SEPAY Wireless Electronic Cash Register (WECR) Specifications

July 16, 2021

Version: 2.0

Status: DRAFT

Author Erik van Loenen
SEPAY B.V.

Version history

0.1 October 4, 2019

First draft

0.2 October 14, 2019

- Added tables with error codes, status codes and request status values
- Field name transactionstatus changed to transactionerror in GetTransactionStatus response

0.3 January 21, 2020

1.0 Finalizing specs:

- Status 13 added
- In case of pending transaction, return transactionref of pending transaction (allowing GetTransactionStatus/CancelTransaction)
- Added indication which status is applicable per type of call

2.0 July 16, 2021

- Fix numbering version history above
- Return IP and port allowing start transaction trigger in case terminal is in same local network subnet as cash register
- Some documentation fixes
- Allow push URL for finished transaction notifications
- Added field with receipt information, allowing the cash register to embed the payment slip into a printed ticket

1 Introduction

This document describes the implementation of the cash register interface of Sepay payment terminals (Nexgo).

The implementation consists of a SOAP service to which cash register applications can connect and through which transactions can be sent to a terminal and the status of a transaction can be checked. Currently, this is only a ‘poll’ mechanism, a push mechanism after finishing a transaction can be implemented later.

Security is taken care of by:

- Requiring an API login
- Include signature in each request/response
- SSL data transport

The authentication of requests consists of 3 levels:

1. Certificate
2. Login
3. Sepay terminal ID (SID)

1.1 Certificates

For authentication of the requests, SEPAY uses a private/public certificate pair. These certificates are used for verification/generation of a signature. Each message that is sent from the WECR system has a signature that is generated using the SEPAY private key based on concatenation all fields in the call (except for the signature field itself) and separating them with a semicolon (;). An example of this ‘string to sign’ will be given for each of the messages sent by the SEPAY WECR system when they are described further down in this document. This signature can be verified on receipt, by reproducing this concatenated string and use the SEPAY public certificate (provided in Appendix A) to verify the signature string passed in the message.

For message sent from the cash register to the SEPAY system, the inverse needs to happen. For this, the implementor of the WECR interface needs to generate a public/private key pair (SHA256/RS2048). The private key should be used to generate the signature for each message sent to the SEPAY WECR system. Upon setup of the provider, a public certificate of this pair should be provided to SEPAY, along with the ‘recognizable name’, so that SEPAY can load this into their system and by that enable it for use by clients/merchants.

It is recommended to give the generated certificate a long lifetime of 30 years for example, to avoid having to change keys often (or at all).

The most common setup is that each supplier of an implementation of WECR to end clients has one certificate that can be used for all clients. Of course, it is possible to have multiple certificates if a supplier wants to separate certain groups. Another reason for using multiple certificates is because Sepay provides a mechanism for clients on the ‘My SEPAY page, which allows a client to select the WECR provider and then provide a login (see further) and select the terminals for which to activate the WECR interface. Each party that implements WECR needs to send a certificate to SEPAY, accompanied with a name that a client would recognize when they want to use a SEPAY terminal in combination with WECR.

1.2 Login

End clients/merchants can, after selecting the WECR provider, define a login for the connection. This login, usually but not necessarily the email address of the client on their My SEPAY page, is then associated with the certificate on the SEPAY system. This login also needs to be known to the cash register, since this is a field in each message that is sent.

Usually, each client uses one login for all their terminals, but if the client so desires, they can group terminals into multiple logins if they have more shop locations for example.

1.3 SEPAY terminal ID (SID)

The final level of authentication is the SEPAY terminal ID, which identifies one specific terminal of a merchant. This is also needed in the cash register, because this ultimately determines for which terminal a transaction is intended to be done

So, for each message that is sent to the SEPAY WECR system, these 3 things must match, meaning the client has setup the relation between the certificate (WECDR supplier), associated login and terminals. Failure of any of these three levels, will result in a 'signature failed' error upon receipt of a message.

