iDIN Merchant Implementation Guidelines

Document version 1.041

CONFIDENTIAL

January 2016

Terms and conditions

Terms and conditions for provision of the iDIN Merchant Implementation Guidelines:

- Currence Services BV (also referred to as 'Currence') provides these iDIN Merchant Implementation Guidelines to Merchant Banks that distribute it to (potential) Merchants and Digital Identity Service Providers (DISPs) to enable them to implement iDIN;
- Currence reserves the right to deny access to the iDIN Merchant Implementation Guidelines to (potential) Merchants and DISPs on reasonable grounds, in consultation with the Merchant Bank with which the Merchant/DISP has a contract;
- These Implementation Guidelines are explicitly and exclusively provided for the purpose mentioned above, and no other use is permitted. No rights can be derived from the information provided in this document or the accompanying notes. Currence is in no way liable for the consequences of later changes to the iDIN Standards or the iDIN Merchant Implementation Guidelines. If banks or other interested parties take decisions and/or make investments on the basis of the information that they obtain via the iDIN Merchant Implementation Guidelines, Currence accepts no liability for this in any way;
- These implementation Guidelines are based on the information in the iDIN Standards documents. In the event of any discrepancy between the iDIN Merchant Implementation Guidelines and the iDIN Standards documents, the text in the iDIN Standards documents prevails.

For any questions concerning this document or requests for further information, please contact your Bank or DISP.

Contents

Terms and conditions **2**

Contents **3**

Tables **8**

Figures **10**

1 General Introduction **11**

1.1 Target audience 11

1.2 Document structure 11

1.3 Other references 11

1.4 Notational conventions 12

1.5 Definitions of Internet and online banking 12

1.6 Revisions 13

1.7 Changes from previous version 13

1.7.1 Changes compared to version 1.00 13

1.7.2 Changes compared to version 1.04 13

2 Overview **14**

2.1 The four-corner model 14

2.1.1 The relations between these roles 14

2.1.2 Exceptions for DISPs 15

2.2 iDIN services 16

2.3 iDIN process 16

2.4 iDIN transaction 16

2.4.1 Delegated authentication 16

2.4.2 Providing consumer attributes 17

2.5 Merchant registration 17

2.6 Dispute management 18

3 iDIN protocols 19
3.1 Basic technical principles 19
3.2 iDx protocols 19
3.2.1 Directory protocol 20
3.2.2 Transaction protocol 20
3.2.3 Status protocol 21
3.2.4 Error protocol 21
3.3 SAML V2.0 21

4 iDIN Message format 22
4.1 General 22
4.2 Character set 22
4.3 HTTP 22
4.4 XML header 22
4.5 XML namespaces 23
4.6 XML Schemas 23

5 iDIN data dictionary 25
5.1 iDx attributes 25
5.2 iDx data elements 25
5.3 iDIN data elements 28
5.3.1 iDIN Requested- and DeliveredServiceID 30
5.4 Consumer attributes 31
5.5 Guaranteed minimal set of requested attributes 33

6 iDIN Directory protocol 35
6.1 General 35
6.2 Directory Request (DirectoryReq) 35
6.3 Directory Response (DirectoryRes) 36
6.4 Presentation of the Issuer Bank selection list 36

7 iDIN Transaction protocol 38
7.1 General 38

7.2 Transaction Request (AcquirerTrxReq) 38
7.3 Transaction Response (AcquirerTrxRes) 40
7.4 Errors when executing Transaction Protocol 40
7.5 Redirect to the online banking environment 41
7.5.1 Specific requirement iDIN Mobile: Redirect to Issuer (no in-app browser) 41
7.6 Redirect to the Merchant environment 41
7.6.1 Requirements for iDIN Mobile: redirect to the Merchant environment. 42
7.7 Errors during execution of the redirect to the Issuer, approving the iDIN request and/or the redirect to the Merchant environment 42
7.8 Four different scenarios for completion of iDIN Mobile transaction 43
7.8.1 Consumer is redirected from the Merchant's (mobile) web page to the Issuer's (mobile) web page. 43
7.8.2 Consumer is redirected from the Merchant's (mobile) web page to the Issuer's mobile banking app 44
7.8.3 Consumer is redirected from the Merchant's mobile app to the Issuer's (mobile) web page 45
7.8.4 Consumer is redirected from the Merchant's mobile app to the Issuer's mobile banking app 46
7.9 Performance and time-out of transaction message 47

8 iDIN Status protocol 48
8.1 General 48
8.2 Status Request (AcquirerStatusReq) 48
8.3 Status Response (AcquirerStatusRes) 49
8.4 Errors during execution of Status Protocol 51
8.5 Restrictions on AcquirerStatusReq 51
8.6 Performance and time-out of status messages 51

9 Error handling 52
9.1 General 52
9.2 Error Response (AcquirerErrorRes) 52
9.3 Non-availability 53

10 Security and certificates 54
10.1 General principles of certificates 54
10.2 Signing iDIN messages 54
10.2.1 Signing of the SAML 2.0 Assertion 55
10.3 SAML EncryptedID and EncryptedAttribute 56
10.4 Authentication of iDIN messages 57
10.5 Creating a key pair 58
10.5.1 Buying a certificate from a Certificate Authority 58
10.6 Signature data elements 58

11 Presentation of iDIN 61
11.1 General 61
11.2 Transaction flow 61
11.3 Redirect to Issuer 61
11.4 Frames 61
11.5 New Window 61
11.5.1 Specific Requirements iDIN Mobile: New window or app 62
11.6 Issuer list 62
11.7 Banners and logo's 62
11.8 Requirements and recommendation for Merchant screens 62
11.8.1 Display last login 62
11.8.2 Explaining iDIN to Consumers 62
11.8.3 Recommended texts on Merchant website per RequestedServiceID 63
11.9 Validation Service front-end 64
11.9.1 Approval information 65

12 APPENDIX A: Error codes and cases 68
12.1 iDx error codes 68
12.2 SAML error codes 69
12.3 SAML error cases 70
12.4 Issuer cannot provide all attributes conform minimal set 71
12.5 Consumer message 71

13 APPENDIX B: Requested- and DeliveredServiceID values 73

14 APPENDIX C: Message examples 75

14.1 DirectoryReq (A) 75

14.2 DirectoryRes (A') 76

14.3 AcquirerTrxReq (B) 77

14.4 AcquirerTrxRes (B') 78

14.5 AcquirerStatusReq (F) 79

14.6 AcquirerStatusRes (F') Unencrypted 80

14.7 AcquirerStatusRes (F') Encrypted 83

14.8 AcquirerErrorRes (B'(X)) 87

15 Appendix D: iDx Merchant-Acquirer Schema 89

Tables

Table 1: References 12
Table 2: Revision table 13
Table 3: Message names and description 19
Table 4: HTTP header 22
Table 5: XML header 23
Table 6: iDx namespaces 23
Table 7: XML schemas 24
Table 8: iDx attributes 25
Table 9: iDx data elements 28
Table 10: iDIN data elements in SAML message 29
Table 11: Overview of `Requested`- and `DeliveredServiceID` 31
Table 12: Consumer attributes 33
Table 13: Minimal set of attributes provided by Issuer per requested attribute group 34
Table 14: Elements/attributes of the DirectoryReq 35
Table 15: Elements/attributes of the DirectoryRes 36
Table 16: Elements/attributes of AcquirerTrxReq 39
Table 17: Elements/attributes inside the container of the AcquirerTrxReq 40
Table 18: Elements/attributes of AcquirerTrxRes 40
Table 19: Different scenarios for the completion of an iDIN mobile transaction 43
Table 20: Scenario: Redirect from Merchant (mobile) web page to the Issuers (mobile) web page 44
Table 21: Scenario: Redirect form Merchant (mobile) web page to the Issuers mobile banking app 45
Table 22: Scenario: Redirect from the Merchants mobile app to the Issuers (mobile) web page 46
Table 23: Scenario: Redirect from the Merchants mobile app to the Issuers mobile banking app 47
Table 24: Performance requirements (for the 95th percentile) 47
Table 25: Elements/attributes of AcquirerStatusReq 48
Table 26: Elements/attributes of AcquirerStatusRes 49
Table 27: Elements/attributes inside the container of AcquirerStatusRes 50
Table 28: Performance requirements (for the 95th percentile) 51
Table 29: Elements/attributes of the AcquirerErrorRes 52
Table 30: Elements/attributes inside the container of AcquirerErrorRes 53

Table 31: Elements/attributes of the Signature 59
Table 32: Signature changes with respect to signing the SAML Assertion 60
Table 33: Recommendation Merchant texts per use case 63
Table 34: Error code categories 68
Table 35: Error codes 69
Table 36: errorDetail 69
Table 37: First level SAML status codes 69
Table 38: Second level SAML status codes 70
Table 39: Consumer messages 72
Table 40: Integer values of `Requested`- or `DeliveredServiceID` per requested attribute group 74

Figures

Figure 1: Four-corner model 15

Figure 2: Transaction, Status and Error protocol 20

Figure 3: `Requested`- and `DeliveredServiceID` layout 30

Figure 4: Example of (open) dropdown list box showing the Issuer list 37

Figure 5: Example of transaction approval or mobile app screen 67

1 General Introduction

1.1 Target audience

This document is intended for Merchants that want to connect to iDIN via their Bank. It provides a detailed description of all messages that are exchanged between the Merchant and the Routing Service of his Bank. The messages that are exchanged between the Routing Service of the Merchant's Bank and the Validation Service of the Consumer's Bank are not of importance to the Merchant, and therefore will not be discussed in this document unless they have specific relevance.

This document is not bank specific, which means that only generic specifications are mentioned in this guide. Information concerning bank specific connections that are not standard and assistance that is provided by banks to connect to iDIN are not part of this guide. Please contact your bank for any information or support on a bank specific connection or implementation.

To further support Merchants, software libraries have been developed in .NET, PHP and Java. Please contact your bank about this for more information.

1.2 Document structure

The document has the following structure:

- Chapter 2: Introduction to iDIN and all parties involved in the process;
- Chapter 3: Introduction to the various messages exchanged within the scope of iDIN and the overall structure of the exchanged messages;
- Chapter 4: Overall message formats;
- Chapter 5: Data dictionary: All parameters that are relevant to the Merchant within the iDIN environment;
- Chapter 6: Directory protocol: Providing a list of participating Issuer banks;
- Chapter 7: Transaction protocol: Requesting consumer authentication/attributes;
- Chapter 8: Status protocol: Receiving consumer authentication/attributes;
- Chapter 9: Error handling;
- Chapter 10: Security and certificates;
- Chapter 11: Presentation of iDIN on the Merchant's website.

1.3 Other references

Title	Version	Issued by
AES, FIPS 197/ SO/IEC 18033-3	-	FIPS/ISO
Base16, Base32, and Base64 Data Encodings http://www.ietf.org/rfc/rfc3548.txt	July 2003	Network Working Group
Base64 Content-Transfer-Encoding http://tools.ietf.org/html/rfc2045iDxsection-6.8	November 1996	Network Working Group
GUIDELINES ON ALGORITHMS USAGE AND KEY MANAGEMENT (EPC342-08)	Version 1.1 approved 23 February 2009	EPC
ISO 9362, 8901 Standard	2014	ISO

Title	Version	Issued by
Multilingual European Subset 2 (MES-2) Unicode.org http://www.utf8-chartable.de/unicode-utf8-table.pl	15 April 2000	CEN
NEN 1888_2002, NEN 5825_2002	2002	NEN
Open SSL Library http://www.openssl.org	March 2015	OpenSSL
Security Assertion Markup Language (SAML) Core	2.0	OASIS
TLS Protocol version 1.0 http://www.ietf.org/rfc/rfc2246.txt	1.0, January 1999	IETF
TLS Protocol version 1.1 http://www.ietf.org/rfc/rfc4346.txt	1.1, April 2006	IETF
TLS Protocol version 1.2 http://www.ietf.org/rfc/rfc5246.txt	1.2, August 2008	IETF
XML Signature Syntax and Processing (Second Edition), W3C Recommendation 10 June 2008 http://www.w3.org/TR/2008/REC-xmldsig-core-20080610/	Second edition, 10 June 2008	World Wide Web Consortium (W3C)
XML Encryption Syntax and Processing, W3C Recommendation 10 December 2002 http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/Overview.html#sec-EncryptedData	10 December 2002	World Wide Web Consortium (W3C)

Table 1: References

1.4 Notational conventions

Messages and redirects are printed like *this*, and data elements are printed like `this`.

1.5 Definitions of Internet and online banking

In this document there are many references to Merchant and Issuer websites or online environments. To facilitate mobile use of iDIN, these references must be supplemented with '..... or mobile website / mobile app' where appropriate. For every instance where Internet or online related-terminology is used, please interpret this as including the mobile channel. Where mobile use of iDIN leads to specific requirements for iDIN, this is indicated separately in the text.

1.6 Revisions

Version	Description	Release date
1.00	Initial version	11th of August 2015
1.04	1. Addition of error handling when the Issuer cannot provide the minimal set of Consumer attributes 2. Adding the possibility to seperately request the Consumer gender by the Merchant 3. Clarification on the format of DateTimestamp **Note**: No other versions have been published between v1.00 and v1.04. This is due to levelling of the version numbering with other documentation	23rd of October 2015
1.041	1. Added recommendations for Merchant screens 2. Update of the standard Consumer error messages 3. Corrected the format of TransactionID in the Data-dictionary. This ID has 16 digits. 4. Added some minor clarifications	13th of Janauray 2016

Table 2: Revision table

1.7 Changes from previous version

1.7.1 Changes compared to version 1.00

1. Addition of error handling when the Issuer cannot provide the minimal set of Consumer attributes: Addition of the elements `BankID.DeliverdServiceID` and the 2nd SAML status code in the AcquirerStatusRes that are both used to communicate to the Merchant whether the attributes are delivered conform the minimal set (see Section 5.5). This error handling is described in Section 12.4;
2. Adding the possibility to separately request the Consumer gender by the Merchant: Added the possibility to request the consumer gender separately (using the `RequestedServiceID`), instead of being part of the group name attributes. This has impact on the number of valid combinations that can be made with the `RequestedServiceID` as described in APPENDIX B: Requested- and DeliveredServiceID values;
3. Clarification of the use of DateTimestamp (for all fields where DateTimestamp is used, such as the createDateTimestamp): Merchants are allowed to use zero to three decimals after the seconds for DateTimestamp elements in the messages they send to their Routing Service. DateTimestamps in messages from the Routing Service will always contain three decimals behind the seconds. See Table 9 for more information.

1.7.2 Changes compared to Version 1.04

1. Added recommendations for Merchant screens: This is described in chapter 11.8;
2. Update of the standard Consumer error messages: This is described in chapter 12.5;
3. Format of TransactionID：In the Data Dictionary the `TransactionID` contained an error which has been corrected in this version. This ID consists of 16 digits; the first four digits are made up of the AcquirerID.

2 Overview

This chapter gives a description of the general elements of iDIN.

2.1 The four-corner model

The iDIN system is based on the 'four-corner' model. Figure 1 shows the roles in this model, along with their mutual primary relationships in the context of iDIN. The roles are those of Consumer, Merchant, Acquirer, Issuer, Routing Service and Validation Service and the Digital Identity Service Provider (DISP)[1]:

- The role of Consumer is fulfilled by a natural person, holding credentials provided by the Issuer;
- The role of Merchant must be fulfilled by a legal entity that wishes to identify its Consumers in an authentic manner;
- The role of Acquirer must be fulfilled by a legal entity, providing iDIN services to its Merchants;
- The role of Issuer must be fulfilled by a legal entity, providing digital identities and credentials to its Consumers;
- The role of Routing Service must be fulfilled by an Acquirer or by a third party endorsed and contracted by an Acquirer;
- The role of Validation service must be fulfilled by an Issuer or by a third party endorsed and contracted by an Issuer;
- The role of Digital Identity Service Provider (DISP) must be fulfilled by a third party, which is endorsed and contracted by the Acquirer as well as the Merchant.

2.1.1 The relations between these roles

Both contractual and technical relations exist between the roles. These are described below.

Contractual relations:

- Merchant - Acquirer: The Merchant has a contract with an Acquirer;
- Consumer - Issuer: The Consumer has a contract with the Issuer. The identity related to this contract is used for identification, authentication and provisioning of attributes of the Consumer;
- Consumer - Merchant: The Consumer may use iDIN services at the Merchant domain to access the service supplied by the Merchant;
- Acquirer - Issuer: Bound by a license agreement with the iDIN scheme.

Technical relations:

- Merchant - Routing Service: The Merchant has a technical relation with the Routing Service. The Routing Service offers the Merchant the possibility of sending iDIN requests to a Validation Service. They exchange messages to this end;

[1] Some of these terms map as follows to Identity Access Management terms: Consumer - Subject, Merchant - Relying Party/Service Provider, Acquirer / Routing Service - Broker and Issuer / Validation Service - Identity Provider.

- Routing Service – Validation Service: The Routing Service and Validation Service have an iDIN solution relationship. They exchange messages in this context;
- Consumer – Validation Service: The Validation Service offers the Consumer the possibility of using their iDIN towards Merchants, by using the credentials issued by the Issuer.

2.1.2 Exceptions for DISPs

In a model where a Merchant outsources its iDIN activities to a DISP[2] the contractual relation between Merchant - Acquirer is replaced by the following relations:

- Merchant - DISP: The Merchant has a contract with a DISP;
- DISP - Acquirer: The DISP has a contract with an Acquirer.

The technical relation between Merchant - Routing Service is replaced by the following relations:

- Merchant - DISP: The Merchant has a technical relation with the DISP. The DISP offers the Merchant the possibility of sending iDIN requests to a Routing and Validation Service. They exchange messages to this end;
- DISP - Routing Service: The DISP has a technical relation with the Routing Service. The Routing Service offers the DISP the possibility of sending iDIN requests to a Validation Service. They exchange messages to this end.

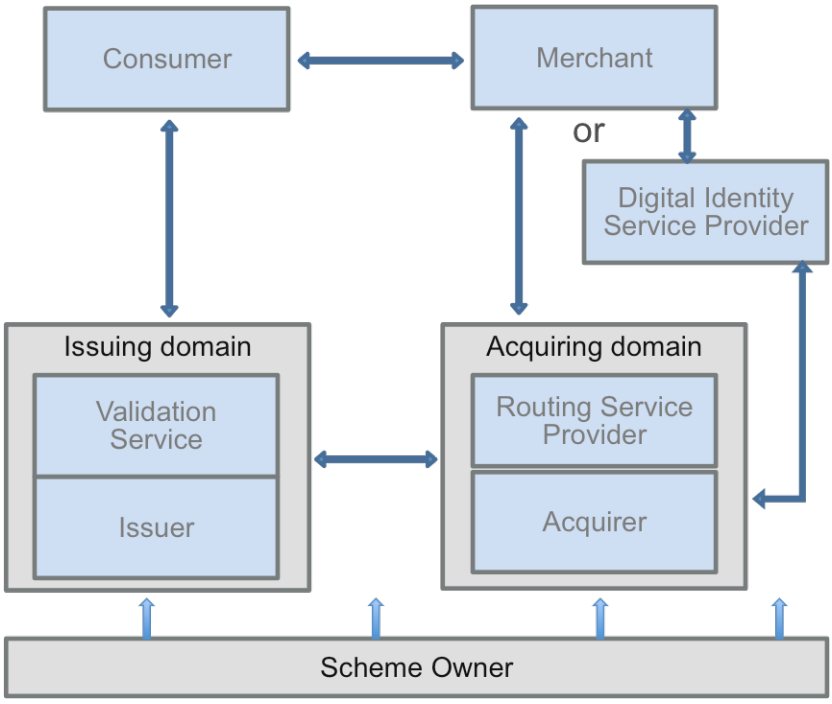

Figure 1: Four-corner model

[2] For due diligence reasons, the DISP may be subject to extra audits

2.2 iDIN services

This paragraph introduces the iDIN services offered to both Merchants and Consumers.

iDIN facilitates the following services:

- Identification and authentication of Consumers based on Issuer issued credentials;
- Provisioning of Consumer attributes from the Issuer administration including age verification.

Note: In this document the iDIN process of providing one of the above services is referred to as an iDIN transaction.

The services provided by iDIN can be used by the Merchant, amongst others, to:

- Set-up new user accounts for Consumers in the Merchant's domain based on the identification and/or attributes provided by the Issuer;
- Login of existing users that created their account with iDIN, or connected an existing account with iDIN;
- Age verification.

The following functionality is **currently** out of scope of the solution:

- More advanced age verification (currently only 18+);
- Document signing;
- Non-bank attribute providers;
- Actual implementation of the DISP role. DISPs are foreseen, but not yet supported.

2.3 iDIN process

At the Merchant website the Consumer initiates the process by selecting iDIN for authentication/provisioning of attributes, and subsequently selects an Issuer from the Issuer list. The Consumer will then be forwarded to his/her Issuer's secure environment where he/she can continue with the Issuer's authentication process (based on the credentials that were issued to the Consumer by the Issuer). After successfully authenticating at the Issuer, the Consumer can approve or cancel the iDIN transaction. The Consumer will then be redirected back to the Merchant website. The Merchant receives the requested authentication/attributes if the Consumer has approved the transaction.

