HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Kriptirani datotečni sustavi na Linux operacijskim sustavima

CCERT-PUBDOC-2005-11-141

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada ovaj je dokument, koji je nastao suradnjom CARNet CERT-a i LS&S-a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr - nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, www.lss.hr - laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1. Uvod

Važan aspekt zaštite podataka pohranjenih na tvrdim diskovima računala, osobito prijenosnih, je enkripcija datotečnog sustava. Ovim relativno jednostavnim postupkom zaštite, moguće je spriječiti otkrivanje povjerljivih podataka u slučaju gubitka ili krađe prijenosnog računala, kao i neovlašten pristup povjerljivim podacima od strane napadača koji je ostvario fizički pristup računalu. Većina modernih operacijskih sustava sadrži ugrađene mehanizme za zaštitu podataka enkripcijom, a poznat je i niz komercijalnih rješenja kojima je moguće nadograditi operacijske sustave.

Linux operacijski sustav korisniku nudi na izbor nekoliko metoda kojima je moguće kriptirati podatke pohranjene na tvrdom disku:

- "loopback" enkripcija na razini jezgre operacijskog sustava
- Sustav enkripcije korisničkih direktorija (EHD – *Encrypted Home Directories*)
- CFS (*Cryptographic File System*) i TCFS (*Transparent Cryptographic Filesystem*) datotečni sustavi
- Ppdd (*Practical Privacy Disk Driver*)

Osim spomenutih, postoji i niz komercijalnih rješenja (npr. BestCrypt, Virtual Private Disk, Itd.) koje je moguće implementirati na Linux operacijskom sustavu, ali ona neće biti obuhvaćena ovim dokumentom.

Idealan sustav za enkripciju podataka na tvrdom disku trebao bi biti koncipiran tako da pruža dovoljnu razinu sigurnosti pohranjenih podataka, a ipak bude jednostavan za implementaciju i svakodnevno korištenje. Pri tome bi trebao ispunjavati sljedeće zahtjeve:

- **Jednostavno upravljanje ključevima** – pristup zaštićenim podacima ostvaruje se pomoću korisničkih ključeva. Sustav mora biti izveden tako da od korisnika ne zahtijeva unošenje ključeva prilikom svake operacije čitanja i pisanja po datotečnom sustavu, već da se jednom unesen ključ smatra pouzdanim sve dok traje korisnička sjednica.
- **Transparentan pristup datotekama** – kriptirane datoteke ne bi se smjele razlikovati od običnih datoteka pohranjenih na disku (jednom kada korisnik unese ispravan ključ), tj. pristup tim datotekama od strane aplikacija mora biti transparentan za krajnjeg korisnika.
- **Transparentne performanse** – brzina pisanja i čitanja podatka na kriptiranom datotečnom sustavu neizbježno je manja od one na uobičajenim datotečnim sustavima. Ipak, kriptirani sustav morao bi biti izveden tako da se ne naruši normalan rad korisnika na računalu i na taj način obeshrabri korisnika da koristi enkripciju datotečnog sustava.
- **Zaštita sadržaja kriptiranih datoteka** – Osim što sadržaj kriptiranih datoteka mora biti nerazumljiv korisniku koji nema odgovarajući pristupni ključ, sustav mora biti izveden tako i da je nemoguće detektirati identične sekvence podataka koje se pojavljuju u nekriptiranoj datoteci ili napraviti usporedbu dvije kriptirane datoteke kako bi se utvrdilo da li su one jednake u nekriptiranom obliku.
- **Zaštita osjetljivih meta podataka** – osim samog sadržaja datoteke, potrebno je zaštititi i njene meta podatke koji neovlaštenom korisniku mogu otkriti povjerljive informacije. Posebno je potrebno onemogućiti čitanje imena kriptiranih datoteka, bez odgovarajućeg ključa.
- **Zaštita mrežnog prometa** – prilikom korištenja distribuiranih datotečnih sustava, postoji mogućnost prisluškivanja mrežnog prometa od strane napadača, u svrhu prikupljanja osjetljivih informacija. Ukoliko su podaci na distribuiranom datotečnom sustavu kriptirani, neophodno je da se u takvom obliku prenose i putem računalne mreže.
- **Kompatibilnost sa svim dijelovima računalnog sustava** – korištenje kriptiranog datotečnog sustava ne smije utjecati na funkcionalnost ostalih dijelova sustava, npr. sigurnosne pohrane podataka (*backup*).
- **Portabilnost** – kriptirani datotečni sustav mora biti izveden tako da omogućuje prenošenje kriptiranih datoteka na bilo koji drugi sustav koji ima implementiranu enkripciju datotečnog sustava tj. uz korištenje odgovarajućeg ključa, datoteke bi trebalo biti moguće otvoriti na bilo kojem sustavu.

