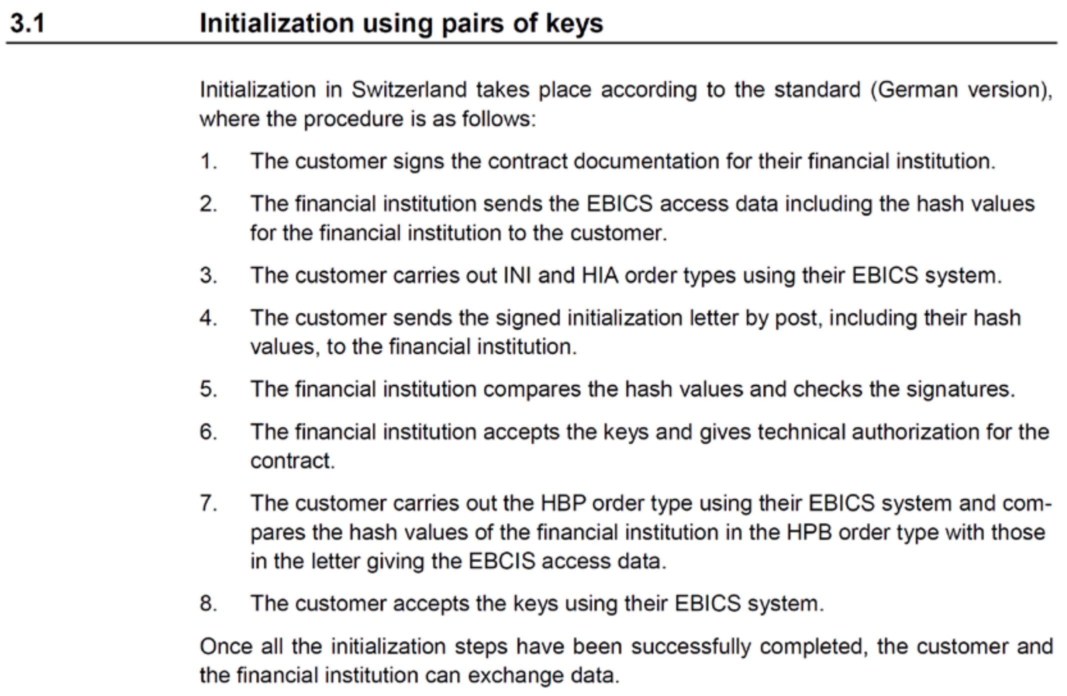
Proposal

A Customer Friendly EBICS Initialization Procedure

The procedure required to initialize an EBICS system is summarized in §3.1 of the « Swiss Market Practice Guidelines EBICS »[1]. It requires the customer and the bank to go through a complex choreography. Having used it in practice with multiple Swiss banks (UBS, Credit Suisse, LUKB, Raiffeisen – June 2017), we have to acknowledge the fact that the process is very technical.



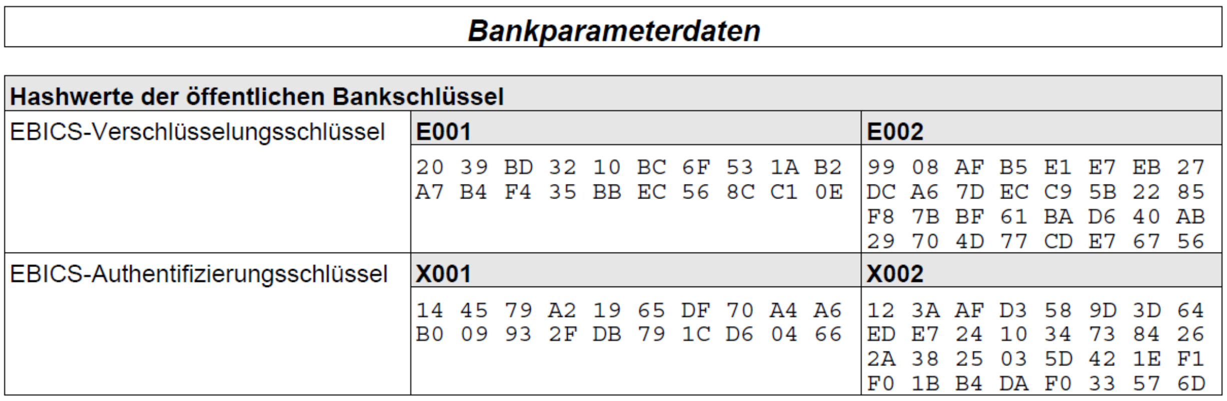
3.1 Initialization using pairs of keys

Initialization in Switzerland takes place according to the standard (German version), where the procedure is as follows:

1. The customer signs the contract documentation for their financial institution.
2. The financial institution sends the EBICS access data including the hash values for the financial institution to the customer.
3. The customer carries out INI and HIA order types using their EBICS system.
4. The customer sends the signed initialization letter by post, including their hash values, to the financial institution.
5. The financial institution compares the hash values and checks the signatures.
6. The financial institution accepts the keys and gives technical authorization for the contract.
7. The customer carries out the HBP order type using their EBICS system and compares the hash values of the financial institution in the HPB order type with those in the letter giving the EBCIS access data.
8. The customer accepts the keys using their EBICS system.

Once all the initialization steps have been successfully completed, the customer and the financial institution can exchange data.

In practice, this means that the customer gets a contract (printed on paper) from the bank, with meaningless gobbledygook, such as:

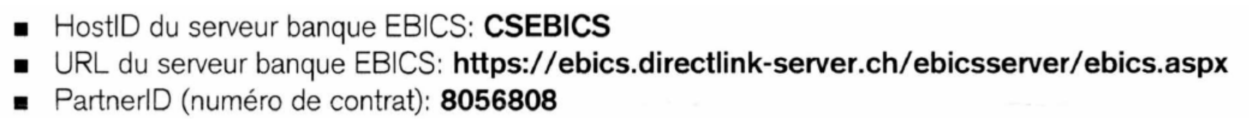


Bankparameterdaten

Hashwerte der öffentlichen Bankschlüssel		
EBICS-Verschlüsselungsschlüssel	**E001**	**E002**
	20 39 BD 32 10 BC 6F 53 1A B2 A7 B4 F4 35 BB EC 56 8C C1 0E	99 08 AF B5 E1 E7 EB 27 DC A6 7D EC C9 5B 22 85 F8 7B BF 61 BA D6 40 AB 29 70 4D 77 CD E7 67 56
EBICS-Authentifizierungsschlüssel	**X001**	**X002**
	14 45 79 A2 19 65 DF 70 A4 A6 B0 09 93 2F DB 79 1C D6 04 66	12 3A AF D3 58 9D 3D 64 ED E7 24 10 34 73 84 26 2A 38 25 03 5D 42 1E F1 F0 1B B4 DA F0 33 57 6D

[1] https://www.six-interbank-clearing.com/dam/downloads/en/standardization/ebics/ebics.pdf

There will also be technical information needed to set up the customer's software:



- HostID du serveur banque EBICS: **CSEBICS**
- URL du serveur banque EBICS: **https://ebics.directlink-server.ch/ebicsserver/ebics.aspx**
- PartnerID (numéro de contrat): **8056808**

The EBICS Host ID, URL, Customer ID[2] and User ID are needed by the customer software to talk to the bank server. They identify the customer, but not in a secure way. This is why EBICS uses actual certificates to secure the whole protocol.

The hexadecimal numbers labeled E001, E002, X001 and X002 in the copy above are *hashes*[3] of the actual cryptographic keys (certificates) used by the EBICS protocol. The keys are used to ensure both authentication (i.e. who is talking with whom) and security (i.e. only the intended audience of a message can decode it).

Setting up trust

The EBICS initialization procedure has been designed so that both the customer and the bank can trust the other party. This trust is ensured by using both a secure electronic communication channel (HTTPS) and plain postal letters (known as the *initialization letters*):

- The electronic channel is used to transmit the heavy payload (i.e. the public keys) from computer to computer, in several steps. INI/HIA send keys from the customer to the bank, HPB retrieves the key from the bank and returns it to the customer.
- The paper trail contains only the hashes. By comparing the printed hashes, an operator can verify that they match those of the keys, which were exchanged electronically.