2.4 iDIN transaction

The result of an iDIN transaction can be delegated authentication, provisioning of attributes (including age verification), or a combination of both.

2.4.1 Delegated authentication

In an iDIN transaction, the Merchant can request a unique, Merchant-specific persistent identifier (BIN) of the Consumer. This allows the Merchant to link the Consumer session to an already existing user account at the Merchant's system that contains the same BIN, or offer the option to create a new account for the Consumer when the BIN is not yet present in the Merchant's system. By approving the transaction, the Consumer consents to logging in at the Merchant's website.

2.4.2 Providing consumer attributes

In an iDIN transaction, the Merchant can request attributes of the Consumer. To this end, at the Issuer website the Consumer explicitly consents to provide the requested attributes to the Merchant. Attributes cannot be requested (and provided) individually, but only as bundled in one of the three[3] attribute groups that are specified as: name, address and age related. See Section 5.3.1 and Section 5.4 for more details. Consent is always given for all requested attributes as a whole, not per data set or per attribute. However, the Merchant is free to ask the Consumer on its website/application, before initiating the transaction, which attribute groups the Consumer would like to give to the Merchant.

Note: Due to possible movement of Consumers it is advised to Merchant to always check the correctness of the address. Also, when dealing with a delivery, the Consumer might want to deliver to another address then his/her residential address.

Consumer attributes can be requested on a one-time basis, without linking the Consumer session to previous or future sessions, or to an existing user account. In that case, the Merchant will request and receive a transient identifier (transient meaning temporary), generated by the issuer for this session. In case attribute provisioning is combined with delegated authentication, the Merchant requests and receives the persistent BIN.

The Merchant can easily set up a new account for the Consumer in the Merchant's domain based on the BIN and attributes provided by the Issuer.

Note that the consumer should always be aware when using iDIN to create a new Consumer account, in order to prevent the Consumer from connecting whilst already having another account at the Merchant. When connecting a Consumer to an existing account, simply matching consumer attributes with the entire database of the Merchant is an erroneous method, because certain Consumers can share the same attributes (e.g. first and last name). First letting the Consumer log in the Consumer to the existing account before connecting the account with iDIN, or connecting with additional unique Consumer information (not provided by Issuer e.g. email address), should counteract this Issue.

2.5 Merchant registration

When registering for iDIN, the Acquirer issues a `Merchant.MerchantID` and a `Merchant.LegalID` to the Merchant that is associated with the `Merchant.Name`. If required, the Merchant also receives a `Merchant.SubID` to register one ore more `Merchant.TradeName`. This is the case when either a Merchant performs iDIN transactions under different trade names, or in a similar manner, a DISP which performs iDIN transactions for different clients. Only the `Merchant.MerchantID` and `Merchant.SubID` are relevant to the Merchant, that is, the Merchant only has to use these to fulfil a

[3] Because the BIN is in another place in the SAML message, namely the subject, it is not considered an attribute in this context.

successful iDIN process. The other identifiers are allocated by the Acquirer and used within the Acquirer-Issuer and Issuer-Consumer domain.

2.6 Dispute management

In case of a dispute, the Issuer must be able to provide the identity of the Consumer. During such a dispute the following steps must be executed:

- The Merchant must supply the original SAML iDIN message that holds the Consumer identity to the Merchant's Acquirer. To ensure the Issuer can read the encrypted attributes, the Merchant must decrypt the AES keys with his private key. The AES keys have to be provided along with the original message;
- The Acquirer forwards this information to the Issuer, with contact information of the Merchant;
- The Issuer must use the certificate from the message to verify the electronic signature, thereby establishing the authenticity and integrity of the iDIN message. The Issuer must take into account the need for an application to be able to perform these checks;
- By using the AES keys provided by the Merchant, it can be easily verified whether these keys are in fact the correct keys;
- In case a message is authentic, the Issuer must provide Consumer details to the Merchant and/or Consumer directly. This can only be done if the Consumer has given his/her consent;
- Providing the details has to be done via telephone[4] where:
    - The Acquirer must provide the telephone number of the Merchant to the Issuer;
    - The Issuer must initiate the call to the Merchant/Consumer to prevent misuse.

[4] If the disputes are more common than expected, the process can be automated in the future.

3 iDIN protocols

This chapter gives a general description of the iDx protocols, which are used as a messaging standard for iDIN.

3.1 Basic technical principles

- All iDIN processes are initiated on the Merchant website by the Consumer;
- iDIN uses the iDx standards as a messaging standard. It uses the generic information container in the iDx protocol to embed SAML 2.0 messages for iDIN specific elements;
- Consumers will be persistently identified using a Bank Identification Number (BIN), see Section 5.3.1, which will be Issuer generated (so not shared among Issuers) and unique per Consumer-Merchant combination;
- A Consumer has only one identity per Issuer, regardless of the number of credentials;
- Consumer information in messages will be shielded from Acquirers (Routing Service) by means of end-to-end encryption.

3.2 iDx protocols

The iDIN process (request-/response-XML messaging and browser redirects) consists of the following protocols:

- The Directory protocol: used to retrieve the most recent Issuer list from the Routing Service.
- The Transaction protocol: the iDIN-transaction process from beginning to end.
- The Status protocol: used to request the status of a transaction from the Validation Service (via the Routing Service).
- The Error protocol: used to inform when errors have occurred.

A specific name (letter) has been assigned to identify each message. The following table applies:

Message	Message description
A	DirectoryReq (Directory Request)
A’	DirectoryRes (Directory Response)
B	AcquirerTrxReq (Transaction Request)
B’	AcquirerTrxRes (Transaction Response)
F	AcquirerStatusReq (Status Request)
F’	AcquirerStatusRes (Status Response)
A’(X) B’(X) F’(X)	AcquirerErrorRes (Error Response)
Redirects:	
D	Merchant redirects Consumer to Issuer
E	Issuer redirect Consumer to Merchant

Table 3: Message names and description

The sequence of the protocols is shown in Figure 2. In order to retrieve the services provided by iDIN both the Transaction and Status Protocol have to be completed. The Directory protocol is not indicated, because it follows a simple request and response from the Merchant to the Acquirer.

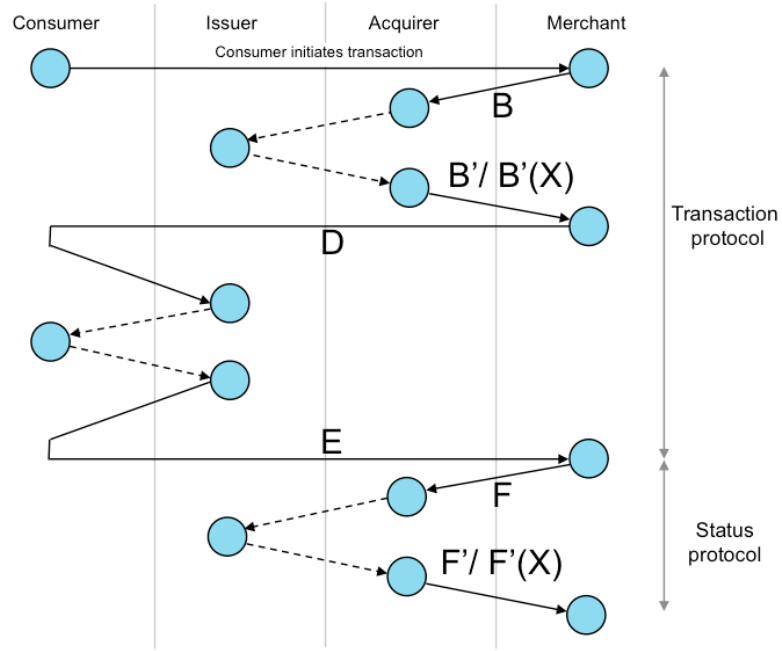

Figure 2: Transaction, Status and Error protocol

3.2.1 *Directory protocol*

By using the Directory protocol, the Merchant sends a DirectoryReq to the Routing Service. This is a request in XML format to obtain the list of participating Issuer Banks from the Routing Service. The Routing Service will provide this list to the Merchant by sending back the DirectoryRes. The Merchant will show the list of banks, which were sent in the DirectoryRes to the Merchant. The Consumer will choose his bank from this list at the beginning of the iDIN process. The Directory protocol is explained in more detail in Chapter 6.

3.2.2 *Transaction protocol*

By using the Transaction protocol the Merchant sends an AcquirerTrxReq to the Routing Service, containing the ID of the Issuer Bank chosen by the Consumer, iDIN information (requesting consumer authentication, attributes or age verification) and other transaction details. This message also contains the `merchantReturnURL`. This URL is used by the Validation Service to redirect the Consumer back to the Merchant's website when he has completed the iDIN transaction in the Issuer domain. After the Routing Service has received the message from the Merchant the message is send to the Validation Service of the Issuer that was selected by the Consumer (after adding some information which is beyond the scope of the Merchant).

In return, the Validation Service responds with the IssuerTrxRes message that contains amongst others the `issuerAuthenticationURL`. The Routing Service forwards this

`issuerAuthenticationURL` together with a unique `Transaction.TransactionID` back to the Merchant in an AcquirerTrxRes.

3.2.2.1 Redirect

The Merchant now redirects the Consumer to the `issuerAuthenticationURL`, which refers to the page of the online banking portal. This takes the Consumer to his internet-banking environment where he can continue the iDIN transaction (i.e. logging in and approve the transaction).
The Issuer adds Consumer information (consumer authentication, attributes or age verification) to the iDIN transaction which can be later retrieved with the AcquirerStatusReq. The Consumer approves the iDIN request and receives a confirmation from the Issuer. The Consumer is then redirected back to the website of the Merchant via the `merchantReturnURL`. The entire Transaction protocol and the 2 redirects are described in more detail in Chapter 7.

3.2.2.2 Specific requirement iDIN Mobile

The Mobile transaction flow is almost identical to the transaction flow in a regular iDIN transaction. The only difference is the redirect to a 'landing page' (using the `issuerAuthenticationURL`) where the Consumer, using a mobile device, can choose to be redirected to the Issuer's (mobile) web page or to the Issuer's mobile banking app (if available).

3.2.3 Status protocol

Finally the Merchant initiates the Status protocol by sending an AcquirerStatusReq message to the Routing Service. The Routing Service will request the transaction status from the appropriate Issuer and returns the status to the Merchant in the AcquirerStatusRes. If all steps in the transaction are successful this status message contains the iDIN consumer authentication, attributes and/or age verification. Chapter 8 contains more detailed information on the Status protocol.

3.2.4 Error protocol

Instead of a regular response to the messages mentioned above, it is also possible that an AcquirerErrorRes is returned. This can be the case if a request (DirectoryReq, AcquirerTrxReq or AcquirerStatusReq) contains an error, or if an error occurs during the processing of the request. The AcquirerErrorRes messages are discussed in Chapter 9.

3.3 SAML V2.0

iDIN uses the iDx standards as a messaging standard. However, it uses the generic information container in the iDx protocol to embed SAML 2.0 messages for iDIN specific elements. This container is **only used** in the AcquirerTrxReq, AcquirerStatusRes and sometimes in AcquirerErrorRes. For the other messages the container is left empty.

4 iDIN Message format

4.1 General

This chapter contains a description of the general message structure for the Directory, Transaction, Status, and Error protocol. The subsequent sections will describe the specific fields within the XML messages for each protocol in more detail. A list of the data elements and their format can be found in the data dictionary in Chapter 5.

The following conventions are used to indicate whether a message element is **mandatory**:

- Yes The element must occur exactly once;
- Yes (1..∞) The element must occur one or more (unlimited) times;
- No The element may be left out or may occur exactly once;
- No (0..∞) The element may be left out or may occur more (unlimited) times.

4.2 Character set

- In all iDIN messages the Unicode character set must be used. Only the MES-2 subset must be supported;
- Encoding must be used as indicated in the HTTP and XML headers UTF-8 (Unicode Transformation Format);
- The use of characters that are not part of the Unicode character set will not lead to a refusal of a post, but the character may be changed to a space, question mark or asterisk in transit;
- The Byte Order Mark (BOM) must not be used. The UTF-8 representation of the BOM is the byte sequence 0xEF,0xBB,0xBF.

4.3 HTTP

- All messages must comply with the HTTP 1.1 standard, as defined in RFC 2616 of W3C. For more information: http://www.w3.org/Protocols/rfc2616/rfc2616.html;
- Each XML request message must be sent as the body of a HTTP POST message;
- Each XML response message must be sent as the body of a HTTP 200 OK message.

The following HTTP header must be used for all messages:

Data element	Mandatory	Explanation
content-type	Yes	Defines how the remainder of the content is to be interpreted. **Must be:** text/xml; charset=”utf-8”

Table 4: HTTP header

4.4 XML header

The following XML header must be used for all messages:

Data element	Mandatory	Explanation
version	Yes	**Must be:** “1.0”

encoding	Yes	**Must be:** “UTF-8”

Table 5: XML header

4.5 XML namespaces

iDIN uses the iDx standards as a messaging standard. It uses the generic information container (an XML element) in the iDx protocol to embed SAML 2.0 messages for iDIN specific elements. In addition to the namespaces of the iDx messages, several namespaces are declared in the SAML message.

The namespaces for all messages described in this document is as follows:

Namespaces	Namespace declaration in example messages
Namespace for the iDIN iDx messages between the Merchant and Acquirer. **Must be:** “http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0”	xmlns
Namespace for the XML signature syntax. **Must be:** “http://www.w3.org/2000/09/xmldsig#”	xmlns:ds
Namespace for schema related mark-up. **Must be:** “http://www.w3.org/2001/XMLSchema-instance”	xmlns:xsi
Namespace for the SAML 2.0 Assertion which is embedded in the container element of the iDx message. **Must be:** “urn:oasis:names:tc:SAML:2.0:assertion”	xmlns:saml
Namespace for the SAML 2.0 Protocol which is embedded in the container element of the iDx message. **Must be:** “urn:oasis:names:tc:SAML:2.0:protocol”	xmlns:samlp
Namespace for the XML encryption syntax. This namespace is declared in the container element of the iDx message in, only in the AcquirerStatusReq message for the encrypted XML elements, see **Error! Reference source not found.**. **Must be:** “http://www.w3.org/2001/04/xmlenc#”	xmlns:xenc

Table 6: iDx namespaces

Namespace declaration can be done in any way allowed by the XML standards (default namespace declaration or namespace qualification/prefixes) with only a few exceptions as discussed in Section 10.3 where all namespace declaration must be present inside the encrypted parts of the message.

4.6 XML Schemas

All messages must be validated against the iDIN, SAML 2.0 and W3C Schemas. The schemas names and locations are shown in the following table:

XML schema	Explanation
idx.merchant-acquirer.1.0.xsd	Schema for the iDIN iDx messages between the Merchant and Acquirer. Provided by Acquirer to the Merchant after registration
xmldsig-core-schema.xsd	Schema for XML signatures. **Available at:** http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/xmldsig-core-schema.xsd#
saml-schema-assertion-2.0.xsd	Schema for the SAML 2.0 assertion which is embedded in the container element of the iDx message. **Available at:** http://docs.oasis-open.org/security/saml/v2.0/saml-schema-

XML schema	Explanation
	protocol-2.0.xsd
saml-schema-protocol-2.0.xsd	Schema for the SAML 2.0 protocol which is embedded in the container element of the iDx message. **Available at:** http://docs.oasis-open.org/security/saml/v2.0/saml-schema-assertion-2.0.xsd
xenc-schema.xsd	Schema for the XML encryption syntax. **Available at:** http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd

Table 7: XML schemas

5 iDIN data dictionary

This chapter describes the data elements and ID's that are used in iDIN. First, two attributes for all iDx messages are described in Section 5.1. All iDx data elements are described in Section 5.2. Subsequently, in Section 5.3 the iDIN specific data elements are described, which are either located in the SAML message, or differ from the normal iDx formatting. Section 5.4 provides a list of all consumer attributes.

5.1 iDx attributes

The following two attributes are given to the root of each message to ensure proper identification of the correct service and version:

Attribute	Description	Messages	Formatting rules
`version`	Indicate the version of the iDIN Service	ALL	**Must be:** "1.0.0"
`productID`	Indicates the service used with the iDx format	ALL	**Must be:** "NL:BVN:BankID:1.0"

Table 8: iDx attributes

5.2 iDx data elements

iDIN relies on the identifiers described in Table 9. If the element must be generated as an input in one of the three request messages (DirectoryReq, AcquirerTrxReq, AcquirerStatusReq) it is indicated in the description in **bold**.

Name	Description	Messages	Formatting rules
`Acquirer.AcquirerID`	Unique four-digit identifier of the Acquirer within an iDx based product, assigned by the product owner when registering the Acquirer.	A', B', F', F'(X)	4 digit-number
`Country.countryNames`	Contains the `countryNames` in the official languages of the country. Used in presentation of the Merchant to the Consumer on the website, see Section 6.4. In official languages of the country, separated by a '/' symbol (e.g. 'België/Belgique')	A'	Max 128 alphanumeric

Name	Description	Messages	Formatting rules
createDateTimeStamp	The time of creation of a particular request or response. Can also have the form directoryDateTimestamp, statusDateTimestamp and transactionCreateDateTime-stamp but follows the same formatting rules. It is indicated in the messages in Chapter 6-9 which particular time is meant **Generated by Merchant in Requests messages**	ALL	ISO 8601 UTC time format (no daylight saving) in format YYYY-MM-DDThh:mm:ss.sssZ e.g. 2015-10-13T14:41:12.123Z • Merchants are allowed to use zero to three decimals behind the seconds. DateTimestamp in messages from the Routing Service will always have three decimals after the seconds • YYYY is the calendar year • hh is 24-hour notation. 12-hour notation must not be used
Error. consumerMessage	An Acquirer can use this field to include a (standardized) message that the Merchant should show to the Consumer	A'(X), B'(X), F'(X)	See APPENDIX A: Error codes
Error.errorCode	Unique identification of an error occurring within the iDx transaction	A'(X), B'(X), F'(X)	See APPENDIX A: Error codes
Error.errorMessage	Descriptive text accompanying Error.errorCode	A'(X), B'(X), F'(X)	Max 128 alphanumeric See APPENDIX A: Error codes
Error.errorDetail	Details of the error. The message-generating party is free to determine this information	A'(X), B'(X), F'(X)	Max 256 alphanumeric
Error. suggestedAction	Suggestions aimed at resolving the problem. The message-generating party is free to determine this information	A'(X), B'(X), F'(X)	Max 512 alphanumeric
issuerAuthentication URL	Contains the issuer authentication URL to which the consumer is send	B'	Max 512 characters
Issuer.IssuerID	Unique identifier of the Issuer that consists of the international Bank Identifier Code (BIC).	A', B	ISO 9362
Issuer.Name	Only given in pairs together with the Issuer.IssuerID used in presentation of the Merchant to the Consumer on the website, see Section 6.4	A'	Max 35 alphanumeric

Name	Description	Messages	Formatting rules
language	This field enables the Issuer's website or mobile app to select the Consumer's preferred language (e.g. the language that was selected on the Merchant's website or mobile app), if the Issuer's website or mobile app supports this. In case of error, this field enables the Acquirer to select the Error.consumerMessage in the Consumer's preferred language (see Section 12.2). If a non-supported or non-existing language is entered, the standard language of the Issuer is used. e.g. 'en' for English, 'nl' for Dutch is used **Generated by Merchant**	B	ISO 639-1
Merchant.MerchantID	This is the contract number for iDIN. The Merchant obtains this ID after registration for iDIN.	A,B,F	10 numeric Made up of Acquirer.AcquirerID (first four positions) and a unique number of exactly six positions
Merchant.subID	The subID that uniquely defines the name and address of the Merchant to be used for the iDIN. The Merchant obtains the subID from its Acquirer after registration for iDIN. A Merchant can request permission from the Acquirer to use one or more subIDs. Unless agreed otherwise with the Acquirer, the Merchant has to use 0 (zero) as subID by default (if no subIDs are used).	A,B,F	Max 6 numeric Number from 0 to and including 999999 in which each value is related to a separate instance registered with the Acquirer. The default value is '0'
merchantReturnURL	URL to which the Consumer must be redirected after authentication and/or authorization of the transaction at the Issuer. The URL does not necessarily begins with http:// or https://, it can also start with an app handler e.g. companyname-nl-service://. The protocol part must always be included. The resource indicated by the URL must be the website or mobile app of the Merchant or a part thereof. **Generated by Merchant**	B	Max 512 characters
Merchant.X509	Certificate of the Merchant. Used in the request to express the Merchant's certificate needed for encryption by the Issuer (excluding the private key) Never included in the response and placed in the extension element of a SAML AuthnRequest	B	Base64 encoding

Name	Description	Messages	Formatting rules
`Transaction.entranceCode`	An 'authentication identifier' created by the Merchant to facilitate continuation of the session between a Merchant and a Consumer, even if the existing session has been lost (e.g. because the cookie expired). It enables the Merchant to recognize the Consumer associated with a (completed) transaction. The `entranceCode` is sent to the Merchant in the Redirect to Merchant. See Section 7.6 for more information on its use. **Generated by Merchant**	B	Max 40 alphanumeric A minimum variation of 1 million is required
`Transaction.TransactionID`	Unique 16-digit number within an iDx based product, assigned by an Acquirer. Received in the AcquirerTrxRes. Used to couple a AcquirerStatusReq to a particular response	B', F, F' F'(X)	16 numeric The first four digits of the `TransactionID` are made up of the `AcquirerID`
`Transaction.status`	Status of the transaction: related to whether the transaction has been authenticated/authorized by the Consumer	F'	Always has one of the following values: **Open**: Final status not yet known. This is the initial status value for all parties for all transactions. **Success**: the Consumer approved the transaction and the Issuer has confirmed this approval. **Cancelled**: the transaction has not been approved; cancelled by the Consumer. **Expired**: the transaction has not been approved within the `BankID.expirationPeriod` set by the Merchant or the default `BankID.expirationPeriod`. **Failure**: the transaction has not been approved; unknown reason

Table 9: iDx data elements

5.3 iDIN data elements

These elements are the iDIN specific data elements that are either located in the SAML 2.0 message, or, differ from the normal iDx standard. If the element must be generated as an input in one of the three request messages (DirectoryReq, AcquirerTrxReq, AcquirerStatusReq) it is indicated in the description in **bold**.