- **Istovremeni pristup** – sustav mora omogućavati istovremeni pristup kriptiranim datotekama svim korisnicima i procesima koji posjeduju ovlasti pristupa tj. pristupne ključeve.
- **Kompatibilnost s novim tehnologijama** – sustav za enkripciju mora biti koncipiran tako da podržava buduće tehnologije za pohranu ključeva, poput trenutno popularnih smart kartica.

Ranije spomenute tehnike enkripcije datotečnih sustava, koje će biti opisane u nastavku, nastoje udovoljiti gore navedenim zahtjevima, iako niti jedna od njih ne udovoljava u potpunosti.

2. Kernel Loopback enkripcija

"Loopback" enkripcija je klasična metoda enkripcije podataka na Linux operacijskom sustavu, koja funkcionira na razini jezgre samog operacijskog sustava. Enkripcija je implementirana pomoću posebnog (virtualnog) "loopback" uređaja u koji se zapisuju podaci, kao da se radi o bilo kojem drugom uređaju za pohranu podataka. Naravno podaci se zapravo ne zapisuju u takav virtualni uređaj, već se kriptiraju i kao takvi pohranjuju na stvarni medij (u našem slučaju tvrdi disk). Budući da se proces enkripcije i dekripcije podataka odvija na razini jezgre operacijskog sustava, ovaj način pohrane podataka u potpunosti je transparentan prema korisniku.

Pomoću ove metode, moguće je kriptirati zasebne particije i direktorije na tvrdom disku, ali i čitav tvrdi disk, uključujući i `swap` datotečni sustav. Budući da se podaci koji se čuvaju na kriptiranom dijelu datotečnog sustava mogu vrlo lako pojaviti u nekriptiranom obliku na nekom drugom dijelu datotečnog sustava, od sva tri navedena načina enkripcija čitavog diska svakako je najsigurnija opcija. Tipičan primjer nehotičnog otkrivanja povjerljivih (kriptiranih) podataka je privremeno pohranjivanje datoteke koja se obrađuje nekom aplikacijom u `/tmp` direktorij. Ipak, enkripcija cjelokupnog tvrdog diska računala zahtjeva posebne metode podizanja operacijskog sustava koje uključuju korištenje vanjskih medija (poput diskete ili vanjske USB memorije) za pohranu jezgre sustava i "boot loader" softvera, što otežava implementaciju i korištenje takovih sustava. Također, prilikom enkripcije čitavog tvrdog diska, otežano je particioniranje tvrdog diska u više od dvije particije (najčešće `swap` i `root`). Zbog navedenih razloga, u većini slučajeva, pribjegava se enkripciji posebnih diskovnih particija ili samo određenih datoteka na tvrdom disku.

Procedura izrade kriptirane dateteke ili diskovne particije opisana u nastavku, primjenjiva je za 2.4 i 2.6 inačice Linux jezgre, s time da je kod inačice 2.6 Linux jezgre, kao i kasnijih inačica 2.4 jezgre podrška za enkripciju već ugrađena u samu jezgru i nije potrebno dodavati posebne zakrpe na izvorni kod.

