InsideBusiness Connect Client - Installation Guide (RPM based Linux distributions)

Table of Contents

Introduction

This document describes the installation of the InsideBusiness Connect Client (IBCC) on an RPM based Linux distribution. It explains the process of installing the service via an RPM installer.

What does it do?

IBCC offers you an easy and safe solution for the automated processing of large transaction files. It gives you direct access to ING's European network for sending payment orders and receiving reports.

What is it?

IBCC is the new ING Wholesale Banking application that allows the customer to easily connect with ING InsideBusiness Connect. This is a Java program that runs as a service and establishes a safe, host-to-host connectivity between your ERP system and ING and ensures safe, fast and automated processing of transaction and report files.

The application provides folders (incoming and outgoing) for the customers to drop their payments files which will automatically be uploaded to ING. Reporting files are generated and are automatically downloaded. They can be picked up for reconciliation purposes by the ERP system of the customer.

Files can be delivered to and collected from IBCC 24 Hours a day, 7 days a week (24x7).

Foundation and Generic modules

- Secure connection via SFTP
- Automated upload every second
- Automated download every 15 minutes
- Application heartbeat monitoring

Version

If a version of IBCC is already installed on the system, it can be checked in a terminal as follows:

```
$ sudo rpm -qi ibcc
```

Installation Requirements

Account Requirements

The user needs to have an FTP account as well as customer certificates that obtain during the ING onboarding process.

Installer Rights

The RPM installer needs to be executed with root privileges, however the code snippets in this document will mostly include the `sudo` instruction in front of every command.

Technical environment

IBCC is installed from an RPM package. After installation, configure it via a standard properties file.

Secure the transfer and configuration directories from unauthorized access before you start using IBCC. Untrusted parties must not be able to *read or write* the data directories or the properties file.

The minimal required space for the installation of the software is 200 MB. File size for the created directories depends on the customer usage.

ING Digital signature

The IBC client software installer (RPM package) is digitally signed. It is important before it is installed in your environment that you verify its signature, in order to make sure that the RPM package hasn't been tampered with.

The process of verifying the signature described below uses the `gpg` tool, available as a standard binary in all RPM based Linux distributions.

In order to verify the signature, the public part of ING's signing key must be imported in your environment. Please contact ING support to receive the public key.

Importing the public key

```
$ sudo rpm --import ING_SIGNING_KEY.pub
```

Note: the import of the public key is only necessary the first time. Upcoming releases of the IBC client software will be RPM signed with the same key, so only the below verification step will be needed.

Verifying the IBCC RPM package

```
$ sudo rpm -Kv InsideBusinessConnectClient_linux_<version>.rpm
```

You should see a success message, containing the following lines:

```
InsideBusinessConnectClient_linux_<version>.rpm:
    Header V4 RSA/SHA1 Signature, key ID 93a24c9f: OK
    Header SHA1 digest: OK (98eef2659cbb7a9926c1ca83123cd1505d5d4f2b)
    V4 RSA/SHA1 Signature, key ID 93a24c9f: OK
    MD5 digest: OK (62f512425f3f0fea7ecdc0e54976894f)
```

Important: for old versions of the `rpm` tool (below 4.8.0) it might not be possible to verify the IBCC RPM package with the above method. The reason for this is that the IBCC RPM package is created with `rpm` version 4.12.0, which uses V4 signing. In this case, please refer to the below instructions to verify the IBCC RPM package using a GPG detached signature file.

Verifying with a GPG detached signature file

The GPG detached signature is provided next to the IBCC RPM package, and is named `InsideBusinessConnectClient_linux_<version>.rpm.sig`. Same as for the ING signing public key, the detached signature has to be provided by ING in a trusted way.

The ING signing public key has to be imported in the GPG store this time:

```
$ sudo gpg --import ING_SIGNING_KEY.pub
```

Then you can set the level of trust for the ING signing public key:

```
$ sudo gpg --edit-key IBCC Support DEVOPS
```

Enter `trust` in the prompt, then `5` if you ultimately trust the key.

As for importing this key with `rpm`, as described above, this step is to be done only once and is not needed for upgrading IBCC.