Name	Description	Messages	Formatting rules
`BankID.MerchantReference`	Unique transaction reference generated by and only for the Merchant (e.g. administration) **Generated by Merchant**	B, F', F'(X)	Max 35 text and must start with a letter (either lower- or upper-case)

Name	Description	Messages	Formatting rules
`BankID. expirationPeriod`	The time in which the consumer has to approve the transaction. Otherwise the status will be set to 'Expired'. Minimum is 60 seconds and maximum and default value is 300 seconds. **Generated by Merchant**	B	ISO 8601 e.g. PT300S or PT5M
`BankID.LOA`	iDIN provides two Levels of Assurance for authentication of Consumers. The exact requirements will be provided in a separate document or added to this document in a later stage. For a request: the minimum level of assurance required For a response**:** the actual level of assurance provided **Generated by Merchant**	B, F'	**Must be:** **"**nl:bvn:bankid:1.0:loa2" **or** **"**nl:bvn:bankid:1.0:loa3"
`BankID. RequestedServiceID`	The number that is used to ask for a particular set of consumer attributes, see Section 5.3.1 for more information. **Generated by Merchant**	B	Integer as specified in Section 5.3.1. See Appendix B for all combinations
`BankID. DeliveredServiceID`	Same structure as `RequestedServiceID`. Returned to the Merchant in the Response to indicate which requested attributes are delivered conform the minimal set (see Section 12.4). If the Issuer is unable to determine the `DeliveredServiceID` a value of '0' must be used. The `DeliveredServiceID` is included as unencrypted Attribute in the Assertion	F'	Integer as specified in Section 5.3.1. See Appendix B for all combinations
`consumer.bin`	BIN is short for Bank identifying Number. It identifies the Consumer. BINs are bank specific and are unique for every Consumer-Merchant-Bank combination	F'	BIN consists of two parts: 1) Prefix: Two-letter country code of the Issuer (ISO 3166-1) followed by four-letter (alphabetic) bank identifier (ISO 9362) 2) Bank specific identifier. Must be transportable as a string of max 1020 chars
`consumer.transientid`	Can be asked instead of the `Consumer.BIN`. Indicates that the content of the element is an identifier with transient semantics and **SHOULD** be treated as an opaque and temporary value by the Merchant	F'	*Max 256 characters and is prefixed with 'TRANS'.*
`Merchant.LegalID`	The `Merchant.LegalID` is the CreditorID of the Merchant. This is based on the number of the Chamber of Commerce. It is returned in the Status Response due to SAML compliance but has no particular use for the Merchant in completing an iDIN transaction flow	F'	-

Table 10: iDIN data elements in SAML message

5.3.1 *iDIN Requested- and DeliveredServiceID*

iDIN provides services that can be requested by the Merchant. Each transaction request can only supply one set of data, based on a single RequestedServiceID. The returned consumer attributes can differ per Issuer and request, therefore some of these elements that are part of a group are optional e.g. some consumers do not have a last name prefix. The `DeliveredServiceID` is used to indicate which attributes are delivered conform the minimal set as defined in Section 5.5. In most cases the `RequestedServiceID` shall be equal to the `DeliveredServiceID`. If the Issuer cannot provide all attributes conform the minimal set the `DeliveredServiceID` is not equal to the `RequestedServiceID`. This is discussed in more detail in Section 12.4.

The `Requested-` and `DeliveredServiceID` are short integers translated from a binary format. See Figure 3 for an overview of the structure of the binary format of both ServiceIDs. The binary numbers are separated into groups. These groups represent the categories of attributes the Merchant may request. The grey binary value is reserved space, which may be used in in a later stadium. Please be aware that this is a bit pattern instead of a number, and that the first bit is on the left not on the right.

Binary ServiceID =	Category	1		2		3		4				5		Reserved			
	Binary value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Bit number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Figure 3: `Requested-` and `DeliveredServiceID` layout

Table 11 provides an overview of which attributes are included and what value it should represent on binary level in order to receive the correct attributes, as defined in Section 5.4.

Nr.	Category	Bits	Description	Values
1.	ConsumerID	1..2	Indicates if ID attributes is requested / delivered.	00 Consumer.TransientID 01 Consumer.BIN
2.	Name	3..4	Indicates if name attributes are requested / delivered. See Section 5.4 for all attributes.	00 No Name attributes 01 Provide Name attributes
3.	Address	5..6	Indicates if address attributes are requested / delivered. See Section 5.4 for all attributes.	00 No Address attributes 01 Provide Address Attributes
4.	Age related	7..10	Indicates what age attribute is requested / delivered. See Section 5.4 for all attributes.	0000 No Age related attribute 0001 18orOlder 0010 Reserved 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 Date of Birth
5.	Gender	11..12	Indicates if gender attribute is requested / delivered. See Section 5.4 for all attributes.	00 No gender attribute 01 Provide gender attribute

Nr.	Category	Bits	Description	Values
6.	Reserved space	13..16	Reserved space for future services	All zero for now

Table 11: Overview of `Requested-` and `DeliveredServiceID`

The service IDs are included in the SAML request message to specify part of the Merchant's request. For example, if the Merchant wishes to do an authentication (by requesting the BIN), only the second bit should be equal to one, which leads to an integer value of 16384.

5.4 Consumer attributes

The consumer attributes are located in the SAML 2.0 message as a result of the requested service with the `BankID.RequestedServiceID`. The `consumer.bin` and `consumer.transientid` **are not** defined as consumer attributes and are located at a different place in the SAML 2.0 message. iDIN can provide the following attributes about a Consumer in a SAML Response message AttributeStatement. Where applicable, the consumer attributes will be formatted according to the NEN-ISO 8601 standard.

Nr.	Consumer attribute	Group[5]	Description	Formatting rules
1.	`consumer.deprecatedbin`	-	Same as BIN but is used to sustain continuous usage of consumer BIN. This element can occur for instance when two Merchants fuse, or, for some particular reason the `consumer.bin` has been reset.	See `consumer.bin`
2.	`consumer.gender`	Can be requested individually	Gender of Consumer	0 (= unknown) 1 (= male) 2 (= female) 9 (= not specified)
3.	`consumer.legallastname`	Name	Legal last name of Consumer without prefixes (from: NEN 1888_2002, p5, "significant deel van de achternaam")	Max200Text (from: NEN 1888_2002, p10)
4.	`consumer.preferredlastname`	Name	Last name of Consumer as preferred by Consumer (analogous to legallastname)	Max200Text
5.	`consumer.partnerlastname`	Name	Last name of Consumer's registered partner (analogous to legallastname)	Max200Text
6.	`consumer.legallastnameprefix`	Name	Last name prefix of Consumer's legal last name	Max10Text

[5] This represent the group as defined in Table 11. Some attributes are not part of any group but can be requested and provided individually depending on the `Requested-` or `DeliveredServiceID`

Nr.	Consumer attribute	Group[5]	Description	Formatting rules
7.	consumer.preferredlastnameprefix	Name	Last name prefix of Consumer's preferred last name. (analogous to legallastnameprefix)	Max10Text
8.	consumer.partnerlastnameprefix	Name	Last name prefix of Consumer's partner last name (analogous to legallastnameprefix)	Max10Text
9.	consumer.initials	Name	Initials of Consumer, as defined by NEN 1888_2002, p6: "voorletters-n". Only the first letter of all first names are used which must be capitalized and concatenated without spaces	Max24Text (from: NEN 1888_2002, p11)
10.	consumer.dateofbirth	Can be requested individually	Date of birth of Consumer	Basis format (CCYYMMDD) from NEN-ISO 8601 If the month or day of the date of birth is unknown '00' shall be returned e.g. 19870400
11.	consumer.18orolder	Can be requested individually	Specifies whether the Consumer is 18 or older of age.	Boolean Age verification when the date of birth has an unknown day or month shall be based on the latest day of the month or latest month of the year respectively.
12.	consumer.street	Address	Street name of the Consumer's residential address (from: NEN 5825_2002, p4, "straatnaam") Used for NL addresses only	Max43Text (from: NEN 5825_2002, 5.3.2)
13.	consumer.houseno	Address	House number of the Consumer's residential address (from: NEN 5825_2002, p4, "huisnummer") Used for NL addresses only	Max5Numerical (from: NEN 5825_2002, 5.3.4)
14.	consumer.housenosuf	Address	House number suffix (from: NEN 5825_2002, p4, "huisnummertoevoeging") Used for NL addresses only	Max6Text (from: NEN 5825_2002, 5.3.5)
15.	consumer.addressextra	Address	Additional address details of Consumer's residential address (from: NEN 5825_2002, p4, "locatieomschrijving") Used for NL addresses only	Max70Text (from: NEN 5825_2002, 5.3.1)

Nr.	Consumer attribute	Group[5]	Description	Formatting rules
16.	consumer.postalcode	Address	Postal code of the Consumer's residential address (from: NEN 5825_2002, p4, "postcode") Used for NL addresses only	Numerical part: n4 Alphabetic part: a2 For example: 0000AA (from: NEN 5825_2002, 5.3.11-12)
17.	consumer.city	Address	City of the Consumer's residential address Used for NL addresses only	Max24Text
18.	consumer.intaddressline1	Address	Used for non-NL addresses only. The diversity of international address formats, is captured in 3 freetext strings.	Max70Text ISO 20022
19.	consumer.intaddressline2	Address	Used for non-NL addresses only (analogous to consumer.intaddressline1)	Max70Text
20.	consumer.intaddressline3	Address	Used for non-NL addresses only (analogous to consumer.intaddressline1)	Max70Text
21.	consumer.country	Address	Country code of country where Consumer currently resides	2 code ISO 3166-1

Table 12: Consumer attributes

Notes:

- Consumer attributes are located in the SAML Attribute element. In the @Name attribute the Name indicated above is prefixed with the following "nl:bvn:bankid:1.0:attribute:". This results in attribute names containing only small-caps letters e.g. "nl:bvn:bankid:1.0:attribute:consumer.city";
- If one of the attributes is longer than the NEN-prescribed maximum length, then the name is truncated and the last character is replaced by a dash "-" (NEN 1888_2002, p14);

Some of the name or address related attributes might not be available at all Issuers, e.g., Consumer's partner last name. In that case the Issuer shall not include the attribute in the attribute statement provided.Example of SAML Attribute:

```
<Attribute Name="urn:nl:bvn:bankid:1.0:consumer.dateofbirth">
  <AttributeValue>19850101<AttributeValue>
<Attribute>
```

5.5 Guaranteed minimal set of requested attributes

Table 13 shows the minimal set of attributes that must be provided by the Issuer for the name and address attributes when these are requested by the Merchant. The other categories (Consumer IDs, age related and gender) are requested and provided individually, and therefore must always be

delivered to the Merchant upon request conform the specified format (i.e. when the Merchant requests a date of birth the Issuer must provide this, and not some other form of age verification).

If the Issuer is unable to provide the attribute groups conform this minimal set, or cannot provide one or more of the individually requested ID, gender and/or an age related attributes, it fails to respond with a complete success. When this occurs the Issuer returns all attributes and/or an ID it can deliver (which are part of the categories requested by the Merchant), but indicates with a status code that the request could not be performed with a complete success, see Section 12.4 for more detail. For this scenario the `DeliveredServiceID` is not equal to the `RequestedServiceID`.

Group	Minimal set of attributes
Name	**One of the three options listed below:**
1	`consumer.legallastname`
2	`consumer.preferredlastname`
3	`consumer.partnerlastname`
Address	**One of the five options listed below:**
1	`consumer.postalcode` AND `consumer.houseno`
2	`consumer.streetname` AND `consumer.houseno` AND `consumer.city`
3	`consumer.postalcode` AND `consumer.addressextra` (for addresses that don't have a house number e.g. a 'woonboot')
4	`consumer.streetname` AND `consumer.addressextra` AND `consumer.city` (for addresses that don't have a house number e.g. a 'woonboot')
5	`consumer.intaddressline1` AND `consumer.country` (for international addresses)

Table 13: Minimal set of attributes provided by Issuer per requested attribute group

6 iDIN Directory protocol

6.1 General

The Directory protocol allows a Merchant to retrieve an up to date list of participating Consumer Banks (Issuers) from his Routing Service, which can be presented to the Consumer. In case of changes in the list of Issuers, the Directory protocol will supply the Merchant with the update of this list.

It is **not allowed to perform the Directory protocol for each transaction**. Since the list of Issuers only changes occasionally, it is sufficient to execute the Directory protocol on a weekly basis and check if the list has changed based on the directoryDateTimestamp. If the Issuer list has changed, the latest version has to be saved and used for any subsequent transaction. Routing Services will normally also inform all Merchants (e.g. by email) about changes in their Issuer list. The Directory protocol should at least be executed once a week.

The Directory protocol (like the Transaction protocol and the Status protocol) consists of a HTTP POST request from the Merchant to the Routing Service, followed by a HTTP response. The DirectoryReq is sent to the URL that is provided to the Merchant by the Routing Service for this specific purpose. This URL can be different from the one that is used for the AcquirerTrxReq and the AcquirerStatusReq, but it can also be the same URL.

The Routing Service validates the authenticity of the message sent by the Merchant by verifying the signature in the message. To do this, the Routing Service needs the public part of the Merchant's Certificate, including the public key. The way in which the Merchant's certificate is communicated with the Routing Service varies per bank.

Please refer to Chapter 10 for more information on authentication and security. APPENDIX B shows an example of a DirectoryReq and DirectoryRes.

6.2 Directory Request (DirectoryReq)

The DirectoryReq consists of an XML message that is sent to the Routing Service with HTTP POST.

Table 14 shows all elements and attributes of the DirectoryReq:

Element/attributes	Mandatory	Contents
DirectoryReq	Yes	Root of the message
@version	Yes	**Must be:** "1.0.0"
@productID	Yes	**Must be:** "NL:BVN:BankID:1.0"
+ createDateTimestamp	Yes	Contains `DateTime` at which this directory request message was created.
+ Merchant	Yes	Contains Merchant sub-elements.
++ MerchantID	Yes	Contains `Merchant.MerchantID` as supplied to the Merchant by the Acquirer
++ subID	Yes	Contains `Merchant.subID` as supplied to the Merchant by the Acquirer
+ Signature	Yes	Contains all signature sub-elements. See Chapter 10 for more information about signing and Section 10.6 for all the elements

Table 14: Elements/attributes of the DirectoryReq

6.3 Directory Response (DirectoryRes)

The Merchant will receive the DirectoryRes as a reply to the DirectoryReq. This XML message contains a list of pairwise `Issuer.Name` and `Issuer.IssuerID`. Issuers are grouped by country. The Issuers in the Consumer's country of choice may be presented at the top in the Issuer selection list, the rest is sorted alphabetically, first by country, then by name. Table 15 shows all elements and attributes that appear in the DirectoryRes message.

Element/attributes	Mandatory	Contents
DirectoryRes	Yes	Root of the message
@version	Yes	**Must be:** "1.0.0"
@productID	Yes	**Must be:** "NL:BVN:BankID:1.0"
+ createDateTimestamp	Yes	Contains `DateTime` at which this directory response message was created.
+ Acquirer	Yes	Contains Acquirer sub-elements
++ AcquirerID	Yes	Contains `Acquirer.AcquirerID`
+ Directory	Yes	Contains all Directory sub-elements
++ directoryDateTimestamp	Yes	Contains `DateTime` at which the list of Issuers was last updated by the Acquirer
++ Country	Yes (1..∞)	Contains al Country sub-elements
+++ countryNames	Yes (1..∞)	Contains all `Country.countryNames`
+++ Issuer	Yes (1..∞)	Contains pairwise issuerID and issuerName sub-elements
++++ issuerID	Yes	Contains `Issuer.IssuerID`
++++ issuerName	Yes	Contains `Issuer.Name`
+ Signature	Yes	Contains all signature sub-elements. See Chapter 10 for more information about signing and Section 10.6 for all the elements

Table 15: Elements/attributes of the DirectoryRes

6.4 Presentation of the Issuer Bank selection list

The Merchant bears primary responsibility for initiating the iDIN process and for the communication to the Consumer regarding the status of this process. Merchants who offer iDIN to their customers must include iDIN in their lists of authentication methods (if any). iDIN must be presented in the list of methods in such a way that it receives at least the same amount of attention as all other competitors. It must be clear for the Consumer that he is about to authenticate with iDIN. The Merchant must clearly indicate to the Consumer what iDIN service he is about to initiate for the Consumer.

To ensure that the Consumer experience of an iDIN transaction is consistent and recognisable through all Merchant websites; all Merchants have to comply with certain presentation standards.

All Issuers in the DirectoryRes have to be shown in a "dropdown list box". The first element in this list is "Kies uw bank…" (*Eng. "Choose your bank…"*), and is selected by default. Subsequently the name of the Consumer's country of choice is shown (either the Merchant's own country or the country where the Consumer is expected to be from). The names of all Issuers that belong to the Merchant's country of choice are presented next in separate elements, in the same (alphabetical) order as they are presented in the DirectoryRes. Following this the names of other available countries are presented alphabetically. Within each country the banks from that country are sorted alphabetically, in the same order as

presented in the DirectoryRes. The Merchant should generate an error message whenever one of the elements “Kies uw bank…” and countryNames elements is selected by the Consumer.

It is recommended to configure the HTML “value” field of the items in the list box to be the `Issuer.IssuerID` (BIC) of the corresponding Issuer, because this value is necessary for subsequent messages. An example of the Issuer selection list is shown in the Figure below.

Figure 4: Example of (open) dropdown list box showing the Issuer list

Merchants are **not allowed** to remove Issuers temporarily from the Issuer selection list or to grey them out.

If a Merchants learns via the iDIN Notification System (Central Reporting tool for iDIN Validation Services and Routing Services to state system non-availability) or via error messages received from the Acquiring bank that a particular Validation Service is currently not available, the Merchant may display a message on its website informing Consumers that the particular bank is currently not available. In other words, it is permissible to display a message conveying such information but it is not permissible to temporarily remove or grey out the Issuer concerned from the Issuer selection list.

7 iDIN Transaction protocol

7.1 General

The Transaction protocol initiates the exchange of messages of the actual iDIN process. After the Consumer has chosen iDIN as an identification method and has selected his bank, the Merchant sends a Transaction Request to the Routing Service. Within the iDIN standards this message is referred to as the AcquirerTrxReq. The Routing Service replies to the AcquirerTrxReq with a Transaction Response (AcquirerTrxRes). This AcquirerTrxRes will also (among other fields) contain the `issuerAuthenticationURL`. This URL will redirect the browser of the Consumer to the Issuer in order to let him authorise the iDIN transaction.

The Transaction protocol consists of a HTTP POST request from the Merchant to the Routing Service, followed by a HTTP response. The AcquirerTrxReq is sent to the URL that is provided to the Merchant by the Routing Service for this specific purpose. This URL can be different from the one that is used for the DirectoryReq and the AcquirerStatusReq, but it can also be the same URL.

The Routing Service validates the authenticity of the message sent by the Merchant by verifying the signature in the message. To do this, the Routing Service needs the public part of the Merchant's Certificate, including the public key. The way in which the Merchant's certificate is communicated with the Routing Service varies per bank.

Please refer to Chapter 10 for more information on authentication and security. APPENDIX B shows an example of an AcquirerTrxReq and AcquirerTrxRes.

7.2 Transaction Request (AcquirerTrxReq)

The XML message sent by the Merchant to the Routing Service to initiate the transaction contains the elements and attributes shown in Table 16. The iDIN product specific information (SAML 2.0 message) is put inside the container element in the AcquirerTrxRes in the form of an AuthnRequest, as shown in Table 17.

