Maksu VPOS 3.0

Merchant integration

Redirection version 5, XML/JSON 5

Revision 1.1

30/ Sep/ 2024

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

Modirum 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 Modirum 3D Secure merchant plugin technology or external payment methods like net payments in shopper local banks and so on. Exact payment methods available should be specified by client.

Modirum VPOS core design enables multiple types of merchant interfaces to be implemented and also the easy to implement default interface and MPI integrated version is provided for reference.

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).

1.1. Security notes

VPOS also delivers great level of security. VPOS encrypts card PANs with 3072 BIT RSA public key into dedicated database table, so retrieval of PAN data is only possible with supplied private key (that could be in protected keystore or supplied externally from security devices). Also sensitive data like (card expiration, card holder name, payer phone and payer email) are encrypted in database, default encryption algorithm used for this is 256 bit AES.

VPOS also has a pluggable interface for risk/fraud scoring. Before completing authorization and after transaction 3D secure processing the risk score calculation service is called. The actual implementation of can take account any data (from payment method type, authentication type to user ip address) that is associated with particular transaction. The risk scoring results are also posted back to merchant shop (see the interface description section 2.2 page 7,8). Merchant profiles have option do deny and not authorize transactions starting with specified score.

1.2. Changes recorded:

1.0 26.09.2024 Intitial version of V5 interface documentation
1.1 30.09.2024 – Added 3DS API desciptions

2. Modirum VPOS interface for merchant shop

Communication between Modirum 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 Modirum VPOS

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	R	Order description text (string 1..128)
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 reommented, 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..20), exact values will depend of implemented methods on service provider side.

blockScore	O	Optional bloack 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 ..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 4: 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). Version 4 available since vpos v 1.2.3 March 30 2017
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 4 available since vpos v 1.2.3 March 2017**
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 4: 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: Since VPOS version 1.1.1 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 “Modirum VPOS”

2.3. Recurring notification POST

The following table describes the parameters of the direct POST from VPOS 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 4 available since vpos v 1.2.3 March 2017**
mid	R	Merchant id supplied (integer number)
orderid	R	Original Merchant shop order id
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 4: Message signature to ensure and verify message security and integrity. RSA with SHA2 256 digest of all the field values above concatenated together.**
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. 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).

2.5. 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.

2.6. Examples how to generate merchant keys

2.6.1 With openssl

Its just possible to do all in one line:
openssl req -x509 -newkey rsa:2048 -sha256 -keyout merchantkey.pem -out merchantcert.pem -days 1460 -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

2.6.2 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 2048 -validity 1460 -alias mykey2017 -storetype **JCEKS** -keystore mykeystore.jceks -keypass strongPassKey -keystore mycerts.jceks -storepass strongPass
Now export Your certificate to a file that can be sent to service provider:
keytool -exportcert -alias mykey2017 -file merchantcert.pem.cer -storetype JCEKS -keystore mycerts.jceks -storepass strongPass -rfc

2.7. Special data cases

Some payment methods may require order item and tax details. To send such info use 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, d-discount, sf -shipping fee, st – sales tax, d – digital , g – giftcard, sc – store credit,
s – surcharge, sc – 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. Maksu VPOS payment page

Payment page in Maksu VPOS is rendered using XSLT template.
The default template named payment.xsl and additional ones are provided with installation.
Also custom vpos artbase location can be configured where custom xsl templates can be located in separate module of VPOS but reachable by VPOS application.

Developer of custom XSLT for payment page must be familiar XSLT, XPATH, XML and other WEB technologies such as HTML and JavaScript and CSS.
In depth knowledge of XSLT can be obtained from http://www.w3.org/TR/xslt
Basic knowledge of XSLT can be found at http://www.w3schools.com/xsl/default.asp

In general merchants do not need their custom payment XSLT developed. Changes in look and feel can be more easily accomplished supplying the custom CSS style sheet that defines the look and feel of payment page such as colors, backgrounds, fonts and even content positioning.
Maksu also has selection of templates to choose from to start with.

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 folder/module of vpos artbase

Each CSS style sheet should have named as it can be later recognized to belong to certain merchant or merchants. Once the custom CSS is installed then in VPOS Manager merchant payment profile settings under section General settings field "CSS style sheet (leave blank for default) " should be filled with the CSS URI {artbase]/css/cssaname.css

where {artbase]/css part is the location of CSS files
and cssname.css is the actual file name of the supplied CSS file.