Finally, the IBCC RPM package can be verified as follows:

`$ sudo gpg --verify InsideBusinessConnectClient_linux_<version>.rpm.sig InsideBusinessConnectClient_linux_<version>.rpm`

You should see a success message, looking like the following:

```
gpg: Signature made Mon 18 Mar 2019 07:28:31 PM CET using RSA key ID 93A24C9F
gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2021-03-17
gpg: Good signature from "IBCC Support DEVOPS <mbxnl15120@internal.ing.com>"
```

This confirms the authenticity and validity of the rpm file, you can then proceed with the installation.

Service User

The IBCC service requires a user *ibccservice* with basic privileges (not root). This user will automatically be created upon installing the RPM, as well as an *ibccservice* group.

This is a standard user, but it has no password. Nobody can log in as this user.

The **transfer** directories (explained just below) must be readable/writable by the *ibccservice* service user.

Installation and transfer directories

The minimal required space for the installation of the software is 200 MB. The following directories are created inside the *Transfer* directory specified during the installation phase:

- **upload**: Payment files dropped here will be encrypted and sent to ING for processing.
- **download**: Decrypted Reporting files are placed here for reconciliation by the customers ERP system
- **archive**: Successful sent Payment files are archived here

The required space on disk depends on the size of the payment files. For a single transfer a minimum amount of twice the size of a payment file is needed.

As transferred payment files are kept in the **archive** directory, this directory must have enough free space to hold the payment files.

For example: For a yearly archive of monthly payments the required disk space is at least 12x the maximum payment file size.

By default, the **archive** directory will never be purged and all succeeded transfer will be kept in the folder. A purge mechanism is in place and can be configured via a property. Please see the **Appendix A: IBCC properties** for more details.

Important: The above directories are created during install into the *ibccservice* users home folder. The IBCC service can be configured to use other directories (see below), in that case the *ibccservice* must have read / write access on all of them.

The default directories have read and write access for all members of the *ibccservice* group. All other users that need to interact with these folders (typically, an ERP user), have to be added to the *ibccservice* group. This can be done as follows:

```
$ sudo usermod -a -G ibccservice <your_erp_user>
```

Upload folders

You will need at least one upload folder, but there may be several, especially if you also use Virtual Ledger Accounts (VLA) or FactorLink (FACTCLT). Configure your upload folders in the properties file, */etc/ibcc/icb.properties*. Look for the field with the key `icb.daemon.upload_dir`.

The field can list any number of upload folders, separated by semicolons: `icb.daemon.upload_dir=/home/ibccservice/transfer/upload/;/home/ibccservice/transfer/upload2/;/home/ibccservice/transfer/upload3/`

Each folder can be listed only once. If you specify the same folder twice, IBCC will quietly ignore one.

If the folder paths themselves contain any semicolons (`;`), question marks (`?`), or pipe symbols (`|`), *escape* them by prefixing them with a question mark. So: * Write `suppliers; consultants` as `suppliers?; consultants`. * Write `buildings|Amsterdam` as `buildings?|Amsterdam`. * Write `legal?` as `legal??`.

By default, IBCC will take your uploads as payment orders. But you can associate an upload folder with a different product type, such as Virtual Ledger Accounts (VLA). If your contract includes VLA, you can earmark one or more upload folders for this product type: `/home/ibccservice/transfer/vla|VLA`.

To send files to FactorLink the upload folder could look like this: `/home/ibccservice/transfer/fl_upload|FACTCLT`.

(If you prefer, you may also specify the product type for your regular upload folders: `/home/ibccservice/transfer/upload|PAYMENT`.)

If you have made any changes to the upload folders, restart the service to apply them:

```
sudo service ibcc restart
```

IBCC works with upload folders you provide; it *does not create the upload folders.* Nor does it remove them from the file system when you deactivate them. It only creates the subfolders.

Atomic writes

When you (or your ERP software) writes a file into an upload folder, this must happen *atomically.* Otherwise, IBCC would have no reliable way to know whether the file is complete. Your ERP package may still be writing to it, or something may have gone wrong.