Element/attributes	Mandatory	Contents
AcquirerTrxReq	Yes	Root of the message
@version	Yes	**Must be:** "1.0.0"
@productID	Yes	**Must be:** "NL:BVN:BankID:1.0"
createDateTimestamp	Yes	Contains `DateTime` at which this Transaction Request message was created
+ Issuer	Yes	Contains Issuer sub-elements
++ IssuerID	Yes	Contains `Issuer.IssuerID`
+ Merchant	Yes	Contains all Merchant sub-elements
++ merchantID	Yes	Contains `Merchant.MerchantID`
++ subID	Yes	Contains `Merchant.subID`
++ merchantReturnURL	Yes	Contains `Merchant.returnURL`
+ Transaction	Yes	Contains all Transaction sub-elements
++ expirationPeriod	No	Contains `expirationPeriod`

Element/attributes	Mandatory	Contents
++ language	Yes	Contains `language` of Consumer
++ entranceCode	Yes	Contains `entranceCode`
++ container	Yes	Contains the SAML 2.0 AuthnRequest message, see Table 17
+ Signature	Yes	Contains all signature sub-elements. See Chapter 10 for more information about signing and Section 10.6 for all the elements

Table 16: Elements/attributes of AcquirerTrxReq

As explained, the SAML 2.0 AuthnRequest message is embedded in the container element. The AuthnRequest message is a standardised message, so it contains elements that are not used in iDIN. This elements and attributes inside the container are shown in Table 17.

Element/attributes (inside container element)	Mandatory	Contents
AuthnRequest	Yes	Root of the message (inside container element)
@ID	Yes	Contains `BankID.MerchantReference`
@Version	Yes	**Must be:** "2.0"
@IssueInstant	Yes	Contains `DateTime` at which this Transaction Request message was created. This must be the same value as in createDateTimestamp.
@Destination		Shall not be present
@Consent	No	May be present. Shall be ignored. No matter what consent has been provided to the Merchant, the Issuer is always responsible for obtaining the Consumer's consent
@ForceAuthn	No	May be present. **Must be:** "true" if present. Explicit authentication shall be enforced for every iDIN service
@IsPassive	No	May be present. **Must be:** "false" if present. Since explicit authentication is enforced the Issuer cannot be passive
@ProtocolBinding	Yes	**Must be:** "nl:bvn:bankid:1.0:protocol:iDx"
@AssertionConsumer-ServiceIndex		Shall not be present
@AssertionConsumer-ServiceURL	Yes	Contains `Merchant.merchantReturnURL`
@AttributeConsuming-ServiceIndex	Yes	Contains `BankID.RequestedServiceID`
@ProviderName		Shall not be present
+ Issuer	Yes	Contains `Merchant.MerchantID`
+ @NameQualifier		Shall not be present
+ @SPNameQualifier		Shall not be present
+ @Format		Shall not be present
+ @SPProviderID		Shall not be present
+ Signature		Shall not be present. Relevant signing is done at iDx level.
+ Subject		Shall not be present
+ NameIDPolicy		Shall not be present
+ Conditions	No	May be present
+ RequestedAuthnContext	Yes	Contains RequestedAuthnContext sub-elements
+ @Comparison	Yes	**Must be:** "minimum"

Element/attributes (inside container element)	Mandatory	Contents
++ AuthnContextClassRef	Yes	Contains `BankID.LOA`
++ AuthnContextDeclRef		Shall not be present
+ Scoping	No	May be present

Table 17: Elements/attributes inside the container of the AcquirerTrxReq

7.3 Transaction Response (AcquirerTrxRes)

If everything goes well the Routing Service will reply to the AcquirerTrxReq with the AcquirerTrxRes. Table 18 shows all fields of the AcquirerTrxRes message. The AcquirerTrxRes has no container, so there is no SAML 2.0 message in this response (this is different in case of an Error Response, see Chapter 9 on error handling).

Element/attributes	Mandatory	Contents
AcquirerTrxRes	Yes	Root of the message
@version	Yes	**Must be:** "1.0.0"
@productID	Yes	**Must be:** "NL:BVN:BankID:1.0"
+ createDateTimestamp	Yes	Contains `DateTime` at which this Transaction Response message was created
+ Acquirer	Yes	Contains Acquirer sub-elements
++ AcquirerID	Yes	Contains `Acquirer.AcquirerID`
+ Issuer	Yes	Contains Issuer sub-elements
++ IssuerAuthenticationURL	Yes	Contains `issuerAuthenticationURL`
+ Transaction	Yes	Contains Transaction sub-elements
++ transactionID	Yes	Contains `Transaction.TransactionID`
++ transactionCreateDate-TimeStamp	Yes	Contains `DateTime` at which the transaction was first registered by the Routing Service. This time can be used by Merchant, Routing Service and Validation Service for reporting on the transaction.
+ Signature	Yes	Contains all signature sub-elements. See Chapter 10 for more information about signing and Section 10.6 for all the elements

Table 18: Elements/attributes of AcquirerTrxRes

7.4 Errors when executing Transaction Protocol

A number of errors may occur when executing the iDIN Transaction Protocol. These may be related to unavailability or an error within your own web environment (Merchant), the Routing Service environment or the Validation Service environment.

The following situations may occur:

- You receive an error response (message B'(X)) from your Routing Service within the set time-out period;
- You do not receive any response within the set time-out period.

In all of the above cases, the Transaction protocol cannot be successfully executed. This means it is not possible for the iDIN transaction to take place at that time. Error handling is explained in more detail in Chapter 9.

7.5 Redirect to the online banking environment

After receiving the AcquirerTrxRes the Merchant has to redirect the Consumer to the `issuerAuthenticationURL` of the selected Issuer, as stated in the AcquirerTrxRes message. If the Merchant's page contains HTML frames, these will be removed by the Issuer ('frame busting'). If and when the Consumer returns to the Merchant's website (with the `merchantReturnURL`), the Merchant will have to completely rebuild its own page to show confirmation of iDIN reception.

7.5.1 Specific requirement iDIN Mobile: Redirect to Issuer (no in-app browser)

The Merchant needs to redirect the Consumer to the selected Issuer from the browser window or Merchant app. If it is not possible to keep the Consumer in the same browser window then this should be communicated to the Consumer (e.g.: *"You will now be redirected to the app or (mobile) website of your bank" or in Dutch "U wordt nu doorverwezen naar de app of (mobiele) website van uw bank").*

In case of a transaction initiated in the mobile Merchant app, it is **not allowed** to present the Issuer approval screens in a web view component within the Merchant's own app (in-app browser). The complete transaction flow, up to the redirect back to the Merchant's app, must take place in an app that is trusted by the Consumer, either the Consumer's chosen browser or the Issuer's mobile app. Thus, the `issuerAuthenticationURL` must, for execution, be offered to the mobile operating system at all times. During the transaction flow it should not be possible for the Consumer to initiate another transaction through the Merchant's original app.

Relevant details about the redirect from the Merchant to the Issuer's mobile channel:

- The Issuer decides which Consumers to redirect to which channel. For example, some Issuers may treat users of tablet devices the same as mobile users, while others will treat them like PC users;
- The Merchant should not intervene with the redirect. The `issuerAuthenticationURL` has also to be used for mobile iDIN transactions.
- If the Issuer has integrated iDIN mobile in its mobile banking app, the Consumer is offered the option, on a 'landing page', to open the app or approve the iDIN transaction via the (mobile) web page. On this 'landing page' the Consumer might be offered the option to download the latest version of the mobile banking app, if it is not yet installed on the Consumer's device.

7.6 Redirect to the Merchant environment

After the Consumer has performed the necessary steps at the Issuer he/she will be presented with a 'Continue' button that must redirect him back to the website of the Merchant with the `merchantReturnURL` as supplied in the AcquirerTrxReq.

Two GET parameters are appended to this URL: the `entranceCode` (see Table 9) with 'ec' as GET parameter name, and the `Transaction.TransactionID` (see Table 9) with 'trxid' as GET parameter name. It is also possible for a Merchant to add additional parameters. For example, if the Merchant defines the `merchantReturnURL` as follows:

http://www.webshop.nl/processtransaction?producttype=electronics

The final URL will look something like:

http://www.webshop.nl/processtransaction?producttype=electronics&trxid=0010123456789012&ec=4hd7TD9wRn76w6gGwGFDgdL7jEtb

The `entranceCode` field should contain a unique value, with the object of preventing message 'sniffing'. Use of the same `entranceCode` each time would allow malevolent individuals to intercept the data from the `merchantReturnURL` and make fraudulent use of this information. This is why using unique values for the `entranceCode` is extremely important.

Note that a Consumer may not always use the redirect back to the Merchant environment that is offered by the Issuer. Also note that in exceptional cases the Issuer may be unable to match the `Transaction.TransactionID` in its system or another error occurs, which makes it impossible to redirect the Consumer back to the Merchant. In all other cases the Consumer is redirected with the parameters defined above, regardless of the final status of the transaction (success, cancelled, failed). The Merchant must then use the Status protocol (see Chapter 8) to determine the status of the transaction.

7.6.1 Requirements for iDIN Mobile: redirect to the Merchant environment.

After the Consumer has been authenticated in either the mobile or regular channel and has approved the transaction, he is redirected back to the Merchant as normal (using the `merchantReturnURL`). The `merchantReturnURL` usually starts with 'https', redirecting the Consumer back to a page on the mobile device's browser. If the Consumer has initiated the transaction from the Merchant's mobile app, the `merchantReturnURL` can be an app handler, which will redirect the Consumer directly to the Merchant app. An app handler is a call that can be used to start an app and request it to initiate a specific action. For example a Merchant's app handler can start with 'nl.companyname.idin://' and this will open the Merchant's app.

NB: the `merchantReturnURL` should always direct to a web page or app of the Merchant (or party acting on behalf of the Merchant).

7.7 Errors during execution of the redirect to the Issuer, approving the iDIN request and/or the redirect to the Merchant environment

The following errors may occur during execution of the redirect to the online banking environment (Issuer), the execution of the iDIN transaction at the Issuer and/or the redirect back to your (Merchant) environment:

1. The bank page is unavailable, resulting in the Consumer being unable to approve the iDIN request, and thus the Consumer cannot be properly redirected to your confirmation page;

2. The bank page is available but the Consumer cannot (after approving the iDIN request or otherwise) be properly redirected to your confirmation page.

In both situations the Consumer cannot (as the result of a disturbance) return to your confirmation page in the normal way. In that case the Consumer can return to your website by using the 'back' button or entering the URL, for example. Considering the short validity of the Assertion the Merchant would be unable to retrieve the status of the transaction. More precisely, the Merchant is only allowed to request the status when the Consumer is successfully returned to the Merchant's website using the return URL, see Section 8.5. Therefore, the Merchant has to make a new transaction when something has gone wrong in the redirections.

7.8 Four different scenarios for completion of iDIN Mobile transaction

To give an overview of all the possible process steps and important notes when dealing with iDIN Mobile transactions, we have specified four different scenarios. There are four different scenarios because either the Issuer or the Merchant or both can use a (mobile) web page or a mobile app.

Because these scenarios (could) differ from the regular (non-mobile) iDIN transactions they will be illustrated in the following paragraphs.

Section	Merchant	Issuing Bank
7.8.1	(Mobile) web page	(Mobile) web page
7.8.2	(Mobile) web page	Mobile banking app
7.8.3	Mobile app	(Mobile) web page
7.8.4	Mobile app	Mobile banking app

Table 19: Different scenarios for the completion of an iDIN mobile transaction

7.8.1 Consumer is redirected from the Merchant's (mobile) web page to the Issuer's (mobile) web page.

This is currently the most common iDIN mobile scenario, as it is identical to the regular desktop iDIN transaction flow. As such there are no specific notes for use in a mobile setting, but this scenario has been added for reasons of completeness.

The Consumer starts the transaction on the Merchant's mobile page and follows these steps:

Step	Description	Important note
1	The Consumer selects iDIN to provide identification or consumer attributes	
2	The Consumer selects his Issuer	
3	The Consumer is redirected to the Issuer of his choice	
4	The Issuer presents the Issuer's 'landing page' to the Consumer, which offers the	

Step	Description	Important note
	option to complete the iDIN transaction in the Issuer's mobile banking app or in the Issuer's (mobile) web page	
5	The Consumer selects the (mobile) web page	
6	The Consumer is redirected to the Issuer's (mobile) web page where he can log in and authorize the iDIN transaction. After completion of the transaction the Issuer shows the complete iDIN identification/authentication to the Consumer	
7	The Consumer is redirected back by the Issuer, to the Merchant's (mobile) web page using the `merchantReturnURL`, which was received from the Merchant	The `merchantReturnURL` usually starts with https:// and contains two parameters (`entranceCode` and `Transaction.TransactionID`) that can be used to correctly identify the Consumer upon his return
8	The Merchant shows the Consumer the result of the iDIN transaction	

Table 20: Scenario: Redirect from Merchant (mobile) web page to the Issuers (mobile) web page

7.8.2 Consumer is redirected from the Merchant's (mobile) web page to the Issuer's mobile banking app

The Consumer starts his customer journey on the Merchants (mobile) web page and follows the following steps:

Step	Description	Important note
1	The Consumer selects iDIN to provide identification or consumer attributes	
2	The Consumer selects his Issuing Bank	
3	The Consumer is redirected to the Issuer of his choice	
4	The Issuer presents the Issuer's 'landing page' to the Consumer, which offers the option to complete the iDIN transaction in the Issuer's mobile banking app or in the Issuer's (mobile) web page	
5	The Consumer selects the mobile banking app	

Step	Description	Important note
6	The Consumer is redirected to the Issuer's banking app where he can log in and authorize the iDIN transaction. After completion of the transaction the Issuer shows the complete iDIN identification/authentication to the Consumer	
7	The Consumer is redirected back by the Issuer, to the Merchant's (mobile) web page using the `merchantReturnURL`, which was received from the Merchant	Because the transaction takes place in the bank's app, outside of the web-browser setting, the browser session may be lost. This means the Merchant may not be able to recognize the Consumer using the browser session. Next to this, when redirecting the Consumer back to the Merchant from the bank-app, the `merchantReturnURL` is handled by the Operating System of the mobile device. The OS uses the native (default) browser to handle this URL. This discontinues the original browser session if the transaction was initiated in a non-native browser. The `merchantReturnURL` starts with https:// and contains two parameters (`entranceCode` and `Transaction.TransactionID`) that can be used to correctly identify the Consumer upon his return
8	The Merchant shows the Consumer the result of the iDIN transaction.	

Table 21: Scenario: Redirect form Merchant (mobile) web page to the Issuers mobile banking app

7.8.3 Consumer is redirected from the Merchant's mobile app to the Issuer's (mobile) web page

The Consumer starts his customer journey on the Merchants mobile app and follows the following steps:

Step	Description	Important note
1	The Consumer selects iDIN to provide identification or consumer attributes	
2	The Consumer selects his Issuer	
3	The Consumer is redirected to the Issuer of his choice	It is mandatory for the Merchant to let the Operating System, which is installed on the Consumer's mobile device, handle the `issuerAuthenticationURL`. See Section 8.5.1 for more information

Step	Description	Important note
4	The Issuer presents the Issuer's 'landing page' to the Consumer, which offers the option to complete the iDIN transaction in the Issuer's mobile banking app or in the Issuer's (mobile) web page	
5	The Consumer selects the (mobile) web page	
6	The Consumer is redirected to the Issuer's (mobile) web page where he can log in and authorize the iDIN transaction. After completion of the transaction the Issuer shows the complete iDIN identification/authentication to the Consumer	
7	The Consumer is redirected back by the Issuer, to the Merchant's app using the `merchantReturnURL`, which was received from the Merchant	The merchantReturnURL contains an app handler and two parameters (`merchantReturnURL` and `Transaction.TransactionID`) that can be used to correctly identify the Consumer upon his return. See Section 8.6 for more information
8	The Merchant shows the Consumer the result of the iDIN transaction	

Table 22: Scenario: Redirect from the Merchants mobile app to the Issuers (mobile) web page

7.8.4 Consumer is redirected from the Merchant's mobile app to the Issuer's mobile banking app

The Consumer starts his customer journey on the Merchants mobile app and follows the following steps:

Step	Description	Important note
1	The Consumer selects iDIN to provide identification or consumer attributes	
2	The Consumer selects his Issuer	
3	The Consumer is redirected to the Issuer of his choice	It is mandatory for the Merchant to let the Operating System, which is installed on the Consumer's mobile device, handle the `issuerAuthenticationURL`. See Section 8.5.1 for more information
4	The Issuer presents the Issuer's 'landing page' to the Consumer, which offers the option to complete the iDIN transaction in	

Step	Description	Important note
	the Issuer's mobile banking app or in the Issuer's (mobile) web page	
5	The Consumer selects the mobile banking app	
6	The Consumer is redirected to the Issuer's banking app where he can log in and authorize the iDIN transaction. After completion of the transaction the Issuer shows the complete iDIN identification/authentication to the Consumer	
7	The Consumer is redirected back by the Issuer, to the Merchant's app using the `merchantReturnURL`, which was received from the Merchant	The `merchantReturnURL` contains an app handler and two parameters (`entranceCode` and `Transaction.TransactionID`) that can be used to correctly identify the Consumer upon his return. See Section 8.6 for more information
8	The Merchant shows the Consumer the result of the iDIN transaction	

Table 23: Scenario: Redirect from the Merchants mobile app to the Issuers mobile banking app

7.9 Performance and time-out of transaction message

The performance of the Issuer and Routing Service systems has a direct influence on the Consumer's user experience. Therefore, iDIN sets a target time and time-out for the Transaction Response message. For a Merchant the relevant target time and time-out concerning the communication with its iDIN Routing Service is:

Communication	Target time (in seconds)	Time-out (in seconds)
AcquirerTrxReq → AcquirerTrxRes	2.0	7.6

Table 24: Performance requirements (for the 95th percentile[6])

The target time is the time (in seconds) in which the Merchant should receive a AcquirerTrxRes message after sending a AcquirerTrxReq. The time-out is the length of time after which the Merchant should no longer expect a response (most likely an error has occurred) and should act accordingly (for example by displaying an appropriate error message to the Consumer).

[6] 95th percentile is a statistical term indicating that 95% of transactions in a tested sample should be within the set target time.

8 iDIN Status protocol

8.1 General

To verify whether an iDIN transaction was successful the Merchant will start the Status protocol by sending a Status Request to the Routing Service. Within the iDIN standards this message is referred to as the AcquirerStatusReq.

To avoid unnecessary system load status requests should not be made unnecessarily, see Section 8.5 for more details on what is allowed. The AcquirerStatusReq message can be sent after the return of the Consumer to the Merchant's website (after the redirect from the Issuer).

Status protocol consists of a HTTP POST request from the Merchant to the Routing Service, followed by a HTTP response. The AcquirerStatusReq is sent to the URL that is provided to the Merchant by the Routing Service for this specific purpose. This URL can be different from the one that is used for the DirectoryReq and the AcquirerTrxReq, but it can also be the same URL.

The Routing Service validates the authenticity of the message sent by the Merchant by verifying the signature in the message. To do this, the Routing Service needs the public part of the Merchant's Certificate, including the public key. The way in which the Merchant's certificate is communicated with the Routing Service varies per bank.

Please refer to Chapter 10 for more information on authentication and security. Appendix C shows an example of an AcquirerStatusReq and AcquirerStatusRes.

8.2 Status Request (AcquirerStatusReq)

Table 25 contains all fields that are part of the AcquirerStatusReq XML message. The Merchant sends this message to the Routing Service. The AcquirerStatusReq does not contain a container with a SAML 2.0 message.

Element/attributes	Mandatory	Contents
AcquirerStatusReq	Yes	Root of the message
@version	Yes	**Must be:** "1.0.0"
@productID	Yes	**Must be:** "NL:BVN:BankID:1.0"
+ createDateTimestamp	Yes	Contains `DateTime` at which this Status Request message was created
+ Merchant	Yes	Contains Merchant sub-elements
++ merchantID	Yes	Contains `Merchant.MerchantID`
++ subID	Yes	Contains `Merchant.subID`
+ Transaction	Yes	Contains Transaction sub-elements
++ transactionID	Yes	Contains `Transaction.TransactionID`
+ Signature	Yes	Contains all signature sub-elements. See Chapter 10 for more information about signing and Section 10.6 for all the elements

Table 25: Elements/attributes of AcquirerStatusReq

8.3 Status Response (AcquirerStatusRes)

The reply to the AcquirerStatusReq is the AcquirerStatusRes. This message and the SAML Response inside the container are created by the Validation Service and sent to the Merchant via the Routing Service. The AcquirerStatusRes contains the elements and attributes as listed in Table 26. This message communicates the status of the transaction (related to the `Transaction.TransactionID` which was sent in the AcquirerStatusReq) to the Merchant. If the status equals "Success", the container element is included in the response. Inside the container element is the SAML 2.0 message containing the consumer data elements. The elements and attributes in the optional container are shown in Table 27.

Element/attributes	Mandatory	Contents
AcquirerStatusRes	Yes	Root of the message
@version	Yes	**Must be:** "1.0.0"
@productID	Yes	**Must be:** "NL:BVN:BankID:1.0"
+ createDateTimestamp	Yes	Contains `DateTime` at which this Status Response message was created
+ Acquirer	Yes	Contains Acquirer sub-elements
++ acquirerID	Yes	Contains `Acquirer.AcquirerID`
+ Transaction	Yes	Contains Transaction sub-elements
++ transactionID	Yes	Contains `Transaction.TransactionID`
++ status	Yes	Contains `Transaction.status`
++ statusDateTimeStamp	No	**Present only If:** `Transaction.status` = "Success", "Cancelled", "Expired" or "Failure" (not present when `Transaction.status` = "Open" or "Pending"). Contains `DateTime` at which the Issuer established the `Transaction.status` for this transaction and recorded it as part of the transaction details
++ container	No	**Present only if:** `Transaction.status` = "Success" Contains the SAML 2.0 Response message, see Table 27
+ Signature	Yes	Contains all signature sub-elements. See Chapter 10 for more information about signing and Section 10.6 for all the elements

Table 26: Elements/attributes of AcquirerStatusRes

As explained, the SAML 2.0 Response message is embedded in the container element. The Response message is a standardised message, and therefore it contains elements that are not used in iDIN. However, because it is up the Validation Service to omit these values it is not indicated in Table 27. The elements and attributes inside the container are shown in Table 27.

