CFS und TCFS

2 kryptografische Dateisysteme für Unix von der Idee zur Anwendung

CFS und TCFS

1. Ziele des Dateisystems
2. CFS als Lösung
3. CFS in der Anwendung
4. Verbesserungen bei TCFS
5. Anwendung von TCFS
6. verwendete Verschlüsselungen

1. Ziele des Dateisystems

Was soll CFS bieten?

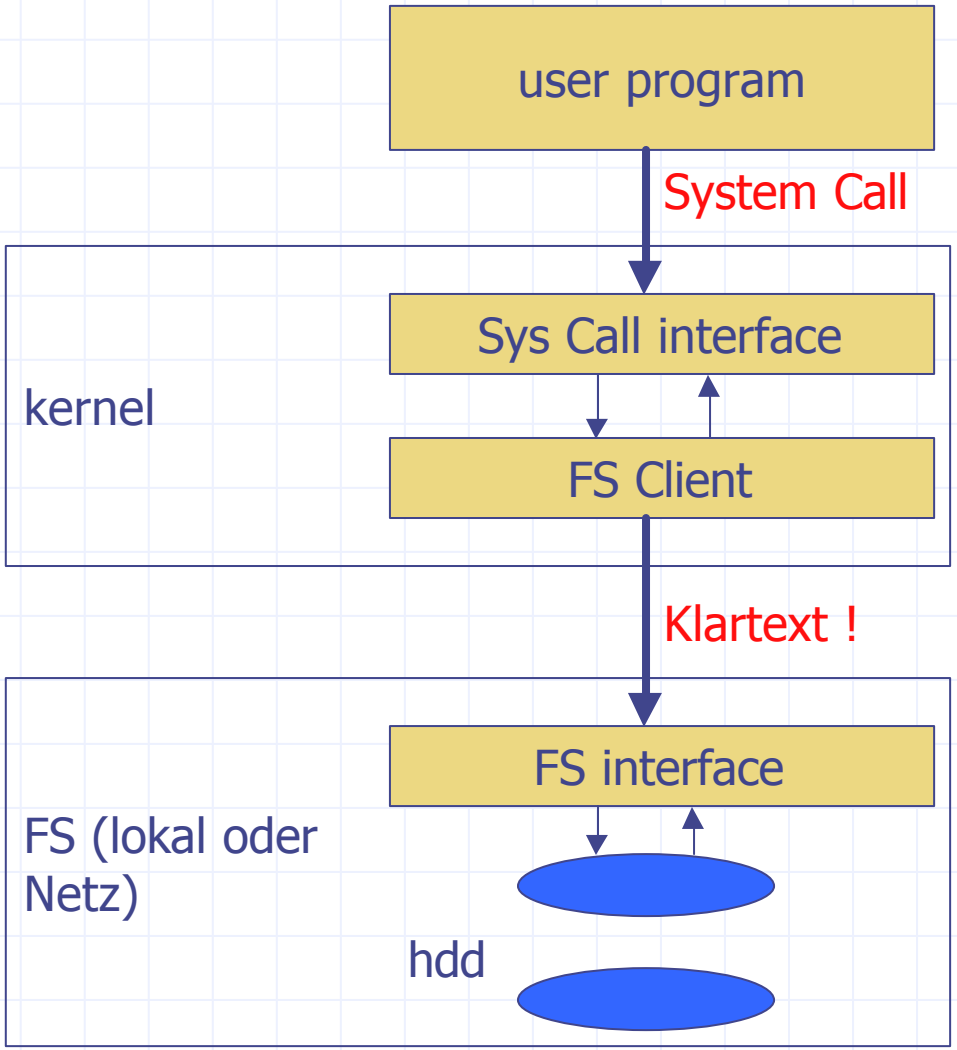
- Effektiven und komfortabler Zugriffsschutz
- Standard Unix Dateisystemschnittstelle (open, read, ...)
- Schutz bei HW-Diebstahl
- Möglichkeiten zum Backup
- Netzwerkfähigkeit

1.1 Effektiven und komfortabler Zugriffsschutz

- Schlüssel soll aus Passwort kreiert werden
- Passwort soll genau einmal eingegeben werden um eine Gruppe von Dateien (directory) zugreifbar zu machen
- mehrere Benutzer sollen mit gemeinsamen Passwörtern gemeinsame Daten verwalten können
- werden die Daten nicht mehr gebraucht sollen sie wieder unzugänglich gemacht werden können

