**Connectivity Guide**

Banks offer a range of digital connectivity options: EBICS, FileAct, API, and SFTP. These methods are distinct and serve different purposes for transferring data between banks and corporate clients. Each of these connectivity methods requires authorization by an appointed Authorized Person within the corporate entity. Additionally, for EBICS, FileAct, API, and SFTP, a corporate seal can be utilized as part of the authorization process. The application of a corporate seal and its acceptance may vary depending on legal considerations in different jurisdictions, with some imposing limitations on its use in electronic transactions.

**Differences Between the Connection Methods:**

**EBICS:** A European protocol supporting ISO 20022 and other formats, offering end-to-end encryption and digital signatures for secure and standardized banking communication within Europe.

**FileAct:** A SWIFT service for exchanging large files in formats like ISO 20022 or EDIFACT. It requires SWIFT membership and may involve charges per file or transaction. Known for its secure and reliable data transfer within the SWIFT network.

**SFTP:** A protocol enabling secure file transfer over SSH. It offers flexibility in file formats but does not impose specific standards. Additional security measures like VPN or PGP encryption may be necessary for enhanced security.

**APIs:** Open APIs facilitate real-time data access. Secure API connectivity between corporates and banks involves the exchange of digital certificates and encryption, typically using TLS.

**Additional Tools in Digital Connectivity: Corporate Seal**

The Corporate Seal is an electronic tool employed by companies, to authenticate their transactions with banks. It facilitates the automation of the signature process, reducing the need for manual intervention. This tool can be used not only in transaction authentication but also in retrieving statements and accessing other official documents.

**File Exchange Explained (XML file example):**

The file exchange process between a corporate client and a bank typically involves the following steps:

1. File Generation by the Client:

The corporate client generates an XML file using their own software. This file is prepared according to a previously agreed-upon format (file specification shared during onboarding), adhering to a standardized schema recognized and accepted by both the client and the bank.

2. Transmission of the XML File:

The client then securely transfers the XML file to the bank using an agreed-upon connection channel, such as EBICS, SFTP, API or FileAct.

3. Authorization and Encryption:

Before transmission, the XML file is authorized and encrypted. Authorization is typically executed by a designated user (Authorized Person), optionally using a Corporate Seal, depending on the client's internal controls and agreement with the bank. The encryption is executed using a method that aligns with the chosen connection channel, ensuring data security and confidentiality during transmission.

4. Bank's Receipt and Acknowledgement:

The bank processes the received XML file and sends back an acknowledgment file (such as pain.002 for positive or negative acknowledgement) or a response file (like an account statement or eBAM message: Account Report). This response, also in XML format, is transmitted over the same channel as the original file and is encrypted to maintain secure communication.

5. Receipt of Acknowledgement or Response by the Client:

The client's software receives the acknowledgment or response file from the bank. The client applies appropriate security measures, such as authorized user validation, with or without the use of the corporate seal, to secure the received file.

Connectivities

1. **EBICS (Electronic Banking Internet Communication Standard):**

Software Requirement: Yes.

EBICS requires specific software to facilitate the communication protocol. BANK does not provide this software, and clients are responsible for its implementation and maintenance. If BANK requires an update to the EBICS version, clients are obligated to update their software to ensure compatibility with the latest protocol standards.

*Note: To process payment or eBAM files effectively, a specific standardized order type, as specified by EBICS, is required to initiate, and retrieve payment or eBAM messages through EBICS.*

Key Exchange:

Involves the exchange of X.509 PKI certificates for authentication and encryption. BANK sends an INI letter with the bank's public key and an HIA letter with the activation code for each user. Clients must verify the INI letter and enter the activation code in their EBICS software to complete the setup.

Corporate Seal Usage:

Compatible with Corporate Seal. Clients are responsible for ensuring that requests sent to the bank are authorized and adhere to the correct security measures.

Key Features:

Supports ISO formats, end-to-end encryption, and digital signatures.

**EBICS Solution Description:**

EBICS is a European interbank protocol designed for secure and standardized data exchange. Developed by the German Central Credit Committee (ZKA) around 2006, it was initially a German standard and later adopted by France and Switzerland, each with their own versions.

The protocol is adaptable, with features and versions tailored to the banking practices of each participating country. Users should verify the current version and any updates, as these can change.

A key strength of EBICS is its flexibility in supporting various currencies, allowing transactions in any currency, provided there is an agreement on the format and content of the payment files.

EBICS differs from FileAct in its use of end-to-end encryption and digital signatures. While EBICS is potentially more cost-effective and simpler to implement in its core markets, FileAct, associated with SWIFT, may incur additional costs due to transaction-based charges.