Zakrpe koje dodaju podršku za enkripciju u jezgru Linux sustava moguće je dohvatiti sa adrese http://www.kerneli.org. Nakon što je programski kod zakrpe ubačen u kod jezgre pomoću naredbe `patch`, jezgru je potrebno konfigurirati i prevesti u binarni oblik. Proces prevođenja jezgre detaljno je opisan u dokumentu Kernel HOWTO, kojeg je moguće pronaći na adresi http://www.tldb.org.

Prilikom konfiguracije jezgre potrebno je obratiti pozornost na slijedeće opcije, koje su neophodne za uspješno korištenje "loopback" enkripcije:

- **CONFIG_CIPHERS** – opcija omogućuje korištenje mnoštva algoritama za enkripciju i dekripciju podataka. Ostale opcije koje započinju sa **CONFIG_CIPHER_** (iza čega slijedi ime algoritma), uključuju podršku za pojedine algoritme. Prije odluke o korištenju specifičnog algoritma za enkripciju, potrebno je proučiti uvjete njegova korištenja. Korisniku su između ostalih na odabir ponuđeni DES, Triple DES, Blowfish, Twofish, Serpent, AES i CAST5 algoritmi.
- **CONFIG_BLK_DEV_LOOP** – označava podršku za "loopback" sučelje koje se koristi u ovoj metodi enkripcije.
- **CONFIG_BLK_DEV_LOOP_USE_REL_BLOCK** – opcija omogućuje korištenje relativnog označavanja blokova podataka zapisanih na "loopback" uređaj. Iako ova opcija nema izravne veze sa enkripcijom, potrebno ju je omogućiti jer određeni algoritmi za kriptiranje podataka ovise o oznaci bloka na koji se zapisuju podaci. Bez uključivanja ove opcije, jednostavne operacije poput premještanja datoteke s jednog dijela diska na drugi promijenile bi oznaku bloka u kojem se datoteka nalazi i učinile ju nečitljivom.
- **CONFIG_BLK_DEV_LOOP_GEN** – predstavlja opciju koja omogućuje korištenje enkripcijskih algoritama sa "loopback" sučeljem. Moguće je koristiti sve algoritme prethodno omogućene **CONFIG_CIPHER_** opcijama.

Nakon što je nova jezgra prevedena i sustav ponovno podignut, moguće je krenuti u izradu kriptiranih datoteka i diskovnih particija. Za primjer, napraviti ćemo datoteku u svom korisničkom direktoriju koja će predstavljati spremište za kriptirane podatke i nazvati je "spremnik".

```
$> dd if=/dev/urandom of=~/spremnik bs=1024k count=100
```

Redak iz primjera kreirati će datoteku veličine 100 MB i ispuniti ju slučajno generiranim podacima. Na taj način neovlašteni korisnik pregledom datoteke neće biti u mogućnosti razlikovati blokove koji sadrže korisne podatke od onih koji su slučajno generirani. U slučaju da se kriptira cjelokupna

diskovna particija, ime datoteke potrebno je zamijeniti imenom particije (npr. `/dev/hda2`). U sljedećem koraku potrebno je kreiranu datoteku (ili diskovnu particiju) asocirati sa odgovarajućim "loopback" sučeljem pomoću kojeg će se vršiti upis i čitanje kriptiranih podataka. U tu svrhu koristi se `losetup` naredba, koja od korisnika zahtijeva i upisivanje zaporke koja će se koristiti za pristup podacima.

```
$>    losetup    -e    ime_enkripcijskog_algoritma    /dev/loop0
~/spremnik
Password:
```

Ovu operaciju potrebno je izvesti kao `root` korisnik, jer će u protivnom svi korisnici biti u mogućnosti pisati po odabranom "loopback" sučelju. Nakon povezivanja sučelja sa zadanom datotekom, virtualni uređaj `/dev/loop0` prividno se ponaša kao disk na koji je moguće zapisivati podatke. Zbog toga je prije zapisivanja bilo kakvih podataka potrebno formatirati uređaj, nakon čega ga je moguće montirati u stablo Linux datotečnog sustava.

```
$> mke2fs /dev/loop0
$> mount -t ext2 /dev/loop0 /mnt
```

