

Verschlüsselte Dateisysteme für NetBSD

Stefan Schumacher
Stefan.Schumacher@Kaishakunin.com

[KAISHAKUNIN.COM](http://kaishakunin.com)
IT-Sicherheitsberatung

Veröffentlicht in der GUUG UpTimes Dezember 2006.

Literaturverzeichnis

Schumacher, Stefan (2006). „Verschlüsselte Dateisysteme für NetBSD“. Deutsch. In: *UpTimes* 4, Seiten 25–31. ISSN: 1860-7683. URL: http://kaishakunin.com/publ/guug-uptimes-cgd_cfs.pdf (besucht am 07. 11. 2009).

```
@article{Schumacher-ut-cgd,  
  author = {Stefan Schumacher},  
  title = {Verschlüsselte Dateisysteme für NetBSD},  
  year = {2006},  
  volume="4",  
  publisher = {German Unix User Group},  
  journaltitle="UpTimes",  
  issn="1860-7683",  
  language = {german},  
  url={http://kaishakunin.com/publ/guug-uptimes-cgd_cfs.pdf},  
  urldate="2009-11-07",  
  pages="25\,--\,31",  
  note=ut,  
}
```

Verschlüsselte Dateisysteme für NetBSD

Stefan Schumacher

*Dieser Artikel stellt zwei Möglichkeiten vor, unter NetBSD verschlüsselte Dateisysteme einzusetzen. Diese wären CGD, eine NetBSD-spezifische Lösung, die Partitionen auf Blockebene verschlüsselt und CFS, das im Prinzip eine Art verschlüsseltes NFS ist. Es läuft unter anderem auf *BSD, Linux und Solaris und kann daher portabel sowie im Netzwerk oder auf Wechseldatenträgern eingesetzt werden.*

Was ist CGD?

CGD ist der „cryptographic disk driver“, der ab NetBSD 2.0 verschlüsselte Partitionen auf Blockebene ermöglicht. Hierbei wird der Zugriff auf die echte Partition über einen Pseudogerätetreiber abgewickelt, der die Daten entschlüsselt an das lesende Programm schickt und verschlüsselt auf die Festplatte schreibt. Dies entkoppelt CGD von verwendetem Dateisystem, so dass die Wahl freigestellt ist. Entwickelt wurde er, um Daten auf Laptops und anderen mobilen Rechnern nach Verlust oder Diebstahl zu schützen. Er bietet also nur Schutz vor unbefugtem Zugriff, wenn die verschlüsselten Partitionen nicht eingebunden sind. Vor anderen Benutzern oder Root schützt eine eingebundene Partition nicht.

CGD verwendet dazu symmetrische Kryptoverfahren mit einem einfachen Passwort, das zum Einbinden der echten Partition in den Pseudogerätetreiber übergeben werden muss. Zusätzlich kann CGD auch beim Starten einen Zufallsschlüssel für eine Partition generieren, der nach dem Herunterfahren verloren geht. Dieses Verfahren bietet sich für die Swap- und tmp-Partition an, da deren Daten nach dem Herunterfahren des Systems nicht mehr von Bedeutung sind. Die Entwicklung und technische Hintergründe werden in [5] erläutert.

Installation und Einrichtung

CGD wird als Pseudogerät im Kernel konfiguriert, dazu muss in der Kernelkonfiguration die Option

`pseudo-device cgd 4` aktiviert werden. Benötigt man mehr als vier verschlüsselte Partitionen, muss man die „4“ als letzte Option erhöhen und mit `/dev/MAKEDEV` die neuen Pseudogeräte erstellen.

