Paymentgateway Documentation

Version: 1.7.1

Document start date: 21-12-06

Maintenance history

Date	By	Version	Comments
21-12-2006	Daniel Bielefeldt	1.0	Document start
12-09-2007	Pelle Smidt	1.1	
07-03-2008	Daniel Bielefeldt	1.2	Added documentation for Mobile payment and better binary proxy explanation.
26-06-2008	Daniel Bielefeldt	1.3	Fixed uniqueorderid definition
08-07-2008	Daniel Bielefeldt	1.4	NEW enhanced security policy published PBS regarding merchants. APPENDIX B. Added documentation for secure proxy and payment window.
29-07-2008	Daniel Bielefeldt	1.5	Introduction added Reconstructed index
23-04-2009	Daniel Bielefeldt	1.5.5	Securitytext option is removed in section 3.1 Extra CSS class is added to submit button in secureproxy Added new section about client requirements.
04-06-2009	Daniel Bielefeldt	1.6.0	Split payment feature is added to secureproxy, pamentwindow and postform api.
16-07-2009	Daniel Bielefeldt	1.6.1	Removed old API description.
25-02-2010	Daniel Bielefeldt	1.6.2	An updated is added too appendix A about md5check enforced security
07-09-2010	Daniel Bielefeldt	1.6.3	Documentation for extended validation and prefilled cvc value is added in section 2 Q8-LIC cardprefix has been removed New option to show the last 4 creditcard digits.
07-04-2011	Daniel Bielefeldt	1.6.4	Added secureproxy examples for edankort, 3dsecure and netbank

Date	By	Version	Comments
11-04-2011	Daniel Bielefeldt	1.6.5	Added documentation for supplementary authorize and split capture
13-04-2011	Daniel Bielefeldt	1.6.6	New option to enable google analytics in paymentwindows and secureproxy
30-06-2011	Daniel Bielefeldt	1.6.7	Supplementary authorize can now be used with all transaction types.
30-11-2011	Daniel Bielefeldt	1.6.8	Added documentaion about MAC function, wich is a replacement for md5checksum. An correction is added to the 3dsecure documentation.
12-09-2011	Daniel Bielefeldt	1.6.9	Note added to MAC description.
07-12-2012	Daniel Bielefeldt	1.7.0	Added versioning chapter to secureproxy documentation.
26-02-2013	Daniel Bielefeldt	1.7.1	Added new secureproxy placeholder for submitbutton. Note added about variables returned with callbackurl.

Index

Introduction

This document describes how you implement online payment into your webshop. We support a wide range of methods to integrate online payment. This intro will hopefully point you in the right direction concerning which method will fit your shop the best.

First of all we highly recommend that **no** card data is stored locally in any database or temporary memory. If that is not an option, then you have to take a closer look at the PCI security standards council. If you want to be able to handle cardholder data, your shop and server where your data is located has to pass a PCI certification. Read more about PCI on the following site: https://www.pcisecuritystandards.org/

If you don't want/need to store cardholder data, please attend to the following 3 methods.

1. Payment window (Hosted solution)
2. Secure SSL Proxy (Hosted solution)

Payment window is a quick start solution where you don't have to make lots of adjustments. Start by setting up a simple postform, and then look at what arguments you can send to the payment window. The window will be able to display your shop logo, total price and order number. You can either choose to open the window in the same window as the shop, or make it open as a popup. The downside about the payment window is that you can't change the design to fit your shop.

Secure SSL proxy is a very different and more time-consuming way to integrate online payment into your shop. The big difference from the payment window is that you are now able to design the layout of your payment page. Please read section 2.1 for more information.

Postform API can only be used by shops that have performed an SAQ, and uploaded quarterly security scans to PBS's security portal. Please be aware that this SAQ only concern API methods that include creditcard information. Section 4.3 and 4.6.

Client requirements

1. Client browser needs to be able to handle cookies

Hosted secure SSL proxy

How Secure SSL proxy works

The payment form has to be displayed through SSL (HTTPS). We provide an SSL Proxy within the paymentgateway product, in case the shop doesn't have its own SSL certificate. Remember that sessions created when the customer is connected to your shop through HTTP, will be lost when data is submitted to the https proxy.

To allow a specific domain through the SSL-Proxy, it first has to be allowed in the access list. The access list is located in the paymentgateway webinterface under "Indstillinger / Settings"

Relaying the payment form through SSL is done by pasting the payment form URL in front of the SSL-Proxy URL.

Example:

SSL-Proxy URL: https://betaling.curanet.dk/secureproxy/proxy.php/v1.1
Webshop URL: http://www.webshop.dk/payform.html

Relay URL:
https://betaling.curanet.dk/secureproxy/proxy.php/http://www.webshop.dk/payform.html

If your website includes special binary elements, that the proxy does not understand, it is possible to insert the binary proxy manually by inserting the following URL before the binary element.

Example:

<img src="
https://betaling.curanet.dk/secureproxy/f.php?f=http://www.shop.dk/images/billede.jpg">"

CSS "get" paramenter
It isn't possible to use "get" parameter with css files

Stripping tags
The big difference between this proxy and the old one is that this proxy will strip tags which can be used to manipulate creditcard data.