Therefore, ensure that the file is first written to a temporary location on the same physical drive as the target location, and then once it has been written completely, it should be moved or renamed to its definite location.

The IBCC service will ignore temporary files in the **upload** folders, so one way of doing this would be to start by writing the file under a name which marks it as temporary, and finally, changing its name.

Temporary files are defined as follows:

- They are hidden (starting with **.**)
- They end with the **.tmp** extension

Here is an example, in pseudocode, of how a file should correctly end up in the **upload** folder:

```
copy input_file into upload/input_file.{uuid}.tmp
move (or rename) upload/input_file.{uuid}.tmp to upload/input_file (this operation is ato
```

We recommend including some random characters (e.g. a UUID or GUID) in the temporary file name, to avoid accidental overwrites in the case where you may write multiple files into the upload folder simultaneously.

Java Runtime Environment 8

IBCC requires at least Java 8u311 because it contains a fix for a bug in JRE that affects IBCC. Please make sure you have it installed before going further with the installation of IBCC.

It is recommended to use BellSoft's OpenJDK JRE as it was used to build IBCC. You can install it from the BellSoft repository by following the official instructions.

The following snippet shows how to install Java 8 from an RPM package (64-bit systems).

After downloading the RPM package from BellSoft's official download website, perform the following:

```
$ sudo rpm -ivh bellsoft-jre8u472+11-linux-amd64-full.rpm
$ sudo rm bellsoft-jre8u472+11-linux-amd64-full.rpm
```

An alternative way of installing Java 8 is to use your distribution's package manager. This will most likely provide *OpenJDK* JRE implementations from another vendor. We do not recommend this, since IBCC is developed using the BellSoft's JRE.

For example, for Java 8, 32 bits, on a **RHEL** system:

`$ sudo yum install java-1.8.0-openjdk-headless.i686`

To check that it successfully installed:

`$ java -version`

It should show Java 8.

chkconfig Utility

IBCC requires chkconfig utility. It's available by default in RHEL7 and 8.

On RHEL9, it can be installed by running the following command

`$ sudo dnf install -y chkconfig`

Heartbeat monitoring

The IBCC service is sending heartbeats to the ING server every 1 hour.

Installing

Executing IBCC RPM file

The IBCC RPM installation file is named *InsideBusinessConnectClient_linux_{version}.rpm* where *{version}* is the version of IBCC you are installing.

Please check the signature verification instruction before installing the RPM.

To start the installation, simply execute the RPM, as follows:

```
$ sudo rpm -ivh InsideBusinessConnectClient_linux_<version>.rpm
```

The IBCC files will be installed in the */opt/ibcc/* folder.

In addition, configuration files will be placed in */etc/ibcc/*. They include, among others, the ING known hosts and certificate files. A log directory will be created in */var/log/ibcc/*. From the output you will see that a new user was created (ibccservice), and that the installation was successful.

A symlink is automatically created in */etc/init.d* pointing to the IBCC service script */opt/ibcc/bin/ibcc*. This will ensure that the IBCC service can be started after machine startup. On RHEL 6, services startup configuration (runlevels etc…) can be configured via the chkconfig utility. On RHEL 7, you might want to add */etc/init.d/ibcc start* to */etc/rc.local* (and don't forget *chmod +x /etc/rc.local*) or create a Systemd configuration.

During the installation process, an SSH key pair will be created automatically and both its public key and fingerprint will be packaged in a zip archive that is to be sent to ING.

To illustrate this point, here is an example of the console output generated during install:

No SSH key pair found, generating one… Public key and fingerprint will be exported as */home/ibccservice /IBCCPubExport_<logon id>_<environment>_<date>.zip*. Please send the exported archive containing SSH public key and fingerprint to ING.

The fingerprint hash, in the above example *00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff* , has to be known to ING in order to allow the client to connect to the ING Sftp server.

Using an existing ssh key

Only *private keys* in PKCS#1 and PKCS#8 PEM format can be used. The *private keys* generated by **ibcc** tooling and **ssh-keygen** are in PKCS#8 PEM format.

