ESKER® e-integration | EDI SERVICES

e-AS2 Enterprise

User manual for Version 9.0

Esker EDI Services

e-AS2 Enterprise

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	4/28/2015	JK	Added description of auto-disable features.
2.2	5/12/2015	JK	Updated description of auto-disable features and about manually disabling an enabling partner profiles.
2.3	7/31/2015	JK	Added documentation of dry-run feature.
2.4	03/11/2016	JK	Added documentation of new property settings and new result file content.
2.5	19.08.2016	JK	Updated manual for e-AS2 7.2. Added documentation of GUI Launcher.
2.6	15.11.2016	JK	Added cluster edition documentation.
2.7	12.01.2018	JK	Added documentation on REST-API.
2.8	19.04.2018	JK	Updated manual for e-AS2 7.6.
2.9	16.08.2018	JK	Documented new option for setting the MIC algorithm.
2.10	04.03.2019	JK	Added documentation on audit log functionality and re-parsing of received messages.
2.11	04.07.2019	JK	Updated manual for e-AS2 7.10.
2.12	27.01.2020	JK	Updated manual for e-AS2 7.11.
3.0	02.07.2020	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	05.02.2021	JK	Updated manual for e-AS2 8.1. Introduced limited error handling for send transactions.
3.2	15.04.2021	JK	Updated manual for e-AS2 8.2. Documented updated test transmission and dry-run functionality in GUI.
3.3	07.12.2021	JK	Updated manual for e-AS2 8.4. Only Cluster Edition is affected by changes. An option to generate Tracking and Tracing events was added.
3.4	04.01.2022	JK	Updated manual for e-AS2 8.5. Added information on copy partner profile functionality.
3.5	11.01.2022	JK	Updated manual for e-AS2 8.6. Removed references to the JCE policy topic. Added copy-to-remote for certificates.

3.6	27.01.2022	JK	Updated manual for e-AS2 8.7. Added information about new Content ID attribute in result file.
3.7	01.03.2022	JK	Updated manual for e-AS2 8.8. Updated information on logging configuration.
3.8	06.09.2022	JK	Updated manual for e-AS2 8.10.
3.9	13.10.2022	JK	Updated manual for e-AS2 8.11.
4.0	06.02.2023	JK	Updated manual for e-AS2 9.0.
4.1	25.07.2023	MS	Document additional REST calls.

Table of Contents

1 Introduction

What is e-AS2 Enterprise?

========== ADDED (JK: 06.02.2023) ==========

This special edition of document version 4.0 highlights the key changes compared to version 3.9 in colour.

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

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. All AS2 protocol features can be actively addressed. Any number of relationships may be configured. In addition to AS2, e-AS2 Enterprise also supports pure HTTP upload as well as SMTP[4] and POP3[5] for securely exchanging EDI messages via e-mail.

This document only addresses e-AS2 Enterprise. Separate documentation is available for the limited version (e-AS2 Connect). In addition, this only addresses the components of the software relevant to exchanging data via AS2 protocol and pure HTTP. The SMTP and POP3 functions are documented in a separate manual. First a brief overview of how this manual is structured.

Chapter 2 (*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 3 (*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 4 (*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 5 (*Configuring e-AS2 Enterprise*) describes the software configuration based on usage examples which build on each other. Each usage example requires a subset of available configuration options.

Chapter 6 (*Data fetch extension*) describes the configuration which allows remote e-AS2 Connect instances to use the data fetch extension.

Chapter 7 (*Interface for asynchronous MDN*) describes the methods for providing meaningful Message Disposition Notifications (MDN) with application information. Use of this interface is optional.

Chapter 8 (*The INF interface*) addresses an alternate output interface based on information files. This interface features some additional options compared to the basic interface for outgoing files. If at all possible it is recommended to use the INF interface.

[1]Electronic Data Interchange Internet Integration, see http://www.ietf.org/html.charters/ediint-charter.html
[2]MIME Based Secure Peer to Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2), see http://www.ietf.org/rfc/rfc4130.txt
[3]Hypertext Transfer Protocol, see http://www.ietf.org/rfc/rfc2616.txt
[4]Simple Mail Transfer Protocol, see http://www.ietf.org/rfc/rfc5321.txt
[5]Post Office Protocol - Version 3, see http://www.ietf.org/rfc/rfc1939.txt

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.

Lastly, in Chapter 12 (*Diagnosing and debugging*) we address various options for diagnostics and debugging in the event of an error.

Appendix A addresses exporting/importing e-AS2 configuration items.

In appendix B installation and operation of e-AS2 Cluster Edition is explained. This is a modified and extended version of e-AS2 Enterprise that is suitable for running in a cluster.

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.[6] This user manual is primarily based on an e-AS2 installation running on Mac OS X Catalina. 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.

[6]see http://www.oracle.com/technetwork/java/index.html

2 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.

2.1 General process

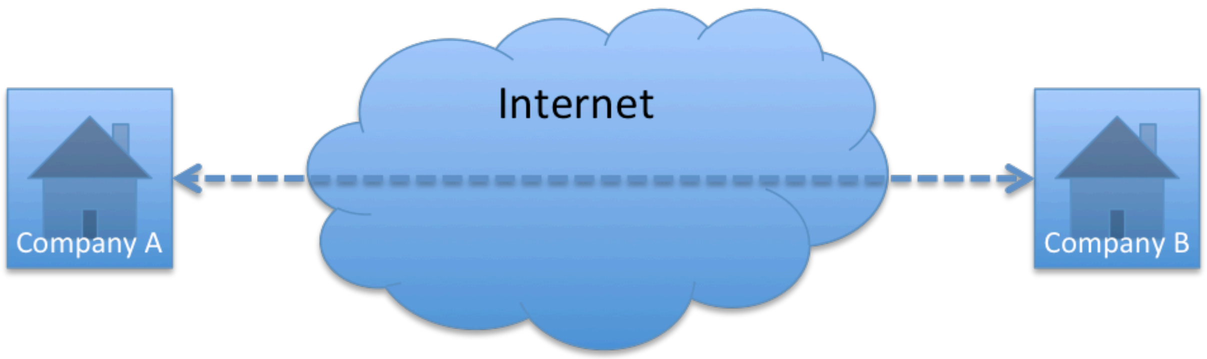


Figure 2.1. Data exchange between company A and company B

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

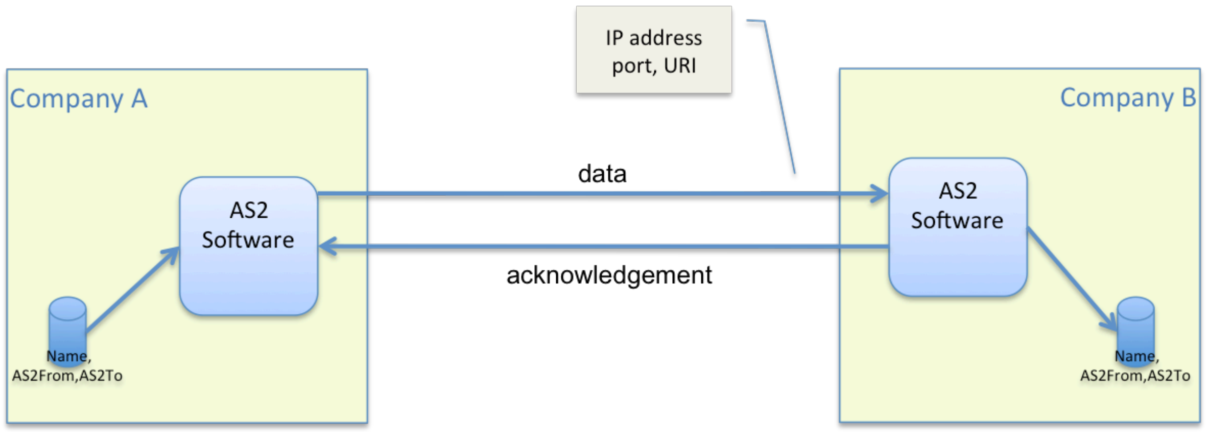


Figure 2.2. Data exchange via AS2

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[7] 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

[7]Multipurpose Internet Mail Extensions, see http://www.ietf.org/rfc/rfc2045.txt

B. Company B acknowledges during the same session, which is then ended. The AS2 software 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.

2.2 Establishing a connection

As specified in Figure 2.2, “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.

2.2.1 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`.[8]

2.2.2 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.

2.2.3 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[9].

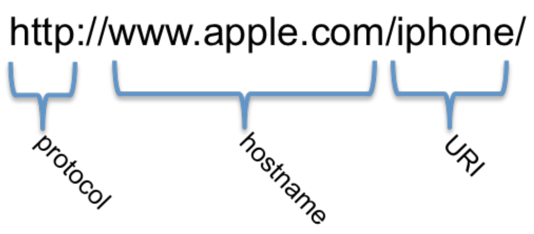


Figure 2.3. Post request URI, example

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

[8]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).
[9]The abbreviation URI means Uniform Resource Identifier.

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”.

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.

2.3 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.

2.4 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. Certificates can be exchanged by e-mail or on portable storage media. e-AS2 Enterprise, however, also supports direct certificate transfer between two AS2 servers.

2.5 Compression

As of protocol version 1.1, AS2 allows the integration of ZLIB algorithm-based compression[10]. The sender of the data decides whether he wishes to compress it before sending. The receiver

[10]Cf. http://www.ietf.org/rfc/rfc1950.txt (ZLIB Compressed Data Format Specification version 3.3)

end automatically recognises if the data received is compressed so a decompression step can be integrated into processing if necessary.

Compression in AS2 is a separate processing step before sending or after receiving. So it does not occur “on the fly” during the communication session.

2.6 Multi-file transmission

As of protocol version 1.2, the AS2 protocol supports transmitting several files during a single session. A MIME message in `multipart/related` format is constructed for sending the data. This option is explicitly not intended as an option for sending batches of several separate files to the same recipient. It should only be used if the sender has several files with related content which should preferably be received as one object.

Version 1.2 is implemented in e-AS2. Please check with the other communicating party if they support it before using protocol properties beyond protocol version 1.0. e-AS2 is always ready to receive multi-file transfers. We will address this in more detail where appropriate further down in the text.

3 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).

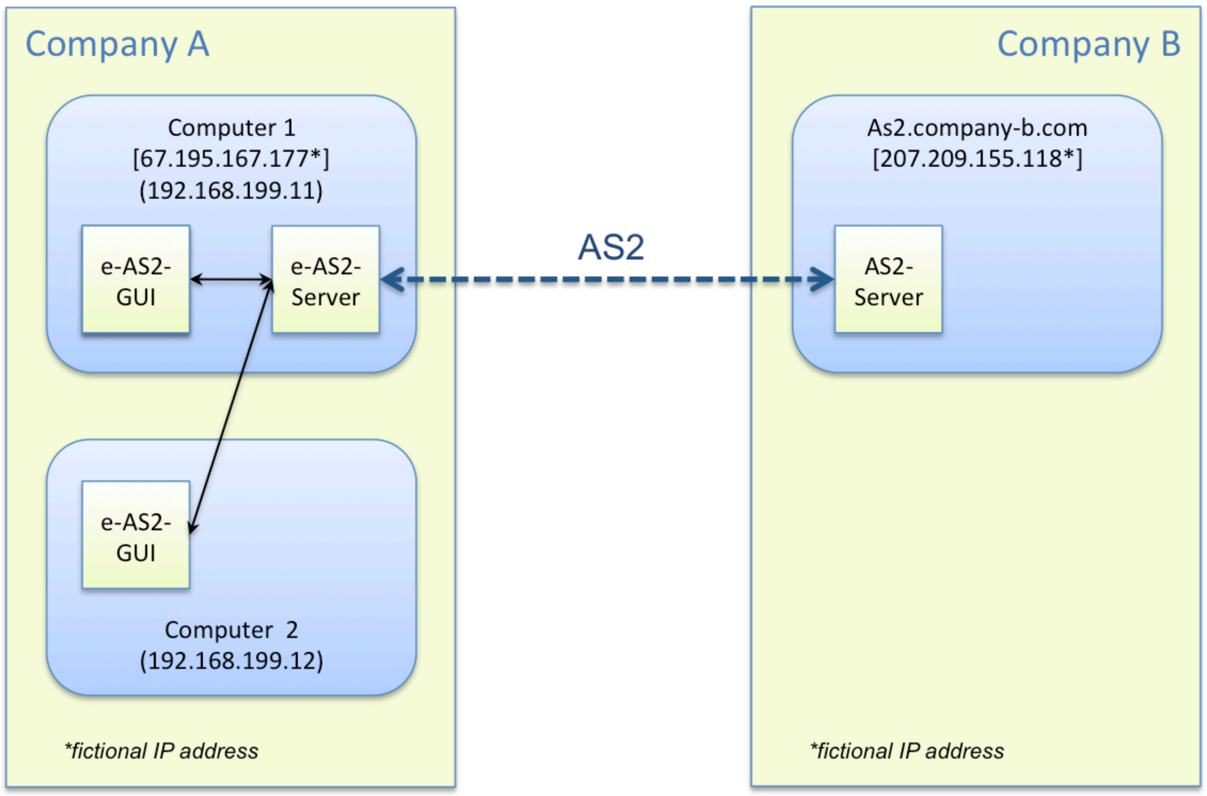


Figure 3.1. Schematic illustration of the e-AS2 architecture

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.

3.1 The e-AS2 server

The e-AS2 server is a Java program which uses multithreading[11] 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:

[11]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.

1. Directory Watcher Thread [DIRWATCHER[12]]
2. Database Watcher Thread [DBWATCHER]
3. Incoming Call Handler Thread for HTTP [ICH_HTTP]
4. Incoming Call Handler Thread for HTTP/S [ICH_HTTPS] (optional, up to four)
5. Incoming Call Handler Thread for SMTP [ICH_SMTP] (optional)
6. POP3 Mail Receiver Thread [POP3MR] (optional)
7. Configuration Server Thread [CFG]
8. Command Executor Thread [CMDEXEC]
9. Main Thread [main]

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

3.1.1 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[13] 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.

3.1.2 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,[14] 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.

3.1.3 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.

[12]Messages from the Directory Watcher Thread have this identifier in the log file.
[13]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 10.2.35 (*Preventing sending unfinished files*))
[14]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.

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.[15]

3.1.4 Incoming Call Handler Thread for HTTP/S

If HTTP/S was activated in the e-AS2 server configuration, up to four additional threads for handling incoming data connections will be started. A separate port is used for each of them, which only expects HTTP/S connections. Apart from that, these threads work the same as the Incoming Call Handler Thread for HTTP.[16]

3.1.5 Incoming Call Handler Thread for SMTP

If you purchased a license for the SMTP feature for e-AS2 Enterprise, an additional thread for handling incoming SMTP connections will be started. A separate port will then again be configured. This manual only documents the use of e-AS2 Enterprise for handling AS2 communication. If you also use the software for secure e-mail data exchange, please read the separate manual for e-AS2 SMTP.[17]

3.1.6 POP3 Mail Receiver Thread

If you purchased a license for the SMTP Feature for e-AS2 Enterprise, an additional thread will be started for retrieving mail boxes using the POP3 protocol. Retrieving mail boxes is an alternative to receiving e-mails per SMTP. This manual only documents the use of e-AS2 Enterprise for handling AS2 communication. If you also use the software for secure e-mail data exchange, please read the separate manual for e-AS2 SMTP.

3.1.7 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. In addition, if you have an SMTP license for e-AS2, the mail servers and the PGP keys will also be maintained via GUI. Configuration changes done by the GUI will be made persistent by the server, never the GUI itself.

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.

3.1.8 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*).

3.1.9 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.

[15] Please also note Chapter 4 (*Firewall and Proxy configuration*).
[16] Learn more about HTTP/S in e-AS2 in Section 5.19 (*Using HTTP/S*).
[17] Please note that operator e-mails can always be sent, regardless of the license for the SMTP Feature.

3.1.10 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.[18]

Open Service Manager

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

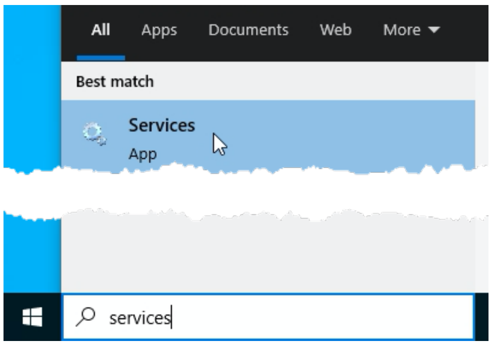
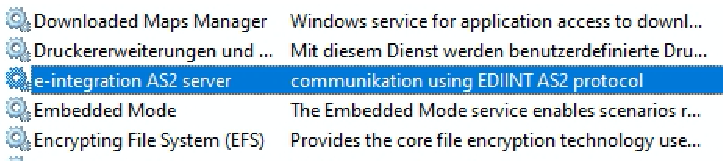


Figure 3.2. Open the Services app

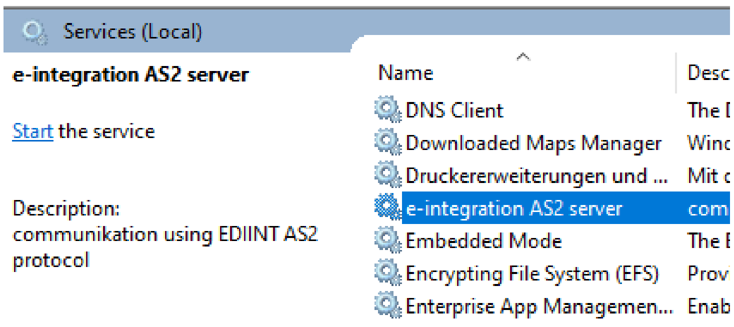
You will find the e-AS2 server in the list of services sorted in alphabetically:



Downloaded Maps Manager	Windows service for application access to downl...
Druckererweiterungen und ...	Mit diesem Dienst werden benutzerdefinierte Dru...
e-integration AS2 server	communikation using EDIINT AS2 protocol
Embedded Mode	The Embedded Mode service enables scenarios r...
Encrypting File System (EFS)	Provides the core file encryption technology use...

Figure 3.3. The e-AS2 server in the Services list

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



Services (Local)

e-integration AS2 server

Start the service

Description:
communikation using EDIINT AS2 protocol

Name	Desc
DNS Client	The I
Downloaded Maps Manager	Winc
Druckererweiterungen und ...	Mit c
e-integration AS2 server	com
Embedded Mode	The E
Encrypting File System (EFS)	Prov
Enterprise App Managemen...	Enab

Figure 3.4. 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

[18]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.

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.

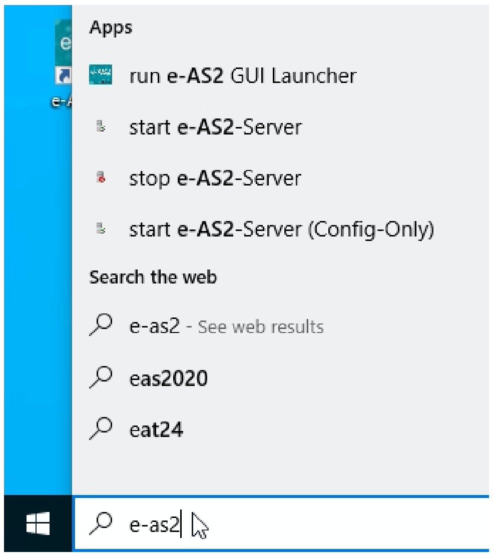


Figure 3.5. Starting and stopping the e-AS2 service

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 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.

3.1.11 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[19].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 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.

3.2 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/delete/add partner profiles
- View/delete/add cryptography certificates
- Set own cryptography parameters (generate certificates)
- Monitor transaction list
- Manually stopping transactions
- Scheduling erroneous transactions for re-processing
- Downloading transaction data to the client
- Send test data
- View GUI client and e-AS2 server version information
- Perform server management tasks and end server

If you purchased a license for the SMTP feature, you will also have the following items:[20]

- View/delete/add PGP keys

[19] 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.
[20] These items will not be addressed further in this manual, as they are documented in a separate e-AS2 SMTP manual.

- Generate new own PGP keys
- Configure mail servers for SMTP and POP3

First start the e-AS2 server as described in Section 3.1 (*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

```
./rungui
```

Mac OS X

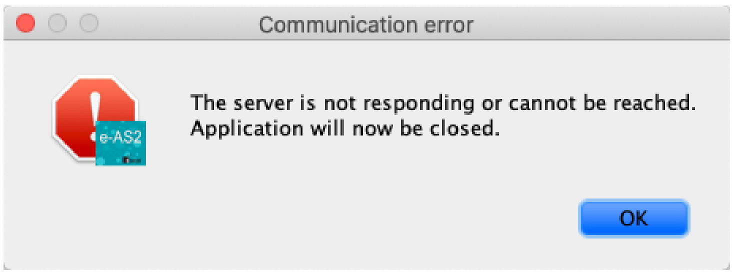
Start the graphical user interface by double-clicking the program

```
rungui
```

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

If you decided to place a shortcut on the Desktop during installation, you will have a convenient alternative for starting the GUI.[21]

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.

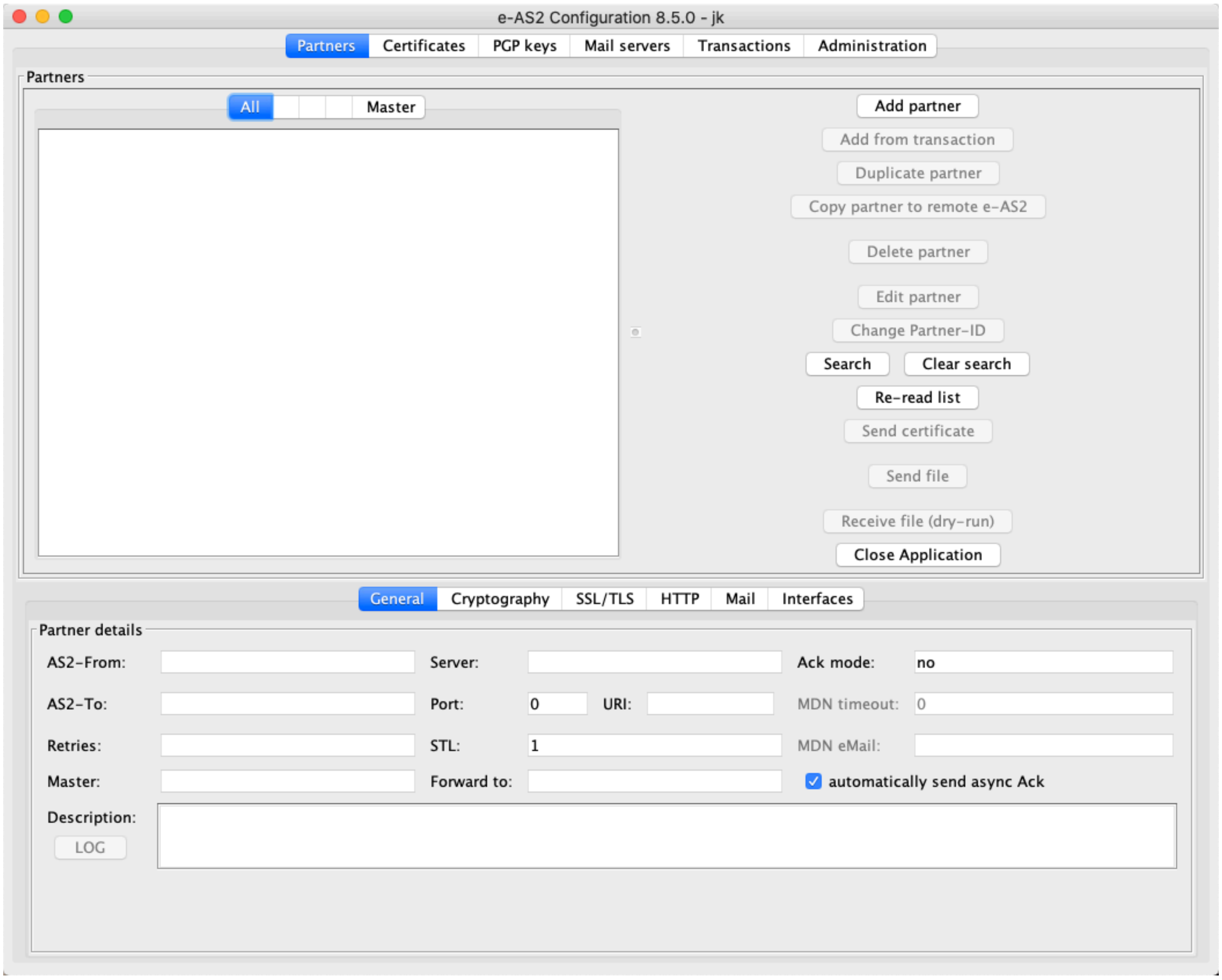

Communication error

The server is not responding or cannot be reached.
Application will now be closed.

Figure 3.6. e-AS2 server not available

Otherwise the user interface will start as follows.

[21]Desktop shortcuts are available in MS Windows and Mac OS X, as well as Linux when using KDE or Gnome.



AS2-From:
Server:
Ack mode: no
AS2-To:
Port: 0
URI:
MDN timeout: 0
Retries:
STL: 1
MDN eMail:
Master:
Forward to:
☑ automatically send async Ack
Description:

Figure 3.7. e-AS2 GUI

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

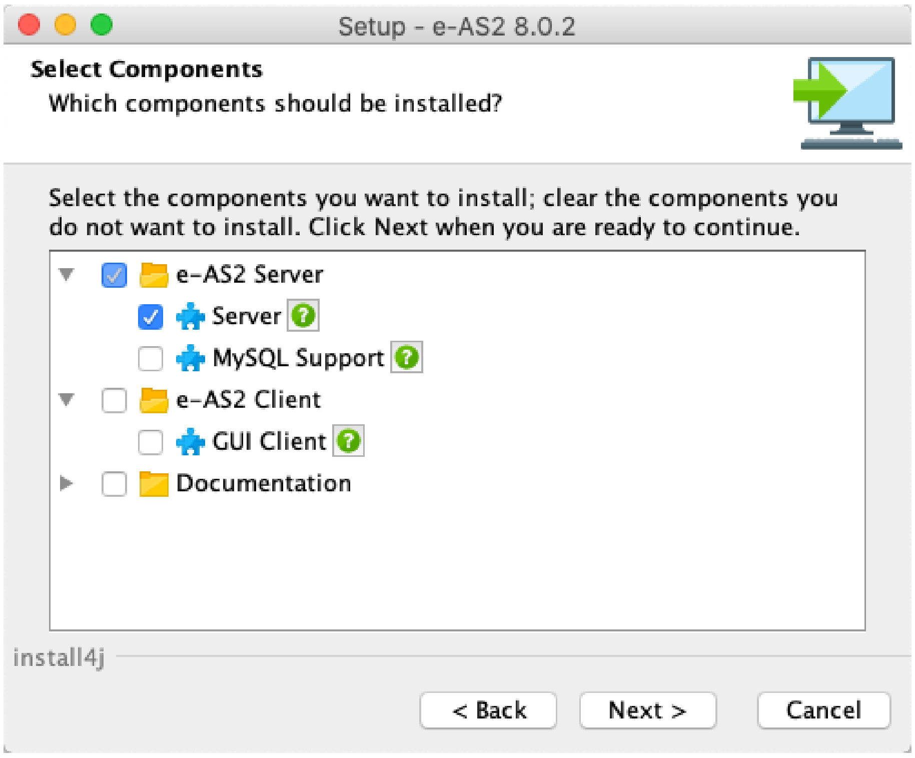
Note: Pressing the “Close Application” button will only end the GUI client, not the server. To end the server, use the Manage server button under the “Administration” tab or use methods presented under Section 3.1 (*The e-AS2 server*).

3.2.1 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 3.1, “Schematic illustration of the e-AS2 architecture”. First, e-AS2 Enterprise is installed on computer 1. In the process e.g. the following parameters are entered during setup:

Parameter	Value
Host name (data traffic)	computer1[22]
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 3.8, “e-AS2 installation - server only” and Figure 3.9, “e-AS2 server installation - parameters”.


Setup - e-AS2 8.0.2

Select Components

Which components should be installed?

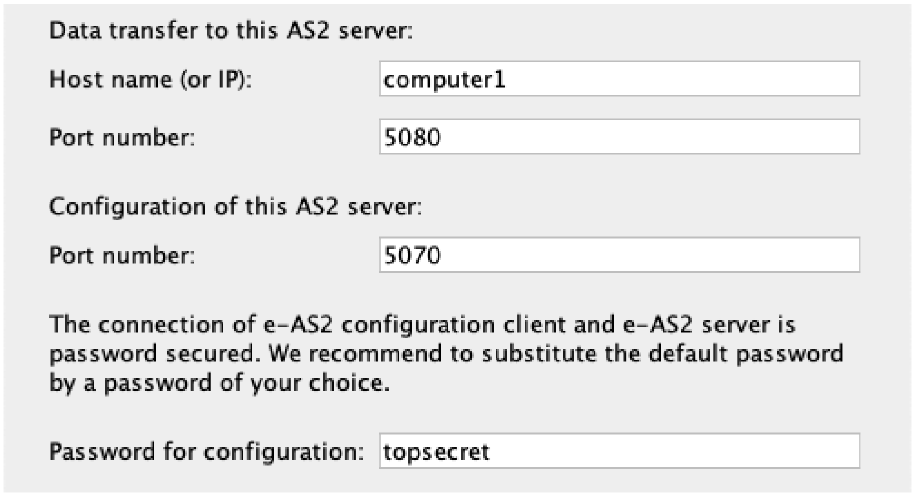
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [✓] e-AS2 Server
  - [✓] Server
  - [ ] MySQL Support
- [ ] e-AS2 Client
  - [ ] GUI Client
- [ ] Documentation

Figure 3.8. 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.

[22]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 4 (*Firewall and Proxy configuration*).



Data transfer to this AS2 server:

Host name (or IP):	computer1
Port number:	5080

Configuration of this AS2 server:

Port number:	5070

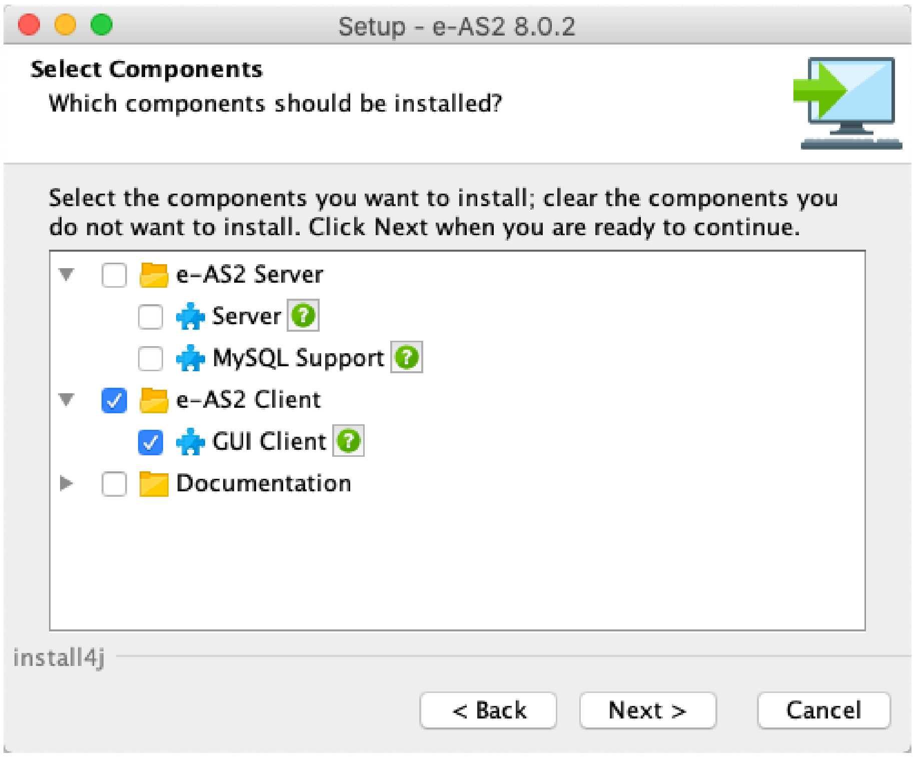
The connection of e-AS2 configuration client and e-AS2 server is password secured. We recommend to substitute the default password by a password of your choice.

Password for configuration:	topsecret

Figure 3.9. 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 3.10, “e-AS2 installation - GUI client only” and Figure 3.11, “e-AS2 GUI client installation - parameters” you will see the two key steps for this installation.



Setup - e-AS2 8.0.2

Select Components

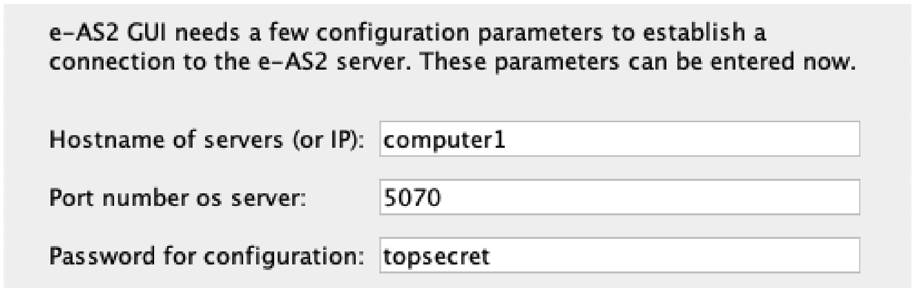
Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- ☐ e-AS2 Server
  - ☐ Server
  - ☐ MySQL Support
- ☑ e-AS2 Client
  - ☑ GUI Client
- ☐ Documentation

Figure 3.10. 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.


e–AS2 GUI needs a few configuration parameters to establish a connection to the e–AS2 server. These parameters can be entered now.

Hostname of servers (or IP): computer1

Port number os server: 5070

Password for configuration: topsecret

Figure 3.11. e-AS2 GUI client installation - parameters

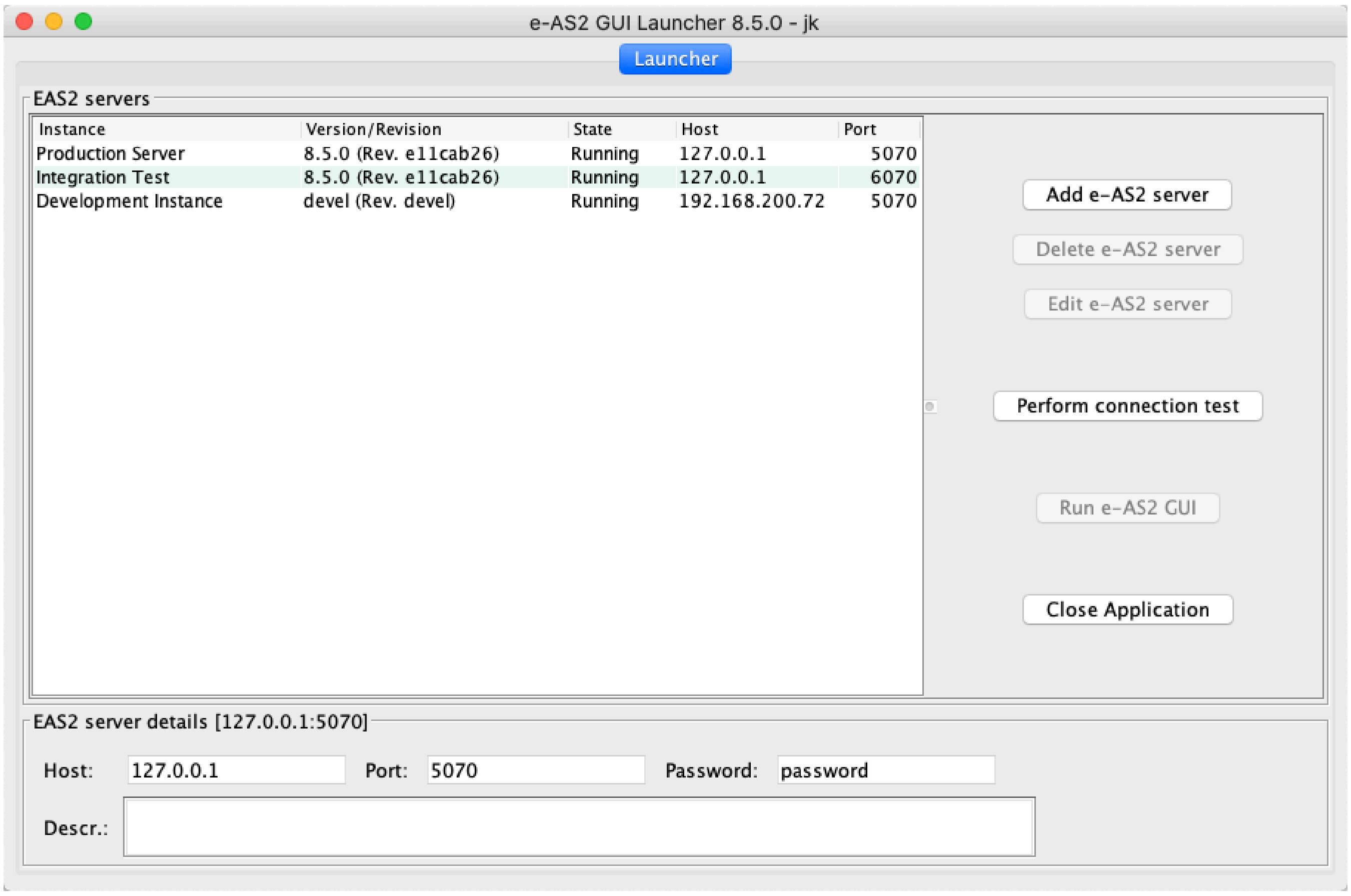
The dialogue for entering the communication parameters will be displayed adapted to this scenario. You will enter the parameters as specified in the following table.

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 10.2.8 (*Communication parameters, configuration*).

3.3 The GUI Launcher

In enterprise environments you may be confronted with the need to manage multiple installed e-AS2 instances from one workstation. Generally this is possible on any desktop platform by multiple installations of the GUI component. However, this approach is no longer practicable from a certain number of server instances to be maintained. We recommend that you then use the GUI Launcher.



Instance	Version/Revision	State	Host	Port
Production Server	8.5.0 (Rev. e11cab26)	Running	127.0.0.1	5070
Integration Test	8.5.0 (Rev. e11cab26)	Running	127.0.0.1	6070
Development Instance	devel (Rev. devel)	Running	192.168.200.72	5070

EAS2 server details [127.0.0.1:5070]

Host: 127.0.0.1 Port: 5070 Password: password

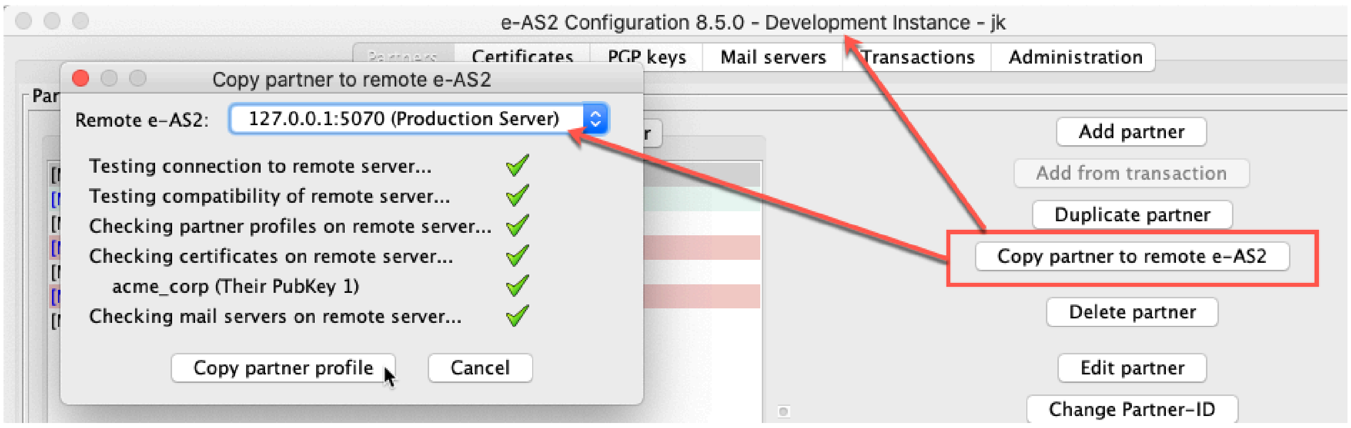
Descr.:

Figure 3.12. GUI-Launcher

After a new installation of e-AS2 or after an update witch installs the GUI Launcher for the first time the connection to the “own” server is automatically preconfigured. More server connections can be added flexibly and easily. The GUI Launcher is compatible with all servers of version 8.0 or higher. In the server list, the server version is shown in the second column.

When starting the GUI launcher and when pressing the button “Perform connection test” the current status of all configured servers is retrieved and displayed in the status column. To start the configuration for a server, select the entry in the list and click on “Run e-AS2 GUI”.

Once e-AS2 instances are configured in the GUI launcher, partner profiles can be easily copied between instances. This feature is available as part of the partner maintenance tab in e-AS2 GUI.



Remote e-AS2: 127.0.0.1:5070 (Production Server)

Testing connection to remote server... ✓
Testing compatibility of remote server... ✓
Checking partner profiles on remote server... ✓
Checking certificates on remote server... ✓
acme_corp (Their PubKey 1) ✓
Checking mail servers on remote server... ✓

Figure 3.13. Copy partner profiles

As of version 8.6 of e-AS2 there is also a stand-alone copy-to-remote function for certificates, PGP keys and mail servers.

4 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 5 (*Configuring e-AS2 Enterprise*), 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 3 (*The architecture of e-AS2*) to understand the reason for this requirement.

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.

4.1 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.

4.1.1 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.

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).

In addition, a second port must be enabled if data will also be exchanged via HTTP/S. We recommend using port 5081 for this purpose.[23] To allow HTTP/S-connections with as well as without client authentication, you will need two different ports. In this case we recommend additionally using port 5082.

If in addition to the AS2 you have also acquired a license for the SMTP feature, your e-AS2 server should possibly also allow incoming SMTP connections. Yet another port will need to be enabled for this purpose. We recommend - based on the standard SMTP port - to use port 5025 for this.[24]

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

[23]This does not impact the standard port for HTTP/S (443) on the system. So you can certainly also run an HTTP/S-secured web application on the system.
[24]Please refer to the separate e-AS2 SMTP manual for more information.

4.1.2 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 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).

4.1.3 Outgoing connections

Firewalls sometimes 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 communication partners must be enabled in the firewall for the ports used by AS2. In most cases, AS2 uses port 5080 or port 4080. So you can consider generally enabling these two ports for outgoing connections. Alternatively, you can implement activation for each communication partner individually. In this case, please advise the colleague configuring e-AS2 to coordinate the configuration of new partner connections with you.

4.1.4 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
5081	Internet (your partners)	e-AS2 server	Data communication via HTTP/S (optional)
5082	Internet (your partners)	e-AS2 server	Data communication via HTTP/S (optional)
5025	Your e-mail server on the company network	e-AS2 server	Data communication via SMTP (optional)
5070	Company network (LAN or VPN connections)	e-AS2 server	Configuring the e-AS2 entity
5080 (or others[25])	e-AS2 server	Internet (your partners)	Data communication

4.2 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.

[25] as coordinated with your communication partner

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 10.2.7 (*Parameters for proxy use*).

4.2.1 Incoming connections

Incoming connections require a reverse proxy configuration. Configure your proxy server to accept HTTP and/or HTTP/S 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 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.

Also remember e-AS2 has a separate certificate manager used for AS2 cryptography, as well as for SSL connections. When using a reverse proxy, you will generally be able to transparently connect HTTP/S connections through to the e-AS2 server, which will address the respective HTTP/S port there. Or you can alternatively implement the SSL handshake in the proxy server based on the certificates installed there, and internally forward the secured connection to e-AS2 as an unsecured HTTP connection.

4.2.2 Outgoing connections

Outgoing, the proxy server must accept HTTP and/or HTTP/S 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 (or port 443 for HTTP/S). So the respective restrictions must be withdrawn for AS2 connections.

4.2.3 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.

5 Configuring e-AS2 Enterprise

The configuration of e-AS2 Enterprise is documented step by step with examples of use.

This requires the firewall to already be configured as described in Chapter 4 (*Firewall and Proxy configuration*).

We further assume the software was started as described in Chapter 3 (*The architecture of e-AS2*) and you have access to the configuration GUI. In addition to the GUI you will also need access to the file system of the computer running the e-AS2 server so you can submit files for transmission or locate received files.

5.1 The interface area

Before starting with the examples of use we'll take a look at the e-AS2 Enterprise interface area.

On initial startup the e-AS2 server will add[26] a new directory named `interface`. The following additional directory structure will be created under `interface`:

`dr`	This is where the e-AS2 server saves delivery reports for completed send jobs.
`in`	This is the directory where the e-AS2 server saves received files which were successfully associated with a configured communication partner.
`in_cert`	This is the directory where the e-AS2 server saves received certificates.
`in_mdn`	This is the directory where the e-AS2 server saves received MDNs (Message Disposition Notifications).
`orphan`	This is the directory where the e-AS2 server saves received files it was unable to associate with a configured communication partner.
`out`	An additional structure with one sub-directory per communication partner is added in this directory. The directory names correspond with the internal partner IDs assigned whilst configuring the partner profiles.
`out_mdn`	This directory is the global interface directory for asynchronous MDN. There further are local `out_mdn` directories per partner. These can be found in each partner directory for each configured partner under `out`.
`tempfiles`	Every file generated by the e-AS2 server is initially added here using a temporary name, and then moved to the destination with the final name. This ensures the complete file appearing in the destination is an atomic event.

We will address the individual directories in more detail over the course of the text once they become relevant for the respective use case being viewed.

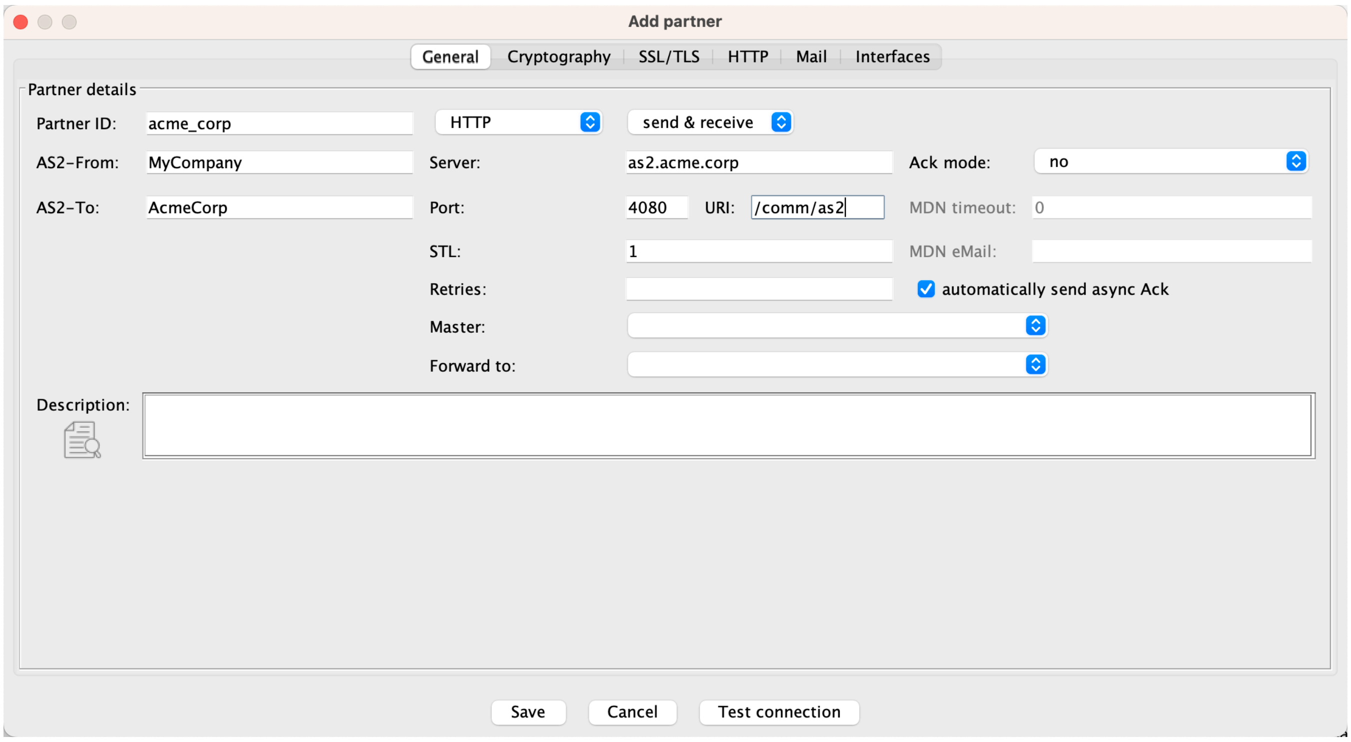
5.2 Configuring the connection to the partner

Before being able to send data to a partner in e-AS2 Enterprise you will first need to configure the connection to the partner. Add a new partner profile by pressing the "Add partner" button. This will open a new dialogue for entering the parameters for the partner connection. The entry

[26]This is the default setting, which can be changed. See Section 10.2.26 (*Interface area*).

fields are arranged under different tabs to group them by context. Most basic parameters are under the “General” tab.

First assign the new profile a meaningful partner ID. This serves as an internal identifier for that partner. Then enter the AS2 relation parameters, i.e. the values for “AS2-From” and “AS2-To” you coordinated with your partner.[27]



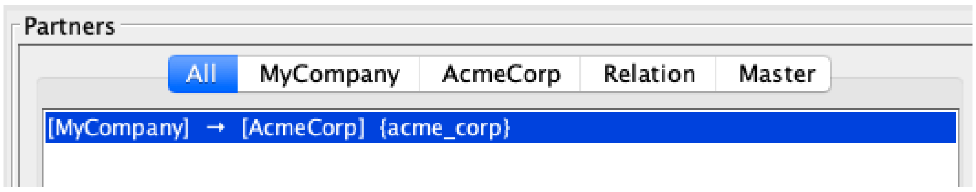
Partner details

Partner ID:	acme_corp	HTTP	send & receive		
AS2-From:	MyCompany	Server:	as2.acme.corp	Ack mode:	no
AS2-To:	AcmeCorp	Port:	4080 URI: /comm/as2	MDN timeout:	0
		STL:	1	MDN eMail:	
		Retries:		✓ automatically send async Ack	
		Master:			
		Forward to:			
Description:					

Figure 5.1. Configuring company “ACME Corporation”

You will also need to enter the parameters for the communication connection. For server, enter a computer name or an IP address. (Please note you can only use computer names if the operating system on the e-AS2 computer features a working name resolution!) Under port, enter the port number under which you can reach the AS2 service on the partner’s server. This value can of course differ from the port you are using for this purpose (probably 5080). You will need to obtain definitive information from the partner. Last, you will need to enter the post request URI under URI as coordinated with the partner.[28]

Click “Save” to save the new partner profile.

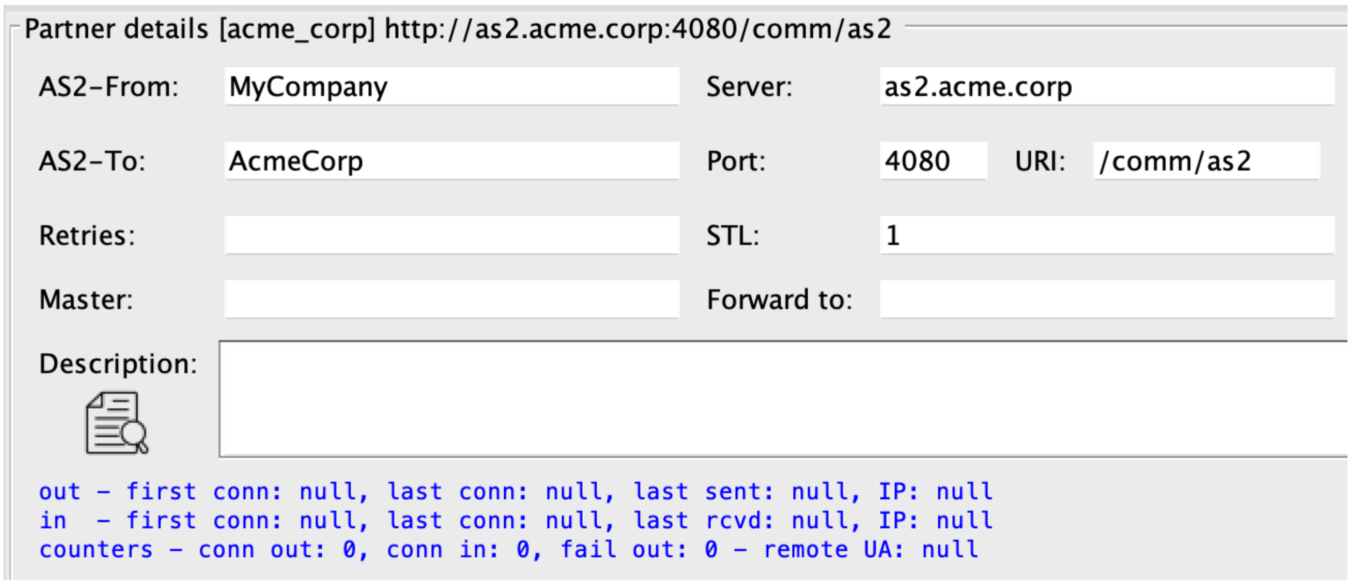


[MyCompany] → [AcmeCorp] {acme_corp}

Figure 5.2. A new entry in the partner list

[27]In practice this is often the GLNs of the two companies. But it can also be just two meaningful strings, both parties agreed on.
[28]Also refer to Section 2.2 (*Establishing a connection*).

The new profile will appear in the partner list. This list is primarily based on the AS2 relation (with the respective identifiers in square brackets) and also shows the internal partner ID (in curly brackets). Details for the currently selected partner entry appear at the bottom of the GUI. At the top of the details panel you will see the communication parameters for this connection in form of a URL (`http://as2.acme.corp:4080/comm/as2`). If the partner provided you with the connection parameters in this format, you can now check if you entered everything correctly.