➔ ein Befehl zum Verfügbarmachen eines directory mit Passwortabfrage und einen zum unzugänglich machen

1.2 Unix Filesystem Schnittstelle



➔hier muss die Verschlüsselung wirken

1.3. Möglichkeiten zum Backup

- Sicherung soll ohne Kenntnis des Schlüssels erstellt werden können
- Klartext darf weder Übertragen noch gespeichert werden
- gezielte Wiederherstellung einzelner Dateien soll möglich sein
- Metainformationen wie Dateinamen und Größe sollen verborgen sein

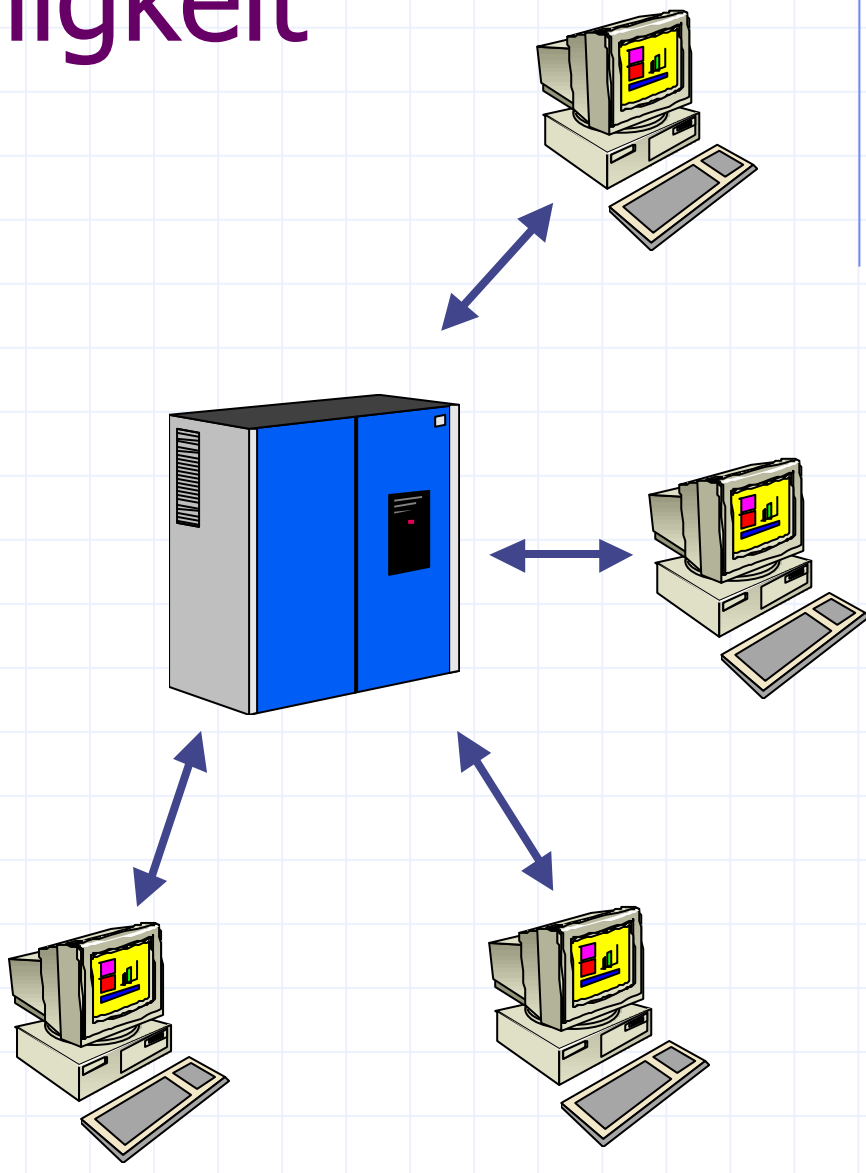
➔ Dateien müssen separat verschlüsselt werden

1.4. Netzwerkfähigkeit

- Auf dem Server sollen die Daten nur verschlüsselt liegen
- Es dürfen keine Daten im Klartext übers Netz geschickt werden
- Server wäre mit Ver-/Entschlüsselung schnell ausgelastet