ING will only accept *public keys* in SSH2 format, therefore the ones generated with **ssh-keygen** should be converted to SSH2. This can be done by one of the following methods:

With **ibcc** tooling:

```
$ java -jar /opt/ibcc/bin/ibcc.jar ssh-key read -e /export/directory/ /path/to/private.ke
```

This will export the private key in SSH2 format and archive it together with the public key fingerprint, producing the same output as explained above (*/home/ibccservice/IBCCPubExport_<logon id>_<environment>_<date>.zip*).

This archive has to be sent to ING.

Or with **ssh-keygen** directly:

```
$ ssh-keygen -e -f /path/to/key.pub > /path/to/ssh2.key.pub
```

The resulting public key has to be sent to ING.

In the IBCC property file, the field **icb.sftp.key_file** has to point to your ssh private key, for example:

```
icb.sftp.key_file=/path/to/private.key
```

Important: the **ibccservice** user needs read access to the ssh private key.

IBCC configuration properties

After the installation is completed, it is now time to configure the IBCC service.

This is done via a properties file available at the following location: */etc/ibcc/icb.properties*

Simply edit the file and change the values as you see fit.

The overview of the relevant properties is available in this document, see *Appendix A: IBCC properties*.

The service will not start or function properly unless the following properties are set:

- icb.customer.key_file
- icb.customer.password.obfuscated
- icb.sftp.user (provided by ING)

The obfuscated password can be obtained with */opt/ibcc/bin/obfuscate-password.sh*:

```
[root@host ~]# /opt/ibcc/bin/obfuscate-password.sh
Enter the password:
Enter the password, again:
The obfuscated password is:
FCg9qSJfNQr+b4cK8rxCYQ==
```

It is a good idea to review the other properties as well:

- icb.daemon.upload_dir (defaults to */home/ibccservice/transfer/upload/*)
- icb.daemon.download_dir (defaults to */home/ibccservice/transfer/download/*)
- icb.daemon.download.use.product.dirs (enabled by default)
- icb.daemon.archive_dir (defaults to */home/ibccservice/transfer/archive/*)
- icb.archive.mode (defaults to *FOREVER*)
- icb.archive.retention.time (defaults to *168* - in hours)
- icb.sftp.host.acc (defaults to *Acceptance* server)
- icb.sftp.host.prd (defaults to *Production* server)
- icb.sftp.port
- icb.sftp.key_file (defaults to */home/ibccservice/id_ibcc*)
- icb.sftp.key.password.obfuscated
- icb.sftp.upload.retry.mode (defaults to *NEVER*)
- icb.sftp.upload.retry.delay (defaults to *60* - in minutes)
- icb.customer.peer (defaults to *Acceptance* server)
- icb.http_proxy.enable (disabled by default)
- icb.http_proxy.host

- icb.http_proxy.port (defaults to *8080*)
- icb.http_proxy.user
- icb.http_proxy.password.obfuscated
- icb.https_proxy.enable (disabled by default)
- icb.https_proxy.host
- icb.https_proxy.port (defaults to *443*)
- icb.https_proxy.user
- icb.https_proxy.password.obfuscated
- icb.https_proxy.cert_file

Important: The *ibccservice* user must have read/write access on the directories defined with the properties *icb.daemon.upload_dir*, *icb.daemon.download_dir* and *icb.daemon.archive_dir*.

Starting IBCC

After IBCC is configured properly, start it by doing the following:

```
$ sudo service ibcc start
```

It should start successfully. You can always check the IBCC status by doing:

```
$ sudo service ibcc status
```

Updating

Upgrading the IBCC version is done by executing:

```
$ sudo rpm -Uvh InsideBusinessConnectClient_linux_<new_version>.rpm
```

Important: It will not touch the already existing configuration (files under */etc/ibcc/*), except for ING certificates and known_hosts file, that will be overwritten on every update.

Updating ING certificate file manually

ING certificate file (per environment) in */etc/ibcc/* can be updated by overwriting it. The file should have the same name as the existing one.