CGD unterstützt zur Zeit drei verschiedene Algorithmen, TripleDES, AES und Blowfish. TripleDES ist definitiv nicht empfehlenswert, da es unsicher und extrem langsam ist, Blowfish und AES sind beide recht sicher¹ und im Allgemeinen recht schnell, wobei Blowfish etwas schneller als AES ist. Ich persönlich setze AES für mein Homeverzeichnis und Blowfish für `/tmp` und `swap` ein. Alle Algorithmen laufen im CBC-Modus, d.h. bevor ein Klartextblock verschlüsselt wird, wird er mit dem vorhergehenden, verschlüsselten Block mittels XOR verknüpft. Dies bringt Muster im Klartext durcheinander und sorgt dafür, dass identische Klartextblöcke unterschiedliche Chiffre erzeugen. Im Falle eines Fehlers ist nur der defekte Block und der darauffolgende betroffen, somit hält sich ein möglicher Schaden in Grenzen.

Eine Datenpartition verschlüsseln

Da CGD auf Partitionsebene arbeitet, benötigt man natürlich zu erst eine Partition die verschlüsselt werden soll. Meist ist dies die Homepartition. Wird die Partition bereits genutzt, müssen die Daten gesichert und in die neue, verschlüsselte, Partition kopiert werden. Die Originaldaten sind danach aus Sicherheitsgründen sicher zu löschen,

also mehrfach zu überschreiben. Dies erledigt man mit `rm -P` oder besser mit `wipe` bzw. `destroy` aus `Pkgsrc`.

Ist die Partition nun leer und kann für CGD verwendet werden, erstellt man zuerst die CGD-Konfiguration mit `cgdconfig(8)` wie in Abbildung 6 beschrieben. Dazu kann man verschiedene Optionen übergeben, unter anderem die Verifikationsmethode, den Dateinamen für die Konfiguration und den gewünschten Algorithmus sowie die Schlüssellänge. Mit der Verifikationsmethode wird überprüft, ob die CGD-Partition korrekt eingebunden wurde, d.h. hier wird geprüft, ob ein FFS-Dateisystem auf der Partition existiert, nachdem das Passwort angegeben wurde. Wird kein Dateisystem erkannt, muss das Passwort erneut eingegeben werden. Nachdem die Konfiguration erstellt wurde, wird mit dem zweiten Befehl das Passwort festgelegt. Anschließend wird ein Dateisystem angelegt und die Pseudopartition nach `/home` eingemountet, wo sie ganz normal verwendet werden kann. Nun können die zuvor gesicherten Daten wieder auf die verschlüsselte Partition zurückgesichert werden. Um eventuelle Angriffe auf die verschlüsselten Blöcke zu erschweren, ist die Partition einmal voll mit Zufallsdaten zu füllen. Dazu kann man bspw. mit `dd if=/dev/urandom of=/home/wipe` aus dem Zufallsgenerator Daten in eine Datei schreiben. Dies ist allerdings zeit- und lastintensiv und da man keine Zufallsdaten im streng mathematischen Sinne benötigt, reicht es auch einfach MP3s, Videos oder sonstige Daten zu kopieren und wieder zu löschen. Somit wer-

¹Blowfish war AES-Finalist

den alle Blöcke befüllt und die später kopierten eigentlichen Daten gehen in dem Rauschen unter. Dies sollte als Root geschehen, um auch den reservierte Sicherheitszuschlag zu befüllen.

Soll die Partition deaktiviert werden, muss man sie unmounten und mit `cgdconfig -u /dev/cgd0a` deaktivieren. Man kann nun nicht mehr auf die Daten zugreifen. Die Konfigurationsdatei (im Bsp. `/etc/cgd/wd0h`) enthält alle Metadaten zur verschlüsselten Partition (u.a. Algorithmus, Schlüssellänge und den *Passwort-Hash*). Daher muss sie unbedingt geschützt gesichert werden, denn ohne diese Datei lässt sich eine CGD-Partition nicht mehr entschlüsseln.

Um die Homepartition bei jedem Startvorgang automatisch einzubinden, muss man in der `/etc/rc.conf` `cgd=yes` setzen und `/dev/cgd0a /home ffs rw 1 1` in die `/etc/fstab` eintragen.