The following tags and code in between will be stripped from the code that parses through the proxy:

<script>,<object>,<embed>,<applet>,<noframes>,<input>
<select>,<textarea>,<form>,<checkbox>,<frameset>,<iframe>

Instead of inserting your own postform and input fields, we have built some placeholders you can insert into your code. When your code parses through the proxy, it will replace the placeholders with either a hidden or text input field. Every text input field has its own css class, with which you will be able to style the inputs to reflect the design of your own website.

CSS classes:
cardnum, emonth, eyear, cvc and formsubmit.

Placeholder syntax:
%%%fieldname_value%%%

All the options listed in the postform API arguments table, are available in the secureproxy. Just remember to use uppercase characters in the fieldname.

To put in extra self defined post fields, use the following syntax
%%%POSTVAR_var1=value&var2=value2%%%

Extra self defined post fields, will be parsed on to accept or decline page.

See this example as a guideline.
There is a placeholder for each of the postfields specified in section 4.1.

```
<html>
<head>

<title> Payment page </title>
</head>

<body>

%%%FORMSTART%%%
%%%SHOPID_200893123123%%%
%%%AMOUNT_500%%%

%%%ORDERID_23123213%%%

%%%CURRENCY_208%%%
%%%PAYTYPE_creditcard%%%
%%%AUTHTYPE_auth%%%
%%%CHECKMD5_MD5CHECKSUM%%%
%%%ACCEPTURL_http://www.shop.dk/acceptpage.php%%%
%%%DECLINEURL_http://www.shop.dk/declinepage.php%%%
%%%CALLBACKURL_http://www.shop.dk/status.php%%%
%%%CARDNUM%%%<BR>
%%%EXPMONTH%%% / %%%EXPYEAR%%%<BR>
%%%CVC%%%<BR>

%%%FORMSUBMIT_Make Payment%%%
%%%FORMEND%%%

</body>
</html>
```

To show danish characters on pages that parses through the proxy, use ASCII syntax.

Split payment

To use split payment together with secure proxy, use the following placeholders.

```
%%%SPLIT_true%%%
%%%TRANSACT_1_amount_500%%%
%%%TRANSACT_1_orderid_23123213%%%
%%%TRANSACT_1_orderidprefix_WF%%%
%%%TRANSACT_1_authtype_auth%%%


%%%TRANSACT_2_amount_500%%%
%%%TRANSACT_2_orderid_23123213%%%
%%%TRANSACT_2_orderidprefix_WF%%%
%%%TRANSACT_2_authtype_subscribe%%%
```

When a split payment is returned to accept or decline page, there is added a number to each GET parameter. This number is the same as the one that are giving to the split payment, as showed in the above example. The same goes for the callbackurl option.

Extended input validation

Use extended input valdation, instead of waiting for the result, when card data is submitted to the gateway. Replace the ordinary placeholders with the following, to add extended input validation. For each input an image will indicate if the entered data is correct or not.

Start by adding placeholder for the validation code. This should be added into your HTML header.

```
<html>
<head>

<title></title>
%%%JS_INCLUDE%%%
</head>
<body>
```

Insert the following placeholders. If you already are using placeholders without the "validate" prefix, then just add the "validate" prefix.

```
<table border="0" cellpadding="0" cellspacing="0">
<tr>
 <td>%%%CARDNUM_VALIDATE%%%<BR></td>
 <td>%%%CARDNUM_VALIDATE_IMAGE%%%</td>
</tr>
<tr>
 <td>%%%EXPMONTH_VALIDATE%%% / %%%EXPYEAR_VALIDATE%%%</td>
```

```
 <td>%%%EXPIRE_VALIDATE_IMAGE%%%</td>
</tr>
<tr>
 <td>%%%CVC_VALIDATE%%%</td>
 <td>%%%CVC_VALIDATE_IMAGE%%%</td>
</tr>
</table>
```

Credit/Debit cards without CVC value

Some credit/debit cards like Forbrugsforeningen has no CVC value, and in such case, it can be handy to prefill CVC field with "000" and only make it readable. This is done by the replaceing the placeholder for CVC with:

```
%%%CVC_000%%%
```

If extended input validation is used, the image will just show that the value is correct.

Versioning

We have now introduced versioning in the proxy, so that we can make changes to the parsing engine, and make sure that we don't break any customer code that depends on a specific parsing version.

Version 1.0 is the standard version. Everytime we change the proxy this wil result in a new version.