➔Clients übernehmen Verschlüsselung

➔ Server muss irgendwie verschlüsselte Daten anbieten



2. CFS als Lösung

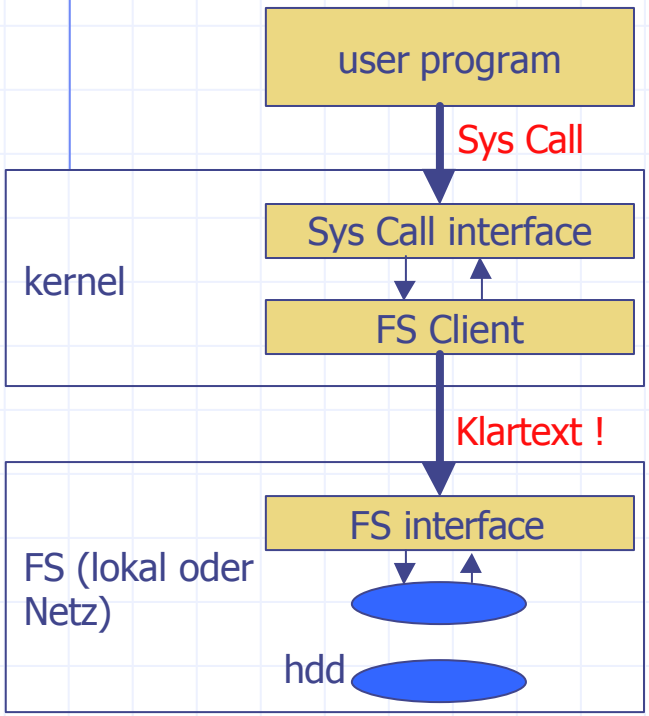
CFS als Lösung

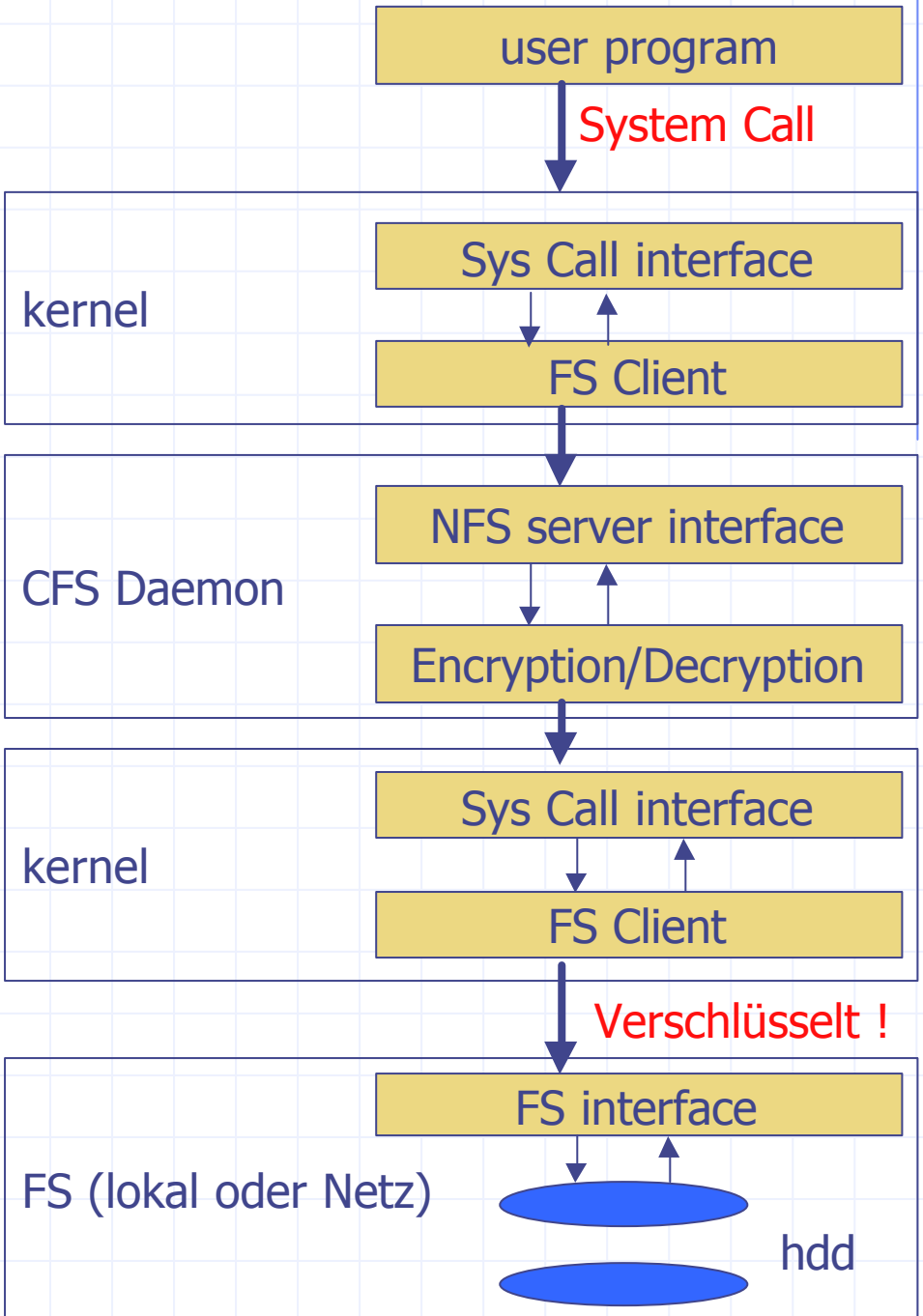
- Dateien sind separat codiert
- die verschlüsselten Dateien liegen in einem beliebigen Verzeichnis im Dateibaum (z.B. ~/crypt, NFS ...)
- entschlüsselte Dateien werden in einem Verzeichnis repräsentiert, das unter Angabe des Schlüssels mit dem verschlüsselten Verzeichnis assoziiert wird
- Zugriff auf das Klartextverzeichnis ist automatisch auf den Benutzer beschränkt; Verzeichnis ist (optional) versteckt
- Problem: root hat Zugriff
- Assoziierung kann wieder gelöst werden

