

# e-AS2 Connect

User manual for Version 9.14

Esker EDI Services  
**e-AS2 Connect**

---

© Esker EDI Services  
Calor-Emag-Straße 3 · 40878 Ratingen  
Phone: 02102/479-0 · Fax: 02102/479-109

# Document history

Vers.	Date	Auth.	Comment
1.0	9/22/2011	JK	Original version of the user manual for Version 5.0.
1.0.1	9/30/2011	JK	Updated information on field lengths in the partner table.
1.1	5/31/2012	JK	Original version of the user manual for Version 5.1.
1.2	5/03/2013	JK	Original version of the user manual for Version 5.5. New SSL properties documented. (e-AS2 Enterprise only.)
1.3	7/16/2013	JK	Manual updated for Version 6.0.
1.4	10/10/2013	JK	Manual updated for Version 6.0.2.
1.5	6/13/2014	JK	Manual updated for Version 6.1.
2.0	4/08/2015	JK	Manual updated for Version 7.0.
2.1	03/11/2016	JK	Documented new property settings.
2.2	08/19/2016	JK	Manual updated for Version 7.2.
2.3	11/23/2016	JK	Manual for Version 7.3 / no changes.
2.4	12.04.2019	JK	Updated manual for e-AS2 7.10.
3.0	23.07.2020	JK	Updated manual for e-AS2 8.0. Starting a new major revision of this document. Complete re-read and adaption to the latest version of the software.
3.1	09.09.2021	JK	Updated manual for e-AS2 8.1. Added documentation of new multi provider logic implemented for connections to e-integration's EDI data centers.
3.2	11.01.2022	JK	Updated manual for e-AS2 8.6. Removed references to the JCE policy topic.
3.3	01.03.2022	JK	Updated manual for e-AS2 8.8. Updated information on logging configuration.
3.4	13.10.2022	JK	Updated manual for e-AS2 8.11. Added documentation of new restricted multi provider logic.
4.0	16.11.2022	JK	Updated manual for e-AS2 9.0.
4.1	27.06.2024	MS	Updated manual for e-AS2 9.14.



# Table of Contents

1	Introduction .....	9
2	Quick Start .....	11
2.1	Starting the server .....	11
2.2	Testing the connection .....	12
3	The AS2 protocol .....	15
3.1	General process .....	15
3.2	Establishing a connection .....	16
3.2.1	IP address .....	16
3.2.2	Port .....	16
3.2.3	URI .....	16
3.3	Acknowledgements .....	17
3.4	Encryption and signature .....	17
3.5	Data routing through your provider .....	17
4	The architecture of e-AS2 .....	19
4.1	The e-AS2 server .....	19
4.1.1	Directory Watcher Thread .....	20
4.1.2	Database Watcher Thread .....	20
4.1.3	Incoming Call Handler Thread for HTTP .....	20
4.1.4	Configuration Server Thread .....	20
4.1.5	Command Executor Thread .....	21
4.1.6	Main Thread .....	21
4.1.7	Starting and stopping the server, Windows 10 .....	21
4.1.8	Starting and stopping the server, Linux/macOS .....	23
4.2	The e-AS2 GUI .....	23
4.2.1	Server and GUI on different computers .....	25
5	Firewall and Proxy configuration .....	29
5.1	Firewall configuration .....	29
5.1.1	Incoming connections .....	29
5.1.2	Client-server connection .....	29
5.1.3	Outgoing connections .....	30
5.1.4	Summary .....	30
5.2	Using proxy servers .....	30
5.2.1	Incoming connections .....	30
5.2.2	Outgoing connections .....	31
5.2.3	Authentication .....	31
6	Traditional logic configuration .....	33
6.1	Routing during file transmission .....	33
6.1.1	Routing via file content .....	33
6.1.2	Routing via AS2 relation .....	33
6.2	Configuring the e-integration connection .....	34
6.3	The interface area .....	35
6.4	Sending files .....	35
6.5	Receiving files .....	36
6.5.1	Accepting a connection .....	36

6.5.2	Comparing the URI .....	36
6.5.3	Receiving data, assignment and acknowledgement .....	37
6.6	Preparing for encryption and signature .....	39
6.7	Sending encrypted files .....	41
6.8	Receiving encrypted files .....	42
6.9	Sending signed files .....	42
6.10	Receiving signed files .....	43
6.11	Requesting acknowledgements .....	43
6.12	Setting a retry pattern .....	44
6.13	Managing transactions, error control .....	45
6.13.1	Transaction list, quick overview .....	45
6.13.2	Transaction list, colour coding .....	46
6.13.3	Releasing for re-transmission .....	47
6.13.4	Stopping transactions .....	48
6.13.5	Release MDN .....	48
6.13.6	Transaction details .....	48
7	Multi-provider logic configuration .....	51
7.1	Configuring the e-integration connection .....	51
7.2	The interface area .....	52
7.3	Restricted multi-provider mode .....	54
7.4	Turning off provider distinction .....	55
8	Data fetch extension .....	57
8.1	Motivation .....	57
8.2	The solution .....	57
8.3	Practical handling .....	57
8.4	Auto fetch function .....	58
8.5	Limitations .....	58
9	Command on receive .....	59
9.1	Activating and configuring command on receive .....	59
9.1.1	Activating command on receive .....	59
9.1.2	Retain asynchronous MDN .....	60
9.1.3	Timeout during program invoke .....	60
9.2	Procedure of handling command on receive .....	60
9.2.1	Command line .....	61
9.2.2	Command termination .....	61
9.2.3	Error handling .....	61
9.2.4	Logging .....	61
9.2.5	Functional test during server startup .....	62
10	Configuration for experts .....	63
10.1	log4j2.xml .....	63
10.1.1	Changes to the log level .....	63
10.2	EAS2.properties .....	63
10.2.1	Communication parameters, HTTP .....	64
10.2.2	Parameters for proxy use .....	65
10.2.3	Communication parameters, configuration .....	65
10.2.4	Connect timeout .....	66

10.2.5	Read Timeout .....	66
10.2.6	DNS cache settings .....	66
10.2.7	Enabling autofetch .....	67
10.2.8	Email configuration .....	67
10.2.9	Command on receive .....	68
10.2.10	IO-Trace .....	69
10.2.11	Checking certificate validity .....	69
10.2.12	Automatic backup .....	69
10.2.13	Interface area .....	70
10.2.14	Building file names for incoming messages .....	71
10.2.15	Changing the structure of a message ID .....	71
10.2.16	Preventing sending unfinished files .....	72
10.2.17	Cleanup routines .....	72
10.2.18	Checking for old files .....	73
10.2.19	Language .....	74
10.2.20	List line colouring in the GUI .....	74
10.2.21	GUI main window size .....	74
10.2.22	ToolTip settings .....	74
10.2.23	Controlling provider related processing logic .....	74
11	Reference .....	77
11.1	Partner details, fields .....	77
11.1.1	AS2-From .....	77
11.1.2	AS2-To .....	77
11.1.3	Server .....	77
11.1.4	Port .....	77
11.1.5	URI .....	78
11.1.6	Retries .....	78
11.1.7	Ack mode .....	78
11.1.8	Description .....	78
11.1.9	sign .....	78
11.1.10	encrypt .....	78
11.2	Partner list, functions .....	78
11.2.1	Add partner .....	79
11.2.2	Delete partner .....	79
11.2.3	Edit partner .....	79
11.2.4	Fetch data .....	79
11.2.5	Send certificate .....	79
11.2.6	Test connection .....	79
11.2.7	Close application .....	79
11.3	Certificate details .....	79
11.3.1	Subject .....	80
11.3.2	Issuer .....	80
11.3.3	Serial no. ....	80
11.3.4	Valid from .....	80
11.3.5	Valid to .....	80
11.4	Certificate list, functions .....	80

11.4.1	Update public key .....	80
11.4.2	Generate private key .....	80
11.4.3	Close application .....	81
11.5	Transaction details .....	81
11.5.1	Partner ID .....	81
11.5.2	AS2-From .....	82
11.5.3	AS2-To .....	82
11.5.4	Server .....	82
11.5.5	Port .....	82
11.5.6	URI .....	82
11.5.7	Status .....	83
11.5.8	Direction .....	83
11.5.9	In Process .....	84
11.5.10	Done .....	84
11.5.11	Filename .....	84
11.5.12	AS2 name .....	84
11.5.13	Message ID .....	84
11.5.14	Size .....	85
11.5.15	Created .....	85
11.5.16	Changed .....	85
11.5.17	MIC .....	85
11.5.18	Algorithm .....	85
11.5.19	Error .....	85
11.5.20	Sign MDN .....	86
11.5.21	MDN error .....	86
11.5.22	MDN done .....	86
11.5.23	Retries .....	86
11.5.24	Description .....	86
11.6	Transaction list, functions .....	86
11.6.1	Select transaction .....	86
11.6.2	Only show errors or open entries .....	86
11.6.3	Re-read list .....	86
11.6.4	all .....	87
11.6.5	Release transaction .....	87
11.6.6	Stop transaction .....	87
11.6.7	Release MDN .....	87
11.6.8	Close application .....	88
11.7	Administration .....	88
11.7.1	Server info .....	88
11.7.2	GUI Info .....	89
11.8	Administration, functions .....	89
11.8.1	Manage Server .....	90
11.8.2	Close application .....	90

# 1 Introduction

## *What is e-AS2 Connect?*

e-AS2 is a flexible tool for processing file transfers via EDIINT<sup>1</sup>/AS2<sup>2</sup> protocol. The goal of AS2 is to securely transmit business data over the Internet using the basic communication protocol HTTP<sup>3</sup>.

The product e-AS2 comes in different editions, the most important being e-AS2 Connect and e-AS2 Enterprise. e-AS2 Connect is a limited version of e-AS2 intended solely for connecting to a specific provider. e-AS2 Enterprise is the full version of e-AS2, with the complete functionality enabled.

The goal in developing e-AS2 Connect was to hide any unnecessary complexity from the software to make connecting to the provider's computer centre as easy as possible. Data communication itself, however, remains entirely AS2 compliant. The core protocol was not limited or modified in any way. Some optional protocol elements, however, are not available in e-AS2 Connect.

This document only addresses e-AS2 Connect. Separate documentation is available for e-AS2 Enterprise. First a brief overview of how this manual is structured.

Chapter 2 (*Quick Start*) will guide you in just a few short steps so you can transmit the first test file to your provider.

Chapter 3 (*The AS2 protocol*) provides a brief outline of the key attributes of the AS2 protocol. For detailed specification, please refer to the sources cited in the footnotes on this page.

Chapter 4 (*The architecture of e-AS2*) addresses the architecture of the software. Before starting the configuration you should familiarise yourself with the client/server architecture used in e-AS2.

Chapter 5 (*Firewall and Proxy configuration*) describes the consequences the e-AS2 architecture has on the firewall configuration. It further highlights what to consider when using proxy servers.

Chapter 6 (*Traditional logic configuration*) describes the software configuration based on usage examples which build on each other. Each usage example requires a subset of available configuration options.

Chapter 8 (*Data fetch extension*) describes a distinctive feature which sets e-AS2 Connect apart from other AS2 products. The data fetch extension allows e-AS2 Connect to run on systems which are not permanently online.

In Chapter 9 (*Command on receive*) you can learn how to connect e-AS2 tighter to adjacent systems when processing incoming data.

Chapter 10 (*Configuration for experts*) documents the configuration options not available from the graphical user interface. These are settings which in most cases do not need to be changed.

Chapter 11 (*Reference*) systematically lists and describes all the options of the graphical user interface again, without using a specific usage scenario.

---

<sup>1</sup>Electronic Data Interchange Internet Integration, see <http://www.ietf.org/html.charters/ediint-charter.html>

<sup>2</sup>MIME Based Secure Peer to Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2), see <http://www.ietf.org/rfc/rfc4130.txt>

<sup>3</sup>Hypertext Transfer Protocol, see <http://www.ietf.org/rfc/rfc2616.txt>

## Introduction

The core of e-AS2 is entirely written in Java and will therefore generally run on any system with a Java Runtime Engine (JRE) installed.<sup>4</sup> This user manual is primarily based on an e-AS2 installation running on MS Windows 10. Users of other operating systems should still be able to easily apply the text to their environment. The operating systems will be addressed separately where deemed necessary due to differences in the architecture.

---

<sup>4</sup>see <http://www.oracle.com/technetwork/java/index.html>

## 2 Quick Start

*Here you will learn how to transmit a file to your provider in just a few simple steps using e-AS2 Connect.*

This chapter will show you how to quickly prepare e-AS2 Connect for use following installation. The description assumes you installed e-AS2 Connect for provider connection to the e-integration EDI Clearing Center. If you connect to a different provider, you selected the respective choice during setup. You will see a fitting preconfigured partner profile and apart from that carry out the same steps described here.

### 2.1 Starting the server

You will first need to start the e-AS2 server.

#### MS Windows

Start e-AS2 from the start menu using the “start e-AS2-Server” menu item.

A graphical server start front end will appear, informing you of the progress of this process.<sup>5</sup> Once the server has been started a respective success message will appear.

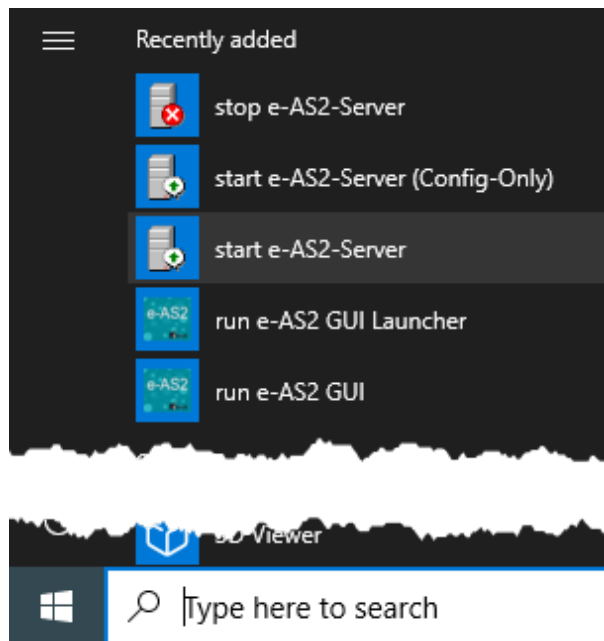


Figure 2.1. e-AS2 Start server from start menu

<sup>5</sup>Depending on how the operating system is configured you will first see User Account Control, prompting you whether you wish to approve this process. Respond with “Yes”.

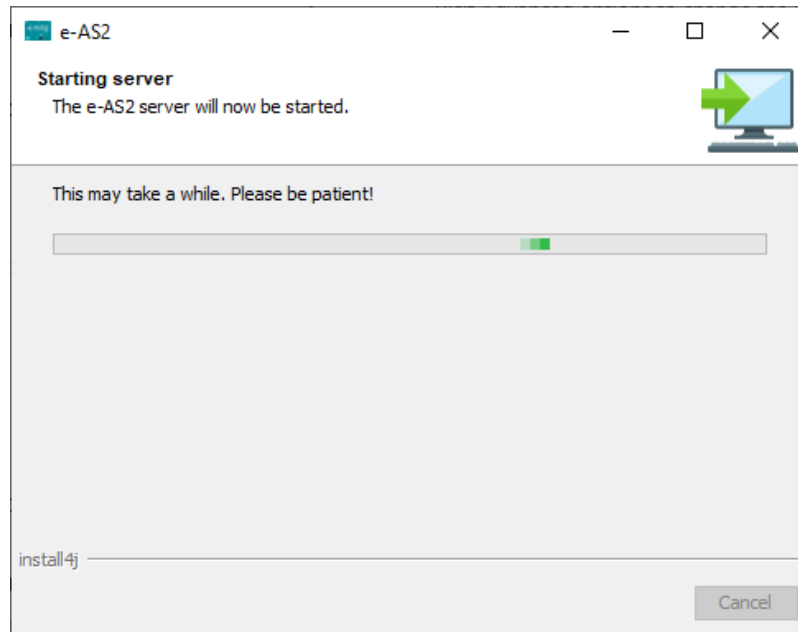


Figure 2.2. e-AS2 Start server

### Unix/Linux/Mac OS X

Switch to the e-AS2 Connect installation directory. Here, enter the command

```
./runserver.sh
```

to start the server. The running server will then block the shell until closed by typing Ctrl-C on the keyboard. Alternatively enter the command

```
./eas2s.srv start
```

This will start the e-AS2 server in the background. The server is ready to use as soon as the shell prompt is returned.

## 2.2 Testing the connection

Once the e-AS2 server is running, start the GUI client.

### MS Windows

Start the graphical user interface from the start menu using the “start e-AS2 GUI” menu item.

### Unix/Linux

Start the graphical user interface using the command

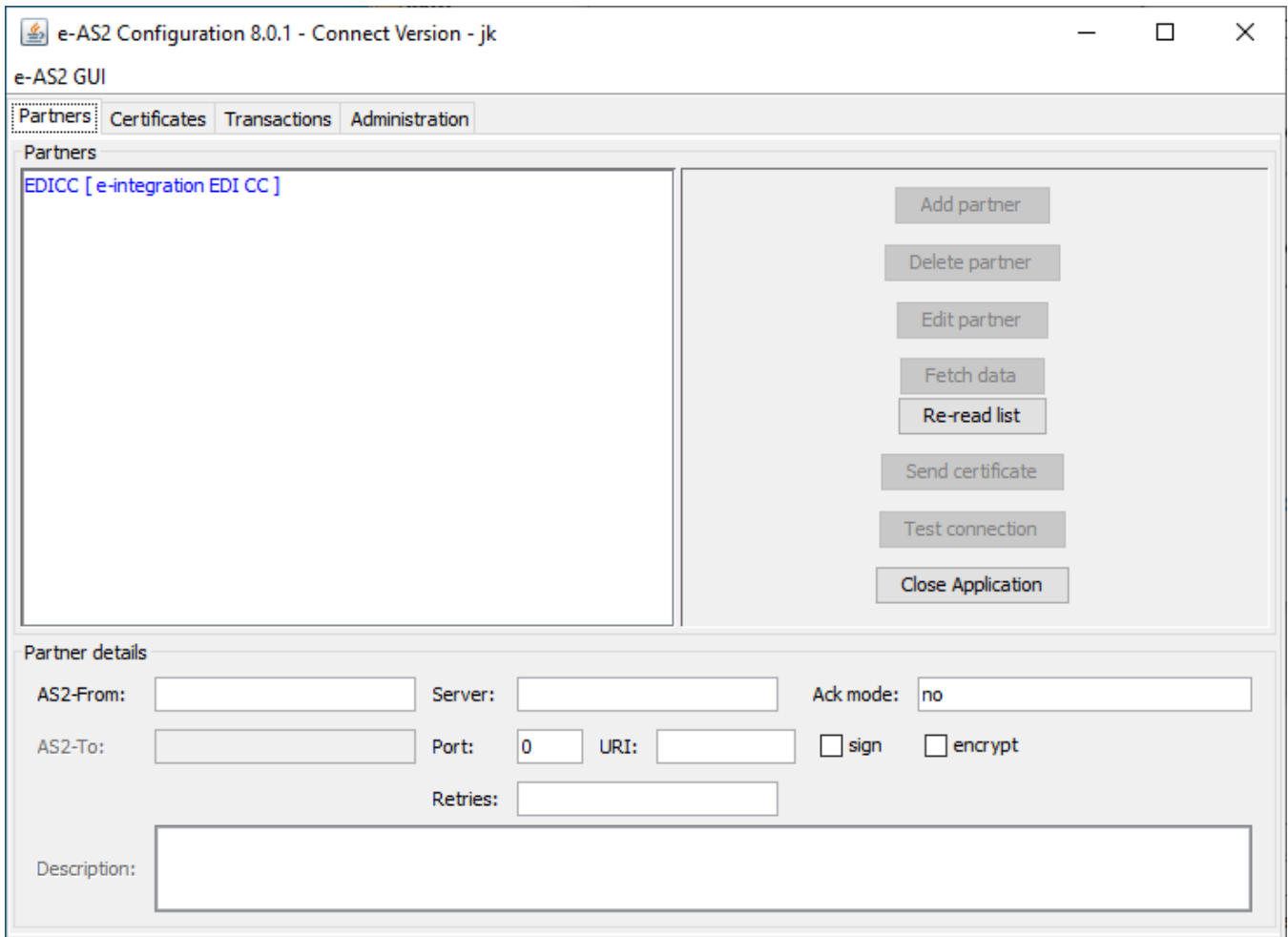
```
./rungui
```

### Mac OS X

Start the graphical user interface by double-clicking the program

```
rungui
```

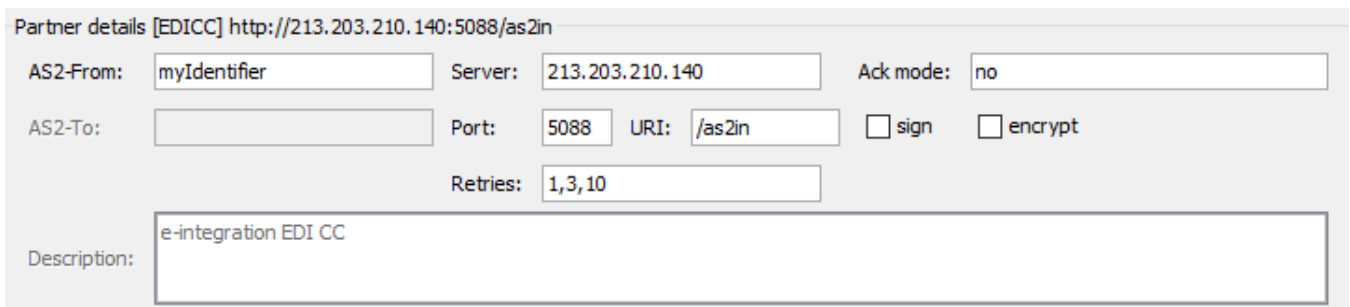
You can also drag this program to the dock and start it from there.



**Figure 2.3. The e-AS2 Connect configuration interface**

The GUI client will connect to the server and display the configuration interface as shown in Figure 2.3, “The e-AS2 Connect configuration interface”.

Select the entry “EDICC [ e-integration EDI CC ]”<sup>6</sup> with your mouse. Then press “Edit partner”. This will open the edit dialogue for the partner details. In field “AS2 From” enter your identifier you received from e-integration<sup>7, 8</sup>. Save this change. You should then see the following partner details at the bottom of the dialogue.



**Figure 2.4. The partner details**

<sup>6</sup> or the entry for your provider

<sup>7</sup> or your provider

<sup>8</sup> If you have not yet received an identifier, use a made up identifier for now as close to your company name as possible. Only use letters and numbers. (Case sensitive.)

You are now ready to test the connection. A connection to the e-integration CC<sup>9</sup> will be established and a generated test file transmitted. To do so, press the “Test Connection” button. Following successful transmission of the test file a message screen will appear to confirm.

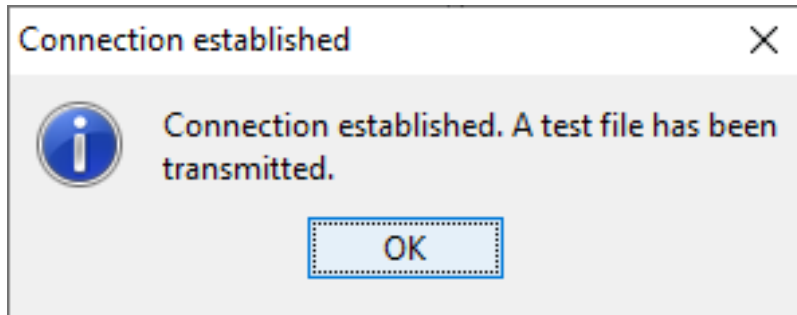


Figure 2.5. Transmission OK

In the GUI activate the tab “Transactions” and press the “Re-read list” button. In the transactions list you will see a new entry for the file that was transmitted.

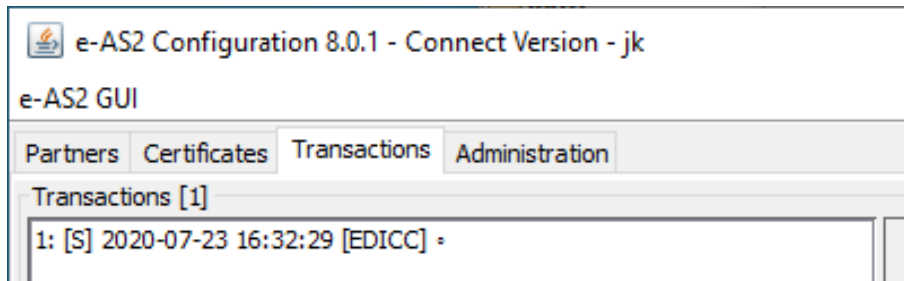


Figure 2.6. An entry in the transactions list

If you’ve made it this far without any problems, you can now proceed straight to Chapter 6 (*Traditional logic configuration*). If one of the steps didn’t work as described, please refer to the instructions for Chapter 3 (*The AS2 protocol*), Chapter 4 (*The architecture of e-AS2*) and Chapter 5 (*Firewall and Proxy configuration*) for more details on how the software works. This will certainly be useful in diagnosing why the quick start was not successful. If this doesn’t help after all, please contact your provider for support!

In the event the connection test fails, the message screen which appears will show an error message, e.g.:

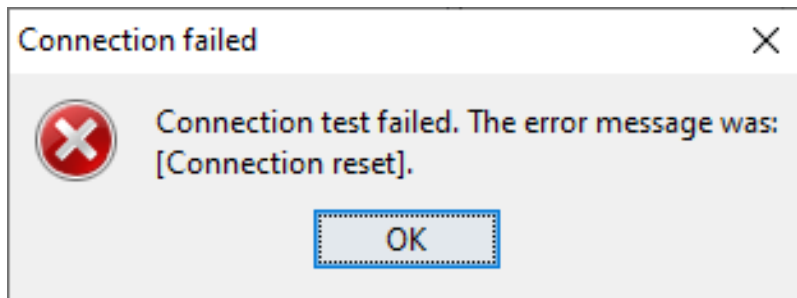


Figure 2.7. Transmission Failed

Please relay the text in the square brackets to your provider. This error message will help the support staff diagnose the cause.

