Transparent Cryptographic File System for BSD. A primer

Luigi Catuogno
`[luicat@tcfs.unisa.it]`

April 15, 2000

Abstract

This document is a “quick and dirty” introduction to installation and usage of the Transparent Cryptographic File System (TCFS), for 4.4BSD systems. This guide refers to the preliminary version released for OpenBSD 2.6 and NetBSD 1.4 on April 17, 2000.

1 Introduction

TCFS is a Cryptographic filesystem developed for Linux, OpenBSD and NetBSD at the Dipartimento di Informatica ed Applicazioni of the Università degli Studi di Salerno by a research group led by Giuseppe Cattaneo, Giuseppe Persiano Luigi Catuogno and Aniello Del Sorbo since 1996.

This document refers to the BSD edition which is quite different from the Linux one, and explains how TCFS for BSD must be installed, configured and used.

TCFS for BSD provides file encryption/decryption using several ciphers (currently have been supported 3DES, RC5 and BLOWFISH), management of keys on per-user, per-group and per-process base, management of different keys for each filesystem mounted and a set of utilities for user and system administrator.

Not all of the planned features have been implemented yet, hence, in this phase the project is still open to contributions, suggestions and ideas of the developers/researchers community. Some of the features we are working on are: Directory encryption/decryption (i.e., encryption/decryption of the filenames) and full independence from underlying filesystem (at moment it is possible to mount TCFS filesystem only on top of a FFS filesystem). TCFS for BSD is still a prototype and it is released only for developers. Every other use is strongly discouraged.

2 Installation

The TCFS/BSD distribution is composed of three tar.gz archives containing the kernel part (the filesystem), the `mount_tcfs` command and the support utilities package, named respectively `XXXbsd-tcfs.tar.gz`, `XXXbsd-mount_tcfs.tar.gz`, and `XXXbsd-tcfs_utils.tar.gz`(where `XXX`∈{`net`,`open`}).

In order to install TCFS, you need to patch the kernel and the `/usr/src/sbin` subtree. The installation process is described by the following phases:
1) Download the TCFS distribution files.
2) Make sure packages `/usr/src/sys` and `/usr/src/sbin/` have been installed.
3) Expand the tar.gz archives as root

```
(cd /; tar xvfz path/to/XXXbsd-tcfs.tar.gz)
(cd /; tar xvfz path/to/XXXbsd-mount_tcfs.tar.gz)
(cd /; tar xvfz path/to/XXXbsd-tcfs_utils.tar.gz)
```

the `XXXbsd-tcfs.tar.gz` file contains some files of the original BSD distribution which have been modified. The original are renamed `namefile.orig`.
4) Add the line:

```
file-system TCFS
```

to your kernel configuration file and save it in
`/usr/src/sys/arch/yourarch/conf/YOURKERNEL`.

5) Configure, compile and install your kernel.
6) Compile the `mount_tcfs command`.

```
cd /usr/src/sbin/mount_tcfs
make install
```

7) Compile and install the utilities package.

```
cd /usr/src/tcfs-utils_0.2
make install
```

8) Add directories `/usr/tcfs/bin` and `/usr/tcfs/sbin` to your PATH.
9) Reboot the system.

3 System administration

In order to configure a TCFS filesystem, you need to edit the `/etc/fstab` and `/usr/tcfs/etc/tcfstab` files. Both these files are read by the `mount_tcfs` command, to mount and initialize the filesystem and the selected cipher.

3.1 Selecting the cipher

The `mount_tcfs` command searches the file `/usr/tcfs/etc/tcfstab` to select the cipher to be used for the new tcfs-filesystem. Each entry of this file is composed by three fields: the filesystem label, the path of the mount point and the cipher number.

The first field indicates an alias which can be used to alternatively refer the associated mount-point (indicated in the second field) by the tcfs support utilities. The third field is the cipher-id: a number which specifies the cipher that must be used to encrypt/decrypt files. Currently Triple DES, RC5 and BLOWFISH are available, and can be referred respectively as number 0, 1 and 2.

