HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Zaštita datoteka na Linux operacijskim sustavima

CCERT-PUBDOC-2007-05-192

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

U situacijama kada sve ostale razine zaštite računalnog sustava zakažu i kada zlonamjeran korisnik neovlašteno ostvari pristup napadnutom računalu, posljednju liniju obrane predstavlja zaštita samih datoteka. Ako ona nije pravilno izvedena, napadač može izmjenama pojedinih datoteka izvesti napad uskraćivanjem usluga (eng. *Denial of Service - DoS*) na ranjivo računalo, steći pristup potencijalno osjetljivim informacijama, ali i preuzeti potpuni nadzor nad napadnutim računalom.

Linux operacijski sustavi pružaju raznolike mogućnosti zaštite datoteka. To se prije svega odnosi na sustav upravljanja dozvolama pristupa ugrađen u ove operacijske sustave, a pomoću kojega je pojedinim korisnicima i skupinama korisnika moguće dodijeliti ili uskratiti ovlasti čitanja, izmjene i izvršavanja pojedinih datoteka ili cijelih direktorijskih struktura. Ako su potrebne naprednije mogućnosti upravljanja dozvolama pristupa potrebno je koristiti neki od programskih paketa koji implementiraju liste kontrole pristupa (eng. *Access Control List - ACL*).

Osjetljive datoteke preporučljivo je tijekom pohranjivanja i prijenosa zaštititi kriptiranjem i potpisivanjem. Ovime se napadaču otežava pristup podacima i omogućuje utvrđivanje njihove autentičnosti. Dodatnu razinu zaštite kriptiranih datoteka moguće je postići skrivanjem podataka unutar neke druge datoteke ili korištenjem tomu prilagođenog datotečnog sustava. Posljednja razina zaštite datoteka je njihovo sigurno brisanje.

Sve spomenute, ali i neke dodatne teme obrađene su u ostatku ovog dokumenta.

2. Osnovne naredbe za interakciju s datotekama

Osnovne naredbe za interakciju s datotekama su:

- `ls` (eng. *List Segmens*) – ispisuje sadržaj direktorija,
- `chown` (eng. *CHange OWNer*) – mijenja vlasnika datoteke,
- `chmod` (eng. *CHange MODe*) – mijenja značajke datoteke i
- `find` – pronalazi datoteke i direktorije.

U ovu skupinu naredbi mogu se uvrstiti i `ln`, naredba za stvaranje veza (eng. *link*), naredba `stat`, koja daje informacije o pojedinoj datoteci, te brojne druge.
Za stvaranje i održavanje datotečnog sustava koriste se:

- `fdisk` i
- `mkfs` (eng. *MaKe FileSystem*) naredbe za formatiranje diskovnih particija te
- `fsck` (eng. *FileSystem Check*) naredba za otkrivanje i uklanjanje pogrešaka datotečnog sustava.

2.1. Naredbe za dohvaćanje podataka o datotekama

Osnovne naredbe za dohvaćanje podataka o datotekama su:

- `df` (eng. *Disk Free*),
- `du` (eng. *Disk Usage*),
- `ls` (eng. *List Segments*) i
- `stat`.

Naredba `df` prikazuje iskorištenost diskovnog prostora, naredbom `df –i` prikazuje se stupanj zauzeća tzv. *inode* struktura. *Inode* je podatkovna struktura koja pohranjuje osnovne informacije o datoteci, odnosno direktoriju. U slučaju pohranjivanja velikog broja malih datoteka moguće je iskoristiti sve raspoložive *inode* strukture prije ispunjavanja diskovnog prostora. Pokušaj pohranjivanja dodatnih datoteka u takvom slučaju rezultira porukom o pogrešci kojom se korisnika izvještava o popunjenosti diska. Naredba `df` tada pokazuje kako postoji još slobodnog diskovnog prostora, ali pomoću `df –i` naredbe moguće je utvrditi iskorištenost svih *inode* struktura.
Naredba `du` prikazuje veličinu direktorija i vrlo je korisna kod racionalizacije potrošnje diskovnog prostora. Osim ispisivanja veličine datoteka i direktorija unutar trenutnog direktorija moguće je zadati direktorij naredbom oblika `du /dir/name`, a uz dodatak `–s` ova naredba prikazuje informacije u sažetom obliku.
Podatke o pojedinoj datoteci moguće je dobiti pomoću naredbe `ls` koja prikazuje imena datoteka i direktorija. Naredbom `ls –l` prikazuju se dodatne informacije kao što su dozvole pristupa, veličina datoteka idr., a `ls –la` prikazuje i skrivene datoteke čija imena započinju točkom (npr. *.bash_history* i *.bash_logout*). Uz odgovarajuće dodatke naredbu je moguće koristiti za sortiranje prikazanih datoteka prema veličini, datumu, u obrnutom poretku te na brojne druge načine.
Detaljnije informacije o pojedinoj datoteci, kao što su datum nastanka, vrijeme zadnjeg pristupa i dr., moguće je dobiti korištenjem naredbe `stat`.

