrj0hB38yMa188iU9vpZ1oePS662tQRkxjAsm4LcM4/3w18
Y/ky0emEWn14p/Yza0IWY1L04SrCEH4EPWKRSzy+RQixLT0CAwEAAaNTMFEwHQYDVR0OBBYEFMYo
d11iUYfyLbLd4HnF3tJwVkVlMB8GA1UdIwQYMBaAFMYod11iUYfyLbLd4HnF3tJwVkVlMA8GA1Ud
EwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAEB5FYdHlSDdrHdAJIsZMN1IuSDthH8dE1tv
HBZ5krHwwpXLBNMiyUaFC4G/f07dXpz9lRhgL25nMOM5mCHVCfTZ/FFTMaJdQi4yZcS9hZjkZwqp
pGp0+G8tGhI/bPW8n4tFnk1HK0FN9/d2jv1qqsWaIUdpHed7uz5OXMkzDdPjb8eO/QjfnJBHN+rm
z1UPvGN0cWATBNkbLiHkWvVM94Z+gVJGv9CnJCdn6xMD5O6uhkYn51LRYTb/yAyuyKhl9mY43U3G
A+/5r2UXQ3Ec8dhNNxvhr1WkQb0Hvuw6vpTf4BRy5MH/ermH5BJpFDoDNpJYUiSl+lxvCctsfNjJ
Xwo=
</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
</ds:Signature></VPOS>

3DS flow applicability:

Typically Apple Pay already contains online cryptogram and 3DS is not required. To determine if particular ApplePay interactions are already having 3ds online cryptogram or needs 3DS performed before payment, send the wallet data in , ThreeDSMethodReq AuthenticationReq, if online cryptogram is missing then authentication api returns infomration needed to continue on normal 3DS flow, if it contained 3ds online cryptogram (authenticated when added to wallet) then those requests return status AUTHWALLET and no 3DS flow needed.

3.7. XML/JSON API plugin example message and signature calculation

3.7.1 XML Signature calculation example

Example code:

```
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import javax.xml.transform.stream.StreamSource;

import org.apache.xml.security.keys.KeyInfo;
import org.apache.xml.security.keys.content.X509Data;
import org.apache.xml.security.keys.content.x509.XMLX509Certificate;
import org.apache.xml.security.signature.XMLSignature;

public class Signer
{
	public byte[] sign(VPOS root, PrivateKey prik, java.security.cert.X509Certificate[] crts)
throws Exception
	{
		org.w3c.dom.Document dom = apis.marschalToDOM(root);
		// apis.normalizeDOM(dom); dom nomralization is very slow using instead
		// msg.setIdAttribute("messageId", true);
		Element vpos = dom.getDocumentElement();
		XMLSignature xmlsigAp = new XMLSignature(dom, null,
				"http://www.w3.org/2001/04/xmldsig-more#rsa-sha256",
				"http://www.w3.org/TR/2001/REC-xml-c14n-20010315");

		Element sigel = xmlsigAp.getElement();
		vpos.appendChild(sigel);

		Element msg = (Element)vpos.getFirstChild();
		// setting id attribute instead of dom normalization
		msg.setIdAttribute("id", true);
		xmlsigAp.addDocument("#" + msg.getAttribute("messageId"), null,
			"http://www.w3.org/2001/04/xmlenc#sha256", null, null);

		for (int i = 0; crts != null && i < crts.length; i++)
		{
			xmlsigAp.addKeyInfo(crts[i]);
		}
		xmlsigAp.sign(prik);
		ByteArrayOutputStream bos = new ByteArrayOutputStream(4096);
		TransformerFactory transfac = TransformerFactory.newInstance();
		Transformer trans = transfac.newTransformer();
		trans.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
		trans.setOutputProperty(OutputKeys.INDENT, "no");
		trans.setOutputProperty(OutputKeys.ENCODING, "utf-8");

		DOMSource source = new DOMSource(dom);
		trans.transform(source, new StreamResult(bos));
		return bos.toByteArray();
	}
}
```