Partner details [acme_corp] http://as2.acme.corp:4080/comm/as2

AS2-From:	MyCompany	Server:	as2.acme.corp		
AS2-To:	AcmeCorp	Port:	4080	URI:	/comm/as2
Retries:		STL:	1		
Master:		Forward to:			
Description:					

```
out - first conn: null, last conn: null, last sent: null, IP: null
in  - first conn: null, last conn: null, last rcvd: null, IP: null
counters - conn out: 0, conn in: 0, fail out: 0 - remote UA: null
```

Figure 5.3. The partner details

At the bottom of the details panel you find statistics for this connection, in a small blue font.

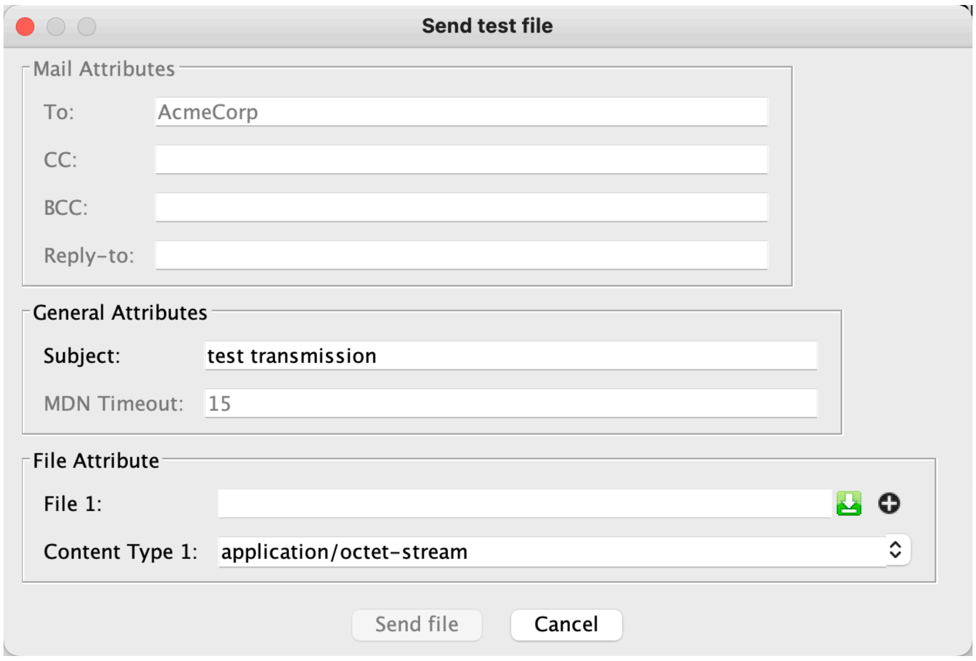
5.3 Sending files

For the first example of use we will entirely skip the cryptography functions in e-AS2 Enterprise. The goal is to send a file from your company (*MyCompany*) to your business partner (*AcmeCorp*).

5.3.1 Interactive transmission of test data

After adding a new partner profile you can interactively send test data as a quick test option. To do so, select a partner profile (in this case the profile with the internal ID *acme_corp*) and click the “Send file” button. This will open a specific dialog as shown in Figure 5.4, “Send test file to recipient”. Click the green button and select any – not too large – file to transmit.[29] The button labelled “Send file” will then become active. Click that button to initiate the test transmission.

[29]Better yet, create a small test file with short but concise text. E.g. enter your contact information (name, telephone number, e-mail address) as the contents of a test file. This will make it easier for the recipient’s staff to determine who sent the test data.



To: AcmeCorp

CC:

BCC:

Reply-to:

Subject: test transmission

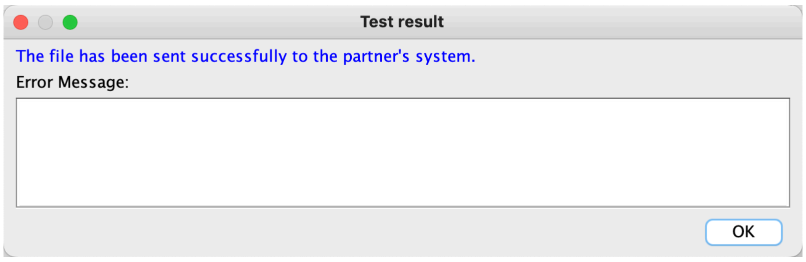
MDN Timeout: 15

File 1:

Content Type 1: application/octet-stream

Figure 5.4. Send test file to recipient

You can change the subject and content type to your liking before transmission.[30] Clicking “Send file” will immediately transmit the selected file to ACME Corporation. After successful transmission a notification window will appear to confirm.



The file has been sent successfully to the partner's system.

Error Message:

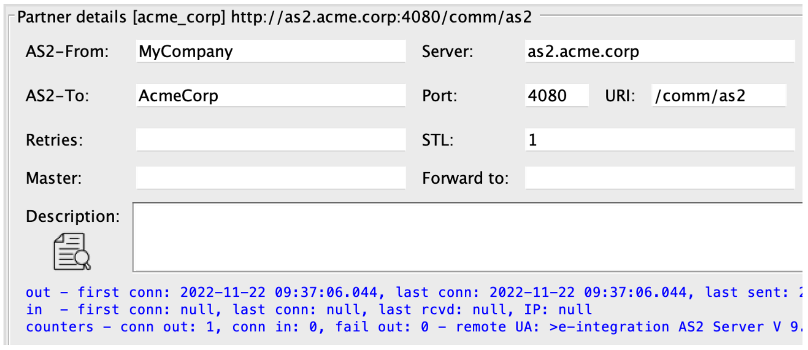
Figure 5.5. File transfer successful

You can send up to five files in one go, which will work only if the partner supports receiving multifile transmissions. Click the black “plus” button next to the file open button to add more files.

If you select a profile which has asynchronous MDN activated, then the MDN timeout field becomes active. You can enter the number of seconds the send file dialog should wait for the MDN to arrive. If the MDN does not arrive within the specified time frame an error message will be displayed.

Take a look at the statistics at the bottom of the partner details panel. You will notice, that they have been automatically updated and now reflect the first successful transmission to that partner having taken place.

[30]The mail related values in the top part of the dialog will only become active, if you have licensed the SMTP feature and select an SMTP partner profile before opening the send file dialog.



Partner details [acme_corp] http://as2.acme.corp:4080/comm/as2

AS2-From:	MyCompany	Server:	as2.acme.corp
AS2-To:	AcmeCorp	Port:	4080 URI: /comm/as2
Retries:		STL:	1
Master:		Forward to:	
Description:			

```
out - first conn: 2022-11-22 09:37:06.044, last conn: 2022-11-22 09:37:06.044, last sent: 2
in  - first conn: null, last conn: null, last rcvd: null, IP: null
counters - conn out: 1, conn in: 0, fail out: 0 - remote UA: >e-integration AS2 Server V 9.
```

Figure 5.6. Partner details after successful transfer

5.3.2 Normal data transmission

Interactive sending is of course only suitable to initially test the connection. During normal use files to be transmitted will be generated offline, e.g. by automatically being exported from your enterprise resource planning system. We will simulate normal use by submitting a file at file system level for transmission.

Open a shell session or use a file manager of your choice to locate the `interface/out` directory in the installation directory of the e-AS2 server. Here you will see a new subdirectory `acme_corp`. This directory was automatically generated when saving the new partner profile.

Go to this new subdirectory. Here you will see a special file named `_ALIVE_`. The existence of this file indicates a partner profile for this partner subdirectory exists in the configuration. On startup, the e-AS2 server scans all subdirectories under `out` and checks if partner profiles with the respective partner IDs exist in the configuration. If so, the file `_ALIVE_` is added, if not, the file `_DEAD_`. This is intended as an aid for the user at the output interface.[31]

If you now want to send a file to *AcmeCorp*, all you need to do is copy this file to the `acme_corp` directory. The e-AS2 server will detect the file and move it to the subdirectory `_work_`. It will then stay there throughout the file transfer to the other party. Following successful transmission the file automatically disappears from this directory. When linking e-AS2 Enterprise to an enterprise resource planning system, you can configure it so outgoing files are saved directly to the respective interface directory for the respective partner.[32]

5.3.3 Specifying the MIME type

When sending files you can explicitly specify the MIME type for the data by attaching the short term for the MIME type to the file name, comma separated. By default, e-AS2 Enterprise assigns MIME type `application/octet-stream` to all files. If you now provide e.g. a file named

```
testfile,EDIFACT
```

in one of the `out` directories for transmission, this will be transmitted to the respective partner using the name `testfile`, with `application/edifact` indicated as the MIME type. The following table lists all the MIME types supported by e-AS2 Enterprise.

e-AS2-Code	MIME types
BINARY	application/octet-stream

[31]Directories tagged as `_DEAD_` can safely be deleted (by the user). e-AS2 Enterprise does not automatically delete interface directories which are no longer needed to ensure important files in the directories aren't accidentally lost.
[32]Please also refer to Section 10.2.35 (*Preventing sending unfinished files*).

e-AS2-Code	MIME types
EDIFACT	application/edifact
X12	application/edi-x12
XML	application/xml
TEXT	text/plain
EDICONS	application/edi-consent

Attaching the MIME type short term to the file name is not case sensitive.

5.4 Receiving files

As long as the e-AS2 server is running and can be reached from the internet via the port configured for data communication, it is ready to receive. This requires no additional action. The process of receiving decrypted and unsigned files is approximately as follows.

5.4.1 Accept connection

When the sender reaches the e-AS2 server using the IP address and port, the connection is initially always accepted. Data will also straight away be received as defined in the AS2 protocol.

5.4.2 URI comparison

Processing of incoming messages starts with comparing the URI, the message is directed to, with the locally configured and expected URI for incoming connections.[33] If the URIs match, the rest of the MIME message will be completely received. If on the other hand the URI is not as expected, the server immediately terminates the connection at this point so no unnecessary data transfer occurs.

5.4.3 receiving data, allocation and acknowledgement

Following successful URI comparison the complete MIME message is received and the essential parameters extracted from the MIME headers. First and foremost the AS2 relation (AS2From, AS2To) is important for further processing. After receiving the data, e-AS2 Enterprise will search for a partner profile with the corresponding AS2 relation.

Please note, searching for the AS2 relation in the e-AS2 configuration database requires the identifiers to be reversed internally. I.e. the AS2From value received from the partner will be searched in the AS2-To field. The AS2To value received from the partner will be searched in the AS2-From field. In the configuration you entered the two values from your point of view in the role of the party sending the data. During the send process, data goes from (AS2-From) you to (AS2-To) the partner. Your partner of course sets these values from his point of view when sending data. These values are therefore reversed depending on the direction of transmission. This fact is particularly of interest when interpreting result files (see below).

If a partner profile with the respective AS2 relation was found,[34] the file received is saved to the `in` subdirectory in the e-AS2 interface. If no partner profile with the AS2 relation you are looking for is found, the file received is moved to the `orphan` subdirectory.

[33]For configuring the URI for incoming connections, see Section 10.2.1 (*Communication parameters, HTTP*).

[34]If several partner profiles have the AS2 relation you are looking for in the configuration it will use the first, sorted by internal partner ID.

The file received is saved to the respective directory under a generated unique file name. In addition, a second file with the same name and the file extension `.res` (for: result file) is generated. This file is an information file containing all the key information on the data transfer which occurred, in a defined syntax. So every time e-AS2 Enterprise receives data, you will have two files, the file with the data received and the additional result file. The structure of the result file is documented further below, in Section 5.4.4 (*The result file*).

The file is now available for further processing. e-AS2 Enterprise will generate an acknowledgement, which is returned to the sender, once the file has been successfully saved to the file system. So the sender is informed whether it was received correctly. The acknowledgement does not provide information on the data being processed further in a subsequent EDI or enterprise resource planning system.

5.4.4 The result file

The name of the result file is generated systematically in e-AS2. Example:

```
RECV-20110721171150-746-000042.res
```

In the case of a data transfer which cannot be assigned a partner profile, as follows:

```
ORPH-20110721171150-746-000042.res
```

The general pattern is:

```
PPPP-YYYYMMDDhhmmss-xxx-nnnnnn.res
```

Where:

PPPP	is a prefix, RECV for regular data received which was allocated to a partner profile. ORPH for data received which was not allocated.
YYYYMMDD	The current date (year, month, day).
hhmmss-xxx	The time received (hour, minute, second, millisecond).
nnnnnn	A sequential six-digit number with leading zeros. (Starting with 0 and constantly incrementing. This is reset to 0 after restarting the e-AS2 server.)

The job of the result file is to provide a complete description of the data reception. It is configured as a series of abbreviation value sets in successive lines.

Example

```
INST_NAME     [Integration Test]
RECEIVED      [2020-04-29 09:20:21.137]
ACKNOWLEDGED  [2020-04-29 09:20:21.149]
AS2FROM       [AcmeCorp]
AS2TO         [MyCompany]
FILENAME      [TESTFILE]
MESSAGE_ID    [EAS2-20200429042020-989-000002@192.168.200.122-komsrv]
SUBJECT       [test transmission]
CERT_SERIAL   [null]
CERT_ALIAS    [null]
P_INT_ID      [acme_corp]
TRANS_ID      [7]
```

```
DOC_ID         [D4000000000000001@EAS2]
ORG_DOC_ID     [D4000000000000001@EAS2]
STATUS         [1]
ERRORCODE      [0]
DESCRIPTION    [null]
CONTENT_TYPE   [application/octet-stream; name=TESTFILE]
MDN_REQUEST    [false]
```

Each line first contains an abbreviation stating the contents. This is followed by the associated value in square brackets. So e.g. the fourth line states the sender's AS2-From identifier was "AcmeCorp". The following abbreviations in particular may appear in the result file.

INST_NAME	If an instance name is configured for your e-AS2 server in `EAS2.properties`, it is indicated here. This allows the result file to be related to the respective instance in environments with several e-AS2 installations.[35]
RECEIVED	This is the time the file associated with this result file was received.
ACKNOWLEDGED	This is the time the file associated with this result file was directly confirmed. A direct confirmation is provided via synchronous MDN or a simple HTTP acknowledgement, if the other party requested an asynchronous MDN.
AS2FROM	This is the AS2 Identifier of the sender, so the From component of the AS2 relation from the sender's perspective. Please note, you will find this information in the AS2-To field of the partner profile in your local configuration.
AS2TO	This is the recipient AS2 identifier, so the To component in the AS2 relation from the sender's perspective. Please note, you will find this information in the partner profile of your local configuration in the AS2-From field.
FILENAME	Name of the file as provided by the sender. Please note, this is not the name of the data file in the e-AS2 in-directory, which is – as already mentioned above – generated systematically so it's always locally unique. The sender includes the file name as it was given on his side for information purposes when transmitting via AS2. This information can be found behind the FILENAME tag in the result file.[36] Please note, that a sender could send a file with the same logical name (from his perspective) several times. This does not result in overwrites or data loss when received by e-AS2, since the physical file names are unique names generated in e-AS2 and the name provided by the sender is only used in the result file. This name assigned by the sender can e.g. be used to control further processing on your end by using it to transmit coded control information such as the name of the message type.
MESSAGE_ID	This is the message ID assigned by the sender of the data. AS2 requires a unique message ID for each transfer. e-AS2 uses the mes-

[35]For instance name configuration, see Section 10.2.39 (*Names for server entities*).
[36]We can think of it as a logical file name as opposed to the physical file name, which is used for storing the data on the hard disk.

	sage ID (among other things) to relate asynchronous MDNs to the correct transaction.[37]
SUBJECT	If the sender included a subject in the data transmission, it is added to the result file behind the tag SUBJECT. Use of the SUBJECT is optional. If none was sent, the value `null` is added to the result file. e-AS2 itself does not analyse the subject in any way. However, the subsequent application could use it for information or control purposes.
CERT_SERIAL	In the event the incoming message was signed, the serial number of the certificate used for the signature will be listed here.
CERT_ALIAS	In the event the incoming message was signed, the internal e-AS2 alias of the certificate used for the signature will be listed here.
P_INT_ID	The internal e-AS2 partner ID for the partner profile the incoming file was assigned to. If a partner profile could not be assigned, this information will remain blank.
CUST_NO	If there is a customer number configured in the partner profile, this number shows up here. Otherwise this line is missing.
TRANS_ID	The internal e-AS2 transaction ID for this process. The subsequent processing program for the data received can detect this value and save it for reference purposes.
DOC_ID	The extended document ID assigned by e-AS2 for this operation. The subsequent processing program for the received data can take this value and save it as a reference.
ORG_DOC_ID	Identical to DOC_ID.
STATUS	The status code for this transaction. This value is currently always 1.
ERRORCODE	The error code for this transaction. For successfully processed transactions this value is 0. In the event of an error it is set to 1. Please note, errors reflected here must have occurred after the data was transferred successfully. Had an error occurred during communication, e.g. because the connection was lost, no file, thus no result file, would have been saved to the local file system. Result files in the `orphan` directory always have the error code set to 1.[38]
DESCRIPTION	If an error occurs, so if the error code shows the value 1, this will contain a plain text description of the cause for the error.
CONTENT_TYPE	This contains the MIME type the sender of the data specified as the content type. e-AS2 Enterprise does not interpret the MIME type in any way. e-AS2 only regards it as a character string. If interpreting the MIME type is useful or required for further processing the received data, it must occur in the subsequent application.[39]

[37] Learn more about asynchronous MDN in Chapter 7 (*Interface for asynchronous MDN*).

[38] The reason for data being saved to `orphan` is always that an error occurred after having completely received the incoming message. No partner profile was found to allocate this process to. The AS2 relation indicated by the sender has not yet been configured in your e-AS2.

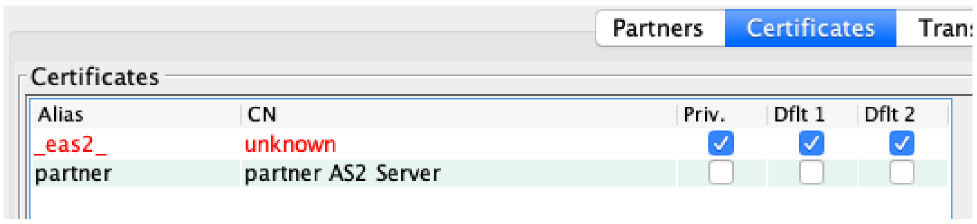
[39] Please note, this information may also occur with or without the logical file name attached depending on the AS2 software used by the sender of the data. When implementing any automatic processing of result files this possible variation must be taken into account.

MDN_REQUEST	This can have the value `true` or `false`. This informs the subsequent application if the sender of the message requested an asynchronous MDN.[40]
CMD_ON_RECV	This line only occurs if e-AS2 intends to execute a successive command for this process. For detailed documentation on successive commands please refer to Chapter 9 (*Command on receive*).
DRY_RUN	This line only occurs if the data was received by dry-running a transaction in the GUI client. For detailed documentation on dry-running transactions please refer to Section 12.5 (*Workflow testing with dry-runs*).

5.5 Preparing for encryption and signature

If you are exchanging non-sensitive data via AS2 which thus does not need to be protected, you can skip the following sections on encryption and signature. Continue with Section 5.10 (*Sending compressed data*)!

Using the cryptography functions in e-AS2 Enterprise requires some preparation. Cryptography in e-AS2 is based on so-called certificates. Think of it as a person's or a software entity's digital identity. Any data exchange requires two certificates describing the two identifies involved, yours and that of the communication partner.



Alias	CN	Priv.	Dflt 1	Dflt 2
eas2	unknown	✓	✓	✓
partner	partner AS2 Server	☐	☐	☐

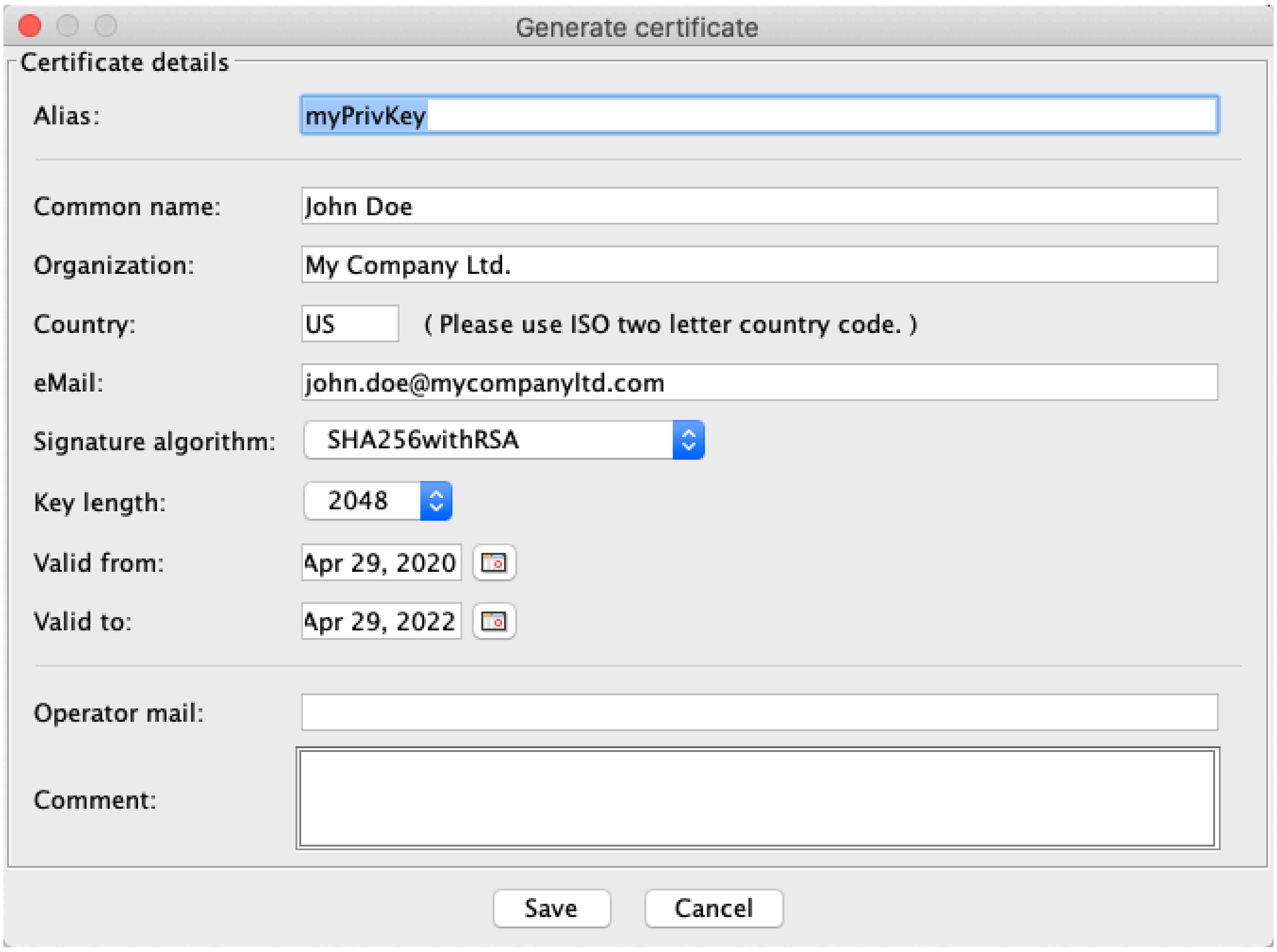
Figure 5.7. Preinstalled certificates

First switch to the "Certificates" tab in the GUI! Here you will see two certificates preinstalled. The alias `_eas2_` is your certificate. The alias `partner` is an initial partner certificate. Both certificates are view-only objects. You will need to generate or import new certificates to use e-AS2 with cryptography.

5.5.1 Generating and exporting your certificate

First switch to the "Private" tab at the right. Then press the "Generate private key" button. This will open a dialogue where you can enter an alias for the new certificate and your certificate details. In principle, you are free in choosing the data entered. However, the certificate should clearly identify you (or your company). View Figure 5.8, "Entering attributes for a new private key" as an example. The designated certificate validity is 2 years. We recommend keeping this.

[40]Learn more about asynchronous MDNs in Chapter 7 (*Interface for asynchronous MDN*).



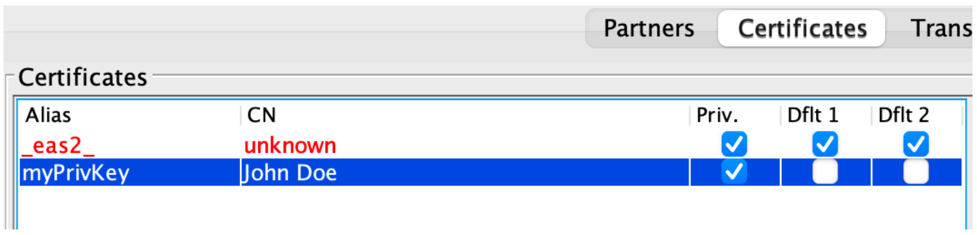
Certificate details

Alias:	myPrivKey
Common name:	John Doe
Organization:	My Company Ltd.
Country:	US (Please use ISO two letter country code.)
eMail:	john.doe@mycompanyltd.com
Signature algorithm:	SHA256withRSA
Key length:	2048
Valid from:	Apr 29, 2020
Valid to:	Apr 29, 2022
Operator mail:	
Comment:	

Figure 5.8. Entering attributes for a new private key

Press “Save” after entering your information. The GUI client will then generate a new cryptography certificate with the specified parameters and transmit it to the e-AS2 server. This process may take some time depending on the key length selected and the performance of the computer you are using (your desktop computer, not the server).

Once the process has completed the entry dialogue will disappear and the certificate list will appear again. Here you will see a new entry for this newly generated certificate.



Certificates

Alias	CN	Priv.	Dflt 1	Dflt 2
eas2	unknown	✓	✓	✓
myPrivKey	John Doe	✓		

Figure 5.9. New private key appears in the list

Check the details shown in the bottom of the GUI. If the generated certificate describes your own identity (or better: the identity of your AS2 system) adequately, this part of preparation is completed. The certificate only needs to be renewed after the validity period expires.[41]

[41]Cf. Section 5.18 (*Encryption and signature, in detail*).

To export the newly generated certificate, press the “Export public key” button. A Save File dialogue will appear where you can specify the directory and the file name to save it under. Please note, this will only save it to the computer running the e-AS2 GUI client, which may not necessarily be the same computer running the e-AS2 server.

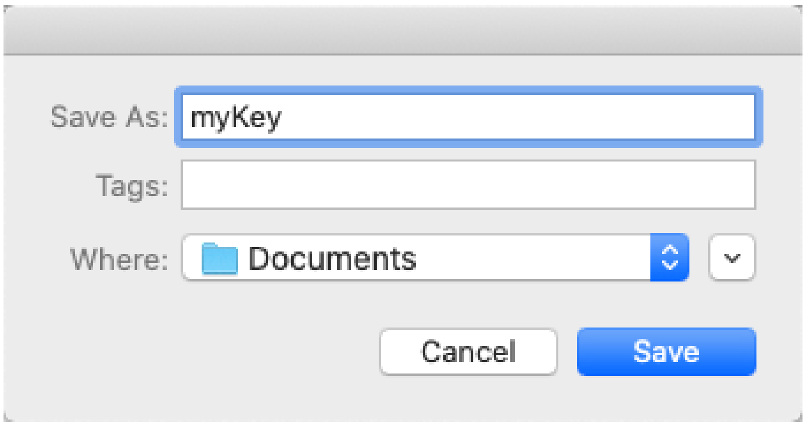

Save As: myKey
Tags:
Where: Documents

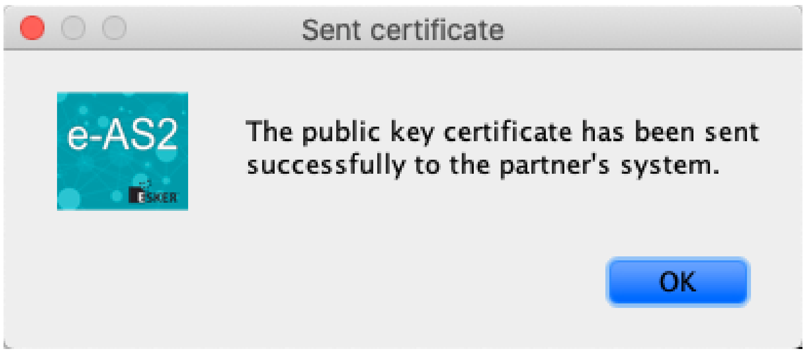
Figure 5.10. Export certificate

Do not add an extension to the file name! Click “Save”. This will generate two files with the file extensions `.der` and `.pem`, so for the above example:

```
myKey.der
myKey.pem
```

The first file contains a binary version of the exported public key certificates. The second file contains a text version (base64 coded) of the exported certificate. Select the desired file[42] and send it to your communication partner, e.g. by e-mail.

If supported by your communication partner’s AS2 solution, you may also transmit your public key certificate by AS2. First select the respective entry in the certificate list. Then go back to the “Partner” tab. Here, select the entry for the respective partner and press the “Send certificate” button. If this action was successful, the following message will appear.


Sent certificate

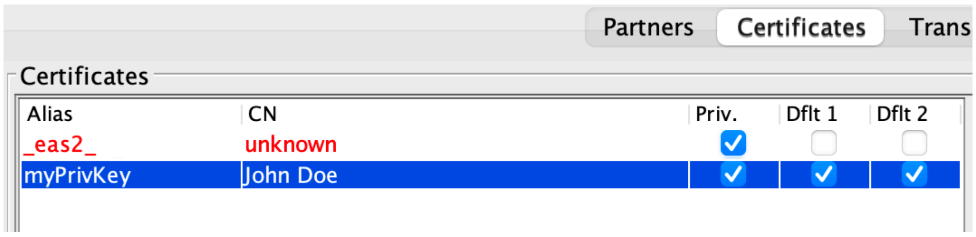
The public key certificate has been sent successfully to the partner's system.

Figure 5.11. Certificate sent successfully

Please note, transmitting the certificate will only be successful if sending data (as described in Section 5.3 (*Sending files*)) is functional. If you have not been able to send a test transmission, an error message will probably appear when attempting to send the certificate.

[42]The various AS2 products available on the market prefer different formats for importing certificates. Please contact your communication partner for the version they need. The contents of the two versions are completely equivalent.

Finally, set the newly generated certificate as the default certificate. To do so, go back to the certificate manager. Your newly generated certificate should still be selected in the list. Now press the buttons “Set default key” and “Set 2nd default key”. The ticks in the last two columns in the list will then change to your new certificate.[43]



Alias	CN	Priv.	Dflt 1	Dflt 2
eas2	unknown	✓		
myPrivKey	John Doe	✓	✓	✓

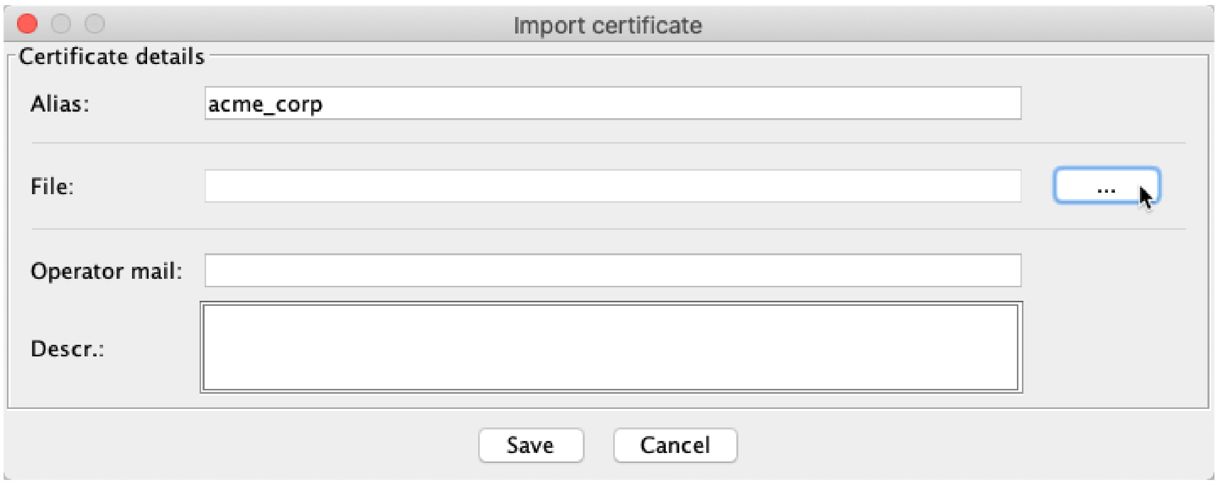
Figure 5.12. Setting default certificates

You can then delete the certificate `_eas2_`. It is no longer required. Select the certificate `_eas2_` and press the “Delete selected key” button. After confirming the security prompt which appears, the entry will be removed from the list.

5.5.2 Import partner certificate

In the previous section you read how to provide your communication partner your public key certificate. Vice versa, your partner also needs to transmit his public key certificate to you. This is often done by e-mail. In this case, save the respective e-mail attachment to a directory of your choice. If supported by your partner’s AS2 solution, he may also transmit his certificate to you directly via AS2. (This requires for general communication via AS2 to be functional.) The e-AS2 server will save the received certificate to the `interface/in_cert` directory under a name ending in `.p7c`. Move this file to the computer running the e-AS2 GUI.

Regardless of how you received the certificate file, import the certificate via e-AS2 GUI afterwards. To do so, in the certificate manager go to the “Public” tab and press the “Import public key” button.



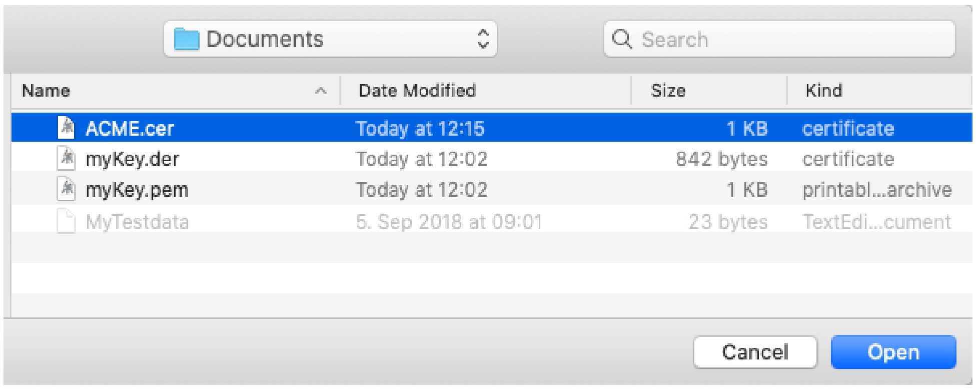
Alias: acme_corp
File:
Operator mail:
Descr.:

Figure 5.13. Import new public key

This will open a dialogue where you can first enter the desired alias for the new certificate. Then click the button with the three dots to the right of the “File” field. This will open a dialogue

[43]The significance of default certificates is detailed in Section 5.18 (*Encryption and signature, in detail*).

for selecting the respective file. The path to the file will be applied. Then click “Save” to start importing the certificate.



Name	Date Modified	Size	Kind
ACME.cer	Today at 12:15	1 KB	certificate
myKey.der	Today at 12:02	842 bytes	certificate
myKey.pem	Today at 12:02	1 KB	printabl...archive
MyTestdata	5. Sep 2018 at 09:01	23 bytes	TextEdi...cument

Figure 5.14. Select public key file

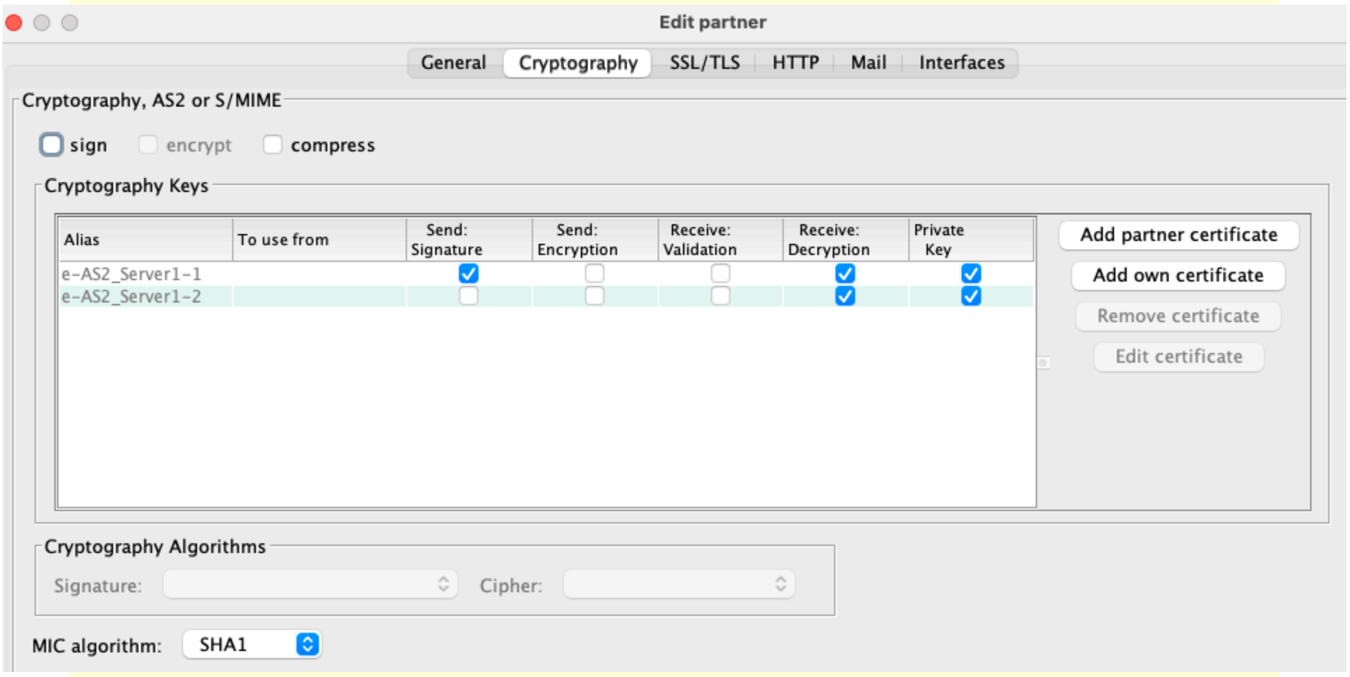
Whether the file is a certificate in binary or in text format, is irrelevant. e-AS2 will automatically detect the format and import the certificate. The certificate will then appear in the certificate list under the specified alias.

5.6 Sending encrypted files

Before continuing you first need to prepare encryption and signature as described in Section 5.5 (*Preparing for encryption and signature*). If you have not yet done so, please so do now!

You can decide to send your data encrypted if you do not wish for them to be transmitted over the internet in plain text. Review the partner details again which you configured for the connection with `acme_corp`! Select the “Cryptography” tab! Then click the “Edit partner” button.

========== CHANGED (MS: 05.01.2023) ==========



Cryptography, AS2 or S/MIME

☐ sign ☐ encrypt ☐ compress

Cryptography Keys

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
e-AS2_Server1-1		✓	☐	☐	✓	✓
e-AS2_Server1-2		☐	☐	☐	✓	✓

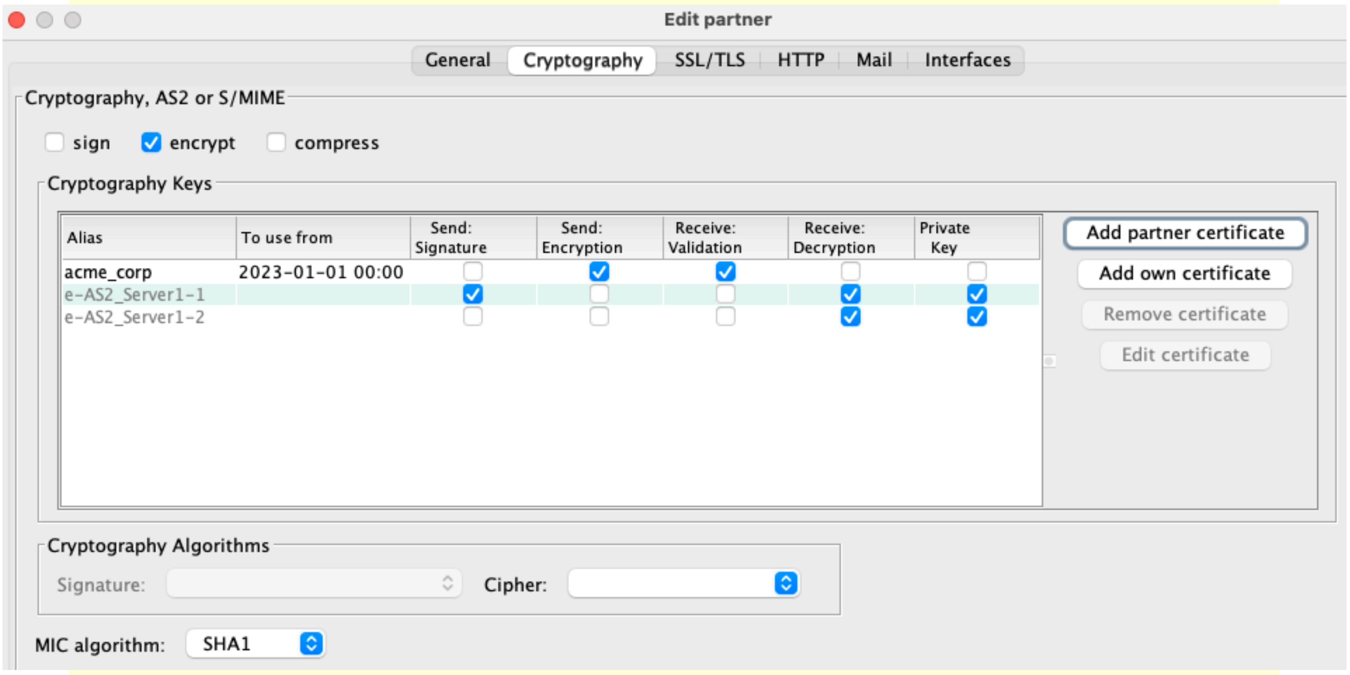
Cryptography Algorithms

Signature: Cipher:

MIC algorithm: SHA1

Figure 5.15. Cryptography parameters in the edit dialogue

You see the check box “encrypt” currently is currently disabled. This is because you have not yet specified which key to use for encryption. Add the “Partner certificate” `acme_corp` to the cryptography key list!



sign [✓ encrypt] compress

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
acme_corp	2023-01-01 00:00		✓	✓		
e-AS2_Server1-1		✓			✓	✓
e-AS2_Server1-2					✓	✓

Signature:

Cipher:

MIC algorithm: SHA1

Figure 5.16. Enabling encryption

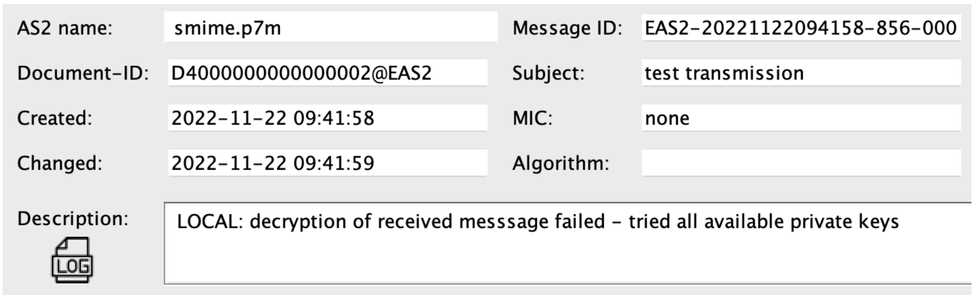
The check box “encrypt” will automatically be activated and encryption enabled.

That’s all you need to do. To send data, proceed as described in Section 5.3 (*Sending files*). The process does not differ from unencrypted sending. From now on, any data sent to this partner will automatically be encrypted. You may disable encryption at any time by unticking the “encrypt” check box.

For additional information on the group of topics “Encryption and signature” please refer to Section 5.18 (*Encryption and signature, in detail*).

5.7 Receiving encrypted files

In AS2, the sender of the data always determines whether files will be encrypted. You as the recipient have no control over this. e-AS2 Enterprise is always ready to receive encrypted data. However, there is no absolute guarantee the data received can also be decrypted. If the sender did not use the correct key for encryption, an error will occur during decryption. In this case the transaction list will show a new receive transaction with the check box “Error” ticked.



AS2 name:	smime.p7m	Message ID:	EAS2-20221122094158-856-000
Document-ID:	D4000000000000002@EAS2	Subject:	test transmission
Created:	2022-11-22 09:41:58	MIC:	none
Changed:	2022-11-22 09:41:59	Algorithm:	

Description: LOCAL: decryption of received messsage failed – tried all available private keys

Figure 5.17. Decryption failed

For transactions with error the cause can be determined in the “Description” field. If the text begins with `LOCAL:`, the error occurred whilst processing in the local system. In the example, the data stream was received fine, but an error occurred in the local system when attempting to decrypt the data. If decryption fails, the AS2 name will always be `smime.p7m`.

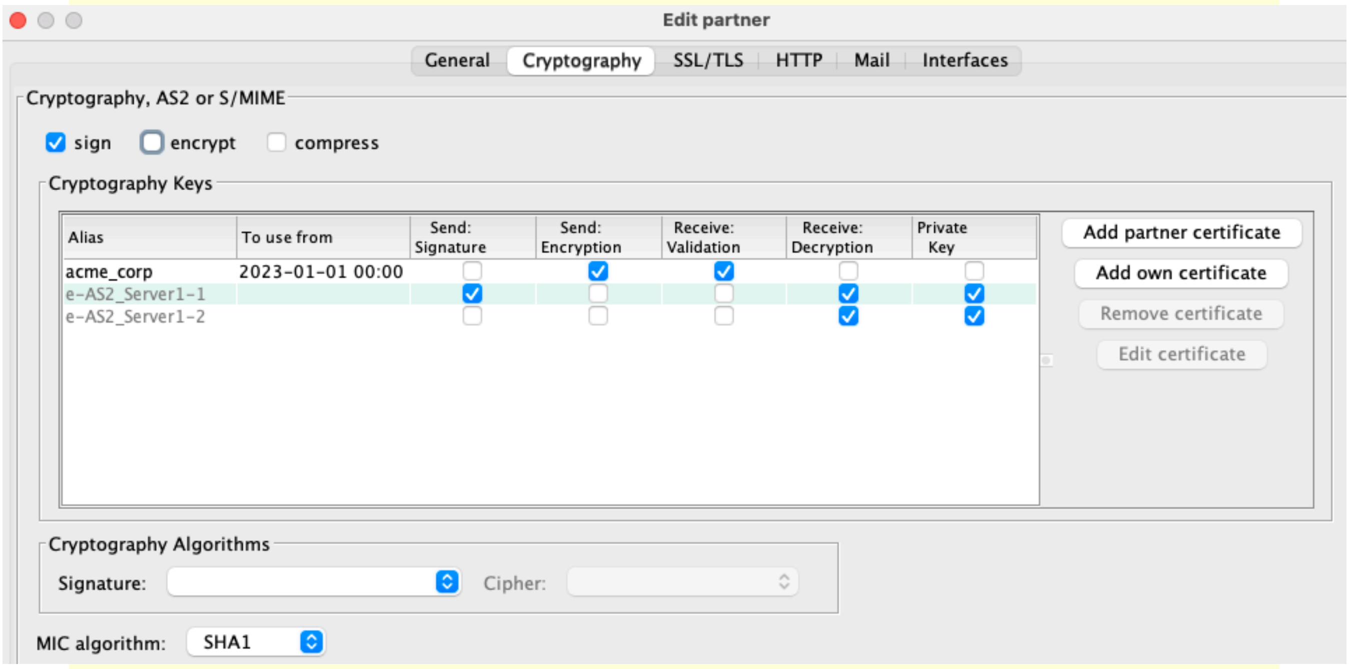
If the data was successfully decrypted, the process does not differ in any way from receiving unencrypted data as described in Section 5.4 (*Receiving files*).

For additional information on the group of topics “Encryption and signature” please refer to Section 5.18 (*Encryption and signature, in detail*).

5.8 Send 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 tick the “sign” check box!

========== CHANGED (MS: 05.01.2023) ==========



Cryptography, AS2 or S/MIME

[✓] sign [ ] encrypt [ ] compress

Cryptography Keys

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
acme_corp	2023-01-01 00:00	☐	✓	✓	☐	☐
e-AS2_Server1-1		✓	☐	☐	✓	✓
e-AS2_Server1-2		☐	☐	☐	✓	✓

Cryptography Algorithms

Signature: Cipher:

MIC algorithm: SHA1

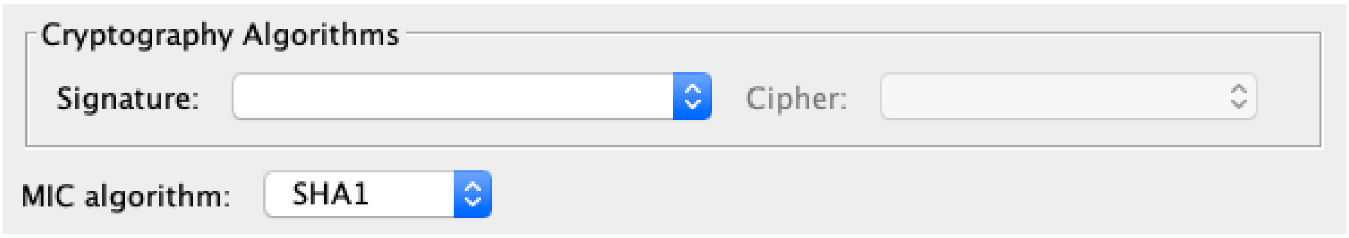
Figure 5.18. Activating signature

All of your files will now automatically be signed. Apart from that, sending files will be exactly the same as described in Section 5.3 (*Sending files*).

For additional information on the group of topics “Encryption and signature” please refer to Section 5.18 (*Encryption and signature, in detail*).

5.8.1 Set MIC algorithm

By default SHA1 is used for the calculation of the AS2-MIC. In e-AS2 this calculation is implemented independently of the digest algorithm for signatures. Before transmission, the MIC is always calculated and stored in the database together with other transaction data. If message signing is activated, the other party is requested to calculate a MIC and send it back within the MDN. After receipt of the MDN e-AS2 compares the MIC of the other party with the MIC that was stored in the transaction record.



MIC algorithm: SHA1

Figure 5.19. MIC-Algorithmus auswählen

If the transmission of signed messages does not succeed and you see error messages in the log in connection with the MIC check, it may be necessary to select a different MIC algorithm. Check the log messages to see which algorithm was used by the other party and adjust the setting in the partner profile accordingly!

5.9 Receiving signed files

The decision, whether or not to sign data is – as with encryption – always taken by the sender. e-AS2 can receive signed files without a problem and in the process automatically validates the signature, matching it with the certificate selected under “Partner key”. If signature validation is successful, the further course is exactly as described in Section 5.4 (*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.

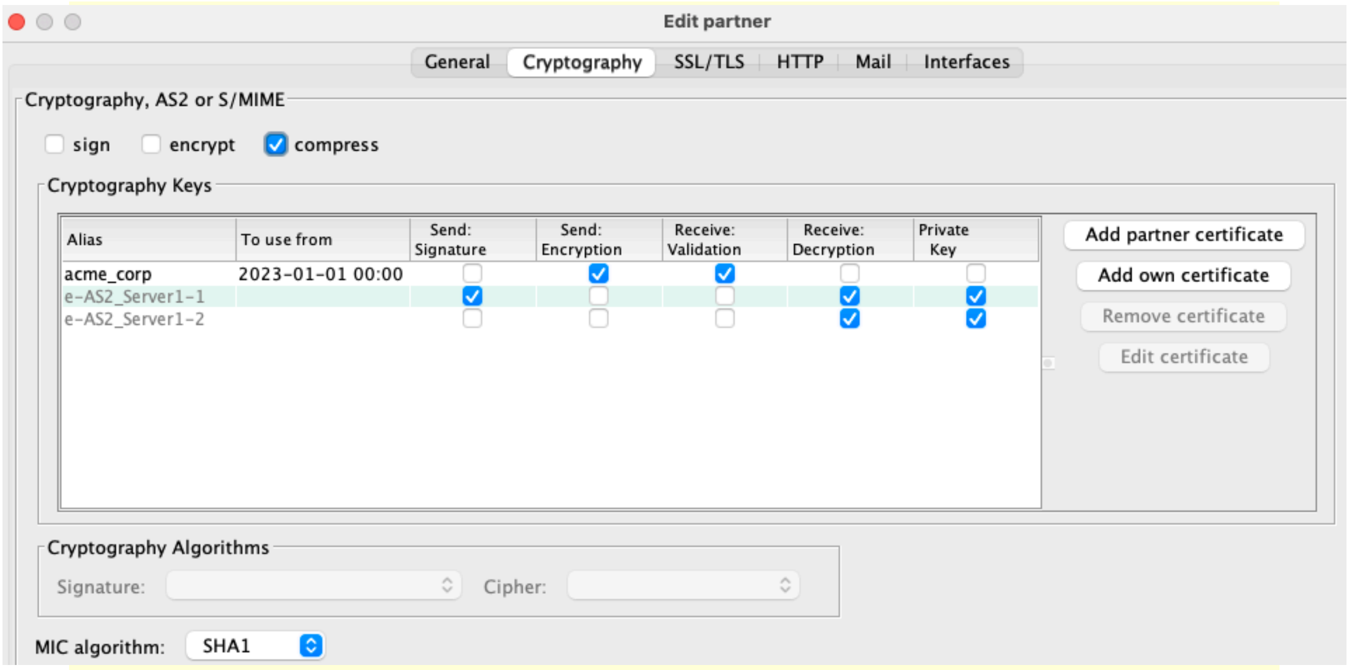
If no partner key alias was configured, signatures will not be validated for incoming messages. This configuration is only recommended in exceptions, since incoming messages will in this case be processed regardless of the validity of the signature.

For additional information on the group of topics “Encryption and signature” please refer to Section 5.18 (*Encryption and signature, in detail*).

5.10 Sending compressed data

You can decide to send your data compressed to reduce the data volume transmitted. To do so, open the edit dialogue for the partner profile and tick the “compress” check box!

========== CHANGED (MS: 05.01.2023) ==========



Cryptography, AS2 or S/MIME

☐ sign ☐ encrypt ✓ compress

Cryptography Keys

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
acme_corp	2023-01-01 00:00	☐	✓	✓	☐	☐
e-AS2_Server1-1		✓	☐	☐	✓	✓
e-AS2_Server1-2		☐	☐	☐	✓	✓

Cryptography Algorithms

Signature: Cipher:

MIC algorithm: SHA1

Figure 5.20. Enabling compression

All of your files will now automatically be compressed prior to sending. Apart from that, sending files will be exactly the same as described in Section 5.3 (*Sending files*).