3.1. XSL Templates and sub templates 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.

4. 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 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&amp;#13;
QMQ7QRKzCD4HQIPwpaBoI4Kk6PFwwk3aD/s9+WWfeJQVpZnRqcDZOlKdeoMvrHj6PYSBqGUIKtT2&amp;#13;
/tKw6ygwcvnX5SBiVO5gZo4DReY8f+GOOBlLfqNFHtyVG+HTUATagpKzEiQBLlDuVsWksGkN5jk+&amp;#13;
oFwoYlev8qB2f4niQEKFcT/KN16WszVlkVnRVcJvl8u7IGKhTv7wfzqnb/AXbf/Qq7n1uAioLbLF&amp;#13;
RE25mHVt6z1138xkX6rlBbeQ4D/HU8SZSiXaDg==
              </ds:SignatureValue>
              <ds:KeyInfo>
                     <ds:X509Data>
                            <ds:X509Certificate>
MIIDxzCCAq+gAwIBAgIUDBt1MUWfMQnotsdXcrZ387CY4oQwDQYJKoZIhvcNAQELBQAwczELMAkG&amp;#13;
A1UEBhMCRUUxCzAJBgNVBAgMAkhNMRAwDgYDVQQHDAdUYWxsaW5uMRMwEQYDVQQKDApNYWtzdSB0&amp;#13;
```

```
ZXN0MQ8wDQYDVQQLDAYyMDAwMDIxHzAdBgNVBAMMFnZwb3N0ZXN0cy5tYWtzdXBheS5jb20wHhcN&#13;
MjQwNDMwMTE1NDA5WhcNMjgwNDI5MTE1NDA5WjBzMQswCQYDVQQGEwJFRTELMAkGA1UECAwCSE0x&#13;
EDAOBgNVBAcMB1RhbGxpbm4xEzARBgNVBAoMCk1ha3N1IHRlc3QxDzANBgNVBAsMBjIwMDAwMjEf&#13;
MB0GA1UEAwwWdnBvc3Rlc3RzLm1ha3N1cGF5LmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC&#13;
AQoCggEBAL81CItSrfk12TTOviXKXy+YQXker6UkpzTDf2ekzGpzA56fbZGgNnweby/RvZ2yGlSn&#13;
KoInWl0oZ8vLHDXXMpsdnAqL8MwOvbVN0otRTZ1W36Ca65e/0l2kTDwjTtl9CHaIYxN3F9vq9dlA&#13;
Y3euGpb0O6gBzGxCr2XIxHoGadB5vEfX9bfMYgswXYWOFxsAEu+lS4uo8bVa+L9reH0cU6aRbfMw&#13;
rzdSvUn/KazECN2VIXD7QPckkV8tksrj0hB38yMa188iU9vpZ1oePS662tQRkxjAsm4LcM4/3w18&#13;
Y/ky0emEWn14p/Yza0IWY1L04SrCEH4EPWKRSzy+RQixLT0CAwEAAaNTMFEwHQYDVR0OBBYEFMYo&#13;
d11iUYfyLbLd4HnF3tJwVkVlMB8GA1UdIwQYMBaAFMYod11iUYfyLbLd4HnF3tJwVkVlMA8GA1Ud&#13;
EwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAEB5FYdHlSDdrHdAJIsZMN1IuSDthH8dE1tv&#13;
HBZ5krHwwpXLBNMiyUaFC4G/f07dXpz9lRhgL25nMOM5mCHVCfTZ/FFTMaJdQi4yZcS9hZjkZwqp&#13;
pGp0+G8tGhI/bPW8n4tFnk1HK0FN9/d2jv1qqsWaIUdpHed7uz5OXMkzDdPjb8eO/QjfnJBHN+rm&#13;
z1UPvGN0cWATBNkbLiHkWvVM94Z+gVJGv9CnJCdn6xMD5O6uhkYn51LRYTb/yAyuyKhl9mY43U3G&#13;
A+/5r2UXQ3Ec8dhNNxvhr1WkQb0Hvuw6vpTf4BRy5MH/ermH5BJpFDoDNpJYUiSl+lxvCctsfNjJ&#13;
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&#13;
qrytSqdNCr3CwYdhlGB2RyZrUE/zhSHUxWXFqrpFVH+Ikb4EYJ2ytakDrG7XzMdyJu9YKMsnakHB&#13;
W6xXyRJkC/P3b90R6myaQJRFJemGXGu75wJ/R+qmF6S5V/eRS2l18YuNX/3qMtINWQNALPfftxLQ&#13;
0ZKKNwT44XIU1GspK8jevPXmV52JkvSE+3uGu3a5t7RD8/7jgOgHa4yNeT7Zbe5OrCLlZsoW6Ypg&#13;
yyVmipG8UEH+h9lhSWIDn/+oeRO6BckuDKZL8FWxuFrA/GUdfWqbnznNrrA30jQEFhRrxVgGhN6g&#13;
PFnEgXPwUSvQLuYcNXryTR3wT3Ndi+Wsc3hvlfl4Ah6HbZMmu5S9m/+5AIqb6lDxsB4AGxDBi7UG&#13;
3jVswAq1uK0zGZPiPPtQGJ/J8wQ1tmJCx1Bb9ssESEQneYfRC1txM01f
            </ds:SignatureValue>
            <ds:KeyInfo>
                  <ds:X509Data>
                        <ds:X509Certificate>
MIIEczCCAtsCBGWfmh4wDQYJKoZIhvcNAQELBQAwfjEmMCQGA1UEAxMdQ2hhb3NQYXkgRS1DT00g&#13;
U2lnbmVyIDIwMjQtMDExFTATBgNVBAsTDEUtQ09NIFNpZ25lcjERMA8GA1UEChMIQ2hhb3NQYXkx&#13;
EDAOBgNVBAcTB1RhbGxpbm4xCzAJBgNVBAgTAkhNMQswCQYDVQQGEwJFRTAeFw0yNDAxMTEwNzM0&#13;
NTRaFw0yODAxMTAwNzM0NTRaMH4xJjAkBgNVBAMTHUNoYW9zUGF5IEUtQ09NIFNpZ25lciAyMDI0&#13;
LTAxMRUwEwYDVQQLEwxFLUNPTSBTaWduZXIxETAPBgNVBAoTCENoYW9zUGF5MRAwDgYDVQQHEwdU&#13;
YWxsaW5uMQswCQYDVQQIEwJITTELMAkGA1UEBhMCRUUwggGiMA0GCSqGSIb3DQEBAQUAA4IBjwAw&#13;
```