Element/attributes	Mandatory	Contents
Response	Yes	Root of this message inside the container element
@ID	Yes	Contains `Transaction.TransactionID` prefixed with 'RES-'
@InResponseTo	Yes	Contains `Merchant.MerchantReference`
@version	Yes	**Must be:** "2.0"
@IssueInstant	Yes	Contains `DateTime` at which this SAML Response message was created
+ Issuer	Yes	Contains `Acquirer.AcquirerID` **NB:** Issuer in this context is reserved SAML terminology and not related to the iDIN Issuer
+ Status	Yes	Contains the status of SAML Response in attribute

Element/attributes	Mandatory	Contents
++ StatusCode	Yes	Contains StatusCode sub-elements
+++@Value	Yes	**Must be:** "urn:oasis:names:tc:SAML:2.0:status:Success"
+++ StatusCode	Yes	StatusCode one level deeper
++++ @Value	Yes	**Has one of the two values:** "urn:nl:bvn:bankid:1.0:status:Success" or, if some attributes are not delivered conform the minimal set: "urn:nl:bvn:bankid:1.0:status:IncompleteAttributeSet" The use of this status code is explained in more detail in Section 12.4
+ Assertion	Yes	Contains Assertion sub-elements
+ @Version	Yes	**Must be:** "2.0"
+ @ID	Yes	Contains unique identifier created by the Validation Service
+ @IssueInstant	Yes	Contains `DateTime` at which this Assertion element was created. Different from the `DateTime` at which this Status Response message was created
++ Issuer	Yes	Contains `Issuer.IssuerID`
++ Signature	Yes	Contains the Signature sub-elements for the SAML signature created by the Validation Service. See Chapter 10 for more information about signing and Section 10.6 for all the elements. Section 10.2.1 deals with the signing of the SAML Assertion in particular.
++ Subject	Yes	Contains Subject sub-elements
+++ EncryptedID	Yes	Contains the encrypted element NameID which in turn contains the `consumer.bin` or `consumer.transientid`. See Section 10.3
++ Conditions	Yes	Contains Conditions sub-elements
++ @NotBefore	Yes	Contains `DateTime` at which the transaction request was created
++ @NotOnOrAfter	Yes	Contains `DateTime` 40 seconds after the Assertion@IssueInstant
+++ AudienceRestriction	Yes	Contains AudienceRestriction sub-elements
++++ Audience	Yes	Contains the `Merchant.LegalID`
++++ OneTimeUse	Yes	Is present but is left empty
++ AuthnStatement	Yes	Contains AuthnStatement sub-elements
++ @AuthnInstant	Yes	Contains `DateTime` at which the authentication took place
+++ AuthnContext	Yes	Contains AuthnContext sub-elements
++++ AuthnContextClassRef	Yes	Contains `BankID.LOA`
++++ Authentication-Authority	Yes	Contains `Issuer.IssuerID`
++ AttributeStatement	Yes	Contains AttributeStatement sub-elements
+++ Attribute	Yes	An unencrypted Attribute
++++@Name	Yes	**Must be:** "urn:nl:bvn:bankid:1.0:bankid.deliveredserviceid"
++++AttributeValue	Yes	Contains `BankID.DeliveredServiceID`
+++ EncryptedAttribute	No (0..∞)	Contains encrypted consumer attributes. One for each consumer attribute. See Section 10.3

Table 27: Elements/attributes inside the container of AcquirerStatusRes

8.4 Errors during execution of Status Protocol

When using the Status Protocol to request an iDIN transaction status, errors can occur which can lead to the Merchant not obtaining the status of the transaction at that moment. It is therefore not possible to show the Consumer the final status of the transaction at that moment. Recommended messages to show to Consumers are defined further on in this document.

8.5 Restrictions on AcquirerStatusReq

A Merchant may only initiate an AcquirerStatusReq in the following case:

- It is triggered by a Redirect to Merchant (E) as part of the Transaction protocol.

The SAML Assertion issued by the Validation Service is only valid for 30 seconds. From the moment the Consumer has been successfully redirected to the Merchant up to the time the Assertion is expired, the Merchant can make status requests. However, the Merchant should only perform more than one status request if it has received a timeout from the Routing Service. The Merchant is not allowed to perform any more status requests once it received notification from the Validation Service that the Assertion is expired, see Chapter 9 for the details.

Merchants will be perceived as committing undesirable actions if they use the Status Request more than the above described limitation, as doing so places unnecessarily heavy demands on the Acquirer's and Issuer's system.

8.6 Performance and time-out of status messages

The performance of the Issuer/Validation Service and Routing Service systems has a direct influence on the Consumer's user experience. Therefore iDIN sets a target time and time-out for the status response message. For a Merchant the relevant target time and time-out concern the communication with its iDIN Routing Service is:

Communication	Target time (in seconds)	Time-out (in seconds)
AcquirerStatusReq → AcquirerStatusRes	2.0	7.6

Table 28**: Performance requirements (for the 95th percentile[7])**

The target time is the time (in seconds) in which the Merchant should receive a AcquirerStatusRes after sending a AcquirerStatusReq. The time-out is the length of time after which the Merchant should no longer expect a response (most likely an error has occurred) and should act accordingly (for example by displaying an appropriate error message to the Consumer).

[7] 95th percentile is a statistical term indicating that 95% of transactions in a tested sample should be within the set target time.

9 Error handling

9.1 General

If an error occurs while processing a DirectoryReq, AcquirerTrxReq or AcquirerStatusReq, for example because a request contains an invalid value, an AcquirerErrorRes will be returned instead of the regular response. Also, for certain special cases an error is returned to the Merchant e.g. when the Assertion is expired. This is discussed in more detail in Appendix A.

The AcquirerErrorRes has the same main structure as indicated in Table 29 for all three types of requests. The container is only present when receiving an error after an AcquirerStatusReq. Appendix C shows an example of an AcquirerErrorRes.

9.2 Error Response (AcquirerErrorRes)

Instead of the regular response (DirectoryRes, AcquirerTrxRes or AcquirerStatusRes) the Routing Service will return an AcquirerErrorRes if an error occurs during the reception or processing of the request, or if the request contains values that are not allowed or do not comply with the required format. Table 29 lists the fields that appear in the AcquirerErrorRes and Table 30 lists the elements and attributes that appear inside the container.

Element/attributes	Mandatory	Contents
AcquirerErrorRes	Yes	Root of the message
@version	Yes	**Must be:** "1.0.0"
@productID	Yes	**Must be:** "NL:BVN:BankID:1.0"
+ createDateTimestamp	Yes	Contains `DateTime` at which this Error Response message was created
+ Error	Yes	Contains Error sub-elements
++ errorCode	Yes	Contains `Error.errorCode` see Appendix A
++ errorMessage	Yes	Contains `Error.errorMessage` see Appendix A
++ errorDetail	No	Contains `Error.errorDetail`
++ suggestedAction	No	Contains `Error.suggestedAction`
++ consumerMessage	No	Contains `Error.consumerMessage` see Appendix A
++ container	No	Contains the SAML 2.0 Response message, see Table 30
+ Signature	Yes	Contains all signature sub-elements. See Chapter 10 for more information about signing and Section 10.6 for all the elements

Table 29: Elements/attributes of the AcquirerErrorRes

Element/attributes	Mandatory	Contents
Response	No	Root of the message inside the container
@ID	Yes	Contains `Transaction.TransactionID` prefixed with 'RES-'
@InResponseTo	Yes	Contains `BankID.MerchantReference`
@Version	Yes	**Must be:** "2.0"
@IssueInstant	Yes	Contains `DateTime` at which this Error Response message was created
+ Issuer	Yes	Contains `Acquirer.AcquirerID`
+ Status	Yes	Contains Status sub-elements

Element/attributes	Mandatory	Contents
++ StatusCode	Yes	Contains StatusCode sub-elements.
++ @Value	Yes	Contains a first level SAML status code equal to: “urn:oasis:names:tc:SAML:2.0:status:Requester” which means the request could not be performed due to an error on the part of the requester
+++ StatusCode	Yes	Contains StatusCode sub-elements.
+++ @Value	Yes	Contains the second level status code. See Appendix A for which status code is returned when
++ StatusMessage	Yes	Provides a human readable hint to which field caused the error

Table 30: Elements/attributes inside the container of AcquirerErrorRes

9.3 Non-availability

It might be possible that one of the Issuers is temporarily unavailable. In this case, transactions that have to be processed by this Issuer will generate an AcquirerErrorRes. When the Routing Service has determined unavailability of an Issuer it will communicate this to the Issuer immediately. This means that Merchant will never have to contact the Issuer directly.

It might also be possible that the Routing Service itself is temporarily unavailable. In this case, unless the Merchant has more than one Routing Service, no iDIN transactions can be processed and the Routing Service will generate an AcquirerErrorRes or time-out.

Lastly, the Merchant return-webpage might not be working properly.

For all three cases we recommend displaying a clear error message to the Consumer.

10 Security and certificates

10.1 General principles of certificates

For asymmetric encryption two keys are used: one public key and one private key. The public key is linked to the certificate and can be shared with anyone. The owner of the certificate must keep the private key confidential at all times. The specific characteristics of the private and public part of the certificates will allow the encryption of a message with the public part while the result can be decrypted with the private part, and vice versa. The certificates should have RSA keys of 2048 bit long. It is not possible to decrypt a text with the same key that was used to encrypt it.

These specific characteristics enable two applications of a certificate:

- Encryption of a message. By encrypting a message with the public key of the receiving party the information can only be read by the recipient (who has sole knowledge of the private key);
- Creating an electronic signature of a message. By encrypting the (hash) message with the private key, the recipient can determine the authenticity of the (sender of the) message by verifying the signature with the public part of the certificate. The recipient will also verify the integrity of the message to make sure the contents of the message was not changed by a third party.

The TLS connection that is used within iDIN between the Merchant and the Routing Service is based on the first application. The TLS connection uses at least 128-bit encryption based on a server side certificate of the Routing Service.

Since iDIN does not put any constraints on the communication between the Consumer and the Merchant, this can be either with or without a TLS connection. Merchants are however strongly advised to always use TLS on the transaction pages of their website. The iDIN standard also uses electronic signatures to ensure the authenticity, integrity and non- repudiation of all messages with the exception of redirects. The electronic signature of the Routing Service in the AcquirerStatusRes message, for example, enables the Merchant to verify the authenticity of the transaction confirmation.

10.2 Signing iDIN messages

All messages that are sent by the Merchant to the Routing Service (DirectoryReq, AcquirerTrxReq and AcquirerStatusReq) have to be signed by the Merchant. Messages are signed in accordance with the "XML Signature Syntax and Processing (2nd Edition) W3C Recommendation" of 10 June 2008[8], with the following settings and restrictions applied:

- The entire XML message[9] must be signed;
- For the purpose of generating the digest, the exclusive[10] canonicalization algorithm must be used;

[8] http://www.w3.org/TR/xmldsig-core/

[9] XML Signature reference to the signed info URI is left blank, see example messages in APPENDIX C

[10] http://www.w3.org/2001/10/xml-exc-c14n

- For the purpose of generating the signature value, the exclusive[11] canonicalization algorithm must be used;
- The syntax for an enveloped[12] signature must be used. The signature itself must be removed from the XML message using the default transformation prescribed for this purpose;
- For hashing purposes the SHA-256[13] algorithm must be used;
- For signature purposes the RSAWithSHA256[14] algorithm must be used. RSA keys must be 2,048 bits long;
- The public key must be referenced using a fingerprint of an X.509 certificate. The fingerprint must be calculated according to the following formula HEX(SHA-1(DER certificate)) [15].

Note: According to Base64 specifications line breaks are allowed to be inserted after each 76 characters using a CR/LF[16].

In general Merchants don't need to have extensive knowledge of RSA since most programming languages have libraries with functionalities that implements XML Digital Signature processing. It is strongly recommended to use these standard libraries.

Standard functionality for creation and verification of RSAWithSHA256 digital signatures is available in commonly used software platforms, from the following versions and higher:

- PHP 5.5;
- Microsoft .NET 4,5;
- Java SE 7.

This functionality may also be available in earlier versions of these platforms and in other platforms (e.g. Python, Ruby).

To further support Merchants, software libraries have been developed for iDIN in .NET, PHP and Java. Please contact your bank about this for more information.

For information about creating the public and private key pair please refer to paragraph 10.5.

10.2.1 Signing of the SAML 2.0 Assertion

Next to the regular signing of the entire XML message in the iDx protocol, the Validation Service signs the SAML Assertion separately. This Assertion is passed along the Merchant via the Routing Service in the AcquirerStatusRes (only for the `Transaction.status` 'Success'). This signing in the SAML Assertion follows the same requirements that apply to signing as discussed before except for the following:

[11] http://www.w3.org/2001/10/xml-exc-c14n

[12] http://www.w3.org/TR/xmldsig-core/#sec-EnvelopedSignature

[13] http://www.w3.org/2001/04/xmlenc#sha256

[14] http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/#sec-SHA256

[15] DER (Distinguished Encoding Rules), SHA-1 (Secure Hash Algorithm – 1), HEX (Hexidecimal)

[16] http://tools.ietf.org/html/rfc2045#section-6.8

- Instead of referring to the certificate of the Issuer to be used for verifying the signature by fingerprint, the **entire certificate is included**. The KeyName element is replaced by an X509Data element containing an X509Certificate element containing the entire X509 certificate of the Validation Service, See Table 32. Other elements in KeyInfo shall not be used. This is done to guarantee availability of the certificate of the Validation Service for the Merchant in an easy manner, without synchronization with a scheme directory for example;
- The attribute Reference@URI must contain the value of the of @ID attribute of Assertion to make the reference to the object being signed;
- Merchants shall only trust and process SAML Assertions signed with a valid Validation Service certificate issued under the trusted root for iDIN Issuers.

10.3 SAML EncryptedID and EncryptedAttribute

For privacy reasons the Routing Service is prevented to see the consumer attributes in readable form. Hence, all consumer data inside the SAML 2.0 assertion are encrypted. Note that the Merchant is allowed to use a different certificate to sign the messages than it uses to decrypt the attributes. For the encryption of both the EncryptedID element and the EncryptedAttribute elements within the SAML Assertion the following requirements apply:

- Encryption of the attributes will be based on a 256 bit AES key;
  - Encryption of the attributes will be performed using the http://www.w3.org/2001/04/xmlenc#aes256-cbc algorithm;
  - Standard XML padding is used[17];
- A Validation Service must generate a new AES key for each EncryptedAttribute and EncryptedID elements;
  - The AES keys are encrypted with the public key of the Merchant, which the Validation Service received from the Routing Service. Encryption of the AES keys is done with RSA in combination with OAEP: http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p;
- XML contents in the encrypted element have all relevant namespace definitions.

The NameID element containing either the `Consumer.BIN` or `Consumer.TransientID` is entirely encrypted (for the corresponding namespaces see Table 6). The following element containing the `Consumer.BIN` is encrypted:

```
<saml:NameID>%Consumer.BIN%</saml:NameID>
```

Encrypting the NameID element yields an EncryptedID in the following fashion. The placeholders (between % signs) indicate where the encrypted AES key and where the encrypted NameID can be found. The EncryptedKey element is embedded inside the EncryptedData element.

```
<saml:EncryptedID xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
                      Type="http://www.w3.org/2001/04/xmlenc#Element">
```

[17] http://www.w3.org/TR/xmlenc-core/#sec-Alg-Block

```
    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes256-cbc"/>
    <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
       <xenc:EncryptedKey Recipient=%Merchant.LegalID%>
          <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p">
             <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
          </xenc:EncryptionMethod>
          <xenc:CipherData>
             <xenc:CipherValue>%AESKey_Encrypted_With_Public_Key_Merchant%</xenc:CipherValue>
          </xenc:CipherData>
       </xenc:EncryptedKey>
    </ds:KeyInfo>
    <xenc:CipherData>
       <xenc:CipherValue>%NameID_Encrypted_With_AESKey%</xenc:CipherValue>
    </xenc:CipherData>
  </xenc:EncryptedData>
</saml:EncryptedID>
```

In a similar fashion the consumer attributes in the SAML Assertion are encrypted. An example below shows how the `consumer.dateofbirth` is encrypted. Depending on the request of the Merchant, zero or more EncryptedAttributes are returned in the SAML Assertion.

```
<Attribute Name="urn:nl:bvn:bankid:1.0:consumer.dateofbirth">
  <AttributeValue>19850101<AttributeValue>
<Attribute>
```

The entire attribute is encrypted which yields the following:

```
<saml:EncryptedAttribute xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
                      Type="http://www.w3.org/2001/04/xmlenc#Element">
    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes256-cbc"/>
    <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
       <xenc:EncryptedKey Recipient=%Merchant.LegalID%>
          <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p">
             <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
          </xenc:EncryptionMethod>
          <xenc:CipherData>
             <xenc:CipherValue>%AESKey_Encrypted_With_Public_Key_Merchant%</xenc:CipherValue>
          </xenc:CipherData>
       </xenc:EncryptedKey>
    </ds:KeyInfo>
    <xenc:CipherData>
       <xenc:CipherValue>%Attribute_Encrypted_With_AES_Key%</xenc:CipherValue>
    </xenc:CipherData>
  </xenc:EncryptedData>
</saml:EncryptedAttribute>
```

10.4 Authentication of iDIN messages

To ensure the status of a transaction, the Merchant has to verify the signature of the Routing Service in all Response messages. Furthermore it has to verify the Signature of the Validation Service on the

Assertion. To verify the signature in the SignatureValue field, it is recommended that Merchants use standard XML Digital Signature libraries, which are available in most (web) programming languages.

10.5 Creating a key pair

If you want to use a so-called “self signed certificate” this paragraph will explain how to do so. It is also possible to purchase a certificate at a company specialized in this field (Certificate Authority), see paragraph 10.5.1.

In order to create a public and a private key execute the following steps:

1. Download the “OpenSSL Library” from http://www.openssl.org. You can find more information on the “certificate generating utility” at: http://www.openssl.org/docs/apps/req.html. You may also generate the key pair using other software. If so please use the manual that comes with your software.
2. Generate an “RSA private key” using the following command
   (choose your own password for the field [privateKeyPass]):

```
openssl genrsa -aes-128 -out priv.pem -passout pass:[privateKeyPass] 2048
```

3. Create a certificate based on the “RSA private key”
   (use the same password as in the previous step for the field [privateKeyPass]):

```
openssl req -x509 -sha256 -new -key priv.pem -passin pass:[privateKeyPass]
-days 1825 -out cert.cer
```

4. The previous OpenSSL command will generate a certificate in X.509 format, with a validity period of 5 years (1825 days), the maximum for iDIN signing certificates.
5. The file priv.pem contains the private key. The file cert.cer contains the certificate with the public key. The Merchant has to keep the priv.pem file private, which is used in the RSA encryption. The cert.cer file has to be communicated to the Routing Service. The method of communication will depend on the Routing Service.

10.5.1 Buying a certificate from a Certificate Authority

When buying a certificate from a Certificate Authority (CA), rather than generating the certificate yourself, it is important to note the following: the CA signing certificate (and the rest of the certificate chain) must use hashing algorithms and key lengths that are at least as secure or better than those of the Merchant certificate. Therefore CA-certificates used to sign certificates for electronic signatures must use at least SHA-256 for hashing and 2,048 bits for RSA keys. Signing certificates should also have a maximum validity period of 5 years.

10.6 Signature data elements

All messages, including the error messages, with the exception of redirects, are signed using an electronic signature. The electronic signature guarantees the authenticity of the sender, non-repudiation and the integrity of the message. The digital signature is an XML Signature data element that is defined in the XML-Signature Syntax and Processing W3C Recommendation 12 February 2002. See Table 7 for the Schema location. All attributes and elements of the Signature element are listed in the Table 31.