5.11 Receiving compressed data

The decision, whether or not to send data compressed is – as with encryption and signature – always taken by the sender. e-AS2 can receive compressed files without a problem and will automatically decompress the data received. The e-AS2 user does not notice this process. The further course is exactly as described in Section 5.4 (*Receiving files*).

5.12 Requesting acknowledgements

During configuration you probably noticed the Ack mode field in the “General” tab. Here you can determine which type of acknowledgements you would like to receive for messages you sent to that partner. AS2 differentiates between four possible types of acknowledgement:

1. HTTP only (no MDN)
2. synchronous MDN
3. asynchronous MDN via HTTP
4. asynchronous MDN via e-mail

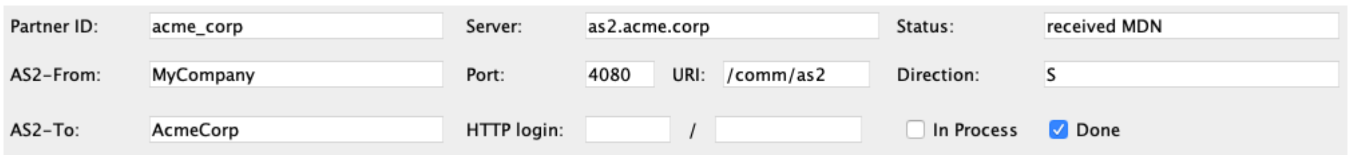
5.12.1 No MDN

Select Ack mode none means you do not wish to receive an MDN (Message Disposition Notification). The remote endpoint will then only return a brief HTTP-based acknowledgement of receipt, which means nothing more than that the message has been completely received. You will not know if problems arise during further processing. The respective transactions in the e-AS2 transaction list end in status “acknowledged”.

This Ack mode is not suitable for production connections. When configuring new connections, however, it may be helpful to execute initial tests with this Ack mode to purely monitor sending data. Once this is successful, select a different Ack mode to find out if the other end was able to process the data sent.

5.12.2 Synchronous MDN

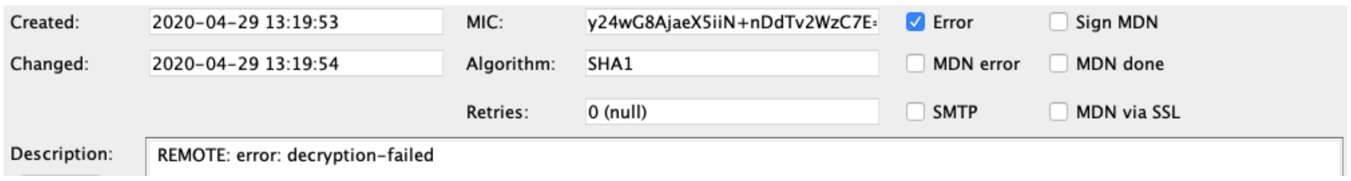
Select Ack mode `synchronous` to prompt the remote endpoint to return an MDN as the acknowledgement of receipt. An MDN allows the receiving AS2 server to return detailed messages on the success of processing to the sender. An MDN is structured and treated the same as a fully-fledged MIME message. This particularly means it is saved to the file system (under directory `interface/in_mdn`). It is further accompanied by a result file as described in Section 5.12.5 (*MDN result file structure*). e-AS2 Enterprise further analyses the contents of the MDN and writes any error messages to the transaction list in form of status changes. This provides you with significantly more information on the process than without MDN. The respective transactions end in status “received MDN”.



Partner ID:	acme_corp	Server:	as2.acme.corp	Status:	received MDN
AS2-From:	MyCompany	Port:	4080 URI: /comm/as2	Direction:	S
AS2-To:	AcmeCorp	HTTP login:	/	☐ In Process	☑ Done

Figure 5.21. Receiving MDNs

If the other end reported an error via MDN, the respective text will be added to the “Description” field.



Created:	2020-04-29 13:19:53	MIC:	y24wG8AjaeX5iiN+nDdTv2WzC7E=	☑ Error	☐ Sign MDN
Changed:	2020-04-29 13:19:54	Algorithm:	SHA1	☐ MDN error	☐ MDN done
		Retries:	0 (null)	☐ SMTP	☐ MDN via SSL
Description:	REMOTE: error: decryption-failed				

Figure 5.22. MDN received with error message

The Prefix `REMOTE:` indicates that this particular error message originated from the remote party.

5.12.3 Asynchronous MDN

By selecting Ack mode `asynchronous` you also request an MDN. However, there is one key difference in processing. The synchronous MDN is – as the term synchronous indicates – returned to the sender of the data during the same HTTP session in which the data was transmitted.

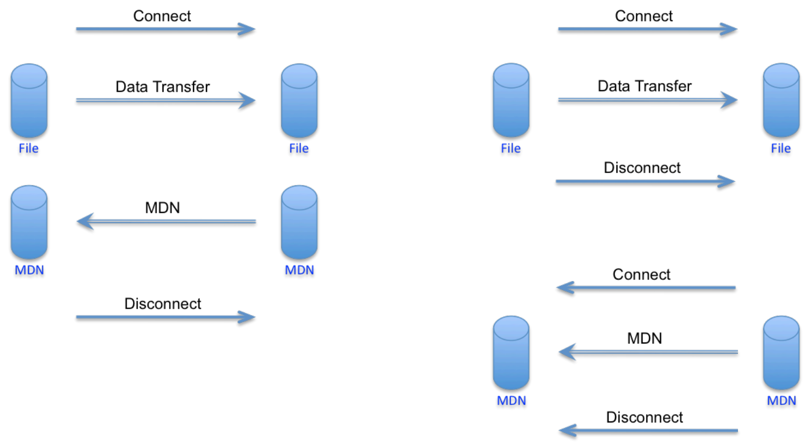


Figure 5.23. Synchronous MDN vs. asynchronous MDN

The process is different for asynchronous MDN. First the sender of the data establishes an HTTP session to transmit the data. This session is then closed immediately following transmission. At an undefined later point the recipient of the data establishes a new session to transmit the MDN. Figure 5.23, "Synchronous MDN vs. asynchronous MDN" compares the two MDN transmission processes.

In both cases, synchronous as well as asynchronous MDN, any error messages from the other end are added to the "Description" field. To distinguish the asynchronous MDN, it is prefixed by `ASYNC REMOTE:`.

MDN Timeout

When selecting Ack mode `asynchronous`, the field below will be enabled. Here you can enter an MDN timeout (in minutes). This specifies you expect the remote party to return an asynchronous MDN within the configured time. If no MDN is returned within this time frame, it is considered an error.

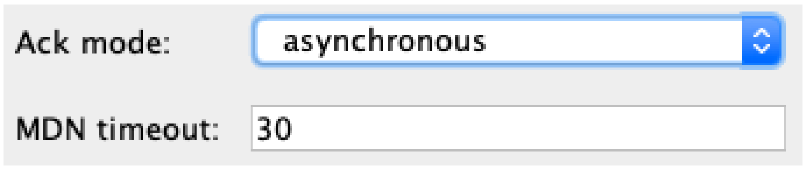

Ack mode: asynchronous
MDN timeout: 30

Figure 5.24. MDN timeout 30 minutes

After the specified time expires, the transaction is automatically set to error status. This also triggers the e-AS2 server to send an operator e-mail. This then requires manual action by the operator for processing of the transaction to be continued. However, if the expected MDN is later received, e-AS2 will respond to this MDN the normal way. I.e. the transaction is completed. If the late MDN contains an error message, the transaction will remain in error status. The error message transmitted by the other party is added to the description field for the transaction.

If you leave the "Timeout" field at `0`, a default MDN timeout will be used, which can be defined under `EAS2.properties`, cf. Section 10.2.13 (*Timeout settings for asynchronous MDN*). If this does not contain a default MDN timeout, no MDN timeout will be set. I.e. the system may wait for an MDN infinitely.

5.12.4 Asynchronous MDN via e-mail

The last alternative is the asynchronous MDN via e-mail. You, the sender of the data, can specify the desired destination e-mail address for this for each partner profile directly in the configuration mask.

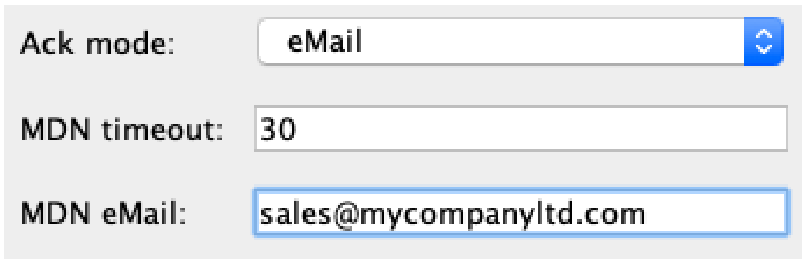

Ack mode: eMail
MDN timeout: 30
MDN eMail: sales@mycompanyltd.com

Figure 5.25. Configure e-mail address for MDN

With this configuration the configured e-mail address will be transmitted to the recipient of the message along with the MDN request. This prompts the recipient of the message to generate an asynchronous MDN and send it to the specified e-mail address. The e-mail address for MDNs is always determined by the sender of the data. So he is responsible for configuring a valid and useful e-mail address here. Therefore only enter a verified e-mail address in the "e-mail" field!

If the field with the e-mail address is left blank, a default address which can be defined in `EAS2.properties` will be used, cf. Section 10.2.20 (*Email configuration*).

5.12.5 MDN result file structure

The e-AS2 server saves all MDNs received to the `interface/in_mdn` directory. The name is the message ID of the respective transaction. Along with it, a result file with the extension `.res` will be added, which systematically contains all the key information for this process. Example:

```
EAS2-20200429145212-433-000008@192.168.200.122-komsrv
EAS2-20200429145212-433-000008@192.168.200.122-komsrv.res
```

The MDN file contains the complete MDN text as received from the other end.

Example

```
The AS2 message has been received. Thank you for exchanging AS2 messages with mendelson
 opensource AS2.
Please download your free copy of mendelson opensource AS2 today at http://
opensource.mendelson-e-c.com

Reporting-UA: mendelson opensource AS2
Original-Recipient: rfc822; AcmeCorp
Final-Recipient: rfc822; AcmeCorp
Original-Message-ID: <EAS2-20200429145212-433-000008@192.168.200.122-komsrv>
Disposition: automatic-action/MDN-sent-automatically; processed
Received-Content-MIC: 3/wMy7hxwhkzb6bmfJuCNn6EIFk=, sha1
```

The purpose of the result file is to provide a complete systematic description of all key data for the MDN received. The file should be – unlike MDN text – suitable for machine processing.

Example

```
AS2FROM       [AcmeCorp]
AS2TO         [MyCompany]
MESSAGE_ID    [EAS2-20200429145212-433-000008@192.168.200.122-komsrv]
P_INT_ID      [acme_corp]
DOC_ID        [82d18b66-9633-42d6-9b68-95df09c194f0]
ORG_DOC_ID    [82d18b66-9633-42d6-9b68-95df09c194f0]
STATUS        [3]
ERRORCODE     [0]
DESCRIPTION   [null]
```

Each line starts with a tag to identify the contents. This is followed by the associated value in square brackets. So e.g. the first line states the sender's AS2-From identifier was "AcmeCorp". In detail, the following abbreviations may appear in the MDN result file.

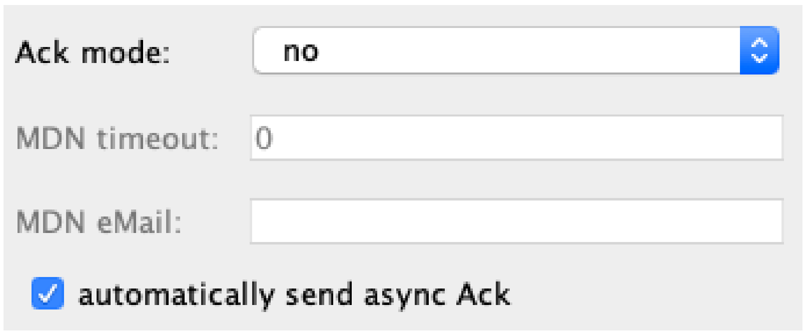
AS2FROM	This is the AS2 Identifier of the sender, so the From component of the AS2 relation from the sender's perspective. Please note, you will find this information in the AS2-To field of the partner profile in your local configuration.
AS2TO	This is the recipient's AS2 identifier, so the to-component in the AS2 relation from the sender's perspective. Please note, you will find this information in the partner profile of your local configuration in the AS2-From field.
MESSAGE_ID	This is the message ID assigned by e-AS2 when the data was sent. The AS2 protocol requires that the message ID is referenced in the corresponding MDN.

P_INT_ID	The internal e-AS2 partner ID for the partner profile the received MDN was assigned to.
DOC_ID	This value does not originate from the other end. When using the INF interface to add new send jobs you have the option to add a unique document identifier for each job. (see Chapter 8 (*The INF interface*).) When using this, the document identifier is added to the MDN result file as a reference number after receiving an MDN. If you did not specify a document identifier for the send job, this value will remain `null` in the MDN result file.
ORG_DOC_ID	========== ADDED (JK: 17.11.2022) ========== This value does not originate from the other end. When using the INF interface to add new send jobs you have the option to add a unique original document identifier for each job. (see Chapter 8 (*The INF interface*).) When using this, the original document identifier is added to the MDN result file as a reference number after receiving an MDN. If you did not specify an original document identifier for the send job, this value will remain `null` in the MDN result file.
STATUS	The status code for this transaction. This value is currently always 3.
ERRORCODE	The error code for this transaction. If you received a positive MDN, this value is 0. If you received a negative MDN, so the other end reported a processing error, it is set to 1.
DESCRIPTION	If an error occurs, so if the error code is 1, you will see the coded error message from the MDN here.

5.13 Auto-generate asynchronous MDN

In the previous section you learned quite a bit about the configuration for asynchronous MDNs from the perspective of the data sender. If e-AS2 is in the reversed role, it will need to respond to MDN requests from the other end adequately. e-AS2 supports all types of MDN and will generate MDN contents suitable for the specific situation.

If the other end requests an asynchronous MDN, from the recipient's perspective there is the option to generated a delayed MDN. In this way, application information for further processing of the received EDI data can be included in the MDN if required. e-AS2 Enterprise features a special interface for generating asynchronous MDNs with application information. This topic will be addressed in detail in Chapter 7 (*Interface for asynchronous MDN*). When configuring a new partner profile, however, the option "auto-generate async Ack" is activated by default. In this way, e-AS2 will automatically generate asynchronous MDNs for incoming messages (the same way as synchronous MDNs) and without external action.



Ack mode:	no
MDN timeout:	0
MDN eMail:	

✓ automatically send async Ack

Figure 5.26. Auto-generate asynchronous MDN

We usually recommend leaving this setting . Only disable auto-generating asynchronous MDNs if you're certain manually generated MDNs are required or useful for your application. You will then be responsible for providing the MDN data in a timely manner.

Please note, this setting applies to MDNs sent by your system after receiving files from the remote endpoint. This setting is entirely separate from the "Ack mode" setting, which refers to requesting MDNs for files you are sending.

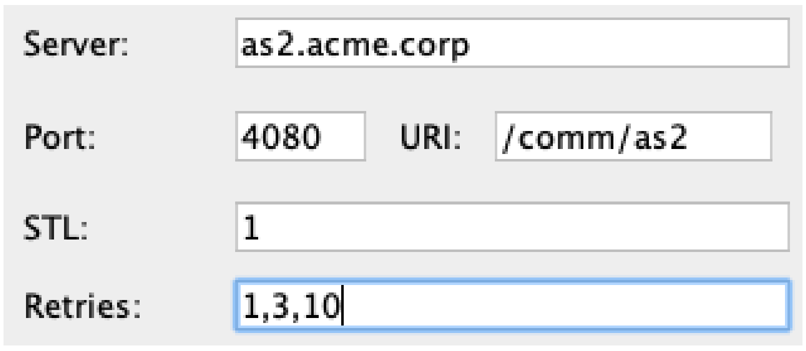
Asynchronous MDN vs. synchronous MDN from the sender's and recipient's perspective

Whether to return a synchronous or an asynchronous MDN is entirely at the discretion of the sender of the data. The sender of the data requests the MDN type convenient to him. The recipient of the data must respect the request and generate the requested MDN type. The recipient of the data may not arbitrarily ignore the request and generate a different MDN type. This indirectly also means the MDN type does not necessarily have to match between the communication partners. One direction may use a synchronous MDN whilst an asynchronous MDN is used in the other direction.

5.14 Setting the retry pattern

With respect to communication applications there's always a possibility individual data transfers will fail. If e.g. your internet connection is temporarily down, you will not be able to connect to any of your partners. Failures on the partner end or configuration errors (incorrect port number, etc.) could also cause errors with individual partner connections whilst others are working. e-AS2 Enterprise features a powerful mechanism for handling these cases.

Normally, network outages interfering with communication are only temporary. I.e. after a connection fails it's worth trying again a little while later. e-AS2 Enterprise handles these additional attempts automatically when configured accordingly. This is done using the "Retries" field in the partner profile. Here you can enter a so-called retry pattern, e.g. `1,3,10`. What does that mean?



Server:	as2.acme.corp
Port:	4080
URI:	/comm/as2
STL:	1
Retries:	1,3,10

Figure 5.27. Entering a retry pattern

After locating a file in the interface, e-AS2 Enterprise will first try to transmit it regardless of the retry pattern in this field. If the attempt is successful, the retry pattern doesn't apply. However, if this initial communication attempt fails, e-AS2 Enterprise will first leave the file in its location and retry after one minute (this is the 1 in the retry pattern 1,3,10). If this also fails, the next attempt will be made after 3 minutes more. Finally, after 10 minutes more a third attempt will be made to send the file. If this also fails, the transaction is set to error status and no further transfer attempts will be made.

You are flexible in adapting the retry pattern to your needs and wishes by entering a series of any number of pauses (measured in minutes), comma separated. If you want e-AS2 Enterprise to attempt infinitely to transmit the file, you can set an asterisk as the last specification in the retry pattern. This will result in the last pause specified being repeated infinitely. For example, the retry pattern `2,10,60,*` would result in repeat attempts after 2, after 10 more, after 60 more, and then infinitely every 60 minutes.[44]

The antipole of “Infinite tries” would be “Stop immediately”. You can also formulate this option, if you wish. Leave the “Retries” field completely blank. This will result in the transaction immediately being set to error status if the first transfer attempt fails. No repeat attempts will be made. However, this configuration is only recommended during the test phase. The partner list shows entries without retry pattern with a coloured background to alert you to this fact and ensure this configuration was not accidentally missed.

The configured retry pattern controls retries for all send processes, so for sending data as well as sending asynchronous MDNs.

If several transactions for the same partner are pending retry at any given time, the implicit retry method may apply. This means: If establishing a connection failed with the first transfer pending, it should be regarded a problem which applies to the connection to the respective partner in general, not just the individual transaction.[45] In this case, e-AS2 will automatically skip all other transactions pending for this partner and implicitly regard them as retried. I.e. the repeat counter for these transactions will increase without having made an explicit repeat attempt. This process eliminates unnecessary connection attempts and reduces the server load. The event is clearly recorded in the log file.

5.15 Select master profile

When using e-AS2 Enterprise to exchange your own business data with business partners, you will probably not need this configuration option. You will then have a number of AS2 relations to manage, all of which match the pattern (MyIdentifier, TheirIdentifier), so e.g.

```
(MyCompany, AcmeCorp)
(MyCompany, Oscorp)
(MyCompany, StarkInd)
...
```

However, you can also use e-AS2 Enterprise to transmit files as a communication hub, e.g. between different companies or between different departments within your company and the outside world. One possible constellation could be e.g.:

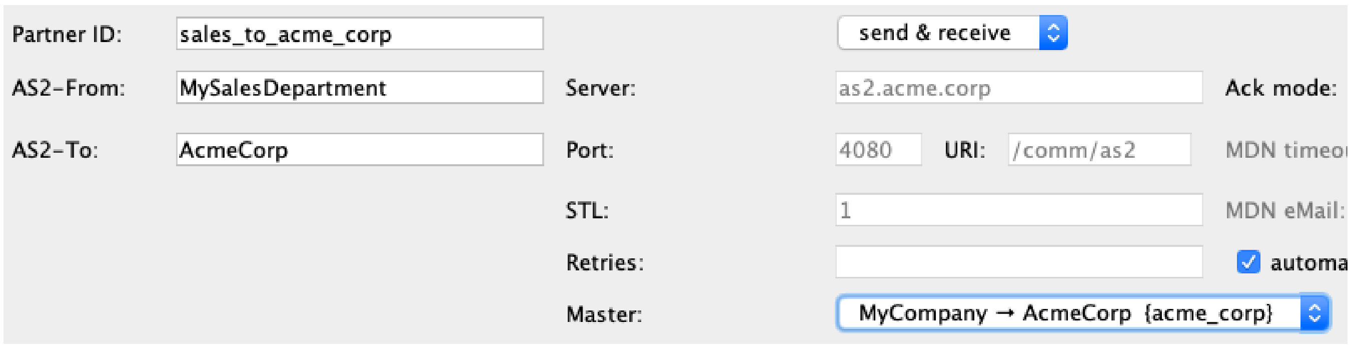
```
(MyCompany, AcmeCorp)
(MyCompany, Oscorp)
(MyCompany, StarkInd)
(MyMarketingDepartment, AcmeCorp)
(MySalesDepartment, AcmeCorp)
(MySupportDepartment, AcmeCorp)
...
```

[44] As a precaution, operating is notified of this situation with a one time e-mail when infinite mode is reached. This is to reduce the probability open transactions perpetually remaining in the system.

[45] Conversely, errors which only occur during processing on the other end and of course returning a negative MDN may only apply to individual transactions.

Different departments within your company communicate with `AcmeCorp` and are identified by separate AS2 identifiers. Most likely the communication parameters for the connection to the company ACME Corporation are always the same. e-AS2 Enterprise explicitly supports this type of constellation.

You configure a profile, explicitly entering the required communication parameters, in our example the profile `(MyCompany, AcmeCorp)`. Then configure additional profiles, selecting the first profile as the master.

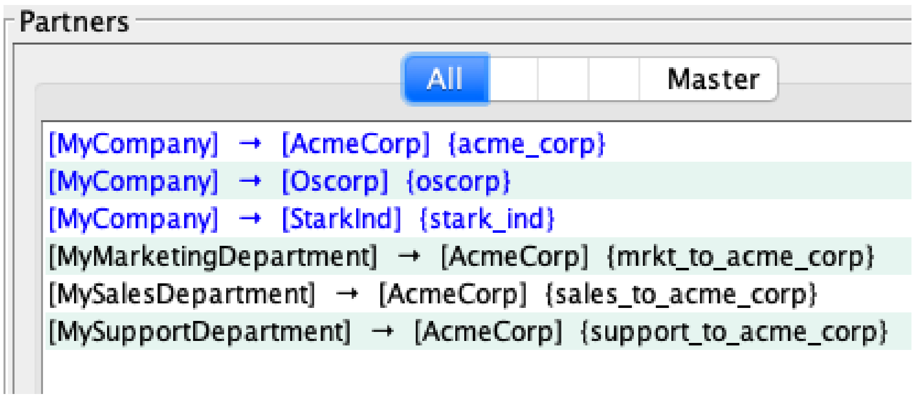


Partner ID: sales_to_acme_corp
AS2-From: MySalesDepartment
AS2-To: AcmeCorp
send & receive
Server: as2.acme.corp
Port: 4080 URI: /comm/as2
STL: 1
Retries:
Master: MyCompany → AcmeCorp {acme_corp}
Ack mode:
MDN timeo
MDN eMail:
✓ automa

Figure 5.28. Select master profile

After selecting a master profile, its communication parameters will be displayed. The respective fields become inactive. The values can no longer be changed. Any previously entered values are replaced by the new ones shown. Figure 5.28, “Select master profile” shows this process using the example.

If the communication parameters later change, only the master profile needs to be edited. This will automatically change all other profiles related to the master profile.



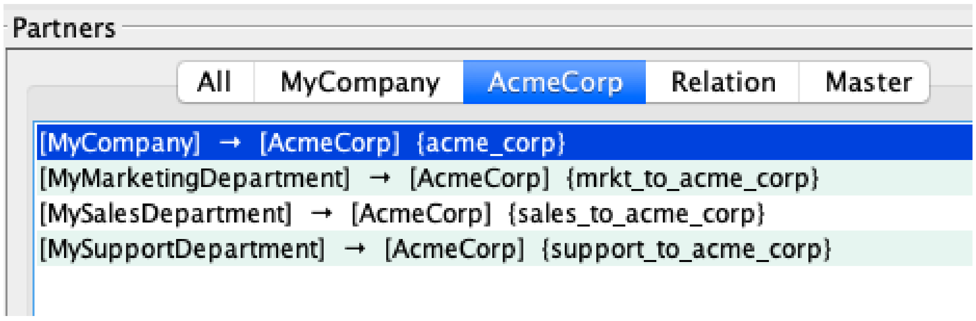
Partners
All | Master

[MyCompany] → [AcmeCorp] {acme_corp}
[MyCompany] → [Oscorp] {oscorp}
[MyCompany] → [StarkInd] {stark_ind}
[MyMarketingDepartment] → [AcmeCorp] {mrkt_to_acme_corp}
[MySalesDepartment] → [AcmeCorp] {sales_to_acme_corp}
[MySupportDepartment] → [AcmeCorp] {support_to_acme_corp}

Figure 5.29. Partner list with master and slave profiles

Profiles referencing other profiles appear in black font in the list. The master profile itself is highlighted in blue. You can use the “Master” tab to only view master profiles. Profiles dependent on master profiles will then be hidden.[46]

[46]De facto, here all profiles with explicitly set communication parameters appear. These are suitable to be chosen as master profiles. The heredity mechanism of communication parameters from a master profile is single-tiered. A profile which is a sub-profile of another cannot be selected as a master itself.



Partners

All | MyCompany | AcmeCorp | Relation | Master

[MyCompany] → [AcmeCorp] {acme_corp}
[MyMarketingDepartment] → [AcmeCorp] {mrkt_to_acme_corp}
[MySalesDepartment] → [AcmeCorp] {sales_to_acme_corp}
[MySupportDepartment] → [AcmeCorp] {support_to_acme_corp}

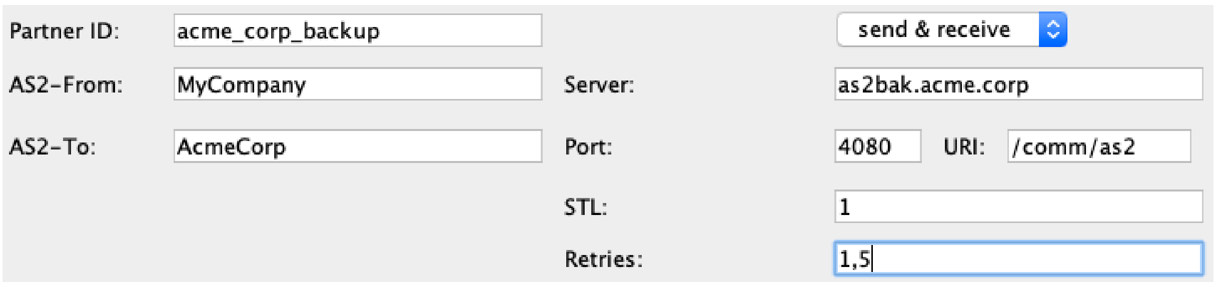
Figure 5.30. Selecting partner profiles via inner tabs

You can also use the tabs within the partner list for other selections. Select e.g. the master profile (`MyCompany, AcmeCorp`). The labels of the tabs will change accordingly. Then select e.g. the tab “AcmeCorp” to select all profiles related to company ACME Corporation.

5.16 Setting up backup connections

Your communication partner may have some redundant AS2 servers. You send your data to a primary server. If this primary server is down, data can be sent to an alternative server. This type of scenario can be configured in e-AS2 Enterprise by setting up backup connection profiles.

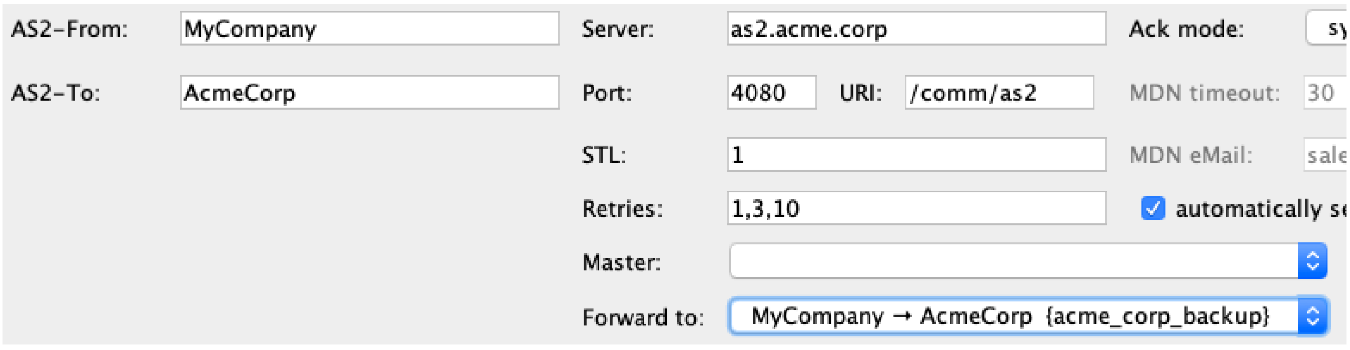
E.g. you could reach company ACME Corporation via an alternative AS2 server:



Partner ID: acme_corp_backup
send & receive
AS2-From: MyCompany
Server: as2bak.acme.corp
AS2-To: AcmeCorp
Port: 4080 URI: /comm/as2
STL: 1
Retries: 1,5

Figure 5.31. Setting up a backup connection

Now you can edit the primary profile and configure it to forward to the backup profile, should the primary connection not work.



AS2-From: MyCompany
Server: as2.acme.corp
Ack mode: sy
AS2-To: AcmeCorp
Port: 4080 URI: /comm/as2
MDN timeout: 30
STL: 1
MDN eMail: sale
Retries: 1,3,10
☑ automatically se
Master:
Forward to: MyCompany → AcmeCorp {acme_corp_backup}

Figure 5.32. Forwarding to backup profile

This will result in the following behaviour when sending data to ACME Corporation. You always put files meant to be sent to ACME into the directory `acme_corp`. Thus the parameters from the primary profile for communication are used. If transmitting data this way fails, first the configured retry pattern will apply. I.e. e-AS2 Enterprise will (in this example) retry after one minute, after three more minutes and finally after five more minutes.

If the file cannot be transmitted after all retries, the system will automatically switch to the configured backup profile. The communication parameters configured there will apply. In addition, the retry pattern configured there will apply.

A series of partner profiles can be cascaded by using this forward-to mechanism. The transaction will only be set to error status, if all retries have been exhausted for the last backup profile which does not itself reference yet another backup profile.

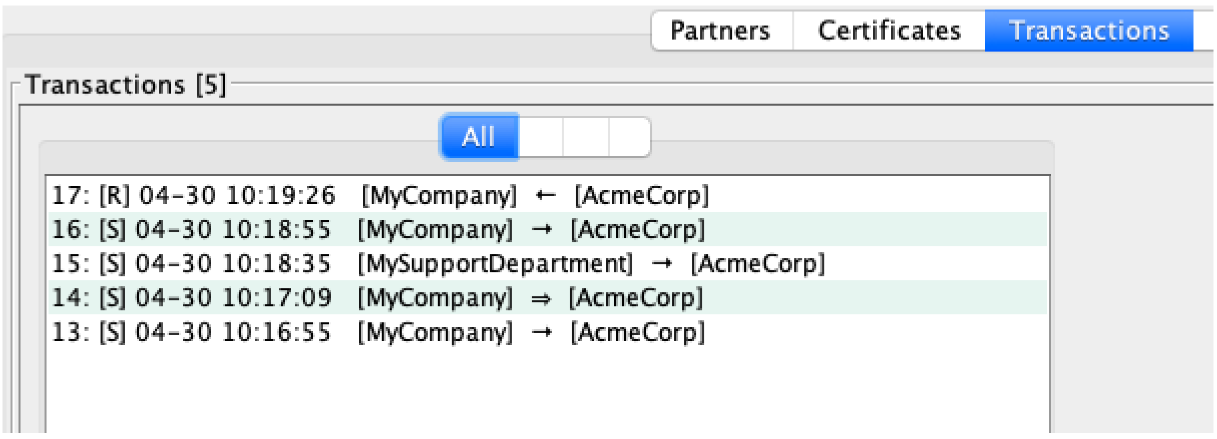
5.17 Managing transactions, troubleshooting

The primary source of information for all file transfers managed in e-AS2 Enterprise, send as well as receive, is the transaction list. Each file sent is reflected in an entry in the transaction list; the same applies to all incoming files. In both cases, the transaction list provides information on the success of the transaction and particularly allows for pending or erroneous transactions to be identified quickly.

5.17.1 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 interested in older transactions, you can use the drop-down box at the top right in the GUI to select a different date.

Transaction entries for sent files begin with an [S]. Entries for received files begin with an [R].



```
17: [R] 04-30 10:19:26  [MyCompany] ← [AcmeCorp]
16: [S] 04-30 10:18:55  [MyCompany] → [AcmeCorp]
15: [S] 04-30 10:18:35  [MySupportDepartment] → [AcmeCorp]
14: [S] 04-30 10:17:09  [MyCompany] ⇒ [AcmeCorp]
13: [S] 04-30 10:16:55  [MyCompany] → [AcmeCorp]
```

Figure 5.33. Transaction list with several entries

In both cases the time when the respective transaction occurred is shown, followed by the two AS2 identifiers involved, each in square brackets. The transmission direction is (in addition to the code letter at the beginning of the line) is also indicated by an arrow depicting the direction of data flow. For send transactions the arrow points to the right, for receive transactions it points to the left. Send transactions triggered manually using the “Send file” button in the GUI are marked with a double arrow.

The transaction list is sorted in reverse chronological order. So, the most recent transactions are displayed at the very top of the list.

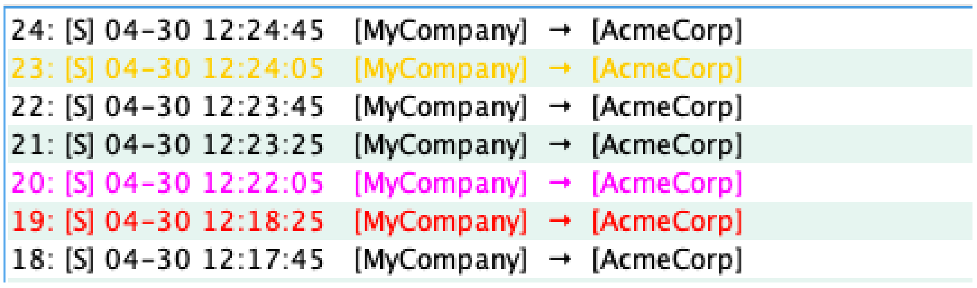
Important! When starting the graphical interface, the transaction list initially reflects the current processing state. After this, it is not automatically updated. If you later want to access the latest processing state with the GUI running, you will need to press the “Re-read list” button to show new and changed transactions. For reasons of efficiency only the part of the list in the currently visible section will be updated. In addition, all new transactions will also be loaded and internally added to the top of the list. To reload the complete transaction list for the selected date, tick the check box “all” before pressing the button.

The number of transactions currently loaded in the GUI always appears in square brackets to the top left of the list, in the header frame. So for the example in Figure 5.33, “Transaction list with several entries” a total of 5 transactions were loaded for the given date. Loading transactions happens in the backgound. So, for larger lists, the first chunk will be displayed quite fast, while the GUI is still busy loading more transactions. The transaction counter will increase until loading is finished.

Please note, the inner tabs of the transaction list feature similar selection options as the partner list.

5.17.2 Transaction list, colour coding

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



```
24: [S] 04-30 12:24:45  [MyCompany] → [AcmeCorp]
23: [S] 04-30 12:24:05  [MyCompany] → [AcmeCorp]
22: [S] 04-30 12:23:45  [MyCompany] → [AcmeCorp]
21: [S] 04-30 12:23:25  [MyCompany] → [AcmeCorp]
20: [S] 04-30 12:22:05  [MyCompany] → [AcmeCorp]
19: [S] 04-30 12:18:25  [MyCompany] → [AcmeCorp]
18: [S] 04-30 12:17:45  [MyCompany] → [AcmeCorp]
```

Figure 5.34. Colour coding in the transaction list

The four possible colours have the following meaning:

- Entries in *black* have successfully completed and require no additional review. For send processes this means the file was transmitted to the remote endpoint and a positive acknowledgement received. For receive processes this means the complete file was received and successfully matched to a partner profile. The file received can be found in the `in` folder of the interface.
- Entries in *red* are ultimately erroneous. I.e. the file to be sent could not be transmitted and there are no retries pending. So this file will no longer be sent without manual action. However, the transaction can be manually released again, see Section 5.17.3 (*Release for reprocessing*). This situation can only occur with send jobs.
- Entries in *magenta* are currently being processed. This can mean the file was just recently detected in the `out` folder and is currently actually being transmitted. Or it could mean one or several communication attempts have failed but additional attempts are still pending based on the configured retry pattern. The transaction will only be set to error once all attempts have failed, and the colour well then change to *red*. Or if the retry is eventually successful, the colour will change to *black*. The colour *magenta*, same as *red*, only applies to send jobs.
- Entries in *yellow* are similar to *red* entries. *Yellow* transactions are also in error status. Unlike *Red* transactions, a file transfer has already occurred. I.e. the error only occurred after the respective file was successfully transmitted or acknowledged. If the transaction belongs to a send job, it indicates the remote endpoint reported an

error in the MDN. So the file was received by the remote endpoint, but could not be processed for some reason. The MDN content should clearly identify that reason.

For receive transactions from the perspective of e-AS2 the following applies accordingly. e-AS2 was able to receive but not process the file for some reason. If the sender of a file requested an MDN, e-AS2 notified the sender of this situation, clearly stating that reason. A *yellow* transaction practically always requires contacting the communication partner to resolve the problem. Especially you will need to coordinate whether the respective file needs to be resent or the recipient will manually forward the file which was already received to processing.

========== ADDED (JK: 17.11.2022) ==========

Also, transactions that were manually stopped by the operator or auto-stopped by the server are coded in *yellow*, regardless of the transmission direction.

5.17.3 Release for reprocessing

As soon as a file becomes an ultimately erroneous entry appearing in the list in red, the associated file is removed from the `_work_` directory under the partner directory. If it doesn't already exist, a new directory named `_errors_` will be created next to the `_work_` directory. The file which was not transmitted will be moved here.

You now take action to correct the error. For example, you ensure the Internet connection is restored. Or you ensure the firewall configuration is corrected or extended to meet your needs. You then want to retry the failed transaction. To do so, select the respective entry in the list and click the “Release transaction” button.

This will first change this transaction to “released” status. After a while it will then be added for processing again. The file to be transmitted will automatically be moved from the `_errors_` folder to the `_work_` folder again.

All entries in red should be treated this way until none remain. There is also an option to batch release multiple transactions in one go. For details on this feature see Section 5.17.10 (*Search function*).

Note: For systems running mostly unattended where the transaction list is only reviewed occasionally, it's important to configure the retry pattern so transactions won't be changed to definitive error too quickly. Using retry patterns with infinite retry will prevent this status in general. See Section 5.14 (*Setting the retry pattern*) for details! It is also a good idea to set up operator mails in order to get alarms whenever transactions need operator attention. See Section 10.2.20 (*Email configuration*) for details!

5.17.4 Trigger immediate re-try

========== ADDED (JK: 28.11.2022) ==========

While there are still retries to be done for a given erroneous transaction, the “Release transaction” button remains deactivated. You need to wait for the next retry as defined by the retry pattern in the regarding partner profile. Sometimes, however, depending on the retry pattern this could mean quite some wait time.

In such cases it is possible for the operator to trigger immediate re-try. In order to do so, click the button “Re-try now”. Technically this will simply reset the retry counter, which sets the transaction to a state similar to the very beginning of it's life-time. This will result in the retransmission being executed very soon, typically withing half a minute.

5.17.5 Re-parse message

An erroneous receive transaction that appears in the list in yellow may under some circumstances be reprocessed without the sender being involved. This is the case, for example, if the first processing failed because no suitable partner profile was found. After you have added the missing partner profile, you can either stop the transaction and ask the sender to resend the data, or re-parse the raw message that has already been successfully received.

When re-parsing a message, processing is performed that is qualitatively completely identical to processing when a message is received. Ideally, new parsing ends with a successfully processed transaction. If, contrary to expectations, the cause of the error has not yet been corrected, the transaction can still be in error status and highlighted in yellow after re-parsing, so that it remains a candidate for repeated re-parsing

The transaction selected in the list is released for new parsing by pressing the “Re-parse message” button. Initially, only the status of the transaction changes. The actual reprocessing takes place independently of the GUI after a while by the server. The process is clearly documented by log messages.

5.17.6 Stopping transactions

In the event infinite retries was configured with the respective retry pattern, you may at some point wish to cancel further attempts for current transactions. The same applies to transactions with a limited number of retries, if you see no chance that further retries could solve the underlying problem with that transaction.

In this case, you can stop processing for the respective transaction by pressing the “Stop transaction” button. A stopped transaction is yellow in the transaction list.

In principle, any transactions which have not yet completed processing can be stopped early this way. If you select an entry from the transaction list which can be stopped, the “Stop transaction” button will become active. For completed transactions which can no longer be stopped, the button will remain inactive.

5.17.7 Release MDN

If transmission of an asynchronous MDN is pending for a transaction received, just as with sending payload, communication problems may arise. In this case the note `{MDNERR}` (incl. curly brackets) will be added to the end of the line of the entry in the transaction list. The retry pattern configured in the partner profile then applies. In the worst case, transfer of the MDN may ultimately fail and still not have succeeded after all repeat attempts have been exhausted. You can tell that hits happened by the check boxes “MDN error” and “MDN compl.” both being enabled in the transaction details.

Selecting this type of transaction will activate the “Release MDN” button. You can then – similar to the process for sending payload – release the MDN for retry later.

Alternatively, in this case you can also stop the transaction, which will cancel sending the MDN. (There is no separate button to stop sending the MDN. This can be done using the “Stop transaction” button.)

Note: Any forward-to profiles (backup connections) configured in the partner profile only work for sending payload, not for sending MDN.

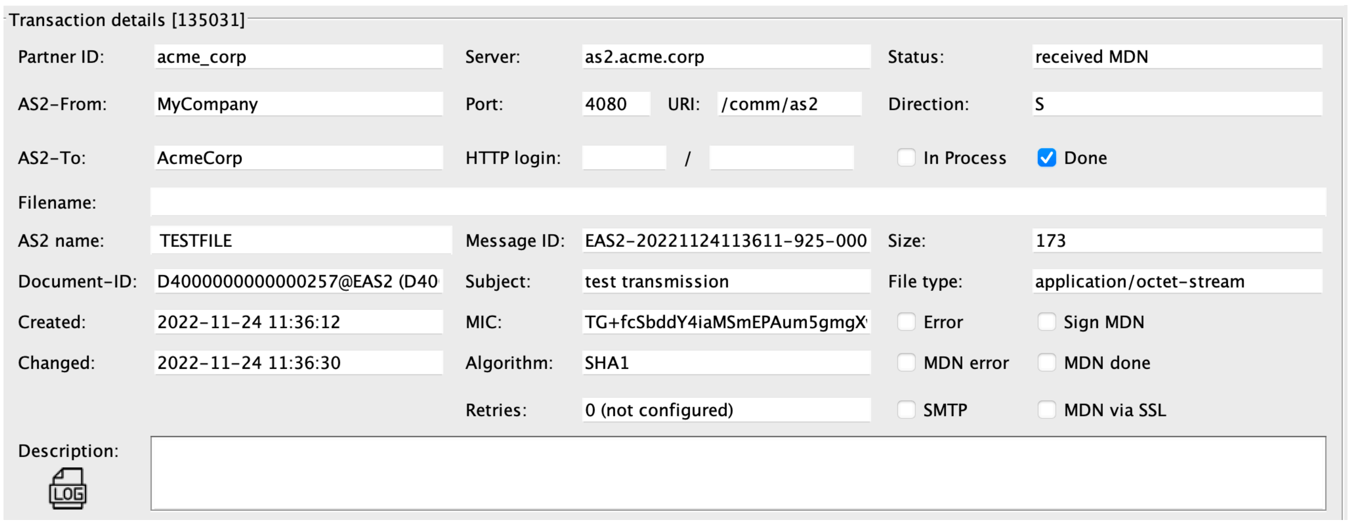
5.17.8 Download

As long as an object awaiting transmission, either payload data or an MDN, is still available on the server, the relevant file can be downloaded by the GUI client. In some cases this can be helpful to review the contents of the respective data. This should always be done when in doubt before stopping a transaction.

To do so, press the “Download” button. The button is context sensitive and either offers payload or an MDN file for download, as applicable.

5.17.9 Transaction details

At this point we will review some important information in the transaction details again. A systematic description of all fields can be found in Section 11.5 (*Transaction details*).



Transaction details [135031]

Partner ID:	acme_corp	Server:	as2.acme.corp	Status:	received MDN
AS2-From:	MyCompany	Port:	4080 URI: /comm/as2	Direction:	S
AS2-To:	AcmeCorp	HTTP login:	/	☐ In Process	☑ Done
Filename:					
AS2 name:	TESTFILE	Message ID:	EAS2-20221124113611-925-000	Size:	173
Document-ID:	D4000000000000257@EAS2 (D40	Subject:	test transmission	File type:	application/octet-stream
Created:	2022-11-24 11:36:12	MIC:	TG+fcSbddY4iaMSmEPAum5gmgX	☐ Error	☐ Sign MDN
Changed:	2022-11-24 11:36:30	Algorithm:	SHA1	☐ MDN error	☐ MDN done
		Retries:	0 (not configured)	☐ SMTP	☐ MDN via SSL
Description:					

Figure 5.35. Transaction details

Your first look at the transaction details should always be at the “Done” check box. If ticked, you will know e-AS2 Enterprise will not independently do anything else with this transaction. If “Error” is not also ticked, this is okay.

Transactions with “Error” and “Done” ticked require manual handling. These are the ones we mentioned above as “ultimately erroneous”. Transactions, marked erroneous but not yet done can be considered “temporary erroneous”. Here the software will retry to complete the transaction without user intervention.

The two check boxes “MDN done” and “MDN error” are to be treated quite similar. They implement the same logic but with respect to MDN transmission.

The field “Status” provides information on the current processing state of this transaction. The following values may appear with normal use:

new	This is the initial status of every transaction directly after detecting the file in the file system (for send jobs) or immediately after beginning transfer of an incoming file (for receive jobs).
new inf	This is the initial status in the event the INF interface was used (see Chapter 8 (*The INF interface*)).
split	This status is used if a new multifile transaction was split into individual single file transactions (see Section 11.1.41 (*Force single file, Interfaces*)).
sent	Send jobs change to this status once the data transfer is complete.

received	Receive jobs switch to this status once the data transfer is complete.
acknowledged	After an HTTP acknowledgement has been sent or received the transaction will change into this status.
received MDN	Send jobs will have this status once the remote endpoint has sent back an MDN.[47]
MDN to be sent	Receive jobs will change to this status if the other end requested an asynchronous MDN.
sent MDN	Receive jobs will change to this status once a file received has been acknowledged by an outgoing MDN.

There are different possibilities for the sequence of the various status values depending on the parameters configured for the transaction, shown in the table below.

Parameter	Status sequence
Sending without MDN	new – sent – acknowledged
Sending with synchronous MDN	new – sent – received MDN
Sending with asynchronous MDN	new – sent – acknowledged – received MDN
Receiving without MDN	new – received – acknowledged
Receiving with synchronous MDN	new – received – sent MDN
Receiving with asynchronous MDN	new – received – acknowledged – MDN to be sent – sent MDN

In addition to these normal status values, in some circumstances the following status values may also appear:

released	Once a transaction in error state has been released for reprocessing, it will first change to this status. The software will then treat it the same as a new transaction and follow the status change described above.
stopped	When an incomplete transaction is stopped prematurely it will change to this status and remain at it.
MDN released	When a transaction where the MDN has not yet been sent has been released for reprocessing the MDN, it will first change to this status. This is followed by the status “MDN to be sent” or “MDN sent” depending on the success of further processing.
frozen	When using the data fetch extension, send transactions can have this status if the other end is temporarily not available. For more information on this topic, read Chapter 6 (*Data fetch extension*).

The field “Retries” always shows how many retries have been made and the configured retry pattern. Along with the last edit time, shown on the left, this allows to predict when the next attempt will be made.

The field “Description” shows error messages if the transaction is in error state. For send jobs with MDN this can of course also be messages originating from the remote endpoint. You can

[47] This is regardless of whether it’s a synchronous or asynchronous MDN.

distinguish messages from your system from those from the remote endpoint by the prefix (`LOCAL:` or `REMOTE:`).

5.17.10 Search function

By default, the transaction list only shows transactions for the current day. After restarting the GUI, these are always the transactions “from today”. A flexible search function allows you to limit the total number of transactions to a manageable subset. This can be accessed with the “Search” button. The following dialogue will open.

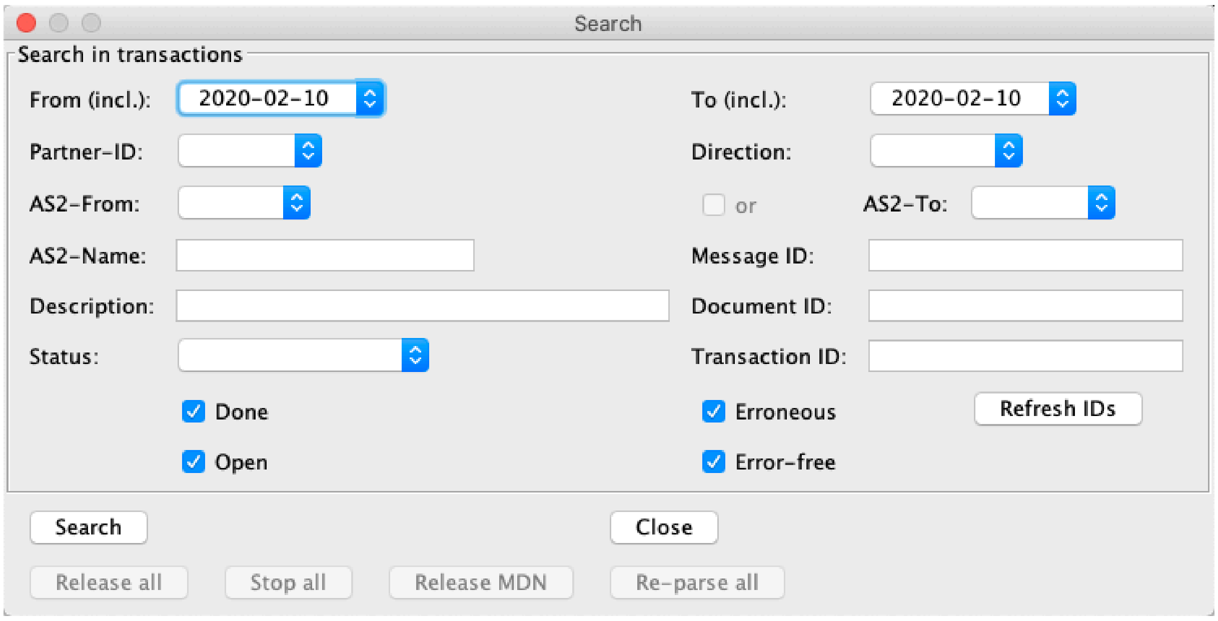


Figure 5.36. Search dialogue

You can now create a complex search of the entire data inventory from various criteria with the respective settings[48]:

From-To	With these you can define the desired time frame to search in.
Partner-ID	Select a specific e-AS2 Partner ID to only view transactions for this partner. After selecting a partner ID the entry options in the following line of the dialogue will be deactivated.
Direction	By default, the system will search transactions in both processing directions. If you’re only interested in one of the two directions, you can enter the respective filter.
AS2-From	Select a specific AS2-From identifier as part of the AS2 relation.
AS2-To	Select a specific AS2-To identifier as part of the AS2 relation. Once you have selected an AS2-From or AS2-To identifier, you will be unable to select a partner ID in the line above.
or	By default, the AS2-From and AS2-To identifiers are and-linked. I.e. the system will find all transactions where the AS2-From and AS2-

[48]For more information on the individual criteria, see Section 11.5 (*Transaction details*).

	To match the selected identifiers. Tick “or” if you wish to use an or-link instead.
AS2-Name	Enter any part of an AS2 name. This will find all transactions where the AS2 name contains this part. Within the specified partial string, you can use the ? character as a placeholder for any character. The character * can be used as placeholder for string of several arbitrary characters in a row.
Message ID	Enter any part of the message ID. This will locate all transaction where the message ID contains this part. The characters ? and * can be used as placeholders in the same way as in the AS2 name.
Description	Enter any part of a description (so: error message). This will find all transactions where the description contains this part. The characters ? and * can be used as placeholders in the same way as in the AS2 name.
Document ID	Enter any part of the document ID. This will locate all transactions where either the document ID or the original document ID contains this part. The characters ? and * can be used as placeholders in the same way as in the AS2 name.
Status	Select the desired status for the transaction you are looking for from the list.
Transaction ID	Enter a transaction ID. It will be searched for an exact match. So, there never will be more than one transaction found.
Done	Tick if you’re interested in completed transactions. (Done along with open will find all transactions.)
Open	Tick if you are interested in open transactions. (Open along with done will find all transactions.)
Erroneous	Tick if you are interested in erroneous transactions only. (Erroneous along with error-free will find all transactions.)
Error-free	Tick if you are interested in error-free transactions. (Error-free along with erroneous will find all transactions.)

All search criteria specified (except for alternative criteria for selecting the AS2 relation precluding both) can be combined arbitrarily. The more criteria are used, the further you are limiting the results.

You can leave the search dialogue open and at the same time navigate through the results or view individual entries, e.g. to determine additional criteria for further filtering. If you no longer need the search dialogue, click the respective button to close it.