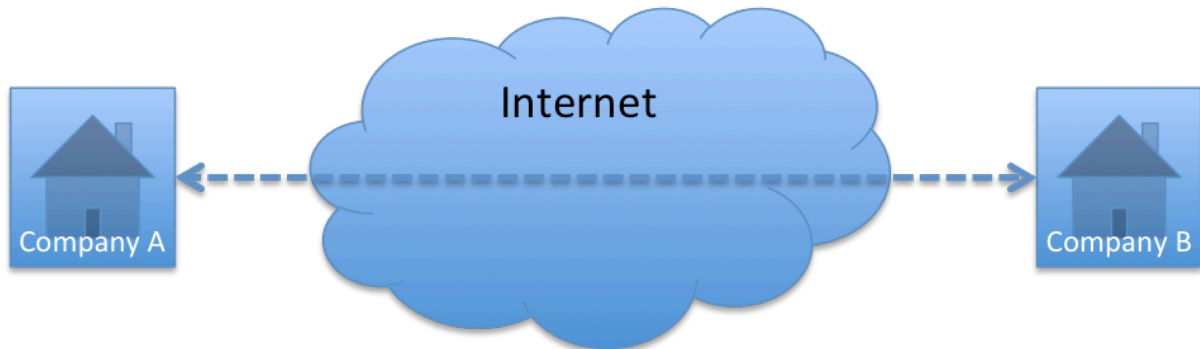
<sup>9</sup>or your provider

### 3 The AS2 protocol

*Explanations on the communication protocol EDIINT/AS2.*

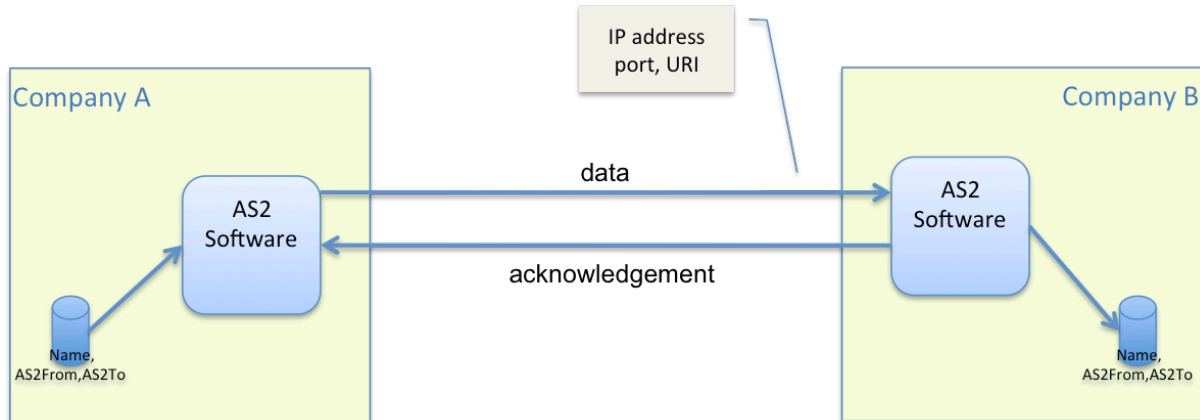
AS2 is defined as a protocol for secure data transmission over the Internet. In our example, company A and company B are exchanging business data. Both have an Internet connection. However, there is no direct link between the IT systems of these two companies.

#### 3.1 General process



**Figure 3.1. Data exchange between company A and company B**

AS2 is based on HTTP and allows the two companies to exchange data using the available permanent Internet connection.



**Figure 3.2. Data exchange via AS2**

The process is approximately as follows. Company A wishes to transmit a file to company B. The AS2 software at company A assigns three key attributes to the company: its name and the pair (AS2From, AS2To). These are the AS2-related identifiers of the two parties exchanging data. I.e. prior to beginning the data exchange, company A and company B agree on this pair of identifiers to clearly identify their data exchange relationship.

Company A encodes the file name, AS2From and AS2To along with the file contents according to MIME<sup>10</sup> standard. The AS2 software at company A then establishes an HTTP connection to company B via the Internet and transmits the MIME message to the AS2 software of company B. Company B acknowledges during the same session, which is then ended. The AS2 software

<sup>10</sup>Multipurpose Internet Mail Extensions, see <http://www.ietf.org/rfc/rfc2045.txt>

of company B can obtain the name of the file and the (AS2From, AS2To) pair from the MIME message received, and use it to further process the data.

## 3.2 Establishing a connection

As specified in Figure 3.2, “Data exchange via AS2”, establishing a connection requires three parameters:

1. IP address
2. Port
3. URI

Explaining the main features of TCP/IP-based communication is beyond the scope of this document. The following should suffice as a description. For more detailed information, please refer to widely available literature on this topic.

### 3.2.1 IP address

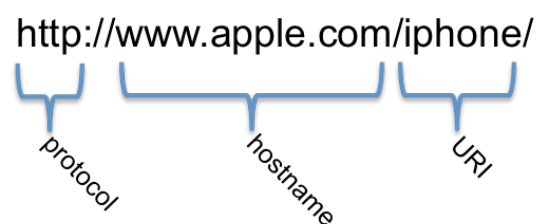
The IP address clearly identifies the computer to which a connection is to be established. Whilst IP addresses on local networks can largely be arbitrarily defined, IP addresses of computers accessible over the Internet must be globally registered to prevent duplicates. The IP address consists of four numerical values separated by dots, e.g. 66.249.93.99.<sup>11</sup>

### 3.2.2 Port

The IP address provides access to a clearly identified computer on the Internet. However, a computer could have several different services implemented at the same time, e.g. FTP, e-mail, web server or AS2 server. Selecting the service on the destination system therefore requires another number in addition to the IP address, the port number.

### 3.2.3 URI

A service on a specific computer on the Internet selected via IP address and port in turn can have various services which must be distinguished. With HTTP this is done through the POST request URI, or URI for short<sup>12</sup>.



**Figure 3.3. Post request URI, example**

In the above example the page for iPhones is selected from the Apple websites. The address begins with `http` as the protocol identifier. It is followed by `www.apple.com` as the computer name, and then `/iphone`. This is the URI within the “Web Server” service on `www.apple.com` which selects the services for “information about the iPhone”.

<sup>11</sup>To make it easier to access computers on the Internet, most cases use computer names in place of IP addresses, mapped to IP addresses via DNS (Domain Name Service).

<sup>12</sup>The abbreviation URI means Uniform Resource Identifier.

Since AS2 builds on HTTP, it too uses an URI to address the potential different applications on one AS2 server identified by the IP address and port. In many cases, e.g. different URIs are specified for test and production service.

### 3.3 Acknowledgements

By default, the AS2 software of company B returns a very short acknowledgement message to the sender of the data specifying the complete MIME message was received. This acknowledgement does not specify if the data was processed successfully.

The AS2 protocol provides for advanced acknowledgements, so-called Message Disposition Notifications (MDN), used to send additional information about the success of processing to the sender of the data.

Which form of acknowledgement is used is determined by the sender of the data. If the sender requests an MDN, the receiver must satisfy this request.

The type of acknowledgement, that e-AS2 as the sender of data requests from the receiver, can be configured per partner system.

### 3.4 Encryption and signature

AS2 provides the use of asymmetric cryptography methods for securing data transmission. At this point we should also mention an exhaustive description of this topic would be beyond the scope of this documentation. Again, please refer to the literature available on this topic. Here a few general descriptions.

With encryption, the data exchanged between two communication parties cannot be viewed by third parties. The encryption is specific to the respective receiver of the data. Only he can allow the encrypted data to be read again. This also protects the data from unauthorised access along the path of communication over the Internet.

A signature allows the receiver to clearly assign the data to the sender. If signature validation is successful following receipt, he will not only know this was actually the sender stated when sending the MIME message, but also that the data was not modified during transmission. The integrity of the data is guaranteed.

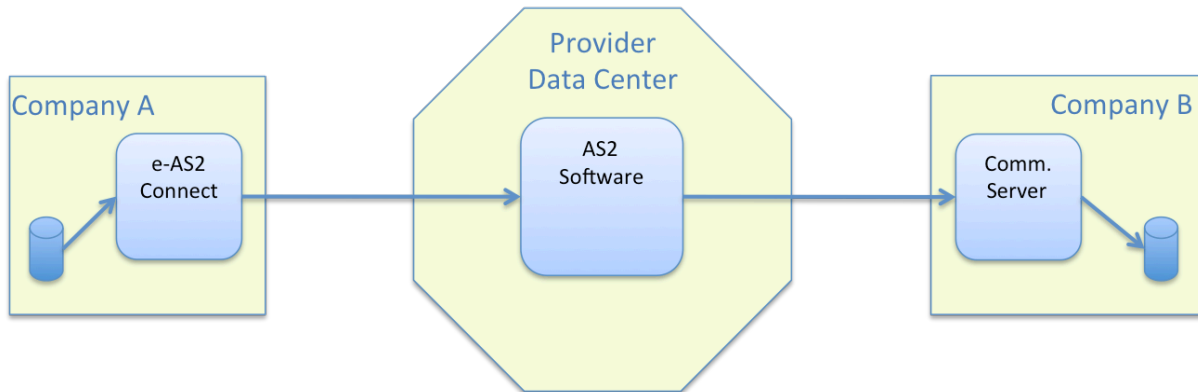
Encryption and signature can be combined. Whether to encrypt and/or sign is determined by the sender of the data. Using cryptography is not mandatory in order to comply to the protocol specification. However a lot of companies won't allow unencrypted message exchange due to company policies.

AS2 cryptography is based on so-called certificates which must be exchanged between the two communicating parties in advance.

*Note:* We have attempted to make using cryptography as simple as possible for you as the e-AS2 Connect user. In just a few simple steps you enable cryptography for your e-AS2 installation. Learn more about this in Chapter 6 (*Traditional logic configuration*).

### 3.5 Data routing through your provider

The previous description assumed the two business partners company A and company B are exchanging data directly. e-AS2 Connect is intended to allow you to reach your business partner via your provider's computer centre. You only establish a single AS2 connection, namely to your provider. It forwards your data to your business partner, and vice versa. This also allows it to be forwarded in both directions, if your business partner does not use the AS2 protocol for communication.



**Figure 3.4. Using e-AS2 Connect with a provider**

The above illustration shows a scenario where the data is transmitted between company A and a provider's computer centre via AS2 protocol. The provider will then forward the data to company B using a communication protocol which is not further specified.

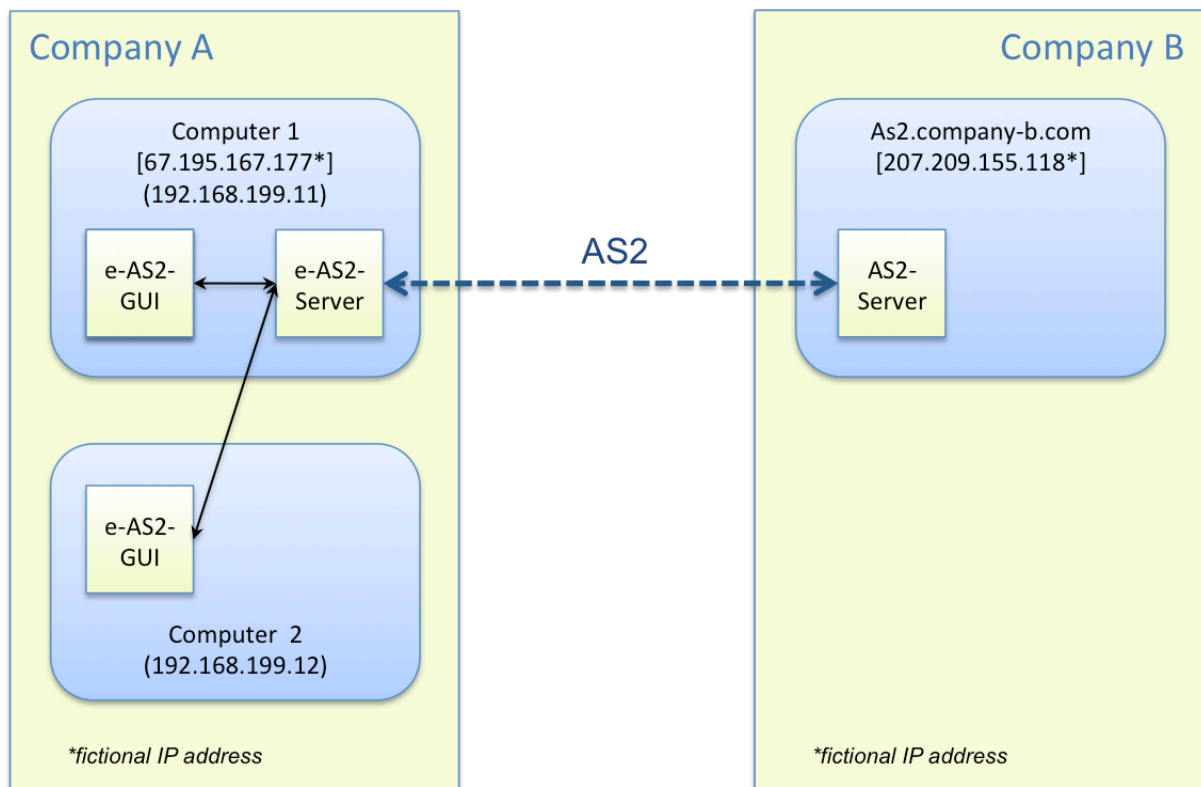
e-AS2 Connect with designed specifically for this type of data exchange through a provider.<sup>13</sup> In Germany e-AS2 Connect will connect directly to the e-integration computer centre. For the description of the configuration in Chapter 6 (*Traditional logic configuration*) we will always keep this constellation in mind.

<sup>13</sup>e-AS2 Connect is not suitable for direct point-to-point communication between several parties. This will require the full version e-AS2 Enterprise.

## 4 The architecture of e-AS2

*Illustration of the client/server architecture of e-AS2 with setup information.*

e-AS2 consists of two separate components, the e-AS2 server and the e-AS2 GUI. The e-AS2 server largely runs quietly in the background. Its task is to handle all send and receive processes per AS2 communication. (This corresponds with the thick dashed arrow in the picture below.) The e-AS2 GUI is the graphical configuration interface for e-AS2 which connects to the running e-AS2 server via a specific client/server protocol. (This corresponds with the thin continuous arrows in the picture below).



**Figure 4.1. Schematic illustration of the e-AS2 architecture**

The advantage of this architecture is that the server and configuration interface are not required to run on the same computer, hence are not required to run on the same operating system. The e-AS2 server can therefore run on Unix on a computer in the company computer centre and the GUI client on Windows or Mac OS X on a desktop PC. This requires the firewall configuration to allow the necessary connections. More on this later.

### 4.1 The e-AS2 server

The e-AS2 server is a Java program which uses multithreading<sup>14</sup> to implement various services. Right after starting the server, the main thread starts other important basis threads so the following threads are permanently active in basic state:

1. Directory Watcher Thread [DIRWATCHER<sup>15</sup>]

<sup>14</sup>If you're not familiar with the term thread, simply imagine it as a separate delimited entity within a running computer program. Multithreading means several of these entities are active at once.

<sup>15</sup>Messages from the Directory Watcher Thread have this identifier in the log file.

2. Database Watcher Thread [DBWATCHER]
3. Incoming Call Handler Thread for HTTP [ICH\_HTTP]
4. Configuration Server Thread [CFG]
5. Command Executor Thread [CMDEXEC]
6. Main Thread [main]

If necessary, additional threads will be started to handle pending tasks.

#### 4.1.1 Directory Watcher Thread

The Directory Watcher Thread permanently monitors the interface directory tree to determine if new files are pending transmission. This makes sending files particularly easy. Copying a file into a directory is sufficient. Once the Directory Watcher detects them<sup>16</sup> a management entry is recorded in the transaction table in the database in e-AS2 and file transmission initiated. The transaction table can be viewed in the GUI, under the "transactions" tab.

The Directory Watcher runs on a 20 second cycle. I.e. there is always a 20 second pause between two runs through the interface directory. The Directory Watcher is only responsible for the first send attempt of the file. If the first attempt – for whatever reason – fails, the Database Watcher takes on further handling.

#### 4.1.2 Database Watcher Thread

The Database Watcher Thread is responsible for managing all send transactions. As soon as a transaction has been generated based on a file in the interface area, the Database Watcher takes on responsibility. In general, however, it has quite little to do, since if the first send attempt for the file is successful, the transaction immediately switches to status "finished" and there is nothing left to do.

If the file transfer fails,<sup>17</sup> the Database Watcher takes over and ensures the attempts are repeated according to the configured retry pattern. The Database Watcher runs on a 60 second cycle.

#### 4.1.3 Incoming Call Handler Thread for HTTP

The Incoming Call Handler Thread for HTTP handles the task of waiting for and responding to incoming data connections. Once such a connection occurs, it triggers data reception being processed, after which it right away prepares for additional connections again.

Files received are stored in the interface area. In addition, a respective entry in the transaction table is generated for each file received.

While waiting for incoming data connections, the e-AS2 server uses the port number specified for data communication during installation. If you accepted the default values, it is port 5080.<sup>18</sup>

#### 4.1.4 Configuration Server Thread

The Configuration Server Thread handles managing the e-AS2 configurations objects accessible through the GUI. These are the partner profiles, the certificates and the transactions. Configuration changes done by the GUI will be made persistent by the server, never the GUI itself.

---

<sup>16</sup>A security mechanism ensures new files will only be detected for sending once they are fully written. Files still being built are ignored. (also see Section 10.2.16 (*Preventing sending unfinished files*))

<sup>17</sup>Reasons or transfer attempts failing are incorrect configurations (e.g. incorrect IP address) or even the Internet connection failing briefly. In this event the transaction in the database contains an error message which will help you with diagnostics.

<sup>18</sup>Please also note Chapter 5 (*Firewall and Proxy configuration*).

The nature of the client/server architecture when configuring e-AS2 is that the GUI client notifies the server which database operations need to be executed, but never executes them itself. Only this will ensure the GUI and server can run on different computers.

The e-AS2 server waits for incoming configuration connections using the port specified as the port number for configuration during the installation. If you kept the defaults, this is port 5070.

#### 4.1.5 Command Executor Thread

The command executor thread is only launched if the execution of successive commands was activated and the program for executing successive commands was found and was able to be invoked in test mode.

This thread monitors the data in directories in the interface area and if necessary starts the successive command with suitable parameters. You can find details on this topic under Chapter 9 (*Command on receive*).

#### 4.1.6 Main Thread

Once the main thread has started the other basis threads it stays idle. The main thread becomes active again when the server is shut down.

#### 4.1.7 Starting and stopping the server, Windows 10

The e-AS2 installation routine sets up the e-AS2 server as a service in the operating system.<sup>19</sup>

##### Open Service Manager

To access Service Manager, enter `services` in the search field next to the start button, then select the menu item `Services`.

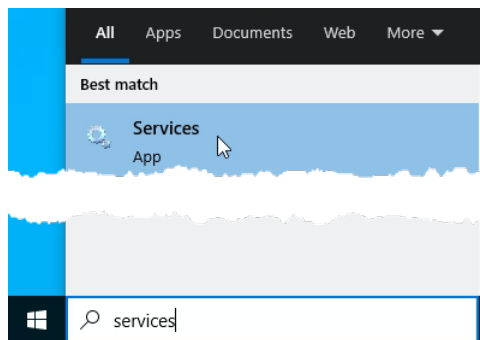


Figure 4.2. Open the Services app

You will find the e-AS2 server in the list of services sorted in alphabetically:

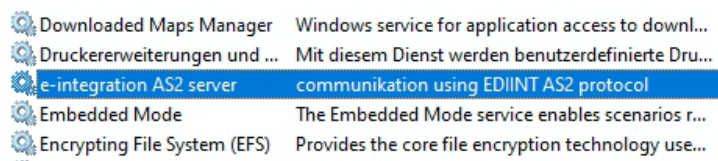


Figure 4.3. The e-AS2 server in the Services list

<sup>19</sup>Here we will be focusing on the latest release of MS Windows. Some minor details will appear a bit different in older versions of Windows. The concept, however, is the same.

Here you will also see the option to start or stop the e-AS2 service.

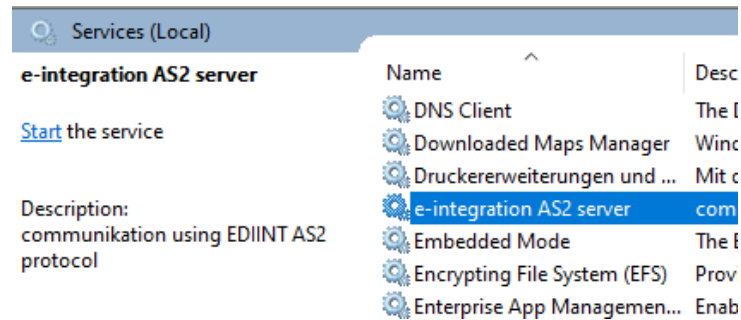


Figure 4.4. Starting and stopping the e-AS2 service

The auto-start type for this service is preset to automatic. With this setting the e-AS2 service will automatically be started after restarting the computer without a user needing to log into the system. This also makes e-AS2 on Windows suitable for running unattended in a data center environment. If you do not want this, in Services Administration please change the auto-start type from automatic to manual.

As an alternative to using the services manager you can use the app entries for e-AS2 added during the installation routine, which can be found in the Windows Start Menu.

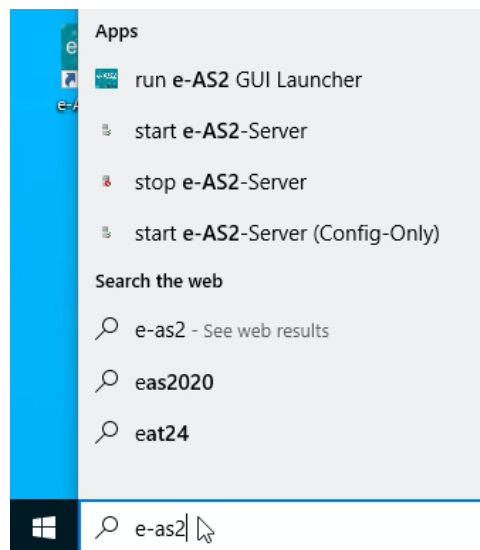


Figure 4.5. Starting and stopping the e-AS2 service

### Start e-AS2

Use the app “start e-AS2-Server” to start the service.

The two methods for starting and stopping the server, using the service manager and the Start app, are equivalent and can be used arbitrarily.

### Stopping e-AS2

Use the app “stop e-AS2-Server” to stop the service.

**Note:** Alternatively, the server can also be stopped using the e-AS2 GUI. You can access the option using the “Manage server” under the tab “Administration”. The server cannot be started from the GUI.

#### 4.1.8 Starting and stopping the server, Linux/macOS

As a Linux or macOS user you’re familiar with the shell and how to use command-line programs. The software can be installed and run on any account. We do not recommend using the root account for this purpose. For easy data exchange between e-AS2 and the subsequent processing systems, e-AS2 can certainly also be run from an existing account with the processing software.

##### Starting e-AS2 in the foreground

Switch to the e-AS2 installation directory. Here, enter the command

```
./runserver.sh
```

to start the server. The shell will then be blocked by the running server until it is ended using Ctrl C.

This option for starting e-AS2 will work with any platform compatible with Java, as it only involves Java Code. The console will only display a brief start message. All other messages from the running server will be in the `eas2s.log` file.

##### Starting and stopping e-AS2 in the background

Switch to the e-AS2 installation directory. Here, enter the command

```
./eas2s.srv start
```

. This will start the e-AS2 server in the background. Once the Shell Prompt is returned the server is ready for use. Use

```
./eas2s.srv stop
```

to stop the e-AS2 server.

This option uses the Service Wrapper by Tanuki Software<sup>20</sup>. This provides you with a basis to adequately integrate the e-AS2 server into your operating system environment to automatically start and stop when starting and shutting down the system. All running messages from the server will also be under `eas2s.log` in this case.

**Note:** Alternatively, the server can also be stopped using the e-AS2 GUI. You can access the option using the “Manage server” button in the “Administration” tab. The server cannot be started from the GUI.

## 4.2 The e-AS2 GUI

The e-AS2 GUI serves as a configuration and administration user interface for e-AS2. The following tasks can be executed using the e-AS2 GUI:

- View/edit partner profile
- Manage cryptography certificates

<sup>20</sup>S. <http://wrapper.tanukisoftware.org>. This tool is not written in Java and is output by e-integration as platform-dependent binary code. The installation routine will try to detect if your system is supported and, if possible, also install the Service Wrapper. But since not all correlations of CPU type and operating system version can be anticipated, we generally cannot guarantee the Service Wrapper will work on your system.

- Monitor transaction list
- Manually stopping transactions
- Scheduling erroneous transactions for re-processing
- Request data retrieval
- Test connection
- View GUI client and e-AS2 server version information
- Perform server management tasks, end server

First start the e-AS2 server as described in Section 4.1 (*The e-AS2 server*). Then you can then launch the GUI.

#### MS Windows

Start the graphical user interface for the Windows start screen using the app “Start e-AS2 GUI”.

#### Unix/Linux

Start the graphical user interface using the command  
`./rungi`

#### Mac OS X

Start the graphical user interface by double-clicking the program  
`rungi`

You can also drag this program to the dock and start it from there.

If you decided to place a shortcut on the Desktop during installation, you will have a convenient alternative for starting the GUI.<sup>21</sup>

The GUI client will connect to the running server and display the graphical user interface. If the GUI Client is unable to connect to the server, you will see an error message.

