Card processing - common introduction

Introduction

Acquirers and connection

Computop Paygate supports many different credit card connections to various acquirers / processors with different protocols.

You can find an overview of all different credit card interfaces here: Payments by Credit Card.

Additional features (e.g. AVS (Address Verification Service), refund, 3-D Secure, ...) may depend on the specific integration and acquirer.

Integration with Computop Paygate

In general we offer two different ways of integration:

	Payment page (payssl.aspx)	**Direct integration (direct.aspx)**
Credit card number (PAN) handling	• Directly handled by payment page. • Credit card number, expiry date, CVV, ... are requested by the payment form • You will not get in contact with PAN, so much easier PCI DSS compliance. • You will receive optional a PseudoCardNumber (PcNr) as a Computop Paygate internal token to represent the PAN.	• Your system handles PAN directly, therefore you have "full control". • As your system gets in contact with the credit card number (PAN) your system will be in fully PCI DSS focus.
3-D Secure handling	• You only need to add KVP "MsgVer=2.0" to indicate that your system is ready for 3-D Secure 2.x • The rest (redirect to issuer bank for consumer authentication) is handled by the Paygate payment page.	• You only need to add KVP "MsgVer=2.0" to indicate that your system is ready for 3-D Secure 2.x • Your system has to consumer redirect to issuer bank in case of consumer authentication
Additional data	• Additional data can be provided via additional JSON parameters, e.g.: ○ "credentialOnFile" (for recurring payments) ○ address data (for AVS) ○ 3-D Secure policy data	
Shop-/System integration	• The payment page can be customized (logos, colors, positions, ...) to match your corporate identify using templates which can be prepared by you. • The consumer is ***redirected*** to the payment page to input credit card details (PAN, expiry date, CVV, ...). • Your shop is informed via Paygate notify for result of payment process.	• Your system has full control of the input fields for credit card details • The consumer is not redirected and your system gets the result of API call via direct response values
Further actions	• After initiating the payment process you may start further actions like capture or credit/refund, cancellations, ... • These actions refer to a previous payment process identified by a PayId - which is fully out of PCI DSS focus.	
Conclusion	Recommended for standard integrations - due to easy integration and simplified compliance. • Computop Paygate takes PAN handling for you ***simplified PCI DSS handling***. • You can customize Paygate payment page using templates.	Recommended if you need full control and you do not want a redirect of the consumer. • Your system will be in full PCI DSS scope.

The documentation below is therefore always devided into two sections:

- integration via payment page (payment form)
  - with common parameters to integrate Computop Paygate payment form
  - with parameters to customize the payment form
  - with specific parameters for the desired acquirer / processor
- integration via Server-2-Server (direct) integration
  - with common parameters to integrate Computop Paygate payment form

- with specific parameters for the desired acquirer / processor

Implementation of 3-D Secure (2.x)

Common notes to 3-D Secure

3-D Secure is a process that authenticates the card holder to ensure that the consumer using the credit card data really is the card holder.

3-D Secure shall provide abuse of credit card data - specially in ecommerce environment.

3-D Secure 1.x has been implemented and asks the card holder typically for a password with each card usage.

3-D Secure 2.x has been implemented to:

- enable strong customer authentication (SCA) by authenticate the card holder with 2 independent factors of these 3 factors:
  - something the card holder ***knows***, e.g. a password
  - something the card holder ***owns***, e.g. a device (like phone to receive a token via SMS or using other OTP, token generator, ...)
  - something the card holder ***is***, e.g. biometrics (like finger print, face-id, ...)
- enable seemless authentication where the consumer is not authenticated and not asked to authenticate himself.

3-D Secure with Computop Paygate

Prepare yourself / your integration to be 3-D Secure 2.x ready - here a short overview with some technical details.