The elements and attributes relevant for creating the signature in all iDx messages and SAML Signature are described below; **other elements should not be used.**

Element/attributes	Mandatory	Contents
Signature	Yes	Root
+ SignedInfo	Yes	Contains SignedInfo sub-elements
++ CanonicalizationMethod	Yes	Must have one attribute
++ @algorithm	Yes	The XML content that will be signed has to be canonicalized. Canonicalization (c14n) is a process for converting data that has more than one possible representation into a canonical form. **Must be:** "http://www.w3.org/2001/10/xml-exc-c14n#"
++ SignatureMethod	Yes	Must have one attribute
++ @algorithm		For all signing RSA with SHA256 must be used as signature algorithm. **Must be:** "http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"
++ Reference	Yes	Contains Reference sub-elements
++ Reference@URI	Yes	Attribute of Reference which must be empty. This indicates that the entire XML document will be signed. **must be:** ""
+++ Transforms	Yes	Contains Transforms sub-elements. This is a list of Transform elements, each of which specifies a processing step before feeding the document to the digest algorithm. All messages use an enveloped signature: the signature is contained within the signed document. A transform is required to remove the signature from the signed data
++++ Transform		Must have one attribute
++++ Transform@algorithm		**Must be: "**http://www.w3.org/2000/09/xmldsig#enveloped-signature"
++++ Transform		Must have one attribute
++++ Transform@algorithm		**Must be:** "http://www.w3.org/2001/10/xml-exc-c14n#"
+++ DigestMethod		Must have one attribute
+++ @algorithm		This element specifies the hashing algorithm used (SHA256). **Must be:** "http://www.w3.org/2001/04/xmldsig-more#rsa-sha256
+++ DigestValue	Yes	The base64 value of the hash of the entire document
+ SignatureValue	Yes	The value of the electronic signature of the Merchant
+ KeyInfo	Yes	Contains KeyInfo sub-elements
++ KeyName	Yes	This value holds the fingerprint which indicates the certificate to be used for validating the signature. The certificate must be referenced using a fingerprint of the X.509 certificate of the Merchant, Acquirer or Issuer alike. The fingerprint must be calculated according to the following formula HEX(SHA-1(DER certificate))

Table 31: Elements/attributes of the Signature

The signing of the SAML 2.0 Assertion is different that instead of the fingerprint of the X.509 certificate the entire certificate of the Issuer is added. Also the @URI attribute of the Reference element should reference to the @ID attribute of the Assertion. This yields to the following change with respect to the Signature, while all other elements and attributes are the same as in Table 31:

Element/attributes	Mandatory	Contents
Signature	Yes	Root

Element/attributes	Mandatory	Contents
+ SignedInfo /Reference@URI	Yes	Must reference to the Assertion element. **Must be**: The value of the @ID attribute of the Assertion.
+ KeyInfo	Yes	Contains KeyInfo sub-elements
++ X509Data	Yes	Contains X509Data sub-elements
+++ X509Certificate	Yes	Contains entire X.509 certificate of the Validation Service.

Table 32: Signature changes with respect to signing the SAML Assertion

11 Presentation of iDIN

11.1 General

There are some requirements regarding the presentation of iDIN on the Merchant's website. The main purpose of these requirements is to create a uniform user experience for Consumers whenever they use iDIN, regardless of which Merchant's website they use. The requirements are further explained in the following paragraphs.

The Merchant bears primary responsibility for initiating the iDIN process and for the communication to the Consumer regarding the status of this process. Merchants who offer iDIN to their customers must include iDIN in their lists of authentication methods (if any). iDIN must be presented in the list of methods in such a way that it receives at least the same amount of attention as all other competitors.

In addition, an iDIN transaction (of any kind) must always be recognisable as such to the Consumer. This means that a Merchant must make sure in the design of their implementation of iDIN that iDIN and the start of an iDIN process are recognisable as such. The Merchant must also distinguish clearly between the processes (authentication, gather attributes or age verification) of different iDIN services.

11.2 Transaction flow

When the Consumer chooses to use iDIN, the Consumer should immediately be presented with the Issuer selection list without any intermediate screens being displayed by the Merchant (e.g. Consumer login and/or registration screens). And when the Consumer has selected the required Issuer, he or she should be immediately redirected to the online bank environment of the selected Issuer (based on the `issuerAuthenticationURL` the Merchant has received in the AcquirerTrxRes).

11.3 Redirect to Issuer

The Merchant needs to perform the redirect to the Issuer, from the browser window where the Consumer selected the Issuer. The complete page of the Merchant will be replaced by the complete page of the selected bank. Therefore it is not allowed to open the redirect to the Issuer in a new browser window. However, it is allowed to open a new window, with visible address bar, before the Consumer selects his bank from the Issuer list.

11.4 Frames

Frames used on the Merchant's site are allowed. The page of the Issuer will remove these frames using a frame busting technique. This will allow the Consumer to verify whether the transaction is really taking place at the bank chosen from the Issuer list. After the redirect to the Merchant, the Merchant must completely rebuild the Merchant page to show the Merchant's confirmation of reception of the iDIN Authentication/attributes.

11.5 New Window

The iDIN authentication may take place in a new browser window, as long as the Merchant makes this window appear at (or before) the moment the Consumer chooses the authentication method. A new

window is only allowed if initiated by the Consumer (no pop-ups are allowed). The complete transaction flow must take place in this window, including the Merchant's confirmation of receiving the iDIN authentication/attributes. The new window must also contain an address bar that allows the Consumer to check the Internet address URL and SSL-certificate of the Issuer. During the transaction flow it should not be possible for the Consumer to initiate another transaction through the Merchant's original browser window.

11.5.1 Specific Requirements iDIN Mobile: New window or app

The Mobile iDIN process may redirect the Consumer to a different mobile page or app as part of the transaction. The Merchant should strive to keep the Consumer on one browser page as much as possible but it is not allowed to make use of an in-app browser in the Merchant app (see Chapter 7 for more details). In those cases where changing to another app or window is necessary (such as the redirect to the Issuer) the Consumer should be informed beforehand in order to avoid confusion (e.g.: *"You are being redirected to the (mobile) website or the app of your bank" or in Dutch "U wordt doorverwezen naar de (mobiele) website of app van uw bank"*).

11.6 Issuer list

The Issuer list has to be presented as described in Section 6.4.

11.7 Banners and logo's

This information shall be made available on idin.nl.

11.8 Requirements and recommendation for Merchant screens

This section describes requirements and recommendation for iDIN related texts on the Merchant website. Note that some information is only available in Dutch.

11.8.1 Display last login

If a Consumer logs into the website of the Merchant using iDIN, the Merchant must show the date and time of the previous login, e.g. 'Welcome, the last time you were logged in was on October 5 2015 at 15:41'. The Merchant may determine the exact text. This makes it possible for the Consumer to verify if this matches the moment of his/her last visit.

11.8.2 Explaining iDIN to Consumers

Merchants can use the following texts to explain iDIN to their users. These texts are only available in Dutch.

Explanation of iDIN to the Consumer

- **Short version:** Makkelijk en veilig online identificeren met uw bank.
  Long version (preferred): Met iDIN kunt u zich online identificeren bij een bedrijf of instelling. Gemakkelijk, vertrouwd en veilig met de inlogmethode van uw bank.

Explanation of the advantages of iDIN

1. Makkelijk en veilig online identificeren.
2. Met de vertrouwde inlogmethode van uw bank.
3. Eén manier van inloggen bij bedrijven en instellingen.
4. Geen aparte gebruikersnamen en wachtwoorden meer onthouden.
5. Zelf invullen van persoonlijke gegevens is niet meer nodig.

11.8.3 Recommended texts on Merchant website per RequestedServiceID

iDIN provides different types of services, which have been divided into the following four use cases:

1. **Gegevens verstrekken:** The Consumer chooses iDIN to provide attributes and/or an ID to the Merchant, with or without the creation of a user account. The Merchant requests this service, either with or without the BIN. This also includes for example the provisioning of the date of birth or name/address attributes with/without BIN;
2. **Inloggen:** The Consumer uses iDIN to login. No attributes are provided to the Merchant, just the BIN;
3. **Leeftijd bevestigen:** Verification that the age is above 18 year (the attribute `18orOlder` is requested, with or without the BIN). This can also be used for verification of the age below 18 years;
4. **Bankbevestiging:** The Consumer verifies to the Merchant that he/she is a customer at a particular Bank. Only the TransientID is provided.

The mapping of all services (which depend on the `RequestedServiceID`) to the appropriate use cases is provided in APPENDIX B: Requested- and DeliveredServiceID values).

Table 33 shows the recommended texts for the Merchant website. The referral page is the webpage where the Consumer chooses to use iDIN. The returnpage is the webpage to which the Consumer returns, after he has approved the iDIN transaction at his own bank.

Use case	Recommended text for referral webpage Merchant	Recommended text for returnpage Merchant
1 Gegevens verstrekken	• Maak uw account aan met iDIN • Gegevens verstrekken/versturen met iDIN	• Uw gegevens zijn succesvol ontvangen • Wij hebben de volgende gegevens ontvangen: [overzicht gegevens]
2 Inloggen	• Inloggen met iDIN	• U bent ingelogd met iDIN • Tonen laatste inlog (**verplicht**)
3 Leeftijd bevestigen	• Leeftijd bevestigen met iDIN	• Bedankt voor het bevestigen van uw leeftijd • Direct toegang tot de site
4 Bankbevestiging	• Bevestig dat u klant bent bij uw bank met iDIN	• U heeft bevestigd dat u klant bent bij `Issuer.Name`

Table 33: Recommendation Merchant texts per use case

11.9 Validation Service front-end

The following described process is outside the scope of the Merchant implementation. It is provided in this document to give insight in the Consumer's experience of iDIN.

The Issuer (Validation Service) bears primary responsibility for handling the iDIN process and for communicating with the Consumer regarding the status of the process of the iDIN transaction. The page sequence and layout (from redirect from Merchant to Issuer until redirect back to Merchant) are determined by the Validation Service. The following conditions must be met:

- After selecting his Issuer at the Merchant website, the Consumer is redirected to the selected Issuer's Validation Service website. In the active browser window, the complete page of the Merchant is replaced by the complete page of the selected Validation Service. Alternatively the Validation Service can choose, based on the header information received from the Consumer, to automatically redirect the Consumer to its mobile payment website or mobile payment app. Criteria for choosing which Consumers/devices are redirected to which channels are left to the Validation Service;
- Optionally, the Validation Service may allow the Consumer to choose a different channel from the one chosen automatically, if the Validation Service offers multiple channels available to the Consumer. If this option is offered and used by the Consumer a second redirect takes place to the authentication website or application of choice;
- At all times during an iDIN process, it must be clear to the Consumer that he is about to start an iDIN transaction. Therefor, the Validation Service must display the iDIN logo on every Validation Service web/mobile application page shown in the iDIN process;
- The Validation Service will not display any information unrelated and irrelevant to the iDIN transaction, that could distract the Consumer from completing the process on the website or in the mobile application. Information that is considered relevant is:
    - A help function offered by the Validation Service, providing the Consumer with an explanation or additional information regarding the iDIN service(s);
    - If the Issuer has a mobile application available then this may be offered for immediate download and activation during the iDIN process to Consumers that have been detected as using a mobile device but who have not yet downloaded the app. Note: downloading an application during the iDIN process may very well result in the process expiring if it exceeds the expiration period set by the Merchant;
- The Validation Service facilitates all services of iDIN. The Validation Service must clearly indicate to the Consumer what iDIN service the Consumer is about to consent to;
- The Validation Service will guarantee that the integrity and layout of the Validation Service website/mobile application is not altered when displaying the contents of textual fields (e.g. due to possible malicious content of the transaction information provided by a malicious Merchant);
- All iDIN transaction related information (i.e. Transaction information, Consumer information and Merchant information, must be presented to the Consumer for approval;
- During the iDIN transaction process, the Consumer can either consent to the provision of all attributes (so not per attribute set or individual attribute) or reject the entire request;
- During the iDIN transaction process, the Consumer is not allowed to change any of the information in the transaction details nor is he/she allowed to give partial consent;

- The Validation Service indicates clearly how the Consumer can approve and give his/her consent for the iDIN transaction;
- The Consumer must be able to see all attributes and their values in order to give informed consent for providing the attributes to the Merchant. In order to retrieve this information, the Consumer must first be logged into Validation Service system. This enables the Validation Service to identify the Consumer and retrieve the right information;
- The Validation Service indicates clearly how the Consumer can cancel the iDIN process;
- As an option, the Validation Service iDIN application may show a message to the Consumer when he/she tries to cancel the iDIN transaction by using the browser close button or tries to navigate backwards or forwards by using the browser arrow buttons. The message may be shown because this can be considered undesired Consumer behaviour. In this message, the Consumer can be advised to cancel or complete the iDIN transaction in a proper way, using the cancel or confirmation facilities of the Validation Service iDIN application;
- In case the Validation Service cannot suffice to the minimal requested attribute set, it may state to the Consumer what attributes are missing that are required to successfully complete the transaction and shorty explains which procedure must be followed to add in the missing information. The Validation Service is free, in choice and implementation, to provide instructions on how the Consumer can change his/her attributes;
- After approval of the iDIN transaction, the Consumer must have the opportunity to instantaneously return to the Merchant where the iDIN transaction was initiated. This is achieved by incorporating a 'Continue' button. Clicking on this button takes the Consumer to the Merchant's return URL provided during transaction initiation by the Merchant.

11.9.1 Approval information

The Validation Service clearly presents all the details of the iDIN transaction that must be approved by the Consumer. The following Merchant, Consumer and iDIN transaction information **must** be displayed, unless indicated otherwise (exact presentation to be determined by the Validation Service). As iDIN provides different types of services it is important to specify the display requirements per service (authentication, providing attributes and signing):

Merchant information[18]:

- `Merchant.Name`
- `Merchant.TradeName` (only when provided)

At all times it must be clear to the Consumer that the iDIN transaction approval is issued to the `Merchant.Name` and not to the `Merchant.TradeName`.

Consumer information:

`Consumer.*` (depending on the requested attributes)

[18] `Merchant.Name` and `Merchant.TradeName` are not used by the Merchant itself, hence they are not shown in the data dictionary in Chapter 5.

Note: `Consumer.TransientID`, `Consumer.BIN` and `Consumer.DeprecatedBIN` are not shown.

The following standard text must also appear with the approval information, depending on the service requested by the Merchant:

1. **Accompanying text authentication for services requesting only a transient ID** (mandatory text)
   U bevestigt aan [`Merchant.Name`] dat u klant bent bij [`Issuer.issuerName`] .
2. **Accompanying text authentication for services requesting only a BIN (mandatory text)**
   U gaat inloggen bij [`Merchant.Name`].
3. **Accompanying text authentication for services requesting a transient ID and one or more attribute sets (mandatory text)**
   [`Merchant.Name`] heeft gevraagd om de volgende gegevens van u te mogen ontvangen:
4. **Accompanying text authentication for services requesting a BIN and one or more attribute sets (mandatory text)**
   U gaat inloggen bij [`Merchant.Name`]. [`Merchant.Name`] heeft gevraagd om de volgende gegevens van u te mogen ontvangen:

When `Merchant.TradeName` is provided to the Issuer the above text should be changed to:

1. **Accompanying text authentication for services requesting only a transient ID (mandatory text)**
   U bevestigt aan [`Merchant.Name`] inzake [`Merchant.TradeName`] dat u klant bent bij [`Issuer.issuerName`].
2. **Accompanying text authentication for services requesting only a BIN (mandatory text)**
   U gaat inloggen bij [`Merchant.Name`] inzake [`Merchant.TradeName`].
3. **Accompanying text authentication for services requesting a transient ID and one or more attribute sets (mandatory text)**
   [Merchant.Name] heeft namens [`Merchant.TradeName`] gevraagd om de volgende gegevens van u te mogen ontvangen:
4. **Accompanying text authentication for services requesting a BIN and one or more attribute sets (mandatory text)**
   U gaat inloggen bij [`Merchant.Name`] inzake [`Merchant.TradeName`].
   [`Merchant.Name`] heeft namens [`Merchant.TradeName`] gevraagd om de volgende gegevens van u te mogen ontvangen:

Please note that the above texts are subject to change prior to or during the pilot.

The following screenshot provides an example of the iDIN approval website or mobile application screen for authentication service plus providing attributes.

Figure 5: Example of transaction approval or mobile app screen

12 APPENDIX A: Error codes and cases

In case something goes wrong on iDx level an AcquirerErrorRes with the appropriate errorCode is returned to the Merchant. If there is an error within the SAML messages an iDx errorCode AP3000 is return and a container is present with an SAML (error) Response, as outlined in Section 9.2.

12.1 iDx error codes

The `Error.errorCode` is composed of:

1 A category (two letters)
2 A number (four digits)

The following categories are distinguished:

Category	Meaning
IX	Invalid XML and all related problems. Such as incorrect encoding, invalid version, otherwise unreadable.
SO	System maintenance. The errors that are communicated in the event of system maintenance or system failure. Also covers the situation where new requests are no longer being accepted but requests already submitted will be dealt with (until a certain time).
SE	Security and authentication errors. Incorrect authentication methods and expired certificates.
BR	Field errors. Additional information on incorrect fields.
AP	Application errors. Errors relating to IDs, transaction, expiration period and iDIN specific errors

Table 34: Error code categories

The following iDx error codes exist:

errorCode	errorMessage	errorDetail	Occurs in
IX1100	Received XML not valid	See 1)	A'(X), B'(X), F'(X)
IX1200	Encoding type not UTF-8	See 1)	A'(X), B'(X), F'(X)
IX1300	XML version number invalid	See 1)	A'(X), B'(X), F'(X)
IX1600	Mandatory value missing	See 1)	A'(X), B'(X), F'(X)
SO1000	Failure in system	See 2)	A'(X), B'(X), F'(X)
SO1100	Issuer unavailable	See 3)	B'(X), F'(X)
SO1200	System busy. Try again later	See 2)	A'(X), B'(X), F'(X)
SO1400	Unavailable due to maintenance	See 2)	A'(X), B'(X), F'(X)
SE2000	Authentication error	See 1)	A'(X), B'(X), F'(X)
SE2100	Authentication method not supported	See 1)	A'(X), B'(X), F'(X)
BR1200	Version number invalid	See 1)	A'(X), B'(X), F'(X)
BR1205	ProductID invalid	See 1)	A'(X), B'(X), F'(X)
BR1210	Value contains non-permitted character	See 1)	A'(X), B'(X), F'(X)

errorCode	errorMessage	errorDetail	Occurs in
BR1220	Value too long	See 1)	A'(X), B'(X), F'(X)
BR1230	Value too short	See 1)	A'(X), B'(X), F'(X)
BR1270	Invalid date/time	See 1)	A'(X), B'(X), F'(X)
BR1280	Invalid URL	See 1)	B'(X)
AP1100	`Merchant.MerchantID` unknown	See 1)	A'(X), B'(X), F'(X)
AP1200	`Issuer.IssuerID` unknown	See 1)	B'(X)
AP1300	`Merchant.subID` unknown	See 1)	A'(X), B'(X), F'(X)
AP1500	`Merchant.MerchantID` not active	See 1)	A'(X), B'(X), F'(X)
AP2600	Transaction does not exist	See 1)	F'(X)
AP2920	Expiration period is not valid	See 1)	B'(X)
AP3000	iDIN specific error	See 1)	A'(X), B'(X),

Table 35: Error codes

The field `errorDetail` in the table above contains one of the values shown in the table below. The italic printed words will be replaced by actual values, as indicated.

Indication	errorDetail
1)	Field generating error: location-reference in XML message
2)	System generating error: *Issuer/Acquirer*
3)	System generating error: *Name of Issuer*

Table 36: errorDetail

12.2 SAML error codes

To communicate to the Merchant that an error has occurred in the SAML message, the SAML Status element is used which has the following structure:

```
<samlp:Status>
  <StatusCode Name=%First level status code% />
     <StatusCode Name=%Second level status code% />
  <StatusMessage>Text determined by Routing or Validation Service</StatusMessage>
</samlp:Status>
```

SAML defines two level of status codes. The first level status code can have the values as shown in the Table below.

StatusCode@Value	Description
urn:oasis:names:tc:SAML:2.0:status:Success	The request succeeded. Not used in case an error has occured
urn:oasis:names:tc:SAML:2.0:status:Requester	The request could not be performed due to an error on the part of the requester.

Table 37: First level SAML status codes

The second level status code is present for the cases as discussed in Section 9.2. There are several standard values for the SAML second level status code, and some defined values that are only used within the iDIN domain. The possible values for the second level status code within iDIN are listed in the Table below.

StatusCode@Value	Description
urn:oasis:names:tc:SAML:2.0:status: RequestUnsupported	The request could not be executed because the requested `BankID.ServiceID` or `BankID.LOA` is not supported. This value is only used in combination with the first level status code value urn:oasis:names:tc:SAML:2.0:status:Requester
urn:oasis:names:tc:SAML:2.0:status: InvalidAttrNameOrValue	The request could not be executed because unexpected or invalid content was encountered within a SAML element. It is used when the contents or attribute value of a SAML element is not in line with the iDIN specifications, other than `BankID.ServiceID` or `BankID.LOA`. E.g. wrong formatting, incorrect version etc. This value is only used in combination with the first level status code value urn:oasis:names:tc:SAML:2.0:status:Requester
urn:nl:bvn:bankid:1.0:status: MismatchWithIDx *This status code is made specifically for iDIN*	The request could not be executed because one or more fields inside the SAML AuthnRequest do not match the fields in the iDx message as required by the iDIN specifications. E.g. the `MerchantID` in the SAML AuthnRequest does not correspond to that in the iDx message. This value is only used in combination with the first level status code value urn:oasis:names:tc:SAML:2.0:status:Requester
urn:oasis:names:tc:SAML:2.0:status: RequestDenied	The request is denied because the Assertion is expired. See next section for the correct use. This value is only used in combination with the first level status code value urn:oasis:names:tc:SAML:2.0:status:Requester
urn:nl:bvn:bankid:1.0:status:Success *This status code is made specifically for iDIN*	Present when the Validation Service has delivered all attributes conform the minimal set as defined in Section 5.5
urn:nl:bvn:bankid:1.0:status: IncompleteAttributeSet *This status code is made specifically for iDIN*	The request has been delivered, however not all of the requested Consumer attributes could be delivered according to the minimal set as defined in Section 5.5. The element `DeliveredServiceID` either indicates which attributes are delivered conform the minimal set, or is equal to '0' if the Issuer was unable to determine this. In both cases it is not equal to the `RequestedServiceID`

Table 38: Second level SAML status codes

12.3 SAML error cases

There are two cases where a SAML Response is returned with a first level status code not equal to "urn:oasis:names:tc:SAML:2.0:Success" but equal to "urn:oasis:names:tc:SAML:2.0:Requester".