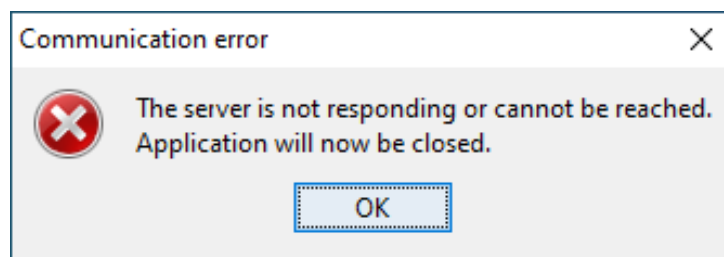


Figure 4.6. e-AS2 server not available

Otherwise the user interface will start as follows.

<sup>21</sup>Desktop shortcuts are available in MS Windows and Mac OS X, as well as Linux when using KDE or Gnome.

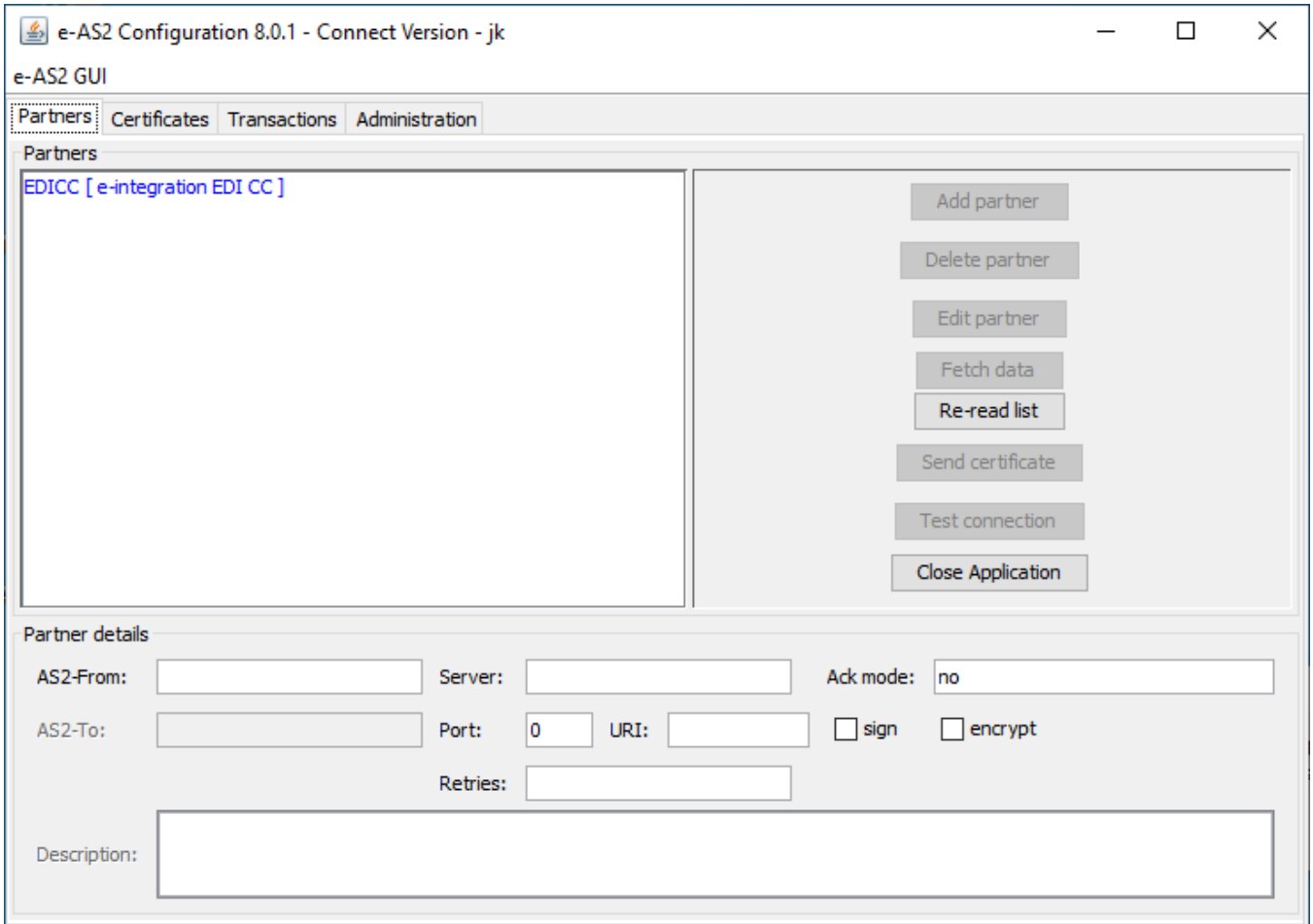


Figure 4.7. e-AS2 GUI

The different application areas of the graphical user interface can be accessed using the four tabs Partners, Certificates, Transactions and Administration.

**Note:** Pressing the “Close Application” button will only end the GUI client, not the server. To end the server, use the Manage server button under the “Administration” tab or use methods presented under Section 4.1 (*The e-AS2 server*).

#### 4.2.1 Server and GUI on different computers

Scenarios where the e-AS2 server and e-AS2 GUI run on different computers are explicitly supported by the installation routine. In this case you will only install the e-AS2 server component on one computer and only the e-AS2 client component on one or more other computer(s). We will go through this type of installation with examples. We will assume a scenario as documented in Figure 4.1, “Schematic illustration of the e-AS2 architecture”. First, e-AS2 Connect is installed on computer 1. In the process e.g. the following parameters are entered during setup:

Parameter	Value
Host name (data traffic)	computer1 <sup>22</sup>

<sup>22</sup>In place of the name you could also specify the respective IP address 192.168.99.11. Setup will automatically enter the computer name as the preset. This can normally simply be accepted without changing. Also read the explanations on this under Chapter 5 (*Firewall and Proxy configuration*).

Parameter	Value
Port number (data traffic)	5080
Port number (configuration)	5070
Password (configuration)	topsecret

The two key steps during server installation with this scenario are illustrated under Figure 4.8, “e-AS2 installation - server only” and Figure 4.9, “e-AS2 server installation - parameters”.

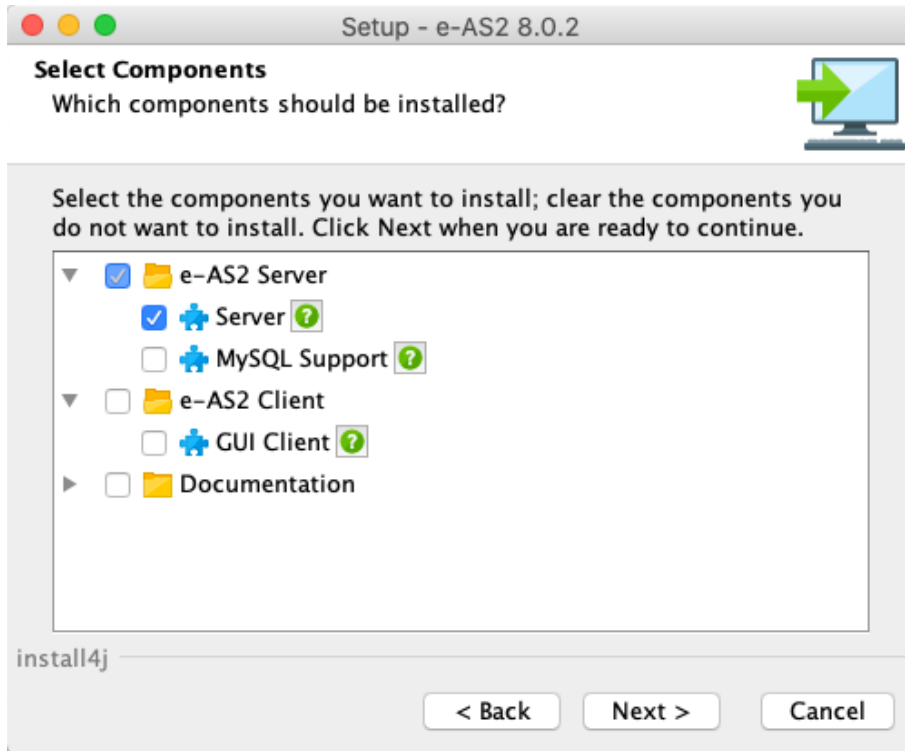


Figure 4.8. e-AS2 installation - server only

When you’re prompted to select the components during installation, deselect the GUI client component. You’re only installing the server component.

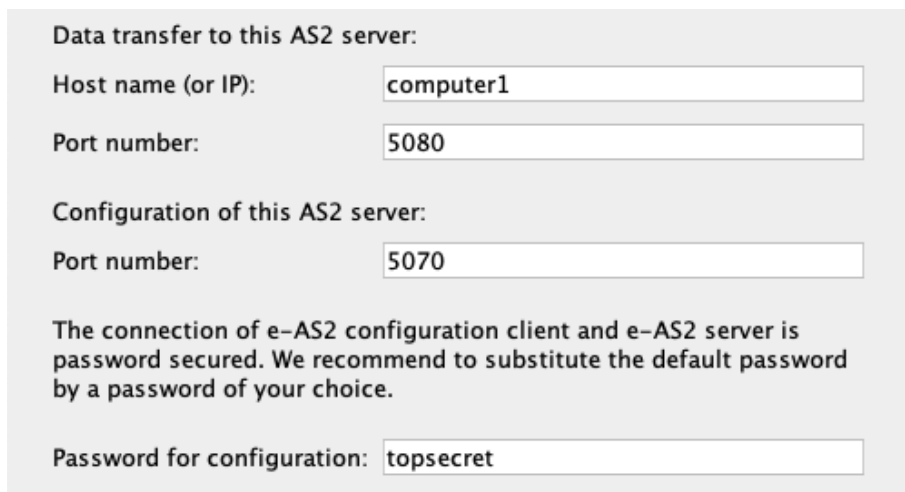


Figure 4.9. e-AS2 server installation - parameters

The form to enter communication parameters for the server end will then be tailored to this constellation. You will enter the parameters as specified in the table above.

After that, e-AS2 will be set up on computer 2. In Figure 4.10, “e-AS2 installation - GUI client only” and Figure 4.11, “e-AS2 GUI client installation - parameters” you will see the two key steps for this installation.

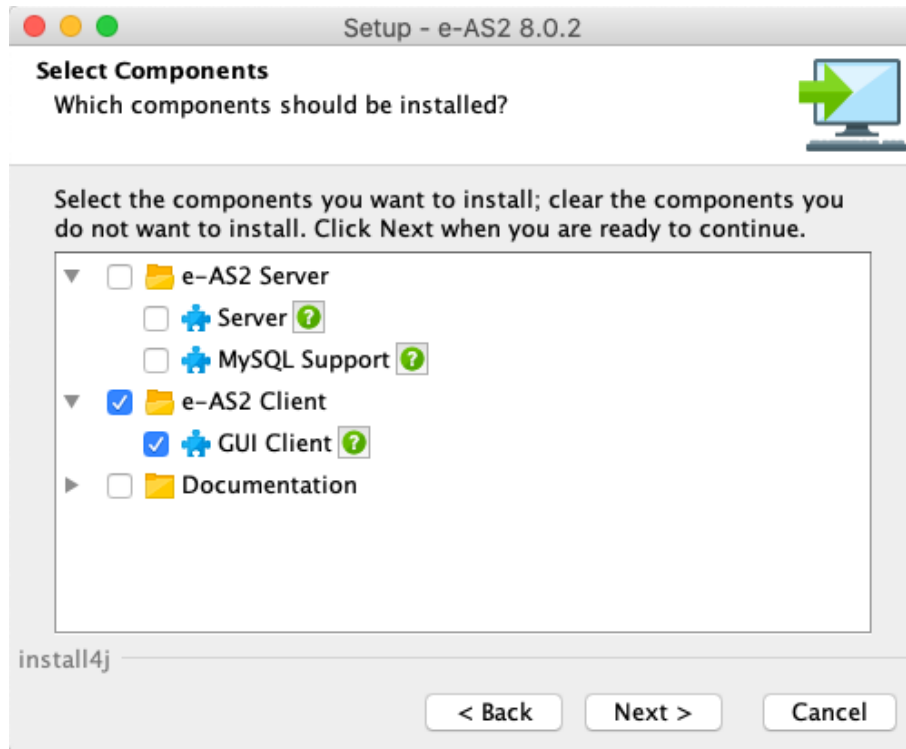


Figure 4.10. e-AS2 installation - GUI client only

When you’re prompted to select the components during installation, deselect the server component now. You’re only installing the GUI client component.

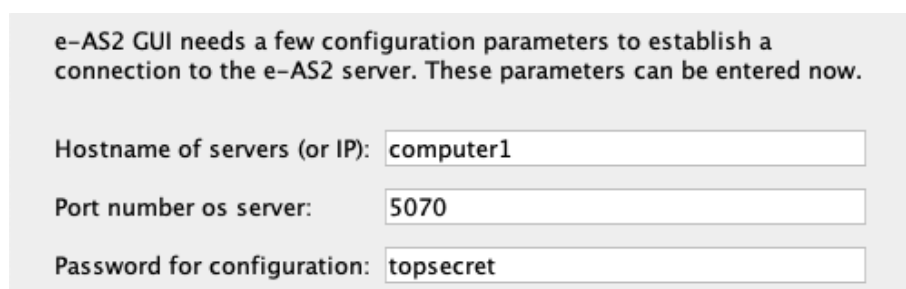


Figure 4.11. e-AS2 GUI client installation - parameters

The dialogue for entering the communication parameters will be displayed adapted to this scenario. You will enter the parameters as specified in the following table.

Parameter	Value
Server host name	computer1
Server port number	5070
Password (configuration)	topsecret

These parameters are in line with the parameters entered during server setup. Now when you start the GUI client you will automatically be connected to this server. If there were errors when entering the parameters, you will see error messages when starting the GUI client. You can change any incorrect parameters later without reinstalling the software. How, is described in Section 10.2.3 (*Communication parameters, configuration*).

## 5 Firewall and Proxy configuration

*Configuring your firewall to run e-AS2 safely. Information on using Proxy servers with e-AS2.*

When running e-AS2 on a company network, the network will most likely have a central firewall. Please give this document to your firewall administrator so he can handle the necessary configuration. You can then skip this chapter and continue with Chapter 6 (*Traditional logic configuration*), however only after your firewall administrator has confirmed he completed the necessary configuration.

If you are responsible for configuring the firewall for your company network, please read this chapter and perhaps also the appendix for Chapter 4 (*The architecture of e-AS2*) to understand the reason for this requirement.

If you're running e-AS2 Connect on a single-user system with dial-up Internet connection, you are responsible for configuring your firewall on your system yourself. The same also applies if in addition to the company firewall your system also has a personal firewall (e.g. the Windows firewall) enabled. Any firewall involved in any way can impact communication between the e-AS2 GUI and e-AS2 server, as well as between the e-AS2 server and your provider's computer centre.

We will now first address the specific aspects of the firewall configuration for e-AS2. This will be followed by a look at the additional factors to consider when using a proxy server.

### 5.1 Firewall configuration

Different requirements for using an e-AS2-compatible firewall configuration can be formulated depending on the type of communication and direction being considered.

#### 5.1.1 Incoming connections

The computer where the e-AS2 server will be installed must be a system which is always online and available over the Internet. If this is not the case, you will not be able to receive data with e-AS2. The AS2 protocol dictates files always are actively transported from the sender to the receiver. The AS2 protocol offers no option for retrieving files from a remote server, as is the case with FTP.<sup>23</sup>

When installing e-AS2 on the selected computer, specify a port number for data traffic during installation. This port number will be used for incoming connections. When using the default, this will be port 5080.

So your firewall configuration must allow incoming connections from the Internet to the selected computer via port 5080 (or the port specified during setup).

Please note the port specified here should be considered from perspective of the server running the e-AS2 software. The firewall can and may certainly be mapped for a different port number using port forwarding.

#### 5.1.2 Client-server connection

When using the GUI client on the same system the e-AS2 server is running on, no further action is required. However, if the GUI client is installed on a different computer on the network, the

---

<sup>23</sup>If necessary, the data fetch extension in e-AS2, however, allows for a system configuration based on computers which was not permanently online. Cf. Chapter 8 (*Data fetch extension*).

connection between it and the e-AS2 server must be enabled. The port number is requested for the configuration during installed for this purpose. If you keep the default, it is port 5070.

So the firewall configuration must allow incoming connections to the e-AS2 server from your company network via port 5070 (or the port specified during setup).

### 5.1.3 Outgoing connections

Firewalls often only implement limitations with respect to incoming connections, but allow all outgoing connections. If your company’s firewall model also limits outgoing connections, then outgoing connections to your provider’s computer centre must be enabled in the firewall for the ports used by AS2.<sup>24</sup> After installing e-AS2 Connect you will find the preconfigured partner profile for accessing your provider. Here you will see the respective parameters.

### 5.1.4 Summary

The following table summarises the required firewall configurations. This is based on using the defaults during the installation process. If these were changed, please use your custom values! Wherever the table states “e-AS2 server”, it refers to the computer at your company where e-AS2 is installed.

Port number	from	to	Purpose
5080	Internet (your partners)	e-AS2 server	Data communication via HTTP
5070	Company network (LAN or VPN connections)	e-AS2 server	Configuring the e-AS2 entity
5080	e-AS2 server	Internet (your provider)	Data communication, production
5088	e-AS2 server	Internet (your provider)	Data communication, test <sup>25</sup>

## 5.2 Using proxy servers

All aspects related to security when using e-AS2 can reliably be handled with the appropriate firewall configuration. However, companies often have an existing infrastructure with HTTP proxy set up for accessing the Internet with a web browser. In this case, it’s desirable to also use the existing HTTP proxy for AS2 communication, which after all is also HTTP based. Just as with the firewall configuration, the two directions for communication should be addressed separately.

The following sections are primarily aimed at the firewall administrator. Using proxy servers requires a special configuration in e-AS2. This is addressed in Section 10.2.2 (*Parameters for proxy use*).

### 5.2.1 Incoming connections

Incoming connections require a reverse proxy configuration. Configure your proxy server to accept HTTP connections for the Internet and forward them to the computer running e-AS2. In addition to port forwarding, this may also require URL rewrite to address the e-AS2 server

<sup>24</sup>In the case of e-integration these are ports 5080 and 5088 at `as2.edicc.de`. Port 5080 is used to access the production server, 5088 the test server for e-integration.

<sup>25</sup>Not every provider allows test access.

appropriately. However, for simplicity and clarity we recommend offering the outgoing AS2 service under the same port and the same post request URI used by the e-AS2 server itself.

### 5.2.2 Outgoing connections

Outgoing, the proxy server must accept HTTP connections from the e-AS2 server and forward them to the Internet accordingly. This is ultimately the typical proxy configuration which is also used to access the Internet with browsers. However, a few details should be considered.

The proxy server is often also used as a cache for outgoing HTTP connections. This aspect has no effect for AS2, as it does not access any sites which would be used again. In place of the typical HTTP GET for browsing, with AS2 an HTTP POST is set to transmit the payload to the opposite party.

In addition, you will need to assume AS2 does not address port 80 HTTP connections typically use . So the respective restrictions must be withdrawn for AS2 connections.

### 5.2.3 Authentication

Access to the HTTP proxy is often restricted by requesting authentication. In these cases, please note e-AS2 supports the method “Basic Authentication” according to RFC 2617 (see <http://www.ietf.org/rfc/rfc2617.txt>). The method “Digest Authentication” is not implemented and therefore cannot be used with e-AS2.



## 6 Traditional logic configuration

*Configuring and using e-AS2 Connect - a step by step documentation based on example scenarios. This deals with the traditional e-AS2 Connect logic, which applies to older software versions (pre-version 8.3) and providers different from e-integration.*

It is assumed the firewall has already been configured as described in Chapter 5 (*Firewall and Proxy configuration*).

### 6.1 Routing during file transmission

If you read Chapter 2 (*Quick Start*), you have sent a test file to your provider. Your actual goal, however, is to exchange data with your business partner through your provider's computer centre. So when sending a file, you will need to notify your provider who to forward the respective file to. Conversely, when receiving files you will need information from whom the provider received the file which was forwarded to you. This matter of correct forwarding of files is referred to as routing.

e-AS2 Connect users have two options for controlling routing, routing via file content or routing via AS2 relation. AS2 relation refers to the pair (AS2From, AS2To). We will now examine these two options in more detail.

#### 6.1.1 Routing via file content

EDI syntaxes often have the option to encode routing information as part of the outgoing message. So e.g. EDIFACT interchanges contain a UNB segment where the relation behind the business transaction (from whom, to whom) is encoded. When using this type of EDI syntax, you can arrange with your provider for your outgoing files to be forwarded based on this routing information so to speak incorporated in the file.<sup>26</sup>

In this case, when sending files you will always use the same AS2 relation, describing the relationship between your company and your provider. You agree once on the two AS2 identifiers involved to define the relation, e.g. (myIdentifier, EDICC). The AS2 relation (myIdentifier, EDICC) will then be used for every transmission. Conversely, you also receive files via this AS2 relation and on your part ensure the files received are assigned correctly using the information in the contents of the file.

#### 6.1.2 Routing via AS2 relation

If the file syntax does not allow for control information for routing or you do not wish to use this option, you can use routing via AS2 relation. This option always works. When exchanging data with business associations you're then using relations (myIdentifier, partner1), (myIdentifier, partner2), (myIdentifier, partner3) etc., whereas the files will continue to only be exchanged with your provider's computer centre. There they will then be forwarded correctly based on the AS2 relation and conversely returns the associated relation to you for files received.

In the further course of this chapter we assume routing will use the AS2 relation. With respect to using e-AS2 Connect, routing based on file content is done in the exact same way although the partner identifier will always be EDICC.

We further assume the software has already been started as described in Chapter 4 (*The architecture of e-AS2*) and you have access to the configuration GUI. In addition to the GUI

---

<sup>26</sup>This of course requires your provider supports this. If you're running e-AS2 Connect in Germany, you will be connected to the EDI-CC by e-integration. They support routing via file content.

you will need to be able to access the file system of the computer where the e-AS2 server is running so you can prepare files for transmission or locate files received.

The description in the rest of this chapter is based on the assumption you're located in Germany and are connected to the e-integration EDI-CC. If you're working with a different provider, the different communication parameters will already be preset during installation. The configuration interface and how the software behaves do not in any way differ from this description.

## 6.2 Configuring the e-integration connection

Before you can send or receive data with e-AS2 Connect you will first need to configure the connection to the e-integration EDI-CC. Luckily, most of the parameters have already been correctly preset during installation.<sup>27</sup>

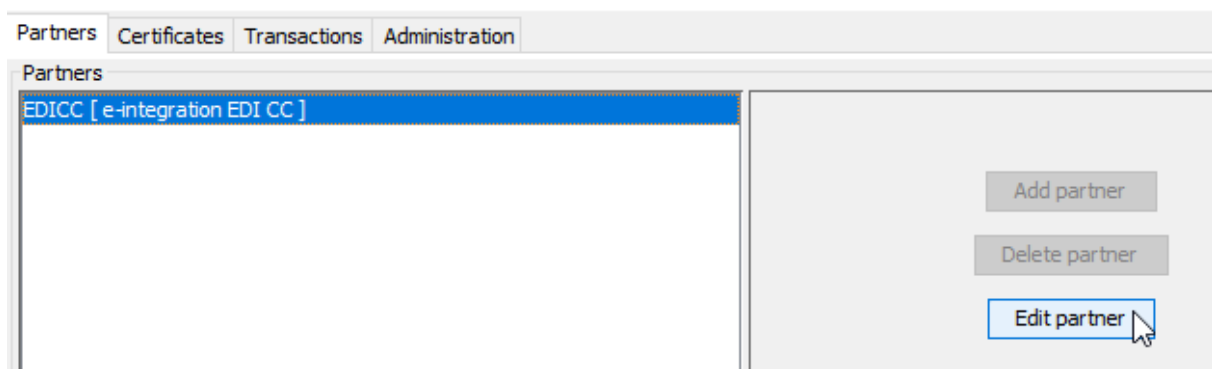


Figure 6.1. Partner list in e-AS2 Connect

Select the entry “EDICC” from the partner list and press the “Edit partner” button.<sup>28</sup>

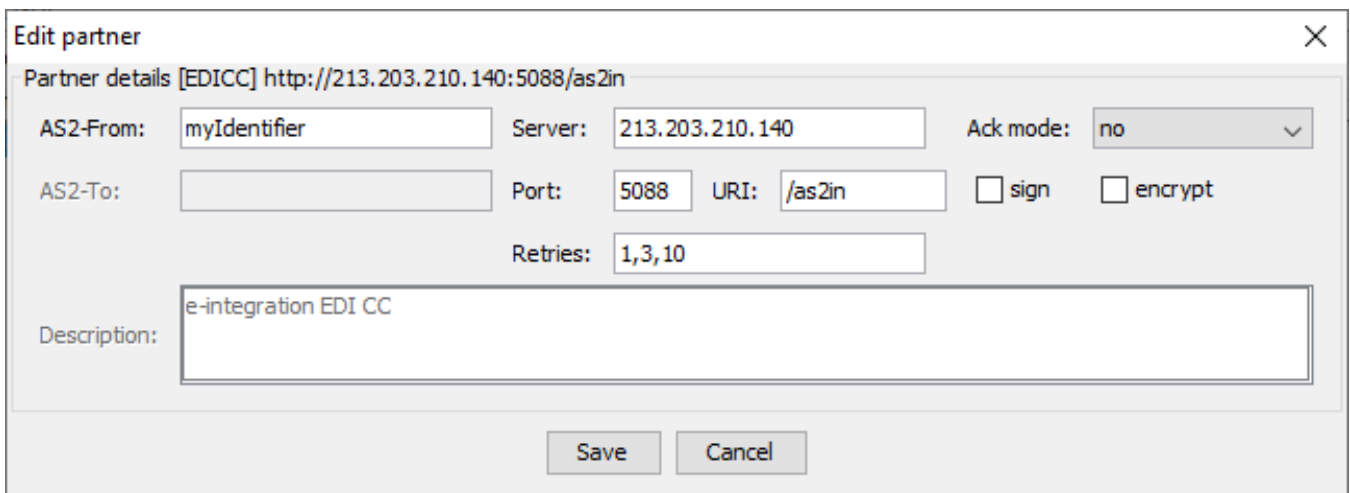


Figure 6.2. Editing partner data

<sup>27</sup>As of e-AS2 8.3 connection parameters are organized differently for provider e-integration. (You will find information on the new UI and the new processing logic in the next chapter.) However, for the time being UI and processing logic remain unchanged for all other providers.

