Adabas for Linux, UNIX and Windows

Adabas Remote Database Access

Version 7.0

October 2022

ADABAS & NATURAL

This document applies to Adabas for Linux, UNIX and Windows Version 7.0 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: ADAOS-REMOTE-70-20220622

Table of Contents

Preface

This document provides an overview of the methods to gain remote access to Adabas.

The Adabas Remote Database Access document is organized as follows:

Entire Net-Work and Entire Net-Work Client	Describes in short the role of Entire Net-Work and Entire Net-Work Client in a remote database access scenario.
Adabas TCP/IP	New access method that provides point-to-point connections between a client application and the Adabas server.

1 About this Documentation

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

In addition, you can also access the cloud product documentation via **https://www.softwareag.cloud**. Navigate to the desired product and then, depending on your solution, go to “Developer Center”, “User Center” or “Documentation”.

Product Training

You can find helpful product training material on our Learning Portal at **https://knowledge.softwareag.com**.

Tech Community

You can collaborate with Software AG experts on our Tech Community website at **https://tech-community.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://hub.docker.com/publishers/softwareag** and discover additional Software AG resources.

Product Support

Support for Software AG products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Entire Net-Work and Entire Net-Work Client

Entire Net-Work (WCP) provides the benefit of distributed processing by allowing you to communicate with Adabas on a network-wide scope. It consists of two parts, the Entire Net-Work Client (WCL) which is a stub linked into the application interface, and the Entire Net-Work Server which is a server process running on the same computer as the database.

An Entire Net-Work Server uses the Entire Net-Work 7 e-business message protocol to access Adabas databases. Using Entire Net-Work Server, you can set up multiple Kernels that can be used to access Adabas databases.

For more information, refer to the *Entire Net-Work* documentation.

3 Adabas TCP/IP

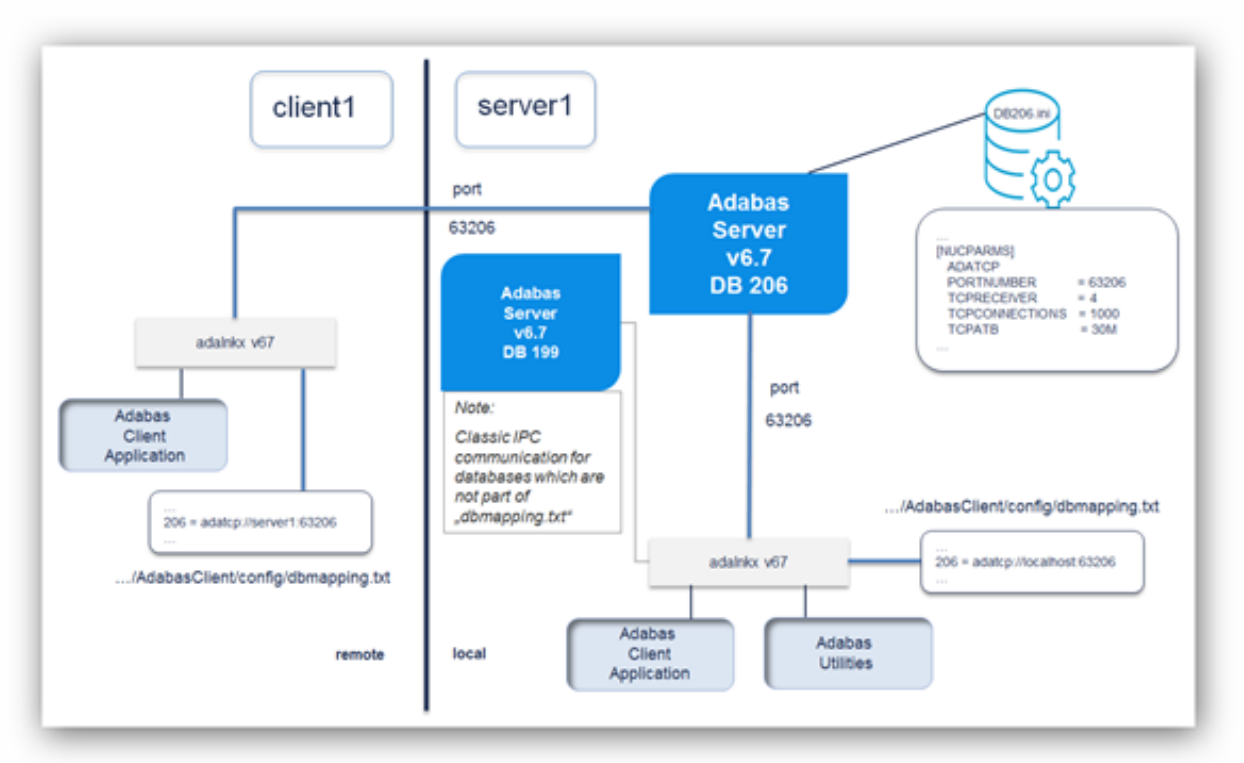
This section contains information about Adabas TCP/IP (ADATCP) and how it can be used as an access method that provides point-to-point connections between the client application and the Adabas Server.

