# Nets Estonia E-Com Documentation

Version 1.1.7

Version date: 2019-05-14

# 1. Introduction

## 1.1. General procedure steps

### 1.1.1. Contract with bank

Merchant has signed an e-commerce contract with the acquirer bank

### 1.1.2. Test account

You will receive an e-mail with the test account details. Nets Estonia creates a test account after receiving information from merchants acquirer bank and sends the account details to the contact person mentioned in the contract.

### 1.1.3. e-com implementation

The iPay (Nets Estonia e-commerce solution) test environment is a copy of the live environment. Generally you need to implement a payment request with two data flows to iPay server:

- Payment request to the iPay server. (Initiator your website, receiver "iPay server").
- Payment verification, this is handled by your web service. (Initiator "iPay servlet", receiver your website)

After the payment is implemented on the merchant side next steps are taken:

### 1.1.4. Merchant testing

Merchant/developer ensures that the required functionality is obtained and the system reacts adequately to different payment scenarios.

### 1.1.5. Nets Estonia testing

Merchant/developer provides instructions by sending an e-mail to webpos@estcard.ee, on how Nets Estonia AS can perform test payments on the ready solution without causing negative impact to the merchant.

### 1.1.6. Bank confirmation

After the tests are successful, Nets Estonia submits a Live environment access request to the bank.

### 1.1.7. Live environment

After confirmation from the bank, the Live account is activated and its details are sent to the contact person mentioned in the contract.

## 1.2. Technical overview

Nets Estonia e-commerce gateway (iPay) works with http queries. To request an authorization form, customer is redirected from merchant's e-shop to iPay with simple hidden html form.

iPay handles actual authorization and card data, ensuring that merchant never has to worry about securing customers card data, it is taken out of merchants domain and never revealed to merchant, making the whole process more secure.

After customer has completed payment, customer is redirected back to the merchants web store with information about the success of transaction.

For merchant iPay also provides online environment to view and search past transactions, making life simpler for merchant.
Regular reports over e-mails are also available.

# 2. Protocol specifications

## 2.1. Protocol version 004

Protocol version 004 is currently our default protocol.
Without extensions it provides simple authorization services.

### 2.1.1. Request addresses

Test: https://test.estcard.ee/ecom/iPayServlet

Production address will be provided after testing has been completed successfully.

### 2.1.2. Requesting authorization form

To request authorization page from Nets Estonia e-commerce gateway a simple hidden html form needs to be generated within a merchants webpage.

*Table 1. Fields for authorization form request*

| Nr | Availability (Mandatory/ Optional) | MAC Value | Request Parameter | Definition | Format | Example value |
|---|---|---|---|---|---|---|
| 1 | M | N | lang | Language: ISO 639-1 (1) | char..2 | en |
| 2 | M | N | action | Action ID: gaf | char..3 | gaf |
| 3 | M | Y | ver | Protocol version: 4 | int..3 | 004 |
| 4 | M | Y | id | Merchant ID | char..10 | 12ABCD1223 |
| 5 | M | Y | ecuno | Transactions unique ID. Format: date[YYYYMM] + random number(in range): 100000-999999 (2) | int..12 | 201610280012 |
| 6 | M | Y | eamount | Payment amount in cents | int..12 | 000000001234 |
| 7 | M | Y | cur | Currency ISO-4217 | char..3 | EUR |
| 8 | M | Y | datetime | Timestamp. Format: [YYYYMMDDhhmmss] ISO-8601 | int..14 | 20161028112930 |
| 9 | M | N | charEncoding | Encoding | char | UTF-8 |
| 10 | M | Y | feedBackUrl | feedback URL given by merchant | char..128 | https://merchant.site/feedback |

| Nr | Availability (Mandatory/ Optional) | MAC Value | Request Parameter | Definition | Format | Example value |
|---|---|---|---|---|---|---|
| 11 | M | Y | delivery | 1. position: Delivery description (Mandatory), **2. position: Pre-Authorization "P" (Optional. Usable only for pre-auhorization)** (S\|T)(P) | char..1 or 2 | First position S or T, second position P<br>S (electronic delivery of product),<br>T (physical delivery of product),<br>P (preorder) |
| 12 | O | Y | additionalinfo | Additional information (3) | char..128 key:value; | refnr:123;100:ABC123;101:kala;001:jama; |
| 13 | M | - | mac | Digital Signature | hexadecimal | 4d5e875a245d42….. |

*lang*

The language of the web terminal (where the customer enters card data)
Currently *en,et,ru,lv,lt,fi,de* are supported, if necessary others can be implemented.

*ecuno*

Unique identifier that is in both request and response and which connects authorization request with authorization response. This needs to be unique within 24 hours. When transaction is declined or cancelled and customer wishes to try the same payment again this value needs to be renewed.