<sup>28</sup>The buttons “Add partner” and “Delete partner” are always inactive in e-AS2 Connect, since you only need this one connection to your provider. At this point the full version of e-AS2 Enterprise has the option to manage any number of partner profiles.

Enter your personal identifier you as agreed upon with e-integration in the field “AS2-From”! Although for the initial test in Chapter 2 (*Quick Start*) we still allowed you to enter a personal identifier here, when preparing for production mode it’s important to use the exact identifier provided to you, or the e-integration EDI-CC will not be able to correctly assign your data and process it.

The defaults of the remaining parameters are initially okay. During the test phase you will be using the test server in the e-integration EDI-CC, behind Port 5088. Once all testing has been completed and your connection goes live, change the port number to 5080 to connect to the production server.<sup>29</sup>

You’ve noticed the field “AS2-To” is blocked. No entry can be made here. This is because the name in “AS2-To” is dynamically taken from the respective outgoing directory on a case by case basis. In the next section (sending files) you will learn how this works.

### 6.3 The interface area

Open Windows Explorer (or a different file manager of your choice) and locate the folder where e-AS2 Connect was installed. Here you will find a subfolder named `interface`. This is where e-AS2 manages incoming and outgoing files. From here on, we will often simply refer to this area as the interface area. There are additional folders within `interface` for the different tasks, e.g. the folder `in` for incoming files and `out` for outgoing files.

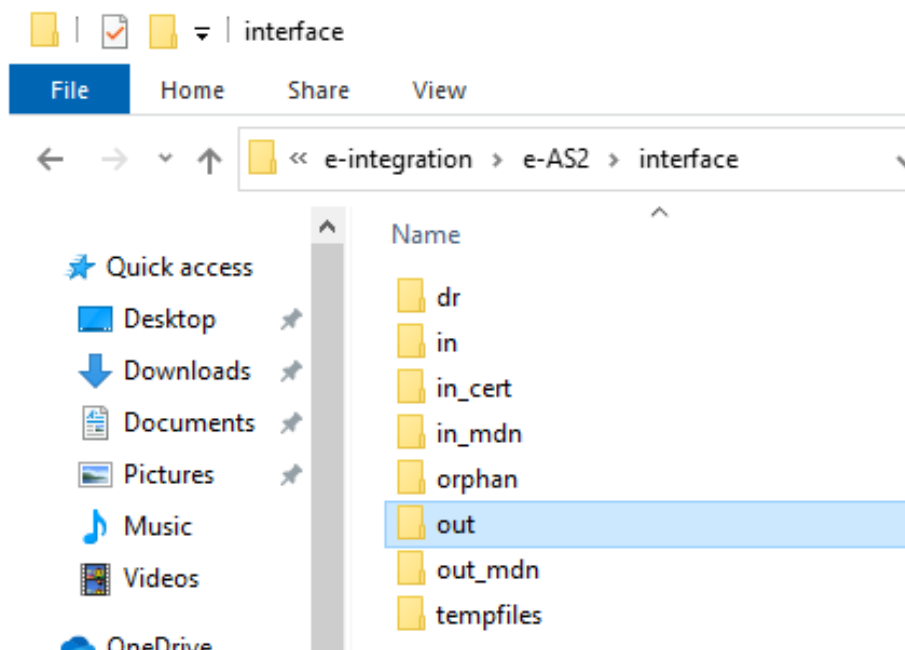


Figure 6.3. The interface area

### 6.4 Sending files

In the first application example we will skip the cryptography functions in e-AS2 Connect entirely. The goal is to send a file from your company (*myIdentifier*) to your business partner (*partner1*), using the e-integration EDI-CC for the transfer.

<sup>29</sup>If you’re connected to a different provider, the distinction between the test and production server may be different. Do not apply a configuration you have not arranged with your provider.

In the `out` folder of the interface area, add a new subfolder named `partner1`. This specifies you want to exchange data via the AS2 relation (`myIdentifier`, `partner1`). I.e. the value for AS2-To you're not able to enter when configuring the EDICC partner profile will be set via the name of this folder.

Please note, AS2 identifiers are case sensitive, select a directory name keeping this in mind!

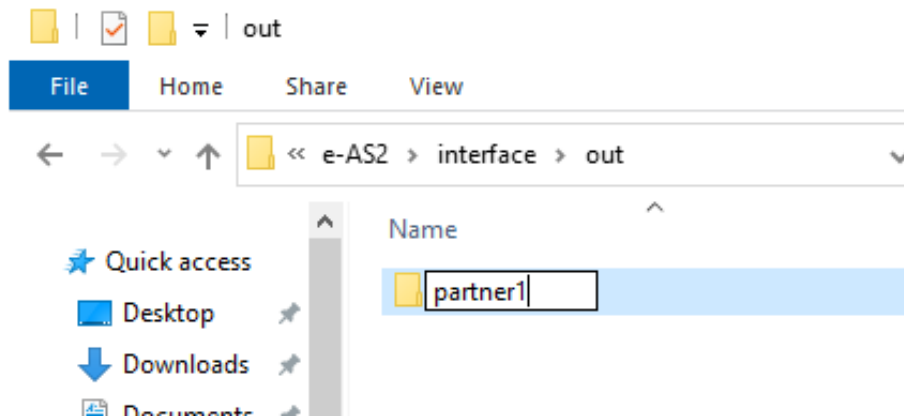


Figure 6.4. A new directory under `out`

Set up additional folders under `out` as needed, one folder for each of your communication partners! Important! The names of these folders must be coordinated with e-integration, since the e-integration EDI-CC will forward these files to the actual respective recipient based on the AS2-To identifiers this creates. In the e-integration EDI-CC this forwarding is explicitly linked to the respective identifier.

To now send a file to `partner1`, you will only need to copy the file to the folder `partner1`. Once the file has been sent, the file will automatically disappear from the directory. If you connect e-AS2 Connect to an enterprise resource planning system, you can configure it to automatically save outgoing files directly to the respective interface directory.

## 6.5 Receiving files

As long as the e-AS2 Server is running and can be reached over the internet via the configured data communication port, it is ready to receive files. No further actions are required for this. The process for receiving decrypted and unsigned files is for instance as follows.

### 6.5.1 Accepting a connection

If the sender reaches the e-AS2 server via IP address and port, the connection is always accepted at first. The reception of data as defined in the AS2 protocol also begins immediately.

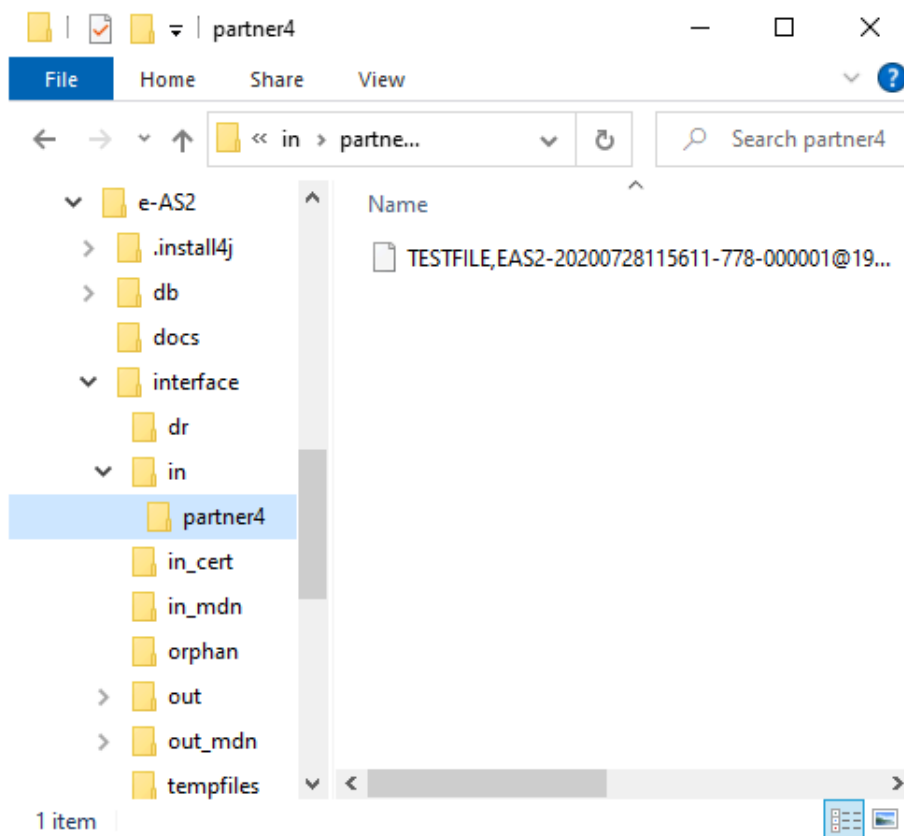
### 6.5.2 Comparing the URI

After reading the first line of the MIME message the server will pause again and examine the POST Request URI received. If the URI transmitted by the sender matches the locally configured URI expected for incoming connections,<sup>30</sup> the rest of the MIME message will be received. However, if the URI is not as expected, the server will immediately disconnect at this point to eliminate any further unnecessary data transfer.

<sup>30</sup>To configure the URI for incoming connections see Section 10.2.1 (*Communication parameters, HTTP*).

### 6.5.3 Receiving data, assignment and acknowledgement

Following successful URI comparison the complete MIME message will be received and the key parameters subsequently extracted from the MIME header. What's important for further processing is mainly the AS2 relation (AS2From, AS2To). When receiving files, e-AS2 Connect expects them to be addressed to its own AS2 identifier, i.e. AS2To must match the identifier specified as the own identifier in the partner configuration<sup>31</sup>, so for this example *myIdentifier*. The sender identifier (AS2From) can be any.



**Figure 6.5. A file has been received**

If AS2To is correct, the save folder will be identified in the `in` folder based on the sender identifier (AS2From) and the file saved to it. In the above example a file was received from the partner *partner4* and saved to this folder accordingly. If the folder does not exist at the time the file is received, it will automatically be created.

The file received will be saved under a local file name based on the following pattern:

#### Local name of received files

[Name],[Message ID]

So the local file name is built of the name transmitted by the sender (in this example: TEST-FILE) and the message ID generated by the sender for this file transfer (in this exam-

<sup>31</sup>Please note, the own identifier can be found in field AS2-From in the partner configuration. This may seem confusing at first, but is coherent, since in a reversed transmission the roles of the two communicating parties are reversed. The configuration of the partner profile is conceptually based on the send process. So during configuration the own identifier is entered under AS2-From. When receiving data, however, the own identifier (from the viewpoint of the sender) is the identifier the data is being sent to, so AS2To. As a rule, remember the sentence "when receiving, AS2From and AS2To are reversed"!

ple: EAS2-20200728115611-778-000001@19...), separated by a comma. The file received will be saved to the partner folder (in this example: `partner4`) under this local name. Including the message ID in the newly created local file name will prevent potential conflicts due to the sender assigning identical file names in multiple transfers.<sup>32</sup>

The file is now available for further processing. e-AS2 Connect will generate an acknowledgement which is returned to the sender once the file has been successfully saved to the file system. So the sender will receive information whether the file was received correctly. The acknowledgement does not provide any information on the data being processed further via a subsequent EDI or ERP system.

If AS2To is incorrect, the file received will be considered a stray message and not saved to a partner `in` folder. These files will instead be saved to the `orphan` folder.

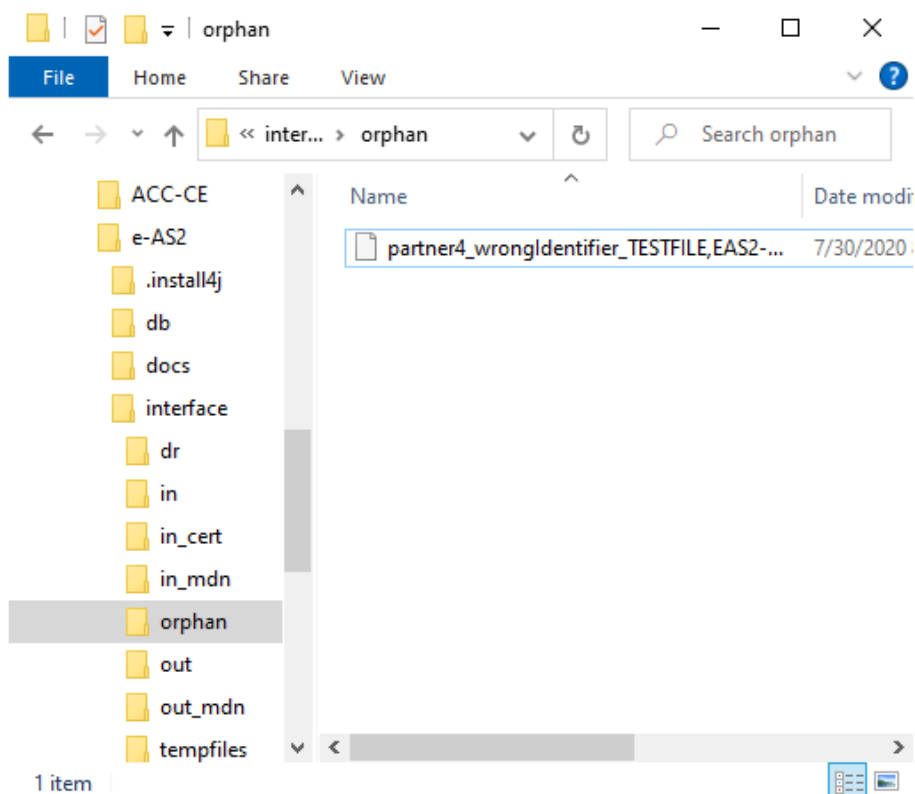


Figure 6.6. A file was addressed incorrectly

In this case the local file name will be configured as follows:

**Local name for incorrectly addressed files**

[AS2From]\_[AS2To]\_[Name],[Message ID]

I.e. the file name indicates which AS2 relation was addressed by the sender of the file so the configuration can be modified accordingly, if necessary, or the sender can be informed specifically which parameters he needs to modify. After taking appropriate measures you simply remove the file from the `orphan` directory.

<sup>32</sup>You can control the rules for how the local file name is structured via the configurations in the file `EAS2.properties`. You can read about how to do this in Section 10.2.14 (*Building file names for incoming messages*).

## 6.6 Preparing for encryption and signature

In the last sections we saw only one parameter, namely the own AS2 identifier, needs to be added to the partner configuration to be able to start exchanging data using e-AS2 Connect. If you're not exchanging sensitive data which needs to be protected in a special way, you can skip the following sections addressing encryption and signature. Continue reading under Section 6.13 (*Managing transactions, error control*)!

Using the cryptography functions in e-AS2 Connect requires a little bit of preparation. The cryptography in e-AS2 is based on so-called certificates. Think of it as a person's or a software entity's digital identity. You will need two certificates describing the two identities involved, yours and that of the e-integration EDI-CC.

Alias	CN	Priv.
<u>_eas2_</u>	unknown	<input checked="" type="checkbox"/>
e-integration	e-integration CC AS2 Server	<input type="checkbox"/>

Figure 6.7. Certificate list

In the GUI, first go to the “Certificates” tab! Here you will already see the two required certificates preinstalled. The e-integration certificate is located under the respective alias in the list. Behind the alias \_eas2\_ you will find a placeholder for your own certificate. Select an entry in the list to see the certificate details in the lower part of the GUI.

Subject:	CN=e-integration AS2 Server, O=EDICC, C=DE, EMAILADDRESS=support@e-integration.de		
Issuer:	CN=e-integration AS2 Server, O=EDICC, C=DE, EMAILADDRESS=support@e-integration.de		
Serial no.:	1470384986 (0x57a44b5a)		
Valid from:	Aug 5, 2016 10:16:26 AM	Valid to:	Aug 3, 2026 10:16:26 AM

Figure 6.8. e-integration certificate

The e-integration certificate is valid until 2026. It's easy to see it defines the identity of the e-integration AS2 server.

Subject:	CN=unknown, O=unknown, C=unknown, EMAILADDRESS=unkown		
Issuer:	CN=unknown, O=unknown, C=unknown, EMAILADDRESS=unkown		
Serial no.:	1143126850072 (0x10a27ab7a18)		
Valid from:	Jan 1, 2005 4:12:57 PM	Valid to:	Dec 31, 2006 4:12:57 PM

Figure 6.9. Certificate placeholder for e-AS2 Connect

The placeholder for your own certificate expires the end of 2006, so as you are reading this, it has already expired. But this does not matter. You will now create your own certificate.

Press the button “Generate private key”. This will open a dialogue where you can enter your certificate details.

The screenshot shows a dialog box titled "Generate certificate" with a close button (X) in the top right corner. The dialog is divided into a "Certificate details" section and a bottom section with buttons. The "Certificate details" section includes the following fields:

- Common name: John Doe
- Organization: My Company Ltd.
- Country: US (Please use ISO two letter country code.)
- eMail: john.doe@companyltd.com
- Signature algorithm: SHA256withRSA (dropdown menu)
- Key length: 2048 (dropdown menu)
- Valid from: Jul 30, 2020 (calendar icon)
- Valid to: Jul 30, 2030 (calendar icon)

At the bottom of the dialog, there are two buttons: "Save" and "Cancel".

Figure 6.10. Generate new certificate

In principle, you are completely free to choose the data to be entered in this form. However, the certificate should clearly identify you. Look at the picture above as an example. The designated certificate term is 10 years. We recommend keeping this.

Press “Save” to confirm after having completed your entry. This will generate a new cryptography certificate for you with the specified parameters to replace the preinstalled (already expired) certificate. This process may take a while depending on the key length selected and the performance of the computer you are using.

Once the process has completed the dialogue will disappear and you will see the certificate list again. Select the entry `_eas2_` again and review the details shown.

The screenshot shows a window titled "Certificate details [\_eas2\_]". It displays the following information:

- Subject: EMAILADDRESS=john.doe@companyltd.com, C=US, O=My Company Ltd., CN=John Doe
- Issuer: EMAILADDRESS=john.doe@companyltd.com, C=US, O=My Company Ltd., CN=John Doe
- Serial no.: 1596100859189 (0x1739f065535)
- Valid from: Jul 30, 2020 11:18:27 AM
- Valid to: Jul 30, 2030 11:18:27 AM

Figure 6.11. View certificate details

If the certificate generated adequately describes your identity (or rather: the identity of your AS2 system) it will need to be transmitted to e-integration to complete the process. To do so, go to the “Partner” tab, activate – if not already done – the entry EDICC and press the “Send Certificate” button. The following message will appear if this action completed successfully.

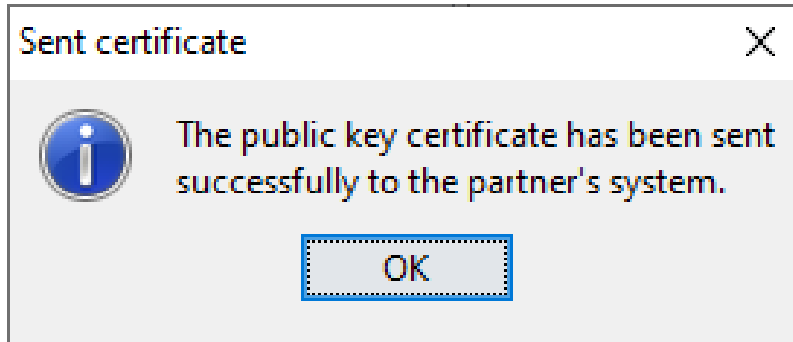


Figure 6.12. Send Certificate

Please note, the certificate can only be transmitted if you are able to send data (as described in Section 6.4 (*Sending files*)). If you have not yet been able to complete a test transmission, an error message will probably appear when trying to send the certificate.

## 6.7 Sending encrypted files

Before continuing, you will first need to prepare encryption and signature as described in Section 6.6 (*Preparing for encryption and signature*). If you have not yet done so, please do so now!

You can decide to transmit your data encrypted if you do not want them to be transmitted over the internet in clear text. Review the partners details which you configured for the connection with the e-integration EDI-CC!

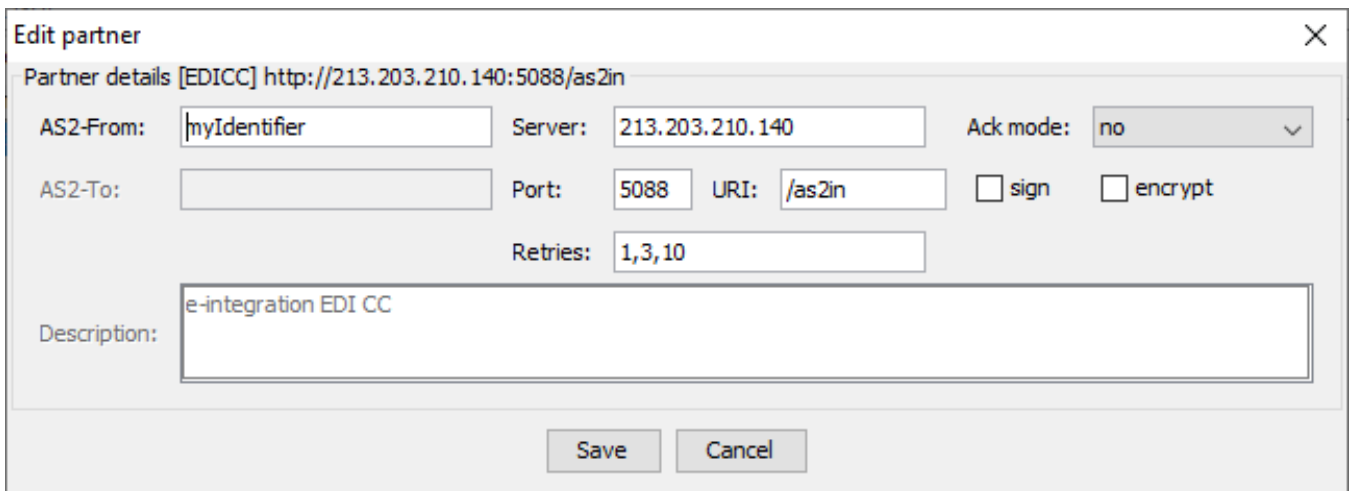


Figure 6.13. Configuration without encryption

You will see two check boxes, “sign” and “encrypt”, which are currently both unchecked. Check “encrypt” to enable encryption.

**Edit partner** [X]

Partner details [EDICC] http://213.203.210.140:5088/as2in

AS2-From:  Server:  Ack mode:

AS2-To:  Port:  URI:   sign  encrypt

Retries:

Description:

**Figure 6.14. Enable encryption**

That's all you need to do. To send data, proceed as described in Section 6.4 (*Sending files*). The process does not differ from decrypted transmission. From now on, any data you send will automatically be encrypted. You can disable encryption at any time again by unticking "encrypt".

## 6.8 Receiving encrypted files

In AS2, the sender of the data always determines whether files are encrypted. As the receiver you do not control this. e-AS2 Connect is always ready to receive encrypted data. However, there is no guarantee you will be able to decrypt the data received. If the sender did not use the correct key for encryption, an error will occur during decryption. In this case a new incoming transaction will appear in the transactions list with the check box "Error" ticked.

AS2 name:  Mess:

Created:  MIC:

Changed:  Algor:

Retri:

Description:

**Figure 6.15. Error decrypting the data received**

For transactions with error status the cause for the error can be found in the field "Description". If the text starts with `LOCAL :`, the error occurred during processing by your local system. In the example the reception of the data stream was okay, but an error occurred in the local system during the decryption attempt. If decryption fails, the AS2 name will always be `smime.p7m`.

If the data received was successfully decrypted, the process will not differ from receiving decrypted data as described in Section 6.5 (*Receiving files*).

## 6.9 Sending signed files

You can decide to send your data signed if you wish to ensure the integrity of the data and further identify yourself as the originator of the data. To do so, open the edit dialogue for the partner profile and check "sign"!

The screenshot shows a dialog box titled "Edit partner" with a close button (X) in the top right corner. The title bar also includes "Partner details [EDICC] http://213.203.210.140:5088/as2in". The dialog contains several input fields and checkboxes:

- AS2-From:** myIdentifier
- Server:** 213.203.210.140
- Ack mode:** no (dropdown menu)
- AS2-To:** (empty field)
- Port:** 5088
- URI:** /as2in
- sign** (checkbox with a mouse cursor over it)
- encrypt**
- Retries:** 1,3,10
- Description:** e-integration EDI CC

At the bottom of the dialog are two buttons: "Save" and "Cancel".

**Figure 6.16. Enabling signature**

From now on, all of your files will automatically be signed. Apart from that, the process for sending files the same as described in Section 6.4 (*Sending files*).

## 6.10 Receiving signed files

Whether or not to sign the data - just as for encryption - is always determined by the sender. e-AS2 Connect can easily receive signed files and in the process automatically validates the signature. If signature validation is successful, the further course is exactly as described in Section 6.5 (*Receiving files*). Otherwise the incoming transaction will be set to error status in the transaction list and an error message will appear in the "Description" field.