Basic Communication

Concepts

ADATCP is the new access method that provides point-to-point connections between the client application and the Adabas Server. It is covered by the Adabas Client Interface and the Adabas Server and it is intended for remote database access, although local access via ADATCP is also possible. The currently used access method based on IPC (local or remote via WCP) is still the default method for client applications. The TCP/IP access method can be chosen by using a configuration file on the client side and by configuring the Adabas server via new nucleus parameters. The usage of the new method is transparent for the client application.

The following diagram shows an overview of typical setup:



To enable ADATCP for an Adabas database, additional nucleus parameter settings are required, like the `ADATCP` switch and `PORTNUMBER`. Classic IPC communication is still possible.

To enable ADATCP for a client application, add a connection string for the database ID to the Adabas Client (ACL) configuration file (*dbmapping.txt*).

In the example above, database 206 is configured to use ADATCP and database 199 to use classic IPC communication.

Adabas Server

With the release of Adabas Version 6.7, a new TCP/IP-based access method is provided. This new method supports the following scenarios:

- Support Adabas running in a Docker container where TCP/IP is the standard communication method between containers.
- Point-to-Point communication.
- Easy configuration.
- Reduced topology.

Therefore, the Adabas server as well as the Adabas client have been extended. The IPC based communication is still available.

The following table provides an overview of the new Adabas nucleus parameters available for ADATCP:

Parameter	Description
[NO]ADATCP	Enable the TCP/IP receiver (NOADATCP to disable it)
PORTNUMBER	The TCP/IP port the database listens on.
TCPRECEIVER	Number of receiver threads.
TCPCONNECTIONS	Maximum number of parallel connections per receiver thread.
TCPATB	Attached buffer size for each TCP/IP receiver (similar to LAB/LABX). These buffers are exclusively used by the TCP/IP receiver and are not IPC based.

The ADATCP option needs to be licensed. If this license (the Entire Net-Work license) is missing, ADATCP cannot be enabled. Please read section ***Licensing*** for a complete overview of licensing options.

To run the Community Edition of Adabas a license is not required, neither the Adabas license nor the Entire Net-Work license. However, the Community Edition restricts the ADATCP parameters to the following values:

Parameter	Value
TCPRECEIVER	1
TCPCONNECTIONS	4
TCPATB	10 MB

Examples and default values can be found in the corresponding *ADANUC* documentation.

Note:

Please ensure that the database was started using the required ADATCP parameter settings. The following initialization message is written to the Adabas nucleus logfile during a successful startup:

```
ADANUC-I-ADATCPENAB, Adatcp enabled, listening on port: ...
```

Then you may continue to change the local Adabas Client configuration file.

Adabas Client

To enable TCP/IP access for existing client applications without any change of the application, a configuration file needs to be updated with the according TCP/IP settings of the database. Please keep in mind that all Adabas utilities are also a kind of client application for the Adabas nucleus.

For each TCP/IP enabled database an entry needs to be added to the configuration file *dbmapping.txt*:

```
<dbid> = adatcp://<fqdn>:<port>
```

where `fqdn` means "fully qualified domain name". Example for database 200 which runs on host `srvada01` and listens on port 61200:

```
200 = adatcp://srvada01.softwareag.com:61200
```

For UNIX/Linux the default path and file name of the configuration file is *${ACLDIR}/config/dbmapping.txt*.

For Windows the default path and file name of the configuration file is *%ACLDIR%\..\config\dbmapping.txt*.

