TCFS v.2.0 FREQUENTLY ASKED QUESTIONS WITH ANSWERS
V.1.7.8
Angelo Celentano <angcel@mikonos.dia.unisa.it>

This is a list of Frequently Asked Questions about TCFS 2.0.x, the Transparent Cryptographic File System [see Q1.1 'What is TCFS?'] developed at the Universita' di Salerno (Italy). It should be read in conjunction with the General Documentation, which is available via HTTP in

* mikonos.dia.unisa.it (192.41.218.253) : /tcfs
* edu-gw.dia.unisa.it (192.41.218.253) : /tcfs
* www.globenet.it (194.184.182.2) : /~ermmau/tcfs

For info and bug-report contact us at tcfs@mikonos.dia.unisa.it.
A mailing list is available at tcfslist@mikonos.dia.unisa.it

Index

Section 1. Introduction and General Information stand-alone system?

- Q1.8 For such a use, does TCFS have any speed advantage over CFS?
- Q1.9 Is TCFS as secure as CFS?
- Q1.10 Does it mean that root can not read my files when they are not in /dev/kmem?
- Q1.11 Which parties are assumed to be trusted by TCSF?
- Q1.12 Why do I need to trust the root of the client machine?
- Q1.13 Is TCFS commercial?

- Question 1.1. What is TCFS?
  TCFS is a Transparent Cryptographic File System that is a suitable solution to the problem of privacy for distributed file system. By a deeper integration between the encryption service and the file system, it results in a complete trasparency of use to the user applications. Files are stored in encrypted form and are decrypted before they are read. The encryption/decryption process takes place on the client machine and thus the encryption/decryption key never travels on the network.

- Question 1.2. How does TCFS works?
  TCFS operates like NFS (Network File System). We added a new flag X to make the file secure (encrypted). A TCFS filesystem can be accessed by applications using the same system calls as NFS (open,read,write,...) whence the name transparent. Blocks of data are correctly decrypted only if the right key is available to the kernel.

- Question 1.3 On which platforms does it run?
  Currently, TCFS works only on Linux Boxes (Kernel version 2.0.x). We are working to port TCFS under Digital Unix (tm) and BSD (tm).

- Question 1.4 Which are the main differences between TCFS and Matt Blaze's CFS?
  The main difference is that using CFS you have a secure directory where all files are encrypted. Before accessing encrypted files you need to attach your directory to a special mount directory (usually /crypt) and provide a key for each attached directory. TCFS instead is completely transparent to the user and can be used just like NFS, where encrypted and decrypted files can be accessed in the same way (easy, isn't it?). Actually, a user needs not be aware the his/her files are encrypted. From a more techinical point of view, CFS works in user space while TCFS works in the kernel space thus resulting in improved performances and security.

- Question 1.5 How fast (or slow) is TCFS?
  To reduce the influence of external factors, such as network traffic and physical disk access time, we run the test in the following framework:
  - The client machine running TCFS is an Intel 80486 100 Mhz processor running Linux.

- The server is a fast file server SPARCstation10 running Linux, with a 2 Gb fast SCSI disk.
We run the test on a 10Mbits Ethernet Network. All the values reported in the table are average values over ten different runs, and the execution of all tests are mixed to avoid caching side effect. The times are expressed in seconds.

Filesystem	10Kb	100Kb	1Mb	10Mb
----- Time To Write -----				
NFS	0.5	0.14	1.74	12.58
TCFS no DES	0.5	0.19	1.69	17.74
TCFS with DES	0.6	0.46	3.77	40.08
CFS	0.15	0.97	10.64	102.17
----- Time To Read -----				
NFS	0.09	0.16	1.14	10.97
TCFS no DES	0.10	0.16	1.30	12.25
TCFS with DES	0.11	0.31	2.71	26.40
CFS	0.21	1.13	13.32	121.98
TCFS DES engine	0.3	0.25	2.38	23.82

The main reason we have a better performance than CFS is due to a ‘‘double-mount’’ mechanism employed by CFS.

- Question 1.6. What about TCFS v.1.0?
The first release of TCFS can be found via HTTP at mikonos.dia.unisa.it:/tcfs but that is an INSTABLE release; it’s just an EXPERIMENTAL product. It runs in user space and is quite similar to CFS in design. The 2.0 version has been completely redesigned and has a better performance; so we don’t recommend using TCFS v.1.0!