<table>
<tr><th></th><th>3-D Secure 1.x</th><th>3-D Secure 2.x</th><th>3-D Secure 2.x Sample</th></tr>
<tr><td colspan="4">Depend on your integration: Payment Form ./. Server-2-Server</td></tr>
<tr><td>Payment Page / Payment Form</td><td>Your existing integration.</td><td>Just add API parameter "MsgVer=2.0", the rest is handled automatically by Computop Paygate</td><td>Add parameter "MsgVer=2.0" to your existing API call to start Payment Form.</td></tr>
<tr><td>URL-processing</td><td>URLFailure and URLSuccess work with http-GET</td><td>URLFailure and URLSuccess work with http-POST (due to amount of data). So pls. prepare to handle both (GET + POST)</td><td></td></tr>
<tr><td>Server-2-Server integration</td><td>Use KVP:<table><tr><td>CCNr</td><td>Credit card number (PAN)</td></tr><tr><td>CCExpiry</td><td>Expiry date of the credit card</td></tr><tr><td>CCCVC</td><td>Card verification number</td></tr><tr><td>CCBrand</td><td>Credit card brand.</td></tr></table></td><td>Use "card"-JSON to provide card data to API</td><td>e.g.:<pre>{
    "securityCode": "569",
    "expiryDate": "202508",
    "cardholderName": "William Thomas",
    "number": "4111111111111111",
    "brand": "VISA"
}</pre>card=ewogICAgInNlY3VyaXR5Q29kZSI6ICI1NjkiLAogICAgImV4cGlyeURhdGUiOiAiMjAyNTA4IiwKICAgICJjYXJkaG9sZGVyTmFtZSI6ICJXaWxsaWFtIFRob21hcyIsCiAgICAibnVtYmVyIjogIjQxMTExMTExMTExMTExMTEiLAogICAgImJyYW5kIjogIlZJU0EiCn0=</td></tr>
<tr><td colspan="4">For specific use cases, find other use cases here: 3DS 2.0 Merchant Use-Cases</td></tr>
<tr><td>Use case</td><td>3-D Secure 1.x</td><td>3-D Secure 2.x</td><td>3-D Secure 2.x Sample</td></tr>
</table>

<table>
<tr>
<td>Recurring payments (initial)</td>
<td>Use parameter "RTF=I"

you may receive TransactionID as Card scheme specific transaction ID</td>
<td>Change "RTF" to parameter "credentialOnFile"-JSON with "recurring" and "initial=true"

you may receive schemeReferenceID as Card scheme specific transaction ID</td>
<td>e.g.:

Recurring with initial=true

<pre>{
    "type": {
        "recurring": {
            "recurringFrequency": 30,
            "recurringStartDate": "2021-09-14",
            "recurringExpiryDate": "2022-09-14"
        }
    },
    "initialPayment": true
}</pre>
The JSON needs to be base64-encoded and then used as value for parameter credentialOnFile (pls. be aware to use the full base64-encoded string, including "=" at the end)

credentialOnFile=ewogICAgInR5cGUiOiB7CiAgICAgICAgInJlY3VycmluZyI6IHsKICAgICAgICAgICAgInJlY3VycmluZ0ZyZXF1ZW5jeSI6IDMwLAogICAgICAgICAgICAicmVjdXJyaW5nU3RhcnREREYXRlIjogIjIwMjEtMDktMTQiLAogICAgICAgICAgICAicmVjdXJyaW5nRXhwaXJ5RGF0ZSI6ICIyMDIyLTA5LTE0Igogl CAgICAgIH0KICAgIH0sCiAgICAiaW5pdGlhbFBheW1lbnQiOiB0cnVlCn0=</td>
</tr>
<tr>
<td>Recurring payments (subsequent)</td>
<td>Use parameter "RTF=R"

and
send Transaction ID as Card scheme specific transaction ID</td>
<td>Change "RTF" to parameter "credentialOnFile"-JSON with "recurring" and "initial=false"

and send schemeReferenceID as Card scheme specific transaction ID</td>
<td>e.g.