New things in version 1.1

1. Supports double slash in url string. If double slash is used the proxy will follow the default protocol scheme.
2. Allowed protocols is now extended to http and https

New things in version 1.2

1. New placeholder added for form submit button. This button will disable itself, when used. This is to insure that customers won't push the submit button more than once.
   The new placeholder is:

```
%%%FORMSUBMITV2_Make Payment%%%
```

To identify the version number for the proxy, call the proxy url like: https://betaling.curanet.dk/secureproxy/proxy.php/v1.1/http://www.yourdomain.dk/page.html

Hosted payment window

Payment window post arguments

Conventions

X	Mandatory
?	Optional
%	N/A
CC	Credit card
3Ds	3Dsecure

Post arguments

Field	Arguments	Type	Description	CC	3Ds
shopid		int(?)	A numeric id to identify the postform.	X	X
currency		int(3)	Define currency for the specific transaction. Use a numeric value from ISO 4217. If this field is empty the gateway will use the default value. The default value is set when logging into the payment gateway webinterface.	?	?
amount		int(?)	Specify an amount for this transaction. You must use minor unit to specify the amount.	X	X
orderid		int(19)	Orderid for the specific transaction.	X	X
orderidprefix		char(4)	Orderidprefix is a way to append a prefix to the orderid.	?	?
paytype	creditcard 3dsecure	char(10)	Assign one of the following arguments to define the type of transaction.	X	X
cardtype	DK V-DK VISA(DK) MC(DK) MC MSC(DK) MSC DINERS(DA) DINERS AMEX(DA) AMEX VISA JCB FBF Q8-LIC	char(76)	With cardtype it's possible to pick which types of credit cards which are accepted in the given transaction. This is useful to calculate the fee for a specific payment. Remember that this list is a whitelist. Separate values using comma.	?	?

authtype	auth subscribe suppauth	char(9)	Authtype defines how a transaction is handled. It can be made as a subscription, supplementary auth or as plain authentication.	?	%
checkmd5		char(?)	Use”checkmd5” to ensure that the postform is unchanged while the transaction is being processed. See ”appendix a” for more details.	?	?
uniqueorderid	true false	char(5)	Use uniqueorderid to ensure that only unique orderid’s are used.	?	?
accepturl		char(?)	The url where the customer is redirected to, when a successful transaction has been made. This option will overrule the accepturl defined in the webinterface.	?	?
declineurl		char(?)	The url where the customer is redirected to, when a transaction has been declined. This option will overrule the declineurl defined in the webinterface.	?	?
callbackurl		char(?)	Callbackurl can make a request back to a predefined URL. Use this option together with mobile payment, credicard, edankort and 3DSecure. Verify the payment by using a new get paramenter called "checkmd5callback"	?	?
lang	da, en, de, es, it, nl, no, pt, ru, se	char(2)	Define which language to display. Default language is Danish.	?	?
split	true false	char(5)	Enable split payment. This option can split one payment into 2 or more transactions. This is useful, if your shop provides split shipping, as split payment will allow you to make one transaction, for each of every product in one order. By enabling this option amount, orderid, orderidprefix and authtype will be obsolete. Be a ware that this feature is only supported with creditcard transactions.	?	%

transact		POST array	This field I used together with split payment. Every split payment is defined as an array. Example is available under section 3.2	?	%
cardnomask	true false	char(5)	The last 4 digits will bee attached to to the accepturl and callbackurl. The variable is named "cardnomask"	?	?
googleanalytics	uacode	char(14)	Enable google analytics javascript code in paymentwindow	?	?
mac		char(32)	Message authentication control, ensures that data isn't tampered during data transmission. See "appendix c" for more details.	?	?

Payment window post example

```
1  <html>
2  <head>
3
4  <title></title>
5  <script language="javascript"
6  src="https://betaling.curanet.dk/customers/curanet/js/openpaymentwin
7  dow.js"></script>
8
9  </head>
10 <body>
11
12 <a href="#" onClick="openPaymenWindow()">Open paymentwindow</a>
13
14 <form action="https://betaling.curanet.dk/paymentwindow.php"
15 method="post" target="curanet_paymentwindow" name="curanet"
16 id="curanet">
17 <input type="hidden" name="shopid" value="SHOPID">
18 <input type="hidden" name="currency" value="CURRENCY (DKK 208)">
19 <input type="hidden" name="amount" value="AMOUNT">
20 <input type="hidden" name="orderid" value="ORDERID">
21 <input type="hidden" name="paytype" value="creditcard">
22 <input type="hidden" name="uniqueorderid" value="true / false">
23 <input type="hidden" name="lang" value="da / en">
24
25 <input type="hidden" name="accepturl" value="ACCEPTURL">
26 <input type="hidden" name="declineurl" value="DECLINEURL">
27 </form>
28
29 </body>
30 </html>
```

To change the top logo in the payment window, log onto the paymentgateway webinterface and select "Indstillinger / Settings". Find the input field called

"Betalings-vindue logo". Type in the URL where the logo is located, this must be defined as a HTTP referal.

Postfields added which are not enforced by the argument list, will be sent to the accept or decline page.

Add the following post fields to enable split payment. Remember that "amount", "orderid", "orderidprefix" and "authtype" is obsolete, when split payment is enabled. The amount of each split transaction is summarized, and displayed as total amount in the paymentwindow.

```
<input type="hidden" name="split" value="true">

<input type="hidden" name="transact[1][amount]" value="AMOUNT">
<input type="hidden" name="transact[1][orderid]" value="OREDERID">
<input type="hidden" name="transact[1][orderidprefix]"
value="ORDERIDPREFIX">
<input type="hidden" name="transact[1][authtype]" value="AUTHTYPE">

<input type="hidden" name="transact[2][amount]" value="AMOUNT">
<input type="hidden" name="transact[2][orderid]" value="OREDERID">
<input type="hidden" name="transact[2][orderidprefix]"
value="ORDERIDPREFIX">
<input type="hidden" name="transact[2][authtype]" value="AUTHTYPE">
```

When a split payment is returned to accept or decline page, there is added a number to each GET parameter. This number is the same as the one that are giving to the split payment, as showed in the above example. The same goes for the callbackurl option.