- Question 1.7. Is TCFS suitable for encrypting file systems on non-networked machines - ie, just a stand-alone system?
Sure, TCFS is a kernel modules and operates like NFS, so you can use it on a standalone machine like you could do with NFS. One needs only to compile the kernel with TCFS file system support, to export a directory on the client to itself and run NFSD and XATTRD (a special daemon required for TCFS).

- Question 1.8. For such a use, does TCFS have any speed advantage over CFS?
Sure... Since TCFS is internal to the kernel, it doesn’t generate any overhead due to use any expecial mounting procedure (CFS needs to do a double mount to operate). The TCFS time is equal to the NFS time plus DES time. [See Q1.5 ’How fast (or slow) is TCFS?’].

- Question 1.9. Is TCFS as secure as CFS?
We use CBC-DES, IDEA or RC5 cipher in TCFS to hide all meta-data [See Q2.1 ’How are files

encrypted?' for details] . Security of the model is based on user password, the keys-database can be accessed only using this password. TCFS doesn't ask user to remember other keys.

In future release we plan to encrypt the encryption/decryption key using a passphrase (not just the login password).

- Question 1.10. Does it mean that root can not read my files when they are not in /dev/kmem?
  Sure, root on client can only access /dev/kmem. Blocks of data are readable only while in the process memory or in the VFS caches. The client root can nonetheless access encrypted blocks of data directly from the server. However, that is not very useful, isn't it?

- Question 1.11. Which parties are assumed to be trusted by TCSF?
  TCFS assumes a minimal model of trust as only the root on the client machine needs to be trusted. In fact, not even the root of the server is trusted by TCFS. Remember that the encryption key is not available to the root of the server. Nonetheless, the root on the server can remove or alter the files. There is nothing we can do about that: it is the server's disk after all.
  We are planning to add in future releases of TCFS a mechanism by which the user can detect if a file has been modified.

- Question 1.12. Why do I need to trust the root (admin) of the client machine?
  Because the root of the client has to be trusted. Keep in mind, that the root can modify the kernel, TCFS, or even keep a log of all the keys you pressed and there is nothing to prevent this. Because often root has access to the hardware then the decrypted data is available to root however you are viewing it.
  So if you have reasons not to trust the root of the client machine, TCFS will not provide you with the privacy that you are looking for. In this case, you should get your own Linux box and install a trusted kernel.

- Question 1.13. Is TCFS commercial?
  Currently TCFS is released under the Gnu's Public License which means that you may copy, modify, distribute TCFS for free. For more details, read the file COPYING in the sorce directory of TCFS.
  TCFS will be commercial no sooner than release n-th.0.0 :). To give you an idea, the current release is 2.0.1 and we areplanning to release TCFS 2.1.0 (within the end on 1997).

Section 2. TCFS Components

- Q2.1 How are files encrypted?

- Q2.2 How does TCFS build the encryption key?

- Question 2.1. How are files encrypted?
  TCFS uses a CBC-DES engine to encrypt. We chose DES because it is fast, well studied, and standard. [See Q1.9 'Is TCFS as secure as CFS?']
  Since TCFS 2.0.1 release, IDEA and RC5 cryptographic engines were added in the package; now you can choice, at compilation time, which engine to use to cypher yuor datas.
  We are planing TCFS to be able to use different engines at run time.

- Question 2.2. How does TCFS build the encryption key?
  To build the encryption key, TCFS uses the MD5 algorithm. The key is built following the ANSI X9.17 standard.

Section 3. Administrator Information

- Q3.1 How do I install TCFS?
- Q3.2 Which Operating System is required for the TCFS Server machine?
- Q3.3 Which Operating System is required for the TCFS Client machine?
- Q3.4 Why must I use your modifyed passwd command?
- Q3.5 Why do I need to use youre mount command?
- Q3.6 What is in the file /etc/tcfspasswd?
- Q3.7 Can I use the standard NFS protocol togeter TCFS?
- Q3.8 Can I install the TCFS client and server on the same machine?
- Q3.9 How can I enable a user to use TCFS Filesystem's features?
- Q3.10 It would be possible to boot up from a server using tcfs instead of the base NFS, or would it be better to bootup with the minimum NFS, then load TCFS as a module?

- Question 3.1. How do I install TCFS?
  The TCFS installation is divided into two steps: server installation and client installation.