Please note, the search criteria entered will remain active after closing the dialogue. I.e. “Re-read list” will now only show the new transactions matching the search criteria entered. To see all transactions again, press the “Clear search” button.

The drop-down lists for selecting partner identifiers do not refer to the partners configured in the system. These lists instead contain the identifiers which actually appear in the existing transactions. It is therefore quite possible that there are fewer entries in these lists than the total number of partner profiles configured. Conversely, the lists in the search dialogue may also contain entries which do not appear in any partner profile. This is the case when IDs in profiles are later changed or if connected communication partners initiate transfers with unconfigured IDs.

These selection lists in the search dialogue are populated once in the beginning and then remain unchanged. To add any missing IDs from additional transactions, press the “Refresh IDs” button. This will refresh the lists for Partner ID, AS2-From and AS2-To, as well as the two date selections.

5.17.11 Client time differs from server time

In globally operating companies with an appropriately designed IT infrastructure, it can happen that the e-AS2 server and the e-AS2 GUI client are operated in different time zones. The software automatically recognises this fact. The time stamps within the transaction list will then automatically be converted to the client time so the user can interpret the times according to his local time zone. The GUI also shows an additional selection box at the top right, next to the date.

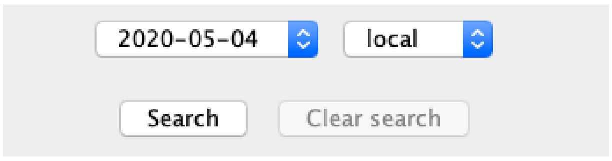


Figure 5.37. Select local time zone

The initial selection “local” indicates the GUI client shows all times per the local time zone. On request, this can also be switched to the server time. Simply select “Server” instead of “local”. This will refresh the transaction list. All timestamps shown now indicate the server time. If you let the mouse cursor rest on the selection box for a while, the name of the corresponding time zone is displayed.

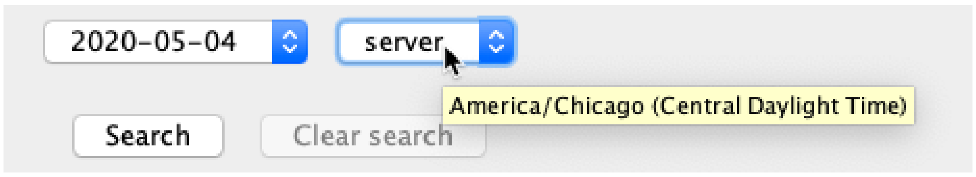


Figure 5.38. Select server time zone

Please note, the number of transactions shown for the respective selection (local time or server time) may differ. This will always show all transactions within the 24 hours of the selected day according to the selected time zone.[49]

[49]Depending on the locations of the server and GUI, one of the following two cases will always occur.

- There are transaction entries which – from a client time perspective – are older than the oldest entry in server time.
- There are transaction entries which – from a client time perspective – is further in the future than the newest entry in server time.

Here the gap to the server time can also exceed the date boundary. The GUI client considers this fact when building the date selection drop-down list. The available data is fetched from the server and adjacent days added across the board to allow each time zone to access all transactions, including those on the brink of the available period.

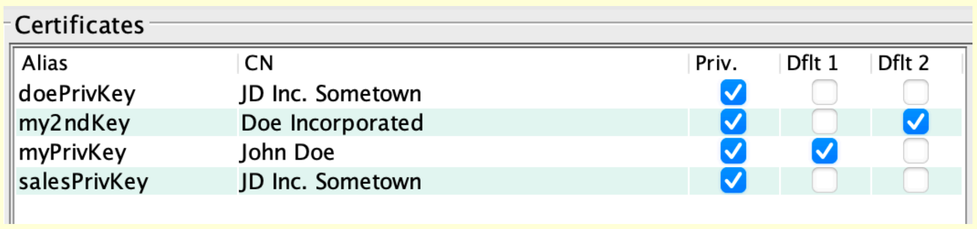
5.18 Encryption and signature, in detail

One of the key points of using the protocol is using AS2 cryptography. Handling the various aspects of encryption and signature well requires understanding the methods of certificate management in e-AS2. The following will look at this aspect in detail. We will pay special attention to the descriptions of the smooth migration methods for certificate expiration and renewal.

5.18.1 Private key

e-AS2 Enterprise allows you to manage any number of private key certificates. This applies regardless of whether you generated the certificates yourself or receive and import them from external sources. This provides for more flexibility with respect to the “own identity”.

========== CHANGED (JK: 06.02.2023) ==========



Certificates

Alias	CN	Priv.	Dflt 1	Dflt 2
doePrivKey	JD Inc. Sometown	✓		
my2ndKey	Doe Incorporated	✓		✓
myPrivKey	John Doe	✓	✓	
salesPrivKey	JD Inc. Sometown	✓		

Figure 5.39. Multiple private keys in the global certificate list

You can use different private key certificates to connect to different partners. In the scenario illustrated in Section 5.15 (*Select master profile*) you can for example use one certificate per department.

You will also use the option to manage multiple private key certificates when transitioning from a certificate which is about to expire to the new certificate. We will address this again later.

You will generally need to select one of the private key certificates as the standard or default certificate. This is ticked in the list, in the “Dflt 1” column. You can further specify a second default certificate as necessary. This will be ticked in the “Dflt 2” column.[50]

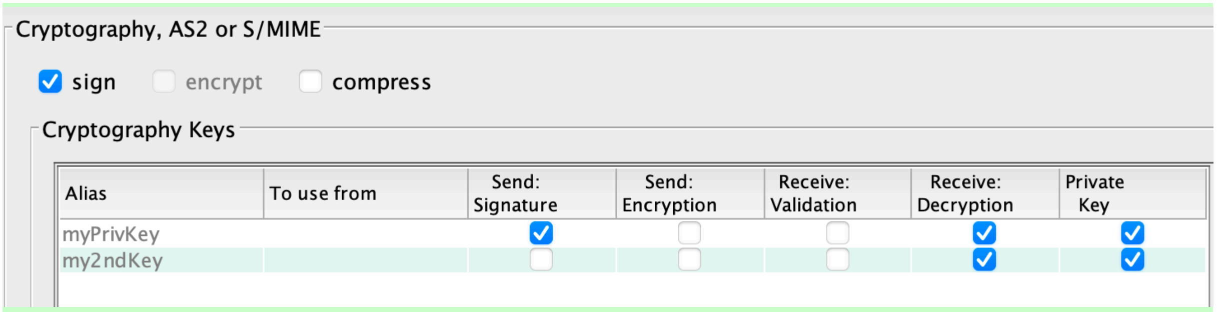
The default certificate defines your “standard identity”. Unless otherwise configured in the partner profile, the global default certificate will be used to sign messages. When receiving encrypted messages, the system will first attempt the decryption with the default certificate. If this fails, decryption will be attempted with the second default. Only when both fail will the system declare the decryption overall as failed.

========== ADDED (JK: 06.02.2023) ==========

You can see this logic clearly documented, when you open the partner profile's “Cryptography” tab and look at the list of keys displayed. Both default keys are listed and check marks indicate their role. The first entry (“myPrivKey”) is being used for signing outgoing messages. Both are used when it comes to decrypting received messages.[51]

[50]If you got only one private key certificate in the list, make this the first as well as the second default certificate. Both will then be ticked in the respective columns.

[51]Since these are global keys, they are greyed out in the list. This indicates that you cannot manipulate them directly from within the partner configuration dialog. But they still have influence in processing during communication with that partner. Therefore it makes sense that the show up in the list to give the user complete information at one glance.



Cryptography, AS2 or S/MIME

[✓ sign] encrypt compress

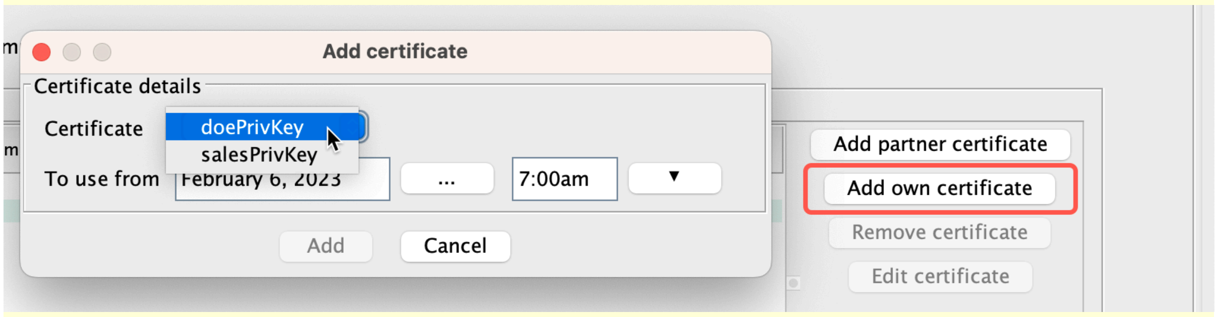
Cryptography Keys

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
myPrivKey		✓			✓	✓
my2ndKey					✓	✓

Figure 5.40. Displaying key usage in partner profile

The settings in the partner profile allow for further scenarios for using private keys.

========== CHANGED (JK: 06.02.2023) ==========



Add certificate

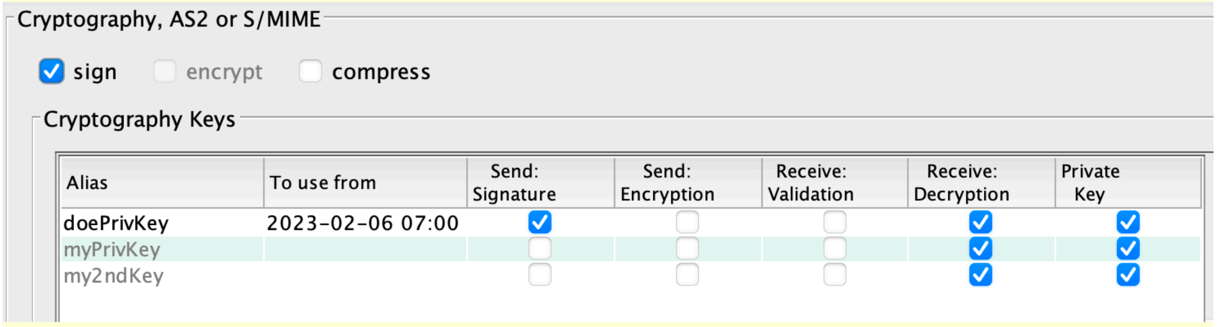
Certificate details

Certificate: doePrivKey / salesPrivKey

To use from: February 6, 2023 ... 7:00am

Figure 5.41. Adding own certificate to partner profile

Associate a new private key certificate with the partner profile by clicking the button “Add own certificate” and then choosing an entry from the drop down list. (Ignore the “To use from” information for now. We come to that further below.) After confirming the pop-up dialog the certificate will show up in the partner's certificate list.



Cryptography, AS2 or S/MIME

[✓ sign] encrypt compress

Cryptography Keys

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
doePrivKey	2023-02-06 07:00	✓			✓	✓
myPrivKey					✓	✓
my2ndKey					✓	✓

Figure 5.42. Own certificate added to partner profile

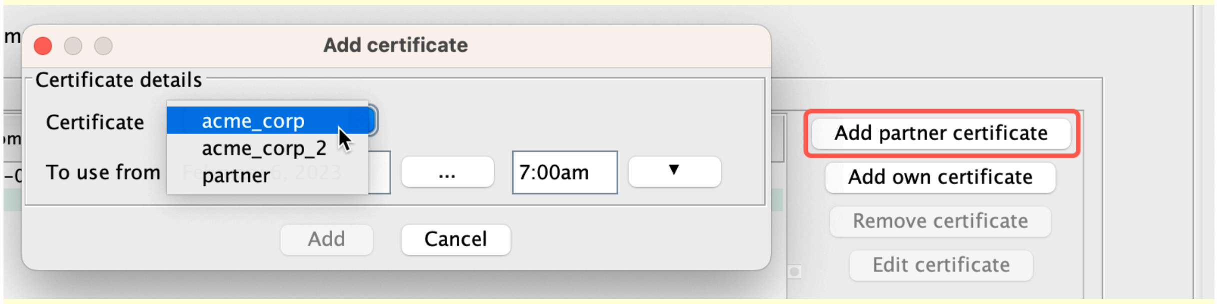
========== ADDED (JK: 06.02.2023) ==========

Observe, that the new certificate “doePrivKey” took the role “Send: Signature”. This is because certificates directly associated to a partner profile take higher priority than the default certificates. All three private keys are now candidates for decrypting received messages.

5.18.2 Public keys

========== CHANGED (JK: 06.02.2023) ==========

Your global certificate management will typically have exactly one public key per communication partner. A public key being present in the global certificate list initially has no impact on processing. In order for it to be used it must be associated with a partner profile after being imported into certificate management. This can be done in a similar way as for private keys. Click on “Add partner certificate”, then choose the desired certificate from the drop-down list.

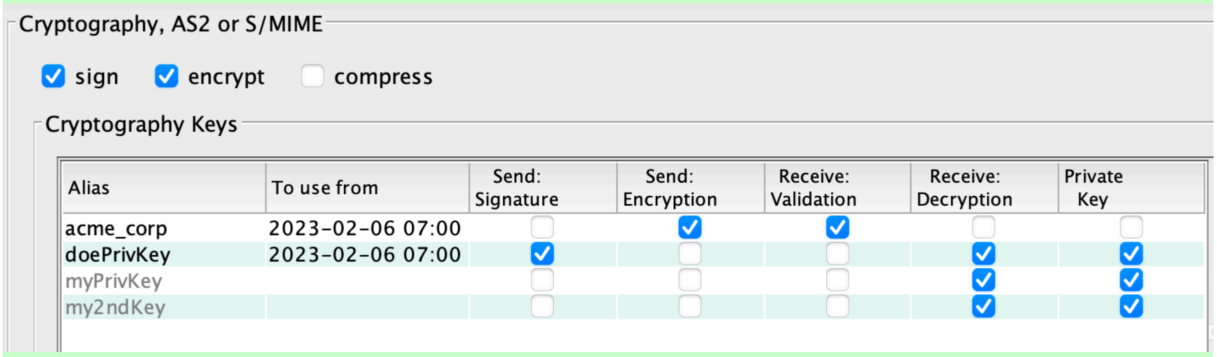


Certificate	acme_corp / acme_corp_2 / partner
To use from	... 7:00am ▼

Figure 5.43. Adding partner certificate to partner profile

========== ADDED (JK: 06.02.2023) ==========

The added certificate will then show up in the list. The certificate roles will be adjusted accordingly.


Cryptography, AS2 or S/MIME

✓ sign ✓ encrypt ☐ compress

Cryptography Keys

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
acme_corp	2023-02-06 07:00	☐	✓	✓	☐	☐
doePrivKey	2023-02-06 07:00	✓	☐	☐	✓	✓
myPrivKey		☐	☐	☐	✓	✓
my2ndKey		☐	☐	☐	✓	✓

Figure 5.44. Partner certificate added to partner profile

The new certificate “acme_corp” is marked as being used for encrypting outgoing messages. It is also used for validating incoming messages.

5.18.3 Automated certificate switch

========== ADDED (JK: 06.02.2023) ==========

When establishing PKI based encrypted message exchange, it is essential that the certificates involved have fairly short validity periods. This makes the platform robust against all kinds of malicious attacks.

However, the robustness achieved by this comes at a cost. Certificates that are about to become invalid must be replaced by new one with prolonged validity.

The process of replacing certificates typically involves manual work by operators. This incurs personnel costs.

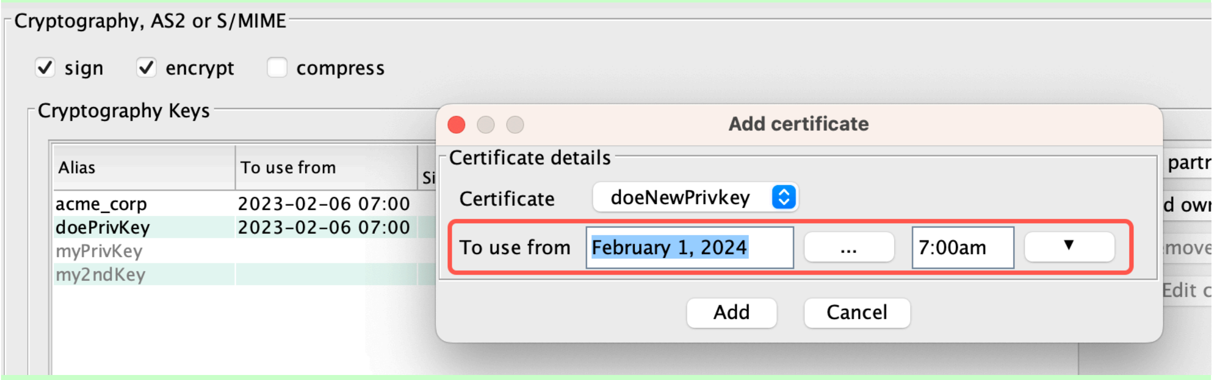
Another aspect of switching certificates is the necessity of synchronizing the process with persons at the communication partner's company. Ideally the public/private key certificate pairs should be switched to the renewed instances at the very same time on both sides. If this cannot be achieved this can lead to intermittent cryptography errors.

In e-AS2 automated certificate switches are done to support the user in this task. This is implemented by adding a “To use from” date/time to every certificate, either private or public, that is being associated with a partner profile.[52]

How to replace private keys with new versions

========== ADDED (JK: 06.02.2023) ==========

If your private key is about to expire, you can generate a new, follow-up version of the key at any time. It is placed in your global certificate list. You then associate it with the partner profile in question and define a “To use from” date in the future.



Cryptography, AS2 or S/MIME

[✓ sign] [✓ encrypt] compress

Cryptography Keys

Alias	To use from
acme_corp	2023-02-06 07:00
doePrivKey	2023-02-06 07:00
myPrivKey	
my2ndKey	

Add certificate

Certificate details

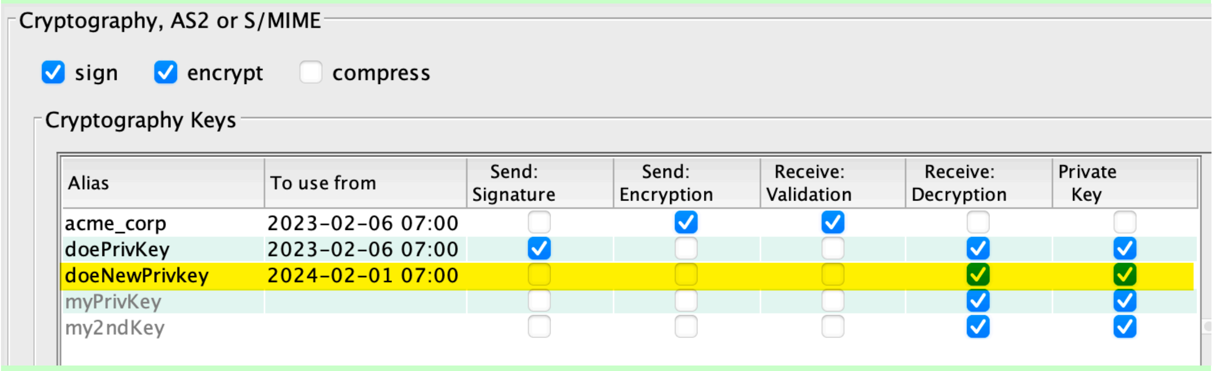
Certificate: doeNewPrivkey

To use from: February 1, 2024 ... 7:00am

Add Cancel

Figure 5.45. Adding new certificate with a future date

Say we add a new private key certificate that shall be used from 2024-02-01 07:00.



Cryptography, AS2 or S/MIME

[✓ sign] [✓ encrypt] compress

Cryptography Keys

Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
acme_corp	2023-02-06 07:00	☐	✓	✓	☐	☐
doePrivKey	2023-02-06 07:00	✓	☐	☐	✓	✓
doeNewPrivkey	2024-02-01 07:00	☐	☐	☐	✓	✓
myPrivKey		☐	☐	☐	✓	✓
my2ndKey		☐	☐	☐	✓	✓

Figure 5.46. New certificate added with future date

The new certificate (“doeNewPrivKey”) shows up in the list. It is not yet marked as the one to be used for signing outgoing messages. But – and that is important – it will be used as a potential candidate for decrypting received messages right away.

The public key belonging to the new certificate can then be given to the partner and they can start using it for encryption any time they want. You are already prepared for decrypting such messages, because the certificate is in the list.

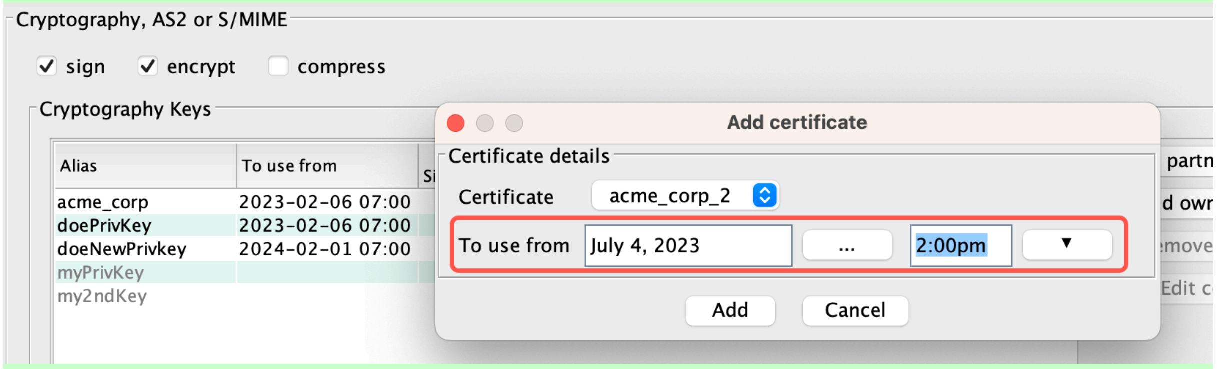
[52]This is not to be confused with the validity period of the respective certificate. It is an independent concept and no plausibility checks are in place to ensure consistent configuration in that regard. This lies in the responsibility of the user.

At February 1st 2024, 7 am automated certificate switch will take place in e-AS2. From that time on e-AS2 will automatically start using the new certificate for signing messages sent to that partner. Since the partner already got the public key, they will be ready to validate these signatures. From that moment on, when the certificate list in the partner profile is opened, the new certificate will be properly marked as the one being used for signing. The previous certificate will not be marked anymore.

How to replace public keys with new versions

========== ADDED (JK: 06.02.2023) ==========

For public keys the initiative is taken by your communication partner. They need to prepare a new private/public key pair on their side, when the certificate currently used is about to expire. They give the public key to you and tell you, as of when they want it to become active. You can immediately associate the partners new public key with the partner profile in question. You don't have to wait for the switch time to come, you define that time as the "To use from" date instead.



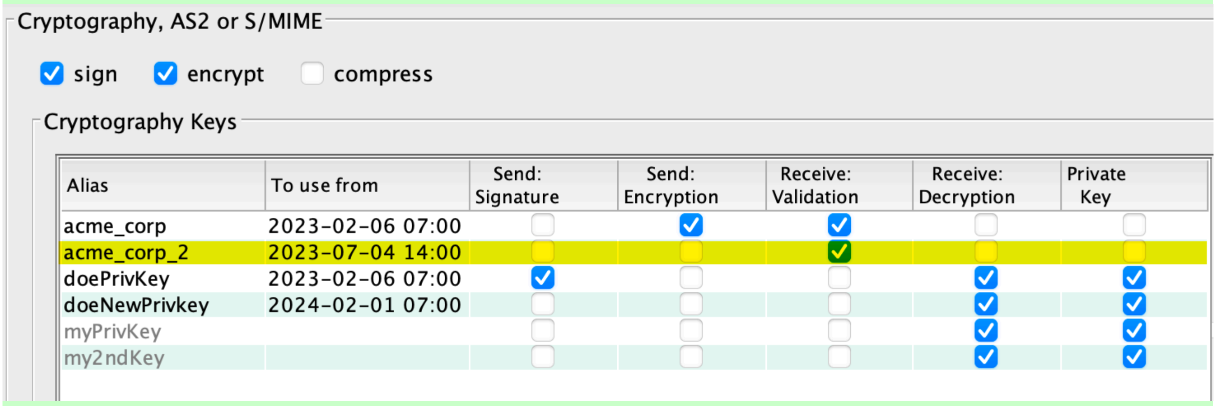
Alias	To use from
acme_corp	2023-02-06 07:00
doePrivKey	2023-02-06 07:00
doeNewPrivkey	2024-02-01 07:00
myPrivKey	
my2ndKey	

Certificate: acme_corp_2

To use from: July 4, 2023 ... 2:00pm

Figure 5.47. Adding new partner certificate with a future date

Say we add a new public key certificate that shall be used from 2023-07-04 14:00.



Alias	To use from	Send: Signature	Send: Encryption	Receive: Validation	Receive: Decryption	Private Key
acme_corp	2023-02-06 07:00	☐	✓	✓	☐	☐
acme_corp_2	2023-07-04 14:00	☐	☐	✓	☐	☐
doePrivKey	2023-02-06 07:00	✓	☐	☐	✓	✓
doeNewPrivkey	2024-02-01 07:00	☐	☐	☐	✓	✓
myPrivKey		☐	☐	☐	✓	✓
my2ndKey		☐	☐	☐	✓	✓

Figure 5.48. New partner certificate added with future date

The new certificate ("acme_corp_2") shows up in the list. It is not yet marked as the one to be used for encrypting outgoing messages. But – and that is important – it will be used as a potential candidate for verifying signatures of received messages right away.

So the partner can start using the corresponding private key for signing messages any time they want. You are already prepared for verifying such messages, because the certificate is in the list.

At July 4th 2023, 2 pm automated certificate switch will take place in e-AS2. From that time on e-AS2 will automatically start using the new certificate for encrypting messages sent to that partner. From that moment on, when the certificate list in the partner profile is opened, the new certificate will be properly marked as the one being used for encryption. The previous certificate will not be marked anymore.

Final remarks

========== ADDED (JK: 06.02.2023) ==========

Note, that although we are talking about automated certificate switch, in reality no certificates are switched in a strict sense of the word. It's just that the e-AS2 server starts using a different certificate for the task at hand. This technical approach comes with the advantage that it is not even necessary for the e-AS2 server to be up and running at the moment of the switch. Should the server be down, then it will simply start using the new certificate when it is started again.

5.19 Using HTTP/S

Some communication partners not only request secure transmission of the actual data, but also the HTTP headers associated with AS2 communication. In such cases you can use HTTP/S to communicate with the respective partner.

HTTP/S means the HTTP protocol is executed via SSL-encrypted TCP/IP connection. So nothing changes with respect to the actual AS2 exchange. However, the connection must be established differently. Here we will provide a rough outline to the extent you need to know for configuring e-AS2. For a detailed and exact explanation of the topic, please refer to the pertinent literature.

When establishing a connection for an HTTP/S session you address a dedicated port on the other end which has an application for accepting HTTP/S connections. The other end now sends a public key certificate to identify itself. The initiator of the connection is free to review the certificate and, if not satisfied, to immediately disconnect. In e-AS2 the public key certificate must be saved to certificate management. If this is not the case, the connection will not be established.[53]

If e-AS2 Enterprise does not accept the public key certificate when attempting to establish an HTTP/S connection, the unknown certificate[54] will automatically be saved to the `interface/in_cert` directory under a meaningful name. You will now be able to view this certificate using an external tool before deciding to import it into e-AS2. After importing the respective certificate[55] you should be able to establish a connection without a problem.

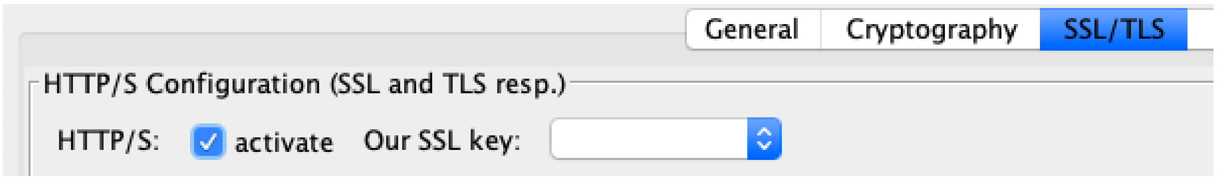
This unilateral authentication will suffice for establishing an SSL-encrypted TCP/IP connection. Although only the server was authenticated and the client was not, transport level encryption will be established anyway. Once the connection has been established, the normal AS2/HTTP protocol will run via the channel secured this way.

Some AS2 servers, however, also require the initiator of the connection to identify. This process is referred to as “Client Authentication” and also supported by e-AS2. We will now show how to configure e-AS2 Enterprise for using HTTP/S connections.

[53]This statement applies when using self-signed certificates. Generally speaking the issuer’s certificate or at a minimum an intermediate certificate or the root certificate from the certificate chain must be available.
[54]and possibly all certificates of the certificate chain
[55]or the root certificate in the certificate chain

5.19.1 Establishing HTTP/S connections (client perspective)

To set up a partner profile for HTTP/S, in the “General” tab, in the “Port” field, enter the respective port number provided by the contact on the other end. All SSL-specific configurations are entered in the “SSL/TLS” tab. Here, tick “Activate HTTP/S”. This is all you need at this time. All AS2 connections you establish will now be made via HTTP/S.



HTTP/S Configuration (SSL and TLS resp.)

HTTP/S: [✓ activate] Our SSL key:

Figure 5.49. Activate HTTP/S

If your partner requires “Client Authentication”, under “Our SSL Key” select the certificate you wish to use to identify yourself as the client for HTTP/S. You will need to export this certificate[56] and send it to the partner to be added to his AS2 application.

You can – from a purely technical perspective – certainly use the same certificate for HTTP/S authentication you are also using to sign messages (your identity). However, SSL often requires using certificates issued by an accredited Trust Center. In this case you will be using two different certificates, one for AS2, another for HTTP/S.

5.19.2 Accepting HTTP/S connections (server perspective)

In the basic configuration, right after installing the software, accepting HTTP/S connections is not activated in e-AS2. When starting the e-AS2 server, the following messages will appear:

Start server without HTTP/S

```
$ ./runserver.sh
using JRE 1.8.0_152
e-integration AS2 server starting ...
e-integration AS2 server 8.0.0 (CONFIG) ready.
e-integration AS2 server 8.0.0 (HTTP) ready.
```

To activate HTTP/S you will need to configure a few parameters in the `EAS2.properties` file. Here you will find a sample configuration.

```
# Additional settings for HTTP/S
#connection.https.port = <some port>
#connection.https.cert = <certificate alias>
#connection.https.clientauth = 1
```

Activate HTTP/S for incoming data by removing the hash mark at the beginning of the lines and adding the missing values. Example:

```
# Additional settings for HTTP/S
connection.https.port = 5081
connection.https.cert = sslcert
connection.https.clientauth = 1
```

Setting the property `connection.https.clientauth` to 1 as in the example above will enable “Client Authentication” globally for all incoming connection through this port. I.e. you

[56] or the root certificate of the certificate chain

expect the remote AS2 server to identify with an SSL client certificate. If you do not wish “Client Authentication”, set the value for this property to 0. SSL connections will then be accepted from any client.

Be sure to add the specified certificate (in the example the certificate with the alias “sslcert”) to the e-AS2 certificate management before activating HTTP/S. If the certificate cannot be found, then e-AS2 will not be able to start the HTTP/S thread. You will get a message like this.

```
e-integration AS2 server 8.0.0 (HTTP/S) could not be started. See
 log file!
```

The certificate configured in `EAS2.properties` is used to identify the e-AS2 server to clients establishing HTTP/S connections to it. It must therefore be available when starting the server. When properly configured and the certificate exists, the startup messages will be:

Start server with HTTP/S

```
$ ./runserver.sh
using JRE 1.8.0_152
e-integration AS2 server starting ...
e-integration AS2 server 8.0.0 (CONFIG) ready.
e-integration AS2 server 8.0.0 (HTTP) ready.
e-integration AS2 server 8.0.0 (HTTP/S) ready.
    client authentication: on
```

The e-AS2 server is now ready to accept HTTP/S connections, in this case using “Client Authentication”.

To accept connections both with and without “Client Authentication”, configure a second HTTP/S thread:

```
# Additional Settings for another HTTP/S thread (optional)
connection.https.port.2 = 5082
connection.https.cert.2 = sslcert
connection.https.clientauth.2 = 0
```

The startup messages will then be:

Start server with two HTTP/S threads

```
$ ./runserver.sh
using JRE 1.8.0_152
e-integration AS2 server starting ...
e-integration AS2 server 8.0.0 (CONFIG) ready.
e-integration AS2 server 8.0.0 (HTTP) ready.
e-integration AS2 server 8.0.0 (HTTP/S) ready.
    client authentication: on
e-integration AS2 server 8.0.0 (HTTP/S (2)) ready.
    client authentication: off
```

If you want to provide additional access via HTTP/S, e.g. to offer additional server certificates, then you can add up to four HTTP/S threads to the system.

5.19.3 Activating SNI (Server Name Indication)

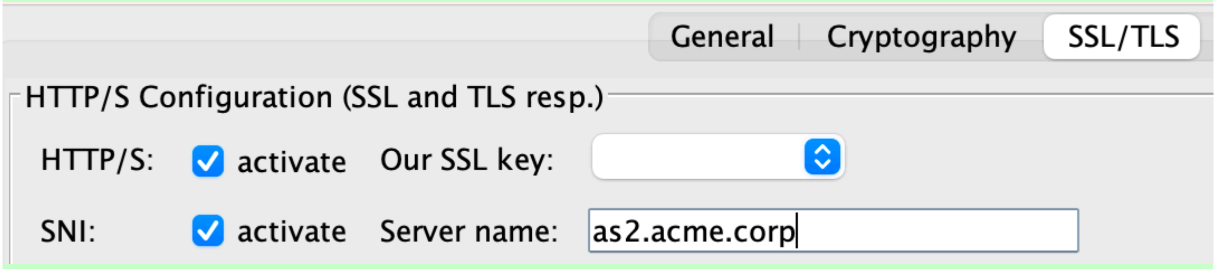
========== ADDED (JK: 16.11.2022) ==========

A more recent protocol feature of HTTP/S is the so called “Server Name Indication”, abbreviated as “SNI”. e-AS2 is prepared to use SNI for both outgoing and incoming connections. We will now show how to activate SNI in e-AS2 but we will not explain in detail what SNI is. If not sure, don't use it! You can find details on SNI on the Internet, eg. on https://en.wikipedia.org/wiki/Server_Name_Indication.

Activate SNI for outgoing connections

========== ADDED (JK: 16.11.2022) ==========

e-AS2 is generally prepared to use SNI for outgoing HTTP/S connections. If you wish to use it, it has to be enabled on a per partner basis. This is done in the partner profile.



HTTP/S Configuration (SSL and TLS resp.)

HTTP/S: ✓ activate Our SSL key:

SNI: ✓ activate Server name: as2.acme.corp

Figure 5.50. Activate HTTP/S

Tick the SNI checkbox on, to activate it. Then enter the server name that shall be sent to the remote endpoint with the SNI protocol extension. With this you will indicate that you intend to connect to a remote server with just that host name.

It lies completely in the responsibility of the remote server, whether or not it uses the Server Name Indication received. You may be confronted with one of three possible scenarios:

- The remote server may ignore the SNI extension. It's then completely irrelevant, whether you indicate a server name or not. And if you do so, it is irrelevant which name you indicate.
- The remote server may be satisfied and accept connections without getting SNI. However, if you sent an SNI server name, then it must match. The connection will be rejected, if it doesn't.
- The remote server may enforce SNI usage. It will not accept connections without SNI enabled. And obviously the server name sent must match the expectations of the server.

We recommend that you enable SNI only for partners where you are explicitly being asked to do so. Also, should you get SNI related error messages when trying to connect to a remote server without SNI enabled, then you should give it a try. Note, that the SNI server name, that's expected by the remote server, does not necessarily have to be identical to it's DNS name, that you entered on the General tab in the Server field.

Activate SNI for incoming connections

========== ADDED (JK: 16.11.2022) ==========

By default e-AS2 simply ignores the SNI protocol extension of incoming connections. If there are no other reasons against it, then every call will be accepted regardless of whether or not SNI is present and which server name is indicated.

This matches the first scenario in the above mentioned list of possible server behaviours. By setting dedicated properties in `EAS2.properties` you can make e-AS2 behave like in the second scenario.[57] If you want to do this, read on.

SNI can be enabled separately for every single of the up to four HTTP/S channels for incoming calls. This is done by adding an SNI related property to the respective channel configuration.

```
    connection.https.sni.server.name = as2\\.example\\.com
    connection.https.sni.server.name.2 = as2\\.example2\\.org
    connection.https.sni.server.name.3 = as2\\.example3\\.(org|de)
    connection.https.sni.server.name.4 = as2\\.example4\\.net
```

It is important to note, that regular expressions are used to define potentially multiple server names that would be accepted. In order for the backslash to escape the regex meaning of the dot, it has to be doubled.

After restarting the server it will be ready to evaluate SNI information on incoming HTTP/S connections. If the SNI does not match the configured regular expression, then the connection will be rejected. Connections without SNI, however, will still be accepted.

5.19.4 Debugging HTTP/S

The causes for problems related to SSL or HTTP/S may possibly be difficult to diagnose. e-AS2 Enterprise builds on the SSL implementation of the Java Runtime and the respective operating systems and forwards any error messages without filtering. Depending on the platform, these error messages may have varying degrees of accuracy and meaning. If the cause for SSL connection problems cannot be determined immediately, you can enable additional debug messages in the Java Runtime to help. To do so, start the server as follows:

Start server with debug option

```
$ ./runserver.sh -ssl-debug
```

You will then see very detailed messages with SSL context on the screen, allowing you to determine possible problems.

Please note, when using HTTP/S, the server and client authentication will occur before AS2-related protocol data is exchanged. This has the respective consequences for incoming connections. The system can only search for and locate partner profiles associated with an incoming connection after the so-called “SSL handshake” has been executed and completed successfully. This particularly results in not being able to add different server-side SSL certificates for each partner for technical reasons.

5.20 Using HTTP upload

You can use e-AS2 Enterprise to send or receive files via HTTP without using the AS2 protocol. It is not advisable making this accessible to the outside world via an unsecured connection. An SSL-secured connection, particularly when using client authentication, is a practicable and frequently used form of data transfer.

Omitting AS2 protocol objects, however, greatly limits the possibilities for parametrising this type of transfer. The data will be sent using a single HTTP POST and confirmed with an HTTP ACK. De facto, you only have the post request URI available for providing meta information.

[57] Note, that there is currently no option to make e-AS2 behave like in the third scenario. The e-AS2 server will never enforce SNI usage on incoming connections. This may be added in future versions of the software, though.

e-AS2 Enterprise handles the post request URI for HTTP uploads in s specific way, documented in the following sections. Contact your communication partner for which available options to use.

5.20.1 Configuration and URI interpreting for receiving via e-AS2

By default, receiving data via HTTP upload is disabled on the e-AS2 server. This function can be configured with the following properties in the `EAS2.properties` file.

```
# Additional settings for pure HTTP file upload (w/o AS2)
http.upload.allow = 1
http.upload.sslonly = 1
http.upload.uribase = /upload
```

`http.upload.allow` is used to enable (value 1) or disable (value 0) the function.

`http.upload.sslonly` specifies to only allow HTTP upload via SSL connections (so as HTTP/S upload). For security reasons we strongly recommend leaving this value at 1 and to force using SSL for HTTP-only uploads.

`http.upload.uribase` defines the start of the post request URI for incoming HTTP uploads. This distinguishes incoming HTTP uploads from incoming AS2 connections. Both types of connections are established using the same port. I.e. the value for `http.upload.uribase` must be different from `connection.http.uri` to allow for differentiation at this level.

When configuring `uribase` as described above, e-AS2 assumes the URI is configured as:

```
/upload/<from>/<to>/<name>/<subj>
```

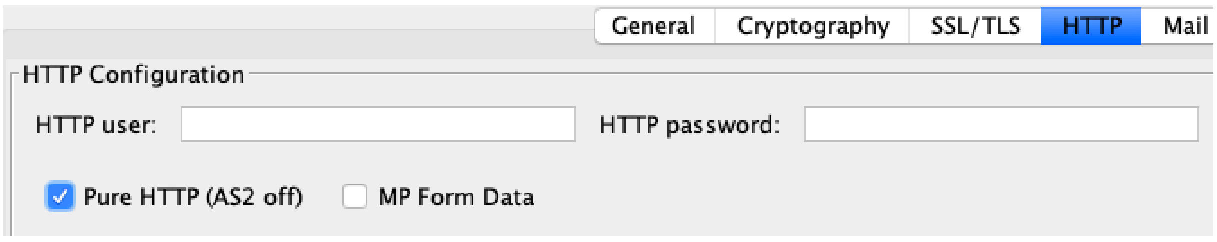
Where

<from>	The sender identifier (from the perspective of the partner connecting to you). This identifier is further interpreted the same as AS2-From.
<to>	The recipient identifier (from the perspective of the partner connecting to you). This identifier is further interpreted the same as AS2-To.
<name>	The name of the file (from the sender's perspective) to be uploaded to you.
<subj>	The subject for this process.

The last two components (file name and subject) are optional and may be omitted. The first two components (sender identifier and recipient identifier) are mandatory and are used to find the associated partner profile, as with data received via AS2.

If no partner profile can be found for an incoming file, the e-AS2 server saves it to the `orphan` directory.

To set up a profile for receiving data via HTTP-only upload, go to the “HTTP” tab and tick “Pure HTTP (AS2 off)”.



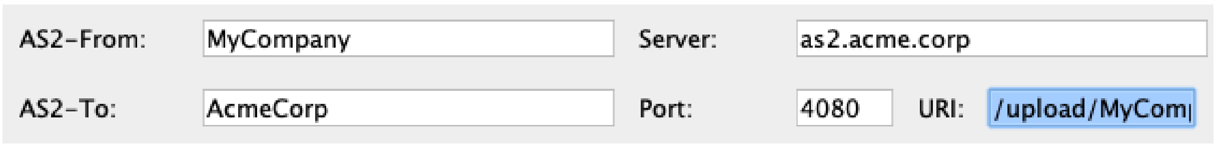
HTTP user:
HTTP password:
[✓] Pure HTTP (AS2 off) [ ] MP Form Data

Figure 5.51. Enable HTTP-only upload

This will disable all the configuration options which are only relevant for AS2. The partner profile for receiving messages via HTTP upload will be activated. The fields “AS2-From” and “AS2-To” will be used for matching the corresponding components in the URI.

5.20.2 Configuration and generating URI for sending via e-AS2

You can always send files to your partners via HTTP upload without needing to activate it in `EAS2.properties`. Activate “Pure HTTP (AS2 off)” in the partner profile and configure a suitable URI as expected by your communication partner.



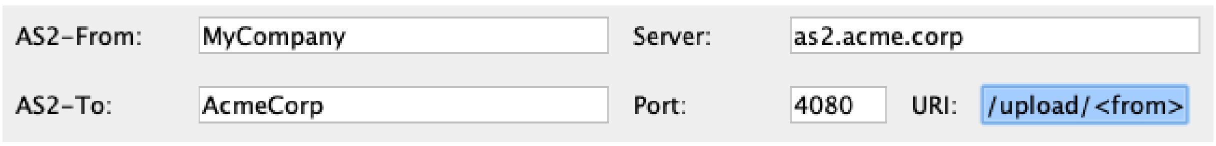
AS2-From:	MyCompany	Server:	as2.acme.corp		
AS2-To:	AcmeCorp	Port:	4080	URI:	/upload/MyCom

Figure 5.52. Set URI for HTTP-only upload in send direction

You would set something like

```
/upload/MyCompany/AcmeCorp/DataFile/ORDERS
```

as a fixed URI for that connection. As an alternative you can use placeholders to have the URI or any parts thereof dynamically generated by e-AS2 at runtime.



AS2-From:	MyCompany	Server:	as2.acme.corp		
AS2-To:	AcmeCorp	Port:	4080	URI:	/upload/<from>

Figure 5.53. URI with placeholders

I.e. you can use placeholders in the URI entry field the same as described in Section 5.20.1 (*Configuration and URI interpreting for receiving via e-AS2*). The following placeholders for establishing a connection will be replaced:

<from>	Is replaced with the contents in field “AS2-From”.
<to>	Is replaced with the contents in field “AS2-To”.
<name>	Is replaced with the actual name of the transmitted file (without path).
<subj>	Is replaced by the subject assigned to this processes (only when using the INF interface; see Chapter 8 (*The INF interface*)).

Using placeholders you would set something like

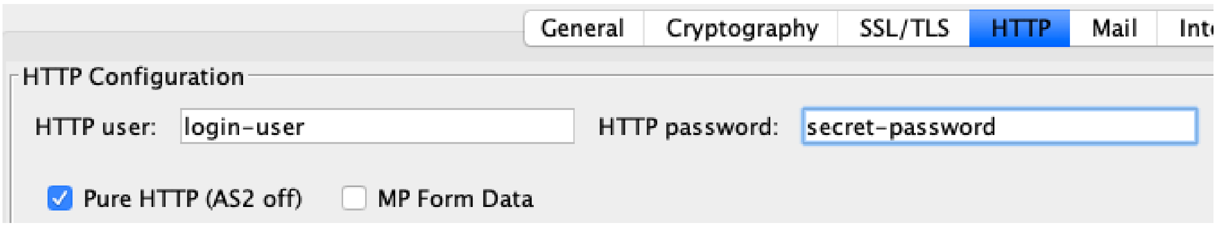
```
/upload/<from>/<to>/<name>/<subj>
```

as a dynamic URI for the connection.

This allows you to respond to requests from the other end with respect to forming the URI quite flexibly.

5.20.3 HTTP Basic Authentication

If the other end requires authentication via user name and password for establishing the HTTP connection, you can enter these in the designated fields in the partner profile.



HTTP Configuration

HTTP user: login-user   HTTP password: secret-password

☑ Pure HTTP (AS2 off)   ☐ MP Form Data

Figure 5.54. HTTP Basic Authentication

5.20.4 Multifile transmission

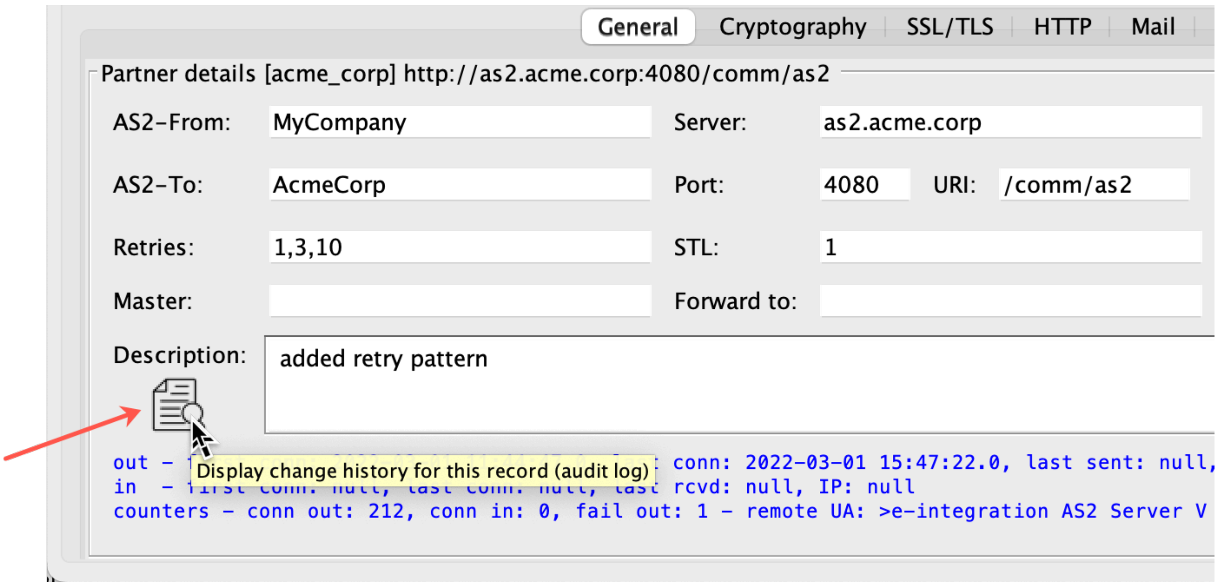
When individual files are sent, they are transferred 1:1 byte by byte unchanged to the other side. So the HTTP header is immediately followed by the payload. However, if a multifile send job is created in the INF interface (see Section 8.3 (*Multifile Mode*)), individual files pending transmission are compiled into a “multipart/related” MIME message and sent as such.

As for pure HTTP upload you can force multipart formattig even for single files by checking “MP Form Data”. Files are then wrapped into a “multipart/form-data” MIME message before being uploaded. This establishes the compatibility for servers unable to process files uploaded directly.

5.21 Audit-Log

During the daily work with the software in teams, the question often arises as to who changed what in the configuration and when. With this information, changes in the behavior of the software due to changes in the configuration can also be traced afterwards.

To support the user in this respect, all changes to partner profiles, certificates and mail server profiles are logged internally in e-AS2. The resulting change log (audit log) is available for retrieval in the e-AS2 GUI at any time. To retrieve the audit log for a configuration profile, select the profile in the list and click on the “audit log” icon to the left of the comment field in the configuration details.



Partner details [acme_corp] http://as2.acme.corp:4080/comm/as2

AS2-From:	MyCompany	Server:	as2.acme.corp		
AS2-To:	AcmeCorp	Port:	4080	URI:	/comm/as2
Retries:	1,3,10	STL:	1		
Master:		Forward to:			

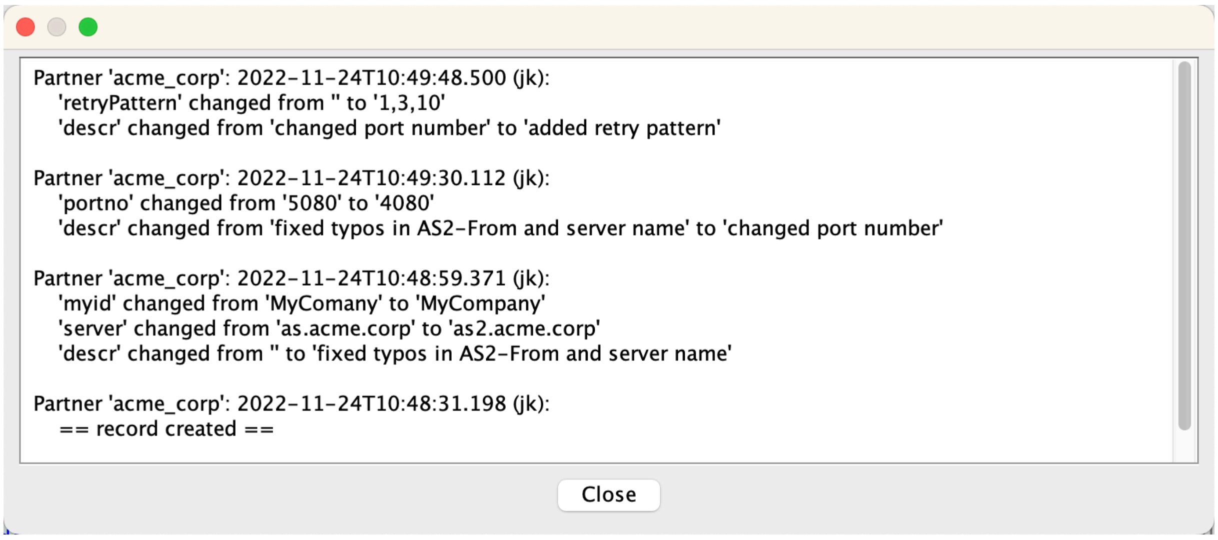
Description: added retry pattern

Display change history for this record (audit log)

```
out - f[illegible] conn: 2022-03-01 15:47:22.0, last sent: null,
in  - first conn: null, last conn: null, last rcvd: null, IP: null
counters - conn out: 212, conn in: 0, fail out: 1 - remote UA: >e-integration AS2 Server V
```

Figure 5.55. Display Audit Log

A new window opens in which the complete history of the selected profile is displayed.



```
Partner 'acme_corp': 2022-11-24T10:49:48.500 (jk):
   'retryPattern' changed from '' to '1,3,10'
   'descr' changed from 'changed port number' to 'added retry pattern'

Partner 'acme_corp': 2022-11-24T10:49:30.112 (jk):
   'portno' changed from '5080' to '4080'
   'descr' changed from 'fixed typos in AS2-From and server name' to 'changed port number'

Partner 'acme_corp': 2022-11-24T10:48:59.371 (jk):
   'myid' changed from 'MyComany' to 'MyCompany'
   'server' changed from 'as.acme.corp' to 'as2.acme.corp'
   'descr' changed from '' to 'fixed typos in AS2-From and server name'

Partner 'acme_corp': 2022-11-24T10:48:31.198 (jk):
   == record created ==
```

Figure 5.56. Audit Log

The entries are sorted chronologically backwards and are always in English, regardless of the GUI language. You close the audit log window in the usual way for the operating system you are using or by clicking on the “Close” button.

On the “Administration” tab you will find a button labeled “Audit log” that allows you to retrieve the global audit log across all configuration types. Information on deleted configuration profiles can also be found here.

6 Data fetch extension

How can e-AS2 Enterprise be configured to use the data fetch extension in e-AS2 Connect?

The so-called data fetch extension was implemented in e-AS2 Connect to support small and minimal installations, which in conjunction with e-AS2 Enterprise allows AS2 to be used on systems which are not always online. For a detailed description of this function from the perspective of the e-AS2 Connect user, please refer to the Connect user manual. Please first read said manual before returning here. It is assumed you are familiar with that text.

In order for the e-AS2 Connect user to use the data fetch extension the respective partner profile must be specially prepared in e-AS2 Enterprise. This special preparation consists of two components:

1. The cryptography certificate of the partner must be added to adequately secure access.
2. A specific character must be entered as the retry pattern.

6.1 Adding cryptography certificates

To import the partner certificate, proceed as described in Section 5.5.2 (*Import partner certificate*). Then proceed as described in Section 5.6 (*Sending encrypted files*) to assign the imported certificate to the partner profile. During assignment, “encrypt” will automatically be checked. This can be removed without a problem if the data transmission is not to be encrypted. The assignment of the certificate only serves the unambiguous authentication of the partner.