Recurring with initial=false

<pre>{
    "type": {
        "recurring": {
            "recurringFrequency": 30,
            "recurringStartDate": "2021-09-14",
            "recurringExpiryDate": "2022-09-14"
        }
    },
    "initialPayment": false
}</pre>
After base64-encoding:

credentialOnFile=ewogICAgInR5cGUiOiB7CiAgICAgICAgInJlY3VycmluZyI6IHsKICAgICAgICAgICAgInJlY3VycmluZ0ZyZXF1ZW5jeSI6IDMwLAogICAgICAgICAgICAicmVjdXJyaW5nU3RhcnREREYXRlIjogIjIwMjEtMDktMTQiLAogICAgICAgICAgICAicmVjdXJyaW5nRXhwaXJ5RGF0ZSI6ICIyMDIyLTA5LTE0Igogl CAgICAgIH0KICAgIH0sCiAgICAiaW5pdGlhbFBheW1lbnQiOiBmYWxzZQp9</td>
</tr>
<tr>
<td>Customer Initiated (initial)</td>
<td>Use parameter "RTF=E"

you may receive TransactionID as Card scheme specific transaction ID</td>
<td>Change "RTF" to parameter "credentialOnFile"-JSON with "CIT" and "initial=true"

you may receive schemeReferenceID as Card scheme specific transaction ID</td>
<td>e.g.

CIT (CustomerInitiated) with initial=true

<pre>{
    "type": {
        "unscheduled": "CIT"
    },
    "initialPayment": true
}</pre>
After base64-encoding (again: don't miss "=" at the end; it has to be part of the value):

credentialOnFile=ewogICAgInR5cGUiOiB7CiAgICAgICAgInVuc2NoZWR1bGVkIjogIkNJVCIKICAgIH0sCiAgICAiaW5pdGlhbFBheW1lbnQiOiB0cnVlCn0=</td>
</tr>
</table>

Merchant Initiated (subsequent)	Use parameter "RTF=M" and send Transaction ID as Card scheme specific transaction ID	Change "RTF" to parameter "credentialOnFile"-JSON with "MIT" and "initial=false" and send schemeReferenceID as Card scheme specific transaction ID	e.g. **MIT (MerchantInitiated) with initial=false** `{ "type": { "unscheduled": "MIT" }, "initialPayment": false }` After base64-encoding: credentialOnFile=ewogICAgInR5cGUiOiB7CiAgICAgICAgInVuc2NoZWR1bGVkIjogIk1JVCIKICAgIH0sCiAgICAiaW5pdGlhbFBheW1lbnQiOiBmYWxzZQp9
Address Verification Service (AVS) (depending on acquirer / processor)	Use parameter • AddrStreet • AddrStreet Nr • AddrZip • AddrCity •	Change address data to "address"-JSON	e.g. **Put address data into JSON structure** `{ "city": "New York", "country": { "countryA3": "USA" }, "addressLine1": { "street": "Park Avenue", "streetNumber": "270" }, "postalCode": "10017-2070", "state": "NY" }` billingAddress=ewogICAgImNpdHkiOiAiTmV3IFlvcmsiLAogICAgImNvdW50cnkiOiB7CiAgICAgICAgImNvdW50cnlBMyI6ICJVU0EiCiAgICB9LAogICAgImFkZHJlc3NMaW5lMSI6IHsKICAgICAgICAic3RyZWV0IjogIlBhcmsgQXZlbnVlIiwKICAgICAgICAic3RyZWV0TnVtYmVyIjogIjI3MCIKICAgIH0sCiAgICAicG9zdGFsQ29kZSI6ICIxMDAxNy0yMDcwIiwKICAgICJzdGF0ZSI6ICJOWSIKfQ==
Apply for frictionless payment processing	• not supported with 3-D Secure 1.x • each payment will be authenticated	Provide additional data as JSON-KVP: JSON Objects	e.g.: **Explicitly apply for customer challenge** `{ "challengePreference ": "mandateChallenge" }` After base64-encoding (again: don't miss "=" at the end; it has to be part of the value): threeDSPolicy=ewogICAgImNoYWxsZW5nZVByZWZlcmVuY2UgIjogIm1hbmRhdGVDaGFsbGVuZ2UiCn0=