  \- The server installation consists in running the xattrd daemon and in configuring the machine to export at least one directory to the TCFS client machine.

  \- The client installation consists in recompiling the kernel to support TCFS filesystems, mounting the remote directory exported by the TCFS server with the command "mount" that you find in TCFS distribution, and enabling the users to use TCFS features using tcfsgenkey. [See Q3.9 'How can I enable a user to use TCFS Filesystem's features?]

- Question 3.2. Which Operating System is required for the TCFS Server?
  At this moment TCFS works only on Linux Boxes (Kernel version 2.0.x). [See Q1.3 'On which platforms does it run?']

- Question 3.3. Which Operating System is required for the TCFS Client?
  See Q3.2.

- Question 3.4. Why must I use your modifyed passwd command?
  Because the encryption key is encrypted with the user password. For this reason, we have modified the passwd command by inserting the routine that decrypt and encrypt the key every time the user changes his login password.

- Question 3.5. Why do I need to use your mount command?
  Because we have inserted the new file system (TCFS) in the standard command line. In this way you can mount a TCFS file system with the standard mount usage - i.e., "mount -t tcfs tcfs-srv:/remotedir /localdir".

- Question 3.6. What is in the file /etc/tcfspasswd?
  The file /etc/tcfspasswd contains some informations about the TCFS users, like login name, encrypted user key. In the next release of TCFS, this file will contain a lot of informations :)

- Question 3.7. Can I use the standard NFS protocol together TCFS?
  Sure... you can use NFS and TCFS independently. They are different filesystems, so, they do not create conflicts. Moreover, you can use the same server machine for both NFS and TCFS.

- Question 3.8. Can I install the TCFS client and server on the same machine?
  Sure... [See Q1.7 'Is TCFS suitable for encrypting file systems on non-networked machines - i.e., just a stand-alone system?']

- Question 3.9. How can I enable a user to use TCFS Filesystem's features?
  Each user must have an encryption/decryption key in the /etc/tcfspasswd file. This is done using tcfsgenkey command.

- Q3.10 It would be possible to boot up from a server using tcfs instead of the base NFS, or would it be better to bootup with the minimum NFS, then load TCFS as a module?
  Booting from a diskless machine is not yet implemented. We are planning to extend NFS ROOT support for TCFS. You can boot with a root filesystem using NFS and mount TCFS filesystem later. We encourage this way since we guess TCFS has to be used as a filesystem for user and system private data not for system administration data.

Section 4. User Information

- Question 4.1. How do I use the TCFS features?
  The first time, you must contact your System Administrator so that he can generate an encryption key for you. Then you must execute the tcfslogin command for each session for which you want to access a TCFS filesystem. It is a good idea to have tcfslogin in your .login file. Also before leaving you need to execute tcfslogout.

- Question 4.2. Why must I execute tcfslogin command?
  tcfslogin pushes the encryption/decryption key in the kernel. If you do not execute tcfslogin the kernel does not know how to encrypt/decrypt your files.

- Question 4.3. Why must I execute tcfslogout before closing my session?
  Tcfslogout removes your key from the kernel memory. If you are running some background process, that needs to access a secure file, don't execute tcfslogout otherwise your process will get encrypted bytes.

- Question 4.4. How do I change the flag X on my files?

To set the file to encrypted you must do:
chattr +x filename
To set the file to decrypted tou must do:
chattr -x filename
Thats'all folks :).

- Question 4.5. If I set the X flag on a directory, what happens to the files in the directory?
  If you set the X flag for a directory, nothing happens to the existing files (i.e., the stay the way they were), but all newly created files gets the flag X automatically.

- Question 4.6. How is the X flag inherited?
  The secure flag 'X' is set for every files created in a secure directory or subdirectory.

- Question 4.7. How do I check if a file is encrypted?
  You can check the file's flags using the lsattr command.

- Question 4.8. Can I share my files with other users in a workgroup?
  No way! The next TCFS release will support this feature.

- Question 4.9. Why can't I read my secure files?
  Probably you did not tcfslogin. [See Q4.2 'Why must I execute tcfslogin command?']