These entries describe the mapping of Adabas DBIDs to a connect string with the appropriate TCP/IP parameters needed on the client side. If a different mapping file should be used, the path name can be specified by the environment variable `ADA_DB_MAPPING`.

Entire Net-Work Client

If the Entire Net-Work Client has been installed, and the appropriate environment is set, the Adabas Client and the Entire Net-Work Client will use a common configuration file and a common syntax for database entries. The file will be searched in the following order:

1. Evaluate the environment variable *WCLCONFIG*.
2. Search in the directory specified by the environment variable *WCLDATADIR* for the file *xts.config*.
3. Evaluate the environment variable *ADA_DB_MAPPING*.
4. Search in the default directory for the default file name (see above).

The common syntax is the same as that known from Entire Net-Work:

```
<partition>#XTSaccess.<dbid>[0]=<protocol>://<host>:<port><?params>
```

where

- <partition> is the partition name as used by Entire Net-Work (see also the Entire Net-Work documentation: *Concepts and Facilities: Understanding Partitioning*);
- <dbid> is the database ID;
- <protocol> is the name of the ADATCP protocol: either *adatcp* or *adatcps*;

- <port> is the listen port of the database;
- <?params> is a list of optional parameters of the form: ?parameter1[&parameter2…].

Important: The database definition in the configuration file decides whether a database will be accessed via TCP or via IPC. If a database was added to the mapping file, only TCP/IP communication will be used. No other communication method (neither local IPC communication nor Entire Net-Work) is used. See also *Troubleshooting*.

Encrypted Communication (TLS/SSL)

Concepts

With the release of Adabas Version 6.7.1, an additional TCP/IP-based access method is provided, which allows encrypted communication between Adabas Server and Clients supporting the Transport Layer Security (TLS), formerly known as Secure Sockets Layer (SSL).

To use the encrypted communication functionality, the ADABAS Encryption for Linux (AEL) license is required.

Transport Layer Security (TLS) is a standard protocol used to manage the security of message transmissions in an open communications network, such as the Internet. Two types of security are provided:

- Authentication
- Encryption

TLS uses TCP/IP for its physical communications. In addition, TLS uses public and private key encryption for both authentication and data encryption keys. These keys are obtained from a certificate authority (CA).

Authentication

Using *digital signatures*, the partners in a conversation (the client and server) can be authenticated.

A digital signature is a digital code that can be attached to an electronically-transmitted message that uniquely identifies the sender. The purpose of a digital signature is to authenticate the identity of the individual sending the message using a private key to sign the message, and a public key to verify the signed message. These keys are obtained from a certificate authority of some kind, as described in

Certificate Authorities.

Encryption and Decryption

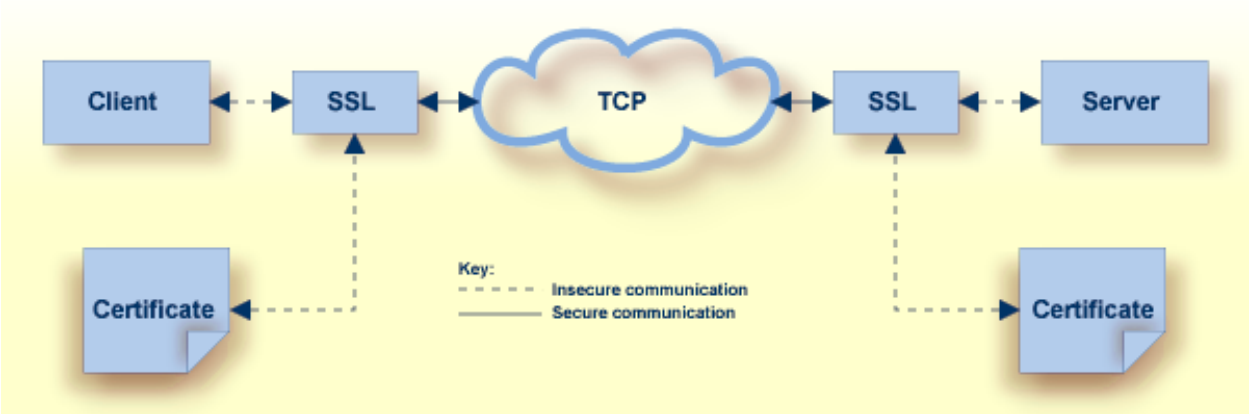
Using data *encryption* and *decryption*, messages are secured as they pass through the network.