CFS Realisierung

- cfsd ist ein modifizierter NFS Server, der sich auf einem speziellem Port localhost mounten lässt
- die entschlüsselten Verz. sind Unterverz. dieses Verz.
- wird eine Assoziierung hergestellt, ändert der cfsd das repräsentierte virtuelle Verz.

Systemcalls





3. CFS in der Anwendung

Booten

```
cfsd
mount -o port=3049,intr localhost:/dummy crypt
```

- cfsd startet und bietet auf port 3049 den NFS Dienst an
- CFS wird typischerweise auf /crypt gemountet

verschlüsseltes Verz. erstellen

```
bill# cmkdir /home/bill/.secrets mp3
Key: **************
Again: **************
```

- das Verzeichnis /home/bill/.secrets wird erstellt und mit dem angegebenen Passwort (leer) verschlüsselt

Attachen

```
bill# cattach /home/bill/.secrets mp3
Key: **************
```

```
bill# ls –l /crypt
total 1
drwx------ 2 bill 512 May  1 12:44 mp3
```

- das Verzeichnis /home/user1/.secrets wird unter /crypt/mp3 unverschlüsselt dem Benutzer präsentiert

Daten erstellen

```
bill# echo „Index: noch leer“ >/crypt/mp3/index.txt
```

```
bill# cat /crypt/mp3/index.txt
Index: noch leer
```

```
bill# ls –l /crypt/mp3
total 1
-rw-rw-r– 1 bill 11 May  1 12:44 index.txt
```

```
bill# ls –l /home/bill/.secrets/
total 1
-rw-rw-r– 1 bill 15 May  1 12:44 8bA23Hipl991133
```

```
bill# cat /home/bill/.secrets/8bA23Hipl991133
)A2^RU^8^J2j~245^GîssSq![FFJ^4^GM-Z
```

Detachen

nach dem Arbeiten sollte das Verzeichnis wieder unzugänglich gemacht werden

```
bill# cdetach mp3
```

```
bill# ls –l /crypt/mp3
ls: /crypt/mp3: No such file or directory
```

```
bill# ls –l /home/bill/.secrets/
total 1
-rw-rw-r– 1 bill 15 May  1 12:44 8bA23Hipl991133
```