Man wird nun automatisch nach der Passphrase gefragt und die Partition wird danach ggf. mit `fsck` überprüft und eingebunden.

Mehr Informationen zu CGD stehen in den Man-Pages [1] und [2].

CGD für Swap und /tmp

Die Swappartition wird verwendet, um Daten aus dem Arbeitsspeicher auszulagern, bspw. wenn eine Anwendung mehr Speicher braucht als aktuell frei ist. Die kann natürlich die Sicherheit des Systems gefährden, wenn es sich dabei um eigentlich verschlüsselte Daten handelt, die unverschlüsselt im RAM liegen und ausgelagert werden. Bei einem Neustart des Systems wird zwar die /tmp-Partition bereinigt, die dort gelegenen Daten können aber problemlos rekonstruiert werden.

Damit dies nicht passiert, kann man auch die Swappartition entweder als RAM-Disk im Speicher anlegen oder mit CGD verschlüsseln. Im Prinzip geht man hier genauso vor wie bei einer normalen Datenpartition, lediglich die Schlüsselgenerierung wird abgeändert. CGD unterstützt „urandomkey“ als Schlüsselmethode. Hierbei werden einfach aus `/dev/urandom`, einem Pseudozufallsgenerator, Zeichen ausgelesen und als Schlüssel verwendet. Der Schlüssel wird dabei nicht gespeichert, so das beim Absturz des Systems oder nach dem Herunterfahren die Partition nicht mehr entschlüsselt

werden kann. Dies ist aber bei einer Swappartition nicht notwendig, da sie bei jedem Systemstart neu initialisiert wird.

In Abbildung 7 wird eine solche Konfiguration erstellt. Hierzu wird die Swappartition zuerst deaktiviert und anschließend eine Konfiguration mit der entsprechenden Schlüsselmethode erzeugt.

Da bei jedem Start des Systems die Swappartition neu angelegt wird, muss auch das Disklabel zurückgespielt werden, da es standardmäßig auf „4.2BSD“ statt „swap“ gesetzt wird. Dies geschieht am einfachsten indem man die Swappartition einmal händisch konfiguriert, das Label sichert und später automatisch bei jedem Start einbindet.

Abbildung 8 demonstriert das Vorgehen. Zuerst wird das CGD-Gerät konfiguriert (Befehl 1), das Disklabel erzeugt (2) und in eine Datei gesichert (3). Die Befehle 4, 5 und 6 spielen das Disklabel zurück, konfigurieren die Swappartition wieder als Swap und zeigen zur Kontrolle alle aktiven Swap-Geräte.

Die tmp-Partition sollte ebenso wie die Swappartition geschützt und daher auch ähnlich konfiguriert werden. Es muss hierbei lediglich das Dateisystem nach der Konfiguration neu angelegt werden, da die /tmp-Partition als normale Datenpartition eingebunden wird.

Automatisieren kann man diesen Vorgang für jeden Systemstart indem man ein Shellskript (`/etc/rc.d/cgdswap`) erstellt, ausführbar macht und anschließend in `/etc/rc.conf` `cgdswap=yes` setzt. Durch diese Variante wird mittels `# BEFORE: LOGIN` das Skript vor der Bereitstellung von Logins ausgeführt und Manipulation verhindert.

Die Abbildung 9 zeigt das ausführbare Shellskript `/etc/rc.d/cgdswap`, welches zuerst einige Variablen für den `rc.d`-Prozess anpasst. Anschließend werden Swap und /tmp konfiguriert und die Swap-Partition eingebunden. Die tmp-Partition wird ganz normal als `/dev/cgd0a /home ffs rw 0 0` in `/etc/fstab` eingetragen und so gemountet.