1.4 Certification

SEPAY does not have a formal certification process. When suppliers are interested in implementing the WECR interface, depending on the potential of the supplier, the supplier can get a temporary terminal from SEPAY, with which the implementation can be developed. After receiving the public certificate of a new supplier, SEPAY will load this into their system, but this will not be usable for other clients, until the supplier has done successful tests. When the supplier has successfully implemented their end of the implementation, SEPAY will investigate the transactions and if all is well, will make the certificate/provider public and selectable by clients/merchants.
Possibly, SEPAY will setup a more formal procedure with test scenario's, but at this time we assess this on a one by one basis before making an implementation publicly available.

1.5 High level functionality overview

The basic functionality currently consists of the following API calls:

1. StartTransaction, which starts a transaction with the given amount
2. GetTransactionStatus, which allows inquiry of an earlier transaction
3. CancelTransaction, which cancels the last transaction started

1.6 Versioning

The 'version' field in the specifications is used to distinguish what is implemented and allows the system to determine if certain features are implemented or not. The previous/current implementations are all based on version 1. Any value smaller than 2, including empty, will be interpreted as version 1.

Starting this version (2 or 2.0), there are parts that are new. Where the data if only applicable for a new/certain version, this is indicated in the specs of the messages.

2 Definitions

2.1 Data types

Data type	Description	Example
Int	Integer	12
String	String	Any text
Boolean	True or False	True
Date	ISO 8601 date string YYYY-MM-DD	2019-10-04
DateTime	ISO 8601 datetime string YYYY-MM-DD-HH:mm:SS	2019-10-04 11:02
Money	Money values, formatted as #0.00	1.23 0.10
Url	This represents a url string	https://a.b.nl/SepayCB/?id=1
A[x..y]	Alphanumeric string with length of x to y	Alphanumeric string
N(x..y)	Numeric string with length of x to y	0012
BASE64	String used to pass binary data. The binary should be converted to base64 standard	C4eO17ACQuTX

2.2 Transactionerror codes

These are error codes providing details about failed transactions.

Code	Description
0	Transaction succeeded
100	Do not honour
101	Expired card
104	Restricted card
105	Check security settings of acquirer
106	Allowable PIN tries exceeded
107	Refer to Card Issuer
109	Invalid Merchant
110	Invalid Amount
111	Invalid Card Number
116	Not sufficient funds
117	Incorrect PIN
119	Transaction not permitted to the customer
120	Transaction not permitted to the POS device
121	Exceeds withdrawal amount limit
123	Exceeds withdrawal frequency limit
128	PIN key synch error
141	Refund declined
181	Card blocked
185	Product(s) not allowed
191	Unknown transponder
200	Pick-up card Declined
2004	Check ACQ
202	Suspected fraud
2033	Check ACQ
204	Restricted card Declined – Capture
2061	Check ACQ
2075	Check ACQ
208	Lost Card Declined – Capture
209	Stolen Card Declined – Capture
902	Invalid transaction Failed
904	Format error
907	Card issuer or switch inoperative
908	Destination not found Declined
909	System Malfunction Declined

911	Card issuer timed out Declined
912	Card issuer unavailable Declined
913	Technical failure
917	MAC key synch error Declined

2.3 Transactionresult codes

These are high level result codes, the error code in the previous table can provide more details about the reason the transaction failed.

Code	Description
0	Transaction succeeded
1803	Time-Out
1804	Transaction declined
1811	Technical failure
1822	Connection failure
1823	Invalid answer
2621	Canceled on PIN entry
2622	Time-out on PIN entry
2623	Declined by card
2625	Corrupted response
2627	Declined by host
2629	Cancellation
3313	MAC verification failure
4021	Declined by card/terminal
4352	Declined by card/terminal

2.4 Status codes

These are the status codes of the API request. The columns to the right indicate if the code is applicable for the call:

1. StartTransaction
2. GetTransac tionStatus
3. CancelTransaction

Code	Description	1	2	3	4
0	OK	V	V	V	V
1	Some of the required fields are missing	V	V	V	V
2	Signature is invalid	V	V	V	V
4	Invalid parameters (invalid amount or, more likely invalid transactionref	V	V	V	
6	Duplicate request	V			
7	Terminal not active or not enabled and/or authorized for transactions through WECR	V	V	V	V
11	Pending request for this terminal. A new transactions can only be submitted after finishing or cancelling the previous transaction. When returned, the transactionref in the response is the reference of the pending transaction	V			
13	Transaction failed		V		
14	This transaction was canceled. The message in the response reports the reason it was canceled: - Canceled on terminal (STOP pressed) - Timeout on terminal - Canceled by WECR - Transaction expired (transactions are valid for 1 hour after StartTransaction)		V		
15	Returned when the transaction is not finished and has not been canceled yet		V		
17	Transaction already in progress, cannot be canceled anymore		V	V	
99	Undefined error	V	V	V	V

2.5 Version dependencies

A version column in the specs below is used to indicate from which version the field will be included and used. This also implies that if the version number in the version field is lower, this field is not supported and not included in the string to be used for the signature. If no version is indicated, the field will be applicable in all versions.

2.6 Signature

The signature value is created by concatenating all fields in the request/response, except the signature field itself, and separating them with a semicolon (;). Fields that have a version indication in the 'Version' column are only included in the signature if the version in the request/response is greater than or equal to the version mentioned. Fields without version indication in this column are always included.

3 API calls

3.1 StartTransaction request

A new field was added to report the 'brand' used, which in currently/typically would contain one of the following values:

- Maestro
- VPay
- Mastercard
- VISA
- Amex

One more field that is new is a field containing the information for a receipt. This is in a special format that has escape characters defining how it should be printed. See Appendix B for detail.

This call is used to start a transaction for a given amount. The request has parameters as described in the table below.

Fieldname	Version	Type	Mandatory	Description
key_index		Int	Yes	Identifier of private key used for signature. If not applicable, use 0
version		String	Yes	Version of the protocol used (in case of future changes requiring version dependent handling). See 2.5 for version dependencies
login		String	Yes	Identifies the client and thereby identifies the terminals that belong to this client for which transactions can be sent. The 'MySepay' user account should be used for this, which must have the appropriate rights for the WECR interface. Each account is assigned a private key for encrypting the signature when setting up WECR connectivity with Sepay.
sid		N[7..7]	Yes	This is the Sepay ID that identifies the terminal to send the transaction to
transactionref		AN[1..255]	Yes	Unique identifier for this transaction
merchantref		A[1..12]	No	Optional reference that will be associated with the transaction and which will be available on the My Sepay transaction overview and exports of transactions
amount		Money	Yes	Amount of the transaction
callback_url	2	Url	No	If provided, the SEPAY system will call this URL to notify the transactions has finished. The data that is posted, will be the same as the response from a GetTransaction. This mechanism can be used instead of polling and when possible is preferred to polling
signature		BASE64	Yes	SHA-256 hash for all properties in the command, separated by a semicolon (excluding this signature property), signed with the private key or pre-agreed customer key.

3.2 StartTransaction response

The response to the StartTransaction request is described in the table below.

Fieldname	Version	Type	Mandatory	Description
key_index		Int	Yes	Copied from the request
version		String	Yes	Copied from the request
login		String	Yes	Copied from the request
sid		N[7..7]	Yes	Copied from the request
transactionref		AN[1..255]	Yes	Copied from the request or transactionref of pending transaction in case the status = 11 (pending transaction)
merchantref		A[1..12]	No	Copied from the request
amount		Money	Yes	Copied from the request
status		N[2..2]	Yes	Status of the request (See Status codes)
message		String	No	Textual details about the status if available
terminalip	2	IP Address	Conditional	The local IP address of the terminal that can be used for triggering the transaction to be picked up by the terminal. This can only work if the terminal is in the same subnet on the local LAN as the cash register
Terminalport	2	[N1..5]	Conditional	The port number on which to trigger the terminal (usually 1234)
signature		BASE64	Yes	SHA-256 hash for all properties in the command (excluding this signature property), signed with the private key or pre-agreed customer key

3.3 GetTransactionStatus request

This call is used to request the status of a previously submitted transaction. The request has parameters as described in the table below.