If authtype is set differently on 2 transactions, the amount will be split up and showed as "amount to be captured right now" and "subscribe amount"

Postform API solutions

Postform API arguments

Conventions

X	Mandatory
?	Optional
%	N/A
CC	Credit card
eDK	eDankort
3Ds	3Dsecure
MB	Mobile pay
NB	Netbank

Postform description

Field	Arguments	Type	Description	CC	eDK	3Ds	MB	NB
shopid		int(?)	A numeric id to identify the postform.	X	X	X	X	X
currency		int(3)	Define currency for the specific transaction. Use a numeric value from ISO 4217. If this field is empty the gateway will use the default value. The default value is set when logging into the payment gateway webinterface.	?	?	?	?	X
amount		int(?)	Specify an amount for this transaction. You must use minor unit to specify the amount.	X	X	X	X	X
orderid		int(19)	Orderid for the specific transaction.	X	X	X	X	X
orderidprefix		char(4)	Orderidprefix is a way to append a prefix to the orderid.	?	?	?	?	?

paytype	creditcard edankort netbank 3dsecure mobilepay icash	char(10)	Assign one of the following arguments, to define the type of transaction.	X	X	X	X	X
outputFormat	html or wml	char(4)	Output format when displaying the mobile payment form. If output format is empty, the default output will be HTML.	%	%	%	?	?
bankname	DanskeBank	char(10)	NetBank authorizations	%	%	%	%	X
bankMerchantID		char(?)	Bank Merchant ID	%	%	%	%	X
cardtype	DK V-DK VISA(DK) MC(DK) MC MSC(DK only 3Dsecure) MSC (only 3Dsecure) DINERS(DA) DINERS AMEX(DA) AMEX VISA JCB FBF Q8-LIC	char(76)	With cardtype it's possible to pick which types of credit cards which are accepted in the given transaction. This is useful to calculate the fee for a specific payment. Remember that this list is a whitelist. Separate values using comma.	?	?	?	?	%
authtype	auth subscribe suppauth	char(9)	Authtype defines how a transaction is handled. It can be made as a subscription, supplementary auth or as plain authentication.	?	%	%	?	%
checkmd5		char(?)	Use"checkmd5" to ensure that the postform is unchanged while the transaction is being processed. See	?	?	?	?	%

			”appendix a” for more details.					
uniqueorderid	true false	char(5)	Use uniqueorderid to ensure that only unique orderid’s are used.	?	?	?	?	%
cardnum		int(19)	Credit card number	X	%	X	%	%
emonth		int(2)	Expiry month ex. (05)	X	%	X	%	%
eyear		int(2)	Expiry year ex. (07)	X	%	X	%	%
cvc		int(4)	Credit card verification number.	X	%	X	%	%
accepturl		char(?)	The url where the customer is redirected to, when a successful transaction has been made. This option will overrule the accepturl defined in the webinterface.	?	?	?	%	?
declineurl		char(?)	The url where the customer is redirected to, when a transaction has been declined. This option will overrule the declineurl defined in the webinterface.	?	?	?	%	?
callbackurl		char(?)	Callbackurl can make a request back to a predefined URL. Use this option together with mobile payment, credicard, edankort and 3DSecure. Verify the payment by using a new get paramenter called	?	?	?	?	%

			"checkmd5callback"					
statusurl		char(?)	When using netbank payment, it's possible to define a call-back URL. This callback request will be made from the bank, to ensure that the transaction was successfully made.	%	%	%	%	?
split	true false	char(5)	Enable split payment. This option can split one payment into 2 or more transactions. This is useful, if your shop provides split shipping, as split payment will allow you to make one transaction, for each of every product in one order. By enabling this option amount, orderid, orderidprefix and authtype will be obsolete. Be a ware that this feature is only supported with creditcard transactions.	?	%	%	%	%
transact		POST array	This field I used together with split payment. Every split payment is defined as an array. Example is available under section 3.2	?	%	%	%	%

cardnomask	true false	char(5)	The last 4 digits will bee attached to to the accepturl and callbackurl. The variable is named "cardnomask"	?	?	?	%	%
mac		char(32)	Message authentication control, ensures that data isn't tampered during data transmission. See "appendix a" for more details.	?	?	?	?	?

eDankort api authorize example

```
1  <form method="post" action="https://betaling.curanet.dk/auth.php">
2   <input type="hidden" name="shopid" value="SHOPID">
3   <input type="hidden" name="currency" value="CURRENCY (DKK 208)">
4   <input type="hidden" name="amount" value="AMOUNT">
5   <input type="hidden" name="orderid" value="ORDERID">
6   <input type="hidden" name="paytype" value="edankort">
7   <input type="hidden" name="uniqueorderid" value="true / false">
8
9   <input type="hidden" name="accepturl" value="ACCEPTURL">
10  <input type="hidden" name="declineurl" value="DECLINEURL">
11
12  <input type="submit" name="submit" value="Payment">
13 </form>
```

We recommend disabling the submit button after the customer has submitted the form.