Encryption is the conversion of data into cipher text, which cannot be easily understood without access to the encryption or decryption key. Decryption is the process of converting encrypted data back into its original form, so it can be understood. To decrypt the contents of an encrypted message, a decryption key is required. Encryption keys are generated automatically after the successful handshake between the client and server. The handshake between the client and server is handled through the use of private and public keys, which are obtained from a certificate authority of some kind, as described in *Certificate Authorities*.

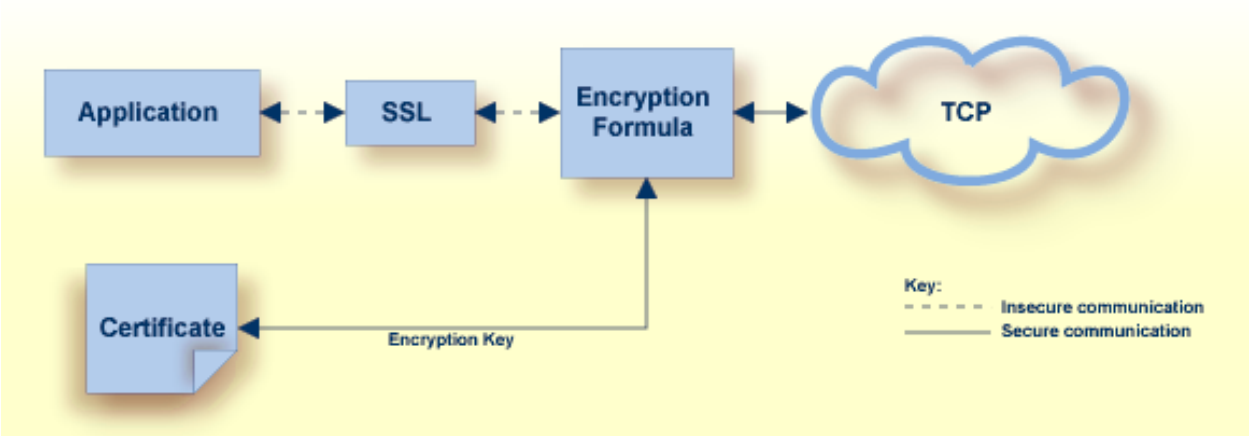
Certificate Authorities

A *certificate authority* issues and manages *certificates* for message encryption. It also verifies (authenticates) the information provided by the requestor of a digital certificate. If verification is successful, the certificate authority can then issue a certificate.

The following diagram shows how certificates are used during authentication:

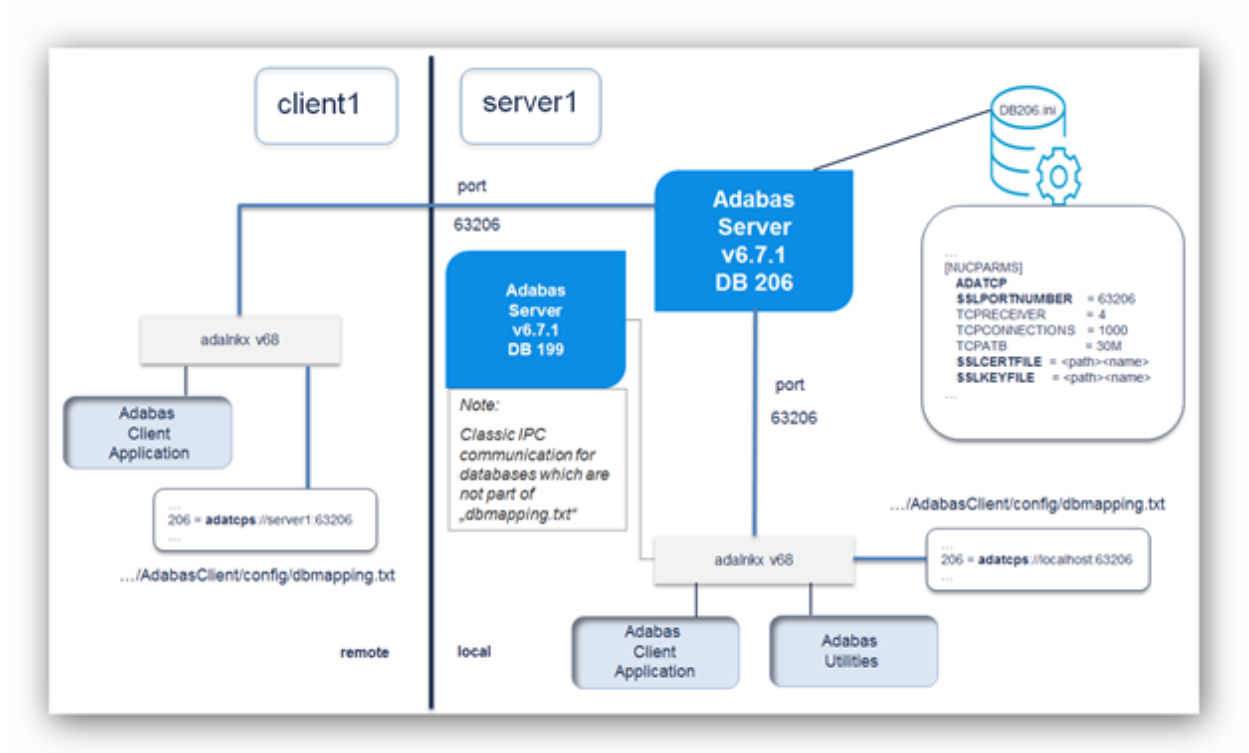


The following diagram shows how certificates are used during data encryption:



Various organizations, such as VeriSign, act as external certificate authorities for other companies and supply certificates for authentication and encryption as requested by their clients. For Adabas, you can use an external certificate authority to provide your certificates or, *for testing purposes only*, you can use the open source SSL Toolkit to become your own certificate authority.

The following diagram shows an overview of typical TLS/SSL enabled configuration:



Adabas Server

In order to use Adabas with TLS, the ADATCP must be enabled, and the SSL port number (SSLPORTNUMBER parameter) must be set. If only SSL communication is desired, the TCP/IP port number (PORTNUMBER parameter) can be set to '0' which disables the non-encrypted communication. Adabas uses version 1.1 based OpenSSL libraries, which are part of the distribution.

Any valid TLS/SSL certificate issued by a certificate authority or a self-signed certificate will be accepted. In the case of self-signed certificates, the verification level (SSLVERIFY) must be set to ‘0’. Client certificates are not validated in this case, but nevertheless the communication is still encrypted. The following table provides an overview of the Adabas nucleus parameters related to the encryption with OpenSSL:

Parameter	Description
SSLPORTNUMBER	The TCP/IP port that the database listens on for SSL encrypted communication.
SSLCERTFILE	The fully-qualified path name to the certificate file.
SSLKEYFILE	The fully-qualified path name to the private key file.
SSLCADIRECTORY	Directory name where the certificates of the Certificate Authority (CA) are stored.
SSLCAFILE	CA certificate file or certificate chain file.
SSLPASSWORD	Passphrase or file name which contains the passphrase. In the case of a file, the password is encrypted on the first access and written back to the file. The file must have write access permissions. After the encryption, change the file access permissions to read-only. Password rules: Leading or trailing white spaces (like tabs and spaces) are ignored when reading the password from the file.
SSLVERIFY	Verification level of client certificates: 0 = no verification 1-10 = maximum depth for the certificate chain verification that shall be allowed.

This feature, called Adabas Encryption for Linux (AEL), must be licensed separately and is available only on Linux. To activate the encryption, copy the AEL license file to *<installation directory>/common/conf/ael68.xml.*

Adabas Client

The Adabas Client allows encrypted connections using the keyword *adatcps* for the connection string (instead of *adatcp*) in order to support protected communications between the client and the server. The format of the secured target entry is:

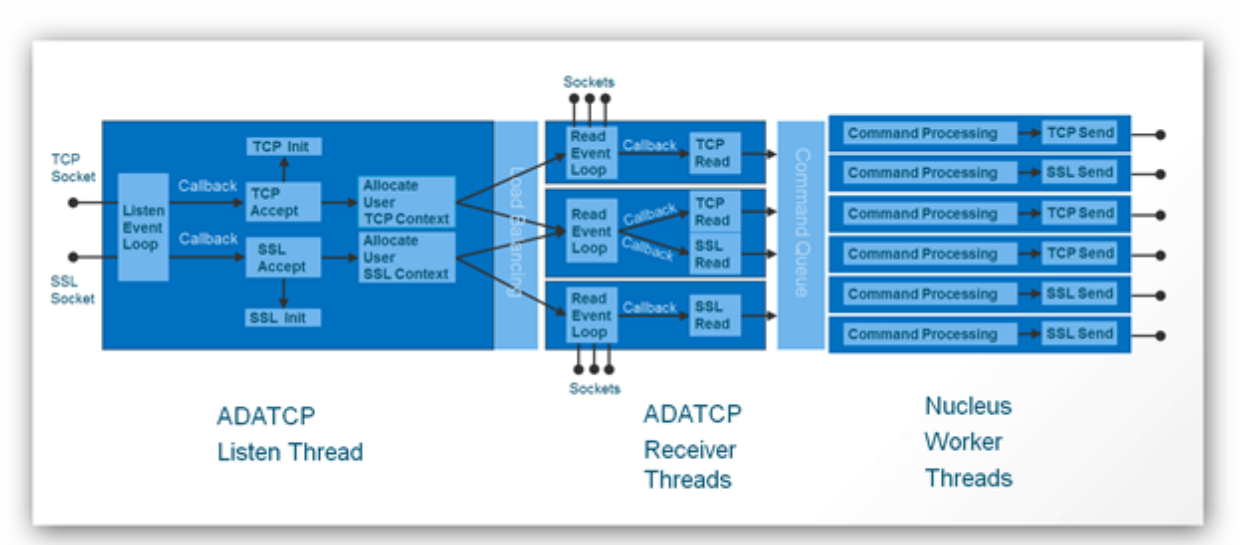
```
<dbid> = adatcps://host:port[?parm=value][&parm=value]…
```

The following table provides an overview of the Adabas Client parameters related to the encryption with OpenSSL:

Parameter	Description
CERT_FILE	The fully-qualified path name to the certificate file.
KEY_FILE	The fully-qualified path name to the private key file.
CA_PATH	Directory name where the certificates of the Certificate Authority (CA) are stored.
CA_FILE	CA certificate file or certificate chain file.
PASSWORD	Passphrase or file name which contains the passphrase. In the case of a file, the password is encrypted on the first access and written back to the file. The file must have write access permissions. After the encryption, change the file access permissions to read-only. Password rules: Leading or trailing white spaces (like tabs and spaces) are ignored when reading the password from the file.
VERIFY	Verification level of server certificates: 0 = no verification 1-10 = maximum depth for the certificate chain verification that shall be allowed.

Internal Architecture

The following diagram shows an overview of the data flow between internal components:



The major building blocks are:

- ADATCP Listen Thread
  - Listens for incoming TCP and/or SSL connections
  - Accepts and initializes or rejects connections
  - Allocates the user context
  - Distributes workload to one of the receiver threads (configurable number of instances)
- ADATCP Receiver Threads
  - Starts a ReadEventLoop per thread
  - Waits for read request of an accepted connection
  - Does a TCP- or SSL-Read, depending on the allocated user context
  - Prepares and hands over the call to the Adabas Command Queue
- Adabas Nucleus Worker Threads
  - Does the Adabas command processing
  - Does a TCP- or SSL-Send,depending on the allocated user context

The listen and receiver threads are working-event driven, which means they do not consume CPU time until the appropriate event happens.

Licensing

The point-to-point communication with Adabas on Linux, UNIX and Windows based on TCP/IP is part of the Entire Net-Work solution regardless whether the client using this communication method is local or remote to the database.

To use the encrypted communication functionality, the ADABAS Encryption for Linux (AEL) license is required.

The following table provides an overview of the different licensing possibilities regarding ADATCP:

License Available	ADATCP Activation
ADA + WCP + AEL	Enabled without restrictions
ADA + WCP	Enabled without encryption feature
ADA	Disabled
WCP	Community Edition with restrictions
None	Community Edition with restrictions

Appropriate messages are provided in the nucleus log file.

Activating the Entire Net-Work License on Linux/Unix

1 Copy the license key to a temporary location on your machine.

2 Change your directory to the *Adabas/INSTALL* directory of your main installation directory.

3 Execute the following *adalic* command to activate the license file:

```
adalic activate license_file ↩
```

Where *license_file* is the fully qualified path to your license file.