4. Verbesserungen bei TCFS

Nachteile von CFS

- gemeinsame Dateien benötigen extra Verzeichnis und Passwort pro Gruppe
- geschützte und ungeschützte Dateien nicht im selben Verzeichnis
- Anwender muss sich mehrere Kennwörter merken
- durch doppelte mount Operation unnötig langsam
- lediglich ein Algorithmus

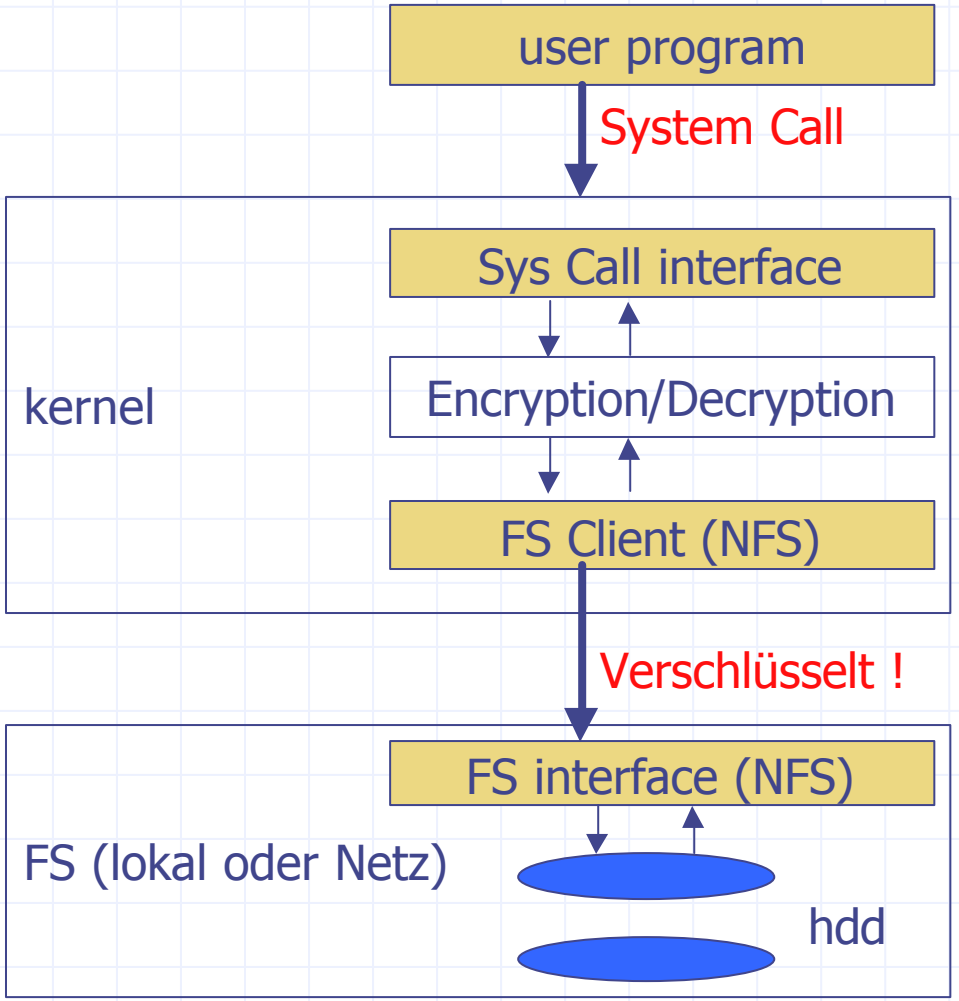
Architektur TCFS

- kernel-space (CFS läuft im user-space)
- transparent gegenüber Benutzeranwendungen, Benutzer, Algorithmen, Dateiserver
- tcfs wird als Dateisystem gemountet und verbindet sich mit NFS Server
- Attribute X und G kennzeichnen die zu verschlüsselnde Dateien
- Verschlüsslungsalgorithmen sind kernel Module

Vorteile TCFS

- Gruppen können leicht organisiert werden, durch threshold-sharing besonderer Schutz
- schneller, sicherer
- Dateien können gemischt werden
- einfache Handhabe mittels Basic KMS (key management scheme)