So, if you want configure the filesystem `/mnt/tcfs` (labeled “foo”) to use RC5, you must insert in the tcfstab file the row:

```
foo:/mnt/tcfs:1
```

Note that the entry labeled `default` (formed as follows) must be present.

```
default:/some/mount/point:some-cipher-mumber
```

3.2 Mounting TCFS filesystem

You can mount TCFS filesystems by running the mount command.

```
mount -t tcfs /mnt2 /mnt/tcfs
```

If you want to mount the filesystem automatically at the boot, edit the file /etc/fstab and insert a line concerning a TCFS filesystem.

```
/mnt2 /mnt/tcfs     tcfs    rw   0 0
```

3.3 Getting started

After configuring a TCFS filesystem, we test it by creating/managing some protected files. To do this, you need to push a user key into TCFS. The easiest way to do this is to run the following command:

```
myhost$ tcfsputkey -k -p /mnt/tcfs
Insert tcfs-key:                                    type an encryption key..
myhost$
```

At this point, it is possible to encrypt/decrypt files in the /mnt/tcfs subtree by using the tcfsflag utility. The following example describes a simple TCFS session.

```
myhost$ tcfsputkey -k -p /mnt/tcfs          give TCFS the encryption key
myhost$ cd /mnt/tcfs
myhost$ echo "Hello World!" > first      the file "first" is now encrypted
myhost$ tcfsflag x first               toggles first's cryptographic flag
now it is stored encrypted
myhost$ cat first                              you see the content of first
Hello World!
myhost$ tcfsflag x first               toggles first's cryptographic flag
now first is stored in clear
myhost$ cp first second                                 create a new file..
myhost$ tcfsflag x second                                  and encrypt it..
myhost$ tcfsrmkey -p /mnt/tcfs              you remove your key from tcfs
myhost$ cat first                   you see the content of first (again)
myhost$ cat second
permission denied                          second is still stored encrypted
```

The filesystem does not permit further accesses to encrypted files to any user which has not registered his key. This will happen even if the user owns the file.

3.4 Key management

Key management is an important issue in cryptographic systems, in general. It has been a TCFS design decision to split the actual cryptographic filesystem from the key management system. TCFS provides a simple interface for user applications to pass keys to the kernel. On top of this, it is possible to develop any key management scheme. We have already seen a very simple KMS: the user, before accessing a cryptographic filesystem, hands the key to TCFS by running `tcfsputkey`. No check is made on the quality of the key and the user must remeber the encryption key. Use of this KMS is deprecated.

In this section we describe another, more convenient, KMS which makes possible to generate a "good" key and store it scrambled in a key database. Keys are encrypted using the user password and stored in the `/usr/tcfs/etc/tcfspwdb` and `/usr/tcfs/etc/tcfsgpwdb` files (GDBM format databases).

Thus, TCFS users must not remember their encryption key, but can use a randomly generated key just giving their password to the `tcfsputkey` command. To do this, each user must be added to the `tcfspwdb` database by the system administrator.

tcfsadduser/tcfsrmuser

The administrator can add an entry into the tcfs key database by running `tcfsadduser`. Then users can generate their key by running tcfsgenkey. The `tcfsrmuser` command removes a user entry from the tcfs-key database.

synopsis:

```
tcfsadduser [-l user]
tcfsrmuser [-l user]
```

options:

-l specifies the user whose entry is being created/removed on the command line.

tcfsaddgroup/tcfsrmgroup

The administrator can add an entry into the group tcfs-key database by running `tcfsaddgroup`. The program asks the administrator these informations:

1. The group-id of the target group.
2. The number of components of the group.
3. The threshold (the minimum number of user's parts of key needed by the kernel to enable the group shared key).
4. The user-id of each component.

`tcfsaddgroup` generates the group shared key and all user's parts. The `tcfsrmgroup` command, removes all database entries relative to the given group-id.

synopsis:

```
        tcfsaddgroup [-g group]
        tcfsrmgroup [-g group]
```

options:
-g specifies the group whose entry is being created/removed on the command line.

4 Utilities

Interaction between user and the kernel part of TCFS is provided by updating the tcfs mount point. This interaction consists in put/remove key operation.

In order to make these operations easier, some support utilities have been designed. We have commands to put and remove users and/or group keys for any TCFS filesystem (`tcfsputkey`, `tcfsrmkey`), a command that manages cryptographic attributes (`tcfsflag`) and a set of utility that manage a database of keys (`tcfsadduser`, `tcfsaddgroup`, `tcfsrmuser`, `tcfsrmgroup`, `tcfsgenkey`). A utility to execute any user program giving it a per-process key, is also provided (`tcfsrun`).

The TCFS utilities are placed into the `/ust/tcfs/bin` and `/usr/tcfs/sbin` directories. The file `/usr/tcfs/etc/tcfstab` contains a *filesystem_label - mount-point - cipher-number* mapping. This file must contain at least the definition of the label "default". All utilities require the target tcfs filesystem. The user can specify a label instead of the path of the mount point. If the user does not indicate any filesystem, default filesystem is assumed as target.