If the e-AS2 Connect user uses the data fetch extension incorrectly, e-AS2 Connect will generate a specific message encapsulating the current communication parameters. e-AS2 Enterprise will take the IP-address, the port number and the URI from this and enter them in the associated partner profile. This allows e-AS2 Connect to remotely change a partner profile in e-AS2 Enterprise it is currently communicating with.

To secure this critical process, e-AS2 Connect always sends the data fetch requirement message signed. The e-AS2 Connect user cannot influence this and cannot disable it. This allows e-AS2 Enterprise as the receiver of the data fetch requirement message to verify the signature and ensure only an authorised user is able to use this function.

6.2 Configuring retry patterns

To have the partner use the data fetch extension, a single minus symbol must be set as the retry pattern for this partner in e-AS2 Enterprise. A retry pattern `-` invokes the following behaviour in e-AS2 Enterprise.

If a new file is due for transmission to the partner, an immediate attempt is made to send this file. If the partner system is coincidentally online at the time, this transmission can be executed, completing the transaction. However, if the partner system is not online or now has a different IP-address[58], the transmission attempt will fail. In this case the transaction will now be put on hold indefinitely. In the transaction list it will be *magenta* in the status `frozen`.

Once the remote e-AS2 Connect user triggers data fetch requests, the communication parameters for this partner are updated in the local partner profile in e-AS2 Enterprise. In addition, any

[58]The data fetch extension also allows installations to be supported where the internet connection is only established on demand. With these configurations the internet provider typically dynamically assigns a new IP-address with every connection.

deferred transactions will automatically be released again. After a brief period e-AS2 Enterprise will then find the released transactions and transmit all the files associated with this partner. In order for this process to succeed, the remote e-AS2 Connect user must remain online for at least a few minutes after triggering the data fetch request until he has received all the pending files. He can then close the internet connection again, if he so desires.

6.3 Error messages

If e-AS2 Enterprise receives a data fetch request but is unable to process it properly, it will return a brief text error message to e-AS2 Connect, which will then appear in a message box in the GUI. The following two error messages which may appear due to configuration errors on your end are important.

6.3.1 406 Not Acceptable

If no minus symbol was entered as the retry pattern, e-AS2 Enterprise will return “406 Not Acceptable” to e-AS2 Connect.

6.3.2 401 Unauthorized

If a minus symbol was entered as the retry pattern but signature validation failed, e-AS2 Enterprise will return “401 Unauthorized” to e-AS2 Connect. The same applies if no matching partner profile was found for the data fetch request.

7 Interface for asynchronous MDN

e-AS2 Enterprise features an interface for generating user-controlled asynchronous MDN.

As already explained in Section 5.12 (*Requesting acknowledgements*), asynchronous MDN are transmitted to the sender of the file received in a separate session. The time which passes between the time the file is received and acknowledged is not defined by the protocol specification. This allows the option to forward received files to a subsequent application system and to receive a qualified processing status from it for delayed transmission to the sender of the file in form of an MDN. e-AS2 Enterprise implemented this mechanism through a specific file interface.

Normally you will not need to deal with this interface, since e-AS2 Enterprise automatically executes the necessary steps for generating asynchronous MDN itself. This creates a standard MDN with a generated text which returns the processing result *within e-AS2* to the opposite party. If this will suffice, you can skip this chapter. However, if you want to transmit information from the subsequent application in the MDN or generally want to influence the contents of the MDN, disable the automatic generation of asynchronous of the MDN in the partner profile. (Section 5.13 (*Auto-generate asynchronous MDN*) shows how.) You will then need to generate the asynchronous MDN described later in this chapter yourself.

7.1 How the file interface works for asynchronous MDN

Receiving a message with e-AS2 Enterprise is completed when a file pair is saved to the `in` directory of the interface section. Read details on this process in Section 5.4.4 (*The result file*) again. One file contains the actual message transmitted (payload data), the second file is a result file with information about the message transmitted. If the sender requested an asynchronous MDN, the attribute `MDN_REQUEST` will be set to `true` in the results file. The following processing program needs to read out this information and behave accordingly. The transaction will remain open until an MDN is generated. The assumption is the same occurs on the sender's end. His transaction will only be completed after the expected MDN has been received.

In the result file you will find the message ID for the associated transaction behind the attribute `MESSAGE_ID`. The message ID is a clear identifier for the transaction. It's required to match the asynchronous MDN with the transaction. The processing program analysing the result file and the data for further processing must now trigger sending the associated MDN. This is done by generating a so-called MDN trigger file. This file is to be saved to the directory `out_mdn` in the interface area.

If the directory interface for outgoing data is activated, a `out_mdn` directory will also be under each partner directory. This can be used equivalent with the global `out_mdn` directory. While MDN trigger files for all partners can be saved to the global directory, only MDN trigger files for the respective partner should be saved to the directories for the partners.

Two tasks are executed via the MDN trigger file. They tell e-AS2 Enterprise which open incoming transaction the MDN should relate to. They specify the desired contents of the MDN. To do this, structure the file as follows.

The first line of the file starts with

```
msgid:
```

in this exact syntax. This is followed by the message ID taken from the result file after receiving the data.

In the event you want to notify the other party of a processing error, i.e. generate a negative MDN, there will be an optional second line starting with

```
error:
```

again in the same syntax in lower-case. The remainder of this line will be added to the MDN in form of a coded error and transmitted to the other party. If the data received was processed successfully, i.e. you want to generate a positive MDN, omit the second line.

Warning MDN can also be generated in the same way. In this case, start the second line with:

```
warning:
```

The rest of the line is included in the MDN as coded warning information.

All other lines in the file will be added to the MDN as free text. Here you can place additional information that is meant to be taken note of by a person on the other side. However, keep in mind there is no guarantee this text will actually be analysed by the receiver of the MDN. This depends on the type of AS2 software being used on that end and its configuration, as well as any organisational provisions. When in doubt, please ask the party you are communicating with.

Here an example of a trigger file triggering a negative MDN:

```
msgid: EAS2-20200429042020-989-000002@192.168.200.122@komsrv
error: product-not-available
Unable to add order to application system.
The following products are not found:
        No. 4711
        No. 8199

Please review your order!
```

The first line identifies the transaction, specifying the message ID. The second line defines the error code (`product-not-available`). You are generally free in specifying the code, but will need to coordinate it with the recipient of the MDN to ensure the MDN can be processed accordingly on the other end. Lines 3 to 8 of the file are added to the MDN as text content. The idea behind this is for the error code to be preferably machine processable, whilst the free text is intended for a person viewing the MDN received for more detailed diagnosis in the event of an error.

Here an example of a trigger file triggering a positive MDN:

```
msgid: EAS2-20200429042020-989-000002@192.168.200.122@komsrv
Order added to our order management system.
The order will be shipped soon.

Thank you for your order.
```

Again, the first line identifies the transaction. Since the second line does not start with the keyword `error:`, a positive MDN will be generated. The entire text starting with line 2 will be added to the MDN as text content.

Note: The name of the MDN trigger file is irrelevant for processing. It only needs to be unique within the `out_mdn` directory.

7.2 Handling problems with the transmission of asynchronous MDN

In principle, the same problems can occur when transferring asynchronous MDN to the other party as with a transfer of user data. In the end, an MDN is only a file to be transferred to the partner. Please note, with asynchronous MDNs the communication parameters to be used are determined by the partner who sent the data. This particularly means these parameters may deviate from the parameters saved under the partner profile. So the fact the data transfer was successful does not necessarily mean the MDN transfer will also be successful. This applies regardless of whether the asynchronous MDN was auto-generated by e-AS2 or manually by the user. When e-AS2 auto-generates asynchronous MDNs, it is done by auto-generating MDN trigger files, so using the same interface as manually generated MDN.

If MDN transmission to the partner fails, the respective MDN file will be moved to the `_errors_` directory under `out_mdn`. The MDN can later be released for retransmission from the graphical user interface as described in Section 5.17.7 (*Release MDN*).

At any rate, after sending or attempting to send an asynchronous MDN, a meaningful message will be added to the “Description” for the respective transaction. There generally are three possible situations which are indicated this way:

1. MDN sending successful, MDN positive
2. MDN sending successful, MDN negative
3. MDN sending failed, MDN type irrelevant

In situation 1 the status will be set to `MDN sent`. The description field will contain the following text:

```
ASYNC MDN: sent [processed]
```

In situation 2 the status will also be set to “MDN sent”. The description field will contain the following text:

```
ASYNC MDN: sent [processed/error: ...]
```

In place of the dots will be the specific error message sent to the other party.

In situation 3 the status of the transaction will remain unchanged with `MDN pending`. I.e. the HTTP acknowledgement sent during the same session has been transmitted to the partner. However, MDN transmission is still pending. The description field will contain the following text:

```
ASYNC MDN: MDN transfer failed - ...
```

In place of the dots will be the reason for the MDN transfer failing. This can help you with diagnosing MDN transmission problems.

8 The INF interface

The INF interface is an alternative to the directory-based interface for sending files.

If you're using e-AS2 Enterprise in standard operating mode, sending and receiving files is asymmetric. Setting up transmission jobs uses many directories where the e-AS2 internal partner ID is taken from the directory name. Receiving files only uses a single directory, with meta information on the file to be received being saved in the result file.

On request the transmission of data can also be controlled comparable to receiving. In this case you will then only use one outgoing directory and (comparable to the receive interface) one file pair, where one file is pending for transmission to the remote party and the second file containing meta information, so the parameters required for processing the job.

Various functions (such as e.g. defining a subject or requesting a delivery report) can only be used if with this alternative interface. Thus, for users of e-AS2 Enterprise we generally recommend using the INF interface to gain more power and control over the process of sending files. The following shows how the INF interface works.

8.1 INF interface for outgoing files

Before you can use the INF interface you will first need to activate it. To do so, in `EAS2.properties` set the property `interface.out.inf-file` to `1`. As usual, after changing the settings you will need to stop and restart e-AS2 for the change to be applied. After restarting the e-AS2 server will now monitor the `interface/out` directory. You can tell because the file `_ALIVE_` and the directories `_work_` and `_errors_` have now been added.

The e-AS2 server will ignore any files in this directory where the name does not end in `.inf`. As soon as a file is found with a name ending in `.inf`, this file is considered to be an information file (hereafter referred to as INF file for short), which contains information about a requested file transfer. Here an example an INF file:

```
P_INT_ID     [acme_corp]
FILENAME     [DATEN]
AS2NAME      [ORDER827364.edi]
CONTENT_TYPE [application/edifact]
SUBJECT      [order no. 827364]
DOC_ID       [4711]
```

INF files have the same structure as result files, so lines with key-value pairs. Each line first contains a key stating the type of content in that line. This is followed by the associated value in square brackets. Specifically, the following keys may appear in the INF file.

P_INT_ID	The internal e-AS2 partner ID for the partner profile this send job should be assigned to.
AS2FROM	The AS2 From portion of the desired AS2 relation for this send job. (When using this, AS2TO must also be specified, and P_INT_ID omitted.)
AS2TO	The AS2 To portion of the desired AS2 relation for this send job. (When using this, AS2FROM must also be specified, and P_INT_ID omitted.)
FILENAME	Name of the file in the file system to be transmitted. The file must also be saved under this name in `interface/out`. Complete paths cannot be specified at this point.

AS2NAME	The name AS2 is to transmit to the other party as the file name. If this information is omitted, `FILENAME` will be applied as the file name. Otherwise, this allows you to assign a descriptive file name if the actual file name was possibly generated, is cryptic and meaningless to the other party.
CON-TENT_TYPE	A MIME type can optionally explicitly be specified for the file to be transmitted. To do so, add a line with the key `CONTENT_TYPE` to the INF file and set the desired MIME type as the value. Use one of the MIME types specified in the right column in the table in Section 5.3.3 (*Specifying the MIME type*).
SUBJECT	The subject of this data transmission. Here you can transmit additional parameters or information characterising the data transferred, or to control processing by the other end. The information is optional.[59]
DOC_ID	A clear identifier of the process from the perspective of the assigning entity. This could be e.g. an SAP document ID. This value is optional and is not transmitted to the remote party. However, it is a reference added to the result file added to the MDN received for this transaction. It will also appear in a delivery report which may have been requested. This is to allow an in-house application to easily associate an MDN or DR received with the previously assigned send jobs to convert it to a respective status update (also see Section 5.4.4 (*The result file*) and Section 8.2 (*Delivery Reports*)).
ORG_DOC_ID	Same as DOC_ID, but possibly assigned by the first system in the over all processing chain for this transaction. It may be identical to DOC_ID but it could be different.
REQUEST_DR	Set this value to `true`, if you want e-AS2 Enterprise to generate a delivery report after completing the send transaction. Learn more on this topic in Section 8.2 (*Delivery Reports*).
DR_ADD_MD-N_MESSAGE	Set this value to `true`, if you want e-AS2 Enterprise to accompany delivery reports with corresponding MDN messages. Learn more on this topic in Section 8.2 (*Delivery Reports*).
PERFORM_ER-ROR_HAN-DLING	With this value you can control the way, error handling is performed by e-AS2. This only applies to send transactions. Set this value to `limited`, if you want e-AS2 Enterprise to generate a delivery report directly after errors without the need to stop transactions manually. Learn more on this topic in Section 8.2 (*Delivery Reports*).
DRY_RUN	Set this value to `true`, if you want e-AS2 Enterprise to perform a dry-run. Data will then be sent to the configured e-mail recipient for dry-runs instead of the configured communication party. Learn more on this topic in Section 12.5 (*Workflow testing with dry-runs*).

So the desired relation for the job will alternatively be determined by specifying the e-AS2 internal identifier of the partner profile or by directly specifying the AS2 relation. A file name and a subject to be transmitted may be explicitly specified. Further processing in e-AS2 after analysing the INF file is then exactly the same as for the usual directory interface.

[59]If you wish or need to send a subject, you can only do so using the INF interface.

If an error occurs whilst analysing the INF file, the INF file will be moved to the `_errors_` directory. The data will remain in the out directory. Upon reactivating the transaction through the GUI, the INF file will be retrieved from the `_errors_` directory and analysed again.

Once analysis of the INF file has completed, it is removed immediately (even before beginning data transmission). It has been converted to a transaction entry in the e-AS2 database containing all the information from the INF file. The associated payload file is moved to the `_work_` directory. If a communication error occurs after this – just as for the directory interface – the data will be moved to `_errors_` and upon reactivating the transaction again moved from there back to `_work_`. Please also note that if the INF file disappears, this does not mean the transmission was successful.

Always first copy the data file to the out directory, then the INF file. Once e-AS2 detects the INF file, the data file must also be accessible to prevent an error.

8.2 Delivery Reports

The AS2 protocol acknowledges messages via MDN. The e-AS2 server receives an MDN from the other party and adds it to the `interface/in_mdn` directory for the user to review. Why do we also need delivery reports (DR) in addition to MDN?

It’s quite simple. MDN coverage is not necessarily complete. Meaning: You will not necessarily receive an MDN for every transaction. The MDN for a transaction will be omitted in two cases. If you don’t request an MDN in the partner profile or request an MDN by e-mail, the e-AS2 server will quite simply not receive an MDN, although the transaction was processed properly – as configured. If you stop a transaction before processing has completed, you may also not receive an MDN, depending on which stage the process was stopped.

We close this gap with delivery reports. As the name indicates, the main job of a DR is to confirm successful delivery of the data transmitted. This occurs regardless of whether or not an MDN was requested. However, the DR can also occur as an NDR (non-delivery report), documenting a failed delivery. Generally, if requested when the transmission job is added, e-AS2 Enterprise will generate a DR when the transaction is actually completed, so in the case of an error, normally not until the transaction has been stopped via the GUI.

Delivery Reports therefore provide you with final information on every transaction, without exception, for which you request it. This allows for reliable, automated external status tracking.

8.2.1 Requesting a delivery report

By default e-AS2 Enterprise does not generate DR's. If you wish to receive a DR, you will need to request the DR to be generated when passing the transmission job to e-AS2. To do so, in the INF file set the attribute

```
REQUEST_DR      [true]
```

If you request a DR for a transaction, you’re of course responsible for monitoring the `interface/dr` directory, reviewing the DR files in it, and deleting them afterwards.

You can request a copy of the MDN message to be added to the DR.

```
DR_ADD_MDN_MESSAGE      [true]
```

In this case an additional file is placed next to the DR file, that contains the complete MDN message incl. all header lines.[60]

[60] This functionality is independent of storing MDN in directory `interface/in_mdn`. However, it is recommended to turn them off when working with DR.

8.2.2 Controlling error handling for send transactions

As mentioned above e-AS2 will not generate DR files until the transaction is completed. In the case of successful transmission of the data (and having received a positive MDN, if requested) there is no question about it. A DR file will immediately be generated to signal successful processing of that transmission job. In the case of an error, though, the affected transaction is still under control of e-AS2, even after all retries have been done and the error flag has been set. The user can still release that transaction for re-transmission. So, there is no final processing result available yet. Only when you stop the transaction via e-AS2 GUI, it will be actually completed and taken out of control of e-AS2. A DR file signalling the error situation (also called NDR) will then be generated.

This behaviour we call “complete error handling”, which is also the default behaviour of e-AS2. There may be environments, however, where the users do not need the option to release failed send transactions for re-transmission. They then most probably also do not want the obligation to manually stop failed send transactions after all retries have been done. If this is the case, e-AS2 offers what we call “limited error handling”. If limited error handling is activated, failed transactions are automatically stopped and set to a “completed” state. NDR's will immediately be generated. Release for re-transmission will not be available. Note, that the user then has full responsibility for managing re-transmissions outside of e-AS2. Otherwise the affected messages will be lost.

Limited error handling is not available as a general setting, but must be requested on a per job basis. In the INF file the attribute `PERFORM_ERROR_HANDLING` must be set to `limited`, to activate limited error handling for that single job. Other possible values for this attribute are `complete` and `default`. The absence of this optional attribute in the INF file is equivalent to setting the value to `default`.

8.2.3 DR file structure

All delivery reports are saved to the `interface/dr` directory. The file name is `DR_nnnnn`, where `nnnnn` is the sequential number of the associated transaction. If the addition of the MDN message is requested, it is stored in `DR_nnnnn_MDN`. Delivery reports have the same syntax as result files or INF files, so a sequence of pairs with keyword and value.

Contents of a DR file, example

```
TRANS_ID       [36]
CREATED        [2020-05-18 13:21:23.691]
REPORTED       [2020-05-18 13:21:24.070]
P_INT_ID       [acme_corp]
FILENAME       [/Applications/e-AS2/interface/out/DATEN.496]
AS2NAME        [ORDERS_324623]
SUBJECT        [PURCHASE ORDER]
DOC_ID         [D63466]
ORG_DOC_ID     [O63466]
STATUS         [3]
ERROR          [0]
DESCRIPTION    [null]
```

We will now explain the various contents of the DR file.

TRANS_ID	This is the sequential number of the respective transaction. This data is for information purposes only. The value is not suitable to use as

	a reference number for identifying the transaction, since when creating a new transmission job the system does not provide the number generated for the transaction.
CREATED	The time the transaction was created.
REPORTED	The time the delivery report was created. Depending on how processing the transaction goes, this time can be significantly later than the time it was generated, in some cases even several days.
P_INT_ID	The partner's internal identifier in e-AS2 Enterprise this transaction relates to.
FILENAME	The complete path to the transmission file in the file system. This is always historical information, as by definition this file will no longer exist when the DR is generated.[61]
AS2NAME	The name of the transmitted file provided to the other party via AS2 protocol.
SUBJECT	The subject for this transaction, if applicable. If no subject was defined in the INF file when the job was created, the value `null` will appear here.
DOC_ID	The document identifier assigned when the send job for this transaction was created. This value is essential and is used to positively assign the DR to the original process in your application system where the send data was generated. If you don't use document identifiers, you may not be able to use the e-AS2 delivery reports purposefully.[62]
ORG_DOC_ID	Yet another (optional) document identifier that may have been assigned when the send job was created.
STATUS	This is the final status of the transaction in e-AS2. You can find a detailed representation of the various status values in Section 5.17.9 (*Transaction details*). However, the status is only saved to the DR in numerical code. The following values may appear. 2 - Transaction confirmed via HTTP Ack. (No MDN received.) 3 - Transaction confirmed via MDN (synchronous or asynchronous). 5 - Transaction stopped via GUI, i.e. was not completed during regular processing.
ERROR	For transactions which were processed correctly and successfully, this value will be equal `0`. For transactions with errors the value will be `1`.[63]
DESCRIPTION	This is the content of the description field for the transaction at the time the DR was generated. In the event of an error, this will provide information on the cause for the error.

[61] It was either transmitted successfully and then deleted, or removed when the transaction was stopped.
[62] The only other alternative for assignment is the file name (attribute `FILENAME`). You will then need to ensure this name is globally distinct. In addition, you will need to add the name to your application so it is available for assignment after receiving the DR.
[63] Delivery reports for transactions with errors will only be generated if processing of the respective transaction was stopped via GUI.

8.3 Multifile Mode

e-AS2 Enterprise fully implements AS2 protocol versions 1.0 to 1.2. This involves supporting the optional multifile mode. I.e. e-AS2 Enterprise is able to receive as well as send AS2 messages with several file attachments.

Receiving several files during an AS2 session is generally enabled without user interaction. If the sender transmits several files during one session, e-AS2 will receive them and provide them to the incoming interface. This is where an advanced type of result file is used.

Sending several files during one AS2 session requires the use of an advanced type of INF file. Please coordinate this with the responsible person at the other end. Since this is an optional protocol feature, there is no guarantee the AS2 software on the other end will be able to process multifile transfers correctly.

8.3.1 Multifile receiving, advanced result file

In regular AS2 mode (one file per session) you will have two files for each transaction in the incoming directory.

Single file receiving, example

```
RECV-20200518133925-983-000006
RECV-20200518133925-983-000006.res
```

The first file contains the data received. The second file (with the file extension `.res`) is the result file containing the meta information for this transaction.

In multifile mode you can have more than one file with payload. In this case, e-AS2 Enterprise will add a sequential number to the file name generated.

Multifile receiving, example using three files

```
RECV-20200518134223-809-000008-0
RECV-20200518134223-809-000008-1
RECV-20200518134223-809-000008-2
RECV-20200518134223-809-000008.res
```

In this example there are three files with payload, numbered sequentially from 0 to 2, and again a result file. In this case the result file will contain three `FILENAME` lines and three `CONTENT_TYPE` lines. Also a `CONTENT_ID` line will be added per file.

Result file for multifile receiving, example

```
INST_NAME     [Integration Test]
RECEIVED      [2020-05-18 13:42:23.808]
ACKNOWLEDGED  [2020-05-18 13:42:23.809]
AS2FROM       [AcmeCorp]
AS2TO         [MyCompany]
FILENAME      [invoice4242.edi]
FILENAME      [invoice4242.pdf]
FILENAME      [invoice4242.p7s]
MESSAGE_ID    [EAS2-20200518084223-725-9-000007@192.168.200.122-komsrv]
SUBJECT       [SUBJECT]
CERT_SERIAL   [null]
```

```
CERT_ALIAS    [null]
P_INT_ID      [acme_corp]
TRANS_ID      [39]
DOC_ID        [D4000000000000005@EAS2]
ORG_DOC_ID    [D4000000000000005@EAS2]
STATUS        [1]
ERRORCODE     [0]
DESCRIPTION   [null]
CONTENT_TYPE  [application/edifact]
CONTENT_TYPE  [application/octet-stream]
CONTENT_TYPE  [application/octet-stream]
CONTENT_ID    [body-part-0@a199d3e7-64a9-4ed5-be31-6dbc7bfcc180]
CONTENT_ID    [body-part-1@a199d3e7-64a9-4ed5-be31-6dbc7bfcc180]
CONTENT_ID    [body-part-2@a199d3e7-64a9-4ed5-be31-6dbc7bfcc180]
MDN_REQUEST   [false]
```

The example shown is fictitious, where three files were received for an invoice. First the invoice in EDIFACT format. Then the same invoice as PDF file. Finally an electronic signature for the invoice. The three `FILENAME` lines, the three `CONTENT_TYPE` lines[64] and the three `CONTENT_ID` lines are to be analysed in order, top to bottom, and belong to the payload files numbered consecutively ending in 0, 1 and 2, in this order.

8.3.2 Multifile sending, advanced INF file

To create a send job for several files, you need to add additional `FILENAME` and `AS2NAME` information to the INF file.

Send job for several files, example

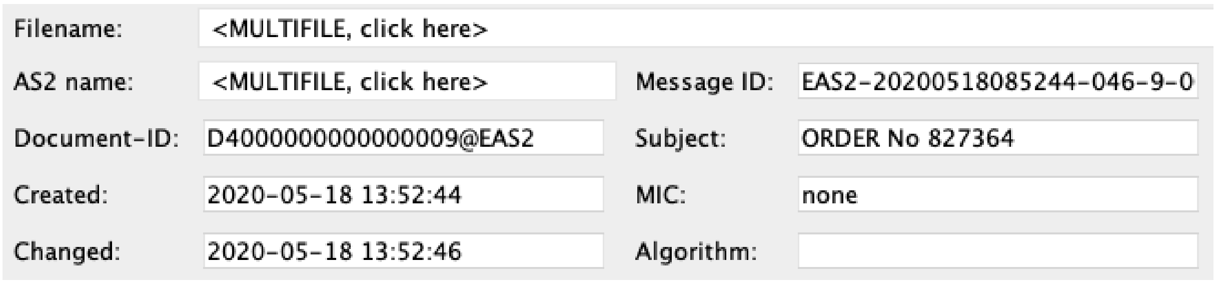
```
P_INT_ID       [acme_corp]
FILENAME       [DATEN.1]
AS2NAME        [ORDER827364.edi]
CONTENT_TYPE   [application/edifact]
FILENAME_2     [DATEN.2]
AS2NAME_2      [ORDER827364.pdf]
CONTENT_TYPE_2 [application/octet-stream]
FILENAME_3     [DATEN.3]
AS2NAME_3      [ORDER827364.txt]
CONTENT_TYPE_3 [text/plain]
SUBJECT        [ORDER No 827364]
DOC_ID         [4711]
```

The name attribute for the additional entries must have a sequential number after an underscore, starting with the 2nd. If e-AS2 detects INF files with more than one `FILENAME` entry, it will automatically generate a multifile transfer.

8.3.3 Displaying multifile details in the GUI

The configuration GUI for e-AS2 shows no file name and no AS2 name for multifile transactions – regardless of the direction of transmission. Instead, an explanation appears.

[64] Please note, this information may also occur with attached file names depending on the AS2 software the sender of the data uses: `[application/octet-stream; name=invoice4242.edi]>`. This possible variation must be considered when implementing any automatic processing of result files.



Filename:	<MULTIFILE, click here>		
AS2 name:	<MULTIFILE, click here>	Message ID:	EAS2-20200518085244-046-9-0
Document-ID:	D4000000000000009@EAS2	Subject:	ORDER No 827364
Created:	2020-05-18 13:52:44	MIC:	none
Changed:	2020-05-18 13:52:46	Algorithm:	

Figure 8.1. A multifile transaction

Pressing the information text will open a new window containing the detail information about the individual files for this transaction.

8.4 Command line programs

e-AS2 Enterprise includes two command line programs which will assist you with generating INF files.

Creating a simple job

```
$ as2snd
usage: as2snd <pid> <physfile> [[<virfile>] <subject>]
```

The program `as2snd` will have the following parameters.

<pid>	The internal partner identifier for the job.
<physfile>	The complete path (absolute or relative) where the physical file to be transmitted is located in the file system.
<virfile>	The virtual file name (AS2 name) to be transmitted to the other end for this file. (optional)
<subject>	The desired subject for the transmission. (optional)

The program is located under the e-AS2 Enterprise installation directory and must be started from there. It generates an INF file from the specified parameters and copies the payload file as well as the INF file to the interface.

Creating a multifile job

```
$ as2multi
usage: as2multi <pid> <subject> <physfile> <virfile> <content-type> [<physfile>
 <virfile> <content-type> ...]
       <content-type> may be '-' to set the default content type
```

The program `as2multi` will have the following parameters.

<pid>	The internal partner identifier for the job.
<subject>	The desired subject for the transmission.
<physfile>	The complete path (absolute or relative) where the physical file to be transmitted is located in the file system.
<virfile>	The virtual file name (AS2 name) to be transmitted to the other end for this file.
<content type>	The MIME type to be transmitted to the other end for this file. Here, enter e.g. `text/plain` or `application/edifact`. Enter a minus sign as the parameter to use the default content type.

Any number of additional pairs of <physfile> and <virfile> can be added for additional files.

The program is located under the e-AS2 Enterprise installation directory and must be started from there. It generates an INF file from the specified parameters and copies all payload files as well as the INF file to the interface.

Note! These command line programs are not intended for productive use, but solely as illustrative material to assist with learning and to better understand the INF interface.

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 erronous.
#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 be available but not yet used. The use of this mechanism can be enabled separately, per communication partner, as shown in Section 9.2 (*Turning on command on receive in the partner profile*).

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`[65] 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.

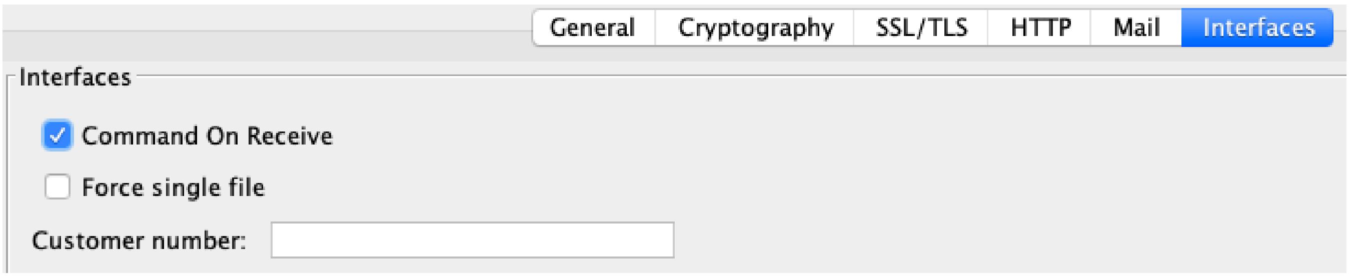
The property `command.wait.timeout` in `EAS2.properties` defines this timeout in seconds. If the invoked program does not finish within the specified time, e-AS2 will stop to wait and the external process will be terminated. This will be considered to be a failure to pass the data to the subsequent system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

9.2 Turning on command on receive in the partner profile

Activating command on receive in `EAS2.properties` will initially not change the server's behaviour. However, it puts e-AS2 into the basic readiness to invoking commands on receive.

[65] Or `process_message.bat` when installing in MS Windows.

This readiness is reflected in the GUI by the fact that an additional check box will appear in the the partner profiles, as shown in Figure 9.1, “Turning on command on receive for a given partner”.



Interfaces

[✓] Command On Receive

[ ] Force single file

Customer number:

Figure 9.1. Turning on command on receive for a given partner

Using this check box you can always determine whether to use commands on receive for a subset of all partners. Checking “Command On Receive” for a partner will have two effects:

- The files for the transactions associated with this partner will then have the file extension `.cor` instead of `.res`. In mixed operation, this means that result files for transactions that are still waiting for the execution of a command on receive can be distinguished from those for which the execution of commands on receive is not configured.
- Shortly after receiving the complete data, the program configured in `EAS2.properties` will be started.

9.3 Procedure of handling command on receive

The e-AS2 server processes the files waiting for a command to be invoked with a separate thread. This monitors the incoming directory in the interface area and looks for files with names ending in `.cor`. If this type of file is detected, the result file will be imported and the command line for invoking the command will be built.

When using the directory interface for incoming data, all files are picked up in the respective partner directory. For each file the command is invoked with a shortened command line as described below.

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

9.3.1 Command line for RES interface

When using the RES interface the command is invoked with six 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.
AS2 From	The-AS2 From information from the HTTP header of the data received, i.e. the sender identification for this data.
AS2-To	The-AS2-To information from the HTTP header of the data received, i.e. the recipient identification for this data.
Message ID	The Message ID for the message received.
Parts	The number of Body Parts gathered from the message received. This corresponds with the number of payload files for this process saved to the interface directory.

File name	Name of the file in the interface directory. The name of the RES file is taken from this plus the file extension `.cor`. For multi-file reception the individual files have a sequence number. (see Section 8.3 (*Multifile Mode*)) The file name in front of the dash is taken as the parameter.

These parameters provide all the information you will need to process the data without needing to parse the result file.

9.3.2 Command line for directory interface

When using the directory interface, 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. For multi-file reception the individual files have a sequence number. (see Section 8.3 (*Multifile Mode*)) The file name in front of the dash is taken as the parameter.

9.3.3 Command termination

The e-AS2 server expects the invoked program to delete all the files associated with the transaction (incl. result file, if applicable) 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.

9.3.4 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 fails 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.

9.3.5 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.

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.

9.3.6 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 Enterprise 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.properties

e-AS2 Enterprise 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 Enterprise 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.properties` file automatically take effect after 20 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.properties`. Here you will see the following section:

```
#
# Default log level, increase to "debug" if needed
#

rootLogger.level = info
```

Here you can change the general e-AS2 Enterprise 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.[66] However, a new version of the `EAS2.properties.template` file will be installed, which may contain samples of new configuration options.

10.2.1 Communication parameters, HTTP

When installing e-AS2 Enterprise 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.

[66]With the exception of changed parameters which may have been entered in one of the forms during installation.

The AS2 protocol requires the data sender to transmit the response address for asynchronous MDN to the recipient as header information. e-AS2 Enterprise 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 Section 10.2.4 (*URL for asynchronous MDN*) for details.)

10.2.2 Communication parameters, HTTP/S

The configuration file contains the following section for HTTP/S communication:

```
# Additional settings for HTTP/S
#connection.https.port = <some port>
#connection.https.cert = <certificate alias>
#connection.https.clientauth = 1
```

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

`connection.https.cert` specifies the alias of the SSL server certificate in the certificate management. Attention! A certificate with this alias must actually exist or the HTTP/S thread will not be started.

`connection.https.clientauth` can be used to specify for incoming SSL connections to only be accepted if the client establishing the connection identifies itself by certificate. Set this property to 1 for this purpose. If you do not wish to use client authentication, set this property to 0

The e-AS2 server will start the thread for HTTP/S communication if both the port number and the certificate alias are configured for HTTP/S.

If you require additional HTTP/S threads, you can configure them (up to four in total) as follows:

```
# Additional Settings for another HTTP/S thread (optional)
#connection.https.port.2 = <some port>
#connection.https.cert.2 = <certificate alias>
#connection.https.clientauth.2 = 0

# Additional Settings for another HTTP/S thread (optional)
#connection.https.port.3 = <some port>
#connection.https.cert.3 = <certificate alias>
#connection.https.clientauth.3 = 0

# Additional Settings for another HTTP/S thread (optional)
#connection.https.port.4 = <some port>
#connection.https.cert.4 = <certificate alias>
#connection.https.clientauth.4 = 0
```

10.2.3 Communication parameters, dedicated IP addresses

Normally when the e-AS2 server starts, it binds to all network interfaces. If there are several network interfaces with different IP addresses in the machine, then e-AS2 is reachable through all interfaces. Incoming connections will be accepted regardless of the interface they used.

In some cases you may wish to restrict access and allow incoming connections only on dedicated interfaces or IP addresses. In these cases you can use the following settings:

```
connection.http.bindip = <IP-Adresse>
connection.https.bindip = <IP-Adresse>
connection.https.bindip.2 = <IP-Adresse>
connection.https.bindip.3 = <IP-Adresse>
connection.https.bindip.4 = <IP-Adresse>
```

With this configuration active the e-AS2 server will bind to exactly the one network interface that's identified by it's IP address.

Use these settings only, if you know what you are doing. If in doubt, do not use the settings. Then the default behaviour applies.

10.2.4 URL for asynchronous MDN

By default, e-AS2 itself composes the URL for asynchronous MDN from host name, port and URI for the respective protocol. The URL composed in this way is sent to the remote system with the MDN request. The remote station is expected to use this URL for sending back the asynchronous MDN.

In some cases, e.g. in cluster configurations, this way of specifying the URL for asynchronous MDN is not useful. In such cases, it is possible to explicitly define the URLs for asynchronous MDNs.

```
async.mdn.url.http = http://external-name:8080/path/as2in
async.mdn.url.https = https://external-name:8081/path/as2in
```

10.2.5 SSL settings

When communicating via HTTP/S, first the connection between the client and server will be established. This is followed by the SSL handshake phase. During this phase the two communication partners involved negotiate

- which protocol implementation and version (protocol) to use, and
- which encryption algorithm (cipher suite) will be used.

Here, e-AS2 accesses the SSL routines of the Java Runtime. It contains a rich set of implementations for protocols and cipher suites. After the SSL handshake the entirety of implementations can be categorised by three groups:

- supported implementations,
- enabled implementations (enabled), and lastly
- implementations used for the specific session.

The implementations available for use form a true subset of implementations supported. During the SSL handshake the two communication partners involved negotiate a protocol implementation and an encryption algorithm for the session from the respective available number of supported implementations. If the intersection of supported implementations for protocols or for encryption algorithms between the two ends is empty, it is considered an error and the HTTP/S connection will not be established.

The number of supported and available implementations may vary for different Java Runtimes. Nevertheless, it’s normally best not to change the default settings.

If you still feel the need to intervene in this area, you may choose from four settings in the `EAS2.properties`.

```
https.server.protocols = SSLv3, TLSv1
https.server.ciphers = TLS_RSA_WITH_AES_256_CBC_SHA, TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
https.client.protocols = SSLv3
https.client.ciphers = TLS_DHE_RSA_WITH_AES_256_CBC_SHA,
 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
```

These settings have the following function:

https.server.protocols	Used to specify which protocols you want e-AS2 to offers for incoming HTTP/S connections.
https.server.ciphers	Used to specify which encryptions algorithms you want e-AS2 to offer for incoming HTTP/S connections.
https.client.protocols	Used to specify which protocols you want e-AS2 to offer for outgoing HTTP/S connections.
https.client.ciphers	Used to specify which encryption algorithms you want e-AS2 to offer for outgoing HTTP/S connections.

The above values are examples. Generally select protocols and encryption algorithms from the number of supported implementations. Configure both as a list of comma separated string constants[67] on a single line. The new settings will take effect after restarting the e-AS2 server.

For an overview of the current configuration and the implementations supported by the system, run the program

```
sslinfo.sh
```

on the command line. This will output a list of supported implementations available for use for protocols and encryption algorithms, for incoming as well as for outgoing connections. The current configuration in `EAS2.properties` is taken into account.

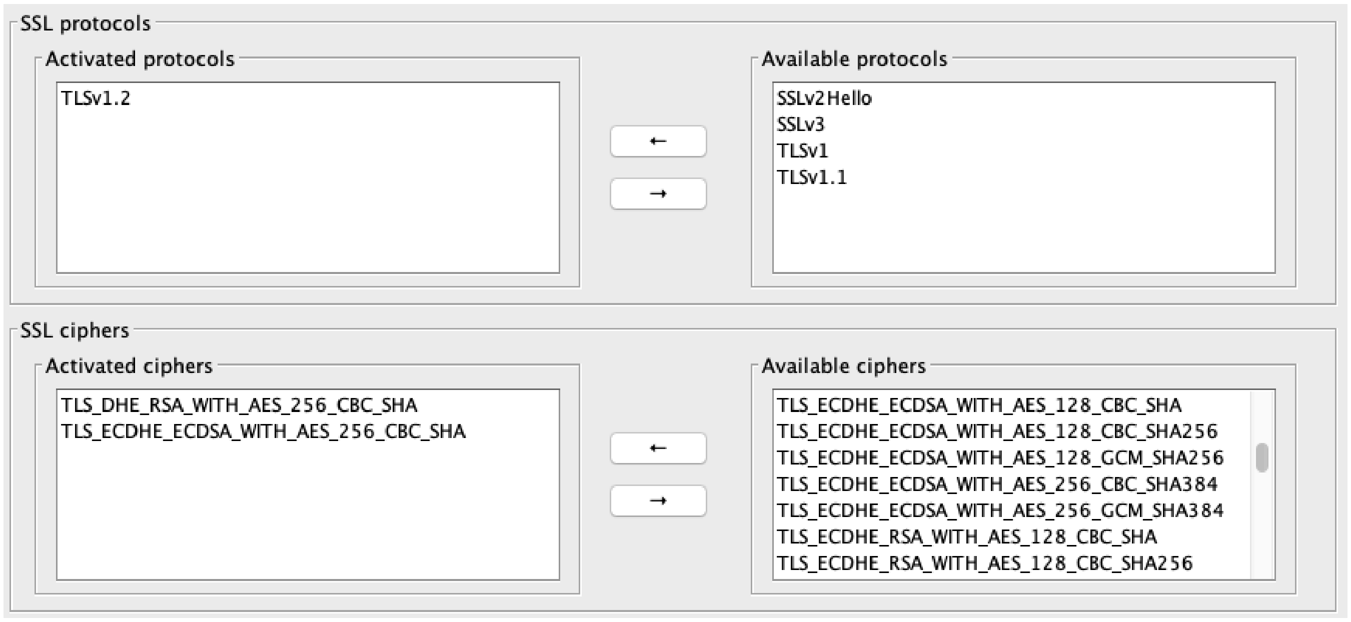
The four properties are solely configured using the string constants output by `sslinfo.sh`. If you make a mistake during configuration, you will not be able to start the e-AS2 server. The cause for the error will be specified in the log. If the configuration is OK, the server will be started and the log record exactly which SSL settings took effect.

Note! Try not to be too restrictive in the configuration. The less choices are available during the SSL handshake, the more likely, HTTP/S connections will not be established.

You can limit and expand the set of enabled implementations from the default state of Java Runtime available for use. However, you can only select from the set of supported implementations. Protocols or encryption algorithms not supported by the Java Runtime implemented cannot be used by e-AS2.

If necessary, you can also configure the SSL settings for outgoing connections for individual communication partners. To do so, switch to the tab “SSL/TLS” and from the set of available protocols and algorithms select the subset to be used for the connections with the respective partner.

[67]Whitespace before and after the comma is allowed.


SSL protocols

Activated protocols

TLSv1.2

Available protocols

SSLv2Hello
SSLv3
TLSv1
TLSv1.1

SSL ciphers

Activated ciphers

TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA

Available ciphers

TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256

Figure 10.1. SSL settings in partner profile

You can configure the SSL settings in the partner profile regardless of the settings in `EAS2.properties`. If the partner profile and `EAS2.properties` both contain settings, the settings from the partner profile take precedence.

Please note, you can only configure the SSL settings for outgoing connections in this way. Partner-specific SSL settings are generally not possible for incoming connections, as the partner profile of the respective party is not yet known at the time the connection is established.[68]

10.2.6 Communication parameters, HTTP upload

The following section in the configuration file is important for configuring the HTTP upload:

```
# Additional settings for pure HTTP file upload (w/o AS2)
#http.upload.allow = 1
#http.upload.sslonly = 1
#http.upload.uribase = /upload
```

`http.upload.allow` enables the HTTP upload on the server. I.e. the e-AS2 server will then accept HTTP uploads via the defined URI.

With `http.upload.sslonly` set to 1, connections using the defined URI for HTTP upload will only be accepted via the Port for SSL connections (HTTP/S).

The beginning of the URI for HTTP uploads is defined with `http.upload.uribase`. For more information on URI evaluation for HTTP uploads, please refer to Section 5.20 (*Using HTTP upload*).

10.2.7 Parameters for proxy use

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

[68]AS2From and AS2To are required to locate the partner profile. However, the HTTP header containing this information can only be transmitted once the complete HTTP/S connection has been established, so only after the SSL handshake.

```
# 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.[69]

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 4.2 (*Using proxy servers*) again at this point and consult with your firewall administrator.

10.2.8 Communication parameters, configuration

During the e-AS2 Enterprise 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 = passwort
```

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 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!

10.2.9 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
```

[69]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.

```
    #smtp.connect.timeout = 2
    #pop3.connect.timeout = 2
```

This determines how long e-AS2 Enterprise 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.

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

10.2.10 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.

10.2.11 Auto-Disable

Partner profiles in e-AS2 can be disabled automatically after a defined number of connection errors has been exceeded.

```
    #partner.disable.threshold = 10
```

In order to activate auto-disable remove the hash mark (#) at the beginning of the line. If more then 10 connections in a row fail, the respective partner profile will automatically be disabled. Only the send direction is affected by auto-disable. The receive direction remains active, if it has not been manually de-activated by the user.

You can manually re-enable a disabled partner profile using the GUI client. Please read Section 11.1.4 (*Profile active, General*) for more information.

10.2.12 Cryptography settings for AS2

This version of e-AS2 uses the following algorithms by default:

AES256-CBC	Messages are encrypted using AES (Advanced Encryption Standard) with 256 Bit key size in CBC mode (Cipher Block Chaining).

SHA256withRSA	Messages are signed using SHA 2 (Secure Hash Algorithm) with 256 Bit digest length and RSA encryption (Rivest-Shamir-Adleman)[70].

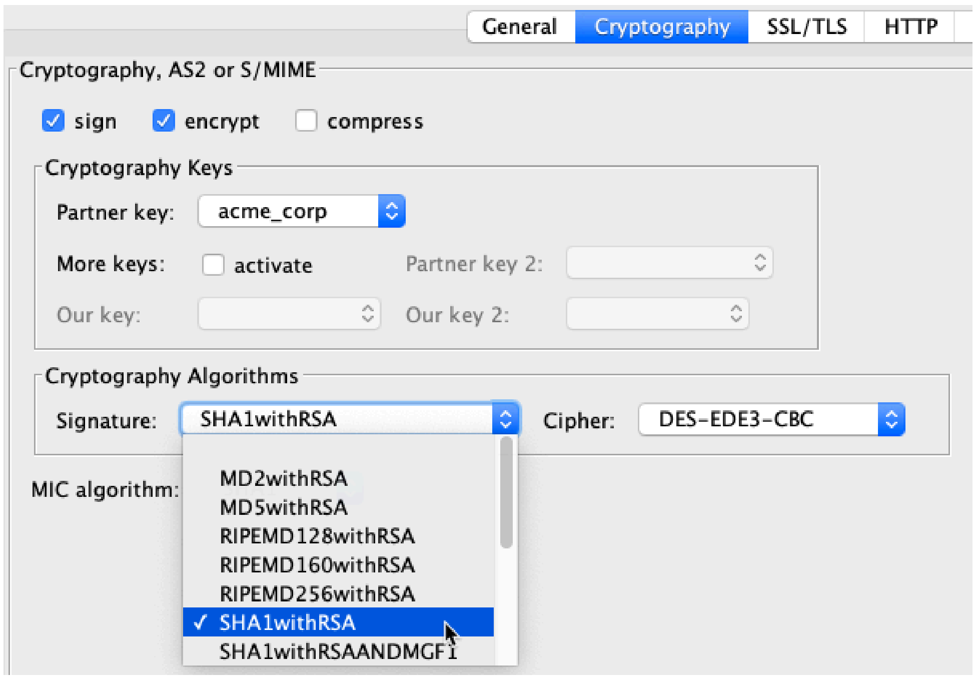
According to the state of technology these algorithms are considered secure at the time of release of the software version documented herein, so not vulnerable to attacks.

In some circumstances deviating from these algorithms in general or at individual partner level may be desirable. Deviations are conceivable in both directions. It may be – for matters of principle or at the request of a communication partner – desirable to use even stronger algorithms. Or a communication partner may be using outdated AS2 software which does not yet support modern cryptography algorithms. One would then want to scale back to weaker algorithms, at least for this partner.

The default cryptography algorithms can be set in `EAS2.properties` as follows:

```
# AS2 cryptography defaults
as2.encryption.cipher = AES256-CBC
as2.signature.algorithm = SHA256withRSA
```

In addition to the global setting there is the option to specify the cryptography algorithms specifically for individual partners. In this case the setting is configured in the partner profile, under the “cryptography” tab.



✓ sign ✓ encrypt ☐ compress

Partner key: acme_corp

More keys: ☐ activate

Partner key 2:

Our key:

Our key 2:

Signature: SHA1withRSA

Cipher: DES-EDE3-CBC

MIC algorithm:

Figure 10.2. Setting cryptographic algorithms for AS2

If the respective fields in the GUI are left blank (no entry selected), the default settings will be used. So you don’t have to explicitly configure the algorithms for each partner, only those where you want to deviate from the defaults.

Note: We recommend keeping the standard settings and only deviating from them when absolutely necessary. Analysing the various available algorithms in respect to quality and security

[70]The key size for RSA is specified when creating the certificate. We recommend using a key size of at least 2048 Bit.

is beyond the scope of this manual. The order of the algorithms in the selection fields in the GUI has no meaning. It is merely an alphabetic listing.

10.2.13 Timeout settings for asynchronous MDN

As you learned in Section 5.12.3 (*Asynchronous MDN*), you can set a specific timeout value for awaiting an asynchronous MDN for each partner. Alternatively, or in addition, in `EAS2.properties` you can define a standard timeout which will then take effect for all partners which do not have a different value explicitly configured.

```
# wait timeout for async MDN (in minutes, default: infinite)
#async.mdn.wait.timeout = 30
```

There is no default here. If this line is not active, the system will wait for an asynchronous MDN infinitely.

The following setting is related but used in a different context:

```
# Timeout for async MDN for GUI (in seconds, default: 15)
#gui.mdn.async.wait = 15
```

This specifies how long the GUI should wait for an asynchronous MDN to be received. This value should be rather low to prevent blocking the program for too long. In this context, not receiving the MDN during the configured period is not interpreted as an error. The system will simply stop waiting without result. The MDN status must be determined later by reviewing the respective transaction in the GUI.

10.2.14 Message IDs for asynchronous MDN

Asynchronous MDN are special messages with the job to transport status information related to previously received messages. To allow the other end to assign the status information, the content of the MDN cites the Original Message ID of the previously received message.

Aside from this, asynchronous MDN are normal messages which in turn deserve their own message ID. In practice, however, opinions differ. There are AS2 implementations which cannot or incorrectly process asynchronous MDN with separate message IDs.

If you work with communication partners encountering problems processing asynchronous MDN sent by e-AS2 Enterprise, you can globally disable generating message IDs for asynchronous MDNs.

```
# Create message IDs for asynchronous MDN
#async.mdn.create.message.id = 1
```

To do so, copy this setting to your `EAS2.properties` and set the value to 0.

10.2.15 Database Logging

Log messages generated whilst e-AS2 Enterprise is running jobs will be written to log files according to the configuration. When using MySQL, all messages will additionally be written to the database, where they are stored for some time. The database logging allows for log messages to be displayed in the GUI client by transaction so the user can usually forego looking at the log file during research. Database logging is controlled by two properties.

```
logging.db = 1
logging.db.age.max = 2
```

The first line enables database logging in general. A value of 0 disables database logging. You can therefore change the standard behaviour in both directions. MySQL users can therefore set the value 0 to disable database logging if it is not needed. HyperSQL users can enable database logging by setting the value 1.[71]

The second line determines the period for which log messages are saved in the database (in days). Here we recommend using the lowest value possible, since research is usually done for recent cases, and older log messages are therefore no longer needed.[72]

10.2.16 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 JVM[73]. 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.

10.2.17 Transfer thread settings, incoming

In its basic state, e-AS2 will start up to 100 parallel threads for incoming data per AS2 and per SMTP. As a result, new connection requests from outside will also be accepted if other transfers are already running. Depending on the system, the high degree of parallelism can negatively impact the entire system. You may therefore modify the number of parallel receive sessions.

```
# Maximum number of simultaneous incoming sessions for AS2
limit.incoming.as2 = 100
# Maximum number of simultaneous incoming sessions for SMTP
limit.incoming.smtp = 100
```

To reduce the maximum number of parallel incoming sessions, set a lower value. If your system is well equipped, you may of course also set a higher value here to increase the degree of parallelism permitted.

[71]Database logging significantly increases the load and increases data traffic in the database. That's why by default it is only enabled for dedicated standalone database servers. Whether a HyperSQL-based installation will increase the data traffic due to log messages depends on the general system performance and the number of transactions processed per time period.
[72]To access historic log messages, use the log files and save them for an adequate period.
[73]Java Virtual Machine

Connection requests received whilst at the maximum number of parallel threads are rejected before data is transmitted. This will implicitly result in these settings having a global effect, so regardless of the communication partner. Partner-specific settings first require data to be transmitted which can then be used to identify the partner.

10.2.18 Transfer thread settings, send

Parameters for e-AS2 thread management can also be changed with configurations for send processes. There are three properties with the following default settings.

```
# Maximum number of simultaneous send sessions
# (incl. pending transfer permission)
limit.outgoing.running = 50
# Maximum number of simultaneously active transfers for send sessions
limit.outgoing.transferring = 50
# Upper limit for maximum number of simultaneously active transfers for
# individual partners (overwrites higher setting in the partner profile)
limit.outgoing.perpartner = 10
```

These settings provide very vast options for impacting the way e-AS2 behaves when sending data. If you’re unsure, you should not change the default settings.

Maximum number of simultaneous send sessions

By sessions we mean here the complete processing of a send job.. This already starts with analysing the INF file and searching for the related partner profile. Only then are (from the partner profile) the parameters for establishing the connection and cryptography known, so that the connection can then be established and data transfer initiated.

Once the maximum number of simultaneous sessions has been reached, new send jobs are initially not processed. In this case processing transmission for a respective directory is in hold mode, which will stop until resources for processing are available again. Processing will then be initiated and a new send thread created.

Maximum number of simultaneously active transfers

In addition to the maximum number of simultaneous sessions, the number of simultaneously active transfers can be set. Active transfer starts after determining the parameters from the partner profile by coding the data and establishing an HTTP connection to the other end.

Set a lower value than the maximum number of simultaneous sessions to reduce the utilisation of communication resources.[74] Once the maximum value has been reached, all send sessions running are switched to hold until resources for executing transfers are available again.

Maximum number of simultaneous active transfers per partner

Within the maximum number of simultaneous active transfers allowed you can further limit the number of active transfers per partner. This ensures individual partners with a high transfer volume don’t use the entire reserve of active transfers, thus blocking transmission to other partners.

As a last configuration step you can explicitly set the maximum number of simultaneous active transfers for each individual partner in the respective partner profile. This is done by entering the desired number in the “STL” field. By default, this value is initially set to 1 for all new partner

[74]It doesn't make sense to set a value higher than the maximum number of simultaneous sessions, since this cannot be exceeded. The number of threads with active transfers is a subset of the number of threads with send sessions running.

profiles so there are no parallel transfers to the same partner. Increase this value for partners where you wish to establish parallel transfer sessions.

Summary

From the perspective of an individual send job the behaviour can be summarised as follows.

First a permission to process is obtained for the send job. This is granted as long as the maximum number of simultaneous send sessions has not been reached. Once granted permission to process, a new send thread is generated and processing starts. For new jobs, the INF file or the interface directory path is analysed. For retries following a failed attempt the existing transaction is analysed. This is used to determine the associated partner profiled, from which the transfer parameters are then read.

Then the transfer permission is obtained. This is granted to the job as long as the maximum number of simultaneous transfers for this partner has not been reached. The check includes the global settings as well as the STL information in the partner profile. The lowest value is applied. Once the permission to transfer has been granted, the connection is established and transmission to the other end starts.

Sending MDN defines generally separate processes which are included in the count, along with all new and retry jobs of payload to be transmitted.

10.2.19 Transfer thread settings, operator e-mails and statistic logging

For you as the e-AS2 administrator it’s important to get a feel for which settings should be changed and which can be left at the standard values. To support you in this process, e-AS2 sends operator e-mails when defined thread limits are reached. (However, this only applies if the respective maximum value reached is greater than 1.) You can adjust the frequency for repeating similar operator e-mails in this context.