Neben einer CGD-Partition kann man für /tmp auch ein MFS (Memory File System) verwenden, dabei wird ein Teil des RAMs für das

Dateisystem verwendet. Somit sind die Daten nur im flüchtigen Speicher vorhanden und können nicht rekonstruiert werden. Allerdings ist hierbei die mögliche Größe – insbesondere bei Laptops – vom installierten RAM abhängig. Der Swap-Speicher muss auch nicht zwangsläufig eine eigene Partition sein, er kann auch als Datei angelegt und eingebunden werden. Diese Partition kann dann z. B. im verschlüsselten Home-Verzeichnis angelegt werden. Dann ist die Swap-Partition allerdings vom Home-Verzeichnis abhängig, wird diese deaktiviert, bricht auch der Swap weg, was meist sehr unangenehme Konsequenzen hat.

CGD für Container oder CDs/DVDs

NetBSD kann mittels `vnd(4)` virtuelle Datenträger bereitstellen. Dies heißt im Klartext, das man eine Datei erzeugt, in dieser Datei ein Dateisystem anlegt und es über ein Loopbackgerät (`/dev/vnd0d`) wie eine normale Platte einbindet.

Da sich CGD nicht dafür interessiert, ob es auf einer „echten“ Platte oder einem VND-Gerät angelegt wird, kann man so verschlüsselte Container erzeugen.

Dazu befolgt man Abbildung 11, welches zuerst eine 695MB große Datei mit `dd(1)` erzeugt, diese mit `vnconfig(8)` als virtuelle Platte einbindet und anschließend mit `cgdconfig(8)` verschlüsselt.

Den entstandenen Container kann man auf Wechseldatenträger kopieren, per NFS importieren oder als Container auf eine CD brennen. Es besteht auch die Möglichkeit den Container *direkt* auf eine CD/DVD zu brennen, dann muß allerdings zum mounten der CD das Disklabel wegen der Blockgröße angepasst werden. Näheres hierzu wird in [7], Abschnitt `encrypted CDs/DVDs` sowie in den Man-Pages [3] bzw. [4] beschrieben. Ein Container kann auch für eine Home-Partition verwendet werden, bspw. dann, wenn man die Platte nicht neu partitionieren möchte. Hubert Feyrer beschreibt dies in seinem Blog-Eintrag [8].

Backups, Passwörter und Co.

CGD-verschlüsselte Partitionen lassen sich verständlicherweise nicht einfach

```
# cgdconfig -g -V ffs -o /etc/cgd/wd0h aes-cbc 256
# cgdconfig -V re-enter cgd0 /dev/wd0e
# newfs /dev/cgd0a
# mount /dev/cgd0a /home
```

Abbildung 6: CGD-Konfiguration erstellen

```
# swapctl -d /dev/wd0b
# cgdconfig -g -k urandomkey -o /etc/cgd/wd0b \
    blowfish-cbc 192

# cat /etc/cgd/wd0b
algorithm blowfish-cbc;
iv-method encblkno;
keylength 192;
verify_method none;
keygen randomkey;
```

Abbildung 7: CGD für die Swappartition konfigurieren

```
1 # cgdconfig cgd1 /dev/wd0b
  # disklabel -i /dev/cgd1a
  partition> a
  Filesystem type [?] [swap]: swap
  Start offset ('x' to start after partition 'x')
                                [0c, 0s, 0M]:
  Partition size ('$' for all remaining)
                                [755.508c, 1547280s, 755.508M]:

  partition> W
  Label disk [n]? y
  Label written
  partition> Q
3 # disklabel -r /dev/cgd1a > /etc/cgd/swap.disklabel
4 # disklabel -R -r cgd1 /etc/cgd/swap.disklabel
5 # swapctl -a /dev/cgd1a
6 # swapctl -l
Device      512-blocks Used Avail Capacity Priority
/dev/cgd1a  1547280    0 1547280    0% 0
```

Abbildung 8: Disklabel für die Swappartition sichern

sichern. Lediglich eine bitweise Kopie mit `dd(1)` ließe sich von einem CGD-Gerät anfertigen. Daher müssen CGD-Partitionen eingebunden werden, um die Daten unverschlüsselt zu sichern. Das Archiv ist dann allerdings zu verschlüsseln, bspw. mit `OpenSSL`, `mrcrypt` oder `GnuPG`.