U slučaju da su svi opisani koraci ispravno načinjeni, unutar `/mnt` direktorija, trebao bi se nalaziti datotečni sustav u koji je moguće zapisivati i čitati zapisane datoteke, koje se zapravo (u kriptiranom obliku) fizički nalaze u datoteci "spremnik". U ovom trenutku, kriptiranim podacima se transparentno pristupa putem `/dev/loop0` sučelja, što je svakako poželjno u trenutku kada korisnik ima potrebu za upisivanjem ili čitanjem kriptiranih podataka. Međutim, u trenutku kada korisniku pristup kriptiranim podacima više nije potreban, vrlo je važno ukloniti uređaj `/dev/loop0` iz stabla datotečnog sustava te odspojiti "loopback" sučelje sa kriptirane datoteke ili particije. To se postiže izdavanjem slijedećih naredbi u naredbenom retku:

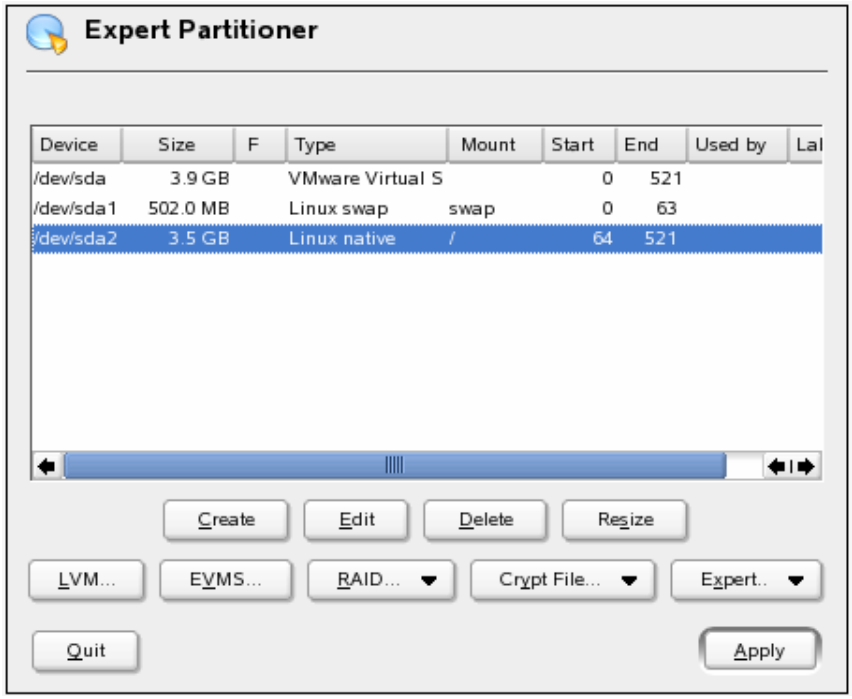
```
$> umount /dev/loop0
$> losetup -d /dev/loop0
```

U protivnom bi bilo koji korisnik sa adekvatnom razinom ovlasti mogao čitati ili zapisivati po kriptiranom datotečnom sustavu. Opisani postupak je za svakodnevnu upotrebu ipak potrebno malo pojednostaviti. To se postiže dodavanjem slijedećih redaka u `/etc/fstab` datoteku:

```
/home/user/spremnik /home/user/kriptirani_disk ext2 \
Defaults,noauto,loop,encryption=ime_enkripcijskog_algoritma,user
0 0
```