1. There is some type of error in the AuthnRequest between the Merchant and the Routing Service. This AuthnRequest is embedded inside the container of the AcquirerTrxReq (B):

   For this case on iDx level an AcquirerErrorRes (B'(X)) is returned to the Merchant with a SAML Response in the container. The iDx errorCode equals AP3000 with an errorMessage 'Product specific error'. The second level status code depends on the type of error that has occurred, see above table.

2. The Merchant has requested the Status, however the Assertion is expired. This can occur when the Consumer has successfully authenticated and approved the transaction, however the

Merchant has not succeeded in requesting the status within 30 seconds from the moment the Consumer has returned to the website of the Merchant.

For this case a normal AcquirerStatusRes is returned to the Merchant. The iDx `Transaction.status` of this message equals “Success”. However the SAML Response does not contain an Assertion, but has the structure as discussed in Table 28: Elements/attributes inside the container of AcquirerErrorRes. It has a second level status code value of “urn:oasis:names:tc:SAML:2.0:status:RequestDenied”.

12.4 Issuer cannot provide all attributes conform minimal set

When the Issuer cannot deliver all requested attributes conform the minimal set, the following shall be done:

- A normal SAML Response is returned. The first level status code is “urn:oasis:names:tc:SAML:2.0:Success”;
- The Issuer returns all attributes that are available in the requested categories;
- Instead of the second SAML status code “urn:nl:bvn:bankid:1.0:status:Success” the second SAML status code “urn:nl:bvn:bankid:1.0:status:IncompleteAttributeSet” is used;
- The `DeliveredServiceID` shall indicate which attribute categories are delivered conform the minimal set as defined in Section 5.5. If the Issuer cannot determine the `DeliveredServiceID` it must use the value ‘0’.

Example:

- The Merchant has requested the date of birth and all address attributes of a Consumer with `RequestedServiceID` 1472;
- When the Issuer processes the request, its systems detects the address attributes cannot be delivered conform the minimal set as defined in Section 5.5 (it has the `consumer.postalcode`, `consumer.city`, `consumer.streetname` but is missing the `consumer.houseno`. Therefore, it cannot match any of the five possible options as defined in Section 5.5.
- The Issuer calculates the `DeliveredServiceID` that corresponds to the requested attributes categories it can deliver conform the minimal set (in this case 448 which indicates the date of birth is delivered conform the minimal set);
- The Issuer returns all requested attributes (date of birth and all attributes in the incomplete address category), and uses the second SAML status code “urn:nl:bvn:bankid:1.0:status:IncompleteAttributeSet” and the `DeliveredServiceID` equal to 448.

12.5 Consumer message

The `Error.consumerMessage` can contain one of four different standardised texts that are sent to the Merchant by the Routing Service, see Table 39. The Merchant must show the `Error.consumerMessage` to the Consumer on his website. The value of `Error.consumerMessage` is specified in the AcquirerErrorRes by the Acquirer.

Situation	Message to be shown to Merchant (English)	Message to be shown to Merchant (Dutch)
Error occurred in sending or receiving message *A*, *A'*, *B*, *B'*	It is currently not possible to use iDIN. Please try again later.	Het is op dit moment niet mogelijk om iDIN te gebruiken. Probeer het later nog een keer.
Error occurred in sending or receiving message *F*, *F''*	It is currently not possible to use iDIN. Please try again later.	Het is op dit moment niet mogelijk om iDIN te gebruiken. Probeer het later nog een keer.
Error occurred because of unavailability of Validation Service (SO1000, SO1100, SO1200 , SO1400 or no response received from Validation Service by Routing Service after sending message C)	The selected bank is currently unavailable. Please try again later.	De geselecteerde bank is op dit moment niet beschikbaar. Probeer het later nog een keer.
Error occurred because of unavailability of Validation Service (see above) AND additional information is available from the Notification System	The selected bank is currently unavailable due to maintenance until projected time of [DateTime received from the Notification System]. Please try again later.	De geselecteerde bank is op dit moment niet beschikbaar i.v.m. onderhoud tot naar verwachting [DateTime ontvangen van Notification System]. Probeer het later nog een keer.

Table 39: Consumer messages

13 APPENDIX B: Requested- and DeliveredServiceID values

The table below shows all valid combinations of the `Requested`- and `DeliveredServiceID` with corresponding integer number and use case number, see Section 11.8.

Service ID	ConsumerID		Name Attributes	Address Attributes	Age Attributes		Gender	BSN	Use case[19]
	BIN	Transient			DateOfBirth	18orOlder			
0		Yes							4
16		Yes					Yes		1
64		Yes				Yes			3
80		Yes				Yes	Yes		1
448		Yes			Yes				1
464		Yes			Yes		Yes		1
1024		Yes		Yes					1
1040		Yes		Yes			Yes		1
1088		Yes		Yes		Yes			1
1104		Yes		Yes		Yes	Yes		1
1472		Yes		Yes	Yes				1
1488		Yes		Yes	Yes		Yes		1
4096		Yes	Yes						1
4112		Yes	Yes				Yes		1
4160		Yes	Yes			Yes			1
4176		Yes	Yes			Yes	Yes		1
4544		Yes	Yes		Yes				1
4560		Yes	Yes		Yes		Yes		1
5120		Yes	Yes	Yes					1
5136		Yes	Yes	Yes			Yes		1
5184		Yes	Yes	Yes		Yes			1

[19] Use case 1: Gegevens verstrekken/versturen. Use case 2: Inloggen. Use case 3: Leeftijd bevestigen. Use case 4: Bank bevestigen.

Service ID	ConsumerID		Name Attributes	Address Attributes	Age Attributes		Gender	BSN	Use case[19]
	BIN	Transient			DateOfBirth	18orOlder			
5200		Yes	Yes	Yes		Yes	Yes		1
5568		Yes	Yes	Yes	Yes				1
5584		Yes	Yes	Yes	Yes		Yes		1
16384	Yes								2
16400	Yes						Yes		1
16448	Yes					Yes			3
16464	Yes					Yes	Yes		1
16832	Yes				Yes				1
16848	Yes				Yes		Yes		1
17408	Yes			Yes					1
17424	Yes			Yes			Yes		1
17472	Yes			Yes		Yes			1
17488	Yes			Yes		Yes	Yes		1
17856	Yes			Yes	Yes				1
17872	Yes			Yes	Yes		Yes		1
20480	Yes		Yes						1
20496	Yes		Yes				Yes		1
20544	Yes		Yes			Yes			1
20560	Yes		Yes			Yes	Yes		1
20928	Yes		Yes		Yes				1
20944	Yes		Yes		Yes		Yes		1
21504	Yes		Yes	Yes					1
21520	Yes		Yes	Yes			Yes		1
21568	Yes		Yes	Yes		Yes			1
21584	Yes		Yes	Yes		Yes	Yes		1
21952	Yes		Yes	Yes	Yes				1
21968	Yes		Yes	Yes	Yes		Yes		1

Table 40: Integer values of `Requested-` or `DeliveredServiceID` per requested attribute group

14 APPENDIX C: Message examples

Most of the example messages given here only use the default method of namespace declaration. At the end of the appendix one example is given of a message with namespace prefixes (this message does not contain an information container, it is merely meant to signify the use of namespace prefixes).
NB:

- Signatures are examples and don't validate against the messages;
- The examples are not necessarily related to each other.

14.1 DirectoryReq (A)

```
<?xml version="1.0" encoding="UTF-8"?>
<DirectoryReq version="1.0.0" productID="NL:BVN:BankID:1.0" xmlns="
http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <createDateTimestamp>2001-12-17T09:30:47.123Z</createDateTimestamp>
  <Merchant>
     <merchantID>1234123456</merchantID>
     <subID>1</subID>
  </Merchant>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
     <SignedInfo>
        <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
        <Reference URI="">
           <Transforms>
              <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
              <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
           </Transforms>
           <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
           <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
        </Reference>
     </SignedInfo>
     <SignatureValue>
IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE/GPHZShuMw+8WHq4
fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
    </SignatureValue>
     <KeyInfo>
        <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
     </KeyInfo>
  </Signature>
</DirectoryReq>
```

14.2 DirectoryRes (A')

```
<?xml version="1.0" encoding="UTF-8"?>
<DirectoryRes version="1.0.0" productID=" NL:BVN:BankID:1.0" xmlns="
http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <createDateTimestamp>2001-12-17T09:30:47.123Z</createDateTimestamp>
  <Acquirer>
     <acquirerID>1234</acquirerID>
  </Acquirer>
  <Directory>
    <directoryDateTimestamp>2004-11-10T10:15:12.123Z</directoryDateTimestamp>
    <Country>
      <countryNames>Nederland</countryNames>
      <Issuer>
        <issuerID>BANKNL2U</issuerID>
        <issuerName>Bank 1</issuerName>
      </Issuer>
      <Issuer>
        <issuerID>BANANL2U</issuerID>
        <issuerName>Bank 2</issuerName>
      </Issuer>
      <Issuer>
        <issuerID>BANBNL2UXXX</issuerID>
        <issuerName>Bank 3</issuerName>
      </Issuer>
      <Issuer>
        <issuerID>BANCNL2U</issuerID>
        <issuerName>Bank 4</issuerName>
      </Issuer>
     </Country>
    <Country>
      <countryNames>België/Belgique</countryNames>
      <Issuer>
        <issuerID>BANKBE2U</issuerID>
        <issuerName>Banque 1</issuerName>
      </Issuer>
    </Country>
  </Directory>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
      <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" />
      <Reference URI="">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
```

```
            <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256" />
          <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
        </Reference>
      </SignedInfo>
      <SignatureValue>
IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE/GPHZShuMw+8WHq4
fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
      </SignatureValue>
      <KeyInfo>
        <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
      </KeyInfo>
    </Signature>
</DirectoryRes>
```

14.3 AcquirerTrxReq (B)

```
<?xml version="1.0" encoding="UTF-8"?>
<AcquirerTrxReq version="1.0.0"
                productID="NL:BVN:BankID:1.0"
                xmlns="http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
                xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <createDateTimestamp>2015-01-01T09:30:00.123Z</createDateTimestamp>
  <Issuer>
    <issuerID>BANKNL2U</issuerID>
  </Issuer>
  <Merchant>
    <merchantID>1234123456</merchantID>
    <subID>1</subID>
<merchantReturnURL>https://merchantwebsite.nl/returnPage.php?param1=true&amp;param2=3</merchan
tReturnURL>
  </Merchant>
  <Transaction>
    <expirationPeriod>PT5M</expirationPeriod>
    <language>nl</language>
    <entranceCode>1234567890abcdefghijABCDEFGHIJ1234567890</entranceCode>
    <container>
      <samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                          xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                          AttributeConsumingServiceIndex="21952"
                          ID="REF1234567890"
                          IssueInstant="2015-01-01T09:30:00Z"
                          Version="2.0"
                          ProtocolBinding="nl:bvn:bankid:1.0:protocol:iDx"

AssertionConsumerServiceURL="https://merchantwebsite.nl/returnPage.php?param1=true&amp;param2=
3">
        <saml:Issuer>1234123456</saml:Issuer>
```

```
        <samlp:RequestedAuthnContext Comparison="minimum">
           <saml:AuthnContextClassRef>nl:bvn:bankid:1.0:loa2</saml:AuthnContextClassRef>
        </samlp:RequestedAuthnContext>
      </samlp:AuthnRequest>
    </container>
  </Transaction>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
      <Reference URI="">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
          <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
        <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
      </Reference>
    </SignedInfo>

<SignatureValue>IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE
/GPHZShuMw+8WHq4fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
</SignatureValue>
    <KeyInfo>
      <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
    </KeyInfo>
  </Signature>
</AcquirerTrxReq>
```

14.4 AcquirerTrxRes (B')

```
<?xml version="1.0" encoding="UTF-8"?>
<AcquirerTrxRes version="1.0.0" productID="NL:BVN:BankID:1.0" xmlns="
http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <createDateTimestamp>2001-12-17T09:30:47.123Z</createDateTimestamp>
  <Acquirer>
     <acquirerID>1234</acquirerID>
  </Acquirer>
  <Issuer>
  <issuerAuthenticationURL>https://issuer.nl?param=true&amp;paramRandom=123456789012345678901
34567890</issuerAuthenticationURL>
  </Issuer>
  <Transaction>
     <transactionID>1234123456789012</transactionID>
     <transactionCreateDateTimestamp>2001-12-17T09:30:47.123Z</transactionCreateDateTimestamp>
  </Transaction>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
```

```
    <SignedInfo>
       <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
       <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
       <Reference URI="">
          <Transforms>
             <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
             <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
       </Reference>
    </SignedInfo>
    <SignatureValue>
IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE/GPHZShuMw+8WHq4
fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
    </SignatureValue>
    <KeyInfo>
       <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
    </KeyInfo>
 </Signature>
</AcquirerTrxRes>
```

14.5 AcquirerStatusReq (F)

```
<?xml version="1.0" encoding="UTF-8"?>
<AcquirerStatusReq version="1.0.0" productID="NL:BVN:BankID:1.0" xmlns="
http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
 <createDateTimestamp>2001-12-17T09:30:47.123Z</createDateTimestamp>
 <Merchant>
    <merchantID>1234123456</merchantID>
    <subID>1</subID>
 </Merchant>
 <Transaction>
    <transactionID>1234123456789012</transactionID>
 </Transaction>
 <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
       <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
       <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
       <Reference URI="">
          <Transforms>
             <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
               <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
       </Reference>
```

```
    </SignedInfo>
    <SignatureValue>
IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE/GPHZShuMw+8WHq4
fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
    </SignatureValue>
    <KeyInfo>
       <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
    </KeyInfo>
 </Signature>
</AcquirerStatusReq>
```

14.6 AcquirerStatusRes (F') Unencrypted

Note: this message will never be sent/received, but is provided for illustrative purposes only.

```
<?xml version="1.0" encoding="UTF-8"?>
<AcquirerStatusRes version="1.0.0" productID="NL:BVN:BankID:1.0"
                   xmlns="http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
                   xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <createDateTimestamp>2015-01-01T09:30:47.123Z</createDateTimestamp>
  <Acquirer>
    <acquirerID>1234</acquirerID>
  </Acquirer>
  <Transaction>
    <transactionID>1234123456789012</transactionID>
    <status>Success</status>
    <statusDateTimestamp>2015-01-01T09:30:47.123Z</statusDateTimestamp>
    <container>
      <samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol" ID="RES-
1234123456789012"
                      InResponseTo="REF1234567890" Version="2.0" IssueInstant="2015-01-
01T09:30:47.123Z”>
        <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">1234</saml:Issuer>
        <samlp:Status>
          <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success">
               <samlp:StatusCode Value=”urn:nl:bvn:bankid:1.0:status:Success”/>
           </samlp:StatusCode>
        </samlp:Status>
        <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" Version="2.0"
                        ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc" IssueInstant="2015-01-
01T09:30:47.123Z”>
          <saml:Issuer>BANKNL2U</saml:Issuer>
          <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:SignedInfo>
              <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
              <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-
sha256"/>
              <ds:Reference URI="#_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
                <ds:Transforms>
```

```
                  <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"/>
                  <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                </ds:Transforms>
                <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>AA+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</ds:DigestValue>
              </ds:Reference>
            </ds:SignedInfo>
<ds:SignatureValue>AAAAwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5ayn
HBE/GPHZShuMw+8WHq4fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9v
eI=</ds:SignatureValue>
            <ds:KeyInfo>
              <ds:X509Data>
<ds:X509Certificate>MIICyjCCAjOgAwIBAgICAnUwDQYJKoZIhvcNAQEEBQAwgakxCzAJBgNVBAYTAlVT
MRIwEAYDVQQIEwlXaXNjb25zaW4xEDAOBgNVBAcTB01hZGlzb24xIDAeBgNVBAoT
F1VuaXZlcnNpdHkgb2YgV2lzY29uc2luMSswKQYDVQQLEyJEaXZpc2lvbiBvZiBJ
bmZvcm1hdGlvbiBUZWNobm9sb2d5MSUwIwYDVQQDExxIRVBLSSBTZXJ2ZXIgQ0Eg
LS0gMjAwMjA3MDFBMB4XDTAyMDcyNjA3Mjc1MVoXDTA2MDkwNDA3Mjc1MVowgYsx
CzAJBgNVBAYTAlVTMREwDwYDVQQIEwhNaWNoaWdhbjESMBAGA1UEBxMJQW5uIEFy
Ym9yMQ4wDAYDVQQKEwVVQ0FJRDEcMBoGA1UEAxMTc2hpYjEuaW50ZXJuZXQyLmVk
dTEnMCUGCSqGSIb3DQEJARYYcm9vdEBzaGliMS5pbnRlcm5ldDIuZWR1MIGfMA0G
CSqGSIb3DQEBAQUAA4GNADCBiQKBgQDZSAb2sxvhAXnXVIVTx8vuRay+x50z7GJj
IHRYQgIv6IqaGG04eTcyVMhoekE0b45QgvBIaOAPSZBl13R6+KYiE7x4XAWIrCP+
c2MZVeXeTgV3Yz+USLg2Y1on+Jh4HxwkPFmZBctyXiUr6DxF8rvoP9W7O27rhRjE
pmqOIfGTWQIDAQABox0wGzAMBgNVHRMBAf8EAjAAMAsGA1UdDwQEAwIFoDANBgkq
hkiG9w0BAQQFAAOBgQBfDqEW+OI3jqBQHIBzhujN/PizdN7s/z4D5d3pptWDJf2n
qgi7lFV6MDkhmTvTqBtjmNk3No7v/dnP6Hr7wHxvCCRwubnmIfZ6QZAv2FU78pLX
8I3bsbmRAUg4UP9hH6ABVq4KQKMknxu1xQxLhpR1ylGPdiowMNTrEG8cCx3w/w==</ds:X509Certificate>
              </ds:X509Data>
            </ds:KeyInfo>
          </ds:Signature>
          <saml:Subject>
            <saml:NameID>NLBANKsd45232432663dd34ja8sjsah439h28834HSh23h192h3</saml:NameID>
          </saml:Subject>
          <saml:Conditions NotBefore="2015-01-01T09:30:47Z" NotOnOrAfter="2015-01-
01T09:31:17.123Z”>
            <saml:AudienceRestriction>
              <saml:Audience>NL00ZZZ12345678</saml:Audience>
            </saml:AudienceRestriction>
            <saml:OneTimeUse></saml:OneTimeUse>
          </saml:Conditions>
          <saml:AuthnStatement AuthnInstant="2015-01-01T09:30:47.123Z”>
            <saml:AuthnContext>
              <saml:AuthnContextClassRef>nl:bvn:bankid:1.0:loa2</saml:AuthnContextClassRef>
              <saml:AuthenticatingAuthority>BANKNL2U</saml:AuthenticatingAuthority>
            </saml:AuthnContext>
          </saml:AuthnStatement>
          <saml:AttributeStatement>
           <saml:Attribute Name=”urn:nl:bvn:bankid:1.0:bankid.deliveredserviceid">
```

```
            <saml:AttributeValue>21952</saml:AttributeValue>
          </saml:Attribute>
          </saml:Attribute>
            <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.gender">
            <saml:AttributeValue>1</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.legallastname">
            <saml:AttributeValue>Vries</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.preferredlastname">
            <saml:AttributeValue>Vries-Jansen</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.partnerlastname">
            <saml:AttributeValue>Jansen</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.legallastnameprefix">
            <saml:AttributeValue>de</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.preferredlastnameprefix">
            <saml:AttributeValue>de</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.initials">
            <saml:AttributeValue>JV</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.dateofbirth">
            <saml:AttributeValue>19850101</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.street">
            <saml:AttributeValue>Gustav Mahlerplein</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.houseno">
            <saml:AttributeValue>33</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.housenosuf">
            <saml:AttributeValue>bis</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.addressextra">
            <saml:AttributeValue>woonboot t.o. de Albert Heijn</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.postalcode">
            <saml:AttributeValue>1082MS</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.city">
            <saml:AttributeValue>Amsterdam</saml:AttributeValue>
          </saml:Attribute>
          <saml:Attribute Name="urn:nl:bvn:bankid:1.0:consumer.country">
            <saml:AttributeValue>NL</saml:AttributeValue>
          </saml:Attribute>
        </saml:AttributeStatement>
```

```
        </saml:Assertion>
      </samlp:Response>
    </container>
  </Transaction>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
      <Reference URI="">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
          <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
        <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
      </Reference>
    </SignedInfo>
<SignatureValue>IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE
/GPHZShuMw+8WHq4fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
</SignatureValue>
    <KeyInfo>
    <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
    </KeyInfo>
  </Signature>
</AcquirerStatusRes>
```