Bei der Wahl eines Passwortes ist natürlich ein starkes Passwort auszuwählen. Es darf nicht zu kurz sein (mindestens 10 Buchstaben), nicht im Wörterbuch stehen und verschiedene Ziffern und Sonderzeichen enthalten. Allerdings enthält der NetBSD-Kernel standardmäßig nur das US-Tastaturlayout und stellt erst nach dem Einbinden der CGD-Partitionen das deutsche Layout zur Verfügung. Daher empfiehlt es sich

im neu zu bauenden Kernel, der auch das CGD-Gerät enthält, mit den Optionen `PCKBD_LAYOUT` bzw. `UKBD_LAYOUT` das deutsche Tastaturlayout einzukompilieren.

Auf Laptops, die Powermanagement mit APM oder ACPI unterstützen, kann man eingebundene Partitionen mithilfe der APM-Skripte deaktivieren und so schützen. Unter `/usr/share/examples/apm/script` befindet sich eine Vorlage für `apmd(8)`, in der in der Funktion `suspend` und `standby` bzw. `resume` die Verschlüsselung de- bzw. reaktiviert werden kann.

Das kryptographische Dateisystem `cfs`

Was ist CFS?

CFS ist das „Cryptographic Filesystem“ von Matt Blaze, welches als erstes Dateisystem Verschlüsselung implementierte und im Prinzip wie NFS arbeitet. Die Entwicklung und Hintergründe werden in [6] näher erläutert. Dies bedeutet, das komplette Verzeichnisse und deren Inhalt verschlüsselt im Dateisystem abgelegt werden. Beim Zugriff werden sie über den `cfsd(8)`, der auf 127.0.0.1 lauscht, unverschlüsselt an das anfragende Programm übergeben. Soll die Datei zurückgeschrieben werden geschieht dies wieder über den `cfsd(8)`, der die

```
#!/bin/sh

# BEFORE: LOGIN
# PROVIDE: disks

$_rc_subr_loaded . /etc/rc.subr

name="cgdswap"
rcvar=$name
start_cmd="cgd_swap"
stop_cmd=":"

cgd_swap()
{
echo "CGD-Swap und -Temp werden konfiguriert"

# Swap einbinden
/sbin/cgdconfig cgd1 /dev/wd0b
/sbin/disklabel -R -r cgd1 /etc/cgd/swap.disklabel
/sbin/swapctl -a /dev/cgd1a

# Temp einbinden
/sbin/cgdconfig cgd2 /dev/wd0h
/sbin/newfs /dev/cgd2a
}

load_rc_config $name
run_rc_command "$1"
```

Abbildung 9: Verschlüsselte Temp- und Swappartition beim Start einbinden

```
# cgdconfig -g -k randomkey -o /etc/cgd/wd0h \
    blowfish-cbc 192

# cat /etc/cgd/wd0h
algorithm blowfish-cbc;
iv-method encblkno;
keylength 192;
verify_method none;
keygen randomkey;
#
```

Abbildung 10: CGD für /tmp konfigurieren

```
$ dd if=/dev/zero of=image bs=1m count=695
# vnconfig -c vnd0 image
# cgdconfig -g -o image.cgd blowfish-cbc 128
# cgdconfig -V re-enter cgd3 /dev/vnd0a image.cgd
# newfs /dev/cgd3a
# mount /dev/cgd3a /mnt/
...
# umount /dev/cgd3a
# cgdconfig -u cgd3
# vnconfig -u vnd0
```

Abbildung 11: CGD-Container

Datei entsprechend verschlüsselt in nicht daß CFS überhaupt installiert ist. möglich sind. Abbildung 12 zeigt bei-
das Dateisystem schreibt. Für den Be- CFS verschlüsselt neben dem Inhalt spielhaft ein verschlüsseltes und ent-
nutzer ist CFS nach der Eingabe der der Dateien auch die Namen, so daß schlüsseltes CFS-Verzeichnis.
Passphrase transparent, d.h. er bemerkt keinerlei Rückschlüsse auf den Inhalt Da CFS auf Dateiebene ver-

schlüsselt, lassen sich die Dateien normal auf Wechseldatenträger kopieren und so zwischen verschiedenen Systemen austauschen. Ebenso ist es möglich, die verschlüsselten Verzeichnisse per NFS von einem Server zu importieren und lokal zu entschlüsseln.