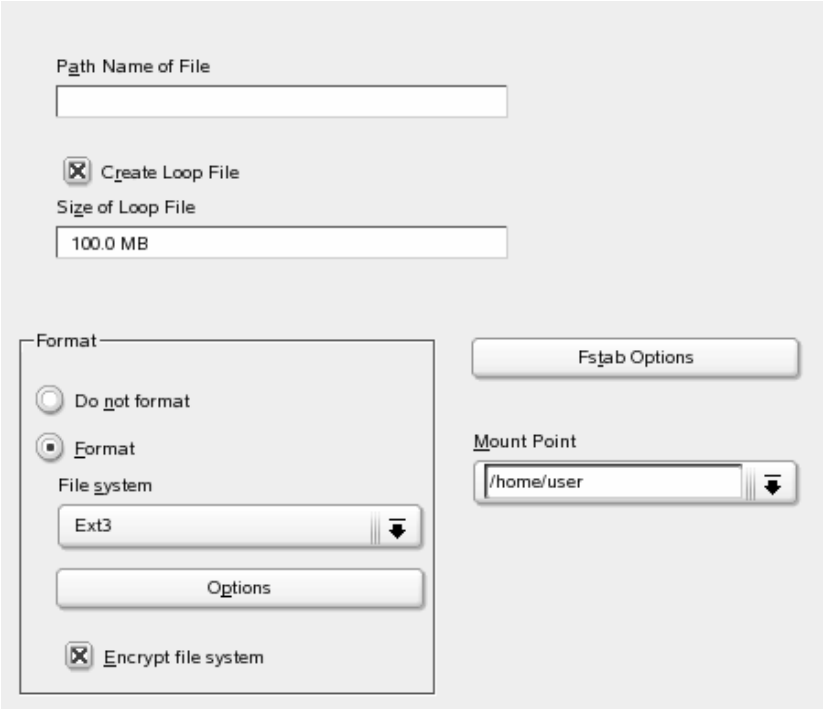
Time izbjegavamo korištenje `losetup` naredbe, te montiranje i uklanjanje kriptiranog datotečnog sustava svodimo na pozivanje `mount` i `umount` naredbi. Opcija `user` označava da i korisnici koji ne posjeduju `root` privilegije na sustavu mogu montirati kriptirani datotečni sustav.

Ovakav oblik enkripcije datotečnog sustava podržava i većina distribucija Linux operacijskog sustava, te već pri samoj instalaciji korisniku na odabir nudi mogućnost izrade kriptirane datoteke. *Slika 1* prikazuje sučelje za particioniranje diska unutar SuSE Linux distribucije, u kojem je odabirom opcije **Crypt File** moguće podesiti parametre kriptirane datoteke (*Slika 2*).



Slika 1 Sučelje za particioniranje tvrdog diska u SuSE Linux distribuciji

U prozoru koji se otvara odabirom **Crypt File** opcije, korisniku se na odabir nudi podešavanje parametara poput imena i veličine enkriptirane datoteke te točke montiranja enkriptiranog datotečnog sustava. Enkriptirani datotečni sustav je moguće formatirati kao Ext2, Ext3, RaiserFS, itd. sustav, a iz istog prozora moguće je podesiti i opcije `fstab` datoteke.



Slika 2 Prozor za podešavanje parametara "loopback" uređaja i kriptirane datoteke

Sa objavljivanjem 2.6 inačice Linux jezgre uveden je i potpuno novi koncept enkripcije datotečnog sustava nazvan `dm-crypt`. Iako je po principu rada vrlo slična "loopback" enkripciji, `dm-crypt` metoda je mnogo fleksibilnija i otpornija na greške, stoga je realno očekivati da će u skorijoj budućnosti zamijeniti "loopback" enkripciju.

3. EHD (Encrypted Home Directories)

Ukoliko je na sustavu već implementirana "loopback" enkripcija, opisana u prethodnom poglavlju, moguće je pomoću zakrpe za `login` program na jednostavan način kreirati kriptirane korisničke direktorije, bez primjene enkripcije na čitav datotečni sustav.

Pod pretpostavkom da je u jezgru sustava uključena podrška za "loopback" enkripciju i da je uspješno primijenjena zakrpa za `login` program, postupak kreiranja kriptiranog korisničkog direktorija je slijedeći:

1. Prilikom prijave na sustav bilo kojeg korisnika čiji je korisnički direktorij definiran kao `/crypt/login`, naredba pronalazi slobodno "loop" sučelje.
2. Korisnik odabire veličinu svog korisničkog direktorija, nakon čega se u `/crypt` direktoriju kreira datoteka odabrane veličine. Za ime datoteke uzima se identifikacijski broj korisnika (UID).
3. Nakon kreiranja direktorija, korisnik unosi zaporku za pristup kriptiranim podacima i odabire algoritam pomoću kojega će se podaci kriptirati.
4. Nad zadanom zaporkom primjenjuje se *hash* algoritam i novo dobiveni zapis se pohranjuje unutar datoteke `/crypt/UID.x`, gdje UID označava identifikacijski broj korisnika na sustavu. Prilikom svakog sljedećeg prijavljivanja korisnika u sustav, zaporka za pristup kriptiranim podacima usporediti će se sa onom pohranjenom u `/crypt/UID.x` datoteci i ukoliko su zaporke jednake, korisniku će se dozvoliti pristup podacima.
5. Unutar "loop" uređaja kreira se ext2 datotečni sustav, nakon čega se uređaj montira u Linux datotečni sustav. Prilikom svakog sljedećeg prijavljivanja korisnika u sustav, u ovom koraku već kreirani datotečni sustav se provjerava na eventualne pogreške.
6. Ukoliko se korisnik više puta simultano prijavi na sustav, `login` program će ga svaki puta zatražiti zaporku za pristup kriptiranim podacima, bez obzira što je "loop" uređaj montiran na datotečni sustav. Ovakav mehanizam sprječava neovlašteni pristup podacima korisnicima koji eventualno otkriju zaporku kojom se korisnik prijavljuje na sustav, ali ne znaju zaporku za pristup kriptiranim podacima. Nakon što se korisnik odjavi sa svih započetih sjednica, kriptirani datotečni sustav se uklanja iz stabla Linux datotečnog sustava.

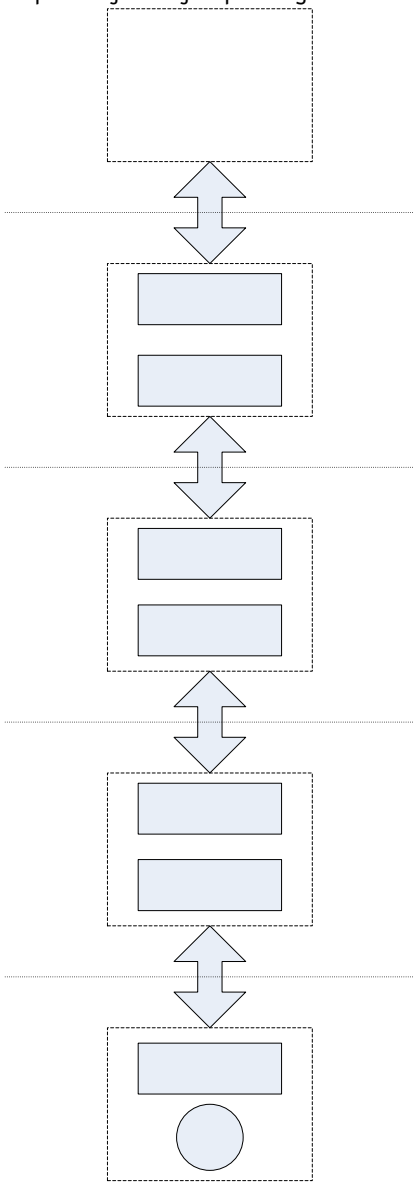
Opisani postupak predstavlja vrlo jednostavan način za enkripciju korisničkih direktorija, pogotovo kada se radi o prijenosnim računalima koje koristi više osoba. Na ovaj način pristup sustavu omogućen je svim korisnicima, ali privatnost osobnih podataka svakog pojedinog korisnika je zaštićena na odgovarajući način.

Budući da se pristup kriptiranim podacima odvija putem "loop" sučelja koje se montira u stablo datotečnog sustava, korištenje EHD enkripcije ne pruža adekvatnu razinu zaštite na računalima koja istovremeno koristi više korisnika, tj. mrežnim računalima, jer svi korisnici mogu ostvariti pristup kriptiranim podacima jednom kada se "loop" sučelje montira u datotečni sustav. Također, ovakav pristup ne podržava niti udaljeni rad, jer ne funkcionira u slučaju korištenja SSH konekcije ili `su` naredbe.