This can be done by replacing the first line in the example above with the one below.
Remember to name your submit button "submit".

```
<form method="post" action="
https://betaling.curanet.dk/auth.php" onSubmit="submit.disabled
= true">
```

When performing an edankort authentication there is no need to have any input fields. When the customer has pressed the button, the customer will automatically redirect to the netbank for authentication. **Remember that it's only possible to test in production mode.**

This form has to be displayed on an SSL (HTTPS) connection.

3D Secure api authorize example

```
1  <form method="post" action="https://betaling.curanet.dk/auth.php">
2   <input type="hidden" name="shopid" value="SHOPID">
3   <input type="hidden" name="currency" value="CURRENCY ( DKK 208 )">
4   <input type="hidden" name="amount" value="AMOUNT">
5   <input type="hidden" name="orderid" value="ORDERID">
6   <input type="hidden" name="paytype" value="3dsecure">
7   <input type="hidden" name="uniqueorderid" value="true / false">
8
9   <input type="hidden" name="accepturl" value="ACCEPTURL">
10  <input type="hidden" name="declineurl" value="DECLINEURL">
11
12  <input type="text" name="cardnum" size="25">
13  <input type="text" name="emonth" size="2">
14  <input type="text" name="eyear" size="2">
15  <input type="text" name="cvc" size="3">
16
17
18  <input type="submit" name="submit" value="Payment">
19 </form>
```

We recommend disabling the submit button after the customer has submitted the form.

This can be done by replacing the first line in the example above with the one below. Remember to name your submit button "submit".

```
<form method="post" action="
https://betaling.curanet.dk/auth.php" onSubmit="submit.disabled
= true">
```

When performing a 3Dsecure authentication, the customer has to fill in the card number, expiry month, expiry year and CVC. After the customer has pressed the button, the customer will automatically be redirected to a verification page, where the customer needs to authenticate the hi/her is the right owner of the card.
Remember that it's only possible to test in production mode.

This form has to be displayed on an SSL (HTTPS) connection.

Mobile Payment api authorize example

```
1  <form method="post" action="https://betaling.curanet.dk/auth.php">
2   <input type="hidden" name="shopid" value="SHOPID">
3   <input type="hidden" name="currency" value="CURRENCY ( DKK 208 )">
4   <input type="hidden" name="amount" value="AMOUNT">
5   <input type="hidden" name="orderid" value="ORDERID">
6   <input type="hidden" name="paytype" value="mobilepay">
7   <input type="hidden" name="outputFormat" value="wml/html">
8
9   <input type="hidden" name="callbackurl" value="CALLBACKURL">
10
11
12  <input type="submit" name="submit" value="Payment">
13 </form>
```

We recommend disabling the submit button after the customer has submitted the form.

This can be done by replacing the first line in the example above with the one below. Remember to name your submit button "submit".

```
<form method="post" action="
https://betaling.curanet.dk/auth.php" onSubmit="submit.disabled
= true">
```

NetBank api authorize example

```
1  <form method="post" action="https://betaling.curanet.dk/auth.php">
2   <input type="hidden" name="shopid" value="SHOPID">
3   <input type="hidden" name="currency" value="CURRENCY ( DKK 208 )">
4   <input type="hidden" name="amount" value="AMOUNT">
5   <input type="hidden" name="orderid" value="ORDERID">
6   <input type="hidden" name="paytype" value="netbank">
7   <input type="hidden" name="bankname" value="BankName">
8   <input type="hidden" name="bankMerchantID" value="BankMerchantID">
9
10  <input type="hidden" name="accepturl" value="ACCEPTURL">
11  <input type="hidden" name="declineurl" value="DECLINEURL">
12  <input type="hidden" name="statusurl" value="STATUSURL">
13
14
15  <input type="submit" name="submit" value="Payment">
16 </form>
```

We recommend disabling the submit button after the customer has submitted the form.

This can be done by replacing the first line in the example above with the one below. Remember to name your submit button "submit".

Transactions made with netbank, will not be shown in the transaction list on the webinterface. They will only be available through Danske Netbank business online login – a system maintained by Danske Bank.

```
<form method="post" action="
https://betaling.curanet.dk/auth.php" onSubmit="submit.disabled
= true">
```

When performing a netbank transaction, after the customer has pressed the button the customer will automatically be redirect to the netbank for authentication. **Remember that it's only possible to test in production mode.**

This form has to be displayed on an SSL (HTTPS) connection.