```
# Mail repeat when thread limits reached (in hours)
limit.reached.mail.repeat = 4
```

With this default setting, once an operator e-mail has been sent, no additional e-mails of the same type will be sent for four hours, even if the event triggering this e-mail occurs again.

In addition to operator e-mails there is statistics logging as an additional source of information, which is written to the log file at regular intervals and when the server is ended.

```
02:00:00,033  INFO [   STATLOG] [-] - === SERVER STATISTICS START ===
02:00:00,033  INFO [   STATLOG] [-] - server uptime: 21 days 23:21:12 hours
02:00:00,033  INFO [   STATLOG] [-] - max parallel send thread statistics:
02:00:00,033  INFO [   STATLOG] [-] - <<global>> - 0
02:00:00,033  INFO [   STATLOG] [-] - p1 - 0
02:00:00,033  INFO [   STATLOG] [-] - p2 - 1
02:00:00,033  INFO [   STATLOG] [-] - p3 - 0
...
02:00:00,033  INFO [   STATLOG] [-] - max parallel AS2 receive threads: 4
02:00:00,033  INFO [   STATLOG] [-] - max parallel SMTP receive threads: 0
02:00:00,033  INFO [   STATLOG] [-] - maximum number of concurrent database connections: 0
02:00:00,033  INFO [   STATLOG] [-] - currently available free memory: 1584 MB
02:00:00,033  INFO [   STATLOG] [-] - currently assigned total memory: 1624 MB
02:00:00,033  INFO [   STATLOG] [-] - currently used memory: 40 MB
02:00:00,033  INFO [   STATLOG] [-] - maximum available memory: 2731 MB
02:00:00,033  INFO [   STATLOG] [-] - total unused memory: 2691 MB
02:00:00,033  INFO [   STATLOG] [-] - === SERVER STATISTICS END ===
```

For more information on statistics logging see Section 12.4 (*Memory management, statistics logging*).

10.2.20 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.[75]
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.

In addition, e-AS2 Enterprise with activated SMTP feature can also send EDI messages via SMTP email instead of AS2. Both cases require access to an email server to be configured.

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

#mail.ssl = 1
# SSL or STARTTLS
#mail.ssl.method = STARTTLS
```

`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 an may be left blank for email servers without authentication.

If you wish to establish a secure connection to the e-mail MTA, you can enable SSL by setting `mail.ssl` to 1. Please note, in this case the server certificate for the SMTP-MTA must be imported into e-AS2 certificate management in order for the connection to be established.

By default the SSL connection to the email server is established using the `STARTTLS` method. The method can be changed by changing `mail.ssl.method` to `SSL`.

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

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

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

# additional mail receipient (for dry-run mails, falls back to mail.to,
#                              if not set)
```

[75]This is the default behavior. A variant of it is available. See Section 10.2.21 (*Incident-Handling*).

```
#mail.to.dry-run = AS2 Dry-Run <as2dryrun@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.

This email address serves as the default address for operator email delivery. You can enter specific email addresses for individual partners or certificates in the partner profiles or in certificate management (see Section 11.1.36 (*If problems occur when sending data, use this address, Mail*) ff. and Section 11.4.10 (*Edit certificate*)).

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

`mail.to.dry-run` is used to configure the email address of the desired recipient of dry-run mails. Only one recipient can be specified. If this is not set, then `mail.to` is used instead.

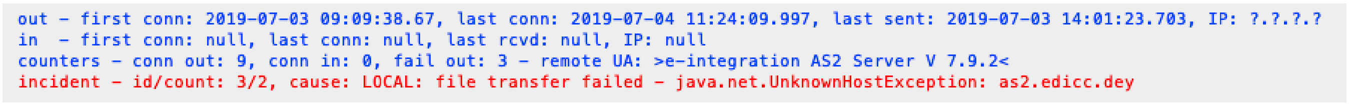
`mail.from` is used to specify the sender's email address for the operator email to be used in place of the standard value generated by the email system.

In addition, for asynchronous MDN via SMTP, `mail.from` is used, if no email address was entered in the partner profile.

10.2.21 Incident-Handling

Since version 7.10 e-AS2 has a so called incident handling built in. This means that errors in the processing of message transmissions are not viewed in isolation, but are related to the partner concerned. If an error occurs in a send transaction, an incident is created for the partner concerned, which is assigned a unique incident number. Further errors in subsequent transactions do not change the incident status. As soon as message transfers to the partner are successful again, the active incident status is reset for the partner.

The current incident status for the partner can be seen in the GUI partner dialog below the detail display.



```
out - first conn: 2019-07-03 09:09:38.67, last conn: 2019-07-04 11:24:09.997, last sent: 2019-07-03 14:01:23.703, IP: ?.?.?.?
in  - first conn: null, last conn: null, last rcvd: null, IP: null
counters - conn out: 9, conn in: 0, fail out: 3 - remote UA: >e-integration AS2 Server V 7.9.2<
incident - id/count: 3/2, cause: LOCAL: file transfer failed - java.net.UnknownHostException: as2.edicc.dey
```

Figure 10.3. Partner status, with incident details

A red line indicates an active incident and describes the details. If there is no incident for the partner, only the three blue lines with statistical data are available.

Incident handling offers an alternative approach for operator mails. In addition to the operator mails described above for each failed transaction, e-AS2 can send operator mails for incidents. The following properties can be used to switch the two categories of operator mails on or off separately.

```
mail.operator.transactionCentric = 1
mail.operator.partnerCentric = 0
```

To get only operator mails for incidents instead of operator mails per failed transaction, set the first property to 0 and the second to 1. However, you can also switch both on or both off. The latter is explicitly not recommended.

Partner-centric, i.e. incident-related operator mails always occur in pairs. A first mail is sent as soon as an incident occurs. A second mail follows as soon as the incident has ended, i.e. the connection to the affected partner functions normally again. The incident ID can be used to relate the two mails at the recipient's end.

Incident handling is switched off by default. It can be activated with the following setting in `EAS2.properties`:

```
incident.handling = 1
```

With incident handling switched off the above mentioned settings regarding operator mails are without effect.

10.2.22 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 erronous.
#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.

10.2.23 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
```

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. For more information on IO-Traces, please read Section 12.2 (*IO-Traces*).

10.2.24 Checking certificate validity

Every day the e-AS2 server checks the validity of all certificates under management.[76] 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.

10.2.25 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 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`.

10.2.26 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 Enterprise installation folder. Under `EAS2.properties` you can configure a primary and secondary interface area. The secondary interface area is detailed further in Section 10.2.27 (*Secondary interface area*).

```
# Root of primary interface directory tree
```

[76]This explicitly also includes SSL certificates.

```
#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
```

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 Enterprise is installed must share a local partition, which is then accessed by another computer in the network.

10.2.27 Secondary interface area

In e-AS2 Enterprise you have the option to activate a secondary interface area. Define the following property for this purpose:

```
# Root of secondary interface directory tree (only for Enterprise version)
interface.root.secondary = interface2
```

After restarting the e-AS2 server the secondary interface area is now active. It can be used for both sending and receiving data. So, the subdirectory structure is identical to the primary interface area.

With the secondary interface area activated an additional selection field will appear in the partner profiles, as shown in Figure 10.4, "Selecting the secondary interface area".

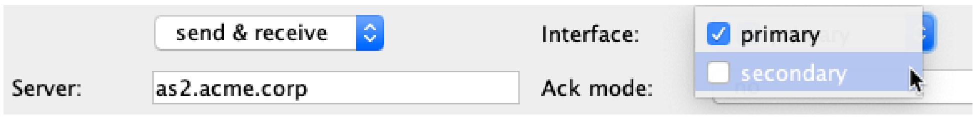


Figure 10.4. Selecting the secondary interface area

Selecting the secondary interface area will result in a respective change to the file storage for incoming messages and MDNs. For delivery reports the logic is different. They are stored in the same interface area that was used for sending the message that is being acknowledged by the report. So the GUI setting is irrelevant for DRs.

Use of the secondary interface area is optional. It allows for incoming files for select partners to be redirected to other channels. So e.g. messages for partners currently being set up and tested can be separated from those which are already received in the production system. Also this mechanism can be used to duplicate incoming messages. This is done by activating both the primary and the secondary interface.

10.2.28 Display names for interface areas

To assist GUI users in interface selection, you can configure specific display names for the interface areas on the server side.

```
interface.root.primary.name = PROD channel
interface.root.secondary.name = QA channel
```

The e-AS2 server transfers this configuration to the GUI client. The configured display names are then offered for selection in the partner profile configuration in the GUI.

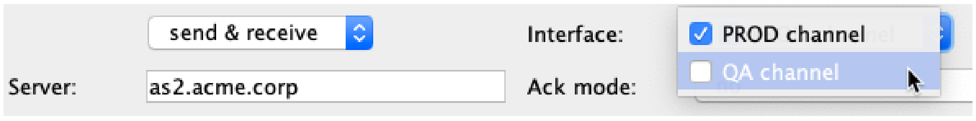


Figure 10.5. Interface selection with display names

10.2.29 RES interface vs. directory interface for incoming messages

Out of the box, e-AS2 Enterprise has the RES interface (see Section 5.4.4 (*The result file*)) activated for receive processes. You can decide foregoing the benefits of this interface and to instead use the e-AS2 Connect directory interface.

```
# controlling the input interface / directory based (0) or RES file based (1)
# (only for Enterprise version - Connect version is always directory based),
# encoding
interface.in.res-file = 1
#res-file.encoding = ISO-8859-1
```

Set the `interface.in.res file` value to 0 to disable the RES interface and activate the directory interface.

You will find information on the directory interface for received files in the e-AS2 Connect user manual.

By default, RES files are written in “ISO-8859-1” character set. Encoding is relevant when receiving subject lines or file names with special characters. If your processing system requires different encoding, set the respective value in `res-file.encoding`.

10.2.30 Settings for Creating Extended Document Identifiers

After the receipt of new messages e-AS2 always generates a globally unique, so-called extended document identifier for the unique referencing of the process also over further subsequent processing steps. This document ID is structured as follows:

```
D<docid>@<sysid>
```

The `<docid>` is a sixteen-digit number (consecutive document number) and `sysid` is a textual identifier of the e-AS2 instance (system ID).[77] At the beginning the count starts with 1, with the first two digits constantly set to 40 as prefix. The system ID is set to “EAS2”.

You can change both the prefix and the system ID according to your own requirements.

```
docid.prefix = 40
docid.sysid = EAS2
```

10.2.31 Building file names for incoming messages

This setting is only pertinent if the RES interface is disabled as described above. Doing so activated the directory interface. Result files will not be written. All relevant information about the data received must now be coded 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].

[77] By convention, the system ID is usually formed from capital letters and digits.

10.2.32 INF interface vs. directory interface for send jobs

Out of the box, the directory interface is activated for send jobs in e-AS2 Enterprise, the INF interface is disabled. Use of these interfaces is not a matter of either-or. Both can be activated or disabled independently.

```
# controlling the output interface / INF or directory, encoding
interface.out.directory = 1
interface.out.inf-file = 0
#inf-file.encoding = ISO-8859-1
```

The value 1 activates the respective interface. The value 0 disables it.

Setting both values to 0 will configure a system which can only receive data, since neither of the two possible outgoing interfaces will be active.

By default, INF files are expected in “ISO-8859-1” character set. This setting is only relevant when generated INF files where contents may contain special characters. Set `inf-file.encoding` to a different value to change encoding to your needs. The e-AS2 server will respect your encoding setting when reading the INF files.

10.2.33 Saving received MDNs to the file system

By default, e-AS2 Enterprise saves received MDNs to the interface area, under the `in_mdn` directory. Here, the contents of the MDN are saved to one file and meta information about the MDN in another file, the result file.

If you don’t wish to interpret received MDNs at this level, you can disable saving MDNs to the file system.

```
# write received MDN to filesystem
# default: 1)
#interface.mdn.write-res = 1
```

Set this value to 0 to disable saving MDNs.

10.2.34 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.

10.2.35 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.

10.2.36 Cleanup routines

e-AS2 Enterprise 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 Enterprise 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 10.2.26 (*Interface area*)!

10.2.37 Checking for old files

On request, e-AS2 Enterprise 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.

Another reason for old files may be the attempt to send more than one asynchronous MDN for a transaction. Once an MDN has been successfully sent, e-AS2 will not process additional MDNs for the specific transaction. The redundant MDN trigger file will then be left behind and does become an “old file” in `out_mdn` after a while.

Another possible cause for old files could be the accidental use of “dead interface directories”.

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 10.2.20 (*Email configuration*)).

10.2.38 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.

10.2.39 Names for server entities

In environments where several AS2 servers need to be managed[78], it may be helpful to assign entity names to the different servers. This server name can be configured as follows:

```
# Instance name for this e-AS2 server
#eas2.instance.name = Sales Department
```

The server name is specified in `EAS2.properties` for the server installation and transmitted to the configuration client from there. In the GUI it is then displayed in the window title of the main screen.

10.2.40 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.

10.2.41 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.

10.2.42 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
```

10.2.43 Managing GUI clients in distributed environments

In environments where several employees in different locations access an e-AS2 server via GUI, some properties can be used for managing GUI access via the server in an orderly fashion.

[78]E.g. one server for test and one server for production mode, or different servers for different departments in large companies.

Each installed entity of the GUI client should be associated with a name. The following should be set in `EAS2.properties` for the GUI installation for this purpose:

```
# Name of this eAS2 GUI client
#eas2.guiclient.name = Tim Tester
```

The configured name will appear in the title bar of the window along with being transmitted to the e-AS2 server so it can provide an overview of the GUI clients currently signed in. (see Section 11.8.4 (*Connected GUI clients*)).

All configured GUI client names must be different. We recommend entering the actual name of the employee using the respective client.

In `EAS2.properties` a server administrator can limit the rights of specific GUI clients as follows.

```
# Configuring client admin access
#eas2.client.admin.1 = Adam Administrator
#eas2.client.admin.2 = Oscar Operator
# ...
# more as needed, number ascending without gaps
```

This explicitly lists the names of all GUI clients to grant admin rights. The numbering at the end of the property description must begin with 1 and be ascending, without gaps. Any GUI clients not listed (incl. clients signing in without name) will not be granted admin rights.

The admin rights enable server management access in the “Manager” tab of the GUI.

10.2.44 GUI policies

You have the option to determine specific restrictions or presets for the connected GUI clients with server end configurations.

Mandatory operator e-mail addresses

Set a property as follows to make entering operator e-mail addresses mandatory.

```
gui.policy.operator.mail.mandatory = 1
```

Users of GUI clients connected to this server will then be required to enter operator mail addresses in each and every case.

The respective fields in the partner profile and in the certificate properties must not be left blank. Otherwise the profile cannot be saved.

Note. This setting only affects future configurations and forces operator mail addresses to be entered. This does not force existing partner profiles to be changed and completed.

Define initial status of check box “command on receive”

Without this setting the initial state of the check box “command on receive” is unchecked (switched off) for new partner profiles. To use the command on receive feature for this partner, the user must explicitly turn it on. Use

```
gui.policy.command.on.receive.checked = 1
```

to reverse the behaviour. “Command on receive” will then initially be checked (switched on) in new partner profiles. The user must explicitly turn the feature off if it is not desired.

Lock status of check box “command on receive”

To protect the status of the “command on receive” check box in the partner profile from being changed, set

```
gui.policy.command.on.receive.locked = 1
```

The setting for “command on receive” can then no longer be changed from the GUI.

Mandatory customer number

Set a property as follows to make entering customer numbers mandatory and secured against changes.

```
gui.policy.customer.number.mandatory = 1
```

Users of GUI clients connected to this server will then be required to enter customer numbers in each and every case.

The respective field in the partner profile must not be blank. Otherwise the profile cannot be saved. However, an option to override is presented in the warning dialog that is shown, when the field is left blank.

Also, for existing partner profiles, if the user tries to change the customer number, they will be presented with a warning, too. Once again, overriding that warning and saving anyway is possible. If the user does so, an operator mail will be sent, to inform e-AS2 operators of a potentially critical change of configuration.

Note. This setting only affects future configurations and forces customer numbers to be entered. This does not force existing partner profiles to be changed and completed.

10.2.45 Database connection

As long as you are running e-AS2 Enterprise with the integrated HyperSQL database, the database connection is not an issue. If you intend to use MySQL as the database server, the database connection must be configured in `EAS2.properties`. When creating the database using the designated setup script, all the required properties will automatically be generated and can be inserted via copy and paste.

```
# database connection
#database.jdbc.driver = com.mysql.jdbc.Driver
#database.url = jdbc:mysql://localhost:3306/eas2db
#database.user = eas2user
#database.password = eas2pw
```

The first property value `database.jdbc.driver` must be fixed to `com.mysql.jdbc.Driver` to use MySQL as the database system instead of HyperSQL. The other three properties can be changed according to your needs. As you can see in the sample lines, the database server does not necessarily have to be running on the same computer as e-AS2 Enterprise. If necessary, replace `localhost` with the host name of the database server.

Important! At this point we would like to point out MySQL 5 and 8 are currently the only external database systems supported. Do not attempt to run e-AS2 Enterprise with other database systems.

Two other properties can be helpful to investigate problems with unstable database connections.

```
# Pool-Debugging
```

```
database.connection.pool.debug = 0
```

Set this value to 1 to enable extended debugging output with regard to the database connection pool. If this is enabled, e-AS2 writes the current overall status of the connection pool to `stderr` every minute.

```
# DB Managed Connection Idle Timeout (in Sekunden)
database.connection.idle.timeout = 0
```

By default, e-AS2 assumes that database connections remain established even if they are not actively used. However, if the database is disconnected, e-AS2 will notice this and re-establish the connection. In certain situations it may be desirable for e-AS2 to terminate unused connections after a certain period of time. To do this, set the above property to a value greater than 0.

10.2.46 Controllable retry handling

Attention! We strongly recommend leaving the defaults for these settings. Only change something if you're absolutely sure it's actually necessary and you understand the impact of these settings. Normally no changes are required here.

As explained in Section 5.14 (*Setting the retry pattern*), in the event of communication errors, e-AS2 Enterprise will automatically retry if it seems reasonable in this situation and a respective retry pattern is configured. With respect to when a retry is sensible, e-AS2 Enterprise makes certain assumptions which should be suitable in most cases. These assumptions can be summarised as:

1. In the event of an error preventing transmitting the data to the partner system, a retry is always sensible, regardless of the specific cause for the error.
2. If an error occurs after transmitting the data, so particularly per interpreting the MDN received, retrying the data transmission is not sensible since the other end already received the data and doing so could potentially result in a duplicate transmission.

Nevertheless, in some cases you may feel otherwise and can influence in detail how e-AS2 Enterprise behaves in this respect with a series of properties. The following properties may be used for this purpose (the default value is always included):

```
# Settings for retry handling for communication problems
errorhandling.connect.retry.implicit = 1
errorhandling.connect.refused.retry = 1
errorhandling.connect.timeout.retry = 1
errorhandling.connect.host.retry = 1
errorhandling.connect.ssl.retry = 1
errorhandling.connect.other.retry = 1
errorhandling.transfer.broken.retry = 1
errorhandling.reply.http.retry = 0
errorhandling.reply.mdn.retry = 0
```

You can use this to specifically change the retry behaviour as follows.

errorhandling.connect.retry.implicit

If there are several transactions pending retry and an error occurs again with the first one, all other transactions pending retry will be skipped and in the process implicitly considered retried. I.e. the retry counter for these transactions will increase without making a specific attempt to establish a connection.

We strongly recommend not disabling this behaviour. Particularly if there are errors which only occur after a timeout, processing the individual transactions can take a relatively long time. If the connection to a partner cannot be established at any time for the first transaction pending retry, it is very likely it will also fail for all other transactions at this time. The cause for the error is related to the partner, not the transaction. It would therefore be a waste of time and computer resources to still attempt to establish a connection for all transactions.

errorhandling.connect.refused.retry

The connection to the partner cannot be established because an incorrect IP address or an incorrect port are being addressed, or because the AS2 software on the other end is not active. This will result in a retry by default. Set this property to 0 if you do not wish retries.

errorhandling.connect.timeout.retry

The attempt to establish a connection to the partner will be cancelled after the timeout. This typically indicates a problem with the firewall configuration on the sender or receiver end. This will result in a retry by default. Set this property to 0 if you do not wish retries.

errorhandling.connect.host.retry

The connection cannot be established, since the name of the computer does not exist or the name cannot be resolved through the DNS. This will result in a retry by default. Set this property to 0 if you do not wish retries.

errorhandling.connect.ssl.retry

An HTTP/S connection cannot be established due to authentication problems. This will result in a retry by default. Set this property to 0 if you do not wish retries.

errorhandling.connect.other.retry

The connection cannot be established for an unknown reason. This will result in a retry by default. Set this property to 0 if you do not wish retries.

errorhandling.transfer.broken.retry

A connection to the partner system was established. However, the connection drops during the transfer. This will result in a retry by default. Set this property to 0 if you do not wish retries.

errorhandling.reply.http.retry

Data was transmitted, but an error occurred receiving the HTTP acknowledgement or the synchronous MDN. By default this does not result in a retry, even with a retry pattern configured. Set this property to 1 if you do wish a retry in this case.

errorhandling.reply.mdn.retry

Data was transmitted and an MDN was received. The other end reported an error in the MDN. By default this does not result in a retry, even with a retry pattern configured. Set this property to 1 if you do wish a retry in this case.

Note: By switching the `errorhandling.connect` properties to 0 you can have the system forward to potentially configured backup connections more quickly.

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 Partner ID, General

Data type: *alphanumeric*, max. 64 characters.

The partner ID is an internal e-AS2 identifier for the partner profile. This is purely internal and in no way used for processing the AS2 protocol. We recommend using preferably short yet meaningful identifiers for the partner ID.

11.1.2 AS2-From, General

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.

11.1.3 AS2-To, General

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.

11.1.4 Profile active, General

Data type: *list*

Four settings are available to choose from:

- send & receive
- send only
- receive only

- DEACTIVATED

Using this control you can activate or deactivate transaction processing separately for send and receive direction.

If the send direction is deactivated, data will not be transferred anymore for this partner. New send requests are detected, but result in transactions with error status. The error message in the comment field of the transaction indicates the partner profile being inactive as the reason for the error. In the course of reactivating the partner profile at a later time, all erroneous transactions collected to this point can be released for re-transmission at that time.

If the receive direction is deactivated, new incoming receive jobs also result in transactions with error status. The data received are saved to the `orphan` directory. The HTTP status “503 Service Unavailable” is returned to the sender of the data.

Note: Test transfers manually initiated directly in the GUI client still work for inactive partner profiles.

11.1.5 Server, General

Data type: *alphanumeric*, max. 128 characters

This is the IP address of the computer e-AS2 Enterprise is to connect to in order to send data to this partner via AS2. Alternatively, a DNS resolvable name can also be specified.

11.1.6 Port, General

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

This is the port number the AS2 server uses to address the partner system.

11.1.7 URI, General

Data type: *alphanumeric*, max. 256 characters

This is the post request URI used to establish the connection to the partner system.

11.1.8 STL, General

Data type: *numeric*, positive

This is the send thread limit for this partner. This value will normally remain at the default of 1.

If you intend to change this, first read the explanations on transfer thread management in Section 10.2.18 (*Transfer thread settings, send*).

11.1.9 Retries, General

Data type: *alphanumeric*, max. 64 characters

Here, enter the retry pattern for failed transfers. You can find detailed explanations about the e-AS2 Enterprise repeat system in Section 5.14 (*Setting the retry pattern*) and Section 6.2 (*Configuring retry patterns*).

11.1.10 Ack mode, General

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.

Ack mode = “asynchronous” requests an asynchronous MDN in a separate HTTP session. Ack mode = “eMail” requests an asynchronous MDN by mail.

11.1.11 MDN timeout, General

Data type: *numeric*, positive

This field will be enabled when Ack mode “asynchronous” or “eMail” is selected. Here you can enter the maximum time (in minutes) to wait for the MDN to arrive. If no MDN has been received when the waiting time expires, the respective transaction will be marked erroneous.

If you leave this value at 0, a standard timeout configured in `EAS2.properties` will apply. (see Section 10.2.13 (*Timeout settings for asynchronous MDN*))

11.1.12 eMail, General

Data type: *alphanumeric*, max. 256 characters

This field will be enabled when Ack mode “eMail” is selected. Here, enter the mail address to send the asynchronous MDN for this partner relation to. If this field is left blank, the standard mail address from `EAS2.properties` will apply.

11.1.13 automatically send async Ack, General

Data type: *boolean*

Tick this field if want e-AS2 Enterprise to fulfil requests for asynchronous MDN for received messages. If this field is not ticked, e-AS2 will not automatically send asynchronous MDNs to the other party. You will then need to use the interface for asynchronous MDN documented in Chapter 7 (*Interface for asynchronous MDN*) to specify the MDN content yourself and initiate sending MDNs.

11.1.14 Master, General

Data type: *list*

Here, select a master profile to use for the communication parameters Server, Port, URI and Retries. This will not copy the parameters from the master profile into the current profile. They only temporarily override the locally configured values. If the master selection is later removed, any existing old values will be visible again.

11.1.15 Forward to, General

Data type: *list*

Here you can select a backup profile to switch to after exhausting all retries for the current profile. By using backup profiles you can also form chains of more than two profiles, or if desired closed chains (resulting in endless repeats).

11.1.16 Description, General

Data type: *alphanumeric*, max. 512 characters

This is an arbitrary text field for entering descriptive comments related to the partner profile.

11.1.17 sign, Cryptography

Data type: *boolean*

Tick “sign” to sign the messages you are sending. Normally the first default certificate from certificate management is used for the signature. However, if you made a selection under “Our Key” in the partner profile, the selected certificate will be used instead to generate the signature.

11.1.18 encrypt, Cryptography

Data type: *boolean*

Tick “encrypt” to encrypt the messages you are sending. The certificate for which you selected the alias in the Partner Key field will be used.

11.1.19 compress, Cryptography

Data type: *boolean*

Tick “compress” to compress the messages you are sending. Compressing uses the standardised ZLIB algorithm.

11.1.20 Partner key, Cryptography

Data type: *list*

Here one of the public key certificates in the e-AS2 certificate management will be selected using the certificate alias. If the check box “encrypt” is ticked, the certificate selected this way will be used to encrypt messages sent to the partner. In addition, selecting a certificates at this point will result in incoming signed messages being verified against this certificate. If no certificate is selected here, neither encryption (in send direction) nor verification (in receive direction) will be used.

11.1.21 More keys, Cryptography

Data type: *boolean*

If this field is ticked, additional selection fields for certificates will be enabled.

11.1.22 Partner key 2, Cryptography

Data type: *list*

Here one of the public key certificates in the e-AS2 certificate management will be selected using the certificate alias. If a certificate was selected here, it is used to define a second candidate for verifying signatures in incoming messages.

11.1.23 Our key, Cryptography

Data type: *list*

Here one of the private key certificates in the e-AS2 certificate management will be selected using the certificate alias. This is used to define the key for signing messages to this partner. The default key defined as global in the certificate management will then no longer be used.

In addition, this will also determine an additional certificate to try when decrypting incoming messages.

11.1.24 Our key 2, Cryptography

Data type: *list*

Here one of the private key certificates in the e-AS2 certificate management will be selected using the certificate alias. This determines yet another additional certificate to try when decrypting incoming messages.

11.1.25 Signature, Cryptography

Data type: *list*

To use a signature algorithm different from the global default (see Section 10.2.12 (*Cryptography settings for AS2*)) for signing messages sent to this partner, it can be specified by selecting from the list.

11.1.26 Cipher, Cryptography

Data type: *list*

To use an encryption algorithm different from the global default (see Section 10.2.12 (*Cryptography settings for AS2*)) for encrypting data sent to this partner, it can be specified by selecting from the list.

11.1.27 MIC algorithm, Cryptography

Data type: *list*

Select one of the available MIC algorithms from the list. If you are unsure, leave it at the default value.

11.1.28 HTTP/S, SSL/TLS

Data type: *boolean*

Tick this field to use HTTP/S instead of HTTP when sending messages. Please note, you will probably also need to configure a different port and/or a different server than for an HTTP connection.

11.1.29 Our SSL key, SSL/TLS

Data type: *list*

Here select the alias of the certificate to use for HTTP/S client authentication. If the other end does not require client authentication, this field can be left blank.

11.1.30 Activated protocols, SSL/TLS

Data type: *list*

Here select the SSL/TLS protocols you wish to activate for connections to this partner. If no selection is made, the system-wide settings will apply. (Learn more about this topic in Section 10.2.5 (*SSL settings*))

11.1.31 Activated algorithms, SSL/TLS

Data type: *list*

Here select the SSL/TLS algorithms you wish to activate for connections to this partner. If no selection is made, the system-wide settings will apply. (Learn more about this topic in Section 10.2.5 (*SSL settings*))

11.1.32 HTTP user, HTTP

Data type: *alphanumeric*, max. 64 characters

If the other end requires authentication via user name and password (HTTP Basic Authentication) to establish the HTTP connection, enter the required user name here. This option is available for HTTP-only upload connections as well as for AS2 connections.

11.1.33 HTTP password, HTTP

Data type: *alphanumeric*, max. 64 characters

If the other end requires authentication via user name and password (HTTP Basic Authentication) to establish the HTTP connection, enter the required password here. This option is available for HTTP-only upload connections as well as for AS2 connections.

11.1.34 Pure HTTP (AS2 off), HTTP

Data type: *boolean*

Tick this field to use e-AS2 Enterprise for HTTP-only uploads. Learn more about this topic in Section 5.20 (*Using HTTP upload*).

11.1.35 MP form data, HTTP

Data type: *boolean*

Tick this field to also force preparation as `multipart/form-data` when sending individual files for an HTTP-only upload. Multifile transmission always (regardless of this field) uses `multipart/form-data`.

11.1.36 If problems occur when sending data, use this address, Mail

Data type: *alphanumeric*, max. 256 characters

Enter the mail address of the operator to be notified when there are problems sending data to this partner. If this field is blank, the `mail.to` mail address in `EAS2.properties` will be used as a fallback.

11.1.37 If problems occur when receiving data, use this address, Mail

Data type: *alphanumeric*, max. 256 characters

Enter the mail address of the operator to be notified when there are problems receiving data from this partner. If this field is blank, the `mail.to` mail address in `EAS2.properties` will be used as a fallback.

11.1.38 If timeout occurs while waiting for async MDN, use this address, Mail

Data type: *alphanumeric*, max. 256 characters

Enter the mail address of the operator to be notified when there is a timeout waiting for asynchronous MDNs. If this field is blank, the `mail.to` mail address in `EAS2.properties` will be used as a fallback.

11.1.39 If dry-running a send transaction, use this address, Mail

Data type: *alphanumeric*, max. 256 Zeichen

Enter the mail address of the person to receive data of dry-run send transactions (see Section 12.5 (*Workflow testing with dry-runs*)). If this field is blank, the `mail.to` mail address in `EAS2.properties` will be used as a fallback.

11.1.40 Command On Receive, Interfaces

This field only appears if execution of commands on receive (Chapter 9 (*Command on receive*)) was activated. Ticking will activate running the configured command on receive for this partner.

11.1.41 Force single file, Interfaces

Data type: *boolean*

Tick this field to force single file transmission when processing multifile INF files. In this case, e-AS2 Enterprise will initially generate a transaction for the multifile job, which itself is not processed. Instead, it is split into individual single file transactions, which will then be processed separately.

Please note, in the event of communication errors with a true multifile transfer, the transaction will fail as a whole. So consistency is maintained in any case, since related files either all arrive, or none. After splitting into individual transactions, partial transmission of individual files may occur whilst others will initially stay behind. Verify if your application can handle this before using the feature.

Multifile transmission in AS2, however, is an optional feature which may not be supported on the other end. In these cases you can then force single file transmission. If you are using the SMTP function to transmit data per e-mail, it is wise to use multifile transfer. The e-mail program used will most likely be able to handle this.

11.1.42 Customer number, Interfaces

If you connect customers via AS2, you can store the customer number assigned to the partner here for reference purposes. When receiving messages from that partner the customer number will be written to the RES file and can be evaluated by the subsequent systems.

11.2 Partner list, functions

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

11.2.1 Incremental search

Using the scrollable partner list, you can select a profile to which a function should refer by clicking on the profile. You can use an incremental search to make it easier to find specific profiles in long lists. Activate the partner list by selecting any profile. Then start entering successive letters in the profile you are looking for. With each letter entered the selection in the list jumps to the next profile matching the letters entered. The typed letters are displayed at the top of the partner list heading. (Not case sensitive. You may also enter numbers in addition to letters.)

Searching the list only uses the core identifiers AS2-From, AS2-To and Partner ID. The fill characters for the display (brackets and arrows) are not taken into account and need not be typed in.

Once you have found an entry you can jump to the next entry matching the same criteria with the spacebar. Press the ESC key to delete the search.

11.2.2 Add partner

This will open a dialogue where you can enter a new partner profile. Please be sure to use an internal identifier which is not already in use!

A validity check will run before saving to prevent generating an inoperable configuration. If this type of problem is detected, you will receive an informative error message and will be able to make corrections.

A new partner profile does not need to be complete right away. You can start with a subset of information you already know and complete the profile later.

Once the new profile has been added, e-AS2 will immediately also create a new subdirectory (corresponding with the internal identifier of the new profile) in the out directory of the interface area. So you can use this directory immediately to add send jobs. This directory will not be added, however, if you deactivated the e-AS2 directory interface in favour of the INF interface.

11.2.3 Add from transaction

To make it easier to add new profiles with preset values, you can import some values from a transaction as defaults. To do so, first switch to the tab “Transactions” and select a transaction. Then switch back to the partner manager and press the “Add from transaction” button. You will now see some fields in the new partner profile pre-populated. This function comes in handy in situations where your communications partner already sent a test message before you created the corresponding partner profile. You can then create the missing profile based on the error transaction of the failed transfer.

11.2.4 Duplicate partner

In order to create new profiles based on already existing profiles, press the “Duplicate partner” button. This button becomes active when an existing profile is selected in the list. The same dialog is opened as for “Add partner”, but all fields are pre-filled with the values of the selected profile. You then do the necessary modifications and then save the profile with a new ID.

11.2.5 Copy partner to remote e-AS2

When pressing this button a dialog is opened, that supports copying partner profiles from the current e-AS2 instance to another instance. The list of instances that is offered to choose the copy target from, is taken from the GUI launcher configuration. So, if the GUI launcher isn't used to connect to various e-AS2 instances in your environment, the copy partner feature will not be available to you.

Extensive plausibility controls are in place to assist the user in the copy process. Dependencies on other configuration items like certificates or mail server profiles are taken into account. If necessary (and possible) such other configuration items are automatically copied to the target instance along with the partner profile in question.

11.2.6 Delete partner

This can be used to delete partner profiles you no longer need. Be sure not to delete profiles referenced from other profiles with the master mechanism!

For safety reasons, deleting a partner profile will not automatically delete the associated interface directory, as it may still contain unprocessed files which must not be lost. Delete the directory manually once it is no longer needed! e-AS2 Enterprise identifies a directory which is no longer needed by adding the file `_DEAD_` in this directory.

11.2.7 Edit partner

Select a partner profile from the list and press this button to edit the data in this profile. With the exception of the internal identifier, which cannot be changed, you can correct or add to any values. The same plausibility checks are executed as for a new entry.

11.2.8 Change Partner-ID

Select a partner profile in the list and then press this button, to change the internal identifier of this profile. On save the profile will be stored with the new ID. Also every existing transaction will be modified so that they reference the new partner ID.

11.2.9 Search

This opens a search dialog for partner profiles. The search dialog offers flexible search based on combinations of:

- Partner-ID

- Customer number
- AS2-From
- AS2-To

You can enter partial strings in the search fields. All profiles will be found, that contain the strings. You can have more than one hit for a search.

If the secondary interface area has been activated in addition to the primary area, GUI elements also appear that allow you to search for interface usage.

11.2.10 Clear search

The search criteria will be removed. The complete list of all partner profile is shown again.

11.2.11 Re-read list

This re-reads the list of partner profiles from the e-AS2 server. Should new partner profiles have been added by other GUI clients that are simultaneously connected to the same server, they will the appear as new entries in the list.

11.2.12 Send certificate

Select a certificate in certificate management. Then select a partner profile from the partner list and press the “Send certificate” button to send the selected certificate to the respective partner. If you selected a private key, the private key will not be sent. So the other party will receive a public key certificate. This is also the background of the function. This allows you to distribute public keys associated with your private key to your communication partners to prepare for using cryptography functions.

Please note, this function will only be helpful if your partner’s AS2 software is capable of exchanging keys via AS2.

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

11.2.13 Send file

This buttons allows you to quickly and easily transfer test data to the other end. This is not intended for productive use with business data.

Select a partner profile from the partner list and press the “Send file” button. This will open a dialogue for selecting the file to be transferred. After making your selection the file will be loaded and immediately sent to the other end. 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. Instead, select a file on your client computer. If the GUI and e-AS2 server are not running on the same computer, this will make things easier in the test phase of new connections.
2. The transfer to the other end will start immediately after selecting the file. 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.

   If the Ack mode is set to “asynchronous” in the partner profile, the GUI client will wait a while for the asynchronous MDN to arrive. If it is received within the configured wait time, the result will be shown in a message box. Otherwise the GUI client will abort and notify the user.

3. A retry pattern in the partner profile will be ignored. In the event of an error, 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. (This cannot work for technical reasons, as a copy of the file to be transmitted is not saved to the server file system.)

11.2.14 Receive file (dry-run)

Press this button to simulate receiving data from the selected partner. This function comes in handy during initial setup and testing of new partner connections. Read more about the dry-run feature in Section 12.5 (*Workflow testing with dry-runs*).

11.2.15 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 Enterprise 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 Enterprise also have this property. This process creates a so-called self-signed certificate, where the owner and issuer are identical.[79]

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.

[79]e-AS2 Enterprise can also use certificates from a Trust Center without a problem. From a technical perspective, self-signed certificates do not differ from certificates issued by a Trust Center.

11.3.6 Operator mail

This shows the e-mail address assigned for operator e-mails. Please note, this e-mail address is internally managed in e-AS2 and is not part of the certificate.

As the certificate expiration date approaches, e-AS2 Enterprise will send an operator mail. The email address configured in this field will be used as the target address for such operator mails. If the field has been left empty, the `mail.to` address configured in `EAS2.properties` will be taken as a fallback.

11.3.7 Description

Data type: *alphanumeric*, max. 512 characters

This is an arbitrary text field for entering descriptive comments related to the certifcate.

11.4 Certificate list, functions

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

The buttons are distributed across three sub-tabs. The tab “All” contains the functions available for all certificates. The tab “Public” contains functions related to the partners’ public key certificates. The tab “Private” contains the functions related to private key certificates. Depending on the tab selected the list at the left will only show the certificates which apply to the selection.

The certificate list consists of five 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.
Dflt 1	This is ticked if this is the first default certificate. (see Section 5.18 (*Encryption and signature, in detail*))
Dflt 2	This is ticked if this is the second default certificate. (see Section 5.18 (*Encryption and signature, in detail*))

11.4.1 Import public key

Use this function to import public key certificates received from your communication partners. The import function processes Base64-encoded as well as binary formats. The certificate file must be located in a file system the GUI has read access to at the time of import. You can assign the certificate an alias of your choice at the time of import. Select a meaningful alias you can easily associate with the respective partner! The alias is later used during partner profile configuration to pick the certificate from the list.

Please note, existing certificates cannot be updated. If your communication partner provides a new version of a certificate which already exists in the e-AS2 manager, import it using a new alias. You will then temporarily have both versions in certificate management and use the methods for smooth certificate transition as described in Section 5.18 (*Encryption and signature, in detail*).

11.4.2 Import private key

This function allows you to import a private key certificate from a file. The certificate can be a Java KeyStore or PKCS12 KeyStore.

This function is used if you have a complete private key certificate from another source and want to use it in e-AS2 Enterprise. These certificates are typically PKCS12 KeyStores with the file extension `.pfx` or `.p12`. The MS Windows certificate manager will export e.g. Private Key certificates as PFX files.

Since this type of file containing the private key contain security related information which should be protected, KeyStores are always password protected. The import process will ask for the password before starting the import.

11.4.3 Generate private key

Alternatively to using certificates generated externally, you can generate your own certificates in e-AS2 Enterprise. If you press the “Generate private key” button you will first be prompted to enter an alias for the new certificate. This will open a dialogue prompting the following information:

Name	Enter the name of a person responsible for this process. This is typically your name.
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.[80]
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 two 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. The new certificate can then be used immediately. Distribute the associated public key to the communication partners you wish to exchange encrypted data with, using the new certificate! (see Section 11.4.7 (*Export public key*))

11.4.4 Set default key

e-AS2 Enterprise may have several private key certificates in management. In the event of a signature process, e-AS2 must decide on a certificate to use for the signature. For this purpose one certificate is set as the default certificate.

When receiving encrypted messages the default certificate is one of several candidates to be used for decryption.

[80]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.

Please note, you may configure additional partner-related certificates for the various cryptography functions.

11.4.5 Set 2nd default key

This allows you to set a second default certificate in order to use the smooth certificate transition as described in Section 5.18 (*Encryption and signature, in detail*). It as being used as a second candidate for decrypting incoming data.

11.4.6 Export private key

If you generated your private key in e-AS2 Enterprise, you may want to save it in a separate file and keep it as a backup copy. This button opens a separate dialogue for saving the file where you can specify the desired location and file name for the save. If you leave out the file extension, `.ks` will automatically be attached. The private key is saved in a Java KeyStore and can then be opened with a separate tool, e.g. the `keytool` in the Java Runtime environment.

11.4.7 Export public key

If you are unable to use the e-AS2 function “Send certificate” (see Section 11.2.12 (*Send certificate*)), you will need to export the public key certificate for the respective private key certificate to be able to transmit to your communication partner using a different method (e.g. via email). Select a directory in the save dialogue and enter a file name without file extension. After completing your entry two files with the name you specified will be generated in the selected directory, of which one will have the file extension `.der` and the other the file extension `.pem`.

The first file contains the public key certificate in binary format (DER encoded). The second file contains the same certificate in ASCII format (Base64 encoded). In practice you may be dealing with partners whose AS2 systems can only import one format or the other. It’s therefore best to provide your partner with both files. One should always be importable.

11.4.8 Export certification request

To have a certificate certified by an official Trust Center (or CA, Certification Authority), start by generating a self-signed certificate as described in Section 11.4.3 (*Generate private key*). You will then be able to export a certification request with this function. Again, only enter a file name without file extension. During the save process two files will be generated, a binary file with the extension `.der` and a text file with the extension `.csr`. Send this file to the CA you selected.

Such service providers often have web applications for uploading the respective file. Sometimes there will also be a text field where you can enter your certification request via copy and paste. You can use the contents of the CSR file for this purpose.

11.4.9 Import certification reply

After the certification process has been completed by the Trust Center, you will receive a certification response. Import it using the respective function in e-AS2. The respective private key will then no longer be self-singed, but signed by the CA.[81] This certificate update pertains to the entry currently selected in the list.

11.4.10 Edit certificate

This is used to assign the selected certificate an email address for operator mail or edit an existing address. Also you may enter or update the comment/description field for that certificate entry in the list.

[81]This fact should also be reflected in the certificate details shown below the list.

11.4.11 Copy certificate to remote e-AS2

When pressing this button a dialog is opened, that supports copying certificates from the current e-AS2 instance to another instance. The list of instances that is offered to choose the copy target from, is taken from the GUI launcher configuration. So, if the GUI launcher isn't used to connect to various e-AS2 instances in your environment, the copy certificate feature will not be available to you.

11.4.12 Delete selected key

This is used to delete certificates provided they are not currently in use. In use are the two default certificates and any certificates referenced in partner profiles. Please note, you can not undo the deletion. Although imported external certificates can be reimported if necessary, a certificate self-generated in e-AS2 is permanently lost after being deleted.

11.4.13 Show partners

Select a certificate from the list and press “Show partners”. The GUI application will then switch to the “Partner” tab and show all the partner profiles referencing the selected certificate. On the internal tabs, press “All” to view all partner profiles again.

11.4.14 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”, “Re-parse message”, “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.

When using the INF interface with AS2-From and AS2-To, 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.

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.

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 or the INF file.

Receive

At the time of receipt this information is taken from the data received and entered so it fits the configuration of the partner profiles. Since the AS2-From and AS2-To fields are configured from the partner profiles from the perspective of a data transmitter, they must be mentally reversed when receiving data. I.e. the two elements in the AS2 relation are reversed accordingly when receiving data before being entered in the AS2-From and AS2-To fields.

11.5.3 AS2-To

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

Send

This information is copied from the AS2-To field of the partner profile.

Receive

At the time of receipt this information is taken from the data received and entered so it fits the configuration of the partner profiles. Since the AS2-From and AS2-To fields are configured from the partner profiles from the perspective of a data transmitter, they must be mentally reversed when receiving data. I.e. the two elements in the AS2 relation are reversed accordingly when receiving data before being entered in the AS2-From and AS2-To fields.

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 HTTP Login

User name and password for HTTP basic authentication.

Send

If authentication with user name and password (HTTP basic authentication) was required when establishing the HTTP connection to the other party, this will contain the respective information.

Receive

This field is always blank.

11.5.8 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.
- (new inf – alternative initial status when using the INF interface)
- (split – status after splitting a multifile transaction)
- 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.
- frozen - data transmission to a partner with data fetch extension activated failed. Transaction has been put on hold (frozen) until the partner connects again.

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.

11.5.9 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.

11.5.10 In Process

“In Process” is ticked if one of the processing threads in e-AS2 Enterprise 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.

11.5.11 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.

11.5.12 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

The complete path under which the received file was saved in the file system is listed under the file name. This is always a file with a generated name saved to the `in` directory or the `orphan` directory of the interface area. Please note, this is not the file name transmitted by the remote system. (see Section 11.5.13 (*AS2 name*))

11.5.13 AS2 name

The virtual name of the transmitted file.

Send

Only the file name itself (without path) is entered here. Unless otherwise specified, it is identical with the physical file name in the file system. The INF interface (see Chapter 8 (*The INF interface*)) may specify an alternate virtual name which will then appear in this field.

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.

11.5.14 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.

Send

e-AS2 Enterprise 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.

11.5.15 Size

This field shows the length of the transmitted file or the length of the message .

Send

For single file transmission, this contains the exact length of the transmitted file.

In the case of multifile transfers, the sum of the individual file lengths is displayed here.

Receive

For single file transmission, this contains the exact length of the transmitted file.

In the case of multifile transfers, the length of the MIME message received is entered here. This is always a bit longer than the sum of the individual file lengths.

11.5.16 Document-ID

This is an identifier from the perspective of the application supplying the data for transmission. This should clearly identify the process locally, which is that application’s responsibility. e-AS2 Enterprise particularly does not force local uniqueness.

Send

The INF interface (see Chapter 8 (*The INF interface*)) can add a unique identifier relating to the respective job as the document ID. This identifier has no meaning for e-AS2. It is neither validated nor otherwise processed. It is merely saved to the transaction record in the database and appears in this field in the transaction details. If an MDN is received for this transaction following transmission, the document ID is saved in the result file associated with the MDN in the `in_mdn` directory. It is further a component of delivery reports, that are generated after transaction processing is finished.

In addition to the document ID a so called "original document ID" can be passed to e-AS2 as content of the INF file. This covers situations, where the application supplying the data for transmission itself got the data from yet another system which already assigned a document ID. This would then be the original document ID.

The GUI field in the transaction details displays the original document ID and after that the document ID in brackets. If either ID is not present a minus character is display instead.

Receive

On receive e-AS2 generates a unique document ID for the transaction that is then passed to the subsequent processing system as part of the RES file. See Section 10.2.30 (*Settings for Creating Extended Document Identifiers*) for more details on this topic.

11.5.17 Subject

This field contains the subject for the MIME message, that was sent or received. The use of a subject is optional in AS2. So the field can be blank. In the send direction the subject can only be added when using the INF interface (see Chapter 8 (*The INF interface*)).

11.5.18 File type

This field contains the file type of the transmitted file, coded as a MIME type. For multifile transaction this field remains empty. The file types of the different files contained in a multifile transmission can be looked up in the multifile details window, which is opened, when you click into the Filename field.

11.5.19 Created

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

11.5.20 Changed

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

11.5.21 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”. There are several digest algorithms available to choose from. By default e-AS2 Enterprise uses SHA1.[82] In addition to that e-AS2 Enterprise supports MD5[83], SHA-224, SHA-256, SHA-384 and SHA-512[84]. 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. [85]

Send

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

[82]US Secure Hash Algorithm 1 (SHA1), see http://www.ietf.org/rfc/rfc3174.txt
[83]The MD5 Message-Digest Algorithm, see http://www.ietf.org/rfc/rfc1321.txt
[84]US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF), see https://tools.ietf.org/html/rfc6234
[85]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 week digest algorithm like SHA1 at this place does in fact pose no security risk. Nonetheless there are AS2 systems, that expect usage of stronger digest algorithms for the MIC. Therefore it has been made possible to configure the MIC algorithm separately per partner profile.

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”.

11.5.22 Algorithm

This field contains the identifier of the algorithm used for computing the message digest. See chapter above for the list of supported digest algorithms.

11.5.23 Error

This is ticked if an error occurred processing this transaction. The contents of the “Description” (Section 11.5.30 (*Description*)) or the e-AS2 log file can help in diagnosing the cause for the error.

11.5.24 Sign MDN

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

11.5.25 MDN error

This is ticked if an error occurs while sending an asynchronous MDN for this transaction. The contents of the “Description” (Section 11.5.30 (*Description*)) or the e-AS2 log file can help in diagnosing the cause for the error.

The field is only used for incoming transactions.

11.5.26 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.

11.5.27 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.

11.5.28 SMTP

This is ticked if the transaction is an outgoing or incoming e-mail. This can only occur if you licensed the SMTP Feature for e-AS2 Enterprise.

11.5.29 MDN via SSL

This is ticked if the other end requested an MDN to be sent via HTTP/S (SSL). This type of request is only useful for asynchronous MDN.

The field is only used for incoming transactions.[86]

11.5.30 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.

[86]If e-AS2 is configured to request asynchronous MDNs for outgoing transmissions, the MDN requests will always match the send protocol. I.e. when sending via HTTP, the MDN will also be requested as HTTP. When sending via HTTP/S, the MDNs will also be requested as HTTP/S.

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 Search

Use the “Search” button to open the dialogue for entering search criteria. You may flexibly select a subset of transactions from all the transactions in the database. This is not limited to the current day. (For details, please read Section 5.17.10 (*Search function*).)

11.6.3 Clear search

The search criteria established using the search dialogue also remain after closing the search dialogue. To return to the default state, use the “Clear search” button.

11.6.4 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”.

11.6.5 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.

11.6.6 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.

11.6.7 Re-try now

========== ADDED (JK: 28.11.2022) ==========

This function only applies to send transactions. It is available for transactions while retries according to the defined retry pattern are still ongoing. If the time to wait for the next retry seems to be to long an immediate retry can be forced by pressing this button. The retry counter is the reset to zero and the retry pattern re-starts at the beginning.

11.6.8 Re-parse message

This function only applies to receive transactions. During setup of new connections it is not uncommon, that incoming messages are completely received but cannot be successfully processed. Two common reasons for that to happen, are the corresponding partner profile not being identified correctly or failing decryption or signature verification due to certificates not matching. In both cases it's worth to try and process the message again after having fixed the configuration problem.

This can be done by pressing the button “Re-parse message”. This button will remain active until the message is finally successfully processed or the transaction is stopped. Using this functionality you can proceed with testing without the need to ask the other party to send the test message again.

11.6.9 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.[87] Such transactions purposely remain in open status as the responsible operator must clarify the situation. Once clarified the transaction should be stopped or the message re-parsed.

11.6.10 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.

[87] 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.

Keep in mind, successful transmission of payload to a communication partner does not automatically guarantee sending asynchronous MDNs will succeed. In this case the communication parameters will be taken from the received messages and could of course be different from the parameters in the partner profile. So when sending asynchronous MDNs you are dependent on the parameters being configured correctly on the other end. When in doubt, you can use an IO-Trace to verify. Learn more in Section 12.2 (*IO-Traces*).

11.6.11 Download

Use this button to download the file (either payload or MDN depending on status) currently pending transmission for the selected transaction to your local PC. The file is automatically selected on the server end depending on the status.

11.6.12 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 `EAS2.properties`.
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 in-

	terest 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 an 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 `EAS2.properties` 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	Die 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[88] 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. For more information on managing memory please refer to Section 12.4 (*Memory management, statistics logging*).

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

- Shut down server
- DB compactification[89]

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 Export / Import

This button is used to open a dialogue providing access to e-AS2 export/import functions. For more information please refer to Appendix A (*Export and Import*).

11.8.3 Audit log

This button is used to open the global audit log. For more information please refer to Section 5.21 (*Audit-Log*).

[88] Java Virtual Machine

[89] 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.

11.8.4 Connected GUI clients

This button provides a list of GUI clients currently signed into the e-AS2 server. The list shows each client name. This is followed by the IP address of the client computer in parentheses, along with information whether the respective client has admin permissions.