**To Facilitate Payments Through EBICS:**

· EBICS Contract: A formal agreement outlining the use of the EBICS protocol, including protocol version, message types, security measures, and roles of Authorized Persons.

· EBICS Certificate: A digital document containing the public key for encryption and digital signing, and identifiable information like name, bank, and EBICS ID.

· INI and HIA Letters: Crucial for establishing a secure communication channel and activating the EBICS account.

· Verification and Activation Process: Involves verifying the bank's public key and entering the activation code from the HIA letter.

· Specific EBICS Software: Necessary for handling encryption, digital signing, and message transmission.

· EBICS License: A legal agreement for using specific EBICS software or services, detailing terms and conditions.

· Standardized Order Types: Predefined formats for financial messages and transactions, ensuring consistency and interoperability across EBICS-using institutions.

2. **FileAct (SWIFT Service):**

Software Requirement: Yes.

Access to the SWIFT network and SWIFTNet integration are required. BANK is not involved in the agreement between the Client and SWIFT, nor does it act as an intermediary in SWIFTNet integration.

Key Exchange:

Utilizes SWIFTNet Public Key Infrastructure (PKI) for authentication. Customers must have a SWIFT-registered BIC. There is no direct cryptographic key exchange between the Client and the bank. Instead, FileAct parameters are exchanged between the Client and the bank.

Corporate Seal Usage: Compatible with Corporate Seal.

Clients are responsible for ensuring requests sent to the bank are authorized and follow correct security measures.

Key Features:

Supports both store-and-forward and real-time delivery modes:

· Store-and-Forward Mode: Transmitted files are temporarily stored at an intermediate station before being forwarded to the final recipient.

· Real-Time Mode: Immediate transmission of files without delay or intermediate storage.

**FileAct Solution Description:**

FileAct is a messaging service within the SWIFT network designed for the secure and reliable exchange of large files and associated information. Suitable for transferring structured messages in bulk, such as bulk payments or securities data.

Supports two primary modes of file transfer:

- Store-and-Forward: Allows deferred file transfer until the recipient is ready.
- Real-Time: Ensures immediate file transmission, crucial for time-sensitive operations.

FileAct usage requires SWIFT membership and a SWIFTNet Link connection, differing from SFTP, which only needs an SFTP client and server. FileAct provides end-to-end security, reliability, and compliance with SWIFT standards.

SCORE and MA-CUG Models:

FileAct operates under two models in the SWIFT network – SCORE and MA-CUG:

- SCORE: Designed for corporate entities to securely communicate with banking partners over SWIFT, offering standardized communication channels.
- MA-CUG: Used by financial institutions, where a member bank administers its closed user group, controlling access and communication within the group.

3. **SFTP (Secure File Transfer Protocol):**

Software Requirement: Yes.

Requires SFTP client and server software.

Key Exchange:

Involves the use of SSH (Secure Shell) public keys to secure data during transit. The renewal of these SSH keys is subject to the security policies of the involved bank, which may define specific periods for key renewal. Additionally, PGP (Pretty Good Privacy) can be utilized as an additional encryption method for securing the data at rest

Corporate Seal Usage: Compatible with Corporate Seal.

Clients are responsible for ensuring that requests sent to the bank are authorized and use the correct security measures.

Key Features: A generic protocol for secure file transfer over SSH, with encryption for both data and commands.

**SFTP Solution description:**

SFTP is a protocol used for securely transferring files over a network. It utilizes encryption and authentication, leveraging SSH (Secure Shell) to create a secure connection between two computers. SFTP encrypts both the data and commands, ensuring their confidentiality and integrity.

While SFTP is cost-effective, it may require additional software installations, depending on the user's setup. The underlying SSH protocol employs encryption algorithms such as AES, Blowfish, and 3DES. While PGP encryption is not a direct feature of SFTP, it is often used in conjunction with SFTP for enhanced security.

To use SFTP, SSH access to the remote server and a compatible SFTP client program are necessary. SFTP supports password authentication and SSH key-based authentication. In the latter, a pair of public and private keys is generated, and the public key is copied to the remote server.

To configure SFTP connectivity with a bank, the following general steps are recommended:

1. Contact your bank to request an SFTP account, including login credentials, server address, or to exchange SSH keys.
2. Select an SFTP client software compatible with the bank's server and supporting necessary protocols.
3. Configure the SFTP client with the bank's credentials and perform connection tests.