Da NetBSDs `mount_nfs(8)` keinen anderen Port als 2049 ansprechen kann, kann man den CFS `cfsd(8)` nicht gleichzeitig mit einem NFS-Daemon betreiben.

Installation und Konfiguration

CFS wird aus `pkgsrc/security/cfs` installiert und kann anschließend konfiguriert werden. Dazu muss ein Loopbackverzeichnis erstellt werden, durch das die Daten durchgeschleift werden, außerdem natürlich das eigentliche verschlüsselte Verzeichnis und ein Mountpunkt, an dem die entschlüsselnden Verzeichnisse eingemountet werden. In Abbildung 13 wird `/home/stefan/crypt` als Daten- und `/null/` als Loopbackverzeichnis angelegt und zum NFS-Export vorbereitet, `mountd(8)` und `cfsd(8)` werden (ggf. neu) gestartet. Anschließend muss bei jedem Start `mountd(8)` via `/etc/rc.conf` gestartet werden. Um `cfsd(8)` zu starten und `/null` zu mounten, können die letzten beiden Zeilen in `/etc/rc.local` eingefügt werden.

Nachdem die ersten beiden Verzeichnisse eingerichtet und die Dienste gestartet sind, kann man das verschlüsselte Verzeichnis einrichten. Dazu existiert mit `cfs_mkdir(1)` ein eigener Befehl. Mit diesem Befehl legt man auch den Verschlüsselungsmodus fest.

CFS unterstützt verschiedene Algorithmen, siehe Abbildung 1. DES ist sehr langsam und definitiv nicht mehr sicher, MacGuffin und SAFER-SK128 sind nicht sehr verbreitet, Blowfish hingegen ist sehr weit verbreitet, sehr schnell und wird bisher als recht sicher betrachtet.

Abbildung 14 generiert ein mit Blowfish verschlüsseltes Verzeichnis. Dabei verlangt es eine mindestens 16-stellige Passphrase, die entsprechend sicher gewählt werden muss. Um das Verzeichnis nutzen zu können, muss es mit `cfs_attach(1)` eingebunden werden. Man übergibt als Option zuerst den Pfad des verschlüsselten

Verzeichnisses und als zweite Option einen frei gewählten Namen, unter dem das Verzeichnis im Loopbackverzeichnis eingemountet wird. Im Beispiel ist `/home/stefan/encrypted` dann als `/home/stefan/crypt/usable.cfsdir` ansprechbar und kann entsprechend genutzt werden. Alle Dateien die nach `/home/stefan/crypt/usable.cfsdir` kopiert werden, werden verschlüsselt unter `/home/stefan/encrypted` abgelegt. Soll das entschlüsselnde Verzeichnis wieder entfernt werden, gibt man `cfs_detach` und den Verzeichnisnamen ein.

CFS auf Wechselmedien

CFS lässt sich hervorragend auf Wechselmedien einsetzen, solange diese ein Unix-Dateisystem unterstützen. USB-Sticks oder ZIP-Disketten müssen also mit FFS, LFS, `ext2`, `ext3` oder Ähnlichem formatiert werden, anschließend kann man ein verschlüsseltes CFS-Verzeichnis komplett auf das Medium kopieren oder auf dem Medium erstellen. Das Medium kann dann auf verschiedenen Rechnern verwendet werden, solange diese (a) das Dateisystem lesen können und (b) CFS installiert haben.

