Data compression is used to optimize data storage and data communication. Security mechanisms are employed to protect data. A combined application of data compression techniques and security mechanisms suggests itself. In the following security mechanisms are discussed in the context of data sealing.

Data sealing is a cryptographic mechanism to ensure data integrity and to provide proof of origin i.e. to guarantee that the data originates from the 'true sender'. Data sealing may be achieved using public-key (asymmetric) cryptosystems with the capability of digital signature as does the RSA-cryptosystem. For several reasons (e.g. time problems) it is convenient to compute a short cryptographic checksum of the data called "imprint". The public-key algorithm is applied to this imprint only. The imprint is produced by hash functions, which should have certain cryptographic properties (e.g. collision free, fast and easy to compute).

The efficiency and security of data sealing depend very much on the chosen hash function. Because of possible attacks, the search for efficient and secure hash functions is an important open problem. Two methods are proposed, combining a hash function with compression algorithms.

Method A: The data are compressed and the hash function is applied to the compressed data. After the transfer the receiver computes the hash value (imprint) with the compressed data and compares it with the received hash value.

Method B: The data are compressed, but now two hash values are computed: one before and one after compressing. The receiver gets two hash values and the compressed data and then computes two hash values and compares them with the received hash values.

Using any of those methods has the advantage that the data are compressed and storage requirements and transfer time are reduced. When applying method A time is saved in comparison with the normal case when the product of compression rate and data throughput of the compression algorithm is greater than the data throughput of the hash function. Experimental investigations with two new hash functions (MD4-Hash function: Rivest, MIT, Cambridge 1990; N-Hash function: Miyaguchi et al., JWCC, Tokyo 1989), especially developed for cryptographic applications, and various compression algorithms (Huffman-, LZW-, Arithmetic-, BSTW-Coding, etc.) have been performed. The compression rate and the data throughputs of the algorithms and the hash functions were compared. Using the MD4-Hash function does not save time if combined with any compression algorithm. Using the N-Hash function saves time if combined with the LZW compression algorithm (partly also with other algorithms). Using a software implementation of the DES algorithm as hash function saves time in combination with nearly every compression algorithm. When using method B it is possible to minimize the time loss only.

Since a clear definition of security does not exist, it is only possible to state that the security increases when combining sealing and compressing of data. In principle it is more difficult to construct corrupted data maintaining a given imprint when using method A. Using method B it is even more difficult for an opponent to corrupt the data maintaining both hash values. Because the probabilities for the occurrence of a collision are independent for both hash values, the probability for a double collision is much smaller than the probability for single collision in the normal case. In addition the compression provides a simple encryption of the data, which may be optimized by using advanced methods for data compression.