*additionalinfo*

Information that is displayed in merchant's report view. Also searchable. Good place to store some relevant data about the transaction. For example ticket number or client id or booking id. Can be up to 128 characters in total length. Key is separated from value using ":". *key:value* pairs are separated with ";"
If key-value pair *refnr:{number}* is present, it will also get included into transaction and settlement data to your bank. If your bank can process this, it should also show in your settlement report.
*additionalinfo* field is optional, but if it is present it has to be included in MAC calculation.

Posted data should be **unpadded.**
In MAC calculation all fields **must** be padded to their maximum length. Ascii (char) fields are padded **with trailing spaces** and numeric fields are padded **with leading zeroes**.

### Example data

```
action=gaf
lang=en
ver=004
id=12ABCD1223
ecuno=201610280012
eamount=1234
cur=EUR
datetime=20161028112930
charEncoding=UTF-8
feedBackUrl=https://merchant.site/feedback
delivery=S
additionalinfo=refnr:123;100:ABC123;101:kala;001:jama;
(mac=9f7eab41d650.....)
```

For testing do not use the example data, generate your own, and use the test service ID provided in the e-mail.

### Example MAC calculation

For MAC calculation data fields will be padded to their max length and concatenated to create a single string. Order of the fields is important.

Sign the padded data (SHA1withRSA).

SHA1 will be deprecated in favour of sha256, however, currently SHA1 is used. New implementations should be capable of using sha256 if the need arises.]

Convert the signature to hex, this will be your MAC.

**signature**=RSA(prikey,SHA1(version+merchant_id+ecuno+amount+currency+datetime+feedbackurl+delivery+additional_info))

**mac**=bin2hex(signature)

### Example MAC value

For fast checking of validity of your MAC calculation, you can use example MAC value, calculated with mytestprivat.key.
Using input's from example below, MAC should be identical to the sample MAC value below.

```
# Original, unpadded data

ver : 004
id : 12ABCD1223
ecuno : 201610280012
eamount : 1234
currency : EUR
datetime : 20161028112930
feedBackUrl: https://merchant.site/feedback
delivery : S
additionalinfo : refnr:123;100:ABC123;101:kala;001:jama;

# Padded MAC string

[00412ABCD12232016102800120000000001234EUR20161028112930https://merchant.site/feedback
Srefnr:123;100:ABC123;101:kala;001:jama;
]

# Calculated MAC

16b6c4ae71cd5a8909c0cc57c9b2906ce816ebe0b22d5752149958079d5a677d9fb5259d13fdb246518290
89dd0317e0bd8902a5725243cb4f47e9b98fb4fbc4bfb47318e1bee86f2bcbabff623d9b788017c4d1195c
4740d0f9336005f76a5ac6d39d16a918c834c38510d2f520d69a1ad01f2b4527e86dfe8cd3df0d8ac068
```

### Example HTML form

All of the above put together, for requesting authorization page would look like this:

```
<form  name='form' action="https://test.estcard.ee/ecom/iPayServlet" method="post">
    <input type="submit" value="To payment page">
    <input type="hidden" name="lang" value="en">
    <input type="hidden" name="action" value="gaf">
    <input type="hidden" name="ver" value="004">
    <input type="hidden" name="id" value="12ABCD1223">
    <input type="hidden" name="ecuno" value="201610280012">
    <input type="hidden" name="eamount" value="1234">
    <input type="hidden" name="cur" value="EUR">
    <input type="hidden" name="datetime" value="20161028112930">
    <input type="hidden" name="charEncoding" value="UTF-8">
    <input type="hidden" name="feedBackUrl" value="https://merchant.site/feedback">
    <input type="hidden" name="delivery" value="S">
    <input type="hidden" name="additionalinfo" value=
"refnr:123;100:ABC123;101:kala;001:jama;">
    <input type="hidden" name="mac" value=
"16b6c4ae71cd5a8909c0cc57c9b2906ce816ebe0b22d5752149958079d5a677d9fb5259d13fdb24651829
089dd0317e0bd8902a5725243cb4f47e9b98fb4fbc4bfb47318e1bee86f2bcbabff623d9b788017c4d1195
c4740d0f9336005f76a5ac6d39d16a918c834c38510d2f520d69a1ad01f2b4527e86dfe8cd3df0d8ac068"
>
</form>
```

### 2.1.3. Response

Response is calculated the same way as request, only it is done by iPay and Merchant's webpage has to check validity of the answer.

Nets test public key to check the response ecomtestpublic.key

RSA with SHA1 (SHA1withRSA).

SHA1 will be deprecated in favour of sha256, however, currently SHA1 is used. New implementations should be capable of using sha256 if the need arises.]