- Question 4.10 Can I change my TCFS encryption/decryption key?
  Yes! You can change your session key.
  This is done using the tcfslogin command with the -k flag (see tcfslogin(1) man page).
  This new feature of tcfslogin permits to have different keys for different file systems or different keys for the same file system. In the latter case the user has to be careful and push the appropriate key in the kernel before accessing files. Anyway, the default key, stored in the tcfspasswd file encrypted using your login password, does not change, Yuo can only change the key currently present in the kernel.
  However, if you are afraid the your login password might be compromised, this allows you to bypass the default mechanism.

- Qestion 4.11 Can I read my current encryption/deryption key?

Yes! You can display your current TCFS key, resident in the tcfspasswd file, using tcfsvk utility. The key will be displayed HEX mode because it is generated by tcfsgenkey and has some undisplayable characters. We have planed to modify tcfsvk to be able to read the key that is in the kernel, a.e. the key that you are using, and to display that in clear mode (if it is possible).

- Question 4.12 What kind of security TCFS can grant if my login password is stolen?
  TCFS can ensure yuo to protect yuor files, if your password is stolen, only if you have used a different, not default, security key to crypt [See Q4.10 Can I change my TCFS encryption/decryption key? for more details].

Section 5. Getting Infos and Bugs Report

- Question 5.1. How do I report bugs?
  Send any bugs you find to <tcfs@mikonos.dia.unisa.it> and tell us as many details as possible.

- Question 5.2. I want to subscribe to TCFS mailing list, how can I do it?
  To subscribe to the TCFS mailing list send an e-mail to <majordomo@mikonos.dia.unisa.it> with the follwoing line in the body of the letter:
  subscribe tcfslist

  If you have troubles, send mail to <tcfslist.owner@mikonos.dia.unisa.it> for help.

- Question 5.3. Where can I find the latest TCFS release?
  The latest release of TCFS can be found via HTTP to:

  * mikonos.dia.unisa.it (192.41.218.253) : /tcfs
  * edu-gw.dia.unisa.it (192.41.218.253) : /tcfs
  * www.globenet.it (194.184.182.2) : /~ermmau/tcfs

- Question 5.4. Where can I found more informations about TCFS?

You can visit our site where you find a lot of information about TCFS [See Q5.3 'Where can I found the latest TCFS release?'], or you can send an e-mail to <tcfs@mikonos.dia.unisa.it>. Also a good source of info is the TCFS mailing list. [See Q5.2 'I want to subscribe to TCFS mailing list,how can I do it?'].

Section 6. Who, What and Why.

- Q6.1 Who made TCFS?
- Q6.2 Who needs TCFS?

- Question 6.1. Who made TCFS?
  Tcfs was developed at DIA at Universita' Degli Studi Di Salerno (Italy). A student workgroup is working on it under the supervision of two leaders.

  The workgroup:
  Andrea Cozzolino
  Angelo Celentano
  Aniello Del Sorbo
  Ermelindo Mauriello
  Raffaele Pisapia

  The group leaders:
  Giuseppe Cattaneo
  Giuseppe Persiano

- Question 6.2. Who needs TCFS?
  Everyone that needs to have a secure file system support needs TCFS. TCFS works well on networked machines either/or on a stand-alone machines, so it can be used in every working environment.

Section 7. Administrative information and acknowledgements.

- Q7.1 Feedback is invited.
- Q7.2 Formats in which this FAQ is available
- Q7.3 Authorship and acknowledgements

- Question 7.1. Feedback is invited.
  Please send me your comments on this FAQ.

I accept submissions for the FAQ in any format; All contributions comments and corrections are gratefully received.

Please send them to <angcel@mikonos.dia.unisa.it>.

By the way, if you wish to refer to a question(s) in the FAQ it's most useful for me if you do so by the question heading, rather than the number.

- Question 7.2. Formats in which this FAQ is available.
  This document is available as an ASCII text file, an HTML World Wide Web page, a PostScript file.

  The HTML version of this FAQ is available as
  http://mikonos.dia.unisa.it/tcfs/tcfs-faq.html.

  The ascii and the PostScript versions are available as
  ftp://mikonos.dia.unisa.it/pub/tcfs/.

- Question 7.3. Authorship and acknowledgements.
  This FAQ was compiled by Angelo Celentano, with assistance and comments from others TCFS coders. Thanks also to who sent me comments about this FAQ, and who answered questions on the mailing list.

  Last but not least, thanks to Linus Torvalds and the other contributors for giving us Linux!

Angelo Celentano