The contract printed by the bank and sent to the customer contains 4 hashes:

- Encryption: E001 or E002.
- Authentication: X001 or X002.

It is my understanding that only one of the hashes is really needed for each class (either E001 or E002 for encryption and either X001 or X002 for validation), depending on the algorithms being used. It is further my understanding that the banks are using the same hashes for every EBICS contract (see document[4] from the ZKB).

The initialization letters printed by the customer's software contain 3 hashes matching the keys produced during the EBICS initialization (typically A005, E002 and X002, which is the case of our own EBICS client implementation in Crésus Banking).

[2] In the example provided above, the Customer ID is called *Partner ID*, and the User ID is missing altogether. This exemplifies why a common vocabulary should be used in order to reduce the user's confusion.
[3] A hash can be viewed as a thumbprint of a key – any change to the key will produce a distinct hash. If two keys are the same, then their hashes are the same. And the opposite is also true (in theory, identical hashes could be forged, but there are no practical attacks for the algorithms used by EBICS).
[4] https://www.zkb.ch/media/dok/efinance/ebics-verbindungsparameter.pdf

Validating the hashes

Hashes have to be validated both by the customer and by the bank.

Our tests have shown that the bank operators typically only compare the first 8 digits of the three hashes (which in the case of A005, E002 and X002 have 64 digits). This excludes tampering of the keys during transport with a very high probability ($p_{tampering} \approx 0.23 \times 10^{-9}$) while keeping the complexity of the verification within reasonable bounds.

The customer should also validate the keys returned by the bank. This can be done by requiring the user to enter a few digits from the E002 and X002 hashes provided in the printed contract. The software takes on the responsibility of this validation. Technically, it might just skip this step, thus reducing the security of the whole system.

Simplifying the bank operator's job

For every user registered with an EBICS contract, the bank typically gets back three sheets of paper, printed by the customer's software. Each of the letters is signed (by hand) by the customer. The signature needs to be verified against the customer's signature (on record at the bank).

When the documents are considered to be trustworthy, the bank operator may validate the printed hashes and authorize the specific user's contract based on the customer ID and the user ID. Basically, the operator needs to enter at least:

- The customer ID.
- The user ID.
- For the three hashes:
    - The key type (A005, E002 or X002).
    - The first digits of the hash.

To our knowledge, this procedure has not been automated by the banks. We suppose that they do not expect high volumes of EBICS activations; therefore, automating does not make sense for them.