## 6.11 Requesting acknowledgements

During configuration you probably noticed the field Ack Mode. Here you can specify whether to request acknowledgements for the files you send. You can choose from two settings.<sup>33</sup>

- |                  |   |
|------------------|---|
| no               | You will receive no MDN (Message Disposition Notification) for data transmitted by you. The other party will only send a brief HTTP based acknowledgement of receipt, which only means the entire data has been received by the other end. You will not know if an error occurs with further processing.  |
| syn-<br>chronous | This request the other party to send an acknowledgement of receipt in form of MDN. The MDN allows the receiving AS2 server to return detailed information on whether processing was successful to the sender. An MDN is laid out and treated the same as a fully-fledged MIME message. This particularly means it is saved to the file system after receipt. In addition, e-AS2 Connect will analyse the content of the MDN and write any error messages to the transaction list in form of a status change. This provides you with a lot more information on the process than without MDN. |

We will now look at the difference between HTTP acknowledgement and acknowledgement via MDN using the following example. If you read Section 6.4 (*Sending files*), you have already sent a file addressed to *partner1* via the e-integration EDI-CC. Search for the associated transaction in the transaction list and review the transaction details!

<sup>33</sup>This applies for e-AS2 Connect when processing data in the outgoing direction. When receiving data, the sender may request a third form of acknowledgement, the "asynchronous MDN". e-AS2 Connect supports this and will generate the requested acknowledgement.

Transaction details [16]			
Partner ID:	<input type="text" value="partner1"/>	Server:	<input type="text" value="192.168.200.122"/>
Status:	<input type="text" value="acknowledged"/>		
AS2-From:	<input type="text" value="myIdentifier"/>	Port:	<input type="text" value="4080"/>
URI:	<input type="text" value="/comm/as2"/>		Direction:
<input type="text" value="S"/>			<input type="checkbox"/> In Process <input checked="" type="checkbox"/> Done
AS2-To:	<input type="text" value="partner1"/>		

**Figure 6.17. Transaction details without MDN request**

The transaction is Done and the status is reported as acknowledged. That means you received an HTTP acknowledgement for this transaction. Now change the Ack Mode for partner EDICC to synchronous. Then send another file and review the associated transaction! (Don't forget to use the "Re-read list" button to show new transactions!)

Transaction details [17]			
Partner ID:	<input type="text" value="partner1"/>	Server:	<input type="text" value="192.168.200.122"/>
Status:	<input type="text" value="received MDN"/>		
AS2-From:	<input type="text" value="myIdentifier"/>	Port:	<input type="text" value="4080"/>
URI:	<input type="text" value="/comm/as2"/>		Direction:
<input type="text" value="S"/>			<input type="checkbox"/> In Process <input checked="" type="checkbox"/> Done
AS2-To:	<input type="text" value="partner1"/>		

**Figure 6.18. Transaction details with MDN request**

Once processing has completed, the transaction is again marked as Done, but the status now shows received MDN. If you look at the transaction details further down, you will probably see this transaction was marked as erroneous.

AS2 name:	<input type="text" value="test.txt.txt"/>	Message ID:	<input type="text" value="EAS2-20200730122341-595-0"/>	Size:	<input type="text" value="41"/>
Created:	<input type="text" value="2020-07-30 12:23:41"/>	MIC:	<input type="text" value="sHu6lO36lMUkRHxN9cNt85kbI"/>	<input checked="" type="checkbox"/> Error	<input type="checkbox"/> Sign MDN
Changed:	<input type="text" value="2020-07-30 12:23:41"/>	Algorithm:	<input type="text" value="SHA1"/>	<input type="checkbox"/> MDN error	<input type="checkbox"/> MDN done
		Retries:	<input type="text" value="0 (1,3,10)"/>		
Description:	<input type="text" value="REMOTE: error: relation-not-found"/>				

**Figure 6.19. Error message in transaction details**

The error message listed is "REMOTE: error: relation-not-found". The prefix REMOTE: indicates a processing error occurred on the other end, so the system of the party you are communicating with (e-integration EDI-CC). Although in this case the transfer of data worked, the process could not be completely processed at the e-integration EDI-CC since it did not have your relation to partner1 configured yet.

## 6.12 Setting a retry pattern

In communication applications there is no absolute guarantee individual data transfers won't fail. If e.g. your internet connection is down temporarily, the e-integration EDI-CC is temporarily unavailable. Or a configuration error (incorrect port number, etc.) could cause the connection to e-integration not to be established as intended. e-AS2 Connect features a powerful mechanism to handle these cases.

Normally, network outages impairing communication are temporary. I.e. it's worth trying again shortly after a connection fails. e-AS2 Connect will handle these additional attempts automatically when configured accordingly. To do so, use the field "Retries" in the partner profile. The default for this field is 1, 3, 10. What does that mean?

Regardless of the retry pattern in this field, e-AS2 Connect will first attempt to immediately transmit files picked up from the interface area. If the attempt is successful, the retry pattern doesn't matter. However, if this initial communication attempt fails, e-AS2 Connect will first leave the file in place and try again after one minute (number 1 in the retry pattern 1,3,10). If this also fails, the next attempt will follow after another 3 minutes. Finally, another attempt to transmit the file will be made after 10 minutes. If this fails again, the transaction will be set to error status and no further transmission attempts will be made.

You can flexibly change the retry pattern to suit your needs and as desired by stringing together any number of pauses (measured in minutes) separated by commas. If you want e-AS2 Connect to attempt to transmit the file infinitely, enter an asterisk as the final character in the retry pattern. This will result in the last pause specified to continuously be repeated. For example, the retry pattern 2,10,60,\* will result the attempt to be repeated after 2, after 10 more, after 60 more, and then infinitely every 60 minutes.

The opposite pole to "infinite attempts" would be "stop immediately". You can also formulate this version if desired. Simply leave the field `Retries` blank. This will result in the transaction immediately being set to error status if the first attempt fails. No further retries will be made.

## 6.13 Managing transactions, error control

The primary source of information for all file transfers managed by e-AS2 Connect, both for send and receive, is the transaction list. Any file sent is reflected with an entry in the transaction list; just as each file received. In both cases the transaction list provides information about the success of the transaction and particularly allows open or error transaction to quickly be detected.

### 6.13.1 Transaction list, quick overview

The list view in the e-AS2 GUI provides a quick overview of all transactions for the current day. If you're looking for older transactions, you can select a different date from the combo box at the top right in the GUI. Transaction entries for sent files begin with [S]. Entries for received files begin with [R].

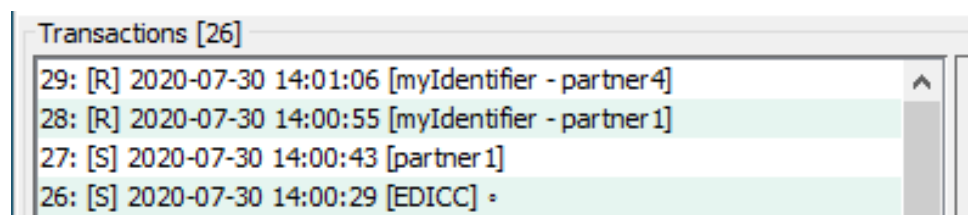


Figure 6.20. Transaction list

In both cases, this is followed by the time when the respective event occurred, to the second, in the format YYYY MM DD hh:mm:ss. For send processes, at the end is the partner identifier according to the folder name added under `out` to identify this partner. For receive processes, the AS2 relation from the MIME message received will appear.

Transactions are sorted backwards by time. So, the youngest transaction is always to be found at the top of the list.

**Important!** When starting the graphical user interface, the transaction list will first reflect the current processing status. However, this is not automatically updated later. If you later want to access the latest processing status with the GUI running, press the "Re-read list" button to show new and changed transactions. For efficiency reasons this only updates the portion of the list visible in the section displayed. In addition, all new transactions will be downloaded and

internally added to the top of the list. To reload the complete transaction list for the selected date, including those transactions that are neither new nor currently visible, tick the check box “all”, before pressing the button.

The number of the transactions loaded to the GUI is always shown in square brackets to the top left of the list, in the header frame. So in the example in Figure 6.20, “Transaction list” a total of 26 transactions were loaded.

### 6.13.2 Transaction list, colour coding

Entries in the transaction list are colour coded as a quick guide.

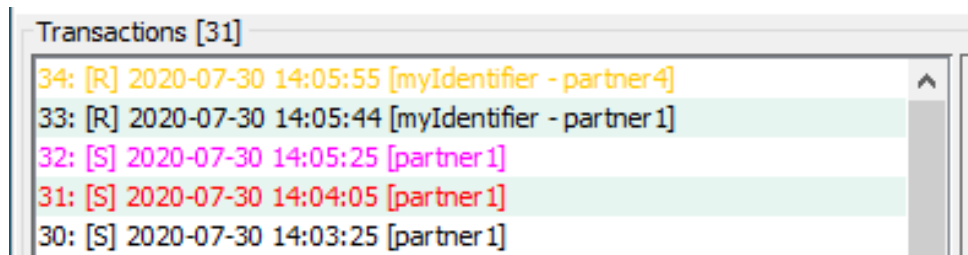


Figure 6.21. Transaction list, colour coded entries

The four possible colours mean:

- Entries in *black* were completed successfully and do not require further action. For send transactions this means the file was transmitted to the remote end and a positive confirmation was received. For receive transactions this means the complete file was received and assigned successfully. The file received can be found in the respective partner folder under the `in` folder.
- Entries in *red* are definitely erroneous. I.e. transmission of the outgoing file failed and no repeated attempts are pending. So without manual intervention, this file will no longer be sent. However, the transaction can be released manually, see Section 6.13.3 (*Releasing for re-transmission*). This situation can only occur for send jobs.
- Entries in *magenta* are currently processing. This can mean the file was only recently picked up from the `out` folder and transmission is currently actually underway. However, this can also mean one or several communication attempts have failed, but based on the retry pattern configured, additional attempts are pending. The transaction will only be set to definitely erroneous after all attempts have failed and the colour will then be changed to *red*. Or, if a retry is eventually successful, the colour will be changed to *black*. The colour *magenta*, just as *red*, is only used for send jobs.
- Entries in *yellow* are related to *red* entries. *Yellow* transactions are also in error status. Unlike *red* transactions, the file has already been transferred. I.e. the error status only occurred or was reported following successful transfer of the respective file. In case of send transactions this indicates the remote end returned an error in the MDN. This means the file was received by the remote end, but it was unable to process it.

From an e-AS2 perspective, the corresponding applies to incoming transactions. e-AS2 was able to receive, but not to process the file. If the sender of the file requested an MDN, e-AS2 reported this situation to the sender. A *yellow* transaction practically always requires contacting the other party to clarify the situation and correct the cause for the problem, if required. You will also need to clarify if the respective file needs to be resent or if the receiver will manually forward the file, which was already received complete, to further processing.

### 6.13.3 Releasing for re-transmission

Once a transaction becomes a definitely erroneous entry, red in the list, the associated file will be removed from the partner directory under `out`. If it doesn't already exist, a new subdirectory named `_errors_` will automatically be created in the partner directory. The file which was not transferred will be moved to it.

In the example, transmission of the file intended for `partner1`, `test.txt`, failed. Following all retries it was moved to the folder `_errors_`. (see Figure 6.22, "A file in the error directory")

You will now take measures to correct the error. For example, you will restore the Internet connection. Or ensure the firewall configuration is modified. You then want to repeat the failed transaction. To do so, select the respective entry in the transaction list in the GUI and press the "Release transaction" button.

This will first change this transaction status to "released". After a while it will then be picked up for processing again, whilst the file to be transmitted will automatically be moved from the `_errors_` folder to the partner folder. This should be repeated for all red entries until none remain.

**Note:** For systems largely running without user interaction where the transaction list is rarely checked, it's important to configure the retry pattern so the transactions are not set to definitely erroneous too soon. Using retry patterns with infinite repeat can quite generally prevent this status. Refer to Section 6.12 (*Setting a retry pattern*)!

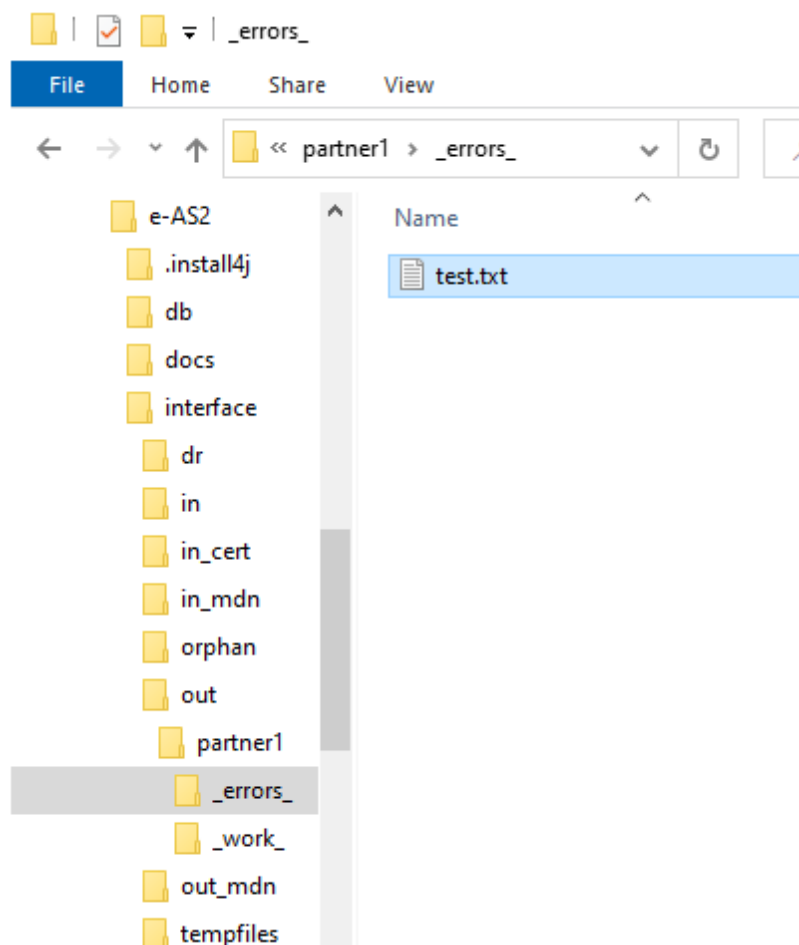


Figure 6.22. A file in the error directory

### 6.13.4 Stopping transactions

When using a respective retry pattern for infinite repeat, you may actually want to stop additional attempts for transactions being processed.

If this is the case, you can end processing for the respective transaction prematurely by pressing the “Stop transaction” button. A stopped transaction appears yellow in the transaction list.

In principle, any transaction which has not yet been processed completely, can be stopped prematurely using this method. If you select an entry in the transaction list which is candidate for being stopped, the button “Stop transaction” will automatically become active. Otherwise it remains inactive and cannot be clicked.

### 6.13.5 Release MDN

If transmission of an asynchronous MDN is pending for an incoming transaction, communication problems may occur just the same as when sending user data. In this case, the tag {MD-NERR} (incl. curly brackets) will be added to the end of the row for this entry in the transaction list. This will trigger the retry pattern in the partner profile. In the worst case, the transfer of the MDN can fail permanently and also still have failed after all retries. You can tell by the check boxes “MDN Error” and “MDN done” both being ticked in the transaction details.

Selecting this type of transaction will activate the “Release MDN” button. This allows you – similar to the process of transmitting user data – to release the MDN to again be transmitted at a later time.

Alternatively, in this case you can also stop the transaction, which will discard transmitting the MDN. (There is no separate button for stopping MDN transmission. You can access it from the “Stop Transaction” button.)

### 6.13.6 Transaction details

At this point we will take a quick look at some important information in the transaction details. You will find a systematic explanation of all fields under Section 11.5 (*Transaction details*).

Transaction details [35]

Partner ID:	<input type="text" value="partner 1"/>	Server:	<input type="text" value="213.203.210.140"/>	Status:	<input type="text" value="acknowledged"/>
AS2-From:	<input type="text" value="myIdentifier"/>	Port:	<input type="text" value="5088"/>	URI:	<input type="text" value="/as2in"/>
AS2-To:	<input type="text" value="partner 1"/>			<input type="checkbox"/> In Process	<input checked="" type="checkbox"/> Done
Filename:	<input type="text" value="C:/E-INTE~1/e-AS2/interface/out/partner 1/testfile.txt.txt"/>				
AS2 name:	<input type="text" value="testfile.txt.txt"/>	Message ID:	<input type="text" value="EAS2-20200730152706-187-0"/>	Size:	<input type="text" value="41"/>
Created:	<input type="text" value="2020-07-30 15:27:06"/>	MIC:	<input type="text" value="sHu6lO36IMUkRHxn9cNt85kbI"/>	<input type="checkbox"/> Error	<input type="checkbox"/> Sign MDN
Changed:	<input type="text" value="2020-07-30 15:27:06"/>	Algorithm:	<input type="text" value="SHA1"/>	<input type="checkbox"/> MDN error	<input type="checkbox"/> MDN done
		Retries:	<input type="text" value="0 (100)"/>		
Description:	<input type="text"/>				

Figure 6.23. Transaction details

You should first always look at the check box “Done” in the transaction details. If you see a check mark there, you know that e-AS2 Connect will no longer do anything with this transaction of its own accord. If “Error” is also ticked, this behaviour is quite okay.

## Traditional logic configuration

Transactions where “Error” as well as “Done” are ticked require manual handling. These are ones we earlier referred to as “definitely erroneous”. Transactions marked erroneous but not yet done can be referred to as “temporary erroneous”. Here the software will make additional attempts to complete the transaction without user intervention.

The two check boxes “MDN done” and “MDN error”, which implement the same logic but in reference to MDN transmission, should be treated the same way.

The field “Status” provides information on the current processing status of this transaction. The following values are possible in normal mode:

new	This is the initial status of every transaction immediately after the file is being detected in the file system (for send processes) or immediately after the transfer of an incoming file begins (for receive processes).
sent	Send processes will change to this status as soon as the data transfer is completed.
received	Receive processes will change to this status once the data transfer is completed.
acknowledged	Once an HTTP acknowledgement has been sent or received, the transaction will change to this status.
MDN received	Send processes will have this status once the remote endpoint has returned an MDN.
MDN pending	Receive processes will have this status if the opposite party requested an asynchronous MDN.
MDN sent	Receive processes will have this status once an incoming file was confirmed with an MDN.

There are various possible sequences for the various status values depending on the parameter settings for the transaction, as shown in the table below.

Parameter	Status sequence
Sending without MDN	new – sent – acknowledged
Sending with synchronous MDN	new – sent – MDN received
Receive without MDN	new – received – confirmed
Receive with synchronous MDN	new – received – MDN sent
Receive with asynchronous MDN	new – received – confirmed – MDN pending – MDN sent

In addition to these normal status values, the following additional statuses may occur in special cases:

released	After a transaction with error status has been released for retransmission, it will first change to this status. The software will then treat it the same as a new transaction and the status changes described above follow.
stopped	If a transaction which is not yet completed was stopped prematurely it changes to this status and keeps it.

## Traditional logic configuration

**MDN released**            Once a transaction where the MDN has not yet been sent is released for reprocessing the MDN, it will first change to this status. This is followed by the status “MDN pending” or “MDN sent” depending on the success of further processing.

The field “Retries” always shows how many retries have been made followed by the configured retry pattern in brackets. Along with the time of the last change, shown at the left, this can vaguely be used to predict when the next attempt will be made.

The field “Description” contains error messages provided the transaction has an error status. For send processes with MDN this can certainly be messages from the remote end. You can distinguish messages from your system from messages from the remote end by the prefix (LOCAL: or REMOTE:).

## 7 Multi-provider logic configuration

*Configuring and using e-AS2 Connect - this deals with the multi-provider logic available in more recent versions of e-AS2 Connect (version 8.3 or later) for connections to e-integration.*

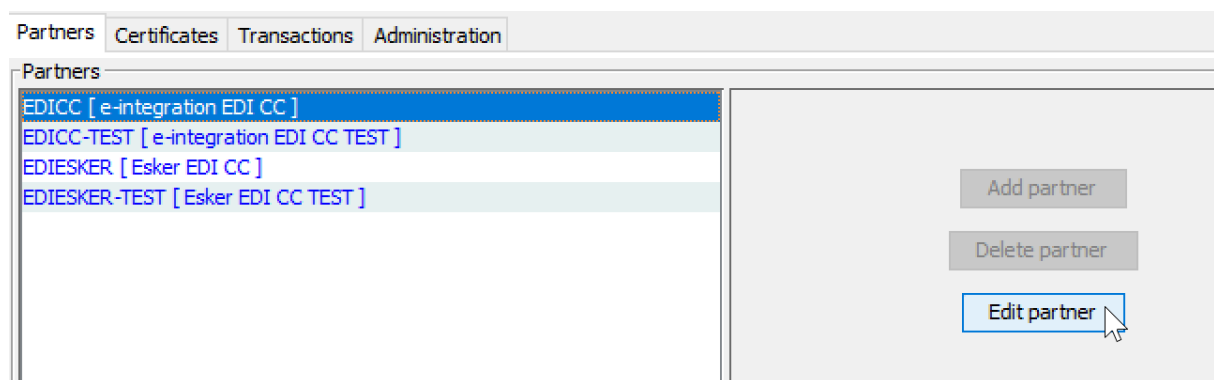
As of version 8.3 configuration is done differently, when connecting to e-integration. This takes into account, that there are now four end points in total, that you can connect to. These are one test connection and one production connection, each at two different locations. Manually re-configuring connection parameters as described above becomes tedious and error prone in this situation. Therefore, e-AS2 Connect now comes with four pre-configured partner profiles pointing to the four different end points in question. The software now acts in a way that you are not connected to one single provider anymore, but to four different providers. We call the new mode of operation multi-provider logic. This allows for seamless switching between test and production connections and also supports parallel use of the two data center locations.

Before continuing please completely read and understand the previous chapter. Most of the configuration details remain unchanged and we will not repeat them here. In this chapter we will just highlight the differences between regular logic and multi-provider logic in e-AS2 Connect.

There are two flavors of multi-provider logic to choose from. The first one keeps the ability to perform routing on the provider's systems based on AS2 IDs. We call this "full multi-provider mode". The second does not support routing based on AS2 IDs. Only content based routing will be possible on the provider's systems. We call this "restricted multi-provider mode". The subsequent documentation starts with full multi-provider mode. Documentation on restricted multi-provider mode follows.

### 7.1 Configuring the e-integration connection

Before you can send or receive data with e-AS2 Connect you will first need to perform minimal configuration of the connections to the e-integration EDI data center locations. The core communication parameters have already been preset during installation.



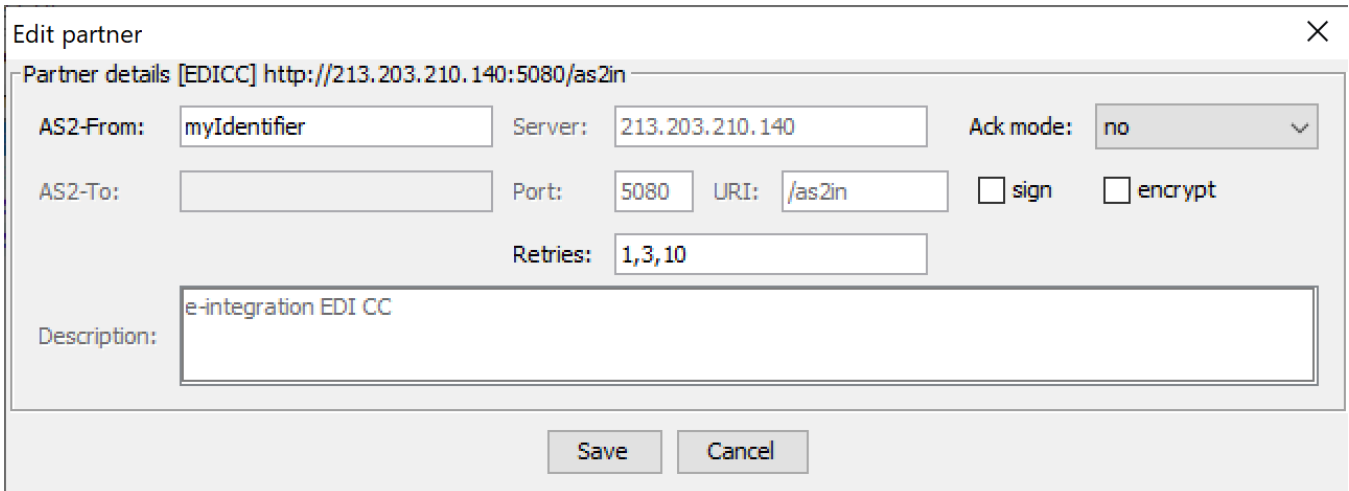
**Figure 7.1. Provider/endpoint list in e-AS2 Connect**

In multi-provider enabled installations you find more than one partner profile in the list. In the case of e-integration there are four entries as follows:

EDICC	This is the traditional e-integration EDI platform, the production instance.
-------	--

## Multi-provider logic configuration

EDICC-TEST	This is the traditional e-integration EDI platform, the test instance.
EDIESKER	This is the new Esker EDI Services EDI platform, the production instance.
EDIESKER-TEST	This is the new Esker EDI Services EDI platform, the test instance.



The screenshot shows a dialog box titled "Edit partner" with a close button (X) in the top right corner. The dialog contains the following fields and controls:

- Partner details [EDICC] http://213.203.210.140:5080/as2in
- AS2-From: myIdentifier
- Server: 213.203.210.140
- Ack mode: no (dropdown menu)
- AS2-To: (empty field)
- Port: 5080
- URI: /as2in
- sign
- encrypt
- Retries: 1,3,10
- Description: e-integration EDI CC

At the bottom of the dialog are two buttons: "Save" and "Cancel".

**Figure 7.2. Editing a provider profile**

When editing a profile, the connection parameters (server, port, URI) or not offered for change anymore. It's not necessary to change them, because there is a dedicated profile for every end point to connect to. You still have to enter your own identifiers in the field "AS2-From". We recommend to use different own identifiers in all four profiles. Otherwise correct mapping of incoming messages to the corresponding provider profiles will not be possible in full multi-provider mode. However, if suitable and if this has been agreed with e-integration, it is also possible to use identical own identifiers in all partner profiles.

## 7.2 The interface area

When running in full multi-provider mode an additional folder level exists within the `out` and `in` folder of the interface area.