3.7.2. Signature calculation with JSON API V5.0

For JSON messages signature is calculated using whole JSON content as POSTED and
signature is sent in http header X-Payload-Signature
Example first part inficates signature algorithm then semicolon as delimiter and then signture value in base64 encoding:
X-Payload-Signature = **RSA-SHA256**;EOZfZG2oFjOXFX9CeFdn1hU1qu7w0dj9Dxdwk/0vlB+Hc6lPBD2l1FLZY4DT1+7vEFr9GGysitTUkmHAjO/nHNUrwNJ3mvN94YZnvbtLpqFkCjhxDapHPVRkOhTAjZ1HkwmN2SJraVgs5MUo7oHymkvfp5FS5RRZLcpV7ibPMzIj4S2YzWKf0s/LkwWNHHESsbCDqaTNbUcny+uDAidRyUdYdx5HlQX27uOIvCmDJNOyXThy14XZAuf3URaaXNVJn+M9qPb9vpQ99gkOReDNL6H5rRjOUcFPm2SMR94i00mkOWPld/fcFAMHVlHL+oFTV8o0uQLL2FIPzWf7MciQy/FLAKoRfXWtkP04tYECmfYkQjIQfDrGVhiL8dOUNWqmzbvQoNdV/d3ZlUI9r/MVOtM74l/qTiR5CAg8z8c0fT8snWMlhJdm5nKBMVZ1oDAl+BcH7CCpej+QZQRHgtFS3qvGFtMG61cmrSuShc237xuHrRqFXHAviVcSvlLAVbIR
Also heade X-Sender-ID is recuired and needs to be value of Your merchanrt account id.
In response messages from Maksu X-Sender-ID contains subject of signing serticate.

Example X-Sender-ID headers
Merchanrt sending
X-Sender-ID: 123123

Maksu sending in response or notification:
X-Sender-ID: C=AT ST=VN, L=Vienna,O=MaksuPay,OU=E-COM Signer,CN=MaksuPay E-COM Signer 2024-01

X-Sender-ID header value must be identical to Message senderId field if not message is rejected.

Example code calculating signatrure for JSON messages.:

Lets assume that you have already json content as bytes, merchant id and priovate key and cerfficate objects then:

```
public HashMap<String.String>
      signJSON(byte[] json, String mid, PrivateKey pkikKey,
            java.security.cert.X509Certificate crt)
      {
            HashMap<String.String> headers=new java.util.HahsMap<>();
            headers.put("X-Sender-ID", mid);

            byte[] pubKey=crt.getPublicKey().getEncoded()));
            MessageDigest mdigest = MessageDigest.getInstance("SHA-256");
      byte[] digestResult = mdigest.digest(toUtf8Bytes(data));
            String pubKeyHash=Base64.encode(digestResult,0);
            headers.put("X-Public-Key-Hash", pubKeyHash)

            Signature sg = Signature.getInstance("SHA256withRSA");
            sg.initSign(pkikKey);
            sg.update(json);
            byte[] sigBytes = sg.sign();
            String sigStr = Base64.encode(sigBytes, 0);
            headers.put("X-Payload-Signature", "RSA-SHA256;"+sigStr);

      return headers;
      }
```

3.8. XML API example messages

XML Examples

Please see XML message samples

SaleReq

https://testtools.maksupay.com/vpostests/examples/SaleReq.xml

and SaleRes

https://testtools.maksupay.com/vpostests/examples/SaleRes.xml

Cancel/Void

Request
https://testtools.maksupay.com/vpostests/examples/CancelReq.xml

Response
https://testtools.maksupay.com/vpostests/examples/CancelRes.xml

3.9. JSON API Example messages

JSON Examples:

Please see JSON message samples

saleReq

https://testtools.maksupay.com/vpostests/examples/saleReq.json

and saleRes

https://testtools.maksupay.com/vpostests/examples/saleRes.json

Cancel/Void

Request
https://testtools.maksupay.com/vpostests/examples/cancelReq.json

Response
https://testtools.maksupay.com/vpostests/examples/cancelRes.json

4. Maksu VPOS payment page

Payment pages in Maksu VPOS is rendered using XSLT template.
Typically merchants can select from Maksu developed templates most applicable version and if needed develop a customized CSS file for accommodated look and feel.

Custom CSS style sheets when supplied by merchant should also be reviewed by technical maintainer of the solution and then the style sheet can be installed into vposart modules once validated.

On special cases its also possible to develop merchant custom XSLT template, but this need to be on case by case basis and need to pass though validation and manual review.

4.1. XSL Templates and sub templates overview in payment XSL

Rendering starts from

```
<xsl:template match="/">
        <xsl:call-template name="page"/>
</xsl:template>
```

Then template page is rendering and calling one of the sub templates

```
<xsl:template name="page">
    <xsl:template name="paymentPage">
    <xsl:template name="errors">
    <xsl:template name="payform">
```

Template named page is rendered first and it renders general page layout and calls appropriate sub template depending on the actions what are performed.

the sub template named **"paymentPage"** is rendering the payment page part where user is selecting payment method and enters card data. This is in general **the only sub template to be modified if custom payment page is needed.**

The sub template **"errors"** is only called if system or application level unrecoverable error occurs.

the sub template **"payform"** renders invisible redirection forms that user does not see normally and they are submitted automatically by javascript.

Other sub templates are used to render specific parts of content. and are individually called by sub templates whenever applicable.