4. Establish a schedule or trigger mechanism for uploading and downloading files to and from the bank’s server.

4. **API (Application Programming Interface):**

Software Requirement: Clients have the option to use libraries provided by the bank to initiate API calls and implement solutions. It is the client's responsibility to embed these API libraries on their side, ensuring proper integration with their systems.

Responsibilities of the Client:

· Ensure system compatibility with the bank's API infrastructure, verifying that their technology stack can integrate with the API.

· Regularly update their systems to conform with the latest API versions and adhere to the security standards set forth by the bank. This includes updating the API libraries and any relevant software components to align with the bank’s updates.

· Procure and manage certificates as required by the bank's regulations, with attention to the specific renewal periods as defined by the bank's security policies.

API encryption methods:

The primary method for data encryption is TLS (Transport Layer Security), providing a secure communication channel over the internet.

Additionally, API security encompasses advanced methodologies such as OAuth (Open Authorization) for secure delegated access, and JWTs (JSON Web Tokens) for securely transmitting information between parties as a JSON object.

For enhanced security, JSON Object Signing and Encryption (JOSE) can also be employed, offering both encryption and digital signing of data.

Corporate Seal Usage: Compatible with Corporate Seal.

Clients are responsible for ensuring that requests sent to the bank are authorized and use the correct security measures.

Key Features:

The API provides a versatile method for data exchange, allowing for the integration of banking services into the client's operational workflows.

The client is responsible for using the most current version of the API service based on the bank's provided documentation.

Clients must ensure compliance with relevant regulatory and security standards, regularly monitoring and updating their API integration to align with changes in the bank's API offerings.

**Push and Pull Communication Overview**

Payments are always created in the customer's ERP or treasury system and then either sent directly (pushed) to the bank or made available for the bank to retrieve (pulled). Similarly, account statements are always generated by the bank and then either sent directly (pushed) to the customer or made available for the customer to retrieve (pulled). Depending on the chosen connectivity method, both payments and account statements can use push or pull communication.

**Push Communication**

**Payments:**

FileAct Real-Time: Payments are immediately transmitted to the recipient without delay or intermediate storage.

SFTP: The bank actively sends (pushes) payment files to the client's SFTP server.

API: Payments can be pushed to the client using webhooks or push notifications, enabling real-time payment updates.

**Account Statements:**

FileAct Real-Time: The bank pushes account statements directly to the client without delay.

SFTP: The bank actively sends (pushes) account statements to the client's SFTP server.

API: The bank pushes real-time account statements to clients using webhooks or similar mechanisms.

**Pull Communication**

**Payments:**

EBICS: Clients initiate the retrieval of payment information from the bank.

SFTP: Clients log in to the bank's SFTP server to download (pull) payment files.

API: Clients make API calls to retrieve payment information from the bank.

**Account Statements:**

FileAct Store-and-Forward: Account statements are stored at an intermediate station until the client is ready to retrieve them.

SFTP: Clients log in to the bank's SFTP server to download (pull) account statements.

EBICS: Clients request account statements from the bank using the EBICS protocol.

API: Clients make API calls to request account statements from the bank.

**PGP Encryption for banking connections:**

Basic Overview:

*Inbound Payments to Bank:*

When a client sends payments to the Bank, they first encrypt the file using bank's public key. This ensures that only Bank, with its corresponding private key, can decrypt the file. Additionally, the client signs the file using their own private key, which corresponds to the public key they have provided to Bank. This signature allows Bank to verify the file's authenticity and integrity using the client's public key.

*Outbound Statements & pain.002 from Bank:*

For outbound files like statements and pain.002 messages, Bank encrypts the file using the client's public key, ensuring that only the client can decrypt it using their private key. Bank

also signs the file with its private key. The client, upon receiving the file, uses its private key to decrypt it and verifies BANK's signature using the public key provided by BANK.

**PGP Technical Details:**

*Inbound to the Bank:*

1. The client encrypts the file using a randomly generated session key (specific to that transaction and created by PGP).
2. This session key itself is then encrypted using Bank's public key.
3. The client signs the file with its private key, which corresponds to the public key shared with the Bank.
4. Bank decrypts the session key using its private key, and then uses the session key to decrypt the file.
5. Bank verifies the client's signature using the public key provided by the client.

*Outbound from the Bank:*

1. Bank encrypts the file using a randomly generated session key.
2. The session key is encrypted using the client's public key.
3. Bank signs the file with its private key.
4. The client decrypts the session key with its private key and then uses it to decrypt the file.
5. The client verifies Bank's signature using the public key provided by the Bank.

**For payments some banks offer Conversion solution:**

Some banks offer Conversion Solution which allows to accept one payment format XML files for all countries. The customer sets up payments that bank can convert to the specific, local country formats.