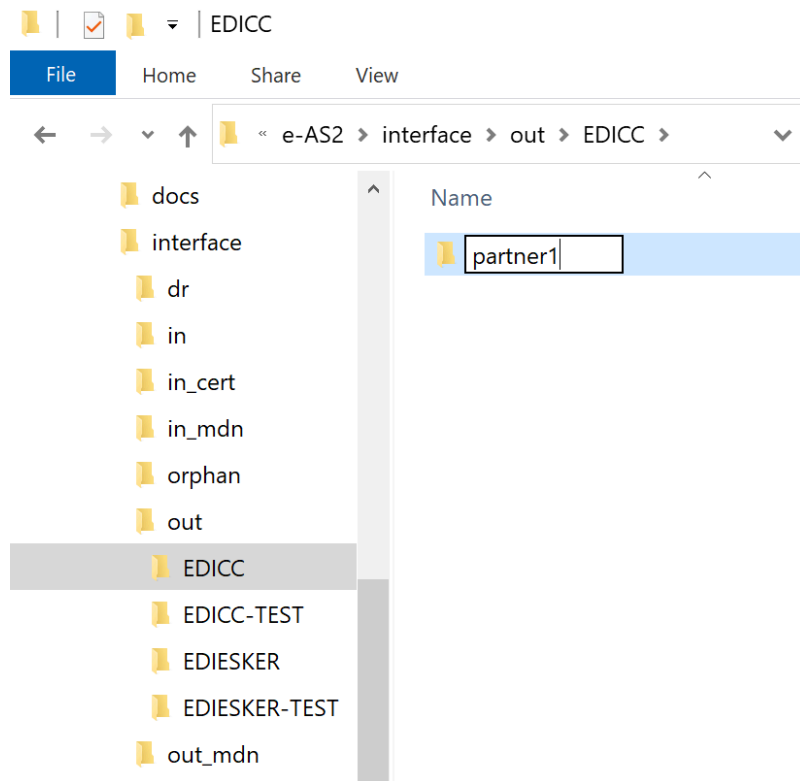


Figure 7.3. The interface area - sending files in full multi-provider mode

Four folders corresponding to the four partner/provider profiles respectively the four communication end points in e-integration's EDI data centers are present directly under `out`. By creating a sub-folder below one of these provider folders you not only identify the communication partner, you want your messages to be routed to, but also the provider that should do this.

So, basically you can distribute the set of partners you communicate with, freely over the four provider endpoints available. (In a production environment you would typically only use two of them, though, those without the term "TEST" in the profile ID or the folder name.)

Apart from that the mechanism of sending files is completely identical to the traditional logic. You place files in one of the partner folders. The files will be picked up and transmitted via the chosen provider.

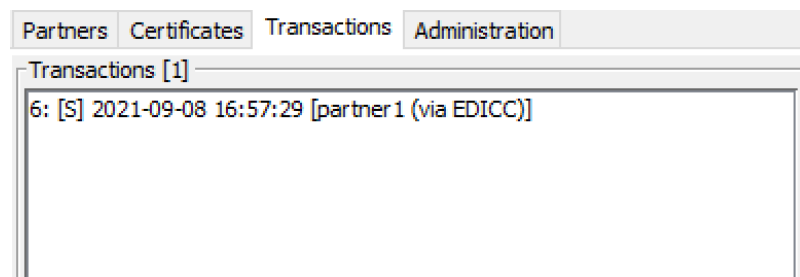


Figure 7.4. Send transactions

In the transactions list in the GUI send transactions now contain information about the provider the message was sent to.

As for received files, folders are created automatically as needed, just like in traditional mode. An additional provider folder level is maintained for received files, too.

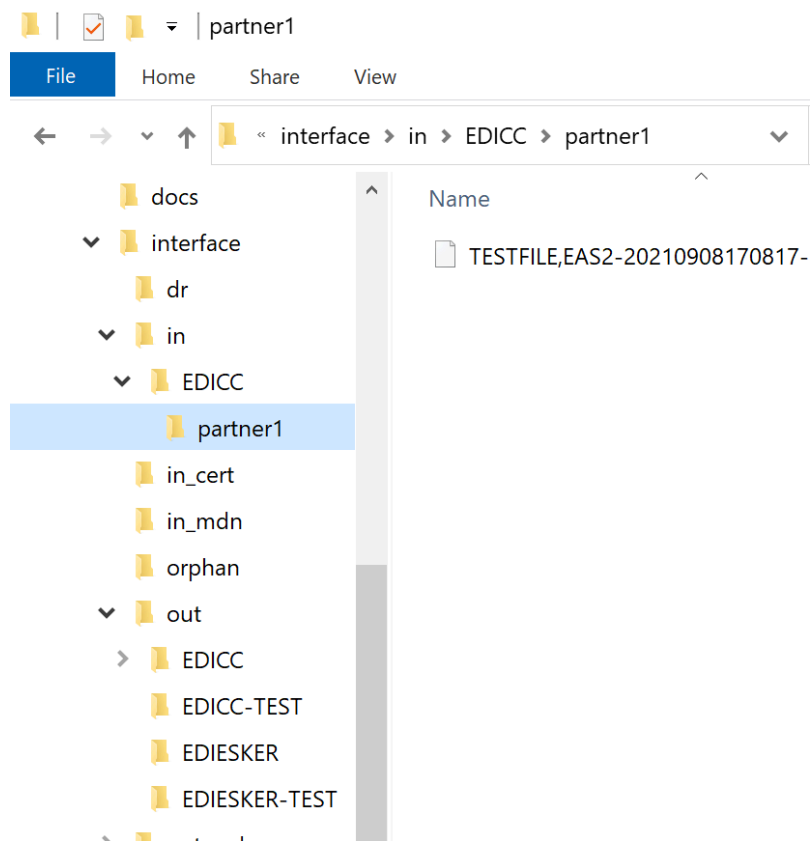


Figure 7.5. The interface area - receiving files in full multi-provider mode

### 7.3 Restricted multi-provider mode

While full multi-provider mode offers the user maximum flexibility, in real life quite often this is not needed. If you agreed with your provider, that routing is done based on content only, restricted multi-provider mode can (and should) be used.

In restricted multi-provider mode there are still four partner profiles corresponding to the four provider end points, but there are no nested folders in the interface area. So, you place files to be sent directly in the provider folder within the `out` directory. And you expect received files directly in the provider folder within the `in` directory.

In restricted multi-provider mode the own identifiers in the field “AS2-From” must be identical in all provider profiles. This is enforced by e-AS2 GUI during configuration. The field “AS2-From” is editable only in the first partner profile. On save any changes to that field will be propagated to the other partner profiles so that they remain identical without the need to touch every profile separately.

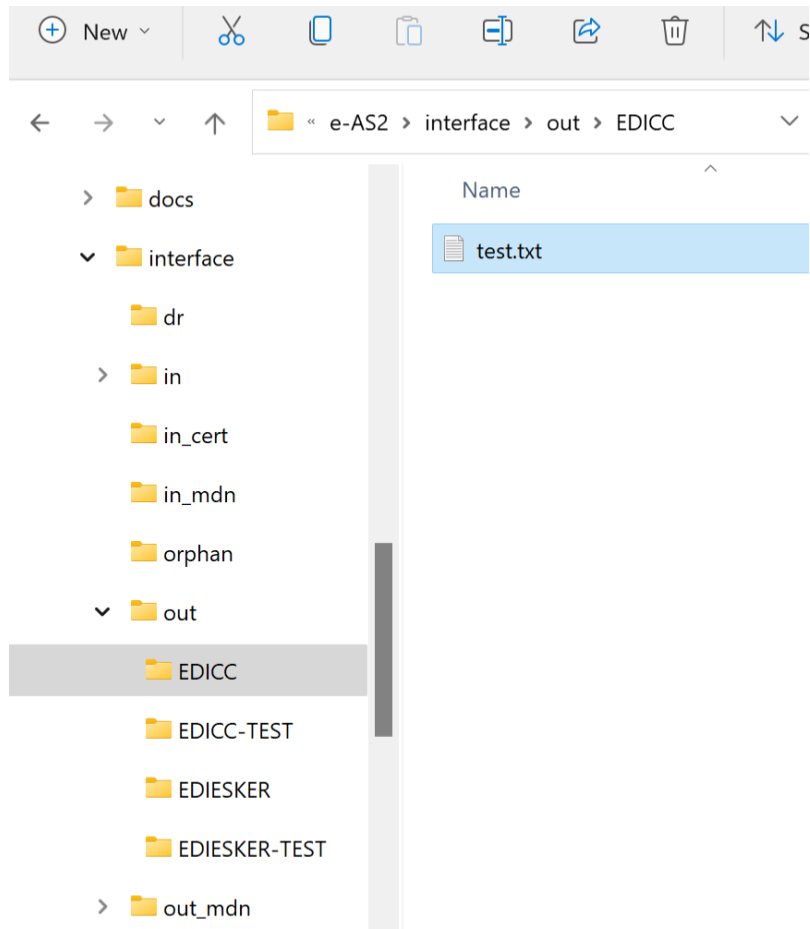


Figure 7.6. The interface area - sending files in restricted multi-provider mode

## 7.4 Turning off provider distinction

In case you do not want or need provider distinction for either incoming or outgoing transmissions, you can turn it off via corresponding property settings. When provider distinction is turned off, the additional folder levels below `out` or `in` are not being used. So, e-AS2 Connect basically behaves like in traditional mode.

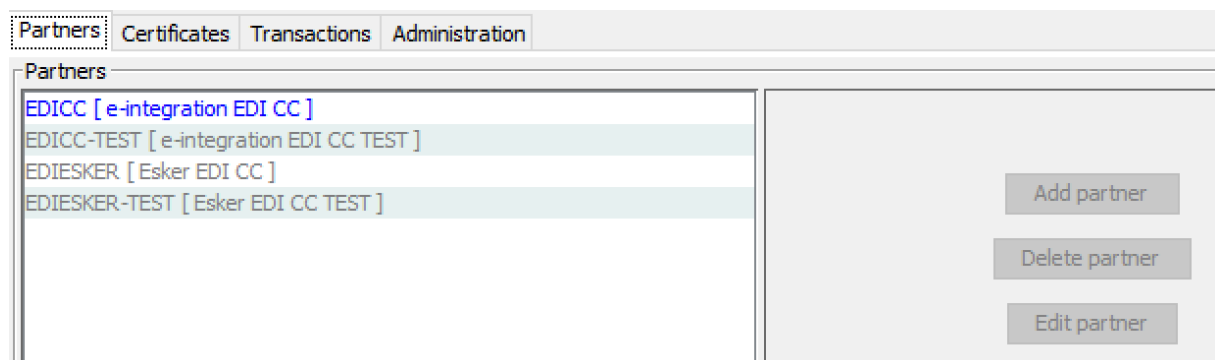


Figure 7.7. Single provider mode

As you can see in above screenshot all but the first entry in the partner list are being greyed out, if you turn off provider distinction. Only the first profile is being used. This is a valid mode of operation for existing customers that are connected to the traditional e-integration EDI platform.

#### Multi-provider logic configuration

Therefore this is also the initial mode of operation after upgrading existing pre-version 8.3 e-AS2 Connect installations.

For details on the above mentioned property settings see Section 10.2.23 (*Controlling provider related processing logic*).

## 8 Data fetch extension

*e-AS2 Connect is the only AS2 product with the option to simulate a data fetch.*

### 8.1 Motivation

AS2 is defined as a protocol where the respective entity with data to send connects to the opposite party and transmits the data. So in a data exchange relationship both communication partners always need to be standing by ready to receive incoming files. This receive standby must generally, with the exception of necessary maintenance times, occur around the clock, 365 days a year.

In a real data center operation, this requirement is easy to meet. Smaller companies which do not have a fully-fledged computer centre, however, may have problems with this requirement. e-integration therefore implemented an extension for e-AS2 Connect which incorporates the needs of such small-scale users without departing from the AS2 protocol.

So we're assuming a situation where you are unable to have the computer where e-AS2 Connect is installed online 24 hours a day. Possible reasons can be that the respective system is only enabled during office hours and is shut down at the end of the day, or there is no permanent internet connection, it is only established on demand. If this does not apply to you, you can simply skip this chapter. You can (and should) simply ignore the "Fetch data" button in the graphical user interface.

### 8.2 The solution

As previously mentioned, the AS2 protocol specification does not allow for data to be fetched like for example FTP does with the get command. Extending the protocol by a true fetch mode was not an option when developing e-AS2, as this would have been beyond the scope of the protocol definition. Instead, the data fetch extension was implemented as a special online message to the e-integration EDI-CC.<sup>34</sup>

If you wish to use the data fetch extension, please notify e-integration. The configuration in the e-integration EDI-CC will then be adapted accordingly so that files to be transmitted to you will be put back after a failed attempt. I.e. the EDI-CC will always first attempt to transmit a file straight away. If your e-AS2 Connect is online at the time, it will also receive the file straight away. However, if your system is not online, no additional transmission attempts will be made. Instead, the EDI-CC will put back this job and wait for your e-AS2 Connect to connect.

e-AS2 Connect is able to send a special message to the e-integration EDI-CC and report itself online. As soon as the EDI-CC receives this online message, any put back jobs will be returned to active and transmission attempted again.

### 8.3 Practical handling

In practice the data fetch extension is quite simple to use. Once your e-AS2 Server is online, start the GUI. Then select the entry `EDICC` from the partner list and press the "Fetch data" button. This will trigger the online message to immediately be transmitted to the EDI-CC. After a few seconds you will receive a return message on the success.

---

<sup>34</sup>If you're connected to a different AS2 provider with e-AS2 Connect, please enquire if the data fetch extension is supported.

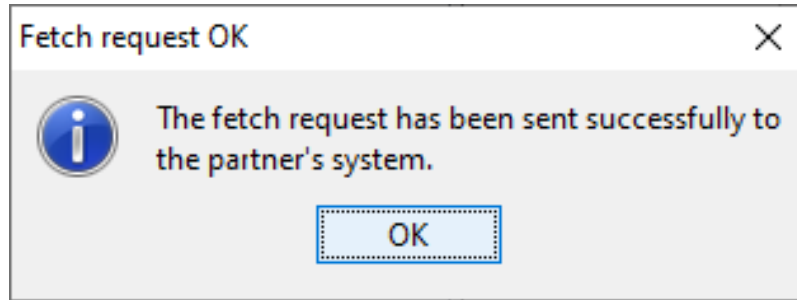


Figure 8.1. Data fetch completed

If a different message appears at this point, please contact the e-integration EDI-CC so they can check the configuration for you and if necessary modify it.

Once you have successfully transmitted your online message to EDI-CC this way, wait a few minutes. If there are files put back for you, they will be transmitted to you after no more than two minutes.

## 8.4 Auto fetch function

The interactive way of reporting yourself online via the GUI as presented above has the advantage of giving the user immediate feedback on the success. If you see an OK message you can rest assured – the e-integration EDI-CC received the message. The data they have put back for you will be transmitted to you shortly after. From this time on new files will be transmitted immediately as long as your e-AS2 server remains online. The “Fetch data” button will not need to be clicked again. You can close the GUI. Your server will remain ready to receive.

In order to forego having to start the GUI after restarting the server you can also automate the step of reporting your system online by enabling the auto fetch function as described in Section 10.2.7 (*Enabling autofetch*). This way the e-AS2 server will send an online message to the e-integration EDI-CC immediately after starting and every hour thereafter to trigger the data fetch. Please remember, when using the auto fetch function you will not receive direct visual feedback. However, errors whilst connecting will be recorded in the log file.

## 8.5 Limitations

Since the data fetch process was implemented AS2 compliant, data can still only be transferred if your e-AS2 server can be reached from the Internet, though not constantly, but during the times when you are online.

So the descriptions in Chapter 5 (*Firewall and Proxy configuration*) still fully apply, with the exception that the server does not need to be online at all times when using the data fetch extension.

Ad hoc internet connections with varying IP addresses do not pose a problem, as the current IP address of the e-AS2 server is transmitted to the e-integration EDI-CC during the data fetch request. They always use the last IP address transmitted for subsequent connections.

Computers that are being operated within company networks where the firewall hides the actual IP address to the outside via the NAT functionality<sup>35</sup>, on the other hand, are normally not suitable for data fetch mode. Additional firewall configurations would be required for these systems to be accessible. If this applies to you, please contact your firewall administrator about this issue. This problem may also occur when using DSL routers. Please refer to the router documentation to determine the suitable configuration.

<sup>35</sup>Network Address Translation

## 9 Command on receive

*The command-on-receive mechanism allows you to actively pass receive data to subsequent systems like ERP systems or similar.*

As you have learned in the course of this document so far, the coupling between e-AS2 and the preceding or subsequent systems is loose. In both directions, sending and receiving, it is based on interface directories in which the data provider places files with the expectation that they will be automatically detected, consumed and processed. This requires continuous directory monitoring on the file consumer's side.

In send direction the data provider is your ERP system (or another source of data to be transmitted). You place your files in the `out` directory. e-AS2 detects the new files with the integrated directory monitor and picks them up for transmission.

In receive direction, e-AS2 is the data provider. After receiving a new message, e-AS2 will place the received file(s) in the `in` directory. The process is then completed for e-AS2. The expectation is that the subsequent processing system detects new files via data monitoring, and delete it from the interface directory after having successfully picked them up for processing.

e-AS2 offers an alternative solution for the receive direction. The prerequisite for this is that a command line program is available which can be run to explicitly pass the received file to the subsequent system. We call this mechanism "command on receive" (the command is invoked after receiving data). This will eliminate directory monitoring by your processing system. The further course of this chapter documents how to configure e-AS2 for using command on receive and what to consider when developing the interface program.

At this time we would like to point out the user is responsible for the program provided as command on receive working correctly. Please carefully read this chapter in its entirety and thoroughly test handling via command on receive before going live with them.

### 9.1 Activating and configuring command on receive

Initially, right after installation, the use of command on receive is not activated in e-AS2. Activating and configuring this interface will require some parameters to be set in the file `EAS2.properties`. In the file `EAS2.properties.template` you will find the following annotated section with examples of the three available parameters.

```
# Command to execute after receiving data
#command.on.receive = ./process_message.ksh

# Postpone asynchronous MDN until command completion. This is effective only if
# "automatically send async Ack" is active in partner profile and the sender
# requested an asynchronous MDN.
# Default: 0
#command.postpone.async.mdn = 0

# Wait timeout for command execution (in seconds). If the command is still
# running then it will be stopped. Processing will be signalled as erroneous.
#command.wait.timeout = 20
```

Copy this sample configuration into your `EAS2.properties` and set the parameters according to your requirements.

#### 9.1.1 Activating command on receive

The interface for command on receive is activated by setting the property `command.on.receive`. Remove the comment character at the start of the line and configure the complete path

to the command to be invoked. After restarting the e-AS2 server the interface for command on receive will then automatically be used for all files received.

A functional skeleton for implementing this type of interface program is part of the e-AS2 delivery. You will find the script `process_message.ksh`<sup>36</sup> in the installation directory. You can use this script (after having copied it to a different name) as a template for your own interface program.

When configuring the property, you can also specify any necessary command line parameters or options that will be taken into account later when the program is called by the e-AS2 server. The other command line parameters dynamically generated by the server then follow right behind the fixed parameters.

### 9.1.2 Retain asynchronous MDN

If the sender of the data requested an asynchronous MDN, e-AS2 will generate it as soon as the data received have been successfully saved to the file system. This will also happen when command on receive is configured. So, a positive MDN will be returned to the sender, regardless of whether the command on receive was successfully called or not.

You can change this standard behaviour by setting the property `command.postpone.async.mdn` to the value 1. This will cause e-AS2 to delay generating asynchronous MDNs until the command has been started and finished processing.

Please note, in this case erroneous termination of the successive commands will cause a negative MDN to be transmitted to the sender of the data. If you don't want this, leave this property unset or set the standard value 0.

### 9.1.3 Timeout during program invoke

Having a server invoke external programs is a delicate matter, as one must at any rate avoid an error in the external program negatively impacting the invoking server. On the e-AS2 server, starting commands on receive is executed with a separate thread. Here, all pending commands are processed sequentially.

Moving to a separate thread adequately partitions the execution of commands from key e-AS2 functions so that normal processing, meaning the actual handling of communication sessions is never impacted by executing the command.

Sequential processing of command ensures the execution of commands does not cause excessive selective peaks in the system load. However, it does make handling susceptible to long command execution times which can occur due to programming errors or errors in the subsequent system. Invoking the command can therefore be secured with a timeout.

The property `command.wait.timeout` in `EAS2.properties` defines this timeout in seconds. If the invoked program does not finish within the specified time, e-AS2 will stop to wait and the external process will be terminated. This will be considered to be a failure to pass the data to the subsequent system and a negative MDN will potentially be created.

## 9.2 Procedure of handling command on receive

The e-AS2 server processes the files waiting for a command to be invoked with a separate thread. Any files ending up in the respective partner directories will be captured. A fitting command line will be constructed for each file and the command will be invoked.

---

<sup>36</sup>Or `process_message.bat` when installing in MS Windows.

A program which is to be suitable as command on receive for e-AS2 must maintain a specific, defined calling interface.

### 9.2.1 Command line

The command is invoked with two dynamically generated parameters. They are the following contents, which are passed to the program in this order.

Partner ID	The internal partner identification from the partner profile this incoming data was assigned to.
File name	Name of the file in the interface directory.

### 9.2.2 Command termination

The e-AS2 server expects the invoked program to delete all the files associated with the transaction from the interface directory once processing the data has been completed.

In addition, the program should return a meaningful exit code. The exit code 0 is considered as success. All other exit codes mean data processing failed.

### 9.2.3 Error handling

In the event of an error, the exit code returned is written to the e-AS2 log file so it can be used to analyse the problem. If applicable, the exit code will further be added to the asynchronous MDN returned to the sender of the data.

Generally, after invoking the command, the e-AS2 server will check the associated files. This ensures files not processed – due to an erroneous function of the program provided by the user – will remain and are processed again in the next run.

If the program invoked as successive command completes successfully, so with exit code 0, e-AS2 will expect the program to delete the files associated with the transaction. However, if all or some of the associated files are still in the directory, the e-AS2 server will move these to a subdirectory named `_done_`. This ensures they are not picked up again and the command invoked for these files.

Regularly check the directory `_done_`. During normal operation this directory should not exist or should be empty. If there are files in this directory, it means your command was not implemented correctly.

If the program invoked as command on receive finishes with an error, i.e. with an exit code other than 0, the associated files should still be in the same place, as processing obviously failed. The e-AS2 server will then move these to a subdirectory named `_errors_`.

Files can occasionally also end up in `_errors_` although the command on receive is working correctly, due to a temporary malfunction. E.g. your ERP system could be down temporarily and data import therefore fail. You can always remove these files from the `_errors_` directory and up one level after correcting the error. They will then be picked up again and the command on receive invoked.

Never change the names of the associated files. The processing logic in e-AS2 is based on the file names originally assigned by e-AS2 and will no longer work if these are changed.

### 9.2.4 Logging

e-AS2 logs every command call on `INFO` level. Detailed messages are output on `DEBUG` level. In addition, the program invoked as command on receive can issue its own messages which are then copied to the e-AS2 log file.

## Command on receive

Messages the command outputs to `stdout` are logged on level `DEBUG`. Messages the command outputs to `stderr` are logged on level `ERROR`.

As the program author you should therefore be sure the command you develop outputs meaningful messages, particularly in the event of an error, for e-AS2 to provide conclusive information about the cause of the error. Use `DEBUG` messages to your liking to better analyse the functionality when invoked by the e-AS2 server when developing the program.

### 9.2.5 Functional test during server startup

In addition to the above call interface for processing incoming files, a program used as command on receive must support an additional call form. If the program is invoked without parameters, it must output a brief text message in a line and end with exit code 0.

During startup the e-AS2 server will test the availability of the command by invoking the configured program without parameters. If the program can't be started or closes with an exit code other than 0, the server will deactivate the feature and the command-on-receive feature won't be available.

## 10 Configuration for experts

*In addition to the configuration options accessible from the graphical interface, there are other parameters which can affect how e-AS2 Connect behaves.*

This chapter documents parameters set or edited via text files, i.e. not via the graphical interface. During installation the parameters presented here are automatically set to reasonable default values, so that the software works properly with them. They normally do not require changes. However, if you do wish to make changes to these, you will need a suitable text editor for your operating system. Edit the respective file and save the edited version. (First make a backup copy of the file you wish to change!)

### 10.1 log4j2.xml

e-AS2 Connect logs its jobs in the `eas2s.log` file, found in the installation folder. The scope of logging is adequate to review which steps e-AS2 Connect has taken in the past. Here a sample excerpt from the log file (partly shortened):

```
11:53:47.446 INFO [DIRTRANS-4] [] [] - ### START DIRECTORY TRANSFER THREAD ###
11:53:47.451 INFO [ IFTRANS-4] [14] [456] - new file 'ORDERS.70258' belongs to partner 'acme_corp'
11:53:47.451 INFO [ IFTRANS-4] [14] [456] - transmitting message - MSGID: EAS2-202203...
11:53:47.453 INFO [ IFTRANS-4] [14] [456] - connecting to as2.acme.corp(207.29.55.18):4080 via HTTP ...
11:53:47.453 INFO [ IFTRANS-4] [14] [456] - transmitting ...
11:53:47.453 INFO [ IFTRANS-4] [14] [456] - data sent (2721 bytes) - waiting for ack ...
11:53:47.601 INFO [ IFTRANS-4] [14] [456] - received ACK or sync MDN
11:53:47.601 INFO [ IFTRANS-4] [14] [456] - MDN received from ...
11:53:47.729 INFO [DIRTRANS-4] [] [] - ### END DIRECTORY TRANSFER THREAD ###
```

At the left of each line you will see the time the entry was generated, then the log level followed by the thread identifier in square brackets, then the transaction number in square brackets, the document ID in square brackets, and finally the actual log message after the dash. All messages for a process can be collected using the thread identifier. Due to the parallel processing of various tasks based on the architecture, messages related to a process do not necessarily need to be directly successive in the log file. In this case the thread identifier helps to identify related lines. The transaction number allows all messages for a transaction to be collected, even if their processing spans several threads.