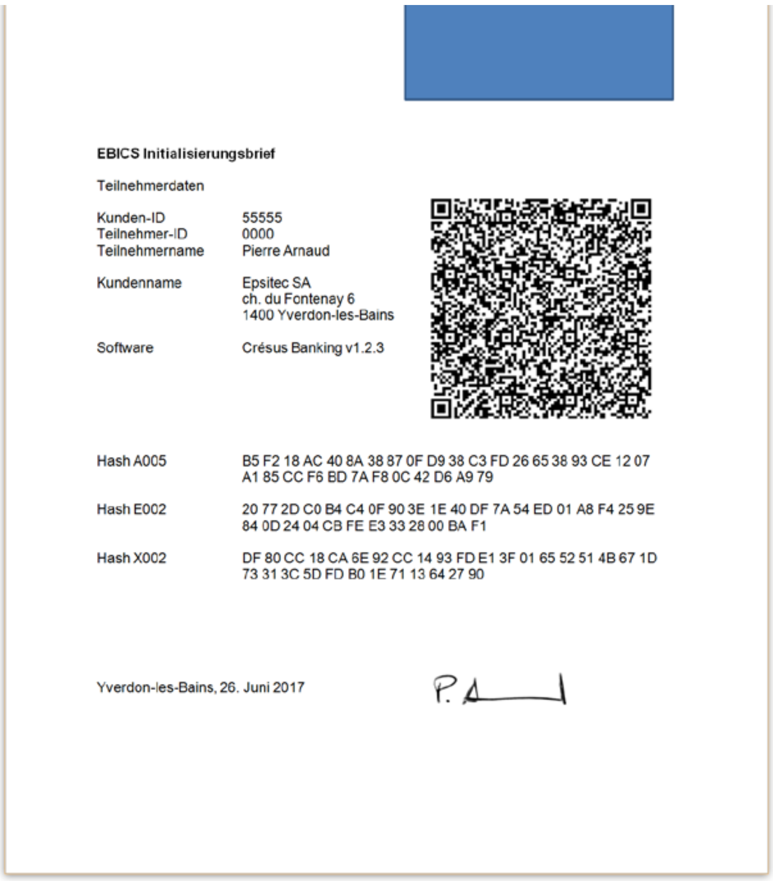
Proposal

In order to simplify the bank operator's job, the software should print one document for every user, which would summarize all the information as both human readable text and machine readable data. The machine readable data could be represented as a QR-code containing the values in a structured format (e.g. XML).

This would allow an automatic processing by the bank operator.

Example

The following example shows a hypothetical standardized Swiss EBICS initialization letter. It contains all the data required to validate a customer's contract:



EBICS Initialisierungsbrief

Teilnehmerdaten

Kunden-ID	55555
Teilnehmer-ID	0000
Teilnehmername	Pierre Arnaud
Kundenname	Epsitec SA ch. du Fontenay 6 1400 Yverdon-les-Bains
Software	Crésus Banking v1.2.3
Hash A005	B5 F2 18 AC 40 8A 38 87 0F D9 38 C3 FD 26 65 38 93 CE 12 07 A1 85 CC F6 BD 7A F8 0C 42 D6 A9 79
Hash E002	20 77 2D C0 B4 C4 0F 90 3E 1E 40 DF 7A 54 ED 01 A8 F4 25 9E 84 0D 24 04 CB FE E3 33 28 00 BA F1
Hash X002	DF 80 CC 18 CA 6E 92 CC 14 93 FD E1 3F 01 65 52 51 4B 67 1D 73 31 3C 5D FD B0 1E 71 13 64 27 90

Yverdon-les-Bains, 26. Juni 2017 [illegible]

The data stored in the QR-code of this example maps to this piece of XML[5]:

```
<ebics>
<date>2017-06-29T06:05:01</date>
<name>Epsitec SA</name>
<street>ch. du Fontenay</street>
<house>6</house>
<zip>1400</zip>
<town>Yverdon-les-Bains</town>
<user>Pierre Arnaud</user>
<cid>55555</cid>
<uid>0000</uid>
<soft>Crésus Banking v1.2.3</soft>
<h t="a005">B5F218AC408A38870FD938C3FD26653893CE1207A185CCF6BD7AF80C42D6A979</h>
<h t="e002">20772DC0B4C40F903E1E40DF7A54ED01A8F4259E840D2404CBFEE3332800BAF1</h>
<h t="x002">DF80CC18CA6E92CC1493FDE13F016552514B671D73313C5DFDB01E7113642790</h>
</ebics>
```

Validating the customer's contract now becomes as easy as *scan the document*. The bank's software can automatically validate the contract based on the structured data contained in the QR-code.

[5] This is a fake example, there is currently no standardized way to convey the information.

Simplifying the customer's job

EBICS requires that the user configures her software with the parameters found in the contract received from the bank, typically:

- EBICS-Host ID.
- EBICS-URL.
- EBICS-Customer ID.
- EBICS-User ID.
- EBICS-Signature Type associated with the user (A/B/E).
- Optionally a list of supported EBICS order types (e.g. XE2, XE3, C52, C53, C54, etc.).

Ideally, the software should also ask the user to enter two of the four hashes (E001/E002 and X001/X002) in order to be able to validate the keys received from the bank through its final HPB order.

Note: we suggest that only two hashes should be provided (E002 and X002), since E001 and X001 are no longer supported by the latest EBICS standard (from 2.4 on) – this helps reducing the confusion.

Simplifying the customer's job may not be the bank's priority, as the work needs to be done only once and the burden is offloaded onto the customer. However, making this process easy for the customer will both improve the user's perception of the bank, and help decrease support calls.

Proposal

If the information provided by the bank is *stable*, the software could be seeded with the bank's parameters (the information is hardcoded in the software), specifically:

- EBICS-Host ID.
- EBICS-URL.
- E002 hash.
- X002 hash.