Fieldname	Version	Type	Mandatory	Description
key_index		Int	Yes	Identifier of private key used for signature. If not applicable, use 0
version		String	Yes	Version of the protocol used (in case of future changes requiring version dependent handling)
login		String	Yes	Identifies the client and thereby identifies the terminals that belong to this client for which transactions can be sent. The 'MySepay' user account should be used for this, which must have the appropriate rights for the WECR interface. Each account is assigned a private key for encrypting the signature when setting up WECR connectivity with Sepay.
sid		N[7..7]	Yes	This is the Sepay ID that identifies the terminal to send the transaction to
transactionref		AN[1..255]	Yes	Unique identifier for the transaction the status is requested for
timeout	2	Int	Yes	Number of seconds the request may wait for the transaction to finish on the terminal before returning the result. This allows the minimization of the number of polls. Values from 5 to 30 seconds are recommended, depending on the level of blocking on the client side.
signature		BASE64	Yes	SHA-256 hash for all properties in the command (excluding this signature property), signed with the private key or pre-agreed customer key

3.3.1 response

The response to the GetTransactionStatus request is described in the table below.

Fieldname	Version	Type	Mandatory	Description
key_index		Int	Yes	Copied from the request
version		String	Yes	Copied from the request
login		String	Yes	Copied from the request
sid		N[7..7]	Yes	Copied from the request
transactionref		AN[1..255]	Yes	Copied from the request
merchantref		A[1..12]	No	Merchant reference from the original transaction
amount		Money	Yes	The amount of the transaction
transactiontime		Datetime	Cond	If the transaction has taken place, this field contains the date and time it took place
transactionerror		Int	Cond	If the transaction has taken place, this field contains the status of the transaction (See Transactionerror codes)
transactionresult		Int	Cond	If the transaction has taken place, this field contains the result of the transaction (See Transactionresult codes)
status		Int	Yes	Status of the request (See Status codes)
message		String	No	Textual details about the status if available
brand	2	String	No	Provides the brand used to pay, like Maestro, VPAY, Mastercard, Payconiq
ticket	2	String	No	When the transaction was successful, this contains a string representing the ticket for the transaction. This ticket is presented in a special 'print format'. See
signature		BASE64	Yes	SHA-256 hash for all properties in the command (excluding this signature property), signed with the private key or pre-agreed customer key

3.4 CancelTransaction request

This call is used to cancel a previously submitted transaction. This is only possible as long as the transaction has not been picked up by the terminal, once processing has started on the terminal there is no way to 'cancel' it remotely. Only a cancel on the physical terminal would be possible then, resulting in a failed transaction with the appropriate status indicating the reason. The request has parameters as described in the table below.

Fieldname	Version	Type	Mandatory	Description
key_index		Int	Yes	Identifier of private key used for signature. If not applicable, use 0
version		String	Yes	Version of the protocol used (in case of future changes requiring version dependent handling)
login		String	Yes	Identifies the client and thereby identifies the terminals that belong to this client for which transactions can be sent. The 'MySepay' user account should be used for this, which must have the appropriate rights for the WECR interface. Each account is assigned a private key for encrypting the signature when setting up WECR connectivity with Sepay.
sid		N[7..7]	Yes	This is the Sepay ID that identifies the terminal to send the transaction to
transactionref		AN[1..255]	Yes	Unique identifier for the transaction that should be canceled
force	2	Boolean	No	If this is set to true, the transaction status is canceled, even though it can still be in progress on the terminal. This can be useful in case of communication issues cause the transaction to 'hang'. Set this to True only when absolutely sure the transaction is not active on the terminal anymore and the terminal is in idle state. The default is False.
signature		BASE64	Yes	SHA-256 hash for all properties in the command (excluding this signature property), signed with the private key or pre-agreed customer key

3.5 CancelTransaction response

The response to the CancelTransaction request is described in the table below.