Creditcard api authorize example

```
1  <form method="post" action="https://betaling.curanet.dk/auth.php">
2   <input type="hidden" name="shopid" value="SHOPID">
3   <input type="hidden" name="currency" value="CURRENCY (DKK 208)">
4   <input type="hidden" name="amount" value="AMOUNT">
5   <input type="hidden" name="orderid" value="ORDERID">
6   <input type="hidden" name="paytype" value="creditcard">
7   <input type="hidden" name="uniqueorderid" value="true / false">
8
9   <input type="hidden" name="accepturl" value="ACCEPTURL">
10  <input type="hidden" name="declineurl" value="DECLINEURL">
11
12  <input type="text" name="cardnum" size="25">
13  <input type="text" name="emonth" size="2">
14  <input type="text" name="eyear" size="2">
15  <input type="text" name="cvc" size="3">
16
17  <input type="submit" name="submit" value="Payment">
18 </form>
```

We recommend disabling the submit button after the customer has submitted the form.

This can be done by replacing the first line in the example above with the one below. Remember to name your submit button "submit".

```
<form method="post" action="
https://betaling.curanet.dk/auth.php" onSubmit="submit.disabled
= true">
```

eDankort authorize example using secureproxy

```
%%%FORMSTART%%%
%%%SHOPID_200893123123%%%
%%%AMOUNT_500%%%
%%%ORDERID_23123213%%%
%%%CURRENCY_208%%%
%%%PAYTYPE_edankort%%%
%%%ACCEPTURL_http://www.shop.dk/acceptpage.php%%%
%%%DECLINEURL_http://www.shop.dk/declinepage.php%%%
%%%CALLBACKURL_http://www.shop.dk/status.php%%%
%%%FORMSUBMIT_Make Payment%%%
%%%FORMEND%%%
```

When performing an edankort authorize there is no need to have any input fields. When the customer has pressed the button, the customer will automatically redirect to the netbank for authentication. **Remember that it's only possible to test in production mode.**

3D Secure authorize example using secureproxy

```
%%%FORMSTART%%%
%%%SHOPID_200893123123%%%
%%%AMOUNT_500%%%
%%%ORDERID_23123213%%%
%%%CURRENCY_208%%%
%%%PAYTYPE_3dsecure%%%
%%%ACCEPTURL_http://www.shop.dk/acceptpage.php%%%
%%%DECLINEURL_http://www.shop.dk/declinepage.php%%%
%%%CALLBACKURL_http://www.shop.dk/status.php%%%
%%%CARDNUM%%%<BR>
%%%EXPMONTH%%% / %%%EXPYEAR%%%<BR>
%%%CVC%%%<BR>
%%%FORMSUBMIT_Make Payment%%%
%%%FORMEND%%%
```

When performing a 3Dsecure authorize, the customer has to fill in the card number, expiry month, expiry year and CVC. After the customer has pressed the button the customer will automatically be redirected to the netbank for authentication. **Remember that it's only possible to test in production mode.**

Netbank authorize example using secureproxy

```
1  %%%FORMSTART%%%
2  %%%SHOPID_200893123123%%%
3  %%%AMOUNT_500%%%
4  %%%ORDERID_23123213%%%
5  %%%CURRENCY_208%%%
6  %%%PAYTYPE_netbank%%%
7  %%%BANKNAME_DanskeBank%%%
8  %%%BANKMERCHANTID_shopname%%%
9  %%%ACCEPTURL http://www.shop.dk/acceptpage.php%%%
10 %%%DECLINEURL_http://www.shop.dk/declinepage.php%%%
11 %%%CALLBACKURL_http://www.shop.dk/status.php%%%
12 %%%FORMSUBMIT_Make Payment%%%
13 %%%FORMEND%%%
```

When performing a netbank transaction, after the customer has pressed the button the customer will automatically be redirect to the netbank for authentication. **Remember that it's only possible to test in production mode.**

Using google analytics with secureproxy

```
1  %%%FORMSTART%%%
2  %%%SHOPID_200893123123%%%
3  %%%AMOUNT_500%%%
4  %%%ORDERID_23123213%%%
5  %%%CURRENCY_208%%%
6  %%%ACCEPTURL_http://www.shop.dk/acceptpage.php%%%
7  %%%DECLINEURL_http://www.shop.dk/declinepage.php%%%
8  %%%CALLBACKURL_http://www.shop.dk/status.php%%%
9  %%%FORMSUBMIT_Make Payment%%%
10 %%%FORMEND%%%
11 %%%GOOGLEANALYTICS_UA-00000000-0%%%
```

Remember to place the placeholder for googleanalytics before your </body> tag at the bottom of the payment page.

API Documentation

All API connections are based on SOAP requests. We recommend using one of the functions that are made public on our websites download section.

The public examples describe how to make connection through ASP 3.x, ASP.NET and PHP.