Ich habe die Portabilität getestet indem ich einen USB-Stick mit `ext2` formatiert und ein CFS-Verzeichnis angelegt habe. Dieses ließ sich von NetBSD, FreeBSD und Gentoo Linux anstandslos einbinden. Andere Unixes sollten damit auch keine Probleme haben.

Neben wiederbeschreibbaren Wechselmedien kann man auch CDs bzw. DVDs mit CFS schützen. Dazu muss man lediglich darauf achten das die Dateien alle auf die CD passen und die verschlüsselten Verzeichnisse mit `mkisofs(8)` in ein ISO-Image überführen, das gebrannt werden kann.

Ist auf dem Zielrechner der `cfsd(8)` gestartet und entsprechend konfiguriert, kann man die CD mounten und das verschlüsselte Verzeichnis normal mit `cfs_attach(1)` einbinden.

CFS und Datensicherung

Da CFS auch die Pfadnamen verschlüsselt, ist eine händische Datensicherung recht problematisch. Entweder man sichert immer das komplette verschlüsselte Verzeichnis oder verwendet `dump(8)` mit `Dumplevels`

oder andere Backupprogramme mit `Levels/Datenoptionen`.

Man kann natürlich auch die Daten entschlüsseln und verschlüsselt sichern, dann ist es aber zwingend erforderlich, das Sicherungsarchiv zu schützen, sprich zu verschlüsseln.

CGD oder CFS?

Welches System ist zu bevorzugen, CGD oder CFS?

Diese Frage lässt sich nicht einfach mit „Entweder – Oder“ beantworten, da beide Systeme andere Anforderungen und Ziele haben.

CGD ist ein Jahrzehnt jünger, wesentlich transparenter und im Allgemeinen auch schneller als CFS. Es arbeitet aber auf Blockebene, CFS hingegen auf Dateiebene. Und das ist auch ein wichtiger Unterschied. Mit CGD verschlüsselte Dateien lassen sich nur entschlüsselt sichern - CFS-verschlüsselte Dateien hingegen kann man auch verschlüsselt sichern.

CGD ist ursprünglich dazu gedacht, Laptops etc. bei Verlust zu schützen, daher schützt es nur eine Partition, CFS hingegen kann beliebig viele *verschiedene* verschlüsselte Verzeichnisse in einer Partition anlegen und ist somit wesentlich flexibler und erlaubt eine bessere Granularität als CGD. Möchte man an einem Mehrbenutzerrechner jedem Benutzer zugestehen, sein Homeverzeichnis mit CGD zu schützen, muss zwangsläufig jeder Benutzer auch eine eigene Partition bekommen.

CGD ist also auf einem Mehrbenutzerrechner nicht wirklich einsetzbar. Hat man hingegen ein Einbenutzerrechner oder einen Rechner für einige wenige Benutzer, kann man CGD problemlos einsetzen.