```
ggGKAoIBgQDI/pOZUHHlatYS/MV0i+xEMQE96tVFdRdbBesVa1P4Y1eiXITdvz5+8c2LEiBbYY6J&#13;
If4AHst0PuFlJSCYPBalEWNJTQCFOw09fHY9D3qhPoIixOM7WD46PGxbwU+Ld9y5ZfNthg+HBQ+F&#13;
OUCVcmXZ/id51FoyldValPlyjzea/5ydNhukfxIKG4zsDOa5Nn/EP2sS17MSMtTeDQbNT75asLTP&#13;
CURueVXDgAZDwTD4J3uGER4hxMAw/ybd32X0PA+MCMmNAUuLkWI1Y6SpRGELr8yFTnIYrvsFuoPM&#13;
qToLklq+E7YoJlWrDgdDXc3rGW5fu+Rk1NPRw8G6+tulkA416u2lOX5LaQkcA8ZQ3ZFHVegw7XUf&#13;
NmMycw2ofKTS5HA3nMwG4Ajr441PefTel0tKnX3qIJkBeBBTsUN+tPH7eZwz++Diru/oi435UFLg&#13;
zKdxfMc/KvjoEFw5nI1P2PZmKXnLfZbhqWxanKADzL1MRhrjCWJNo//B6puJasGDgoUCAwEAATAN&#13;
BgkqhkiG9w0BAQsFAAOCAYEAPTN5kuS/I6nsE5jBIr7zaxV7UM+ooSEdzFme6A83VvBIxJztyLMr&#13;
7kYJX6NQFbCSjIWdFs9uppKqW19VdutAhPrtLZHohdrWKHtEQOVppuarU8pNp2+MR5CAW9FmbJww&#13;
QAvh5BwJfmgIOUC+WZ5CCR5KEdPcizJwQV3j8iddysEyrdAUYetE/unZv291JtvpmTJpGu9nnb/r&#13;
cCLqbfCT/3mdxQ+idYvGCSKbEkPYfnlWm3rK9Z+BnRqsMHxmteDYlB6EHdNLzOzf/N3IKkwxrtld&#13;
Cm5VLx2XbvpP40vBs48yny++I+4aa2dhMAJNlg4YMtB1fRbJI7A9IBxWRuNQwX1wAX6mnyOF3loO&#13;
+KlCV5jezX7TjA97GTrSR0O4Zr0w3S1D4/dDctnlnOfQlE3/sFyTmzjwNoMzgBxhlhcRU6Tohb27&#13;
HJDV0yJRb27dX9SowapkUIKYhJ+zAxkWaIU3uHPAOiWp73e7p5yPCvRoYjZZDlKyGi1l2iOh+iGk&#13;
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.