Action code list

Action code	Description
0	Successful action
1	Transaction declined
2	Possible fraud
3	Communication error
4	Card is expired
5	PBS internal system error
6	Invalid Transaction
7	System error
8	Wrong merchant number
9	No card record
10	Card entry below low range
11	Transaction not permitted to terminal
12	Transaction not permitted to cardholder
13	Invalid card number
14	Unauthorized content in cardnum field
15	Unauthorized content in expiry month
16	Unauthorized content in expiry year
17	Unauthorized content in CVC
18	Card number is not authorized according to cardtype.
19	Not a unique orderid
20	Empty amount
21	Not a valid md5checksum
22	Netbank authorize failed
23	Netbank authorize cancelled by customer.
24	Icash payment failed
25	MAC hash is invalid

Subscribe authentication

Request acct example

```
<form method="post" action="https://betaling.curanet.dk/auth.php">
 <input type="hidden" name="shopid" value="SHOPID">
 <input type="hidden" name="currency" value="CURRENCY (DKK 208)">
 <input type="hidden" name="amount" value="AMOUNT">
 <input type="hidden" name="orderid" value="ORDERID">
 <input type="hidden" name="paytype" value="creditcard">
 <input type="hidden" name="authtype" value="subscribe">
 <input type="hidden" name="uniqorderid" value="TRUE / FALSE">

 <input type="hidden" name="accepturl" value="ACCEPTURL">
 <input type="hidden" name="declineurl" value="DECLINEURL">

 <input type="text" name="cardnum" size="25">
 <input type="text" name="emonth" size="2">
 <input type="text" name="eyear" size="2">
 <input type="text" name="cvc" size="3">

 <input type="submit" name="submit" value="Payment">
</form>
```

It's only possible to use credit card when performing subscribe transactions. After a successful action, the transaction will be listed under "subscribe payments" in the webinterface. The transaction is only a subscribe payment. This means that before a capture can be made, an "acct authenticate" has to be created.

Request acct authenticate example

When performing an acct authenticate, we recommend using of ASPtear or file_get_contents.
It is also possible to make this request directly from a browser.

Fill in the following parameters, and request the URL. The acct authenticate, can be requested as a batchlist, or as a single request.

When sending a single request. Just fill in the transaction ID, that was delivered whit the accepturl when the request acct was made. The second option is the amount for this specific transaction. The last option is the orderid, which can be used as a reference number between the customer order and the transaction.

Batchlist=transacnum1;amount1;orderid1

When sending more than one request, just duplicate the options, and separate them with a comma.

Batchlist=transacnum1;amount1;orderid1;orderidprefix1,transacnum2;amount2;orderid2;orderidprefix2

URL:
https://betaling.curanet.dk/authsubscribe.php?batchlist=transacknum1;amount1;orderid1;orderidprefix1

The response from this action will be returned as plain text.

If a successful action was made, the response will look like this.
```
APPROVED;ORIGINALTRANSNUM;NEWTRANSNUM;AMOUNT;ORDERID;ORDERIDPREFIX
```

If the action fails, the response will look like this.
```
FAILED;ORIGINALTRANSNUM;0;AMOUNT;ORDERID;ORDERIDPREFIX
```

Accept and decline return parameters

Following GET parameters are returned together with the accepturl and callbackurl.

Field	Parameters		
transacknum	A unique id which is used, to identify the transaction. The id will be available at the account balance.		
orderid	Orderid provided when the authentication was requested		
amount	Amount returned in minor unit		
currency	Currency returned as numeric format. ISO 4217		
cardtype	**Prefix**	**Cardname**	**Country**
	DK	Dankort	Danish
	V-DK	Visa Dankort	Danish
	VISA(DK)	Visa Electron	Danish
	MC(DK)	Euro/Mastercard	Danish
	MC	Euro/Mastercard	Foreign
	MSC(DK)	Maestro	Danish
	MSC	Maestro	Foreign
	DINERS(DA)	Diners club	Danish
	DINERS	Diners club	Foreign
	AMEX(DA)	American Express	Danish
	AMEX	American Express	Foreign
	VISA	Visa	Foreign
	EDK	eDankort	Danish
	JCB	JCB	Foreign
	FBF	Forbrugsforeningen	Danish
	Q8-LIC	Q8-LIC	Danish
	DanskeBank	Danske Bank	Danish
	N/A	Unknown	Unknown
actioncode	A numeric value which refers to the action code list		

Note: All custom post variables that are sent to the gateway, will be relayed as GET parameters to callbackurl.

Following GET parameters are returned together with the declineurl.

Field	Parameters
orderid	Orderid is provided when an authentication is requested
actioncode	A numeric value which refer to action code list

Extra return values from callbackurl

Field	Parameters
checkmd5	Use"checkmd5" to ensure that the postform is unchanged while the transaction is processed. See "appendix a" for more details.

Supplementary authorize

Supplementary authorize is a way of extend transaction default life time. If your shop is selling products that are not made available or shipped within 7 days, you can extend transaction lifetime with as many days as you want, until the product is ready to be shipped or made available for the customer.

Update: Supplementary authorize can be used for all transaction types. Including eDankort and 3Dsecure.

The way supplementary authorize is working, is by setting postfield "authtype" to "suppauth". The first card authorize will be made with zero amount, and the original amount will not be reserved on the customers bank account. This check is not for guaranteeing any coverage on the credit card, but only to make sure that the card is valid.

When the product is ready to be shipped or made available for the customer, press the capture button in the webinterface or by using the API. The capture process now makes another authorize, which first of all makes sure that the card is valid, and second, that the money is garanteed. If the capture failed, it's properly because that the card is reported stolen, not valid anymore or that there isn't coverage for the money on the customers bank account.

To implement supplementary authorize, please look at the "authtype" option in the postform API agruments list. The same option is avalible under secureproxy and paymentwindow.