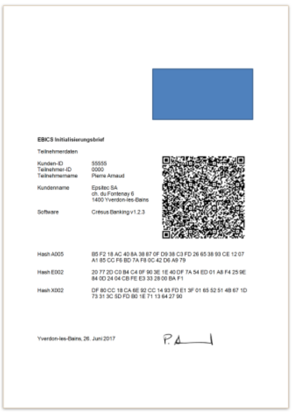
Depending on how the software implements the EBICS connection, it already knows which IBAN will be used with the communication channel. Therefore, it can derive the bank from the IID found in the IBAN, and from the bank, the bank's parameters.

The user who uses smart software only needs to enter what is really meaningful to her:

- EBICS-Customer ID.
- EBICS-User ID.
- EBICS-Signature Type.

The software provides the rest.

In order to further simplify the handling of the initialization letters, the software should print the bank's postal back-office address directly on the letter (represented by a blue box on the illustration).



EBICS Initialisierungsbrief

Teilnehmerdaten

Kunden-ID	55555
Teilnehmer-ID	0000
Teilnehmername	Pierre Arnaud
Kundenname	Epsitec SA ch. du Fontenay 6 1400 Yverdon-les-Bains
Software	Crésus Banking v1.2.3
Hash A005	B5 F2 18 AC 40 8A 38 87 0F D9 38 C3 FD 26 65 38 93 CE 12 07 A1 85 CC F6 BD 7A F8 0C 42 D6 A9 79
Hash E002	20 77 2D C0 B4 C4 0F 90 3E 1E 40 DF 7A 54 ED 01 A8 F4 25 9E 84 0D 24 04 CB FE E3 33 28 00 BA F1
Hash X002	DF 80 CC 18 CA 6E 92 CC 14 93 FD E1 3F 01 65 52 51 4B 67 1D 73 31 3C 5D FD B0 1E 71 13 64 27 90

Yverdon-les-Bains, 26. Juni 2017 [illegible]

Open questions

Following questions should be considered:

- What happens when banks need to change their keys and provide new hashes?
- What happens if the EBICS-URL needs to be updated?

Hardcoding the bank parameters into the software might be far from ideal, so other approaches should be considered:

- The bank provides the initialization parameters as a file, over a trustworthy channel.
- The bank provides the initialization parameters on paper, both human readable and as a QR-code, which can be scanned by the user's software to read the parameters.
- The bank provides the initialization parameters over HTTPS, in a machine readable format.
- A central repository provides the initialization parameters over HTTPS, in a machine readable format.

For software partners, a central repository would be ideal, as it would simplify the integration of new banks. Moreover, it could also provide other technical details, such as a list of orders supported by a given bank, etc.

The future of EBICS in Switzerland

The introduction of EBICS as a secure channel between the customer and the bank helps automating the business processes. We have shown that the EBICS initialization is very technical. It was certainly not designed to primarily be user-friendly. However, when switching to EBICS, a bank should consider carefully how it integrates its customers. It should strive to reduce the administrative load bound with EBICS initialization.

As long as EBICS is limited to few large corporations, these aspects are secondary. However, democratizing EBICS on the Swiss marketplace – as could be expected from market leaders like PostFinance – means that potentially over 500 000 enterprises[6] might use EBICS. This requires a careful rethinking of the whole onboarding process to make it as simple and as user friendly as possible.

For EBICS to become a major player in the Swiss payment system, improvements have to be made to the initialization process. **This can only happen if all actors get together and define a common standard**.

Disclaimer

This document was written for PostFinance by Dr. Pierre Arnaud, CEO of Epsitec.

Our knowledge of EBICS is by no means complete and Epsitec SA will not be held accountable for providing potentially inaccurate information. The main objective of this document is to start a discussion.

We consider that such a discussion is a prerequisite to the success of EBICS as the future common Swiss standard for customer/bank communication for ISO 20022 messages.

Yverdon-les-Bains, 30 June 2017, ed. 3

[6] In 2014, 518 795 micro-enterprises (1-9), 48 858 small enterprises (10-49) for a total of 576 559 SMEs.Information published by the Swiss Federal Administration, available online at www.kmu.admin.ch/kmu/de/home/kmu-politik/kmu-politik-zahlen-und-fakten/kmu-in-zahlen/firmen-und-beschaeftigte.html .