Viel wichtiger ist aber, daß man CGD und CFS problemlos kombinieren kann - in einer Partition. Auf meinem Laptop und meinen Arbeitsrechnern sind die Homepartitionen (natürlich auch `/tmp` und `Swap`) jeweils mit CGD verschlüsselt. Zusätzlich sind auch noch `~/mail`, `~/gnupg` und einige andere Verzeichnisse mit CFS verschlüsselt. Dies ist bspw. recht praktisch, wenn ich einen Vortrag halte und mein Rechner in einem unsicheren Netz ist. Ich kann meine Homepartition mit CGD problemlos einmounten, die CFS-geschützten Dateien sind aber

```

$> ls -la loop/entschluesselttest/
total 18104
drwxr-xr-x 2 stefan stefan      512 Apr 21 21:52 ./
drwx----- 6 stefan stefan     1024 Apr 21 21:51 ../
-rw-r--r-- 1 stefan stefan    311296 Apr 21 21:52 brief.tex
-rw-r--r-- 1 stefan stefan     8094 Apr 21 21:51 dmesg
-rwxr-xr-x 1 stefan stefan   8898275 Apr 21 21:51 netbsd*
$> ls -la crypto/cd98d68fd9f031bc/
total 18104
drwxr-xr-x 2 stefan stefan      512 Apr 21 21:52 ./
drwxr-xr-x 6 stefan stefan     1024 Apr 21 21:51 ../
lrwxrwxrwx 1 stefan stefan        8 Apr 21 21:51 \
.pvect_11f8d60dcf7ef480@ -> fe4fdf15
lrwxrwxrwx 1 stefan stefan        8 Apr 21 21:51 \
.pvect_2ff8f5c49324f41a@ -> 69837ab5
lrwxrwxrwx 1 stefan stefan        8 Apr 21 21:52 \
.pvect_de90918d9ef9382867086595913ba1c6@ -> 475d934f
-rwxr-xr-x 1 stefan stefan    8898283 Apr 21 21:51 \
11f8d60dcf7ef480*
-rw-r--r-- 1 stefan stefan     8102 Apr 21 21:51 \
2ff8f5c49324f41a
-rw-r--r-- 1 stefan stefan    311296 Apr 21 21:52 \
de90918d9ef9382867086595913ba1c6

```

Abbildung 12: Verschlüsseltes und entschlüsseltes CFS-Verzeichnis

```

$ mkdir /home/stefan/crypt
# mkdir /null
# chmod 0 /null
# echo /null localhost >> /etc/exports
# /etc/rc.d/mountd [re]start && /usr/pkg/sbin/cfsd
# echo "mount -o intr,-2,-w=4096,-r=4096 127.0.0.1:/null \
/home/stefan/crypt" >> /etc/rc.local
# echo `which cfsd` >> /etc/rc.local

```

Abbildung 13: CFS-Verzeichnisse einrichten und Dienste starten

Option	Algorithmus
1	einfaches DES im Hybridmodus mit zwei Schlüsseln
3	3DES mit drei Schlüsseln
b	Blowfish mit 128-Bit-Schlüssel
m	MacGuffin
s	SAFER-SK128

Tabelle 1: verfügbare CFS-Algorithmen

```

$ cfs_mkdir -b /home/stefan/encrypted
$ cfs_attach /home/stefan/encrypted usable_cfsdir
$ cfs_detach usable_cfsdir

```

Abbildung 14: CFS Verzeichnis anlegen, einbinden und deaktivieren

immer noch verschlüsselt und somit geschützt. Man sollte dann aber aus Sicherheitsgründen starke und unterschiedliche Passwörter für CGD und CFS wählen.

Autor

Stefan Schumacher beschäftigt sich in seiner Freizeit mit japanischen Kampfkünsten sowie mit NetBSD und PostgreSQL. Seine persönliche Webseite ist unter www.net-tex.de erreichbar.

bar.

Literaturverzeichnis

- [1] `cgd(4)`
<http://netbsd.>

- gw.com/cgi-bin/
man-cgi?cgd
- [2] cgdconfig(8)
<http://netbsd.gw.com/cgi-bin/man-cgi?cgdconfig>
- [3] vnd(4)
<http://netbsd.gw.com/cgi-bin/man-cgi?vnd>
- [4] vnconfig(8)
<http://netbsd.gw.com/cgi-bin/man-cgi?vnconfig>
- [5] Roland C. Dowdeswell & John Ioannidis: *The Cryptographic Disk Driver*
<http://www.imrryr.org/~elric/cgd/cgd.pdf>
- [6] Matt Blaze: *A Cryptographic File System for Unix*
<http://www.cryptocom/papers/cfs.pdf>
- [7] The NetBSD Developers: *The NetBSD Guide*
<http://NetBSD.org/guide/en/chap-cgd.html>
- [8] Hubert Feyrer: *Cryptographic File (CGF), or how to keep sensitive data on your laptop* http://www.feyrer.de/NetBSD/blog.html/nb.20060823_2311.html