**MAC**=RSA(prikey, SHA1(ver+id+ecuno+receipt_no+eamount+cur+respcode+datetime+msgdata+actiontext))

*Table 2. Response request*

| Nr | Availability (Mandatory/ Optional) | MAC Value | Request Parameter | Definition | Format | Example value |
|---|---|---|---|---|---|---|
| 1 | M | Y | action | Action ID: afb | Char..3 | afb |
| 2 | M | Y | ver | Protocol version: 4 | int..3 | 004 |
| 3 | M | Y | id | Merchant ID | char..10 | 12ABCD1223 |
| 4 | M | Y | ecuno | Transactions unique ID. Format: date[YYYYMM] + random number(in range): 100000-999999 | int..12 | 201610280012 |
| 5 | M | Y | receipt_no | Receipt number | int..6 | 000015 |
| 6 | M | Y | eamount | Payment amount in cents | int..12 | 000000001234 |
| 7 | M | Y | cur | Currency ISO-4217 | char..3 | EUR |
| 8 | M | Y | respcode | Response code. [000] is OK, all the others deny transaction. (1) | char..3 | 000 |
| 10 | M | Y | datetime | Timestamp. Format: [YYYYMMDDhhmmss] ISO-8601 | int..14 | 20161028112930 |
| 10 | M | Y | msgdata | Payment description, cardholder name, etc. | char..40 | |
| 11 | M | Y | actiontext | Description of response code | char..40 | OK, approved |

| Nr | Availability (Mandatory/ Optional) | MAC Value | Request Parameter | Definition | Format | Example value |
|---|---|---|---|---|---|---|
| 12 | M | N | auto | Y - automatic feedback (2)<br>N - feedback via browser | char 1 | OK, approved |
| 13 | M | - | mac | Digital signature | hexadecimal | 4d5e875a245d42..... |

*Response Code*

We recommend to use different Response Codes only for your own purposes.
For clients display if the transaction was successful, and if not, advise them to contact their card issuer.

*Automatic feedback*

We always try to send automatic feedback the moment card transaction has finished on our side. We send it directly from our server to the feedBackUrl specified in the request. This only works on https (port:443)
Second time feedback will be sent when client clicks "Back to Merchant" button or progress bar reaches end.

**Example feedback data**

```
action=afb
ver=4
id=12ABCD1223
ecuno=201610280012
receipt_no=000015
eamount=1234
cur=EUR
respcode=000
datetime=20161028112930
msgdata=Cardholder Name
actiontext=OK, tehing autoriseeritud
mac=6EE6B987374E5DE0FAAD9ABB0DEB3ABA52E1CA4C715D6B67D7AD50D59913A09BCD69475C71F29D99C0
7D9F1D578E4452E2A427C767B7DDDF4F06B197E071FC9621A11B94596BF27764D69D22FED06A28AA72535A
80ACA3238A3A0D82C7CE543A13B5C1AB17CB662CF2F5BAF535E58018B10C73F6FE36D947104B0F79FBB8DC
81
charEncoding=UTF-8
auto=N
```

### 2.1.4. PHP code examples

#### example for request MAC calculation

```
# Prepare key

$key =('private.key'); # key file name and location
$fp = fopen("$key", "r");
$fs = filesize("$key");
$priv_key = fread($fp, $fs);
fclose($fp);

# concatenate data for MAC calculation
# padded to specification
# mb_str_pad function used for additionalinfo

$data = sprintf("%03s", $ver) . sprintf("%-10s", $id) . sprintf("%012s", $ecuno) .
sprintf("%012s", $eamount) . sprintf("%-3s", $cur) . sprintf("%014s", $datetime) .
sprintf("%-128s", $feedBackUrl) . $delivery . mb_str_pad($additionalinfo, '128');

# calculate sha1 signature and sign it

openssl_sign($data, $signature, openssl_get_privatekey($priv_key), OPENSSL_ALGO_SHA1);

# As this provides binary signature, the
# signature needs to be converted into hex.

$mac=bin2hex($signature);
```

*multibyte characters*

Please note that some fields (*additionalinfo* in request, *msgdata* and *actiontext* in response)might contain multibyte characters and therefore multibyte safe operations are needed. PHP's sprintf is NOT multibyte safe. Some custom multibyte safe function could be used instead. For example mb_str_pad function .