Changes to the `log4j2.xml` file automatically take effect after 30 seconds at the latest. The software does not need to be restarted for this.

#### 10.1.1 Changes to the log level

In the event something does not work as expected, more detailed logs may be useful. For this purpose, open the file `log4j2.xml`. Here you will see the following section:

```
<!-- Log level for e-AS2 -->
<Logger name="de.ei.eas2" level="info"/>
```

Here you can change the general e-AS2 Connect logging from log level “info” to level “debug”. (For detailed information on log4j2 and the supported log levels, please refer to <https://logging.apache.org/log4j/2.x/manual/>.)

### 10.2 EAS2.properties

The installation folder also contains the file `EAS2.properties`. This contains some key parameters. Open the file with a text editor of your choice. Lines beginning with a hash mark (#)

are comment lines and do not affect the software's behaviour. Commenting out a line with a parameter by prefixing it with the hash mark, an internal default setting will be applied to the respective parameter.

When installing e-AS2, `EAS2.properties` will be added in an initial state. During installation you will be entering some parameters in the respective forms. Following installation these can then be found in `EAS2.properties`. You can change the respective values by repeating the installation process. However, it's easier and faster to make the necessary changes directly in the file using a text editor.

After changing the `EAS2.properties` file, you will need to stop and restart the software for the changes to take effect.

Most of the parameters in `EAS2.properties` control how the e-AS2 server behaves. However, some also affect the behaviour of the GUI client. All the parameters which can be defined in `EAS2.properties` are documented in the course of this text.

A sample of the basic changes which may be made can be found in the file (commented out, i.e. inactive). The more rarely used settings will need to be added manually if necessary. To avoid typos, we recommend copying and pasting new settings from this manual and then editing them according to your needs.

Software updates do not affect the `EAS2.properties` file.<sup>37</sup> However, a new version of the `EAS2.properties.template` file will be installed, which may contain samples of new configuration options.

**Note.** This manual only describes the parameters relevant to e-AS2 Connect. The `EAS2.properties` file contains samples of other parameters not documented here. Please refer to the e-AS2 Enterprise manual for information on these parameters.

### 10.2.1 Communication parameters, HTTP

When installing e-AS2 Connect you were queried for data communication information. Your entries in the respective setup dialogue were automatically copied to `EAS2.properties`. You will find these parameters in the following section:

```
# your hostname and port / POST request URI for HTTP connection
connection.http.host = yourhost
connection.http.port = 5080
connection.http.uri = /as2in
```

The computer name for your system is automatically entered as the host name (`connection.http.host`) during setup. You will find this name here unless you have already changed it.

Behind `connection.http.port` you will find the port under which the e-AS2 server you are currently configuring expects incoming data connections.

Behind `connection.http.uri` you will find the post request URI for incoming connections. If a remote server attempts to establish a data connection using a different URI, the connection will be rejected, even if the port is correct.

**Important!** These parameters define how remote AS2 systems establish a connection to your e-AS2 server. This should particularly be kept in mind with respect to asynchronous MDN. The AS2 protocol requires the data sender to transmit the response address for asynchronous

<sup>37</sup>With the exception of changed parameters which may have been entered in one of the forms during installation.

MDN to the recipient as header information. e-AS2 Connect accesses the above properties for this purpose. So you must not configure an internal computer name here. Instead, enter the host name under which the e-AS2 server can be accessed from outside. The same applies correspondingly for port and URI. This must particularly be considered with respect to firewalls or proxy servers, which could potentially transcribe (i.e. change) the three values. If you find it hard to set these properties so that they work properly in this context, you can also define the URLs for async MDN explicitly with separate properties. (See ??? for details.)

## 10.2.2 Parameters for proxy use

If you want e-AS2 Connect to establish the outgoing connection via proxy server, you will need to set the following parameters:

```
# Additional settings for proxy connection
#http.proxy.host = <some host>
#http.proxy.port = <some port>
#http.proxy.user = <some user>
#http.proxy.password = <some password>
```

`http.proxy.host` is used to specify the host name of the proxy server. `http.proxy.port` is used to specify the port number to use for contacting the proxy server.<sup>38</sup>

The following two specifications are optional and only needed if the proxy server requires authentication. In this case, use `http.proxy.user` and `http.proxy.password` to specify the parameters for “Basic Authentication”.

Please note, when using a proxy server for incoming connections (reverse proxy) the HTTP parameters must be adjusted accordingly. In Section 10.2.1 (*Communication parameters, HTTP*) we already referred to this situation. Please read Section 5.2 (*Using proxy servers*) again at this point and consult with your firewall administrator.

## 10.2.3 Communication parameters, configuration

During the e-AS2 Connect installation you were prompted for parameters for the configuration connection. Your entries in the respective setup dialogue were automatically copied to `EAS2.properties`. You will find these parameters in the following section:

```
# settings for configuration connection
connection.config.host = localhost
connection.config.port = 5070
connection.config.password = password
```

If the server and GUI client are installed on the same computer, you were not prompted for a computer name for the configuration connection during setup. The setting `connection.config.host` is automatically set to “localhost” and should not be changed.

If you are using a computer which does not have a GUI only installation yet, you were prompted for the host name of the e-AS2 server during setup. The information entered at that point can then be found in `EAS2.properties` under `connection.config.host`. Only change this setting when moving the e-AS2 server to a different host.

Behind `connection.config.port` is the port under which the e-AS2 server you are configuring expects configuration connections. The GUI client uses this port to connect to the server. Please note, the GUI and server will each have their own copy of `EAS2.properties` when

<sup>38</sup>At this point we do not distinguish between HTTP and HTTP/S, since proxy servers usually accept requests for both protocol types via the same port.

running on different computers! If the port numbers do not match, a configuration connection will not be established.

Behind `connection.config.password` is the password for configuration connections. Use a password which is not trivial if the configuration port is open and you are not running the GUI and server on the same computer!

#### 10.2.4 Connect timeout

The communication parameters for the configuration connection is followed by the definition for the connect timeout:

```
# connect timeout (in seconds)
#tcpip.connect.timeout = 10
#smtp.connect.timeout = 2
#pop3.connect.timeout = 2
```

This determines how long e-AS2 Connect will wait for a connection to be established. This value will be configured separately for TCP/IP in general, SMTP and POP3. The first value applies to all HTTP and HTTP/S connections, as well as the connection between the GUI client and e-AS2 server. Since this mainly applies to outgoing connections (over the Internet), the value should be a bit higher. The default is 10 seconds. If you frequently encounter errors establishing connections due to timing out, you can increase this value.

The two other values determine the timeouts for connections to SMTP servers and POP3 servers. Here the defaults are a bit lower, since it is assumed these servers are within the LAN. If you connect to external mail servers via STMP or POP3 connections, set the values adequately high so your connections can be established with certainty.<sup>39</sup>

Please remember to remove the comment sign at the beginning of the respective line for your changes to take effect.

#### 10.2.5 Read Timeout

The following section facilitates changing the read timeouts when communicating using the various protocols:

```
# read timeout for communication (in seconds)
#tcpip.read.timeout = 60
#smtp.read.timeout = 10
#pop3.read.timeout = 30
```

The read timeout defines how long after sending a protocol unit the communication thread will wait for a response from the other end before considering the connection dropped and ending the session. Analogous to the connect timeout, this value can be specified separately for the various protocols.

You will find the defaults in the commented out sample lines in `EAS2.properties`. We recommend only changing these values if you require longer timeouts. Do not set the timeouts shorter than the defaults.

#### 10.2.6 DNS cache settings

If you use e-AS2 server names instead of IP addresses during configuration, the name will first be DNS resolved each time a connection is established. This is implicitly executed by the

---

<sup>39</sup>The timeout value for POP3 is irrelevant to e-AS2 Connect, as it does not establish POP3 connections. SMTP is only used to send operator e-mails. Here the mail server should ideally be located on the LAN and changing the timeout therefore not be required.

JVM<sup>40</sup>. The JVM saves the result of the DNS resolution to a DNS cache and will use that saved IP address for additional requests related to the same server name to avoid repeating the DNS query each time a connection is established.

If necessary, you can change the age of entries in the JVM DNS cache with settings in `EAS2.properties`. If you have e.g. connected communication partners which regularly have new IP addresses assigned for the server name configured on their end, you rely on a relatively short cache entry age, as this will otherwise result in connection errors for an extended period after the IP address has changed.

```
#networkaddress.cache.ttl = 30
#networkaddress.cache.negative.ttl = 10
```

`networkaddress.cache.ttl` is used to specify the age (in seconds) of entries in the DNS cache generated based on successful DNS name resolution for an IP address. `networkaddress.cache.negative.ttl` is used to specify the age (in seconds) for entries generated based on a failed DNS resolution.

The e-AS2 server records these values in the log file on each startup. As long as the properties described here are not set, the JVM default values (typically 30 seconds and 10 seconds) will be recorded. By setting the respective properties in `EAS2.properties` you can change both values in both directions, so defining a longer or shorter age. Unless there is good reason to change these, we recommend keeping the proven standard values.

## 10.2.7 Enabling autofetch

You can enable the autofetch function to automate the data fetch extension.

```
# Automatic data-fetching for Connect Client (Default: off,
# set to 1 to enable)
#interface.dbwatcher.fetch.automatic = 0
```

Set the parameter to 1 to enable the function. A data fetch connection to the e-integration EDI-CC will be established immediately after restarting the server and once an hour after that.

## 10.2.8 Email configuration

The e-AS2 server can send SMTP emails to operations to draw attention to irregularities. With this function activated the following events will trigger email notifications:

1. A file could not be sent and all configured retries have been exhausted.
2. A pending transaction was manually stopped by the user.
3. A file received was not processed correctly.
4. A certificate is about to expire.
5. Old files were found in the interface.

The email generated contains more information on the event classification.

This function is disabled by default. It can be activated by configuring the following properties.

```
# mail configuration (for operator mails and EDI mails)
#mail.host = localhost
#mail.port = <some port>
```

---

<sup>40</sup>Java Virtual Machine

## Configuration for experts

```
#mail.user = <some user>
#mail.passwd = <some user>
```

`mail.host` is used to configure the server to be used as the SMTP-MTA. This can be a computer name or an IP-address. The e-AS2 server establishes a direct SMTP connection to this computer. The firewall rules must allow this connection.

The port for the SMTP connection can be specified under `mail.port`. This information is optional. It is only required when using a port other than the standard SMTP port.

If the SMTP-MTA user requires authentication, you can configure the required login parameters (user name, password) in the properties `mail.user` and `mail.passwd`. This information is also optional and may be left blank for email servers without authentication.

This concludes the connection parameters for using the email MTA. With respect to sending operator emails there are three other configuration options.

```
# mail receipt (set this one, too, to activate operator mails)
#mail.to = AS2 Operator <operator@domain.tld>

# additional mail receipt (set this one, too, to activate info mails)
#mail.to.info = AS2 Info <as2info@domain.tld>

# mail sender (optional, default: mail address of OS user)
#mail.from = eAS2-Server <as2@domain.tld>
```

`mail.to` is used to configure the email address of the desired recipient of operator mails. Only one recipient can be specified.

`mail.to.info` is used to configure the email address of the desired recipient of info mails. Only one recipient can be specified.

`mail.from` is used to specify the sender's email address for the operator email to be used in place of the standard value generated by the email system.

### 10.2.9 Command on receive

The following settings control the processing of incoming files with commands on receive.

```
# Command to execute after receiving data
#command.on.receive = ./process_message.ksh

# Postpone asynchronous MDN until command completion. This is effective only if
# "automatically send async Ack" is active in partner profile and the sender
# requested an asynchronous MDN.
# Default: 0
#command.postpone.async.mdn = 1

# Wait timeout for command execution (in seconds). If the command is still
# running then it will be stopped. Processing will be signalled as erroneous.
#command.wait.timeout = 20
```

Set `command.on.receive` to the name of a program (including complete path) to run for processing received files. Incomplete paths will be interpreted relative to the e-AS2 installation directory. Attach options and fixed parameters to the program name if necessary.

After restarting the e-AS2 server (and subsequently every time the server is started) the configured program will be run once without parameters for testing to ensure it exists and is executable. If the test is successful, the interface for executing commands on receive will be activated on the server. You will also see the controls required for the configuration in the GUI.

The setting `command.postpone.async.mdn` to 1 will make the server postpone returning asynchronous MDNs. Async MDNs are then no longer generated directly after data is received, but instead only after the command on receive has been executed.

Set a timeout for waiting for the program invoked to end with `command.wait.timeout`. Without this configuration, the server will wait for the program to end indefinitely.

### 10.2.10 IO-Trace

By default, the so-called “intelligent IO-Trace” is activated in e-AS2. This means IO-Trace files will automatically be generated for erroneous transactions. Generating IO-Trace files for transactions without error, on the other hand, must be activated by a trigger file.

```
# Activate intelligent IO trace (write traces only for erroneous sessions,
# default: 1)
#iotrace.mode.intelligent = 1
# IOTrace directory
#iotrace.directory = tmp
```

If you generally do not want IO-Trace files to be generated, set this value to 0. This does not impact the functionality of the trigger file.

### 10.2.11 Checking certificate validity

Every day the e-AS2 server checks the validity of all certificates under management.<sup>41</sup> If a certificate is about to expire, a log entry is generated and an operator email sent, provided the e-AS2 email system is activated.

The time of the check and the advance warning time for expiring certificates (in days) can be configured in `EAS2.properties`.

```
# The time for executing the daily certificate validity check. (Default: 23:59)
# Configure mail.host and mail.to to receive warnings by mail.
#certificate.validity.check = 23:59
# Or turn off validity checks.
#certificate.validity.check = off
# Warning threshold for expiring certificates (in days)
#certificate.validity.threshold = 30
#certificate.validity.threshold.2 = 3
```

Use `certificate.validity.check` to configure the time to run the daily check.

Use `certificate.validity.threshold` to specify the desired advance warning time. Without any explicit configuration, the default value of 30 days will apply.

Use `certificate.validity.threshold.2` to specify the second advance warning time. Without any explicit configuration, the default value of 3 days will apply.

**Note:** The GUI shows expired certificates in red. Certificates that are about to expire are shown in yellow.

Regardless of the time for the regular certificate validation, the e-AS2 server will perform validation of all certificates on every startup.

### 10.2.12 Automatic backup

The e-AS2 server automatically backs up data on a daily basis. The entire content of the database is exported and written to a ZIP file. The export consists of SQL INSERT statements

---

<sup>41</sup>This explicitly also includes SSL certificates.

which can be used to – starting with a new database with empty tables – restore the full database contents at the time of export in the case of emergency.

By default, data is backed up at 00:01 a.m. The export file is written to the `backup` directory under the e-AS2 installation directory using a generated file name including timestamp. You can change the behaviour as needed using the following parameters.

```
# The time for executing the daily database export. (Default: 00:01)
database.export.time = 00:01
# Directory for saving the export files
database.export.directory = backup
```

If you don't wish to automatically back up data, you can disable it by setting `database.export.time` to `off`.

### 10.2.13 Interface area

The key element for data exchange with internal processing systems is the interface area, which is by default rooted in the e-AS2 Connect installation folder.

```
# Root of primary42 interface directory tree
#interface.root.primary = interface
```

The default configuration uses the `interface` folder as the start of the primary interface area. To change this, remove the hash mark (#) at the beginning of the line and enter a different path in the file system as the `interface.root.primary`.

If the information doesn't begin with a slash (/) or a drive letter (C:), it is a path relative to the installation folder. If the information begins with a slash, an absolute path will be configured on the installation partition. For Windows users: To address a different partition, begin with the partition identification in form of a drive letter, followed by a colon.

**Important!** As a Windows user, always also use the forward slash (/) to separate the different path components. Do not use the backslash (\) common in MS Windows!

#### Examples for Windows users

e-AS2 installed under `C:\e-integration\e-AS2`.

```
interface.root.primary = data/edi
```

This specifies `C:\e-integration\e-AS2\data\edi` as the start of the interface area.

```
interface.root.primary = /data/edi
```

This specifies `C:\data\edi` as the start of the interface area.

```
interface.root.primary = D:/data/edi
```

This specifies `D:\data\edi` as the start of the interface area.

#### Examples for Linux/Mac OS X users

e-AS2 installed under `/opt/e-AS2`.

```
interface.root.primary = data/edi
```

---

<sup>42</sup>When using e-AS2 Enterprise you can additionally configure a secondary interface area. e-AS2 Connect, however, can only use one interface area.

## Configuration for experts

This specifies `/opt/e-AS2/data/edi` as the start of the interface area.

```
interface.root.primary = /data/edi
```

This specifies `/data/edi` as the start of the interface area.

In any case, the entire folder structure of the interface area is automatically created when the e-AS2 server is started, if it does not already exist. So after changing the configuration you will not need to manually add these folders.

**Note:** Network drives cannot be used as a location for the interface area. If you want to exchange data in the local network based on network shares, the computer on which e-AS2 Connect is installed must share a local partition, which is then accessed by another computer in the network.

### 10.2.14 Building file names for incoming messages

This setting is a flexible method for controlling which information about the data received to code into the file name.

```
# filename pattern to be used if input interface is directory based
# The strings in brackets are placeholders for dynamically generated values.
#interface.in.filename.pattern = [TIMESTAMP]-[THREAD]-[TRANSACTION]-[AS2NAME]-
[MSGID]
# Default, if not set:
# interface.in.filename.pattern = [AS2NAME],[MSGID]
```

The file names are generated as specified, in which the character string to the right of the equal sign in addition to fixed text may contain any number of the following place holders in any location:

[AS2NAME]	Name of the transmitted file as specified by the sender of the data.
[MSGID]	Message ID for the message transmitting the file.
[TIMESTAMP]	Current timestamp, to milliseconds.
[THREAD]	Name of the Java thread processing reception.
[TRANSACTION]	Sequential transaction number.

These place holders including square brackets are replaced by the contents determined dynamically.

Special characters the file system does not allow in file names are replaced by underscores. These are the following characters:

```
\ / ? : * " < > |
```

Such characters could possibly appear in [AS2NAME] and [MSGID].

### 10.2.15 Changing the structure of a message ID

For outgoing messages e-AS2 structures the messages ID as shown in the example below to ensure global uniqueness:

```
EAS2-20110803155745-428-000002@192.168.62.140-sirius.e-
integration.de
```

Where:

EAS2	Is a fixed prefix all message IDs start this. This is followed by a dash.
20110803155745-428	The timestamp, to the millisecond, YYYYMMDDhhmmss-xxx. This is followed by the character @ as separator.
000002	Is a six-digit sequential number managed in the main memory of the e-AS2 server. Each time the server is restarted it is reset to 00000.
192.168.62.140	The IP address of the computer running e-AS2. This is followed by a dash.
sirius.e-integration.de	The DNS name of the computer running e-AS2. This is followed by a dash.

The general structure of the message ID cannot be changed. The component ID address and DNS name, however, can be replaced by actual differing values if necessary:

```
message.id.address = <some address>
message.id.hostname = <some hostname>
```

The values set here do not have to be valid IP addresses or host names. They will be copied to the message IDs as character strings as they appear. We recommend only configuring this in justified exceptions.

### 10.2.16 Preventing sending unfinished files

In e-AS2 data is sent based on the monitoring of directories. The outgoing directories in the interface area are scanned regularly. New files are detected and added for processing.

This process holds the potential risk of sending unfinished files. A new file becomes visible in the file system as soon as it is added. This happens, when writing of the file begins. The application generating the file may be busy for quite a while until writing the contents to the file is completed and the file is closed. This significantly depends on the total length of the file. If e-AS2 detects a file too soon, before the entire contents have been written, this file may potentially only be transmitted to the other end in part.

e-AS2 has a built-in mechanism to prevent this. If a new file is detected in the file system, e-AS2 determines the time the file was last changed. If this time is less than one second back, the file is assumed as still growing, and ignored for processing. The next time the directory is scanned it is again detected as a new file.

The standard value of one second is – measured by today’s EDV standards – very high. It should practically always be sufficiently large to prevent unfinished files from being transmitted. If you still find your communication partner is complaining of receiving unfinished files, you can adjust the wait time as follows:

```
interface.dirwatcher.modified.wait = 1
```

The value is specified in seconds. Only increase it moderately when needed to prevent overly large delays in processing new outgoing files.

### 10.2.17 Cleanup routines

e-AS2 Connect manages partner master data and transaction records together in an integrated database. To ensure the database does not grow endlessly due to records continuously being added, e-AS2 Connect has built-in automatic cleanup routines. By default, transactions older

than 30 days are automatically deleted. In addition, the oldest transactions are automatically deleted if the number exceeds the maximum of 10000 transactions. They are deleted only, if their status allows it. So, transactions, that are still considered open or pending for further processing, will not be removed. You can change the preconfigured limits in `EAS2.properties`.

Additional cleanup routines ensure the folders `tmp` and `interface/orphan` are not populated with too many files. Files older than a defined maximum age are deleted regularly. The user is responsible for the in and out folders.

```
# Parameters for clean-up routines (age in days)
#dispose.queue.age.max = 30
#dispose.queue.count.max = 10000
#dispose.files.age.max = 30
```

Use `dispose.queue.age.max` to specify the maximum age of transaction records allowed. Older records will automatically be deleted.

Use `dispose.queue.count.max` to set the maximum number of transaction records allowed. If this limit is exceeded, the oldest records are automatically deleted until the limit has been reached again.

Use `dispose.files.age.max` to set the maximum age allowed for files in `tmp` and `interface/orphan`. Older files are automatically deleted.

Please note, for security reasons deleting old files is disabled if the interface area is outside the e-AS2 installation directory (see Section 10.2.13 (*Interface area*))!

### 10.2.18 Checking for old files

On request, e-AS2 Connect can regularly check the `out` and `out_mdn` directories for suspiciously old files. If files are found which are too old, you will be notified by operator mail.

The reason for this check is the way the software behaves in the event of communication problems with outgoing communication. As you read in a different section in this manual, files which cannot be transmitted after all retries are moved to the `_errors_` subdirectory. An operator e-mail will immediately notify you of this situation. The transfer must again be initiated by manually releasing the respective transaction once suitable measures have been taken to correct the problem. If as the operator you fail to manually release it, the file in `_errors_` will become an "old file" after a while. You will regularly receive additional warnings until the problem has been resolved.

Even after receiving data a situation may arise with old files if the subsequent application fails to detect and process files received.

The check for old files can be configured as follows:

```
# Check for old files in out directories (1 = yes)
old.files.check.outgoing = 0
# Check for old files in in directories (1 = yes)
old.files.check.incoming = 0
# Parameters for old files warnings (age in minutes)
#old.files.warning.threshold = 5
# Repetition of warning mails (in hours: default: 1)
#old.files.warning.repeat = 1
```

Adjust the values for these properties to your needs. Please note, these configurations are only useful if e-AS2 was configured to send operator e-mails (see Section 10.2.8 (*Email configuration*)).

### 10.2.19 Language

The language for the graphical interface can be changed. You can already choose between the German and English version when installing the software. You can change this selection anytime later using the settings in `EAS2.properties`:

```
# Language (de or deu for german, en or eng for english)
eas2.language = de
```

Currently only German and English are supported. The two- or three-letter ISO code is configured in lower case.

### 10.2.20 List line colouring in the GUI

For a better overview the lines of lists in the e-AS2 GUI alternate colours. If desired, the standard colour can be changed in `EAS2.properties`.

Likewise, you can also change the colour for highlighting particularly notable list entries.

```
# Color for alternate and highlighted rows in list views
#gui.alternate.rows.color.rgb = 230,245,240
#gui.highlight.rows.color.rgb = 240,200,200
#gui.highlight.rows.off = 1
```

The three numbers from 0 to 255 can be used to select any colour in RGB code. The background colour for the selected row in the list cannot be changed at this time.

Highlighting cells in lines currently only applies to partner profiles without retry pattern configured. If you do not want these entries to be highlighted, you can disable this by setting `gui.hightlight.rows.off` to 1.

### 10.2.21 GUI main window size

You can adjust the initial size of the GUI main window to your needs.

```
# Size of e-AS2 window
gui.main.win.width = 750
gui.main.win.height = 700
```

The values for the initial window width and window height are specified in pixel. The values in the commented out pattern entries correspond to the internal preset defaults.

### 10.2.22 ToolTip settings

By default ToolTips appear in the GUI after 5 seconds and they then stay for 5 seconds. The timing settings for ToolTips can be changed.

```
# Tooltip settings
gui.tooltip.delay = 5
gui.tooltip.duration = 5
```

### 10.2.23 Controlling provider related processing logic

Version 8.3 of e-AS2 Connect introduced a new type of processing logic which is based on provider distinction. There can be multiple partner profiles in the system, each of them representing a separate provider or a separate AS2 communication endpoint of a given provider.

Whether or not the software uses the new logic, can be controlled with the following settings.

```
# Version of interface logic for directory interface,
```

## Configuration for experts

```
# only relevant for e-AS2 Connect, supported values:
# 1 - single provider, exactly one partner profile, directory defines AS2-To
# 2 - full multi-provider, multiple partner profiles, directory identifies partner
#     profile, sub-directory defines AS2-To
# 3 - restricted multi-provider, multiple partner profiles, directory identifies
#     partner profile and defines AS2-To
#interface.out.logic.version = 1
#interface.in.logic.version = 1
```

Processing logic version 1 stands for the well known, traditional processing logic of e-AS2 Connect. There is exactly one partner profile in the system, representing the provider. The real communication partners, where messages are to be routed to, are identified by sub-directories below `interface/out` and `interface/in`.

Processing logic version 2, full multi-provider mode, introduces an additional directory level, identifying multiple different providers or provider end points. The sub-directories directly under `interface/out` and `interface/in` are matching the partner profiles that are visible in the GUI. The real communication partners, where messages are to be routed to, are identified by sub-directories below these provider directories.

