Data Compression in EntireX Broker

Data compression within EntireX Broker allows you to exchange smaller packet sizes between clients and servers. This helps to reduce response time during transmissions as well as improve the overall network throughput, especially with low-bandwidth connections.

This chapter gives an overview of data compression in EntireX Broker. It includes the following topics:

- Introduction
- zlib
- Implementation
- Sequencing Summary
- Sample Programs

See also: `COMPRESSLEVEL` under *Broker ACI Fields* | *Data Compression* for client and server | publish and subscribe.

Introduction

Compression is performed only on the `SEND` and `RECEIVE` buffers. The client or server application has the option of setting the level of compression/decompression for data transmission. The compression level can be set to achieve either no compression or a range of compression/decompression. If during a data transmission the data buffer does not compress, a logged warning message 00200450 indicates that the data has not been compressed during transmission.

Note:
The compression level is used to control compression only between the application and the Broker kernel.

zlib

zlib is a general-purpose software implementing data compression across a variety of platforms. Version 1.1.4 of zlib is implemented starting with EntireX Broker version 7. The functions used within EntireX Broker represent a subset of those available within the zlib software.

The compression algorithms are implemented through the open source software zlib.

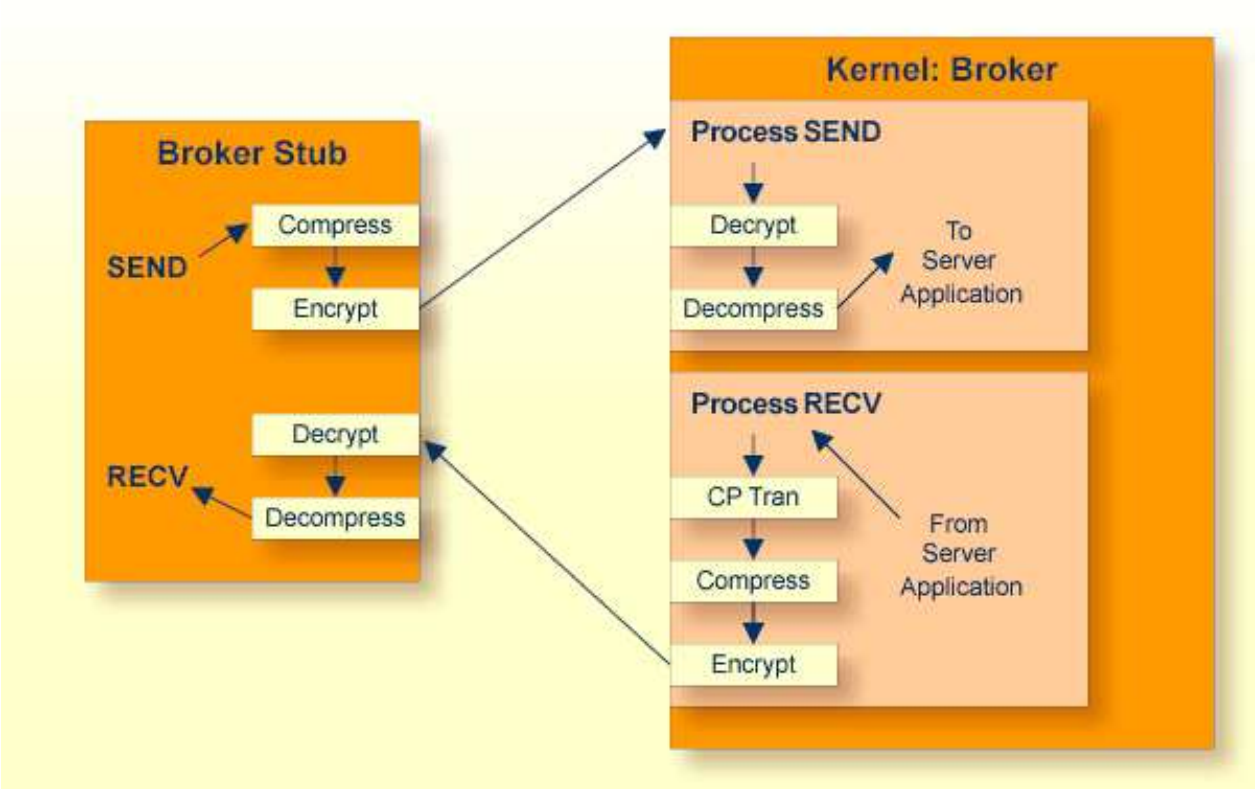
Implementation

Compression of the data is implemented by the following components of EntireX:

Components	Description
Broker control block	The Broker control block (ETBCB) contains a field that is used to set the compression level. This field determines for any SEND/RECEIVE transmission whether the data buffer will be compressed/decompressed. Possible values: **0 - 9** 0 = no compression, 9 = maximum compression/decompression **N** Default. No compression. **Y** Compression level 6 If the data buffer does not compress, the kernel or stub generates a logged warning message 00200450 indicating that the transmitted data is not compressed. **Note:** See also ACI control block field COMPRESSLEVEL.
Stubs: Broker stub and Java stub	The behavior of the Broker stub and Java stub is identical with respect to compression. The logic of a client or server application sets the compress level of the Broker control block when it issues the SEND or RECEIVE command. If the application issues a SEND, the stub compresses the data buffer before transmission of the data. If the application issues a RECEIVE, the stub decompresses the data buffer after reception of the data. **Note:** The compression level is used to control compression only between the application and the Broker kernel.
Broker kernel	When a client or server application SENDs the data to the Broker kernel, the application specifies the level at which the kernel is to decompress the data. When the client or server application issues the RECEIVE command, the Broker kernel compresses the data before returning it to the application. The application specifies the level at which the kernel is to compress the data.

Sequencing Summary

The following graphic shows the sequencing of data compression within EntireX Broker:



Sample Programs

convClt and convSrv

Sample programs `convClt` and `convSrv` in directory *examples/ACI/conversational/C* can be used as an example of performing compression/decompression. Using the `-rn` option will cause compression to be used at level <*n*>.

- `convSrv` can be instructed to use compression/decompression by specifying, for example:

  ```
  convSrv -7 -r4
  ```

  - `-r4`: This will cause a compression/decompression level of 4 to be used on all transmissions between the server and the Broker.
  - `-7`: The -7 that is needed as compression/decompression is only supported at Version 7 or above.

- `convClt` can be instructed to use compression/decompression by specifying, for example:

  ```
  convClt -7 -r2
  ```

  - `-r2`: This will cause a compression/decompression level of 2 to be used on all transmissions between the client and the Broker.
  - `-7`: The -7 that is needed as compression/decompression is only supported at Version 7 or above.

Option -g<*filename*>convClt and convSrv

To test how well various types of data will compress, you can use the option `-g<filename>`. You can use, for example, the following syntax to specify that input is to be extracted from a pre-existing file, using the two arguments from above.

```
convClt -7 -r2 -gmyfile1.txt
```

This will read in *myfile1.txt* and send it to a registered server. If `convSrv` is the server, `convSrv` will reverse the data sequence and return the data.

```
convSrv -7 -r4 -gmyfile2.txt
```

This will write in *myfile2.txt* the data sent from the client.