**example for response message MAC calculation**

```
# concatenate data for MAC calculation
# padded to specification
# mb_str_pad function used for msgdata and actiontext

$data = sprintf("%03s",$ver).sprintf("%-10s",$id).sprintf("%012s",$ecuno).sprintf(
"%06s",$receipt_no).sprintf("%012s",$eamount).sprintf("%3s",$cur).sprintf("%03s",$resp
code).$datetime.mb_str_pad($msgdata, '40').mb_str_pad($actiontext, '40');

# function to convert hex back to string

$mac = hex2bin($mac);


# Load NETS public key
$key = nets_estonia_pub_key
$fp = fopen("$key", "r");
$fs = filesize("$key");
$pub_key = fread($fp, $fs);
fclose($fp);

# verify the signature

$result = openssl_verify($data, $mac, $pub_key);
if ($result == 1) {
echo "Signature check OK<br>";
} elseif ($result == 0) {
echo "Signature NOT OK<br>";
} else {
echo "error checking signature<br>";
}
```

### mb_str_pad function

```
function mb_str_pad($input, $pad_length, $pad_string = ' ', $pad_type = STR_PAD_RIGHT,
$encoding = 'UTF-8')
{
    $input_length = mb_strlen($input, $encoding);
    $pad_string_length = mb_strlen($pad_string, $encoding);

    if ($pad_length <= 0 || ($pad_length - $input_length) <= 0) {
        return $input;
    }

    $num_pad_chars = $pad_length - $input_length;

    switch ($pad_type) {
        case STR_PAD_RIGHT:
            $left_pad = 0;
            $right_pad = $num_pad_chars;
            break;

        case STR_PAD_LEFT:
            $left_pad = $num_pad_chars;
            $right_pad = 0;
            break;

        case STR_PAD_BOTH:
            $left_pad = floor($num_pad_chars / 2);
            $right_pad = $num_pad_chars - $left_pad;
            break;
    }

    $result = '';
    for ($i = 0; $i < $left_pad; ++$i) {
        $result .= mb_substr($pad_string, $i % $pad_string_length, 1, $encoding);
    }
    $result .= $input;
    for ($i = 0; $i < $right_pad; ++$i) {
        $result .= mb_substr($pad_string, $i % $pad_string_length, 1, $encoding);
    }

    return $result;
}
```

### 2.1.5. Key generation

For live environment you need to generate new private and public key pair.

```
openssl genrsa -out myprivate.key 2048
openssl rsa -in myprivate.key -pubout > mypublic.key
```

Send the public key to us: webpos@estcard.ee

Private key goes into your web-shop environment.

### 2.1.6. Response Codes

https://en.wikipedia.org/wiki/ISO_8583#ISO_8583_version

*Table 3. Codes that approve transaction*

| Code | Explanation |
|---|---|
| 000 | Approved, OK |
| 001 | Approved with identification only |
| 002 | Approved partially (only for a device that has a partial dispense capability; amount from response) |
| 003 | Approved, OK (VIP) |

*Table 4. 1xx Codes that deny transaction, card to be returned to owner*

| Code | Explanation |
|---|---|
| 100 | Do not honor |
| 101 | Expired card |
| 102 | Fraud suspected, do not honor |
| 104 | Restricted card (ATM only) |
| 105 | Call the NETS |
| 106 | Allowable PIN tries exceeded |
| 109 | Invalid Merchant |
| 110 | Invalid Amount |
| 111 | Invalid Card Number |
| 112 | PIN required |
| 116 | Not sufficient funds |
| 117 | Incorrect PIN |
| 118 | Unknown Card |
| 119 | Transaction not permitted to cardholder |
| 120 | Transaction not permitted to terminal |
| 121 | Exceeds withdrawal amount limit |
| 123 | Exceeds withdrawal frequency limit |
| 125 | Card is not effective |
| 126 | Invalid PIN block |
| 127 | PIN length error |
| 128 | PIN key synch error |

*Table 5. 2xx Codes that deny transaction, card to be picked up*

| Code | Explanation |
|---|---|
| 200 | Do not honor |

| Code | Explanation |
|---|---|
| 201 | Expired card |
| 202 | Fraud suspected |
| 203 | Merchant contact NETS |
| 204 | Restricted card |
| 205 | Call the police |
| 206 | Allowable PIN tries exceeded |
| 208 | Lost card |
| 209 | Stolen card |
| 210 | Suspected counterfeit card |

*Table 6. 9xx Error codes from the system*

| Code | Explanation |
|---|---|
| 902 | Invalid transaction |
| 903 | Re-enter transaction |
| 904 | Format error |
| 907 | Issuer is Signed Off |
| 908 | Destination of message unknown |
| 909 | System malfunction |
| 911 | Card issuer timed out |
| 913 | Duplicate transmission |
| 918 | Missing transportation keys |
| 920 | Security device error, try again |
| 921 | Security device error |
| 923 | Request already in progress |
| 933 | Bad DateTime in reversal |
| 934 | Format error in response (for use only in offline transaction log and reversal) |
| 935 | Chip declined (for use only in offline transaction log) |