4.1 Payments API

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" Required or 2.1
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	element	RequestMessage element depending on request type Equvalent JSON objects: saleReq,authorisationReq, originalCreditReq, refundReq, cancelReq recurringOperationReq, statusReq, tokenizationReq, deTokenizationReq
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..128	Order description defined by Merchant / JSON string orderDesc
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..1024	Free variable defined by merchant. /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
C>VV2	xsi:string N3..4	CVV2/CVC2 security code from card./JSON string cardCvv2
EncData	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
RecurringParameters	element	Recurring parameters element /JSON object **recurringParameters**
ExtRecurringfrequency	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 extRecurringferequency
ExtRecurringenddate	xsi:string N8	Recurring end date Format yyyymmdd /JSON string extRecurringenddate
InstallmentParameters	element	Installments parameters element /JSON object **installmentParameters**
ExtInstallmentoffset	xsi:integer N1..2	Defines the number of months between the entering of the transaction, n case installment payment /JSON number extInstallmentoffset
ExtInstallmentperiod	xsi:integer N1..2	Defines the number of monthly payments in case installment payment. Valid value must be >1 /JSON number extInstallmentperiod

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

WalletInfo	element	A wallets extension element if merchant inititated the xml api payment with Wallets.
Attribute	element, attr eg name="status"	Attribute names and values depend on exact wallet case and will be communicated separately
Responses/ Notification		
VPOS	element	XML root element
Message	element type Message	Message contents element
version	attribute, xsi:string	Message version default value "1.0" 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	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 requrring operation response if supsequent requrrings 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	
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>
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 CardEncData 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
Or EncData	C																Used if RSA card encryption then Pan, ExpDate, HolderName and CVV2 shal not be present but are in EncData encrypted
PayerName* PayerPhone	O																PayerName applicable for non card payments* else in card data Optional but highly recommended. Not present if CardEncData present. Payer email requirer
PayerEmail	R																
PayerIpAddress	R																Payer ip address
ExtXOrderI ExtTokenOptionsd	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
ExtRecurringfrequency	C																Required for recurring payment
ExtRecurringenddate	C																Required for recurring payment
InstallmentParameters	C																Required for installment payment
ExtInstallmentoffset	C																Required for installment payment
ExtInstallmentperiod	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	R	Required for all
WalletInfo	C																Depends on wallet case
Attribute el name="status"	C																Depends on wallet case
Attribute el	C																Depends on wallet case

name="txId"																	
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	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:

AuthorisationReqt -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

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)
XE – Invalid XML 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

4.2 3D Secure API

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

Messages structure is indentical to Payements 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 AuthResultsReqAuthResultsRes	element	RequestMessage element depending on request type Equvalent JSON objects: saleReq,authorisationReq, originalCreditReq, refundReq, cancelReq recurringOperationReq, statusReq, tokenizationReq, deTokenizationReq tdsmethodReq authReq authResultsReqReq
TDSMethodReq	El, Message object	Message to check if Three DS method applies for card range
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
orderId	attribute	Required
merchantId	attribute	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
WalletInfo	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	El xsi: string	3DS Server trans id
dsTransID	El xsi: string	Directory server trans id
acsTransID	El xsi: string	ACS trans id
Attribute	El attribute typ	0 to many
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
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 requred, the content is to be rendered in browser iframe in raletion to main payment page.
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

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	El, xsi string	Peresent on authentication success or proof of attempt cases
tdsTransID	El xsi: string	3DS Server trans id
dsTransID	El xsi: string	Directory server trans id
acsTransID	El xsi: string	ACS trans id
Attribute	El attribute typ	0 to many
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

4.3. 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.
Messages sent by VPOS service are signed by service provider private key and validated with service provider provided certificate.

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 API with Three D Secure:
This is 2 step processing at first step merchant should implement MPI plugin session as decribed in Modirum MPI manual and obtain the Three D Secure authentication results from there and then next step is to fill the corresponding values to XML API ThreeDSecure element and proceed with XML api request to VPOS.

5. Browser client side card data encryption for XML/JSON api using PSP javascript

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:

```
<script type="text/javascript"
src="https://serviceproviderhost/vpos/csescript.js?version=4&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.

5.1 Browser client side card data encryption for XML api 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://serviceproviderhost/vpos/js/vposcseclient.js">      </script>
<iframe id="vposcseiframe" style="display: block;" width="480" height="110"
src="https://serviceproviderhost/vpos/cseiframe.html?
version=4&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");  }
}
```

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

6.1 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();
	}
}
```

6.2. 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/MVOtM74I/
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;
      }
```

7. XML/JSON API example messages

7.1. 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

7.2. 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