14.7 AcquirerStatusRes (F') Encrypted

Note: this example only shows the first two encrypted attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<AcquirerStatusRes version="1.0.0" productID="NL:BVN:BankID:1.0"
                   xmlns="http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
                   xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <createDateTimestamp>2015-01-01T09:30:47.123Z</createDateTimestamp>
  <Acquirer>
    <acquirerID>1234</acquirerID>
  </Acquirer>
  <Transaction>
    <transactionID>1234123456789012</transactionID>
    <status>Success</status>
    <statusDateTimestamp>2015-01-01T09:30:47.123Z</statusDateTimestamp>
    <container>
      <samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol" ID="RES-
1234123456789012"
                      InResponseTo="REF1234567890" Version="2.0" IssueInstant="2015-01-
01T09:30:47.123Z”>
        <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">1234</saml:Issuer>
        <samlp:Status>
```

```
          <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success">
               <samlp:StatusCode Value=”urn:nl:bvn:bankid:1.0:status:Success”/>
           </samlp:StatusCode>
        </samlp:Status>
        <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" Version="2.0"
                        ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc" IssueInstant="2015-01-
01T09:30:47.123Z”>
          <saml:Issuer>BANKNL2U</saml:Issuer>
          <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:SignedInfo>
              <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
              <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-
sha256"/>
              <ds:Reference URI="#_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
                <ds:Transforms>
                  <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"/>
                  <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                </ds:Transforms>
                <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>AA+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</ds:DigestValue>
              </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>VGhpcyBpcyBhIHRlc3QgbWVzc2FnZSE=</ds:SignatureValue>
            <ds:KeyInfo>
                  <ds:X509Data>
<ds:X509Certificate>MIICyjCCAjOgAwIBAgICAnUwDQYJKoZIhvcNAQEEBQAwgakxCzAJBgNVBAYTAlVT
MRIwEAYDVQQIEwlXaXNjb25zaW4xEDAOBgNVBAcTB01hZGlzb24xIDAeBgNVBAoT
F1VuaXZlcnNpdHkgb2YgV2lzY29uc2luMSswKQYDVQQLEyJEaXZpc2lvbiBvZiBJ
bmZvcm1hdGlvbiBUZWNobm9sb2d5MSUwIwYDVQQDExxIRVBLSSBTZXJ2ZXIgQ0Eg
LS0gMjAwMjA3MDFBMB4XDTAyMDcyNjA3Mjc1MVoXDTA2MDkwNDA3Mjc1MVowgYsx
CzAJBgNVBAYTAlVTMREwDwYDVQQIEwhNaWNoaWdhbjESMBAGA1UEBxMJQW5uIEFy
Ym9yMQ4wDAYDVQQKEwVVQ0FJRDEcMBoGA1UEAxMTc2hpYjEuaW50ZXJuZXQyLmVk
dTEnMCUGCSqGSIb3DQEJARYYcm9vdEBzaGliMS5pbnRlcm5ldDIuZWR1MIGfMA0G

CSqGSIb3DQEBAQUAA4GNADCBiQKBgQDZSAb2sxvhAXnXVIVTx8vuRay+x50z7GJj
IHRYQgIv6IqaGG04eTcyVMhoekE0b45QgvBIaOAPSZBl13R6+KYiE7x4XAWIrCP+
c2MZVeXeTgV3Yz+USLg2Y1on+Jh4HxwkPFmZBctyXiUr6DxF8rvoP9W7O27rhRjE
pmqOIfGTWQIDAQABox0wGzAMBgNVHRMBAf8EAjAAMAsGA1UdDwQEAwIFoDANBgkq
hkiG9w0BAQQFAAOBgQBfDqEW+OI3jqBQHIBzhujN/PizdN7s/z4D5d3pptWDJf2n
qgi7lFV6MDkhmTvTqBtjmNk3No7v/dnP6Hr7wHxvCCRwubnmIfZ6QZAv2FU78pLX
8I3bsbmRAUg4UP9hH6ABVq4KQKMknxu1xQxLhpR1ylGPdiowMNTrEG8cCx3w/w==</ds:X509Certificate>
              </ds:X509Data>
                </ds:KeyInfo>
          </ds:Signature>
          <saml:Subject>
            <saml:EncryptedID xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
              <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
                                  Type="http://www.w3.org/2001/04/xmlenc#Element">
```

```
                <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes256-
cbc"/>
                <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
                  <xenc:EncryptedKey Recipient="NL00ZZZ12345678">
                    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-
oaep-mgf1p">
                      <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    </xenc:EncryptionMethod>
                    <xenc:CipherData>
<xenc:CipherValue>jenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBsVW+V=</xenc:CipherValue>
                    </xenc:CipherData>
                  </xenc:EncryptedKey>
                </ds:KeyInfo>
                <xenc:CipherData>
<xenc:CipherValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</xenc:CipherValue>
                </xenc:CipherData>
              </xenc:EncryptedData>
            </saml:EncryptedID>
          </saml:Subject>
          <saml:Conditions NotBefore="2015-01-01T09:30:47Z" NotOnOrAfter="2015-01-
01T09:31:17.123Z”>
            <saml:AudienceRestriction>
              <saml:Audience>NL00ZZZ12345678</saml:Audience>
            </saml:AudienceRestriction>
            <saml:OneTimeUse></saml:OneTimeUse>
          </saml:Conditions>
          <saml:AuthnStatement AuthnInstant="2015-01-01T09:30:47.123Z”>
            <saml:AuthnContext>
              <saml:AuthnContextClassRef>nl:bvn:bankid:1.0:loa2</saml:AuthnContextClassRef>
              <saml:AuthenticatingAuthority>BANKNL2U</saml:AuthenticatingAuthority>
            </saml:AuthnContext>
          </saml:AuthnStatement>
          <saml:AttributeStatement>
           <saml:Attribute Name=”urn:nl:bvn:bankid:1.0:bankid.deliveredserviceid">
              <saml:AttributeValue>21952</saml:AttributeValue>
           </saml:Attribute>
            <saml:EncryptedAttribute xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
              <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
                                  Type="http://www.w3.org/2001/04/xmlenc#Element">
                <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes256-
cbc"/>
                <ds:KeyInfo >
                  <xenc:EncryptedKey Recipient="NL00ZZZ12345678">
                    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-
oaep-mgf1p">
                      <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    </xenc:EncryptionMethod>
                    <xenc:CipherData>
<xenc:CipherValue>jenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBsVW+V=</xenc:CipherValue>
```

```
                    </xenc:CipherData>
                  </xenc:EncryptedKey>
                </ds:KeyInfo>
                <xenc:CipherData>
<xenc:CipherValue>BTeoxDflQ4yfR8KYFvwPVinVPqBsVW+VjenRyZVFCNf=</xenc:CipherValue>
                </xenc:CipherData>
              </xenc:EncryptedData>
            </saml:EncryptedAttribute>
            <saml:EncryptedAttribute xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
              <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
                                  Type="http://www.w3.org/2001/04/xmlenc#Element">
                <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes256-
cbc"/>
                <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
                  <xenc:EncryptedKey Recipient="NL00ZZZ12345678">
                    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-
oaep-mgf1p">
                      <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    </xenc:EncryptionMethod>
                    <xenc:CipherData>
<xenc:CipherValue>jenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBsVW+V=</xenc:CipherValue>
                    </xenc:CipherData>
                  </xenc:EncryptedKey>
                </ds:KeyInfo>
                <xenc:CipherData>
<xenc:CipherValue>nVPqBsVW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVi=</xenc:CipherValue>
                </xenc:CipherData>
              </xenc:EncryptedData>
            </saml:EncryptedAttribute>
          </saml:AttributeStatement>
        </saml:Assertion>
      </samlp:Response>
    </container>
  </Transaction>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
      <Reference URI="">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
          <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
        <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
      </Reference>
    </SignedInfo>
<SignatureValue>IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE
/GPHZShuMw+8WHq4fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
```

```
</SignatureValue>
    <KeyInfo>
       <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
    </KeyInfo>
  </Signature>
</AcquirerStatusRes>
```

14.8 AcquirerErrorRes (B'(X))

```
<?xml version="1.0" encoding="UTF-8"?>
<AcquirerErrorRes version="1.0.0" productID="NL:BVN:BankID:1.0"
xmlns="http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
 <createDateTimestamp>2015-01-01T09:30:47.123.123Z</createDateTimestamp>
 <Error>
    <errorCode>AP3000</errorCode>
    <errorMessage>Product specific error</errorMessage>
    <container>
       <samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol" ID="RES-
1234123456789012" InResponseTo="REF1234567890" Version="2.0" IssueInstant="2015-01-
01T09:30:47.123Z”>
          <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">1234</saml:Issuer>
          <samlp:Status>
             <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Requester">
                <samlp:StatusCode
Value=”urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported”/>
              </samlp:StatusCode>
               <samlp:StatusMessage>Requested BankID.ServiceID not
supported</samlp:StatusMessage>
          </samlp:Status>
       </samlp:Response>
    </container>
 </Error>
 <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
       <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
       <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
       <Reference URI="">
          <Transforms>
             <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
             <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <DigestValue>VW+VjenRyZVFCNfBTeoxDflQ4yfR8KYFvwPVinVPqBs=</DigestValue>
       </Reference>
    </SignedInfo>
    <SignatureValue>
IELLwKSGFMk64US23YrpZ8//hJ8DeJEtYht5knlxJvBOr8dcI+aJTBq+YtyzP9ClcK62Obs5aynHBE/GPHZShuMw+8WHq4
fCMInOwKURgwjDOz8UYaIMqG0Ojiz8dFYGn+dH2lL0QVss4jmIIAD8MCijb27oqij6PclXw9Y9veI=
```

```
    </SignatureValue>
    <KeyInfo>
       <KeyName>7D665C81ABBE1A7D0E525BFC171F04D276F07BF2</KeyName>
    </KeyInfo>
  </Signature>
</AcquirerErrorRes>
```

15 Appendix D: iDx Merchant-Acquirer Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- iDx Messages version 1.0.0: interface Merchant/Acquirer -->
<!-- Copyright © Betaalvereniging -->
<xs:schema xmlns="http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
targetNamespace="http://www.betaalvereniging.nl/iDx/messages/Merchant-Acquirer/1.0.0"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0.0">
   <xs:import namespace="http://www.w3.org/2000/09/xmldsig#" schemaLocation="xmldsig-core-
schema.xsd"/>
   <xs:annotation>
      <xs:documentation>elements defined</xs:documentation>
   </xs:annotation>
   <xs:element name="DirectoryReq">
      <xs:annotation>
         <xs:documentation>Directory Request (A)</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="createDateTimestamp" type="dateTime"/>
            <xs:element name="Merchant">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="merchantID" type="Merchant.merchantID"/>
                     <xs:element name="subID" type="Merchant.subID"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element ref="ds:Signature"/>
         </xs:sequence>
         <xs:attributeGroup ref="MessageAttributes"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="DirectoryRes">
      <xs:annotation>
         <xs:documentation>Directory Response (A')</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="createDateTimestamp" type="dateTime"/>
            <xs:element name="Acquirer">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="acquirerID" type="Acquirer.acquirerID"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element name="Directory">
```

```
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="directoryDateTimestamp" type="dateTime"/>
                  <xs:element name="Country" maxOccurs="unbounded">
                     <xs:complexType>
                        <xs:sequence>
                           <xs:element name="countryNames" type="Country.countryNames"/>
                           <xs:element name="Issuer" maxOccurs="unbounded">
                              <xs:complexType>
                                 <xs:sequence>
                                    <xs:element name="issuerID" type="Issuer.issuerID"/>
                                    <xs:element name="issuerName" type="Issuer.issuerName"/>
                                 </xs:sequence>
                              </xs:complexType>
                           </xs:element>
                        </xs:sequence>
                     </xs:complexType>
                  </xs:element>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
         <xs:element ref="ds:Signature"/>
      </xs:sequence>
      <xs:attributeGroup ref="MessageAttributes"/>
   </xs:complexType>
</xs:element>
<xs:element name="AcquirerTrxReq">
   <xs:annotation>
      <xs:documentation>Acquirer Transaction Request (B)</xs:documentation>
   </xs:annotation>
   <xs:complexType>
      <xs:sequence>
         <xs:element name="createDateTimestamp" type="dateTime"/>
         <xs:element name="Issuer">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="issuerID" type="Issuer.issuerID"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
         <xs:element name="Merchant">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="merchantID" type="Merchant.merchantID"/>
                  <xs:element name="subID" type="Merchant.subID"/>
                  <xs:element name="merchantReturnURL" type="url"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
```

```
            <xs:element name="Transaction">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="expirationPeriod" type="Transaction.expirationPeriod"
minOccurs="0"/>
                     <xs:element name="language" type="Transaction.language"/>
                     <xs:element name="entranceCode" type="Transaction.entranceCode"/>
                     <xs:element name="container" type="Transaction.container"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element ref="ds:Signature"/>
         </xs:sequence>
         <xs:attributeGroup ref="MessageAttributes"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="AcquirerTrxRes">
      <xs:annotation>
         <xs:documentation>Acquirer Transaction Response (B')</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="createDateTimestamp" type="dateTime"/>
            <xs:element name="Acquirer">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="acquirerID" type="Acquirer.acquirerID"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element name="Issuer">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="issuerAuthenticationURL"
type="Issuer.issuerAuthenticationURL"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element name="Transaction">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="transactionID" type="Transaction.transactionID"/>
                     <xs:element name="transactionCreateDateTimestamp" type="dateTime"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element ref="ds:Signature"/>
         </xs:sequence>
         <xs:attributeGroup ref="MessageAttributes"/>
```

```
      </xs:complexType>
   </xs:element>
   <xs:element name="AcquirerStatusReq">
      <xs:annotation>
         <xs:documentation>Acquirer Status Request (F)</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="createDateTimestamp" type="dateTime"/>
            <xs:element name="Merchant">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="merchantID" type="Merchant.merchantID"/>
                     <xs:element name="subID" type="Merchant.subID"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element name="Transaction">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="transactionID" type="Transaction.transactionID"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element ref="ds:Signature"/>
         </xs:sequence>
         <xs:attributeGroup ref="MessageAttributes"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="AcquirerStatusRes">
      <xs:annotation>
         <xs:documentation>Acquirer Status Response (F')</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="createDateTimestamp" type="dateTime"/>
            <xs:element name="Acquirer">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="acquirerID" type="Acquirer.acquirerID"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element name="Transaction">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="transactionID" type="Transaction.transactionID"/>
                     <xs:element name="status" type="Transaction.status"/>
                     <xs:element name="statusDateTimestamp" type="dateTime" minOccurs="0"/>
```

```
                     <xs:element name="container" type="Transaction.container" minOccurs="0"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element ref="ds:Signature"/>
         </xs:sequence>
         <xs:attributeGroup ref="MessageAttributes"/>
      </xs:complexType>
   </xs:element>
   <xs:element name="AcquirerErrorRes">
      <xs:annotation>
         <xs:documentation>Acquirer Error Response (X')</xs:documentation>
      </xs:annotation>
      <xs:complexType>
         <xs:sequence>
            <xs:element name="createDateTimestamp" type="dateTime"/>
            <xs:element name="Error">
               <xs:complexType>
                  <xs:sequence>
                     <xs:element name="errorCode" type="Error.errorCode"/>
                     <xs:element name="errorMessage" type="Error.errorMessage"/>
                     <xs:element name="errorDetail" type="Error.errorDetail" minOccurs="0"/>
                     <xs:element name="suggestedAction" type="Error.suggestedAction"
minOccurs="0"/>
                     <xs:element name="consumerMessage" type="Error.consumerMessage"
minOccurs="0"/>
                     <xs:element name="container" type="Transaction.container" minOccurs="0"/>
                  </xs:sequence>
               </xs:complexType>
            </xs:element>
            <xs:element ref="ds:Signature"/>
         </xs:sequence>
         <xs:attributeGroup ref="MessageAttributes"/>
      </xs:complexType>
   </xs:element>
   <xs:annotation>
      <xs:documentation>simpleTypes defined</xs:documentation>
   </xs:annotation>
   <xs:simpleType name="Acquirer.acquirerID">
      <xs:restriction base="xs:token">
         <xs:length value="4" fixed="true"/>
         <xs:pattern value="[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="Country.countryNames">
      <xs:restriction base="xs:token">
         <xs:minLength value="1"/>
         <xs:maxLength value="128"/>
      </xs:restriction>
```

```
    </xs:simpleType>
    <xs:simpleType name="Error.consumerMessage">
       <xs:restriction base="xs:string">
          <xs:maxLength value="512" fixed="true"/>
          <xs:minLength value="1" fixed="true"/>
       </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Error.errorCode">
       <xs:restriction base="xs:token">
          <xs:length value="6" fixed="true"/>
          <xs:pattern value="[A-Z]{2}[0-9]{4}"/>
       </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Error.errorDetail">
       <xs:restriction base="xs:string">
          <xs:maxLength value="256" fixed="true"/>
          <xs:minLength value="1" fixed="true"/>
       </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Error.errorMessage">
       <xs:restriction base="xs:string">
          <xs:minLength value="1"/>
          <xs:maxLength value="128"/>
       </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Error.suggestedAction">
       <xs:restriction base="xs:string">
          <xs:maxLength value="512" fixed="true"/>
          <xs:minLength value="1" fixed="true"/>
       </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Issuer.issuerAuthenticationURL">
       <xs:restriction base="url"/>
    </xs:simpleType>
    <xs:simpleType name="Issuer.issuerID">
       <xs:restriction base="BIC"/>
    </xs:simpleType>
    <xs:simpleType name="Issuer.issuerName">
       <xs:restriction base="xs:token">
          <xs:maxLength value="35" fixed="true"/>
          <xs:minLength value="1" fixed="true"/>
       </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Merchant.merchantID">
       <xs:restriction base="xs:token">
          <xs:length value="10" fixed="true"/>
          <xs:pattern value="[0-9]+"/>
       </xs:restriction>
    </xs:simpleType>
```

```
<xs:simpleType name="Merchant.merchantReturnURL">
   <xs:restriction base="url"/>
</xs:simpleType>
<xs:simpleType name="Merchant.subID">
   <xs:restriction base="xs:nonNegativeInteger">
      <xs:maxInclusive value="999999" fixed="true"/>
   </xs:restriction>
</xs:simpleType>
<xs:simpleType name="Transaction.entranceCode">
   <xs:restriction base="xs:token">
      <xs:minLength value="1" fixed="true"/>
      <xs:maxLength value="40" fixed="true"/>
      <xs:pattern value="[a-zA-Z0-9]+"/>
   </xs:restriction>
</xs:simpleType>
<xs:simpleType name="Transaction.expirationPeriod">
   <xs:restriction base="xs:duration">
      <xs:minInclusive value="PT1M" fixed="true"/>
   </xs:restriction>
</xs:simpleType>
<xs:simpleType name="Transaction.language">
   <xs:restriction base="language"/>
</xs:simpleType>
<xs:simpleType name="Transaction.status">
   <xs:restriction base="xs:token">
      <xs:pattern value="Open|Success|Failure|Expired|Cancelled|Pending"/>
   </xs:restriction>
</xs:simpleType>
<xs:simpleType name="Transaction.transactionID">
   <xs:restriction base="xs:token">
      <xs:length value="16" fixed="true"/>
      <xs:pattern value="[0-9]+"/>
   </xs:restriction>
</xs:simpleType>
<xs:annotation>
   <xs:documentation>basic simpleTypes defined</xs:documentation>
</xs:annotation>
<xs:simpleType name="BIC">
   <xs:restriction base="xs:token">
      <xs:pattern value="[A-Z]{6,6}[A-Z2-9][A-NP-Z0-9]([A-Z0-9]{3,3}){0,1}"/>
   </xs:restriction>
</xs:simpleType>
<xs:simpleType name="dateTime">
   <xs:restriction base="xs:dateTime">
      <xs:pattern value=".+Z"/>
   </xs:restriction>
</xs:simpleType>
<xs:simpleType name="language">
   <xs:restriction base="xs:token">
```

```
         <xs:length value="2" fixed="true"/>
         <xs:pattern value="[a-z]+"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="productID">
      <xs:restriction base="xs:string"/>
   </xs:simpleType>
   <xs:simpleType name="url">
      <xs:restriction base="xs:anyURI">
         <xs:maxLength value="512"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="version">
      <xs:restriction base="xs:string">
         <xs:pattern value="1\.0\.0"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:annotation>
      <xs:documentation>complexTypes defined</xs:documentation>
   </xs:annotation>
   <xs:complexType name="Transaction.container">
      <xs:sequence>
         <xs:any namespace="##any" maxOccurs="unbounded"/>
      </xs:sequence>
   </xs:complexType>
   <xs:annotation>
      <xs:documentation>attributeGroups defined</xs:documentation>
   </xs:annotation>
   <xs:attributeGroup name="MessageAttributes">
      <xs:annotation>
         <xs:documentation>attributes of each message</xs:documentation>
      </xs:annotation>
      <xs:attribute name="version" type="version" use="required"/>
      <xs:attribute name="productID" type="productID" use="required"/>
   </xs:attributeGroup>
</xs:schema>
```