Split capture

Sometimes it can be usefull to split a transaction in to pieces, if the product is shipped in parts. Split capture can be activated by setting "authtype" to "suppauth". When a transaction is made as a supplementary authorize, it will be possible to specify the amount you want to capture. The transaction is first marked as captured, when the entire amount is captured or the transaction is marked canceled.

If only part of the amount is captured, and the transaction is marked as canceled, it will still be marked as captured. Only mark the transaction as canceled if not the entire amount needs to be captured.

If the amount field is zero, when using API to capture with, the entire amount is captured.

Note: eDankort transactions lifetime is 30 days regardless if supplementary authorize is used. This means that if split captured is used with a eDankort transactions, it has to be fully captured within 30 days.

Appendix

Appendix A

MD5 checksum is used to verify data, which have been posted from the webshop to the paymentgateway. Every important data filed in the postform, is included in this verify check. We highly recommend enabling this feature.

The feature has to be enabled, before the paymentgateway will react on this field. Enabling md5checksum is done by clicking the checkbox that refer to md5checksum. The checkbox is located in the paymentgateway webinterface under "Indstillinger / Settings".

After the checkbox is marked, there are 4 dropdown menus and one key field which have to be filled out. The most appropriate way to do this is to select a different value in all of the 4 dropdown menus. In the last key field, type in a secret that is only used in this md5checksum.

The last part is to add the md5cheksum field to the post form, and fill in the right md5checksum.

If using PHP, ASP 3.x or asp.net, we provide some examples that are made public at our websites download section.

When using split payment, the fields are added together. That means if you have 2 split transactions, you have to add both amounts and orderid, if those are the md5 criteria.

Update!!
Callbackurl function now has a different md5checksum to ensure that it isn't possible to fake it. By looking under settings in the paymentgateway webinterface, you will find 5 new fileds, that applies to the callbackurl md5checksum. The new row is located below the old md5checksum. The difference between these two rows is the title that defindes them as auth and callback. The upper row is the same as always, but the new row applies to the new field that is returned as "get" parameter with the callback url. The new field is named "checkmd5callback"

Appendix B

Because of PBS' new enhanced security policies regarding merchants, it is no longer possible to use our normal postform procedure. As of 30th of September 2008, PBS require all new merchants to perform an SAQ (Self Assessment Questionnaire) together with quarterly security scan. The results of the SAQ and quarterly scans, has to be made available at the PBS trustkeeper portal.

If your shop is already running, the new requirements only have to be enforced by the end of September 2009.

Please be aware that the new requirements from PBS only concern merchants that uses Application Protocol Interface (API), also known as postform procedure. To avoid the SAQ, we will recommend implementing our secure proxy or payment window. Payment window and secure proxy are both referred to as hosted solutions, and they are thereby not a subject to the new requirements published by PBS.

Please contact PBS for more information about the SAQ requirements.

Appendix C

Short intro about MAC function
Message authentication code is a function that will ensure postform data isn't tampered during data transmission. This function is a replacement for the old m5checksum, that didn't include all postform data. If you are using md5checksum, we highly recommend that you upgrade to this mac function instead.

The principles are the same in MAC and md5 checksum. The big difference is that all fields that are sent to the gateway must be included in the MAC hash.

Usage
Login to the paymentgateway interface, and enable MAC by clicking the checkbox in the MAC section under settings and press the button "Aktiver Mac". Next step is the Mackey. The key is also generated in the same place, as where you enabled MAC. Press the button "Genererer nøgle", and a new key is displayed in the field above.

To secure your form with MAC, add a new postform field named "mac". The value is an md5 hash of every field value you are posting. Concatenate the values in the same order as they are listed in your form. In the end of the string attach the Mackey, that you generated under settings, and make an md5 hash of the entire string.

Fields you don't want to include in the hash is "checkmd5", "cardnum", "eyear", "emonth" and "cvc".

This pseudo-code shows the basic procedure of how to making the md5 hash.

```
mackey =
'67ce19247bd5765002e02db2c4cc7f81a1a7da416c29b9c3bacc9ab243c7e0564b0
c5e63f6ea0fb12a71bad8a9c564e1ea53fb9c4a23eb6bffeecb2cb4f9d03d'

string  = concatenate(shopid,
                      orderid,
                      orderidprefix,
                      currency,
                      amount,
                      mackey)

md5hash = md5(string)

# Insert md5hash value into the mac field value
<input type="hidden" name="mac" value="md5 hash value">
```

Almost the same procedure goes for the callbackurl. When Mac is enabled, a new GET variable named mac is attached to the callbackurl. The value of this GET variable is an md5 hash of every GET variables that is returned to the callbackurl including the secret key. To calculate an md5 hash, you need to concatenate the values of every returned GET variables, and attach the mackey to the end of the string. Then make an md5 hash and match it against the returned mac hash. If the hashes isn't identical, the data is tampered or you did something wrong when calculating the md5 hash.

Using MAC with paymentwindow

When using MAC together with paymentwindow, remember to add 2 extra fields to the MAC hash. If your not using authtype in your postform, paymentwindow will as default add authtype with value "creditcard. You then need to add "creditcard" and "true" at the end of the has string, before the mac key.