0.1 Disclaimer

MD5SUM has been written by Branko Lankester and Colin Plumb and has been modified by me to compile with SAS/C on the Amiga. The complete source and an smakefile are included in the distribution. This utility is public domain.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

0.2 What is MD5SUM?

MD5SUM is a small utility that allows you to build checksums over text- or binary files, using the "MD5 Message Digest Algorithm".[1] These checksums can be stored in a file and can be validated later, using MD5SUM.

For example, the checksums of the MD5SUM sources look like this:

```
107788ec46ef6ab9dbb190215f5b6f1a  source/getopt.c
ca1ada43f73cff4466adde1e88a8cf60  source/md5.c
4a6dafa2f6282430e65d4728a1303a53  source/md5sum.c
```

So, what is this useful for? Say, you're releasing a software package including sources to the public and want to make shure, nobody modifies the program (for example: add backdoors) and redistributes the modified version.[2] You could include the digest-file with a checksum of every single file in release archive and everybody could easily check the validity of the files.

0.3 How to invoke MD5SUM

If you need a short overview of the options MD5SUM provides, just call ``MD5SUM -h'' to get this help output:

```
usage: md5sum [-bv] [-c [file]] | [file...]
Generates or checks MD5 Message Digests
    -c  check message digests (default is generate)
    -v  verbose, print file names when checking
    -b  read files in binary mode
The input for -c should be the list of message digests and file names
that is printed on stdout by this program when it generates digests.
```

If you want to generate checksums, just call MD5SUM with the paths of the files you want to include in the digest file.[3] The output is printed to standard output and may be redirected using the > sign. For example:

```
MD5SUM >ram:test.md5 file1 file2 path1/file3 path2/file4
```

MD5SUM defaults to text mode and does not care about the end-of-line code, which may vary between platforms (like Amiga <–> MS-DOS).

---

[1] Placed in the public domain by RSA Data Security, Inc.

[2] Do not laugh, this already happened with BBS packages or common utilities. Once, even a virus was added to a utility!

[3] Currently, MD5SUM does not support pattern matching. You'll need a shell that does this for you or you'll have to generate a script with "List" and the `LFORMAT` keyword.

If you want to generate exact checksums for binaries, you'll have to set the `-b` flag. Files with a binary checksum are prefaced with a `*` in the digestfile:

```
ca1ada43f73cff4466adde1e88a8cf60  source/md5.c
34d0ee57bd7fa4ab699c1c3c0a522eb7 *md5sum
4a6dafa2f6282430e65d4728a1303a53  source/md5sum.c
```

Checking the validity of the files is quite easy, too. Just call MD5SUM with the `-c` option and the name of the digest file.[4] Say, we'd have saved the above example under `test.md5`, the correct commandline would be:

```
MD5SUM -c test.md5
```

If no filename is specified, MD5SUM tries to read the digest file from standard input

```
MD5SUM -c <test.md5
```

Due to this, you can pipe data to (or from) MD5SUM.

MD5SUM silently reads each file and checks the MD5 fingerprint. It only complains about files that fail the test. Example:

```
md5sum: MD5 check failed for 'md5.c'
```

If you specify the `-v` flag, MD5SUM prints the filename of the file it is currently working on. Example:

```
getopt.c       OK
md5.c          FAILED
md5sum.c       OK
md5sum: 1 of 3 file(s) failed MD5 check
```

MD5SUM is smart enough to use only the lines of a textfile, really containing MD5 fingerprints. For example, the following digest file would be interpreted without any problems:

```
----------------------------------------------------------------
This is a digest file for the MD5SUM utility. To check the
validity of all files in the distribution, please unpack the
archive, go into the "ProjectName/" directory and start "MD5SUM
-cv ‘projname.md5’".

        Sources:

107788ec46ef6ab9dbb190215f5b6f1a  getopt.c
3e51da0ccff62cf16cf60df6d3afdeb3  md5.c
4a6dafa2f6282430e65d4728a1303a53  md5sum.c

        Binaries:

34d0ee57bd7fa4ab699c1c3c0a522eb7 *md5sum

END OF FILE
```

[4] A common problem is, that the user starts MD5SUM in another current directory, than the file has been generated in and MD5SUM can't find the file under the given path. Watch this, please.

Last, but not least, I'd like to mention, that MD5SUM is pure and may be made resident.

0.4 One word about security

Mainly, there're two questions concerning the "real" security of MD5SUM. How secure is the MD5 algorithm? and How can I protect my digest file from beeing modified?

The first questions is answered by Ronald Rivest, MD5's designer:

> "It is conjectured that the difficulty of coming up with two messages having the same message digest is on the order of 2^64 operations, and that the difficulty of coming up with any message having a given message digest is on the order of 2^128 operations. The MD5 algorithm has been carefully scrutinized for weaknesses. It is, however, a relatively new algorithm and further security analysis is of course justified, as is the case with any new proposal of this sort. The level of security provided by MD5 should be sufficient for implementing very high security hybrid digital signature schemes based on MD5 and the RSA public-key cryptosystem."

The second problem is more important. Say, you have generated a digest file for every single file in your distribution and released it. But what happens if somebody modifies the source and just generates a new digest file for the new files??

To avoid this, you can, for example, post the digestfile in an appropiate newsgroup. This is nice, but far from secure! A much better way is to append your PGP (see later in text for more information on PGP) public key to the digest file and sign the complete file.

Then one could do the following to validate the package with 100% security:

- add your public key to the keyring
- check the validity of the signed digest file
- check all files using MD5SUM

For those who do not know, Pretty Good(tm) Privacy (PGP), is a high security cryptographic software application for MSDOS, Unix, VAX/VMS, Amiga and other computers. PGP allows people to exchange files or messages with privacy, authentication, and convenience. Privacy means that only those intended to receive a message can read it. Authentication means that messages that appear to be from a particular person can only have originated from that person. Convenience means that privacy and authentication are provided without the hassles of managing keys associated with conventional cryptographic software. No secure channels are needed to exchange keys between users, which makes PGP much easier to use. This is because PGP is based on a powerful new technology called "public key" cryptography.

The latest version of PGP should be available on these FTP servers:

```
Finland:    nic.funet.fi  (128.214.6.100)
            Directory: /pub/unix/security/crypt/

Italy:      ghost.dsi.unimi.it  (149.132.2.1)
            Directory: /pub/security/
```

USA: pencil.cs.missouri.edu (128.206.100.207)
Directory: /pub/crypt/

For those lacking FTP connectivity to the net, nic.funet.fi also offers the files via email.