Activating the Entire Net-Work License on Windows

1 Copy the license key to a temporary location on your machine.

2 Start a Command Prompt to activate the license via the **Start** menu under: **All Programs> Start Menu group name > Administration > Adabas 6.8 > Activate License**.

3 In the Command Prompt execute the following *adalic* command to activate the license file:

```
adalic activate license_file ↩
```

Where *license_file* is the fully qualified path to your license file.

Troubleshooting

New keywords are available or output has been changed for the following utility commands:

- ADAOPR ... DISPLAY=TCPCONNECTIONS
- ADAOPR ... TCPDISCONNECT=<ID>
- ADAOPR ... DISPLAY=STATIC_PARAMETERS

The Adabas Client program `getdbinfo` has been extended to print not only Adabas version and operating system information, but also an Adabas subcode in case of problems as well as the established communication type.

Output in case of problems:

```
adabas@server1> getdbinfo 199

   get platform and version info of database 199

Database 199 .. ** Response code 148 (subcode=1034) from ADABAS for Open call


adabas@server1> getdbinfo 206

   get platform and version info of database 206

Database 206 .. ** Response code 148 (subcode=1038) from ADABAS for Open call, ↩
adatcp access to 'server1.aan.xy.sag:63206'
```

Output in case of success:

```
adabas@server1> getdbinfo 199

   get platform and version info of database 199

Database 199 is active, V6.7.0.0, Platform = 20, opsys=UNIX/Windows, classic local ↩
access


adabas@server1> getdbinfo 206

   get platform and version info of database 206
```

```
Database 206  is active, V6.7.0.0, Platform = 20, opsys=UNIX/Windows, adatcp access ↩
to 'server1.aan.xy.sag:63206'
```

Performance

Interaction of NT and TCPRECEIVER parameter:

After analyzing our heavy load test results, we recommend to maximum increase/decrease the value for "number of receiver threads" (TCPRECEIVER) by a factor of 2 in comparison to the "number of threads be established for the Adabas session" (NT).

```
NT/2 <= TCPRECEIVER <= 2*NT
```

Use Cases - FAQ

- **Q:** Is it possible to get information about the client communication type without checking the configuration file?

  A: Yes, the Adabas Client program `getdbinfo` has been extended to provide such information.
- **Q:** Different applications on one node need to be able to access the same database by either TCP or IPC. How can this been achieved?

  A: Set the required TCP parameters like `ADATCP` and `PORTNUMBER` in the *DBINI* file of your database and start it. Use an Adabas Client environment for the first application without changing the configuration file to use classic IPC communication. Use an additional Adabas Client environment for the application which should use TCP and add the necessary connection information in the configuration file or use the `ADA_DB_MAPPING` environment variable with a prepared configuration file before starting this application.
- **Q:** Customer has hundreds of clients and wants to migrate to TCP step by step. Do you have any recommendations?

  A: To avoid copying a prepared configuration file or adapt the existing *dbmapping.txt* file in the Adabas Client (ACL) *config* folder, we recommend to set the environment variable `ADA_DB_MAPPING` and use a prepared configuration file on a shared folder for all clients.
- **Q:** After trying out ADATCP, how can I switch back to the classic IPC communication method?

  A: Delete the database entry from the *dbmapping.txt* file. The database might still be ADATCP enabled, but all clients are using the classic IPC access method from now on.
- **Q:** Is it possible to use SSL communication without a client certificate and key?

A: Client certificates are not necessary if the verification level of the server (`SSLVERIFY`) is set to `0`. We do not recommended to use such a configuration in a production environment, but nevertheless the communication will be encrypted.

- **Q:** How to disallow non-encrypted connections via ADATCP?

  A: Enable the TCP/IP receiver (`ADATCP` parameter), define `SSLPORTNUMBER` and set `PORTNUMBER=0` in the *DBnnn.INI* file.

- **Q:** Is there a tool available to encrypt the `SSLPASSWORD` (server parameter) or `PASSWORD` (client parameter) value stored in a file?

  A: No, but we recommend to give your configuration a try in a test environment. After the first access, the value is encrypted and written to a file which must have write access permissions. Copy the encrypted version of the file with read-only access permissions to your production environment.

Known Restrictions

- ADATCP nucleus parameter values can neither be added nor updated while the database is up.
- ADAOPR display=tcpconnections returns incomplete data on HP-UX and SUN.