NAME

cgdconfig − configuration utility for the cryptographic disk driver

SYNOPSIS

```
cgdconfig [−nv] cgd dev [paramsfile]
cgdconfig −C [−nv] [−f configfile]
cgdconfig −U [−nv] [−f configfile]
cgdconfig −G [−nv] [−k kgmeth] [−o outfile] paramsfile
cgdconfig −g [−nv] [−i ivmeth] [−k kgmeth] [−o outfile] alg [keylen]
cgdconfig −s [−nv] [−i ivmeth] cgd dev alg [keylen]
cgdconfig −u [−nv] cgd
```

DESCRIPTION

cgdconfig is used to configure and unconfigure cryptographic disk devices (cgds) and to maintain the configuration files that are associated with them. For more information about cryptographic disk devices see `cgd`(4).

The options are as follows:

−C Configure all the devices listed in the cgd configuration file.

−f *configfile* Specify the configuration file explicitly, rather than using the default configuration file `/etc/cgd/cgd.conf`.

−G Generate a new paramsfile (to stdout) using the values from *paramsfile* which will generate the same key. This may need to prompt for multiple passphrases.

−g Generate a paramsfile (to stdout).

−i *ivmeth* Specify the IV method (default: encblkno).

−k *kgmeth* Specify the key generation method (default: pkcs5_pbkdf2).

−o *outfile* When generating a *paramsfile*, store it in *outfile*.

−s Read the key from stdin.

−U Unconfigure all the devices listed in the cgd configuration file.

−u Unconfigure a cgd.

−V *vmeth* Specify the verification method (default: none).

−v Be verbose. May be specified multiple times.

For more information about the cryptographic algorithms and IV methods supported, please refer to `cgd`(4).

Key Generation Methods

To generate the key which it will use, **cgdconfig** evaluates all of the key generation methods in the parameters file and uses the exclusive-or of the outputs of all the methods. The methods and descriptions are as follows:

pkcs5_pbkdf2 This method requires a passphrase which is entered at configuration time. It is a salted hmac-based scheme detailed in “PKCS#5 v2.0: Password-Based Cryptography Standard”, RSA Laboratories, March 25, 1999, pages 8-10. PKCS #5 was also republished as RFC 2898.

randomkey The method simply reads `/dev/random` and uses the resulting bits as the key. It does not require a passphrase to be entered. This method is typically used to present disk devices that do not need to survive a reboot, such as the swap partition. It is also handy to facilitate overwriting the contents of a disk volume with meaningless data prior to use.

storedkey This method stores its key in the parameters file.

Verification Method

The verification method is how **cgdconfig** determines if the generated key is correct. If the newly configured disk fails to verify, then **cgdconfig** will regenerate the key and re-configure the device. It only makes sense to specify a verification method if at least of the key generation methods is error prone, e.g. uses a user-entered passphrase. The following verification methods are supported:

none	perform no verification.
disklabel	scan for a valid disklabel.
ffs	scan for a valid FFS file system.

/etc/cgd/cgd.conf

The file `/etc/cgd/cgd.conf` is used to configure **cgdconfig** if either of **-C** or **-U** are specified. Each line of the file is composed of either two or three tokens: cgd, target, and optional paramsfile.

A '#' character is interpreted as a comment and indicates that the rest of the line should be ignored. A '\' at the end of a line indicates that the next line is a continuation of the current line.

See **EXAMPLES** for an example of `/etc/cgd/cgd.conf`.

Parameters File

The Parameters File contains the required information to generate the key and configure a device. These files are typically generated by the **-g** flag and not edited by hand. When a device is configured the default parameters file is constructed by taking the basename of the target disk and prepending `/etc/cgd/` to it. E.g., if the target is `/dev/sd0h`, then the default parameters file will be `/etc/cgd/sd0h`.

It is possible to have more than one parameters file for a given disk which use different key generation methods but will generate the same key. To create a parameters file that is equivalent to an existing parameters file, use **cgdconfig** with the **-G** flag. See **EXAMPLES** for an example of this usage.

The parameters file contains a list of statements each terminated with a semi-colon. Some statements can contain statement-blocks which are either a single unadorned statement, or a brace-enclosed list of semicolon terminated statements. Three types of data are understood:

integer	a 32 bit signed integer.
string	a string.
base64	a length-encoded base64 string.

The following statements are defined:

algorithm *string*
Defines the cryptographic algorithm.

iv-method *string*
Defines the IV generation method.

keylength *integer*
Defines the length of the key.

verify_method *string*
Defines the verification method.

keygen *string statement_block*
Defines a key generation method. The *statement_block* contains statements that are specific to the key generation method.

The keygen statement's statement block may contain the following statements:

key *string*
: The key. Only used for the storedkey key generation method.

iterations *integer*
: The number of iterations. Only used for pkcs5_pbkdf2.

salt *base64*
: The salt. Only used for pkcs5_pbkdf2.

FILES

`/etc/cgd/`	configuration directory, used to store paramsfiles.
`/etc/cgd/cgd.conf`	cgd configuration file.

EXAMPLES

To set up and configure a cgd that uses AES with a 192 bit key in CBC mode with the IV Method 'encblkno' (encrypted block number):

```
# cgdconfig -g -o /etc/cgd/wd0e aes-cbc 192
# cgdconfig cgd0 /dev/wd0e
/dev/wd0e's passphrase:
```

When using verification methods, the first time that we configure the disk the verification method will fail. We overcome this by supplying **−V** *none* when we configure the first time to set up the disk. Here is the sequence of commands that is recommended:

```
# cgdconfig -g -o /etc/cgd/wd0e -V disklabel aes-cbc
# cgdconfig -V none cgd0 /dev/wd0e
/dev/wd0e's passphrase:
# disklabel -e -I cgd0
# cgdconfig -u cgd0
# cgdconfig cgd0 /dev/wd0e
/dev/wd0e's passphrase:
```

To create a new parameters file that will generate the same key as an old parameters file:

```
# cgdconfig -G -o newparamsfile oldparamsfile
old file's passphrase:
new file's passphrase:
```

To configure a cgd that uses Blowfish with a 200 bit key that it reads from stdin:

```
# cgdconfig -s cgd0 /dev/sd0h blowfish-cbc 200
```

An example parameters file which uses PKCS#5 PBKDF2:

```
algorithm aes-cbc;
iv-method encblkno;
keylength 128;
verify_method none;
keygen pkcs5_pbkdf2 {
        iterations 39361;
        salt AAAAgMoHiYonye6Kog \
              dYJAobCHE=;
};
```

An example parameters file which stores its key locally:

```
algorithm       aes-cbc;
iv-method       encblkno;
keylength       256;
verify_method   none;
keygen storedkey key AAABAK3QO6d7xzLfrXTdsgg4 \
                    ly2TdxkFqOkYYcbyUKu/f60L;
```

An example `/etc/cgd/cgd.conf`:

```
#
# /etc/cgd/cgd.conf
# Configuration file for cryptographic disk devices
#

# cgd          target         [paramsfile]
cgd0           /dev/wd0e
cgd1           /dev/sd0h      /usr/local/etc/cgd/sd0h
```

Note that this will store the parameters file as `/etc/cgd/wd0e`. And use the entered passphrase to generate the key.

SEE ALSO

`cgd`(4)

"PKCS #5 v2.0: Password-Based Cryptography Standard", RSA Laboratories, March 25, 1999.

HISTORY

The **cgdconfig** utility appeared in NetBSD 2.0.

BUGS

Since **cgdconfig** uses `getpass`(3) to read in the passphrase, it is limited to 128 characters.