

PGP

Pretty Good Privacy or PGP was written mostly by Phil Zimmermann.

He used the best available crypto algorithms as building blocks to create a system for enciphering both files and email.

It provides confidentiality and/or authentication.

It is independent of OS and machine.

It has a small number of easy-to-use commands.

It is freely available on the Internet.

Authentication is provided by SHA signed by either RSA or DSS.

Confidentiality is provided by encryption using either CAST-128, IDEA or 3DES with a one-time key generated by the sender.

PGP also provides ZIP compression, radix-64 conversion (for e-mail) and segmentation and reassembly of long messages.

The signature is generated before compression because:

(a) It is better to sign an uncompressed message so that you can store only the uncompressed message and signature for later verification. If you signed a compressed document, you would either have to store the compressed document or else re-compress it at verification time.

(b) The ZIP compression algorithm is not deterministic. Different versions of ZIP produce different compressed files. Signing after compression would require the use of just one version of ZIP.

The message is enciphered after compression because the compressed message has less redundancy, so its cryptanalysis is harder.

The random numbers for generating session keys come from the timing of the users keystrokes.

Users may have more than one set of RSA keys (to change keys or to communicate with different sets of correspondents, say). Each public key is identified by its low-order 64 bits in messages sent to the recipient.

Each user of PGP has two data structures to hold keys: one for his own public/private key pairs and one to store the public keys of other users. These data structures are called the user's private-key ring and public-key ring.

The private keys are encrypted via a passphrase. SHA produces a 160-bit hash of the passphrase and 128 of these 160 bits are used as the key for CAST-128. Private keys are indexed either by their id (= low-order 64 bits) or by a user id.

The public keys are stored in a similar data structure, but which has additional fields for a timestamp and trust information.

Suppose Alice gets a public key for Bob from a source which has been compromised by Chuck, so that the key Alice thinks is Bob's really comes from Chuck. Then Chuck could send a message to Alice signed "Bob" and Alice would accept it as coming from Bob. Furthermore, Chuck could read any encrypted message from Alice to Bob.

One way to solve this problem would use X.509. PGP uses the notion of “trust” instead.

PGP provides a way for a public key to be “signed” by another public key. It also has a level of “trust” associated to each public key. The higher the level of trust, the stronger the binding of userid and key. A key which is signed by trusted keys is also trusted to a degree determined by number and degree of trust of the trusted keys.

The degrees of trust are: undefined, unknown user, usually not trusted to sign other keys, usually trusted to sign other keys, always trusted to sign other keys, and present in the secret key ring (ultimate trust).

If a user wishes to change one of his public keys or if he believes it has been compromised, then he widely disseminates a Key Revocation Certificate, signed by the associated private key.