Systemcall


user program
System Call
Sys Call interface
kernel
Encryption/Decryption
FS Client (NFS)
Verschlüsselt !
FS interface (NFS)
FS (lokal oder Netz)
hdd

5. Anwendung von TCFS

Mounten

```
mount –t tcfs server1:/secrethomes /home
```

- unter /home präsentiert der kernel nun die Daten, die er über NFS von server1 im Verzeichnis /secrethomes angeboten bekommt

Benutzer erzeugen/löschen

```
root# tcfsadduser gates
```

```
root# tcfsrmuser gates
```

- fügt Einträge in die Basic KMS Schlüsseldatenbank ein, bzw. entfernt diese wieder

User-Key erzeugen

```
gates# tcfskeygen
password:
please press 10 random keys:
```

- ermittelt Hashwert aus Passwort (muss mit login-Passwort identisch sein)
- erzeugt User-Key aus Zufallszahlen
- speichert den – mit dem Hashwertverschlüsselten - User-Key

Neue Session beginnen

```
gates# tcfsputkey
password:
```

- holt Schlüssel aus Datenbank
- entschlüsselt diesen durch Hashwert
- trägt ihn in den Kernel ein
- verschlüsselte Dateien sichtbar und zugreifbar

Session beenden

```
gates# tcfsrmkey
```

- entfernt Schlüssel aus kernel
- verschlüsselte Dateien sind nicht mehr zugreifbar

Dateien, Verz. verschlüsseln

```
gates# cd /home/gates/
gates# echo „Hello World“ >world1.txt
gates# chattr +x world1.txt
gates# mkdir win-src
gates# chattr +x win-src
```

- Attribut x markiert eine Datei als verschlüsselt

Beispielsession

```
gates# cd /home/gates/
gates# tcfsputkey
password: *****
gates# echo „Hello World“ >world1.txt
gates# chattr +x world1.txt
```

```
gates# ls
world1.txt
```

```
gates# cp world1.txt world2.txt
gates# mkdir win-src
gates# chattr +x win-src
gates# cp world2.txt win-src/world3.txt
gates# tcfsrmkey
```

```
gates# ls
sdfa33jGeeKa
zU43kKbmwepe
world3.txt
```

Gruppenverwaltung

Gruppe muss als Unixgruppe existieren

```
root# tcfsaddgroup –g team1 –t 1
```

um Zugriff zu bekommen müssen mind. threshould Mitglieder ihre Gruppensession geöffnet haben

```
gates# tcfsputkey –g team1
password:
```

Verschlüsseln und Session schließen analog

```
gates# chattr +g world2.txt
```

```
gates# tcfsrmkey –g team1
```

6. Verschlüsselung

Algorithmen

CFS

- lediglich DES, zwar Verbesserungen, aber keine freie Auswahl

TCFS

- Verschlüsselungs-algorithmus unabhängig von Dateiverschlüs-selung ➔mehrere Algo. möglich(3DES, IDEA, RC5, Blowfish)

Schlüsselverwaltung

CFS

- für jedes Verzeichnis eigener Schlüssel mit eigenem Kennwort

TCFS

- ausgeklügelte Schlüsselkombinationen (master, file & block-key)
- Passwort identisch mit login-pass
- Gruppenschlüssel komplex zusammengesetzt
- alternativ mit Kerberos

verwendete Quellen

- „A Cryptographic File System for Unix“, Matt Blaze (ftp://ftp.research.att.com/dist/mab/cfs.ps)
- „The Design and Implementation of a Transparent Cryptographic Filesystem for UNIX“, Guiseppe Cattaneo, Luigi Catuogno, Aniello Del Sorbo, Pino Persioano (http://www.tcfs.it/docs/freenix01.ps)
- http://www.cbi.pku.edu.cn/Doc/CS/cfs/index.html
- „Abenteuer Kryptologie“, Reinhard Wobst, Addison-Wesley Verlag, München 2001 (aus Uni-Bibliothek)
- FAQs von TCFS (http://tcfs.dia.unisa.it/tcfs-faq.html)
- ftp://ftp.research.att.com/dist/mab/cfs.announce
- http://www.tcfs.it/docs/linux-expo-2001/Diapositiva1.JPG.html