Fieldname	Type	Mandatory	Description
key_index	Int	Yes	Copied from the request
version	String	Yes	Copied from the request
login	String	Yes	Copied from the request
sid	N[7..7]	Yes	Copied from the request
transactionref	AN[1..255]	Yes	Copied from the request
status	N[2..2]	Yes	Status of the request (See Status codes)
message	String	No	Textual details about the status if available
signature	BASE64	Yes	SHA-256 hash for all properties in the command (excluding this signature property), signed with the private key or pre-agreed customer key

Appendix A

This is the public certificate that should be used to verify the signature in each message sent from the SEPAY WECR system.

```
-----BEGIN CERTIFICATE-----
MIIDdDCCAlygAwIBAgIBATANBgkqhkiG9w0BAQsFADB0MQswCQYDVQQGEwJOTDEL
MAkGA1UECBMCWkgxDjAMBgNVBAoTBVNFUEFZMQswCQYDVQQLEwJJVDEfMB0GA1UE
AxMWd2Vjci5zZXJ2aWNlcy5zZXBheS5ubDEaMBgGCSqGSIb3DQEJARYLaXRAc2Vw
YXkubmwwIBcNMjAwMTA4MDgyMjAwWhgPMjA1MDAxMDgwODIyMDBaMHQxCzAJBgNV
BAYTAk5MMQswCQYDVQQIEwJaSDEOMAwGA1UEChMFU0VQQVkxCzAJBgNVBAsTAklU
MR8wHQYDVQQDExZ3ZWNyLnNlcnZpY2VzLnNlcGF5Lm5sMRowGAYJKoZIhvcNAQkB
FgtpdEBzZXBheS5ubDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAOaK
wct1tB4nfTYfIppFAYUmb+kfvNzvmmd2Woui+YajYlHVWhLeTXl2H/DEUuY1/4VT
ku1UTsOi0uzjTJJYryqdKqkPCtvThsOUQ0JNjbbXZM5kEdcddeJMvULnSS/4dWVF
/lsv0vn2uxNysuprq7VEXmCSENOWMtg5r8D20PeAYEh/U+VavZ5FaNjIEqlKyjYt
kGsI2ABgUSJnrlZhDRq5Cb9AVgjRzu42JvTkaky9sjHP1569IqOiThlu6xEfACzg
ZaYnsIRh6ajXghzpBzbiXqVGhjvXUQbaZ+FV4JwQVsxfTokYl1E8MWVfAYWbO0FF
OYCUQR1AilacXTOl6JcCAwEAAaMPMA0wCwYDVR0PBAQDAgeAMA0GCSqGSIb3DQEB
CwUAA4IBAQAVXH65ARMMkBLrHQ1UjuWkuAXSI2pulqEe/L9U00f01ExlbAonxQg2
NJCJSd98jQZsD9mYBBQc5XfcC2y/nW81zDxyPkpd7vzN3JK02WP+MVgZI2wwH7h+
Nn454/JM8hI7Eye3aQWHMqWVD0bRJFz0vy8ELahxYfWzVg3DbdxAkhx++XasC/e5
sSmwDVag1KHt5qN4dH0MKLf8DnzldvIJuUREQpxmuIHyYlB/+4+CMraE5KZQqnEe
Jm+9FrcK8HMl7Bt7U3rAk68IhE4+mdlsI9+wHBEHksqsPqcjb0tTnIzBujPCcYCJ
CiomXPrKbr6nkxHM1EMuP47kCBf/LsQP
-----END CERTIFICATE-----
```

Appendix B

The string representing the payment ticket uses the following escape sequences that define the way it should be represented on paper:

@RS Reset printing definitions to default (small font and positions printing to the first column at left margin)
@LF Line feed causes that next character is printed in the new line in the first column at the left margin
@SS Selects small font size
@SM Selects medium font size
@SL Selects large font size
@HT Positions printing of next character to the next tabulator, where tabulators are defined in every 8th column starting from the left margin
@AR Align printing to right
@AM Align printing to middle
@@ Prints @ character