11.8.5 Close application

This will close the GUI client.

12 Diagnosing and debugging

Information on diagnosing communication errors and problems processing outgoing and incoming messages.

Normally the options in the graphical interface will fully suffice for monitoring operation and to respond to any irregularities. This particularly applies when working with existing and tested communications profiles. Occasionally – and particularly when setting up and testing new communication relations – you will need more advanced options for diagnosing and debugging to determine the cause for problems which may have occurred.

In this chapter we will try to familiarise you with the most important procedures for diagnosing problems. Some of them have already been outlined in other sections of this manual. Here we will address the key concepts in more detail. At this point we are assuming you have read and understand the message processing and particularly the AS2 cryptography information in this manual.

This chapter frequently uses the word “error” or “erroneous”. As a precaution, we’d like to mention we do not mean program errors which need to be resolved by correcting the software. The processing errors mentioned are a natural part of complex communication applications such as AS2. These errors can have different causes. We hope this chapter will help you quickly determine the cause just in case.

12.1 Logging

e-AS2 handles logging in files using the Apache library log4j2[90]. During initial installation of e-AS2 a configuration will be prepared, that results in log messages being written to the file `eas2s.log` in the e-AS2 installation directory.

12.1.1 Adjusting logging parameters

As the number of daily transactions increases, the default logging configuration may no longer be ideal. The file `log4j2.properties` contains two prepared configurations for alternative models of creating log files.

One of these configurations sets a limit to the maximum log file size, triggering a rolling file mechanism, when that limit is reached. The other configuration works with a daily log file; rolling takes place shortly past midnight. Choose whatever configuration seems suitable to you and modify to your liking.

12.1.2 Log Level

The log4j2 library knows the following log levels:

- TRACE
- DEBUG
- INFO
- WARN
- ERROR
- FATAL

[90] For detailed information on log4j2 please refer to https://logging.apache.org/log4j/2.x/manual/.

The default log level used in e-AS2 is `INFO`. The higher up in the list the log level is, the more detailed the logging, so the more log output is written. We recommend setting the log level to `INFO` during normal production operation. When setting up new connections and for diagnosing malfunctions you may temporarily move the log level up to `DEBUG`. Moving up to `TRACE` should only be done if prompted to do so by e-integration support. This will generate some very specific messages the end user may not comprehend.

The log level is at the beginning of each row of the log, right next to the timestamp. Watch for messages with level `WARN`, `ERROR` and `FATAL` to identify irregularities.

12.1.3 General approach for diagnosing

In most cases, the need to study log entries will be triggered by an operator mail or an erroneous transaction entry in the GUI. In both cases, you will know the transaction number. Look at the following excerpt from an `INFO` level e-AS2 log:

```
09:19:53,209  INFO [DIRTRANS-5] [-] [-] - ### START DIRECTORY TRANSFER THREAD ###
09:19:53,211  INFO [ IFTRANS-5] [-] [-] - parsing inf file 'DATEN.4621.inf'
09:19:53,212  INFO [ IFTRANS-5] [-] [O4621] - ==> file to transfer: 'DATEN.4621'
09:19:53,215  INFO [ IFTRANS-5] [520] [O4621] - new file 'DATEN.4621.inf' belongs to partner 'acme_corp'
09:19:53,224  INFO [ IFTRANS-5] [520] [O4621] - transmitting message - MSGID: EAS2-20200630091953...
09:19:53,229  INFO [ IFTRANS-5] [520] [O4621] - connecting to as2.acme.corp(207.209.155.118):4080 ...
09:19:53,231  INFO [ IFTRANS-5] [520] [O4621] - transmitting ...
09:19:53,232  INFO [ IFTRANS-5] [520] [O4621] - data sent (271 bytes) - waiting for ack ...
09:19:53,295  INFO [ IFTRANS-5] [520] [O4621] - received ACK or sync MDN
09:19:53,296  INFO [ IFTRANS-5] [520] [O4621] - ACK received from ...
09:19:53,324  INFO [DIRTRANS-5] [-] [-] - ### END DIRECTORY TRANSFER THREAD ###
```

The first information in square brackets is the thread name. Then is – also in square brackets – the transaction number follows. We are looking at all `INFO` level log messages for transaction 520. So, if you know the transaction number, you can use the full text search in your text editor for the string “open square bracket - transaction number - close square bracket” to find the log entries related to this transaction. The last technical information added in square brackets is the document ID for the transaction, in this case: O4621.

If you read the text messages to the right of the dash in the individual log messages in chronological order, you will easily recognise the process as shown in Figure 2.2, “Data exchange via AS2”.

Please note, the transaction number is listed in all log entries for this transaction. These are not necessarily consecutive. On systems with high usage the log entries for different processes could be mixed. In the case of a retry following a communication error, log entries for the same transaction will appear spaced farther apart. For a full picture of the processes, look at all entries for the respective transaction.

12.1.4 Case examples

The following are some case examples describing typical diagnosis scenarios.

Logging a receive transaction

The following listing shows the log entries for a receive transaction requesting an asynchronous MDN. (For lack of space we omitted the timestamps.)

```
09:36:33,858  INFO [  ICH_HTTP] [-] [-] - waiting for data connection - listening on port 6080
09:36:33,858  INFO [    RCV-13] [-] [-] - ### START RECEIVER THREAD ###
09:36:33,859  INFO [    RCV-13] [-] [-] - protocol: HTTP, local port: 6080, remote host: 207.209.155.118
09:36:33,870  INFO [    RCV-13] [-] [-] - AS2-From: AcmeCorp
09:36:33,871  INFO [    RCV-13] [-] [-] - AS2-To: MyCompany
09:36:33,901  INFO [    RCV-13] [-] [-] - message received from >e-integration AS2 Server V devel<
09:36:33,902  INFO [    RCV-13] [-] [-] - received file 'ORDERS.632'...
09:36:33,905  INFO [    RCV-13] [525] [D4000000000000033@EAS2] - received data from partner 'acme_corp',
                                                                   generated docid: D4000000000000033@EAS2
```

```
09:36:33,922  INFO [    RCV-13] [525] [D4000000000000033@EAS2] - received message - sending ACK
09:36:33,928  INFO [    RCV-13] [525] [D4000000000000033@EAS2] - asynchronous MDN will be sent later
09:36:33,965  INFO [    RCV-13] [525] [D4000000000000033@EAS2] - ### END RECEIVER THREAD ###
09:36:53,846  INFO [DIRTRANS-7] [-] [-] - ### START DIRECTORY TRANSFER THREAD ###
09:36:53,866  INFO [ IFTRANS-7] [525] [D4000000000000033@EAS2] - MDN belongs to partner 'acme_corp'
09:36:53,878  INFO [ IFTRANS-7] [525] [D4000000000000033@EAS2] - transmitting message
09:36:53,907  INFO [ IFTRANS-7] [525] [D4000000000000033@EAS2] - connecting to as2.acme.corp ...
09:36:53,914  INFO [ IFTRANS-7] [525] [D4000000000000033@EAS2] - transmitting ...
09:36:53,915  INFO [ IFTRANS-7] [525] [D4000000000000033@EAS2] - sent MDN - waiting for ack ...
09:36:53,989  INFO [ IFTRANS-7] [525] [D4000000000000033@EAS2] - ACK received from >AS2 Server<
09:36:53,994  INFO [DIRTRANS-7] [-] [-] - ### END DIRECTORY TRANSFER THREAD ###[91]
```

Please note, the first log messages for the receiver thread (RCV-13) do not contain a transaction number. This is always the case for receive transactions, since the transaction will only be added to the database after receiving has completed and the process has been assigned to a partner profile. As you can see, receiving the data is reflected in the transaction as number 525. Once you have determined the processing thread from the transaction number, RCV-13 in the above example, you can keep all log messages for the respective processing phase together using this thread name.

The job of the receiver thread ends with the statement an asynchronous MDN will be sent later. This will occur in a separate HTTP session, as demanded by the AS2 protocol.

The log entries for sending the asynchronous MDN originate from the sender thread (IFTRANS-7). Please note, these log entries also contain transaction number 82724525. This allows the entire process to be kept together, even if it spans a long period and is processed in various threads.

Receive transaction with cryptography

Whilst using cryptography doesn’t pose much of a problem when sending data, the probability of cryptography processing errors when receiving data is much greater. First we will show a session without errors.

```
15:14:46,421  INFO [  ICH_HTTP] [-] - incoming HTTP connection to port 6080 from 207.209.155.118
15:14:46,422  INFO [  ICH_HTTP] [-] - waiting for data connection - listening on port 6080
15:14:46,422  INFO [     RCV-2] [-] - ### START RECEIVER THREAD ###
15:14:46,422  INFO [     RCV-2] [-] - protocol: HTTP, local port: 6080, remote host: 207.209.155.118
15:14:46,431  INFO [     RCV-2] [-] - AS2-From: AcmeCorp
15:14:46,431  INFO [     RCV-2] [-] - AS2-To: MyCompany
15:14:46,432  INFO [     RCV-2] [-] - DECRYPTING...
15:14:46,440  INFO [     RCV-2] [-] - DECOMPRESSING...
15:14:46,443  INFO [     RCV-2] [-] - VERIFYING...
15:14:46,444  INFO [     RCV-2] [-] - verifying - trying 1st certificate (acme_corp)
15:14:46,447  INFO [     RCV-2] [-] - message received from >e-integration AS2 Server V devel<
15:14:46,447  INFO [     RCV-2] [-] - received file 'TESTFILE' (23 bytes, Message-ID: EAS2-20200630101446...)
15:14:46,449  INFO [     RCV-2] [54] - received data from partner 'acme_corp', ...
15:14:46,473  INFO [     RCV-2] [54] - received message - sending sync MDN
15:14:46,476  INFO [     RCV-2] [54] - ### END RECEIVER THREAD ###
```

As you can see, that in this case processing takes a few more steps before the transaction record is created and assigned number 54. Processing the data required – in this order – the following cryptography related three steps:

DECRYPTING	The data received was decrypted.
DECOMPRESSING	The resulting data was decompressed.
VERIFYING	The resulting data underwent signature verification.

If you don’t see any additional messages in the log, you can assume all steps were completed successfully. Please note, the AS2 standard does not specify an order for the two steps signature and compression. There are AS2 solutions which first sign, then compress the result. How-

[91]Note, that we removed the document ID in all further examples, to save some space in the printed log lines.

ever, there also are some which fist compress, then sign. In either case, encryption is last.[92] Accordingly, depending on the sender, after decryption you may first see decompressing and then signature verification, as shown in the above example. In other cases, however, following decryption you may also first see signature verification and then decompressing. Both variants are conformant to the standard and fine as long as there are no error messages in the log.

Errors during decryption

We will now look at the receive process with an error during decryption.

```
15:27:11,753  INFO [  ICH_HTTP] [-] - incoming HTTP connection to port 6080 from 207.209.155.118
15:27:11,755  INFO [  ICH_HTTP] [-] - waiting for data connection - listening on port 6080
15:27:11,755  INFO [     RCV-3] [-] - ### START RECEIVER THREAD ###
15:27:11,755  INFO [     RCV-3] [-] - protocol: HTTP, local port: 6080, remote host: 207.209.155.118
15:27:11,769  INFO [     RCV-3] [-] - AS2-From: AcmeCorp
15:27:11,769  INFO [     RCV-3] [-] - AS2-To: MyCompany
15:27:11,770  INFO [     RCV-3] [-] - DECRYPTING...
15:27:11,774  WARN [     RCV-3] [-] - decryption error
15:27:11,775  INFO [     RCV-3] [-] - message received from >e-integration AS2 Server V devel<
15:27:11,775  INFO [     RCV-3] [-] - received file 'smime.p7m' (2402 bytes, Message-ID:
 EAS2-20200630102711...)
15:27:11,777  INFO [     RCV-3] [55] - received data from partner 'acme_corp', ...
15:27:11,778  WARN [     RCV-3] [55] - transaction is erroneous - sending operator mail
15:27:11,798  INFO [     RCV-3] [55] - received message - sending sync MDN
15:27:11,800  INFO [     RCV-3] [55] - ### END RECEIVER THREAD ###
```

As you can see, processing is cancelled after decryption failed. An operator mail will be sent. And of course an MDN containing an error message is returned to the sender of the data. However, no attempt is made to continue processing the data. This would not make sense after decryption failed. As a result of processing the reception, the still encrypted file `smime.p7m` is saved to the file system.

In this case, the GUI will show the error message “LOCAL: recipient key mismatch”, indicating the certificate used by the other party for encrypting the data does not match any of the certificates used locally to attempt to decrypt after receiving the data.

For more details about the process, we can set the log level to `DEBUG` and request the opposite party to retransmit the data. The log will then contain significantly more information than before.

```
15:54:29,106  INFO [  ICH_HTTP] [-] - incoming HTTP connection to port 6080 from 207.209.155.118
15:54:29,120  INFO [  ICH_HTTP] [-] - waiting for data connection - listening on port 6080
15:54:29,120  INFO [     RCV-4] [-] - ### START RECEIVER THREAD ###
15:54:29,121  INFO [     RCV-4] [-] - protocol: HTTP, local port: 6080, remote host: 207.209.155.118
15:54:29,121 DEBUG [     RCV-4] [-] - [[POST /as2in HTTP/1.1]]
15:54:29,121 DEBUG [     RCV-4] [-] - [[Host: as2.doecorporation.com:6080]]
15:54:29,122 DEBUG [     RCV-4] [-] - [[Connection: close]]
15:54:29,122 DEBUG [     RCV-4] [-] - [[User-Agent: e-integration AS2 Server V devel]]
15:54:29,123 DEBUG [     RCV-4] [-] - [[Date: Tue, 30 Jun 2020 10:54:29 -0300]]
15:54:29,123 DEBUG [     RCV-4] [-] - [[AS2-Version: 1.2]]
15:54:29,123 DEBUG [     RCV-4] [-] - [[EDIINT-Features: multiple-attachments, CEM]]
15:54:29,123 DEBUG [     RCV-4] [-] - [[AS2-To: MyCompany]]
15:54:29,123 DEBUG [     RCV-4] [-] - [[AS2-From: AcmeCorp]]
15:54:29,123 DEBUG [     RCV-4] [-] - [[Content-Length: 2402]]
15:54:29,124 DEBUG [     RCV-4] [-] - [[MIME-Version: 1.0]]
15:54:29,124 DEBUG [     RCV-4] [-] - [[Message-ID: <EAS2-20200630105429-064-000011@192.168.200.122-komsrv]]
15:54:29,125 DEBUG [     RCV-4] [-] - [[Subject: test transmission]]
15:54:29,126 DEBUG [     RCV-4] [-] - [[Disposition-Notification-To: http://as2.acme.corp:4080/comm/as2]]
15:54:29,127 DEBUG [     RCV-4] [-] - [[Content-Type: application/pkcs7-mime; smime-type=enveloped-data...]]
15:54:29,127 DEBUG [     RCV-4] [-] - doclen: Optional[2402]
15:54:29,128  INFO [     RCV-4] [-] - AS2-From: AcmeCorp
15:54:29,128  INFO [     RCV-4] [-] - AS2-To: MyCompany
15:54:29,128 DEBUG [     RCV-4] [-] - MIC requested: optional
15:54:29,129 DEBUG [     RCV-4] [-] - MIC algorithm: SHA1
15:54:29,129  INFO [     RCV-4] [-] - DECRYPTING...
15:54:29,130 DEBUG [     RCV-4] [-] - loaded private key 'priv' from keystore
15:54:29,130 DEBUG [     RCV-4] [-] - loaded private key 'priv2' from keystore
```

[92] In e-AS2, coding of data before transmission happens in the fixed order signature-compression-encryption. This cannot be changed.

```
15:54:29,131 DEBUG [     RCV-4] [-] - cipher algorithm: AES256-CBC
15:54:29,132 DEBUG [     RCV-4] [-] - loaded private key 'tlscert' from keystore
15:54:29,132 DEBUG [     RCV-4] [-] - loaded private key 'my2ndKey' from keystore
15:54:29,133 DEBUG [     RCV-4] [-] - decrypt - trying private key 1 from partner profile
15:54:29,133 DEBUG [     RCV-4] [-] - decrypt - trying private key 2 from partner profile
15:54:29,133 DEBUG [     RCV-4] [-] - decrypt - trying global private key 1
15:54:29,133 DEBUG [     RCV-4] [-] - decrypt - trying global private key 2
15:54:29,133  WARN [     RCV-4] [-] - decryption error
15:54:29,134 DEBUG [     RCV-4] [-] - BAOS
15:54:29,135 DEBUG [     RCV-4] [-] - micAlg: SHA1 (O+yeE9ldzsGaZW4QdoBjbhjOLYc=)
15:54:29,135  INFO [     RCV-4] [-] - message received from >e-integration AS2 Server V devel<
15:54:29,135  INFO [     RCV-4] [-] - received file 'smime.p7m' (2402 bytes, Message-ID: EAS2-2020...)
15:54:29,137  INFO [     RCV-4] [56] - received data from partner 'acme_corp', generated docid: D40000000...
15:54:29,138  WARN [     RCV-4] [56] - transaction is erroneous - sending operator mail
15:54:29,139 DEBUG [     RCV-4] [56] - preparing MDN
15:54:29,154  INFO [     RCV-4] [56] - received message - sending sync MDN
15:54:29,156  INFO [     RCV-4] [56] - ### END RECEIVER THREAD ###
```

Using log level DEBUG will log all full header lines received at the start of the session.[93] After the INFO line “DECRYPTING...” the various steps are logged in detail during the attempt to decrypt the data. Four certificates are loaded and their validity reviewed at this time. Up to four decryption attempts may possibly be made, in the order using Key 1 and Key 2 from the partner profile, then using the global default Key1 and Key2. This would be cancelled after the first successful attempt. In this case, all four attempts fail.

As you can see, debug logging provides very detailed information on internal data processing in e-AS2. Carefully studying the log entries will typically quickly show the causes for a processing error.

Error during signature check

The final case example shows an error during signature check. I.e. the data received was signed, but the signature was not successfully verified. Generally, there can be two possible types of invalid signatures.

1. The signature is mathematically correct, but the certificate that was used for signing the message has either expired or is wrong. This case is quite common – especially when new communication relationships are being established – and generally indicate a simple (inter-personal) communication problem. Further down we will see how e-AS2 supports you in determining the cause for the problem.
2. The signature is mathematically incorrect. I.e. the digest on the user data transmitted by the sender is different from the digest computed by e-AS2 after receiving the data. This is a serious error indicating either the sender installed the signature component incorrectly or – worse! – the data was changed along the communication path. This can be due to a technical error, but worst case also due to a malicious third party attack. In either case, the cause should be thoroughly investigated.

We will start with an example of the first case: unexpected signature certificate. The following excerpt from the log file shows a typical message for this case: “signature verify failed - serial numbers don't match”. This also lists the two serial numbers involved, first the serial number of the expected certificate, then the serial number of the certificate actually found in the signature.

```
 INFO [     RCV-6] [-] - ### START RECEIVER THREAD ###
 INFO [     RCV-6] [-] - protocol: HTTP, local port: 6080, remote host: 207.209.155.118
 INFO [     RCV-6] [-] - AS2-From: AcmeCorp
 INFO [     RCV-6] [-] - AS2-To: MyCompany
 INFO [     RCV-6] [-] - DECRYPTING...
 INFO [     RCV-6] [-] - DECOMPRESSING...
 INFO [     RCV-6] [-] - VERIFYING...
 INFO [     RCV-6] [-] - verifying - trying 1st certificate (acme_corp_2)
```

[93]Documenting the meaning behind the various header information in detail is beyond the scope of this manual. Those interested in the topic should study the RFCs.

```
 WARN [     RCV-6] [-] - serial numbers don't match: 1588153266037 (0x171c54faf75) -
                                                     1588155246439 (0x171c56de767)
 WARN [     RCV-6] [-] - verification error (signature verify failed - serial numbers don't match:
                         1588153266037 (0x171c54faf75) - 1588155246439 (0x171c56de767))
 INFO [     RCV-6] [-] - message received from >e-integration AS2 Server V devel<
 INFO [     RCV-6] [-] - received file 'TESTFILE' (23 bytes, Message-ID: EAS2-20200630111653...)
 INFO [     RCV-6] [58] - received data from partner 'acme_corp', generated docid: D4000000000000011@EAS2
 WARN [     RCV-6] [58] - transaction is erroneous - sending operator mail
 INFO [     RCV-6] [58] - received message - sending sync MDN
 INFO [     RCV-6] [58] - ### END RECEIVER THREAD ###
```

The following is an excerpt from the log for this process for log level “DEBUG”. Here you will receive additional information on the key alias of the locally configured certificate and the details of the actual certificate found.

```
 INFO [     RCV-6] [-] - ### START RECEIVER THREAD ###
 INFO [     RCV-6] [-] - protocol: HTTP, local port: 6080, remote host: 207.209.155.118
 INFO [     RCV-6] [-] - AS2-From: AcmeCorp
 INFO [     RCV-6] [-] - AS2-To: MyCompany
DEBUG [     RCV-6] [-] - MIC requested: optional
DEBUG [     RCV-6] [-] - MIC algorithm: SHA1
 INFO [     RCV-6] [-] - DECRYPTING...
 INFO [     RCV-6] [-] - DECOMPRESSING...
 INFO [     RCV-6] [-] - VERIFYING...
 INFO [     RCV-6] [-] - verifying - trying 1st certificate (acme_corp_2)
DEBUG [     RCV-6] [-] - verifying for key alias 'acme_corp_2'
DEBUG [     RCV-6] [-] -           serial number 1588153266037
DEBUG [     RCV-6] [-] - trying default verification mode
DEBUG [     RCV-6] [-] - signer: 1588155246439 EMAILADDRESS=john.doe@mycompanyltd.com, C=US, O=My Comp...
                       - issuer: CN=ACME Corporation,O=ACME Corporation,C=US,E=info@acme.corp
DEBUG [     RCV-6] [-] - digest algorithm: SHA-256
DEBUG [     RCV-6] [-] - encryption algorithm: RSA
 WARN [     RCV-6] [-] - serial numbers don't match: 1588153266037 (0x171c54faf75) -
                                                     1588155246439 (0x171c56de767)
DEBUG [     RCV-6] [-] - v4 extended key usage activated
DEBUG [     RCV-6] [-] - no 2nd key configured - verification failed
 WARN [     RCV-6] [-] - verification error (signature verify failed - serial numbers don't match:
                         1588153266037 (0x171c54faf75) - 1588155246439 (0x171c56de767))
DEBUG [     RCV-6] [-] - BAOS
DEBUG [     RCV-6] [-] - micAlg: SHA1 (py+16rNGf6Ki8N1/K+GQwmg2evM=)
 INFO [     RCV-6] [-] - message received from >e-integration AS2 Server V devel<
 INFO [     RCV-6] [-] - received file 'TESTFILE' (23 bytes, Message-ID: EAS2-20200630111653...)
 INFO [     RCV-6] [58] - received data from partner 'acme_corp', generated docid: D4000000000000011@EAS2
 WARN [     RCV-6] [58] - transaction is erroneous - sending operator mail
DEBUG [     RCV-6] [58] - preparing MDN
 INFO [     RCV-6] [58] - received message - sending sync MDN
 INFO [     RCV-6] [58] - ### END RECEIVER THREAD ###
```

The serial number, which is also indicated in the certificate details in the e-AS2 GUI, clearly identify the locally used certificates. In any case, the serial number is also a good reference when discussing this event with the sender of the data. They should also be able to identify the certificate based on the serial number. Discrepancies in the configuration can then quickly be clarified.

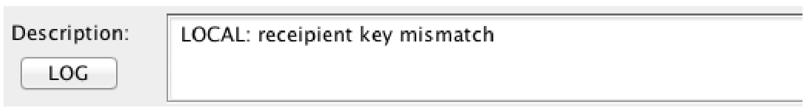
Next is an example (at log level “INFO”) for the critical case which you will hopefully never encounter in production.

```
INFO [    RCV-11] [-] - ### START RECEIVER THREAD ###
INFO [    RCV-11] [-] - protocol: HTTP, local port: 6080, remote host: 207.209.155.118
INFO [    RCV-11] [-] - AS2-From: AcmeCorp
INFO [    RCV-11] [-] - AS2-To: MyCompany
INFO [    RCV-11] [-] - VERIFYING...
INFO [    RCV-11] [-] - verifying - trying 1st certificate (acme_corp)
WARN [    RCV-11] [-] - verification error (message-digest attribute value does not match calculated value)
INFO [    RCV-11] [-] - message received from >e-integration AS2 Server V devel<
INFO [    RCV-11] [-] - received file 'TESTFILE' (24 bytes, Message-ID: EAS2-20200630113452...)
INFO [    RCV-11] [61] - received data from partner 'acme_corp', generated docid: D4000000000000013@EAS2
WARN [    RCV-11] [61] - transaction is erroneous - sending operator mail
INFO [    RCV-11] [61] - received message - sending sync MDN
INFO [    RCV-11] [61] - ### END RECEIVER THREAD ###
```

The error message “message-digest attribute value does not match calculated value” recorded in the log indicates the signature is mathematically incorrect, i.e. the integrity of the data has been breached. In this case, contact the sender of the data and ask them to resend the data. If the behaviour can be reproduced with every transfer, the sender's AS2 software is obviously not adding correct signatures. If the problem cannot be reproduced, it could indicate the data was corrupted along the communication path, either due to a technical error or due to an attack.

12.1.5 Displaying log messages in the GUI

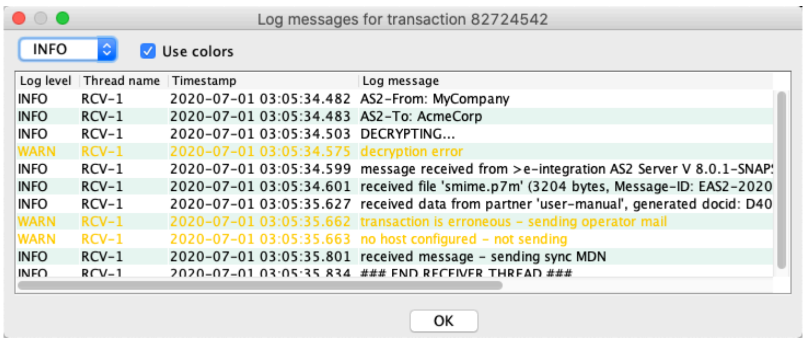
If you enabled database logging for your installation,[94], log messages will appear directly in the GUI for the configured period. The log messages can be accessed using the "LOG" button at the bottom left in the transaction details.



Description: LOCAL: receipient key mismatch

Figure 12.1. Displaying log messages

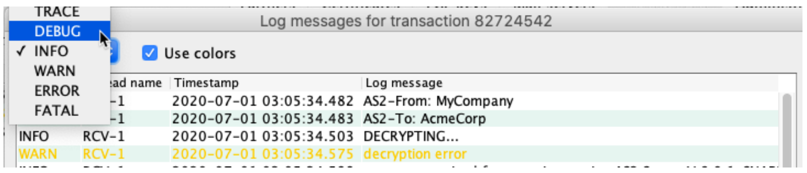
Pressing this button will open a dialogue showing all the log messages for the currently selected transaction.[95]



```
Log level  Thread name  Timestamp                Log message
INFO       RCV-1        2020-07-01 03:05:34.482  AS2-From: MyCompany
INFO       RCV-1        2020-07-01 03:05:34.483  AS2-To: AcmeCorp
INFO       RCV-1        2020-07-01 03:05:34.503  DECRYPTING...
WARN       RCV-1        2020-07-01 03:05:34.575  decryption error
INFO       RCV-1        2020-07-01 03:05:34.599  message received from >e-integration AS2 Server V 8.0.1-SNAP!
INFO       RCV-1        2020-07-01 03:05:34.601  received file 'smime.p7m' (3204 bytes, Message-ID: EAS2-2020
INFO       RCV-1        2020-07-01 03:05:35.627  received data from partner 'user-manual', generated docid: D40
WARN       RCV-1        2020-07-01 03:05:35.662  transaction is erroneous - sending operator mail
WARN       RCV-1        2020-07-01 03:05:35.663  no host configured - not sending
INFO       RCV-1        2020-07-01 03:05:35.801  received message - sending sync MDN
INFO       RCV-1        2020-07-01 03:05:35.834  ### END RECEIVER THREAD ###
```

Figure 12.2. Log messages at INFO level

By default, all messages are shown at INFO level. I.e. also warnings (level WARN) and error messages (level ERROR) can immediately be identified. Messages are colour coded according to log level. INFO messages are black, WARN messages yellow, and ERROR messages red. You can use the selection at the top left to also show DEBUG and TRACE messages. These appear in grey. Colour coding can be disabled with the respective check box.

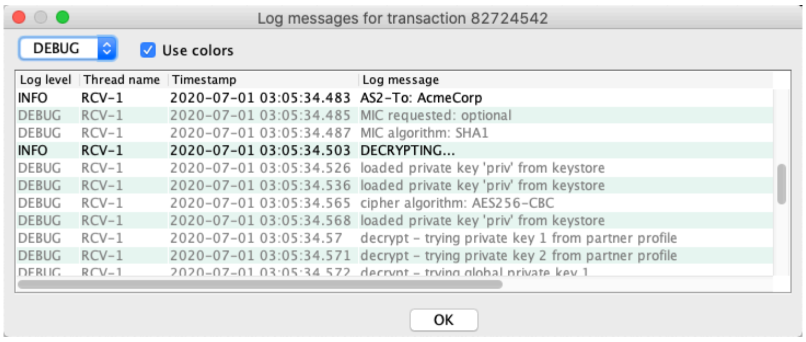


```
     ad name  Timestamp                Log message
     -1       2020-07-01 03:05:34.482  AS2-From: MyCompany
     -1       2020-07-01 03:05:34.483  AS2-To: AcmeCorp
INFO RCV-1    2020-07-01 03:05:34.503  DECRYPTING...
WARN RCV-1    2020-07-01 03:05:34.575  decryption error
```

Figure 12.3. Showing debug messages

Please note, *all* messages are always written to the database. So you can still select the log level you wish to see whilst viewing the messages for the selected transaction. The log level for logging in files still needs to be determined beforehand.

[94]see Section 10.2.15 (*Database Logging*)
[95]For technical reasons, in some rare cases it's impossible to prevent individual messages related to other transactions from also appearing. But these can then also be clearly identified as such. Generally however, at a minimum it will contains all messages related to the selected transaction.



Log level	Thread name	Timestamp	Log message
INFO	RCV-1	2020-07-01 03:05:34.483	AS2-To: AcmeCorp
DEBUG	RCV-1	2020-07-01 03:05:34.485	MIC requested: optional
DEBUG	RCV-1	2020-07-01 03:05:34.487	MIC algorithm: SHA1
INFO	RCV-1	2020-07-01 03:05:34.503	DECRYPTING...
DEBUG	RCV-1	2020-07-01 03:05:34.526	loaded private key 'priv' from keystore
DEBUG	RCV-1	2020-07-01 03:05:34.536	loaded private key 'priv' from keystore
DEBUG	RCV-1	2020-07-01 03:05:34.565	cipher algorithm: AES256-CBC
DEBUG	RCV-1	2020-07-01 03:05:34.568	loaded private key 'priv' from keystore
DEBUG	RCV-1	2020-07-01 03:05:34.57	decrypt – trying private key 1 from partner profile
DEBUG	RCV-1	2020-07-01 03:05:34.571	decrypt – trying private key 2 from partner profile
DEBUG	RCV-1	2020-07-01 03:05:34.572	decrypt – trying global private key 1

Figure 12.4. Debug level log messages

12.2 IO-Traces

In some cases it can be useful to look at what data actually went "over the wire" to debug unexpected software behavior. For this purpose e-AS2 Enterprise generates an IO-Trace for all failed sessions.

========== CHANGED (MS: 25.07.2023) ==========

These trace files are saved by default to the `tmp` directory under the installations directory with a unique name. The timestamp, the thread name and the transfer direction are part of the file name. The directory where the trace files are stored can be configured with the “iotrace.directory” property.

```
# IOTrace directory
#iotrace.directory = tmp
```

Note that trace files are only created if data has already been transferred. If the connection was never actually established for a send transaction, no trace file is written. Likewise, no trace file is written when data is received for a disabled partner.

This type of IO-Trace for failed session is referred to as “intelligent IO-Trace”. If you wish trace files to be written for all sessions regardless of the error status, disable intelligent IO-Trace in `EAS2.properties` as follows:

```
iotrace.mode.intelligent = 0
```

After restarting the e-AS2 server, at first no trace files will be written at all.

To enable IO-Trace files being generated at the current time regardless of the error status of the respective transaction, in the `tmp` directory, add a file named `_IO_` in the installation directory, written just this way and upper case, and including the underscores. The content of this file is irrelevant, only the name is important. Having this file triggers e-AS2 to generate the IO-Trace files, which are also saved to the `tmp` directory. Example (for Linux/macOS):

Enabling trace

```
$ cd tmp
$ touch _IO_
```

Listing of trace files

```
$ ls -tr1 *snd-IFTRANS*
20200430101655-532-eas2s-snd-IFTRANS-1
20200430101835-580-eas2s-snd-IFTRANS-2
```

```
20200430101855-588-eas2s-snd-IFTRANS-3
20200430121745-637-eas2s-snd-IFTRANS-1
20200430122325-822-eas2s-snd-IFTRANS-4
20200430122345-836-eas2s-snd-IFTRANS-5
20200430122445-889-eas2s-snd-IFTRANS-7
20200506013736-614-eas2s-snd-IFTRANS-1
20200506014036-690-eas2s-snd-IFTRANS-3
20200518131823-567-eas2s-snd-IFTRANS-3
20200518131903-591-eas2s-snd-IFTRANS-4
20200518132123-694-eas2s-snd-IFTRANS-5
20200519134223-433-eas2s-snd-IFTRANS-2
20200630150434-086-eas2s-snd-IFTRANS-1
```

The trace files are always assigned a thread number and separated by direction of transmission.

Viewing a trace file

```
$ cat 20200518132123-694-eas2s-snd-IFTRANS-5
POST /comm/as2 HTTP/1.1
Host: as2.acme.corp:4080
Connection: close
User-Agent: e-integration AS2 Server V 8.0.0
Date: Mon, 18 May 2020 13:21:23 +0200
AS2-Version: 1.2
EDIINT-Features: multiple-attachments, CEM
AS2-To: AcmeCorp
AS2-From: MyCompany
Content-Length: 272
MIME-Version: 1.0
Message-ID: <EAS2-20200518132123-690-000004@192.168.200.122-komsrv>
Content-Disposition: attachment; filename="DATEN.63466"
Content-type: application/octet-stream; name="DATEN.63466"...
```

At this point we'd like to point out analysing IO-Traces yourself requires good knowledge of MIME and S/MIME. However, it is possible that you may be asked by e-integration in a support case to provide IO-Traces for the affected data exchange.

Please note, the `tmp` directory is also included when old files are automatically deleted. Please also refer to Section 10.2.36 (*Cleanup routines*)again.

If you only want to analyze the data traffic from and to a specific partner, you can also activate IO-Trace for the given partner only, if you wish. To do this, append the partner ID of the given partner and another underscore to the trigger file name.

Switch IO-Trace on for partner "acme_corp"

```
$ cd tmp
$ touch _IO_acme_corp_
```

12.3 The msgdecrypt tool

Searching for the cause of problems in processing encrypted and/or signed messages can often be tedious. This was already addressed above, in Section 12.1 (*Logging*). You can use the msgdecrypt tool to confirm the conclusions drawn from analysing the log.

You will find the program in the e-AS2 Enterprise installation directory, named `msgdecrypt.sh` (for Linux/macOS) or `msgdecrypt.bat` (for MS Windows).

Sign into the computer running e-AS2 server, switch to the e-AS2 installation directory and run the tool as follows.

```
msgdecrypt.sh <IO-Trace file>
```

Following a processing error when receiving a message, the `tmp` directory will contain the associated IO-Trace. Run msgdecrypt with the name of this file as a parameter. The tool will then execute processing the same way as the e-AS2 server, but log the individual processing steps on the screen in detail. In addition, for encrypted, signed and/or compressed messages, all intermediate results will be saved to the file system to analyse separately. These intermediate result files are generated in the same directory where the IO-Trace file is.

Equipped with the protocol output on the screen and the intermediate files you can normally quickly determine the causes for problems processing messages.

The tool is called “msgdecrypt”, as it is primarily intended for analysing encrypted messages. However, it can generally be used to analyse all incoming messages, including MDNs received. The only requirement is having an IO-Trace file for the respective message.

Please note, msgdecrypt retrieves master data from the e-AS2 Server, using the client/server protocol defined for the GUI. So the e-AS2 Server must be active when using the tool. In addition, the `EAS2.properties` must contain the correct connection parameters for configuration connections. Find a usage example in the following console transcript.

```
$ ls -l tmp/20200701092759-982-eas2s-rcv-RCV-1
-rw-r--r--  1 jk  staff  3045  1 Jul 09:28 tmp/20200701092759-982-eas2s-rcv-RCV-1

$ ./msgdecrypt.sh tmp/20200701092759-982-eas2s-rcv-RCV-1

using JRE 1.8.0_152
connecting to server
loading default aliases from server
default alias 1: Optional[priv]
default alias 2: Optional[priv]
creating Decrypter
    loaded private key 'priv' from keystore
    loaded private key 'priv' from keystore
reading MimeMessage
parsing headers
    AS2-From: AcmeCorp
    AS2-To: MyCompany
    AS2-From: AcmeCorp
    AS2-To: MyCompany
loading partner profile from server
    found partner: acme_corp
creating Verifier
start processing

Content Type: application/pkcs7-mime; smime-type=enveloped-data; name=smime.p7m
S/MIME Type: enveloped-data
DECRYPTING...
    cipher algorithm: AES256-CBC
    loaded private key 'myPrivKey' from keystore
    loaded private key 'my2ndKey' from keystore
    decrypt - trying private key 1 from partner profile
    decrypting for 'E=john.doe@mycompanyltd.com,C=US,O=My Company Ltd.,CN=John Doe'

Content Type: application/pkcs7-mime
S/MIME Type: compressed-data
DECOMPRESSING...
```

```
Content Type: multipart/signed
S/MIME Type: multipart/signed
VERIFYING...
    verifying - trying 1st certificate (acme_corp)
    verifying for key alias 'acme_corp'
              serial number 1588155246439
    trying default verification mode
    signer: 1588155246439 EMAILADDRESS=info@acme.corp, C=US, O=ACME Corporation, CN=ACME
 Corporation - issuer: CN=ACME Corporation,O=ACME Corporation,C=US,E=info@acme.corp
    digest algorithm: SHA-256
    encryption algorithm: RSA
    signature verified
    BAOS
    DIGEST SHA1: py+16rNGf6Ki8N1/K+GQwmg2evM=
    BAOS
    DIGEST MD5: 8S7wD0Z9ZcvZTZqr2odqdg==

Content Type: application/octet-stream
    javax.mail.internet.MimeMessage
disconnecting from server

DONE.

$ ls -l tmp/20200701092759-982-eas2s-rcv-RCV-1*
-rw-r--r--  1 jk  staff  3045  1 Jul 09:28 tmp/20200701092759-982-eas2s-rcv-RCV-1
-rw-r--r--  1 jk  staff  3045  1 Jul 10:10 tmp/20200701092759-982-eas2s-rcv-RCV-1-0.pre
-rw-r--r--  1 jk  staff  1910  1 Jul 10:10 tmp/20200701092759-982-eas2s-rcv-RCV-1-1.decrypted
-rw-r--r--  1 jk  staff  2698  1 Jul 10:10 tmp/20200701092759-982-eas2s-rcv-RCV-1-2.decompressed
-rw-r--r--  1 jk  staff   171  1 Jul 10:10 tmp/20200701092759-982-eas2s-rcv-RCV-1-3.verified
-rw-r--r--  1 jk  staff    23  1 Jul 10:10 tmp/20200701092759-982-eas2s-rcv-RCV-1-4.bodypart
```

You can run msgdecrypt repeatedly to test the impacts of changes to the configuration of your e-AS2server.

12.4 Memory management, statistics logging

An important aspect of operating Java-based server software is the matter of memory management. On startup, the JVM is assigned an upper limited for main memory usage. The Java program running on the respective JVM entity, in our case e-AS2 Enterprise, is assigned main memory dynamically, to the extent required to create the application objects. But only until the upper limit is reached.

When the upper limit for the allocable main memory is reached, an OutOfMemoryException occurs. The program then doesn't have any option but to close immediately, which it does.

You should ensure this does not happen in production mode. For this purpose it's essential to monitor the trend of main memory usage in production mode. This is based on statistics logging. Immediately after starting the e-AS2 server and from then on every two hours, statistical information is written to the e-AS2 log file. If necessary, you can specify a different interval in `EAS2.properties`.

```
# statistics logger interval (in hours, default: 2)
#statistics.logger.interval = 2
```

Statistics logging, example

```
INFO [INFO [   STATLOG] [-] - === SERVER STATISTICS START ===
INFO [   STATLOG] [-] - server uptime: 0 days 03:21:59 hours
INFO [   STATLOG] [-] - max parallel send thread statistics:
INFO [   STATLOG] [-] -    <<global>> - 0 - 0
INFO [   STATLOG] [-] -    acme_corp - 0
INFO [   STATLOG] [-] -    acme_corp_backup - 0
INFO [   STATLOG] [-] -    mrkt_to_acme_corp - 0
INFO [   STATLOG] [-] -    sales_to_acme_corp - 0
INFO [   STATLOG] [-] -    support_to_acme_corp - 0
INFO [   STATLOG] [-] - max parallel AS2 receive threads: 0
INFO [   STATLOG] [-] - max parallel SMTP receive threads: 0
INFO [   STATLOG] [-] - maximum number of concurrent database connections: 0
INFO [   STATLOG] [-] - currently available free memory: 93 MB
INFO [   STATLOG] [-] - currently assigned total memory: 103 MB
INFO [   STATLOG] [-] - currently used memory: 10 MB
INFO [   STATLOG] [-] - maximum available memory: 910 MB
INFO [   STATLOG] [-] - total unused memory: 900 MB
INFO [   STATLOG] [-] - === SERVER STATISTICS END ===
```

Statistics logging begins with information on the send thread and receive thread parallelism. For details on this subject, please refer to Section 10.2.18 (*Transfer thread settings, send*) and Section 10.2.17 (*Transfer thread settings, incoming*). This is followed by information on concurrent database connections.

The information on memory usage is read as follows:

currently available free memory	The is the amount of main memory reserved by the JVM which is currently free.
currently assigned total memory	This is the amount of main memory JVM currently has reserved for the program running.
currently used memory	The difference between the two previous items is the amount of main memory currently being used.
maximum available memory	This is the maximum amount of main memory currently available to the JVM.
total unused memory	The difference between the two previous items is the amount of currently unused but potentially available main memory.

How much main memory e-AS2 actually requires greatly depends on the typical load. A high level of parallelism results in occasional high memory usage. Also frequently processing large files, particularly using cryptographic algorithms, may increase the amount of memory used. Main memory which is no longer needed is generally automatically released after a certain delay. This discussion focuses on how well equipped your system is for load peaks.

Exceeding the amount of main memory specified under “maximum available memory” will result in a fatal error, and subsequently the e-AS2 server spontaneously closing. In the e-AS2 environment this maximum value is derived from a setting in the `wrapper.conf` file. Here you will find the following lines:

```
# Maximum Java Heap Size (in MB)
wrapper.java.maxmemory=1024
```

The value specified under “maximum available memory” is always a little less than the value configured here, but directly depends on it. If you find this value to be unnecessary high, you can lower it in `wrapper.conf` as needed. Here you can also increase it, if it turns out to be too low. Actively monitor the statistic output in the log for a while to get a feeling for how much memory e-AS2 uses in your environment. Please note, you will need to restart e-AS2 for the changes to the `wrapper.conf` file to take effect.

During e-AS2 installation, the default value of 1024 is preset in `wrapper.conf`. However, this will not be more than half the system’s actual main memory. So if you install e-AS2 on a system with less than 2 GB RAM, the setting in `wrapper.conf` will be less, correspondingly.

Regardless of regular memory usage logging, the current values can always be retrieved via the GUI. How this works is described in Section 11.8.1 (*Manage Server*).

12.4.1 Automatic low memory checks

You can have e-AS2 assist you in regularly monitoring memory usage with the automatic low memory checks. To do so, configure the two following properties according to your needs:

```
# Check for low memory (in percent of total memory)
low.memory.warning.threshold = 25
low.memory.shutdown.threshold = 10
```

This will set the thresholds for two automatic actions built into e-AS2.

The first value specifies the warning threshold. Unless explicitly configured, the default is 25%. As soon as the amount of currently unused main memory is less than 25% of the maximum available main memory, a warning will be generated in form of an operator mail.

Check the situation and assign JVM more memory if necessary to prevent critical situations.

The second value specifies the shutdown threshold. Unless explicitly configured, the default is 10%. As soon as the amount of currently unused main memory is less than 10% of the maximum available main memory, e-AS2 will shut down safely as a precaution before a failure occurs due to low memory.

Unlike regular statistics logging, which only provides a snapshot of the current memory usage, the automatic low memory checks work continuously, i.e. immediately respond to the value being below the respective threshold.

If you don’t want to use the automatic low memory checks, set the respective properties to `0`.

12.5 Workflow testing with dry-runs

For a new partner connection you have to face three basic tasks.

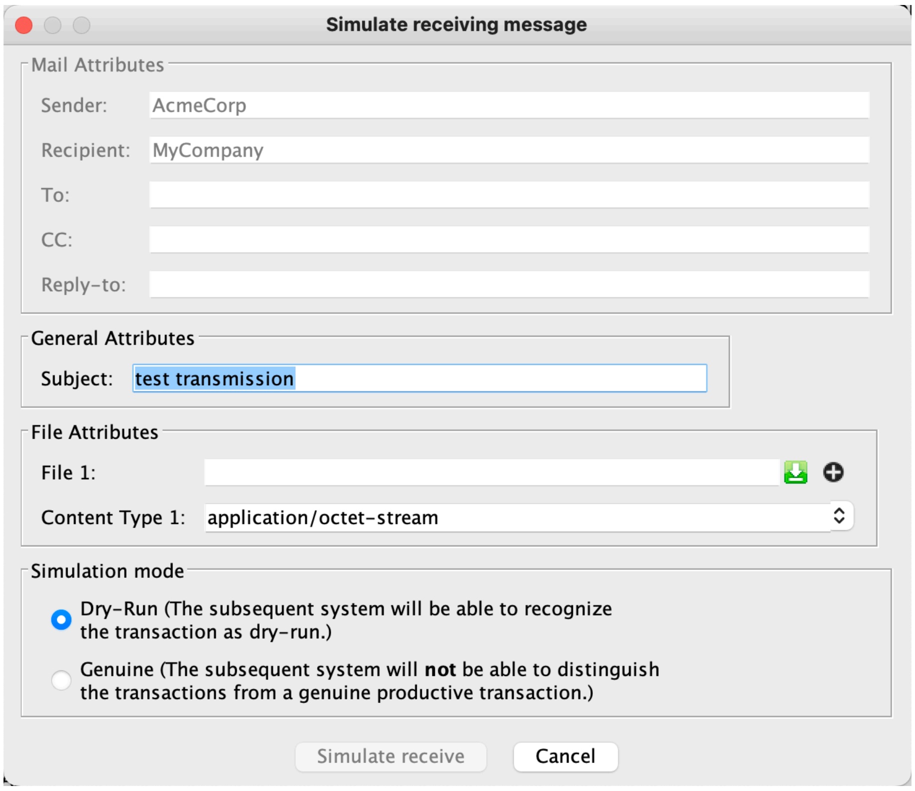
- AS2 communication to and from the partner must be configured and tested.
- Forwarding and processing of incoming data by the internal systems must be set up.
- It must be ensured, that outgoing data from the internal systems are handed over to e-AS2 in a way that the right partner profile for transmission is selected.

Often it is desirable that the workflow configuration, i.e. the interaction of e-AS2 with the internal systems, can be tested without actually transferring data to the external communication party. For the receive direction, this means that the partner does not have to repeatedly send test data during workflow configuration. For the send direction, this means that you may use completely configured partner profiles for transmission without really transferring data to the other party. So you can test autonomously and then involve the partners, after the configuration is completed on your side.

To support this approach e-AS2 offers the ability to execute dry-runs on existing partner profiles. Dry-runs for send transactions are quite different from dry-runs for receive transactions. We discuss both scenarios in more detail now.

12.5.1 Dry-runs for receive transactions

When dry-running receive transactions the receipt of a message is realistically simulated without involving the respective communications party. Open the GUI client and select a partner profile. Then click the button “Receive file (dry-run)”. The following dialog is then displayed.



Mail Attributes

Sender: AcmeCorp

Recipient: MyCompany

To:

CC:

Reply-to:

General Attributes

Subject: test transmission

File Attributes

File 1:

Content Type 1: application/octet-stream

Simulation mode

[✓ Dry-Run (The subsequent system will be able to recognize the transaction as dry-run.)]

Genuine (The subsequent system will **not** be able to distinguish the transactions from a genuine productive transaction.)

Figure 12.5. Initiating a dry-run

Click the green button to choose a test file from your desktop PC. The button labelled “Simulate receive” will then become active. Click that button to initiate the dry-run.

e-AS2 will simulate the receipt of the selected file as if it had actually been received by the respective partner. During the simulation a transaction is created. This transaction is displayed in the GUI with the addition `{DRYRUN}` so that it can be distinguished from “real” transactions. Then, the “received” data is passed to the internal processing systems in the usual way by saving it in the `in` directory.

The result file that is created contains an extra line at the end as follows:

```
DRY_RUN        [true]
```

In this way, the information that this is a dry-run, is passed to the internal processing system. The decision to enable your processing systems to evaluate this information is up to you.

The dry-run dialog is being filled with reasonable default values for various information items that are being passed to the internal systems for dry-run transactions. In general there is no need to change these values. However, if subsequent processing for incoming messages depends on specific subjects or content types, you can change them to your needs. The changed values and also the file chosen will be remembered by the e-AS2 GUI as long as it is kept running.[96]

The dry-run dialog supports simulating multifile scenarios. You can add more files by clicking the black “plus” button next to the “file open” button. In this way receive transactions with up to five files can be simulated.

[96]The mail related values in the top part of the dialog will only become active, if you have licensed the SMTP feature and select an SMTP partner profile before opening the dry-run dialog.

In the bottom part of the dialog you may choose the simulation mode. The recommended simulation mode is the dry-run mode, obviously, which gave the feature its name. However, if you decide so, you may also run receive simulation in genuine mode. This means, that the dry-run flag is not being set in the result file. The message is passed to the internal systems like real, productive transactions and will be handled as such. Use genuine mode for simulated receives with care.

12.5.2 Dry-runs for send transactions

When dry-running send transactions the goal is to avoid unnecessary data transfer to the other communications party due to internal workflow tests. Common practice among users of older e-AS2 versions was to temporarily set invalid communications parameters, in order to prevent the delivery to partners. (This has the great disadvantage that erroneous transactions occur.) With the introduction of dry-runs this is no longer needed.

The INF interface has been enhanced so that you can mark a send job as dry-run. For this purpose the following additional line must be written to the INF file:

```
DRY_RUN        [true]
```

For such jobs transactions are created, which are marked as dry-run transactions, visible in the GUI with the addition `{DRYRUN}`. For the purpose of sending the data e-AS2 will now ignore the configured communication parameters. Instead, the data is sent as an attachment in a specially designated operator mail. The transaction is then marked as successfully completed.

If desired, you can enter a separate email address for dry-runs in the partner profile, which differs from the global operator-mail address. You can find this option in the tab “mail” in the partner profile.

A Export and Import

Information on using the export/import function of e-AS2 Enterprise.

There are various situations where you may want or need to copy the entire or parts of the e-AS2 Enterprise configuration from one installed instance of the software to another. Possible use cases are:

- You're using several e-AS2 instances, e.g. a test instance and a production instance. After completing testing the respective configuration is to be copied from the test instance to the production instance without needing to re-enter it.
- You reinstall e-AS2, e.g. to use the software on a new platform (computer or operating system). You want to copy the entire configuration from the existing installation to the new installation.
- You want to create a backup of your configuration to later restore the backed up state.

An export saves the configuration data to an XML file. This contains all the information to restore the configuration objects during the import.[97] Newer versions of e-AS2 will generally also import files exported from older e-AS2 versions, but not vice versa.

A.1 Opening the Export/Import dialog

In the tab “Administration” you find the button “Export / Import”, which opens the Export/Import dialog.

For security reasons the full export/import functionality is only available when running e-AS2 Enterprise in maintenance mode (Config Only). Shut down the e-AS2 server and restart using the following command:

```
eas2s.srv console config-only
```

In Windows, start e-AS2 in Config Only mode using the respective menu item from the start menu. In this case the software will not start as a service, but will run in a console window. Closing the console window will also shut down the e-AS2 server.

With the server in maintenance mode, the Export/Import dialog will have all GUI components activated. You will then be able to use the full functionality.

If you use the export/import functionality during normal operation, the import option “Replace” will not be available.

[97]Please do not manually change the exported XML file. By changing the file there is no guarantee the file will import properly.

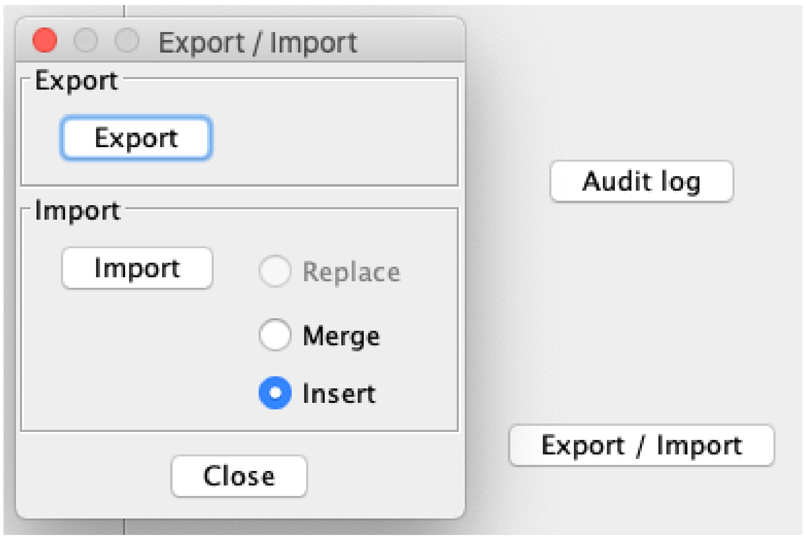


Figure A.1. The export/import dialogue

A.2 Exporting configuration data

Open the export/import dialogue and press “Export”. A Save As dialogue will appear, where you can select the target directory and enter the name for the export file. Select the file extension `.xml` or enter no file extension. The file extension will then be added automatically.