Uninstalling

Before uninstalling the IBCC service, make sure to stop it:

```
$ sudo service ibcc stop
```

Then, to uninstall the IBCC service, use the following rpm command:

```
$ sudo rpm -e ibcc
```

It will remove everything under the */opt/ibcc/* directory, but will not remove */etc/ibcc/* and */var/log/ibcc/* directories.

The *ibccservice* user will also remain, as well as its home folder (where the transfer directory resides).

The reason for this behavior is that you might want to keep your transfer, configuration or existing log files. If not, you can always remove them manually yourself, as well as the *ibccservice* user.

Troubleshooting

If the service does not start properly, the first action to take is to inspect the log files under */var/log/ibcc/*.

Common problems

The service does not start

Most likely it is a configuration/permission issue. Inspecting the logs in */var/log/ibcc/* will show which property is missing/invalid or what path is inaccessible in case of permission issues.

Solution: Input valid property in the property file or fix permission issues.

Log messages

All the messages below are identified by a unique ID.

IBCC service

ID	Message
685b3162	An error occurred while loading the property file. The property file (*icb.properties*) from configuration directory is not present or not accessible.
930dd5b0	An error occurred while setting up the security context. Possible problems with ING or customer certificate (.pfx). Please see log file for detailed error.
059b657c	An error occurred while uploading IBCC service heartbeat. Possible problems with generating heartbeat content.
f233eec7	An error occurred while purging archive directory. Possible problems with service user access rights to transfer directory. Please check the IBCC service user.
d8092f56	An error occurred while signing the file. Possible problems with data-signing key (.pfx). Please see log file for detailed error.
51d9e0f0	An error occurred while encrypting the file. Possible problems with incompatible encryption algorithm or encoding. Please see log file for detailed error.
374f5b1f	Failed to decrypt file. The received file was not properly encrypted or encrypted with unexpected parameters. Please see log file for detailed error.
016c771a	Failed to verify file. The received file might have been signed by an untrusted source, or its signature is invalid. Please see log file for detailed error.
2159d07e	General upload error. Please investigate the logs for more detailed messages.
0787ae47	Failed SFTP connection while uploading file. Please investigate the logs for more detailed messages.
4cbcff29	Failed SFTP remote operation while attempting to upload a file. The requested remote directory might not exist or its access might be forbidden.
63468fca	Transfer was interrupted when uploading a file. It can be due to network or connectivity issues.

	Please see log file for detailed error.
4466f4db	General download error. Please investigate the logs for more detailed messages.
fcaab123	Failed SFTP connection while attempting to download file(s). Please investigate the logs for more detailed messages.
dea1ea2c	Failed SFTP remote operation while attempting to download file(s). The requested path might not exist or its access might be forbidden.
523c1d2c	Transfer was interrupted when downloading a file. It can be due to network or connectivity issues. Please see log file for detailed error.

Fallback scenario

Please contact ING for information regarding the fallback procedure.

Appendices

Appendix A: IBCC properties

The list below presents all the available properties, their description and default value when applicable.

Application directories

Property name	Description	Default value
icb.daemon.upload_dir	Path to the upload directories. Drop files into an upload directory in order to upload them to the ING SFTP Server. If you have more than one upload folder, separate them with semicolons.	/home/ibccservice/transfer/upload/
icb.daemon.download_dir	Path to the download directory. Location where the IBCC application downloads report files from the ING SFTP Server.	/home/ibccservice/transfer/download/
icb.daemon.download.use.product.dirs	A flag to indicate if sub directories should be created in the download folder for different product names, as it is on the ING SFTP server.	true
icb.daemon.archive_dir	Path to the archive directory. Files that are uploaded to the ING SFTP Server are moved into the archive directory.	/home/ibccservice/transfer/archive/
icb.archive.mode	Three archive modes are available: **FOREVER** (keeps uploaded files forever), **WITH_RETENTION** (keeps files for a certain time, defined by the below property **icb.archive.retention.time**) or **NEVER** (files are deleted after a successful upload)	FOREVER
icb.archive.retention.time	Archive retention time. Files in the archive folder that are older than the retention time are deleted automatically once per day. This property is only effective if the above property **icb.archive.mode** is set to **WITH_RETENTION**	168 (1 week)