4. CFS i TCFS kriptirani datotečni sustavi

CFS je jedan od prvih besplatnih UNIX/Linux softvera za enkripciju podataka na datotečnom sustavu. Za svoj rad ovaj program koristi NFS (*Network File System*) poslužitelj što olakšava njegovu implementaciju jer nema potrebe za izmjenom jezgre operacijskog sustava. Ovakav pristup također čini CFS vrlo portabilnim jer ne ovisi o datotečnom sustavu i konfiguraciji diskova na računalu, a omogućuje korištenje enkriptiranog datotečnog sustava i udaljenim mrežnim računalima. Osnovna ideja kreatora ovog sustava je korisniku pružiti zaštitu na svim razinama koje su kritične za sigurnost informacija, uz minimalno narušavanje jednostavnosti korištenja sustava tj. potpunu transparentnost prema korisniku. CFS na izbor nudi više kriptografskih algoritama od kojih se preporuča korištenje Triple DES ili Blowfish algoritama.

Kao što je već spomenuto, sustav funkcionira na korisničkoj razini, komunicirajući s jezgrom sustava putem NFS sučelja. Svako klijentsko računalo ima pokrenutu instancu `cfsd` poslužitelja, koji interpretira korisničke zahtjeve za čitanjem i pisanjem po enkriptiranom datotečnom sustavu. Drugim riječima, korisnik prividno podatke upisuje u NFS dijeljeni direktorij, dok se oni fizički u enkriptiranom obliku pohranjuju na tvrdi disk. *Slika 3* prikazuje dizajn opisanog sustava.



Slika 3 CFS sustav za enkripciju podataka

Osnovni nedostatak CFS-a, čak i pri korištenju relativno brzih enkripcijskih algoritama, jest sporost kod operacija čitanja i pisanja podataka. Glavni razlog tome je sama izvedba ovog sustava koja pri svakoj operaciji čitanja i pisanja zahtjeva višestruko kopiranje podataka između jezgre i korisničkih aplikacija.

TCFS (*Transparent Cryptographic File System*) izravni je nasljednik CFS sustava koji se fokusira na poboljšanje performansi i bolju integraciju sa jezgrom operacijskog sustava. Nažalost, TCFS zahtjeva modifikaciju jezgre Linux operacijskog sustava, čime se gubi jedna od osnovnih prednosti CFS-a. Također, razvoj zakrpi za novije inačica jezgri vrlo je spor što također ne ide u prilog korištenju TCFS-a.

5. Practical Privacy Disk Driver

Ppdd (*Practical Privacy Disk Driver*), kao što mu i samo ime govori, predstavlja modul za Linux jezgru operacijskog sustava koji omogućuje enkripciju podataka na tvrdom disku. Ova metoda koristi kvalitetne metode enkripcije podataka, koje je moguće primijeniti na diskovima velikog kapaciteta, a istovremeno je relativno jednostavna za implementaciju. `Ppdd` nudi mogućnost enkripcije čitavog tvrdog diska, uključujući i `swap` particiju.

Na adresi http://linux01.gwdg.de/~alatham/ppdd.html moguće je pronaći zakrpe koje Linux jezgri dodaju `ppdd` funkcionalnost. Službeno su podržane inačice 2.2 i 2.4 jezgre, dok za 2.6 jezgru postoji isključivo neslužbena inačica `ppdd`-a čija se sigurnost i stabilnost ne garantira. Uz zakrpe za jezgru s iste adrese potrebno je dohvatiti i izvorni kod popratnih alata.

Nakon što se prevede i instalira nova inačica jezgre, popratne alate je potrebno prevesti naredbom `make`, te nakon toga izdati naredbe `make devices` za kreiranje sučelja za virtualne uređaje za enkripciju i `make install` za instalaciju prevedenog softvera.