After his registration, the user can generate his encryption key, by running the `tcfsgenkey` command.

example:

```
myhost$ tcfsgenkey
password:                                    insert the login-password
please press 10 keys:                                     ..randomly...
myhost$
```

Each time, the user want use his default key, he must just run:

```
myhost$ tcfsputkey -p /mnt/tcfs
password:                                    insert the login-password
```

If the user needs a “hand made” key (a pass-phrase), he needs run `tcfsputkey` with the **-k** option:

4.1 User/group key management

tcfsputkey/tcfsrmkey

`tcfsputkey` puts the user key into the target filesystem's key table. The key can be taken either from the key database `/usr/tcfs/etc/tcfspwb` database or from standard input. By default the user's key pushed into the kernel is taken from the key database. `tcfsrmkey` removes the user key.

synopsis:

```
tcfsputkey [-k][-f filesystem-label | -p mount-point]
tcfsputkey [-g group][-f filesystem-label | -p mount-point]
tcfsrmkey [-g group][-f filesystem-label | -p mount-point]
```

options:

-k user will be asked for a key that is put into the key table of the target filesystem. By default tcfsputkey search for an user's entry into the key database. The -k option can not be used with -g.

-f indicates the label (defined in `/usr/tcfs/etc/tcfstab`) of the target filesystem. It can not be used with -p

-p indicates the path of the mount-point of the target filesystem. If neither -k or -p are provided on the line command, `tcfsputkey` assumes the default label as target

-g pushes the user's part of key into the group key table of the target filesystem

tcfsgenkey

Generates a random key and saves it into the entry of the tcfs key database relative to the user.

synopsis:

```
tcfsgenkey
```

tcfsflag

Toggles/read tcfs file/directory attributes.

synopsis:

```
tcfsflag {g|r|x} file
```

options:

x option set/unset the user/process tcfs flag to `file`

g option set/unset the group tcfs flag to file

r reads tcfs attributes of file

tcfsrun

Creates a process, give him a process key, and let it to execute the given command line.

synopsis:

```
tcfsrun [-k][-p mount-point | -f filesystem-label] command [args...]
```

options:

-k user will be asked for a key that is put into the key table of the target filesystem. If this flag is not given, tcfsrun searches for a user's key into the key database.

-f indicates the label (defined in `/usr/tcfs/etc/tcfstab`) of the target filesystem. It can not be used with -p

-p indicates the path of the mount-point of the target filesystem. If neither -k or -p are provided on the line command, `tcfsputkey` assumes the default label as target

5 Group sharing

TCFS includes the possibility of threshold sharing files among users. Threshold sharing consists in specifying a minimum number of members (the threshold) that need to be "active" for the files owned by the group to become available. TCFS enforces the threshold sharing by generating an encryption key for each group and giving each member of the group a share using a Threshold Secret Sharing Scheme (see [1]). The group encryption key can be reconstructed by any set of at least threshold keys.

A member of the group that intends to become active does so by pushing her/his share of the group key into the kernel. The TCFS module checks if the number of shares available is above the threshold and, if it is so, it attempts to reconstruct the group encryption key. By the properties of the Threshold Secret

Sharing Scheme, it is guaranteed that, if enough shares are available, the group encryption key is correctly reconstructed.

Once the group encryption key has been reconstructed, the files owned by the group become accessible. Each time a member decides to become inactive, her share of the group encryption key is removed. The TCFS module checks if the number of shares available has gone under the threshold. In this case, the group encryption key is removed from the TCFS module and files owned by the group become unaccessible.

The current TCFS implementation of the group sharing facility requires each memeber to trust the kernel of the machine that reconstructs the key to actually remove the key once the number of active users goes below the threshold. Future implementations will remove this requirement by performing the reconstruction of the key in a distributed manner.

Setting up a TCFS group requires the following steps to be executed:

1. The superuser creates a normal unix group; editing `/etc/group` should be sufficient.

2. The superuser creates a TCFS group executing the command:

   ```
   tcfsaddgroup -g group
   ```

   This utility asks for the usernames of the members of the TCFS group. For each member a share is created and encrypted with the user passwd and then is saved in the TCFS group key database.

3. Create a directory on a TCFS filesystem owned by the group. Usual permission restrictions apply to the files in the directory. For example, the directory should be writeable if users are to create files.

4. o become active, a member of a TCFS group pushes her share into the kernel. This can be accomplished by executing the command:

   ```
   tcfsputkey -g <group>
   ```

   This utility asks for the user login passwd, decrypts the user's share from the TCFS group key database, and passes the share to the TCFS module. Shares of a group are treated by the utilities `tcfsputkey, tcfsrmkey` similarly to user's keys.

5. If a user wants to become inactive, she removes her share of the key from the TCFS module. This can be accomplished by executing the command:

   ```
   tcfsrmkey -g <group>
   ```

References

[1] A. Shamir, How to Share a Secret, Comm. ACM, v. 24, n. 11, Nov. 1979.