The export routine will record the entire configuration in the XML file. This specifically includes

- all partner profiles,
- all certificates in binary code,
- all PGP keys in binary code and
- all mail server configurations.

The transactions are not explicitly included in the export. When using the export/import function to copy the configuration to a different e-AS2 server, you will then start with a blank (or unchanged) list of transactions.

A.2.1 Limited export using internal tabs

You can create a limited export by using the internal tabs.

If you restrict the view of the partner profiles using inner tabs, then only the partner profiles that are still visible and all other profiles that are referenced by those partner profiles are exported. These are the certificates and mail servers under the partner profile, but also other partner profiles set as master or backup profiles.

The same in principle applies to the certificates. By using internal tabs you can ensure only public key certificates or only private key certificates are exported.

You can use limited export to specifically copy individual partner profiles to other e-AS2 instances and automatically include other related profiles.

Check the inner tabs before exporting to ensure you’re not accidentally only viewing part of the configuration.

A.3 Importing configuration data

Open the export/import dialogue and press “Export”. An Open File dialogue for selecting the import file will appear. You will generally see all XML files (file extension `.xml`) to choose from.

A validity check during import will ensure the selected file is actually an e-AS2 export and is consistent in content. Files which do not pass this check are rejected.

A.3.1 Full import, overwrite existing configuration

Select the option "Replace" to launch a full import. This will delete all existing data in e-AS2 and then import the profiles from the selected XML file. So the entire existing configuration will be replaced with a new configuration.

This option is not available if the e-AS2 server is in normal operation, i.e. was not started in maintenance mode. Be cautious using it. This process will permanently delete your existing configuration. There is no undo option after configuration import.

A.3.2 Partial import, add only

A safe import option is to only add profiles. To do so, select the option "Add". This will only copy those profiles from the selected XML file which do not exist in the existing configuration. All other profiles will be ignored and will not be imported.

Please note, this may result in a new configuration not being fully imported if there are conflicts with the profile identification.

A.3.3 Partial import, merge configuration data

A third option is to merge the existing configuration with the imported configuration. To do so, select the option "Merge". In this case, profiles which are not found in the existing configuration will be added from the XML file. The contents of existing profiles with the specific profile identification will replaced with the version from the XML file. The remaining profiles of the existing configuration will remain unchanged.

B e-AS2 Cluster Edition

Information on architecture and usage of e-AS2 Cluster Edition.

In high availability environments there is a need for increased reliability and load balancing. With e-AS2 Cluster Edition these needs are addressed. Basically e-AS2 Cluster Edition is the same as e-AS2 Enterprise but modified in a way, that several server instances are working simultaneously on the same database and transaction processing is distributed. The following picture illustrates this scenario.

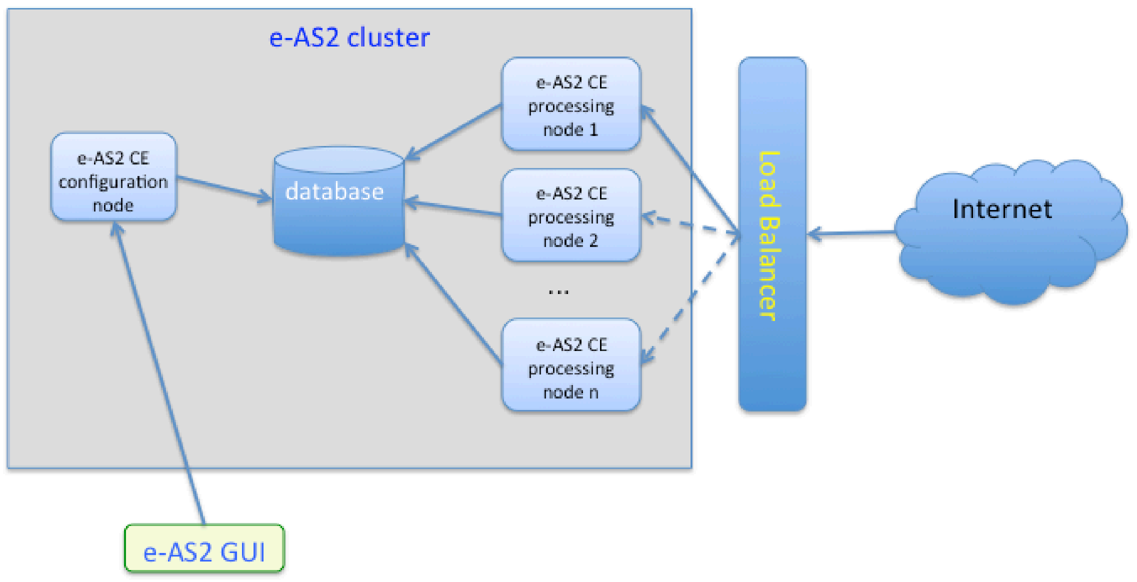

e-AS2 cluster
e-AS2 CE configuration node
database
e-AS2 CE processing node 1
e-AS2 CE processing node 2
...
e-AS2 CE processing node n
Load Balancer
Internet
e-AS2 GUI

Figure B.1. Die e-AS2 Cluster Edition

An e-AS2 cluster consists of n processing nodes and one configuration node. The GUI client connects to the configuration node. All n+1 nodes are connected to one database and share one DB schema. Therefore when using the GUI client for configuration, changes are immediately propagated to all processing nodes. Also all transaction records, that are created by processing nodes, are immediately visible in the GUI.

A load balancer must be used to distribute incoming connections to the available cluster nodes.[98] As with e-AS2 Enterprise the received files are stored in `interface/in`. For e-AS2 Cluster Edition to work correctly this has to be placed on a shared file system.

The same is true for outgoing data. File to be sent are copied to `interface/out`, which is located on a shared file system. Each file will be picked up by exactly one cluster node, which will then process the corresponding transaction. Which cluster node declares itself responsible for processing the transaction is left to chance. Due to the statistical effect on the long run transactions will be evenly distributed over all cluster nodes.

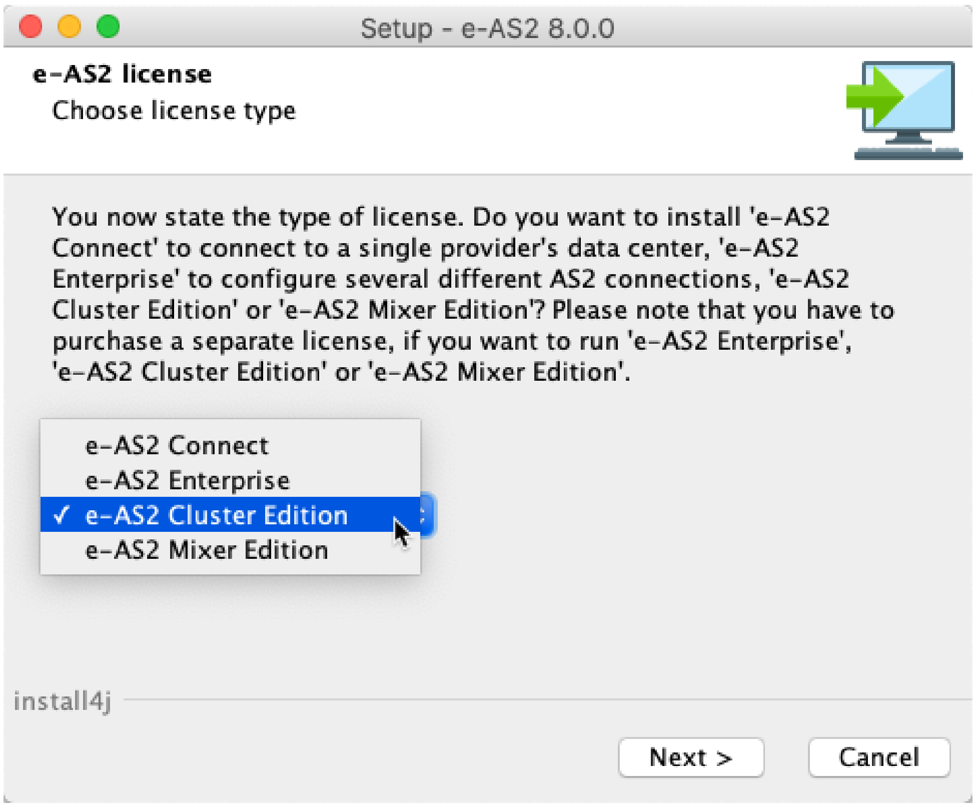
Cluster nodes can be distributed at will on several machines, as long as it is ensured, that the interface area is accessed through a shared file system. When planning a cluster, please be sure to assign a unique combination of IP address and port to every node. Furthermore we recommend to make use of dedicated health check ports, if this is supported by the load balancer. The following table shows a sample cluster plan.

[98]The load balancer is not part of e-AS2 Cluster Edition. You are free to use a product of your choice for load balancing.

Node	IP address	Port	HC port
configuration	192.168.99.10	5070	-
processing 1	192.168.99.11	5080	5090
processing 2	192.168.99.12	5080	5090
processing 3	192.168.99.13	5080	5090
processing 4	192.168.99.13	5081	5091

B.1 Installation and basic configuration e-AS2 Cluster Edition

In order to create an e-AS2 cluster you have to install one configuration node and one or more processing nodes. Every node is installed separately using the e-AS2 installer as described in the setup manual. The e-AS2 Cluster Edition cannot be operated using HyperSQL as the database. During installation choose MySQL. The e-AS2 Cluster Edition will only run with this database. You may deselect the GUI component for the processing node. Choose “e-AS2 Cluster Edition” as the license type.[99]

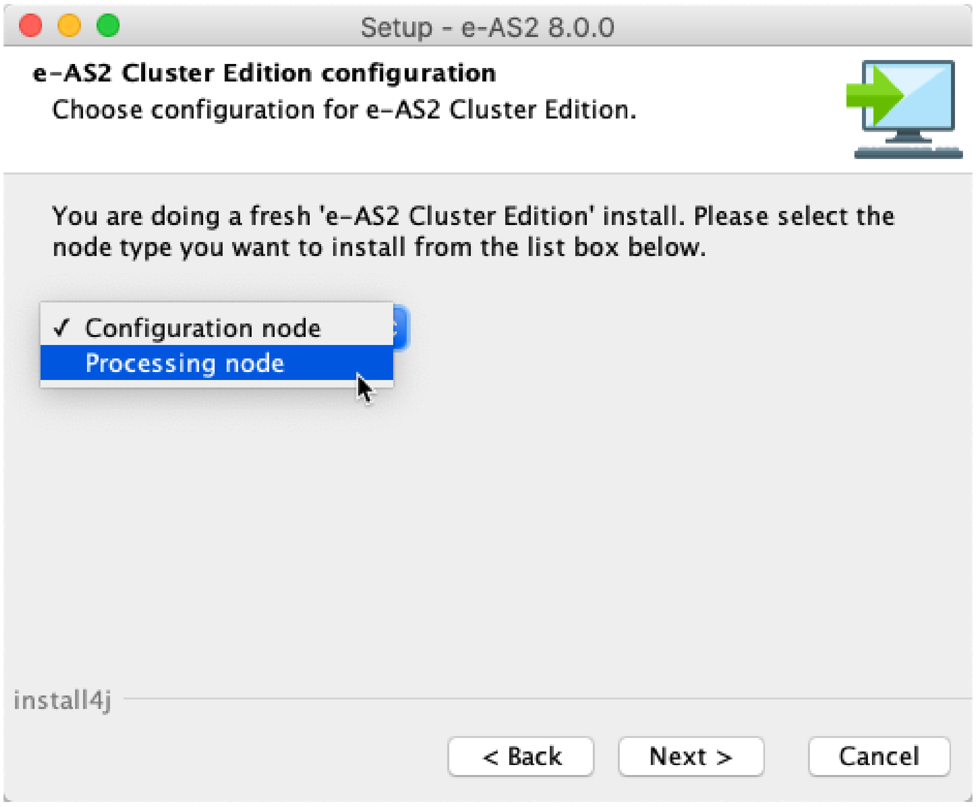


You now state the type of license. Do you want to install 'e-AS2 Connect' to connect to a single provider's data center, 'e-AS2 Enterprise' to configure several different AS2 connections, 'e-AS2 Cluster Edition' or 'e-AS2 Mixer Edition'? Please note that you have to purchase a separate license, if you want to run 'e-AS2 Enterprise', 'e-AS2 Cluster Edition' or 'e-AS2 Mixer Edition'.

Figure B.2. Installation: choose license type

This choice makes the installer ask for the requested node type. Choose “Configuration node” or “processing node” depending on the type of node you want to install.

[99]Please note that you need an appropriate license to run an e-AS2 cluster. This includes a dedicated license file that has to be copied to the installation directory. You cannot run a cluster using an Enterprise license.



e-AS2 Cluster Edition configuration

Choose configuration for e-AS2 Cluster Edition.

You are doing a fresh 'e-AS2 Cluster Edition' install. Please select the node type you want to install from the list box below.

✓ Configuration node

Processing node

Figure B.3. Installation: choose node type

Depending on the selected node type the installer will automatically add or remove certain installation components, if necessary. After that the software is installed as described in the setup manual.

As was already metioned before, e-AS2 Cluster Edition will not run on HyperSQL.

B.2 Start/stop cluster, property settings

B.2.1 Starting the cluster

You start an e-AS2 cluster by starting all cluster nodes in arbitrary order. A processing node will issue the following messages on the console, when started.

```
e-integration AS2 server starting ...
e-integration AS2 Server 8.0.0 (HTTP) ready.
e-integration AS2 Server 8.0.0 (HEALTH CHECK) ready.
e-AS2-Cluster: processing node up and running.
```

Processing nodes do not start up a CONFIG thread. Processing nodes do not accept connections from GUI clients. The HEALTH CHECK thread is optional and gets started only, if a health check port is configured that differs from the data port.

Configuration nodes do start up only the CONFIG thread.

```
e-integration AS2 Server starting ...
e-integration AS2 Server 8.0.0 (CONFIG) ready.
e-AS2-Cluster: configuration node up and running.
```

GUI clients connect to the configuration node in order to manage the cluster configuration. No transaction processing carried out in configuration nodes.

B.2.2 Stopping the cluster

The configuration node can be stopped at any time without having an impact on transaction processing. The processing nodes are dependent on an active configuration node in order to do their work.[100]

The processing nodes are all equivalent. There can be stopped in arbitrary order. As long as at least one processing node is running, the cluster is ready to process transactions. With decreasing number of nodes the throughput will get worse, though. To completely stop a cluster you have to stop all nodes.

B.2.3 Property settings

The installer will perform specific changes to the file `EAS2.properties` depending on the type of node you are installing,

Processing nodes

```
# your hostname and port / POST request URI for HTTP connection
connection.http.host = sirius.e-integration.de
connection.http.port = 5080
connection.http.uri = /as2in

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

# additional setting for Cluster Edition (optional)
connection.healthcheck.port = 5090

# Cluster Edition Node Type
# Set by installer. DO NOT CHANGE!
# This setting is irrelevant for Enterprise and Connect.
ce.node.type = processing
```

The settings for configuration connection will be deactivated. The node type will be set to “processing”.

Configuration node

```
# your hostname and port / POST request URI for HTTP connection
# connection.http.host = sirius.e-integration.de
# connection.http.port = 5080
# connection.http.uri = /as2in

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

[100]If you want to be prepared for the case that the configuration node fails, you may install additional configuration nodes. But do not start more then one configuration node at the same time.

```
connection.config.password = password

# additional setting for Cluster Edition (optional)
# connection.healthcheck.port = 5080

# Cluster Edition Node Type
# Set by installer. DO NOT CHANGE!
# This setting is irrelevant for Enterprise and Connect.
ce.node.type = config
```

The data connection settings will be deactivated. The node type will be set to “config”.

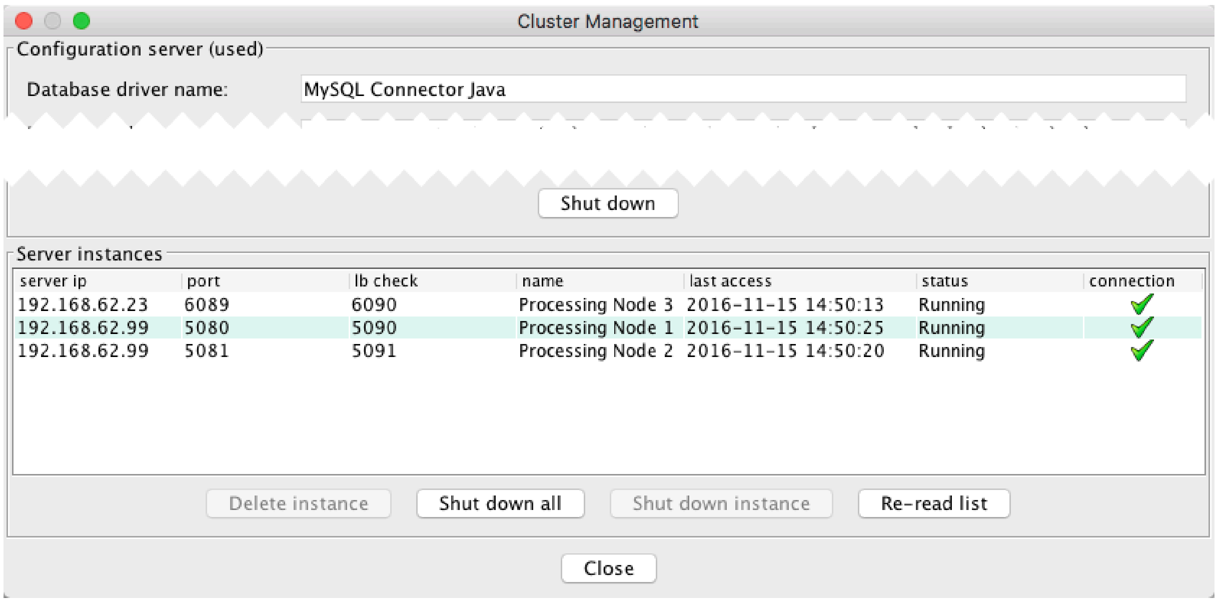
B.2.4 Cluster management using the GUI

There is no need of a special version of the GUI client for managing an e-AS2 cluster. The regular GUI client can connect to any type of server and adjusts itself automatically to present a suitable user interface. When configuring a cluster you work with partner profiles (and other configuration entities) and with the transaction list in the same way as when configuring a single instance Enterprise installation. All configuration changes will become immediately visible to all processing nodes. Also you will see all transactions of all cluster nodes in one global transaction list.

When connected to a cluster configuration node you will find the button “Manage cluster” on the tab “Administration”. This button opens the cluster management dialog. A list of all processing nodes including their current status is shown on this dialog.

From there you can shutdown single cluster nodes or the complete cluster. Select a node and hit the button “Shut down instance” to stop one single node. Press “Shut down all” to stop the cluster completely.[101].

There is no automatic refresh. Press “Re-read list” to update the status information shown in the list.



Database driver name: MySQL Connector Java

server ip	port	lb check	name	last access	status	connection
192.168.62.23	6089	6090	Processing Node 3	2016-11-15 14:50:13	Running	✓
192.168.62.99	5080	5090	Processing Node 1	2016-11-15 14:50:25	Running	✓
192.168.62.99	5081	5091	Processing Node 2	2016-11-15 14:50:20	Running	✓

Figure B.4. Cluster management

[101]Keep in mind, that it is not possible to start cluster nodes using the GUI client. Start up every node in the defined way depending on the operating system (as documented in this manual).

B.3 Shared file system

A shared file system for storing the payload files of processed transactions is an essential prerequisite for running an e-AS2 cluster.

The picture on the right shows the directory structure of an e-AS2 installation. The interface area (marked in red) should be placed on a shared file system. All other directories and files should stay on a local file system of the machine that runs the instance. This is especially true for the properties files that hold the partner-independent parts of the configuration.

Every node stores incoming files in the shared `interface/in` directory. In order to avoid naming conflicts, we extended the filename pattern for those files.

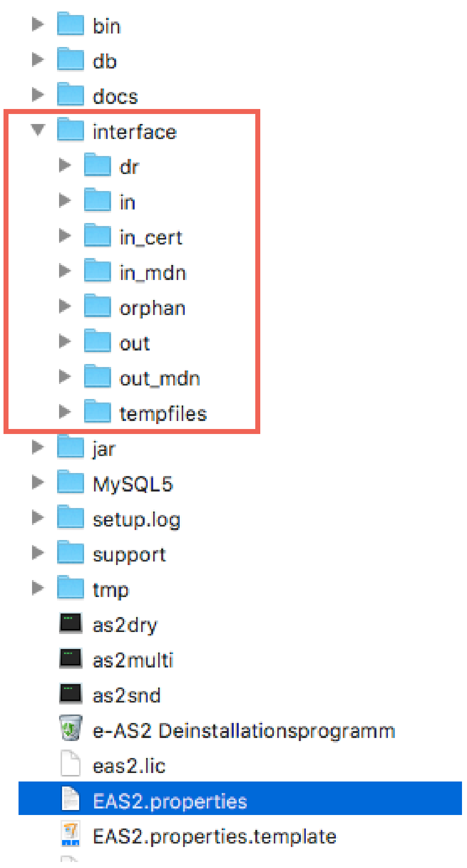


Figure B.5. Shared files system

The name of the result file is now generated as follows. Example:

```
RECV-20161114153642-156-2-000327.res
```

The general pattern is:

```
PPPP-YYYYMMDDhhmmss-xxx-i-nnnnnn.res
```

Where:

`PPPP`	is a prefix, `RECV` for regular data received which was allocated a partner profile. `ORPH` for data received which was not allocated.
`YYYYMMDD`	The current date (year, month, day).
`hhmmss-xxx`	The time received (hour, minute, second, millisecond).
`i`	The cluster node id. This id is generated and stored in the database when the node is first started. So the cluster node id remains the same for the whole lifetime of the node. The id is numerical and may consist of one or more digits depending on the number of nodes that participate in the cluster.
`nnnnnn`	A sequential six-digit number with leading zeros. (Starting with 0 and constantly incrementing. This is reset to 0 after restarting the e-AS2 server.)

Outgoing files (including their associated INF files) are to be stored in the shared `interface/out` directory. Files stored there are immediately visible to all processing nodes. The Directory Watcher threads of the processing nodes are scanning the `out` directory every 20 seconds. The 20 second time spans will generally be somewhat shifted between nodes and therefore be overlapping. So under normal circumstances only one processing node will find new files and start processing them. Over time the processing load will be statistically distributed equally on all nodes.

In very rare cases it may happen, that two nodes are starting to scan the `out` directory practically at the same time. The e-AS2 Cluster Edition nodes are prepared to handle such a situation so that double transmission of data is avoided. Even though two or more nodes are scanning the directory at the same time, every single file in the directory will always be processed by just exactly node.

B.3.1 Transfer thread settings

Transfer thread settings are configured in the local `EAS2.properties` file. They take effect in the local e-AS2 instance (processing node), So in order to get the limits for the cluster, you have to add the limits for all currently running processing nodes. This applies for both the send and the receive direction. As a result the total capacity and throughput of the cluster is increased by adding more processing nodes.

Note, that an STL setting of 1 in the partner profile does not anymore mean, that no parallel sessions will be established to that partner. Different processing nodes can be transferring in parallel to the same partner regardless of the STL setting.[102]

B.3.2 Configuration node

The configuration node has to use the shared file system in the same way as the processing nodes. This is necessary, because the configuration GUI can initiate direct transfers. You can send test data and receive the corresponding MDN. Also direct certificate exchange between e-AS2 and the AS2 system of the remote station is done by the GUI.

B.4 Updating a cluster installation

When we are talking about updating a cluster installation we have to consider two different scenarios, depending on whether the update includes a database schema upgrade or not.

B.4.1 Updating without changing the database schema

If only the software has changed, but not the database schema, then an update of the cluster is possible without downtime of transaction processing. Only the configuration of the cluster will not be possible for a short period of time.

First update all GUI installations your network. Newer e-AS2 GUIs can always (within certain limits) connect to older servers. This applies the the Cluster Edition, too.

After having updated all GUI installations perform the update of the configuration node. (If you have set up more than one configuration node for redundancy reasons, update all relevant instances.)

In the last phase, you update all the processing nodes of the cluster, one node at a time. Since all nodes are equivalent, the order does not matter. While a single node has been shut down because of the update, the processing of transactions is still ongoing because all other nodes

[102]In the very unlikely case, that the remote station does not allow simultaneous incoming transmission, temporary connection errors may occur.

are still active. After the first node updates, the cluster runs with a mixture of e-AS2 instances of different versions until all updates are completed.

B.4.2 Updating with changing the database schema

If the database schema changes in the context of a software update, the situation is a bit more complicated. In particular, an update of the cluster is no longer possible without downtime of transaction processing.

First update all GUI installations your network. Newer e-AS2 GUIs can always (within certain limits) connect to older servers. This applies the the Cluster Edition, too.

Then the complete cluster must be stopped. This means you have to stop all processing nodes and the configuration node. Start the update on the configuration node. During the update you perform the database schema upgrade as documented. After that the configuration node has to be restarted. You can then connect an e-AS2 GUI and check the configuration.

Next you begin updating the processing nodes. For each processing node you perform only the software update. The database schema upgrade is not necessary again, because it was already done during update of the configuration node. Each processing node can be restarted immediately after finishing the update. Thereby, step by step, you reach full capacity again.

B.5 Upgrading an existing Enterprise installation

If you have a productiv installation of e-AS2 Enterprise running already, it's possible to upgrade this to a cluster installation without data loss or re-entering configuration entities.

To achieve this you first have to move from HyperSQL to MySQL, because cluster operation is not supported with the HyperSQL database.

After that you have to install a license file for the Cluster Edition.

Finally you add the following lines to `EAS2.properties`:

```
# Cluster Edition Node Type
# Set by installer. DO NOT CHANGE!
# This setting is irrelevant for Enterprise and Connect.
ce.node.type = config
```

Thereby you declare, that this instance is a configuration node. After restarting e-AS2 there will be no transaction processing anymore. But you can still connect the GUI to that instance in order to add or edit configuration entities.

Then you install one or more processing nodes, that are directed to the same, existing database schema by adding the appropriate lines to `EAS2.properties`.

Finally, make sure that the interface area is shared between all nodes as described above. Re-use the existing interface area of the former Enterprise installation as a master copy, if there are still files stored that must not be lost, because they belong to open transactions. If there are no open transactions left, then the shared interface area can be freshly created as an empty directory.

B.6 REST API for Send and Receive

In addition to the INF/RES interface as documented in Chapter 8 (*The INF interface*) and Section 5.4.4 (*The result file*), a REST API is available in the Cluster Edition, via which file transmission can be requested and received data can be retrieved.

The base URL for all operations is

```
/rest
```

Assuming that e-AS2 runs on the computer “host”, the following complete base URL results:

```
http://host:5060/rest
```

This is followed by the paths to the various API functions, which are documented below. [103] You can change port and base URL `EAS2.properties`.

```
# Configuration for REST service
rest.service.port=5060
rest.service.path=rest
```

We now document the various API functions by means of exemplary curl[104] calls. Some functions require detailed information in the form of a JSON document. We show a sample document for each of them.

B.6.1 Sending data

Client call

```
curl --data @request.json \
     --header "Content-Type: application/json" \
     http://host:5060/rest/transactions/send/primary
```

The complete path of the URL contains three additional path components after the base URL:

transactions	This calls the REST API for transactions, that is, for sending and receiving data.
send	This calls the REST API for sending data.
primary	This selects the primary send channel.

Similar to the INF interface, the REST API supports two (or, if activated, even up to four) separate channels for providing data for transmission. The desired channel is selected via `primary` or `secondary` (or, if activated, `tertiary` or `quaternary`).

Important! The delivery report belonging to a transmission must be collected via the same channel via which the data was provided.

Request document, partner identification by internal ID

```
{
  "partnerId" : "int-partner-id",
  "subject" : "test data",
  "docId" : "4242",
  "requestDeliveryReport" : true,
  "dryRun" : false,
  "files" : [ {
    "data" : "ICoqKiBERU1PIFRFWFQgKioqCiAqIGUtaW50ZWdyYXRpb24gKgo=",
    "as2Name" : "demo.txt",
    "contentType" : "application/octet-stream"
  } ]
}
```

The JSON request document contains exactly the same information as an INF file stored in the file system would contain. As with the INF interface, the partner is identified by specifying the internal partner ID. Alternatively, the partner can be identified by the pair (AS2-From, AS2-To).

[103] Only HTTP connections are supported. TLS secured connections are currently not possible.
[104] s. https://curl.haxx.se/

Request document, partner identification by external IDs

```
      {
        "as2From" : "local-id",
        "as2To" : "partner-id",
        "subject" : "test data",
        "docId" : "4242",
        "requestDeliveryReport" : true,
        "dryRun" : false,
        "files" : [ {
          "data" : "ICoqKiBERU1PIFRFWFQgKioqCiAqIGUtaW50ZWdyYXRpb24gKgo=",
          "as2Name" : "demo.txt",
          "contentType" : "application/octet-stream"
        } ]
      }
```

The interface is prepared for multifile transfer. The `files` array must contain at least one entry. All files defined in this array are transmitted in an AS2 session. The file content is specified in the attribute `data` in base64-encoded form.

The attributes `as2Name` and `contentType` are optional. If `as2Name` is not specified, e-AS2 generates a unique name for this transmission. If the attribute `contentType` is missing, e-AS2 falls back to the default content type `application/octet-stream`.

The interface also supports the extended attributes when using e-AS2 to send e-mails.

Request document, extended attributes for sending e-mails

```
      {
        "as2From" : "local.address@local-domain.tld",
        "as2To" : "partner.address@partner-domain.tld",
        "carbonCopy": "cc-receiver@cc-domain.tld",
        "blindCarbonCopy": "bcc-receiver@bcc-domain.tld",
        "replyAddress": "reply-to@local-domain.tld",
        "subject" : "test data",
        "docId" : "4242",
        "requestDeliveryReport" : true,
        "dryRun" : false,
        "files" : [ {
          "data" : "ICoqKiBERU1PIFRFWFQgKioqCiAqIGUtaW50ZWdyYXRpb24gKgo=",
          "as2Name" : "demo.txt",
          "contentType" : "application/octet-stream"
        } ]
      }
```

When using the REST API for sending, no synchronous data transfer to the partner system takes place during the current session. The REST call works in such a way that the regular INF interface of e-AS2 is utilized based on the parameters specified. The payload files are stored in the `out` directory and an INF file is placed next to them. The actual processing of the transmisison job then takes place in the usual manner.

JSON attribute	Corresponding INF file key
partnerID	P_INT_ID
as2From	AS2FROM
as2To	AS2TO
carbonCopy	M_CCADR
blindCarbonCopy	M_BCCADR

JSON attribute	Corresponding INF file key
replyAddress	M_REPLY
subject	SUBJECT
docId	DOC_ID
orgDocId	ORG_DOC_ID
requestDeliveryReport	REQUEST_DR
deliveryReportAddMdnMessage	DR_ADD_MDN_MESSAGE
performErrorHandling	PERFORM_ERROR_HANDLING
dryRun	DRY_RUN
files.data	FILENAME
files.as2Name	AS2NAME
files.contentType	CONTENT_TYPE

The directory structure in the interface area has been extended to distinguish transmission jobs entered via REST call from jobs entered in the traditional way. At the top level there is now another subdirectory `cache`, under which separate directories `in`, `out` and `out_mdn` are located.

Both incoming messages and DRs are stored in the directory `cache/in`.

Important! Unlike the other directories in the interface area, the cache directories are explicitly not intended for direct access by processing systems. Access to these files is exclusively indirect via the REST API. Direct access will lead to inconsistencies and must be avoided by all means.

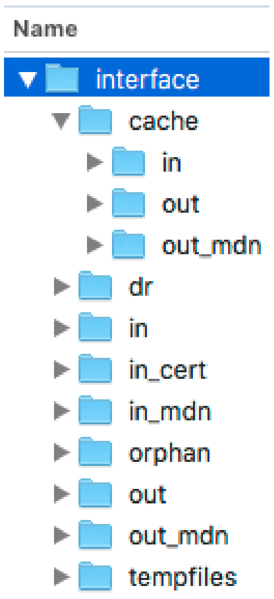


Figure B.6. Interface directory with cache

Status code	Description
204 No Content	OK.
400 Bad Request	Validation failed. The request could not be processed, for example, because mandatory attributes are missing.
416 Requested range not satisfiable	Partner profile does not exist.
500 Internal Server Error	General processing error.

In the case of an error, a JSON response is sent back to the client. This contains details on the cause of the error.

Response in case of en error, example

```
{
  "exception" : "javax.validation.ConstraintViolationException",
  "path" : "/rest/transactions/send/primary",
  "error" : "Bad Request",
  "message" : "Constraint violation(s): The TransactionSendDocument
 as2To is mandatory",
  "status" : 400,
  "timestamp" : "2017-12-12T09:14:54.366+0000"
}
```

When you enter a transmission job using the REST call, the system always checks whether the partner profile addressed by the transferred JSON request exists in the configuration. If no partner profile exists for the specified identification characteristics, the call is rejected with error 416.

This check results in a certain performance overhead, since a database query for access to the partner profile is executed for each REST call. However, it is possible to switch off the check.

Client call without partner profile check

```
curl --data @request.json \
     --header "Content-Type: application/json" \
     http://host:5060/rest/transactions/send/primary?
checkIfPartnerExists=false
```

Adding parameter `checkIfPartnerExists` with value `false` switches partner profile check off. The REST call then always succeeds, regardless of the details in the JSON request, that identify the partner profile. If the addressed partner profile does not exist, an internal e-AS2 error occurs when trying to process the transmission job. It's recommended that you only disable the partner profile check in stable environments and only if the marginal performance gain due to the omission of the check really matters.

B.6.2 Retrieving Delivery Reports

REST CLients must retrieve delivery reports at regular, self-defined intervals to ensure that the transmission job entered beforehand are processed correctly. (This is of course not necessary if no delivery report was requested with the transmission job.)

Client call

```
curl -X POST http://host:5060/rest/transactions/dr/primary
```

This REST call returns the next not yet retrieved delivery report for the specified channel.

Response document

```
{
  "transactionId" : "9504",
  "created" : "2017-12-28T10:55:46.008+0000",
  "reported" : "2017-12-28T10:55:46.436+0000",
  "partnerId" : "int-partner-id",
  "subject" : "test data",
  "documentId" : "4242",
  "status" : 2,
  "errorCode" : 0,
  "files" : [ {
    "fileName" : "Rest-SND-int-partner-
id-20171228115542-435-3-000003.dat",
    "as2Name" : "demo.txt"
  } ]
```

Response document

```
        }
```

The response document contains all the information that enables the client to link the DR to the original send transaction and to adjust its status accordingly. The central anchor point for this is of course the document ID.

Response document in case of error

```
        {
          "transactionId" : "9516",
          "created" : "2017-12-28T11:01:06.000+0000",
          "reported" : "2017-12-28T11:03:07.615+0000",
          "partnerId" : "int-partner-id",
          "subject" : "test data",
          "documentId" : "4242",
          "status" : 5,
          "errorCode" : 1,
          "description" : "manually stopped by jk at 2017-12-28 12:02:27.494
         Europe/Berlin (Central European Time)<br>was: LOCAL: file transfer
         failed - java.net.UnknownHostException: nohost.nowhere.no",
          "files" : [ {
            "fileName" : "Rest-SND-int-partner-
        id-20171228120051-860-3-000009.dat",
            "as2Name" : "demo.txt"
          } ]
        }
```

In case of an error, the `errorCode` is set to “1” and the attribute `description` contains a textual description of the error. Delivery reports after error situations are not returned to the client until the affected transaction has been stopped by the operator. Before the stop the situation is not yet final, since retries could still lead to success.

If no delivery report is available, HTTP status code 404 and a corresponding response document are returned to the client.

Response document, if no delivery reports are available

```
        {
          "exception" : "de.ei.eas2.db.TransactionJDBCRepositoryImpl
        $NoTransactionFoundException",
          "path" : "/rest/transactions/dr/primary",
          "error" : "Not Found",
          "message" : "No new delivery report(s) for primaryResponse document,
         falls keine Delivery Reports vorliegen interface.",
          "status" : 404,
          "timestamp" : "2017-12-28T11:04:39.054+0000"
        }
```

Each delivery report can be retrieved exactly once. Repeated API calls return additional delivery reports as described until none are pending. However, it's possible that problems processing the DRs on the client side occur. It may therefore be necessary to retrieve delivery reports again for individual transactions. Four further REST calls are available to take this into account.

========== CHANGED (MS: 25.07.2023) ==========

Client calls for repeated retrieval and deletion

```
        curl -X POST http://host:5060/rest/transactions/dr/primary/undeliver/9504
        curl -X GET http://host:5060/rest/transactions/dr/primary?delivered=true
        curl -X GET http://host:5060/rest/transactions/dr/primary/9504
        curl -X DELETE http://host:5060/rest/transactions/dr/primary/9504
```

To reactivate an already retrieved DR for a new retrieval, the “undeliver”-URL is used. The number at the end is the ID of the transaction to which the DR in question belongs.

========== ADDED (MS: 25.07.2023) ==========

Alternatively, you can use the “GET” call without transaction ID to get a list of all delivery reports that are already delivered but not yet deleted. With the second “GET” call you can then fetch individual delivery reports without first executing an undeliver.

Response document for list of delivery reports

```
{
  "ids" : [ 9504, 9506 ]
}
```

With the two “GET” calls you can also download “new” delivery reports. If you set the parameter “delivered” to “false” you will get a list of the ids of all not yet downloaded delivery reports. With the second “GET” call you can download them individually.

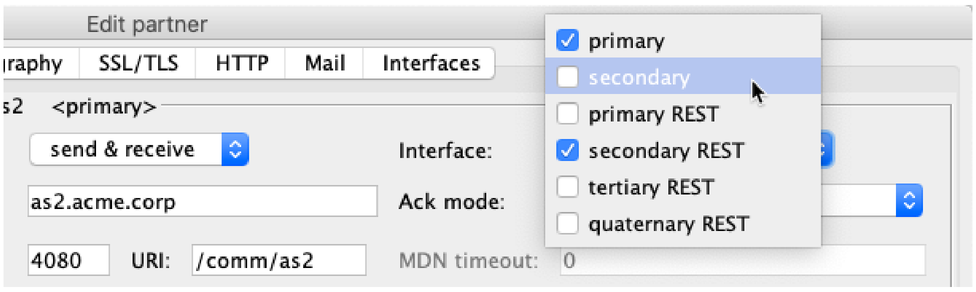
If, however, the client has successfully processed the retrieved DR, the relevant DR must be finally deleted in e-AS2 using the “DELETE” method using another REST call.

In practice, delivery report retrieval will always take the form of a pair of REST calls: Retrieve and Delete if processing is successful, or Retrieve and Undeliver if there are problems during processing.

Status-Code	Description
200 OK	OK. Next DR was returned.
204 No Content	OK. Undeliver or Delete was executed.
404 Not Found	No pending DRs. DR does not exist.
410 Gone	DR file in cache is gone.
412 Precondition Failed	Current status does not allow the desired action.

B.6.3 Retrieving received data

The REST API for receiving messages is only available if the secondary interface area[105] has been activated. In the e-AS2 Cluster Edition, up to four possible channels for forwarding received messages are then available in the partner profile. (If the tertiary and quaternary interface areas are activated, you even end up with six channels.)



send & receive

Interface:

- [✓] primary
- [ ] secondary
- [ ] primary REST
- [✓] secondary REST
- [ ] tertiary REST
- [ ] quaternary REST

as2.acme.corp

Ack mode:

4080 URI: /comm/as2

MDN timeout: 0

Figure B.7. Choosing channels for forwarding received messages

[105] s. Section 10.2.27 (*Secondary interface area*)

You can select any combination of channels. With “primary” and “secondary”, the two interface areas in the file system are still available for storing received files. Additionally, the e-AS2 Cluster Edition now offers the options “primary REST” and “secondary REST”. (And, if activated, “tertiary REST” and “quaternary REST” are available as additional options.)

Alternative display names can be defined for all six channels.

```
interface.root.primary.name = PROD channel
interface.root.secondary.name = QA channel
interface.root.primary.rest.name = (REST) - not used
interface.root.secondary.rest.name = (REST) parallel run
interface.root.tertiary.rest.name = (REST) migrate
interface.root.quaternary.rest.name = (REST) production
```

The setting is made on the server side. The self-defined display names then appear in the log files and are also used in the GUI for interface selection.

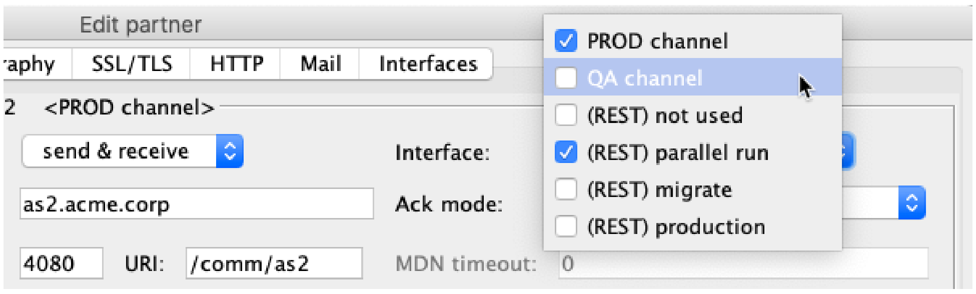


Figure B.8. Channels with self-selected display names

Note that if you select more than one channel, you will create duplicates of the received data. It is the user's responsibility to use this option only for testing purposes or to support migration projects. Productive data should always be passed on via exactly one channel. If no channel is selected, the received data is stored in the `orphan` directory.

REST clients must retrieve received messages at regular, self-defined intervals.

Client call

```
curl -X POST http://host:5060/rest/transactions/receive/primary
```

This REST call returns the next not yet retrieved received message for the specified channel.

Response document for received messages

```
{
  "instanceName" : "e-AS2 Development Server",
  "received" : "2017-12-28T11:10:26.733+0000",
  "acknowledged" : "2017-12-28T11:10:27.000+0000",
  "as2From" : "partner-id",
  "as2To" : "local-id",
  "messageId" : "EAS2-20171228121026-519-3-000018@192.168.62.99-
sirius.e-integration.de",
  "subject" : "test data",
  "certificateSerial" : "1504250180704",
  "certificateAlias" : "partner-pubkey",
  "partnerId" : "int-partner-id",
  "companyId" : "5138632",
  "transactionId" : "9534",
  "documentId" : "D4000000000000007@EAS2",
  "status" : 1,
  "errorCode" : 0,
```

Response document for received messages

```
        "mdnRequest" : false,
        "commandOnReceive" : false,
        "dryRun" : false,
        "files" : [ {
          "data" : "ICoqKiBERU1PIFRFWFQgKioqCiAqIGUtaW50ZWdyYXRpb24gKgo=",
          "as2Name" : "demo.txt",
          "contentType" : "application/octet-stream; name=demo.txt"
        } ]
      }
```

The structure of the JSON document corresponds to the contents of the RES interface.

JSON-Attribut	Correspinding RES file key
instanceName	INST_NAME
received	RECEIVED
acknowledged	ACKNOWLEDGED
as2From	AS2FROM
as2To	AS2TO
messageId	MESSAGE_ID
subject	SUBJECT
certificateSerial	CERT_SERIAL
certificateAlias	CERT_ALIAS
partnerID	P_INT_ID
companyId	CUST_NO
transactionId	TRANS_ID
documentId	DOC_ID
originalDocumentId	ORG_DOC_ID
status	STATUS
errorCode	ERRORCODE
mdnRequest	MDN_REQUEST
commandOnReceive	CMD_ON_RECV
dryRun	DRY_RUN
files.data	FILENAME
files.as2Name	AS2NAME
files.contentType	CONTENT_TYPE

The `files` array may contain multiple files. The respective file content is stored in the `data` attribute base64-encoded.

The attribute `errorCode` is always equal to “0”, since incorrectly processed reception processes are not delivered via the REST API.

Repeated calls of the REST API deliver further received messages as long as messages that have not yet been fetched are available.

Response document, if no received messages are available

```
{
  "exception" : "de.ei.eas2.db.TransactionJDBCRepositoryImpl
$NoTransactionFoundException",
  "path" : "/rest/transactions/receive/primary",
  "error" : "Not Found",
  "message" : "No new message(s) for primary interface.",
  "status" : 404,
  "timestamp" : "2017-12-28T11:18:07.058+0000"
}
```

As with the delivery reports, the client is also responsible for the subsequent cleanup of retrieved messages. Four REST calls are available for this purpose.

========== CHANGED (MS: 25.07.2023) ==========

Client calls for repeated retrieval and deletion

```
curl -X POST http://host:5060/rest/transactions/receive/primary/
undeliver/9534
curl -X GET http://host:5060/rest/transactions/receive/primary?delivered=true
curl -X GET http://host:5060/rest/transactions/receive/primary/9534
curl -X DELETE http://host:5060/rest/transactions/receive/primary/9534
```

The “undeliver”-URL is used to reactivate a message that has already been retrieved for further retrieval. The number at the end is the ID of the transaction in question.

========== ADDED (MS: 25.07.2023) ==========

Alternatively, you can use the “GET” call without transaction ID to get a list of all transactions that are already been retrieved but not yet deleted. With the second “GET” call you can then retrieve individual transactions without first executing an undeliver.

Response document for list of transactions

```
{
  "ids" : [ 9534, 9536 ]
}
```

With the two “GET” calls you can also retrieve “new” transactions. If you set the parameter “delivered” to “false” you will get a list of the ids of all not yet retrieved transactions. With the second “GET” call you can retrieve them individually.

If the client has successfully processed the retrieved message, it must be finally deleted in e-AS2 via another REST call using the “DELETE” method.

In practice, the retrieval of messages will therefore always take the form of a pair of REST calls: Retrieve and Delete if processing is successful, or Retrieve and Undeliver if there are problems during processing.

Status code	Description
200 OK	OK. Next message was delivered.
204 No Content	OK. Undeliver or Delete was executed.
404 Not Found	No more messages available for retrieval. Message does not exist.
410 Gone	Message file in cache is gone.
412 Precondition Failed	Current status does not allow the desired action.

B.6.4 Sending asynchronous MDNs

If the attribute `mdnRequest` is set to `true` in the response document for received messages, the client is responsible for creating an asynchronous MDN to be sent back to the original messsage sender. The facts are described in detail in Chapter 7 (*Interface for asynchronous MDN*). It is also possible to provide the information for setting up the asynchronous MDN using the REST API.

Client call

```
curl --data @mdn.json \
     --header "Content-Type: application/json" \
     http://host:5060/rest/transactions/mdn/primary
```

The desired content for the MDN is passed to the API in the form of a JSON file.

Request document for asynchronous MDN

```
{
  "messageId" : "EAS2-20171228112537-104-3-000001@192.168.62.99-
sirius.e-integration.de",
  "errorText" : "product-not-available",
  "mdnBody" : "Unable to add order to application system.\n\n     The
 following products are not found:\n             No. 4711\n
 No. 8199\n\n     Please review your order!\n"
}
```

This request causes the following MDN to be returned to the original sender of the data.

```
Unable to add order to application system.

     The following products are not found:
             No. 4711
             No. 8199

     Please review your order!


Reporting-UA: e-integration AS2 Server V devel
Original-Recipient: loop
Final-Recipient: loop
Original-Message-ID:
 <EAS2-20180103094135-853-0-000003@192.168.62.99-sirius.e-
integration.de>
Disposition: automatic-action/MDN-sent-automatically; processed/
error: product-not-available
Received-Content-MIC: A2ulJlq19GSoL5KK6ceckKuMNoA=, sha1
```

MDN content is controlled by three attributes in the JSON document.

JSON attribute	Description
messageId	The AS2 message ID from the received message is specified here as a reference for the association of the MDN to the transaction in question. In the response document for receipt, the message ID was found in the attribute `messageId`.
errorText	This attribute may only be used in case of an error, that is, if you want to send a negative MDN. There a short, possibly coded, textual information

JSON attribute	Description
	about the cause of the error is expected. If a positive MDN is to be generated, this attribute must be omitted completely.
mdnBody	A free, detailed text describing the facts is given here. The text should be meaningful for the sender of the data to which the MDN is sent.

Status code	Description
204 No Content	OK.
400 Bad Request	Validation failed. The request could not be processed, for example, because mandatory attributes are missing.
500 Internal Server Error	General processing error.

B.7 Generating T&T events

Processing nodes can be configured to generate so called T&T (Tracking and Tracing) events. This provides an option for tight integration with a central T&T service, that enables end users to get real time insight in what happens in the system.

When this feature is enabled, the e-AS2 server acts as a client to the T&T service, sending information about events in real time. Sending events to the service is done asynchronously and has no impact whatsoever on core e-AS2 operations.

The T&T service API is proprietary to Esker EDI Services. API documentation and reference implementation are available. Cluster Edition users interested in implementing such a service themselves, perhaps as a proxy between e-AS2 and an already existing T&T system, should contact Esker EDI Services Sales for more information.

B.7.1 Configuring the T&T handler

The T&T handler is enabled by configuring the connection to the T&T service. This is done in `EAS2.properties` with the following properties.

```
tt.service.host = ttservice.some-host.com
tt.service.port = 5051
tt.service.user = ttuser
tt.service.password = ttpassword
```

Replace the sample values with the real values that apply to your environment.

The e-AS2 server connects to the T&T service via HTTP. Basic authentication is mandatory.

On server startup e-AS2 first checks the settings. If it finds the T&T connection configured, then it starts the internal T&T handler thread. Otherwise neither the thread is started nor are T&T events generated.

On server startup e-AS2 also probes the service. If the connection does not succeed, an operator mail is sent. The T&T handler thread is started anyway, because connection problems may be temporary.

Generally T&T events are internally queued in memory. Every 15 seconds (by default) all queued events are sent to the T&T service. The pause between event transmissions can be changed with a property setting, if necessary.

```
        tt.handler.pause = 15
```

Set this to a value more suitable to your needs.

Should the T&T service be temporarily unavailable, no events are lost. Only the event queue will grow. As soon as the service is available again, all queued events will be transmitted. If the T&T service is unavailable when the e-AS2 server is shut down, then all events queued in memory will be stored on disk. When the server is started again, the stored events will be read from disk and the event queue recovered to the state it had before shutdown.

The T&T API provides a means of separating events by context. This makes it possible to apply the technology to complex, segmented environments. T&T events can be collected at one single place, but still can be displayed in separated views per application segment. In order to support this approach an application context must be defined. In e-AS2 this is done as follows.

```
        tt.context = EDI
```

Should no context be defined, then the context is internally set to “EDI”.

B.7.2 Events covered

The e-AS2 T&T event handler covers events from four categories.

Receive Events

These events are triggered, when something is received by e-AS2. This means, there is an incoming connection and data is transferred from the outside world to e-AS2. Note that this is conceptually different from the transactions “direction”. A SEND transaction can have a Receive Event. This happens when an asynchronous MDN is received.

Here is a complete list of Receive Events.

- When a message is received through an incoming connection, the first event to be generated is the *File Delivered Event*. This means that the received message was written to a file on disk. In the case of processing errors, no file is stored on disk and accordingly this event will not be generated.
- When a message is received through an incoming connection, a *Message Receive Event* is generated as the second event, as soon as the corresponding RECEIVE transaction is stored in the database. In error situations this may be the first event to be generated. The error flag is then set to `true` and the error description is added to the event as a generic attribute.
- When an MDN is received through an incoming connection, an *MDN Receive Event* is generated. The event contains all relevant information of the regarding SEND transaction, that is being updated by the MDN.

Send Events

Theses events are triggered, when something is sent by e-AS2. Note that this is conceptually different from the transactions “direction”. A RECEIVE transaction can have a Send Event. This happens when an asynchronous MDN is sent.

Here is a complete list of Send Events.

- A *File Picked Up Event* is created every time a new file is picked up for transmission to the remote site. This is the first event generated for outgoing file transmissions (SEND transactions). The event is created regardless of whether the INF interface is used or the directory interface.

If there is an original document ID available on pick-up, then it is assumed that a T&T document already exists for this transaction. The insert flag is then set to `false`. If no original document ID is present, then document IDs are generated internally and the insert flag is set to `true`.

- When a message is transferred to the remote site, a *Message Send Event* is created. Should the transfer not be successful, then this event will contain relevant error information.

  After an error, if the transfer is tried repeatedly according to a configured retry pattern, then every repetition will trigger yet another event. However, if the retry pattern makes the system enter endless mode after a number of retries, then no more events will be created until eventually either the transfer succeeds or the transaction is stopped.
- When an MDN is sent the remote site, an *MDN Send Event* is generated. The event contains all relevant information of the regarding RECEIVE transaction, to which the MDN belongs.

REST API Events

If e-AS2 is connected to the adjacent processing environment via the REST API, then three additional event types are generated.

- When an outgoing message is passed from the processing system to e-AS2, then a *REST API Send Event* is created. Note that at this point the system does not access/evaluate the corresponding partner profile. So, not all transaction details are present in this event. The full set of information will show up in the next event (the *File Picked Up Event*).
- When an outgoing MDN is passed from the processing system to e-AS2, then a *REST API Send MDN Event* is created. Note that at this point the system does not access/evaluate the regarding transaction. So, not all transaction details are present in this event. The full set of information will show up in the next event (the *File Picked Up Event*).
- When an incoming message is retrieved from e-AS2 by the processing system, then a *REST API Receive Event* is created. Basically this event contains the same information as the *Message Receive Event* that preceded it.

Operator Intervention Events

Various types of operator intervention via the e-AS2 GUI client trigger events of their own.

- When an erroneous transaction is released for re-transmission, then a *Release Transaction Event* is generated. This event contains the error information that was present in the transaction at that moment.
- When an erroneous transaction is stopped by the operator, then a *Stop Transaciton Event* is generated. This event contains the error information that was present in the transaction at that moment. It will be the final event created for the affected transaction.
- When an operator issues a re-parse message command, then a *Transaciton Re-Parse Event* is generated. If processing now succeeds, it will be followed by a *File Delivered Event* and a *Message Receive Event*, just like with regular receive processing.