Processing logic version 3, restricted multi-provider mode, works like full multi-provider mode by supporting multiple provider profiles and corresponding provider directories in the file system. But it comes without nesting directories for routing purposes. There's just one directory per provider. The payload files are placed in these directories. On the provider's systems routing must be done based on message content.

In logic and out logic can be controlled separately with the corresponding properties. However, setting in logic version to 2 or 3 while out logic version is 1 doesn't make sense and is not supported.

Find more information about the multi-provider processing logic in Chapter 7 (*Multi-provider logic configuration*).

Note, that you need an e-AS2 Connect license, that is generally multi-provider enabled, in order for these properties to take effect. At the time of this writing this is only the case for installations connected to e-integration.

**Configuration for experts**

# 11 Reference

*Systematic documentation of all fields and functions in the graphical user interface.*

## 11.1 Partner details, fields

The following documents the individual fields in the partner profile. The data type and a brief text description is provided for each field.

Data type *Data type*: alphanumeric contains all letters (case sensitive) and numbers as well as special characters. (We recommend avoiding special characters, if possible.)

Data type *Data type*: numeric only contains numbers.

With data type *Data type*: list a value is selected from a specified list.

Data type *Data type*: boolean describes a Yes/No statement.

Please be aware of the data types when creating the new profile or editing existing profiles!

### 11.1.1 AS2-From

Data type: *alphanumeric*, max. 256 characters.

From the perspective of the party sending data this is the AS2From component in the AS2 relation (AS2From, AS2To). I.e. when sending data this value is entered in the header of the MIME packet as AS2From.

When receiving data the perspective is reversed. Then, this value is expected as the AS2To in the header of the MIME packet received. Partner profiles are assigned accordingly.

### 11.1.2 AS2-To

Data type: *alphanumeric*, max. 256 characters.

From the perspective of the party sending data, this is the AS2To component in the AS2 relation (AS2From, AS2To). I.e. when sending data this value is entered in the header of the MIME packet as AS2To.

When receiving data the perspective is reversed. Then, this value is expected as the AS2From in the header of the MIME packet received. Partner profiles are assigned accordingly.

In e-AS2 Connect, AS2-To cannot be configured directly in the graphical interface. Instead it ensues from the name of the out folder for the respective partner in the interface area.

### 11.1.3 Server

Data type: *alphanumeric*, max. 128 characters

This is the IP address of the computer your provider's AS2 server is running on. This field is initialised with the correct value during installation of e-AS2 Connect and normally does not need to be changed.

### 11.1.4 Port

Data type: *numeric*, positive, max. 5 characters

This is the port number used to address your provider's AS2 server. In the case of e-integration you can reach the test server under 5088 and the production server under 5080. In the event a different value needs to be entered here, your provider will notify you.

This field is initialised accordingly during installation, in the case of e-integration with the port number of the test server. Do not change this value until all tests have been completed, and e-integration asks you to switch to the production server. e-integration will never productively process data sent to the test server and forward it to your business partner.<sup>43</sup>

### 11.1.5 URI

Data type: *alphanumeric*, max. 256 characters

This is the post request URI used to establish the connection to your provider. Do not change the value that was automatically set during installation unless you're asked to do so by your provider.

### 11.1.6 Retries

Data type: *alphanumeric*, max. 64 characters

Here, enter the retry pattern for failed transfers. You can find detailed explanations about the e-AS2 Connect repeat system in Section 6.12 (*Setting a retry pattern*).

### 11.1.7 Ack mode

Data type: *list*

The Ack mode is used to determine whether to request an MDN (Message Disposition Notification) for sent file. Set the Ack mode to "synchronous" to request an MDN. Set the Ack mode to "no" to not request an MDN. In this case only an HTTP acknowledgement will be expected back from the remote system.

### 11.1.8 Description

Data type: *alphanumeric*, max. 512 characters

This is a text field with a descriptive comment about the partner profile. In e-AS2 Connect the text in this field is fixed and cannot be edited.

### 11.1.9 sign

Data type: *boolean*

Tick "sign" to sign the messages you are sending. Your own certificate will be used for signing. This is always listed first under the certificate tab in the certificate list. You can combine signature and encryption.

### 11.1.10 encrypt

Data type: *boolean*

Tick "encrypt" to encrypt the messages you are sending. Your provider's certificate will be used for encryption. This is always listed second under the certificate tab in the certificate list. You can combine encryption and signature.

## 11.2 Partner list, functions

Buttons in the right area of the main window provide some partner-related functions documented below.

---

<sup>43</sup>If you're connected to a different provider, the differentiation between test and production server may be resolved differently.

### 11.2.1 Add partner

This function is blocked in e-AS2 Connect. You can not add any other partner profiles other than the profile for accessing your provider.

### 11.2.2 Delete partner

This function is blocked in e-AS2 Connect. You cannot delete the profile for accessing your provider.

### 11.2.3 Edit partner

Select the partner profile from the list and press this button to edit the data for the profile. Please coordinate any changes to the communication parameters with your provider.

### 11.2.4 Fetch data

Pressing this button will initiate fetching data from your provider. Please refer to Chapter 8 (*Data fetch extension*) for information on the e-AS2 data fetch extension.

### 11.2.5 Send certificate

Once you have generated your own certificate, the public key must be transmitted to your provider to be able to use the associated cryptography functions. To do so, select your new certificate in certificate management. This is the certificate with the alias `_eas2_`. Then select the partner profile from the partner list and press the “Send certificate” button.

Please note, this function for communicating with your provider uses the same parameters from the partner profile also used for regular data communication. Before attempting the key exchange you should first transmit test data (without cryptography) to ensure general communication works.

### 11.2.6 Test connection

This allows you to quickly and easily test the connection to your provider.

Select the partner profile from the partner list and press the “Test connection” button. A connection to your provider will now be established. If successful, internally generated test data will be transferred. Almost all aspects of this transfer correspond to a transfer triggered by adding a file to the interface area with the following exceptions:

1. You will not need access to the server file system. The data to be transferred is auto-generated.
2. The transfer to your provider will be executed immediately. The client will wait for the response from the receiving system and show the result of the transaction, particularly any error messages, in a message box.
3. A retry pattern in the partner profile will be ignored. In the event of an error, generally no repeat is executed. The respective transactions are visible in the transaction list, but in the event of an error they cannot be released for retransmission.

### 11.2.7 Close application

This will close the GUI client.

## 11.3 Certificate details

The graphical interface offers a selection of the primary certificate properties for the installed certificates to display for the user to review.

### 11.3.1 Subject

This field shows the owner of the certificate. The information consists of a series of attributes which taken together clearly identify the owner and distinguish it from others. Each attribute is structured as *key=value* and the various attributes are comma separated. Typical attributes identifying a certificate owner are his name or his email address. Depending on the origin of the certificate, the information in this field may appear very different.

The certificates generated by e-AS2 Connect have a very simple structure. They consist of the attributes CN (Common Name), O (Organization), EMAILADDRESS and C (Country).

### 11.3.2 Issuer

This field shows the issuer of the certificate, using the same syntax as the owner field. The issuer is often identical with the owner, if he issued a certificate himself. Certificates generated with e-AS2 Connect also have this property. This process creates a so-called self-signed certificate, where the owner and issuer are identical.

### 11.3.3 Serial no.

This shows the serial number of the certificate. If two certificates with identical attributes are produced accidentally or intentionally, they will always differ in their serial numbers. The serial number is displayed as a decimal number, followed by the identical value in hexadecimal format in brackets.

### 11.3.4 Valid from

Beginning of the period the certificate is valid for.

### 11.3.5 Valid to

End of the period the certificate is valid for.

## 11.4 Certificate list, functions

The buttons at the right side of the main window offer cryptography-related functions as documented below.

The certificate list consists of three columns as follows:

- Alias The alias for the respective certificate.
- CN The Common Name of the certificate owner as entered in the certificate.
- Priv This is ticked if the certificate contains a private key.

### 11.4.1 Update public key

This button becomes active after selecting your AS2 provider's public key certificate. If you receive an updated certificate from your provider, you can use this button to import it. Please note, the new certificate will become active and be used immediately. The existing certificate will be overwritten and cannot be restored in e-AS2 Connect.

### 11.4.2 Generate private key

Before using e-AS2 Connect with cryptography routine for the first time, you need to generate a private key certificate for your e-AS2 entity. If you press the "Generate private key" button, will open a dialogue prompting the following information:

- Name Enter the name of a person responsible for this process. This is typically your name.

## Reference

Organisation	Enter the name of your company.
Country	Enter the ISO country code, so typically “DE” for Germany.
E-mail address	Enter your mail address. This should be an address where the person whose name you entered above can be reached.
Key length	There are three key lengths to choose from. Currently a 2048 bit key length is recommended and will be automatically pre-selected in the drop down list. If you need higher security – at the expense of performance – select 4096 bit. If performance is a major goal you can reduce key length to 1024 bit. <sup>44</sup>
Valid from	Select the date as of when you want the certificate to be valid. Generating certificates with a future validity date is permitted, although this usually doesn’t make much sense.
Valid until	Select the date the certificate will be valid until. Once this date expires the certificate generated can no longer be used. Do not select too short a period. The default is ten years.

After pressing the “Save” button the certificate will be generated with your specifications and then transmitted to the e-AS2 server, where it is added to certificate management. It replaces your e-AS2 server’s existing private key certificate. You can then transmit the related public key certificate to your provider. Do so using AS2 key exchange as described in Section 6.6 (*Preparing for encryption and signature*).

### 11.4.3 Close application

This will close the GUI client.

## 11.5 Transaction details

The transaction details provide detailed information on the status of the respective transaction. The transaction results in the GUI are generally read-only. The transaction results cannot be freely edited at field level. Using the buttons “Release transaction”, “Stop transaction” and “Release MDN” however, will trigger specific changes which will then also immediately be reflected in the transaction details.

The individual fields of the transaction details will be populated differently depending on the processing direction. The following sections are therefore split – where necessary – to reflect this difference.

### 11.5.1 Partner ID

The internal ID of the partner (resp. this AS2 Relation in general).

#### Send

The ID will be copied from the partner profile.

#### Receive

The software will attempt to find a matching partner profile based on the AS2 relation. If it was found, the respective partner ID will be copied to the transaction details. Otherwise the partner ID field will remain blank.

---

<sup>44</sup>Please note, generating a certificate with with bigger key length may take significantly longer than for smaller key lengths. The certificate is generated by the GUI Client. You can particularly expect a respective waiting time when running the GUI on a slow computer.

### 11.5.2 AS2-From

The AS2-From component for the AS2 relation behind this transaction.

#### Send

This information is copied from the AS2-From field in the partner profile.

#### Receive

On receipt this information is copied from the data received, and the AS2-To component interpreted due to the reversed direction of transmission.

### 11.5.3 AS2-To

The AS2-To component for the AS2 relation behind this transaction.

#### Send

The name of the partner directory directly under `interface/out` is copied for this information.

#### Receive

At the time of receipt this information is taken from the data received, and the AS2-From component interpreted due to the reversed direction of transmission..

### 11.5.4 Server

The IP address of the remote endpoint.

#### Send

This information is copied from the “Server” field in the partner profile.

#### Receive

Normally this is the IP address from which the incoming connection for receiving data *actually* originated. This IP address does not necessarily have to match the content of the field in the partner profile.

If the remote system requested an asynchronous MDN, the request contained the full communication address for returning the MDN. In this case, in place of the sender’s actual IP address the IP address or the server name from the MDN request will be copied to this field.

### 11.5.5 Port

The port number under which the remote endpoint can be reached when sending data.

#### Send

This information is copied from the “Port” field in the partner profile.

#### Receive

This is normally 0, since this port number is irrelevant when receiving data.

If the remote system requested an asynchronous MDN, the request contained the full communication address for returning the MDN. In this case the port number is copied to this field from the MDN request.

### 11.5.6 URI

The post request URI the remote endpoint can be reached under when sending data.

### **Send**

This information is copied from the “URI” field in the partner profile.

### **Receive**

This field normally remains blank, since this URI is irrelevant when receiving data.

If the remote system requested an asynchronous MDN, the request contained the full communication address for returning the MDN. In this case the post request URI is copied to this field from the MDN request.

## **11.5.7 Status**

The status of this transaction.

### **Send**

A send transaction normally runs through the following status values:

- new – initial status immediately after the file is detected in the file system.
- sent – data transfer to the remote system is complete.
- confirmed – an HTTP acknowledgement has been received.
- MDN received – an MDN has been received (optional).

In addition, in some cases the following other status values may appear:

- released – erroneous transaction was released for reprocessing.
- stopped – incomplete transaction was stopped prematurely.

### **Receive**

A receive transaction runs through the following status values:

- new – initial status immediately after transfer of an incoming file begins.
- received – data transfer from the remote system complete.
- confirmed – an HTTP acknowledgement was sent.
- MDN pending – an MDN is pending for send (optional).
- MDN sent – an MDN was sent (optional).

In addition, in some cases the following other status values may appear:

- MDN released – erroneous transaction was released for reprocessing an MDN.
- stopped – incomplete transaction was stopped prematurely.
- Re-parse – erroneous transaction was released for re-parsing a received (orphaned) message.

## **11.5.8 Direction**

This field contains a code for the data flow direction.

### **Send**

The field direction contains an S for send.

**Receive**

The field direction contains an R for receive.

**11.5.9 In Process**

“In Process” is ticked if one of the processing threads in e-AS2 Connect is currently processing this transaction. So processing only refers to the acute processing of a pending partial step, not the overall processing of the transaction. I.e. a transaction could be “In Process” several times before being completed.

**11.5.10 Done**

Unlike “In Process” (see there) a tick here refers to the overall process. If e.g. communication problems occur resulting in several retries for a transaction, it will only be marked complete after all retries have been exhausted or an attempt was successful.

**11.5.11 Filename**

The physical name of the transferred file in the file system of the e-AS2 server.

**Send**

The complete path in the file system is listed as the file name. During retries after failed attempts this information will be used to retrieve the file to be transmitted.

**Receive**

e-AS2 Connect will save received files to the `in` folder of the respective partner. The files are saved under the name (followed by message ID)<sup>45</sup> the sender assigned to the file. Based on limitations forming file names in the file system, certain characters are not allowed. e-AS2 Connect recognises and automatically filters out the following problematic characters:

\ / ? : \* " < > |

e-AS2 Connect will replace any of these characters in the name transmitted by the sender with an underscore (`_`) before saving the file to the file system under the resulting name. So the actual file name may not actually completely match the name shown in the transaction details.

**11.5.12 AS2 name**

The virtual name of the transmitted file.

**Send**

Only the file name itself (without path) is entered here.

**Receive**

This shows the file name as transmitted by the sender. This is a logical name of the file which always is different from the physical file name which appears in the file name field.

**11.5.13 Message ID**

AS2 provides for each file transferred (in form of a MIME message) to be assigned a globally unique message ID. Among other things, the message ID is used to recognise duplicate transfers, thus prevent duplicate processing.

<sup>45</sup>The structure of the file name for files received can be changed with a setting in `EAS2.properties`. Please see Section 10.2.14 (*Building file names for incoming messages*).

### Send

e-AS2 Connect generates message IDs with, in addition to text constants, the exact time of the transaction (to the millisecond), IP address and server name plus a global sequential number. This ensure uniqueness. The generated message ID is saved in the transaction record in this field.

### Receive

This is the message ID the sender assigned the transmitted file. The sender is responsible for global uniqueness. On receipt, e-AS2 checks whether a message with the same ID already exists. If so, receiving the same message again will be rejected.

#### 11.5.14 Size

This field shows the length of the transmitted file .

#### 11.5.15 Created

The time when the transaction record was added to the database.

#### 11.5.16 Changed

The time when the transaction record was last changed in the database. So this corresponds with the last status change for this transaction.

#### 11.5.17 MIC

MIC stands for “Message Integrity Check”. To allow the other party to check the integrity of a message received, the sender forms a so-called “Message Digest”. In e-AS2 Connect the algorithm SHA1<sup>46</sup> is used to calculate the digest. The name of the digest algorithm is transmitted along with the message itself. If the message or the digest is changed in transit (intentionally or through malicious attack), the digest will no longer match the message. The recipient can identify the discrepancy and respond accordingly.<sup>47</sup>

### Send

e-AS2 Connect writes the computed message digest to this field in the transaction record.

### Receive

This is the message digest transmitted by the sender along with the file. Transmitting a message digest is optional. If the sender did not transmit it, the field MIC will contain the text “none”.

#### 11.5.18 Algorithm

This field contains the identifier of the algorithm used for computing the message digest. e-AS2 Connect uses SHA1 when sending data, but can handle other algorithms too when receiving data.

#### 11.5.19 Error

This is ticked if an error occurred processing this transaction. The contents of the “Description” (Section 11.5.24 (*Description*)) or the e-AS2 log file can help in diagnosing the cause for the error.

---

<sup>46</sup>US Secure Hash Algorithm 1 (SHA1), see <http://www.ietf.org/rfc/rfc3174.txt>

<sup>47</sup>Note, that when message encryption and/or signature is activated, the MIC check on AS2 protocol level is highly redundant. Therefore, to use a relatively weak digest algorithm like SHA1 at this place does in fact pose no security risk.

### 11.5.20 Sign MDN

This field is ticked if the sender of a received message requested a signed MDN. It is only used for incoming transactions.

### 11.5.21 MDN error

This is ticked if an error occurs while sending an asynchronous MDN for this transaction. The contents of the "Description" (Section 11.5.24 (*Description*)) or the e-AS2 log file can help in diagnosing the cause for the error.

The field is only used for incoming transactions.

### 11.5.22 MDN done

This is ticked after sending an asynchronous MDN for an incoming transaction was completed successfully.

The field is only used for incoming transactions.

### 11.5.23 Retries

This field only applies to send transactions. It contains the number of retries so far for failed transfers, followed by the configured retry pattern in parentheses.

### 11.5.24 Description

Error messages and other additional comments about the transaction are saved to this field. This particularly also contains error messages reported back from the remote system.

Messages originating from the local system have the prefix `LOCAL :`. Messages copied from an incoming MDN have the prefix `REMOTE :`.

## 11.6 Transaction list, functions

Buttons at the right in the main window provide some transaction-related functions documented below.

### 11.6.1 Select transaction

Select a transaction from the list to view the details in the bottom part of the GUI. Selecting an entry will load the current transaction details from the server. If your connection is slow, you may notice a brief delay before updated data is displayed. This is normally not a reason for concern. When a transaction is being selected, the status of the buttons at the right automatically changes accordingly so only those buttons which can be used in the specific context are active.

### 11.6.2 Only show errors or open entries

Tick this box to only list transactions which are not yet complete. This will then show all error and pending transactions. The second group contains e.g. all new transactions or those with retries pending. Untick the box to show all transactions for the selected date again.

### 11.6.3 Re-read list

This button will re-import the current status of the transaction list from the server.

If you have a longer session using the GUI, the transaction list may not be current anymore after some time. The server continues working regardless of the GUI session and new transaction entries are generated in the database which are not automatically propagated to the client. Re-

import the list for an update! If you configured a filter of results for the search dialogue, “Re-read list” will only find the new transactions matching these criteria.

To minimise data traffic between the GUI client and e-AS2 server, re-importing the list will only fetch those transactions from the server currently visible in the section of the list. Further all transactions which have been added since the last fetch. You can change this behaviour by ticking “all”.

#### 11.6.4 all

This should normally always be unchecked. When pressing the “Re-read list” button, e-AS2 will then only fetch a minimum of transactions from the server required for updating the display in the GUI. If for some reason you find it useful or necessary to re-read the complete transaction list, tick “all”.

Please note, a complete fetch of large amounts of transactions will take significantly longer than the efficient partial fetch. To ensure the complete flag does not accidentally remain active, it is automatically turn off after the complete fetch.

#### 11.6.5 Release transaction

This function only applies to send transactions. If e-AS2 fails to transfer a file to the desired recipient after exhausting all retries, the transaction is set to error status. It will appear in read in the list.

The file has not yet been sent and is still available. It was moved to the `_errors_` directory. Once you have determined and resolved the cause for the error, you now want this file to be sent. Release this transaction to do so.

Once released, it may take up to one minute for e-AS2 to detect the transaction again. The file to be transmitted is retrieved from the `_errors_` directory and now transmitted to the recipient. Should there still be problems transmitting the data, then the configured retry pattern applies again, i.e. will be started from the beginning.

#### 11.6.6 Stop transaction

If you, as the operator, notice e-AS2 is still unable to transmit a file during retries, you can stop the process prematurely. It may take up to a minute after stopping for e-AS2 to set the transaction to status “stopped” and “done”.

Send transactions may be stopped so long as the file has not yet been successfully transmitted to the other party and acknowledged.

Under certain circumstances receive transactions can be stopped also. This is the case e.g. with transactions which could not be processed completely after receiving the MIME message.<sup>48</sup> Such transactions purposely remain in open status as the responsible operator must clarify the situation. Once clarified the transaction should be stopped.

#### 11.6.7 Release MDN

If outgoing asynchronous MDNs could not be transmitted after exhausting all retries, the MDN error flag is set for this transaction.

The MDN file has not been sent and is still available. It was moved to the `out_mdn/_errors_` directory. Once you have determined and resolved the cause for the error, you may want to resume trying to send this file. To do so, release the transaction for MDN re-transmission.

---

<sup>48</sup>One typical cause is cryptography problems due to the certificates not matching.

Once released, it may take up to one minute for e-AS2 to detect the transaction again. The MDN file to be sent is retrieved from the `out_mdn/_errors_` directory and now transmitted to the recipient. If problems should happen to occur again, the configured retry pattern applies.

If sending the MDN continues to fail, the process can be cancelled by stopping the transaction as described in the previous section.

### 11.6.8 Close application

This will close the GUI client.

## 11.7 Administration

The administrative information is shown on the left, split into a server section and a GUI section. This allows e.g. comparing the installed software/protocol versions of the server and GUI, which may be useful if server and GUI are running on different computers. Also detailed license information is displayed.

### 11.7.1 Server info

The server info box contains the following information:

Instance name	This is the name assigned the e-AS2 server in <code>EAS2.properties</code> .
IP address	This is the IP address the e-AS2 server as detected on startup. This information is particularly of interest for systems with several network interfaces, as this IP address is used for license validation (see below).
Configuration port	This is the number of the port the e-AS2 server listens to for incoming configuration connections.
Program version	The version number of the e-AS2 server. You can compare this with the version number of the GUI client to determine any discrepancies. The program versions being different does not automatically mean GUI and server being incompatible. Compatibility depends on the CS protocol version and DB schema version.  Generally GUI installations will work with the same or older versions of the server as long as the major release number (currently: 8) is identical.
Program revision	The source code revision number for the e-AS2 server. This information can normally be ignored. However, it may be of interest when contacting e-integration support. You should provide it for support requests.
CS protocol version	The version of the client/server protocol. This defines the overall scope of functions the GUI client can get from the e-AS2 server. The client protocol version must be compatible with server version. When attempting to access a non-compatible server with your GUI client, you will clearly be notified of the problem.
DB schema version	The version of the database schema on which the present e-AS2 version is based. With respect to compatibility between

	the client and server, the information under CS protocol version applies.
License type	This shows the type of e-AS2 license (ENTERPRISE or CONNECT). If you see CONNECT despite having an ENTERPRISE license, you probably installed your license incorrectly.
License valid until	For time limited licenses this shows the expiration date.
License holder	The company the license was issued to. This is not used in e-AS2 Connect.
Licensed for IP address	The IP address the license was issued to. If there are problems, please compare it with the actual system IP address shown in the first field.
License for port number	The port number the license was issued for. This refers to the port number for incoming HTTP data connections.
License partner limit	The maximum number of partner profiles you can set up based on the license issued to you.
License for HTTP	This indicates whether you licensed the e-AS2 HTTP Feature. This is required for sending and receiving AS2 messages.
License for SMTP	This indicates whether you licensed the e-AS2 SMTP Feature. This is required for sending and receiving EDI messages via e-mail. The absence of this license does not affect operator e-mails, which will be sent in any case.

### 11.7.2 GUI Info

The GUI Info box shows the following information:

Client Name	This is the name assigned the GUI instance in the client's <code>EAS2.properties</code> file. This is followed by the operating system user signed in, displayed in parentheses. If no GUI client name was configured, this will only show the operating system user.
IP address	This is the IP address of the system the GUI client was started on.
Program version	The program version of the GUI client. When starting the server and client from the same installation on the same system, this version number will always be identical to that of the server. When running the GUI client on a different computer, it may differ. (Also see the explanation under Server Info.)
Program revision	The source code revision of the GUI client. The same as for the program version applies correspondingly.
CS protocol version	The version of the client/server protocol. Please read the explanation under Server Info on this topic!
DB schema version	The version of the database schema. Please read the explanation under Server Info on this topic!

## 11.8 Administration, functions

The buttons at the right of the main window provide some administrative functions documented below.

### 11.8.1 Manage Server

Use this button to open the server management dialogue. Here you will find detailed information on the database system used. It also contains various information on the status of the server process.

The current status of the server process is updated every time the server management dialogue is opened. This essentially leads to a data record consisting of the following five fields:

- Number of threads currently active in the system. The higher this number, the higher the level of parallel processing by the server. This is a good way to measure system utilisation.
- Number GUI clients currently signed in.
- Current free JVM<sup>49</sup> memory.
- Total memory currently reserved by the JVM.
- Memory currently actually used by application objects.

You can monitor the current trend of this data record by opening the server management dialogue several times in intervals. It always shows the last three records. The latest is in the left column.

At the bottom of the server management dialogues you can select two independent actions:

- Shut down server
- DB compactification<sup>50</sup>

If you selected at least one of these actions, the “Execute” button will become active. Confirm this action instead of “Close” to execute the selected actions.

### 11.8.2 Close application

This will close the GUI client.

---

<sup>49</sup>Java Virtual Machine

<sup>50</sup>This action is only available when using the HyperSQL database. This will reduce the database files in the file system during operation. This implicitly also applies when shutting down the server.