Enkriptirani datotečni sustav inicijalizira se naredbom `ppddinit` koja asocira virtualni `ppdd` uređaj sa fizičkom particijom na tvrdom disku. Pri tome se od korisnika traži unos zaporke koja će se koristiti za pristup enkriptiranim podacima.

```
$> ppddinit /dev/ppdd0 /dev/hdc1
```

Naravno, ovim postupkom brišu se svi podaci koji su prethodno bili zapisani na odabranu particiju. Korištenjem opcije `-r` uz `ppddinit` naredbu, čitava particija se prilikom inicijalizacije prepisuje slučajno generiranim podacima što će neovlaštenom korisniku dodatno otežati raspoznavanje blokova sa podacima na enkriptiranoj particiji. Inicijalizirana particija aktivira se naredbom `ppddsetup` nakon čega ju je moguće formatirati i montirati u stablo Linux datotečnog sustava.

```
$> ppddsetup -s /dev/ppdd0 /dev/hdc1
$> mkfs /dev/ppdd0
$> mount /dev/ppdd0 /crypt
```

U montiranu particiju korisnik upisuje i iz nje čita podatke kao da se radi o klasičnoj diskovnoj particiji, pri čemu se podaci na fizički disk zapisuju u kriptiranom obliku. Kada korisnik završi sa radom, particiju je potrebno deaktivirati, kako bi se spriječilo da korisnici koji ne posjeduju zaporku ostvare pristup podacima.

```
$> umount /crypt
$> ppddsetup -d /dev/ppdd0
```

Zaporka koja se unosi prilikom inicijalizacije particije smatra se "master" zaporkom i praktički se koristi isključivo kod inicijalizacije sustava i u slučaju da korisnik zaboravi "radnu" zaporku. Za svakodnevni pristup kripritanim podacima koristi se tzv. "radna" zaporka, koju je moguće mijenjati na redovitoj osnovi naredbom `ppddpassw`. Na taj način smanjuje se mogućnost da neovlašteni korisnik sazna pristupnu zaporku korisnika.

```
$> ppddpassw /dev/ppdd0 /dev/hdc1
```

Što se ovlasti pristupa enkriptiranom datotečnom sustavu tiče, ove dvije zaporke su u potpunosti ravnopravne.

Korištenje dvije zaporke naročito je korisno kod sigurnosne pohrane (*backup*) kriptiranih podataka. Radnu zaporku je moguće ukloniti prije backup-a enkriptirane particije, što omogućuje pristup pohranjenim podacima isključivo administratoru koji poznaje master zaporku.

Iz svega navedenog, vidljivo je da se korištenje `ppdd`-a u osnovi ne razlikuje mnogo od "loopback" enkripcije. Kao manu ovog pristupa može se navesti korištenje vlastitog enkripcijskog algoritma, dok se "loobback" enkripcija oslanja na enkripcijske algoritme koji su uključeni u jezgru sustava.

6. Zaključak

Iako niti jedna od opisanih metoda ne zadovoljava u potpunosti sve zahtjeve koji se postavljaju pred idealan sustav za enkripciju podataka na tvrdom disku računala, svaka od njih primjenjiva je u nekim specifičnim situacijama. Svakako najprihvatljivija, najčešće korištena i najlakša za implementaciju je metoda "loopback" enkripcije na razini jezgre Linux operacijskog sustava. Implementacija ostalih metoda daleko je kompliciranija i uglavnom zahtijeva korištenje starijih inačica Linux jezgre jer se većina navedenih softvera ne održava na redovitoj osnovi.
Korištenje bilo kakvog oblika enkripcije podataka svakako utječe na brzinu čitanja i pisanja podataka, ali kada se u obzir uzme snaga modernih mikroprocesora, prosječan korisnik ne bi trebao imati ikakvih poteškoća u radu s kriptiranim datotečnim sustavom.

7. Reference

[1] Matt Blaze: „A Cryptographic File System for Unix“
[2] Giuseppe Cattaneo: „The Design of TCFS“
[3] Allan Latham: „PPDD HOWTO“
[4] Ryan T. Rhea: „Loopback Encrypted Filesystem HOWTO“
[5] David Braun: „Disk Encryption HOWTO“