Customer security

Property name	Description	Default value
icb.customer.key_file	Path to the customers data signing key. It is used to sign content before uploading to the ING SFTP Server.	N.A
icb.customer.password.obfuscated	Passphrase or password associated with the above key.	N.A

icb.customer.peer	Alias that identifies the ING certificate of the customers encryption and signing peer (at ING).	Acceptance (Acceptance), Production (Production)

ING SFTP endpoint configuration

Property name	Description	Default value
icb.sftp.key_file	Path to the SSH private-key file used to authenticate this host to ING SFTP Server.	/home/ibccservice/id_ibcc
icb.sftp.key.password.obfuscated	Passphrase or password associated with the above key.	N.A
icb.sftp.user	SSH username to logon to ING SFTP server. Each ING customer has a unique account provided by ING	N.A (Given by ING)
icb.sftp.host.acc	ING SFTP Server Acceptance hostname.	coe.insidebusinessconnect.ingwb.com (Acceptance)
icb.sftp.host.prd	ING SFTP Server Production hostname.	insidebusinessconnect.ingwb.com (Production)
icb.sftp.port	TCP port number used by the SFTP server.	6321
icb.sftp.upload.retry.mode	Three upload retry modes are available: **NEVER** (failed uploads are not retried), **FOR_DELAY** (failed uploads are retried for a certain delay, defined by the below property icb.sftp.upload.retry.delay), **FOREVER** (failed uploads are be retried forever)	NEVER
icb.sftp.upload.retry.delay	Upload retry delay. Failed uploads are retried for the specified delay. This property is only effective if the above property icb.sftp.upload.retry.mode is set to FOR_DELAY	60 (1 hour)

HTTP/S Proxy configuration

Property name	Description	Default value
icb.http_proxy.enable	Flag to enable / disable the usage of an HTTP proxy using the below configuration.	false
icb.http_proxy.host	HTTP proxy host name.	N.A
icb.http_proxy.	HTTP proxy port.	8080

port		
icb.http_proxy.user	HTTP proxy username for authentication. If left blank, the proxy is used without authentication.	N.A
icb.http_proxy.password.obfuscated	HTTP proxy password for above username. If left blank, the proxy is used without authentication.	N.A
	WARNING: HTTP proxy accepts password unencrypted, exposing it to man-in-the-middle attacks.	
	Use HTTPS for secure communication.	
icb.https_proxy.enable	Flag to enable / disable the usage of an HTTPS proxy using the below configuration.	false
icb.https_proxy.host	HTTPS proxy host name.	N.A
icb.https_proxy.port	HTTPS proxy port.	8080
icb.https_proxy.user	HTTPS proxy username for authentication. If left blank, the proxy is used without authentication.	N.A
icb.https_proxy.password.obfuscated	HTTPS proxy password for above username. If left blank, the proxy is used without authentication.	N.A
icb.https_proxy.cert_file	Optional HTTPS proxy CA certificate. If left blank, only CAs of the Java Truststore are used. The application does not check for certificate revocation.	N.A

The obfuscated password can be obtained with */opt/ibcc/bin/obfuscate-password.sh*.

When configuring the HTTPS proxy, first attempt to connect without providing any certificate. If the connection succeeds, no certificate is required. If you encounter SSL/certificate errors, provide a certificate. You need to provide the certificate that Java doesn't already trust from your proxy's certificate chain. This is typically one of: - Corporate root CA certificate (most common in enterprise environments) - Proxy server's own certificate (if the proxy uses a self-signed certificate) - Intermediate CA certificate (if missing from the proxy's certificate chain)

If the HTTPS proxy's certificate has a restricted Key Usage, it must include: digitalSignature, keyEncipherment or keyAgreement. If the Extended Key Usage specified, it must include: serverAuth. You can generate a suitable certificate and private key using the following OpenSSL command:

```
$ openssl req -x509 -newkey rsa:2048 -days 730 -keyout proxy.key -out proxy.crt -subj "/CN=your-proxy-server-domain.com" -addext "keyUsage=digitalSignature,keyEncipherment" -addext "extendedKeyUsage=serverAuth"
```