2.2. Naredbe za rukovanje datotekama

Osnovne naredbe za rukovanje datotekama su:

- `cp` (eng. *CoPy*) naredba za kopiranje datoteka,
- `mv` (eng. *MoVe*) naredba za premještanje datoteka i
- `rm` (eng. *ReMove*) naredba za uklanjanje datoteka.

U ovu skupinu spadaju i naredbe za rukovanje sigurnosnim postavkama datoteka kao što su, već spomenute, `chown` i `chmod` naredbe. Sigurnost datoteka kod Linux operacijskih sustava uvelike se temelji na dozvolama pristupa. Nad pojedinom datotekom postoje tri vrste ovlasti:

- ovlasti korisnika (eng. *User*),
- ovlasti grupe (eng. *Group*) ili
- ovlasti ostalih (eng. *Other*).

Naredba chown mijenja grupu odnosno korisnika koji su vlasnici datoteke:

```
$ chown user:group object
```

gdje je *object* datoteka ili direktorij kojemu se mijenja vlasništvo.

2.3. find naredba

Naredba find koristi se za pronalaženje datoteka, a omogućuje filtriranje rezultata pretrage prema različitim kriterijima, kao što su dozvola pristupa, vlasništvo, veličina i vremenske oznake. Primjer korištenja ove naredbe za pronalaženje svih paketa s postavljenom *setuid* značajkom:

```
$ find / -perm +4000
```

ili za pronalaženje svih paketa s postavljenom *setguid* značajkom:

```
$ find / -perm +2000
```

3. Korištenje dozvola pristupa

Dozvole pristupa na Linux operacijskim sustavima moguće je postavljati korištenjem, već spomenute, chmod naredbe ili pomoću neke od implementacija lista za kontrolu pristupa (eng. *Access Control List - ACL*). Ako je npr. korisniku *Ivan* potrebno dodijeliti potpuni pristup određenoj datoteci, korisnici *Marija* dozvoliti njeno čitanje, skupini korisnika *prodaja* omogućiti izmjenu, skupini *računovodstvo* dozvoliti čitanje, a svim ostalim korisnicima potpuno uskratiti pristup istoj datoteci nužno je to učiniti pomoću ACL liste jer naredba chmod jednostavno ne omogućuje tako razrađene postavke.

3.1. chmod

Naredba chmod prvi se puta pojavila kod inačice 1 *AT&T Unix* operacijskog sustava, a njezin općeniti oblik je:

```
$ chmod [options] mode <filename> …
```

gdje su *[options]* postavke koje određuju način rada ove naredbe, *mode* su oznake izmjena koje se vrše nad datotekama navedenim u nastavku naredbe. Uobičajeno korištene postavke su:

- -R postavka se koristi za rekurzivan rad s direktorijima i datotekama te
- -v ispisuje sve datoteke nad kojima se vrše izmjene (eng. *verbose*).

Parametar *mode* sastoji se od tri znakovna niza koji se spajaju i čine jedan niz:

```
$ chmod [references][operator][mode] <filename> …
```

U prethodnom primjeru:

- oznaka *[references]* određuje na koje se korisnike odnose izmjene:
  - ***u*** – (eng. *user*) dozvole se dodjeljuju vlasniku datoteke
  - ***g*** – (eng. *group*) dozvole se dodjeljuju skupini korisnika pridijeljenih datoteci
  - ***o*** – (eng. *other*) odnosi se na sve korisnike koji nisu vlasnici datoteke niti članovi pridijeljene skupine
  - ***a*** – (eng. *all*) obuhvaća sve korisnike i ekvivalentno je oznaci ***ugo***
- oznaka *[operator]* određuje kako se vrše izmjene dozvola:
  - ***+*** – korisniku se pridjeljuju dozvole
  - ***-*** – korisniku se oduzimaju dozvole
  - ***=*** – dozvole se postavljaju na one navedene u oznaci *[mode]*

- oznaka *[mode]* određuje koje dozvole se pridjeljuju ili oduzimaju:
  - ***r*** – (eng. *read*) mogućnost čitanja datoteke ili sadržaja direktorija
  - ***w*** – (eng. *write*) mogućnost izmjene datoteke ili sadržaja direktorija
  - ***x*** – (eng. *execute*) mogućnost izvođenja datoteke ili ulaska u direktorija
  - ***X*** – (eng. *special execute*) poseban oblik ***x*** oznake
  - ***s*** – *setuid* (eng. *Set User ID*) i *setguid* (eng. *Set Group ID*) omogućuju izvođenje datoteka s višim korisničkim ovlastima od onih koje posjeduje korisnik koji pokreće datoteku
  - ***t*** – (eng. *sticky*) izmjenu sadržaja direktorija omogućuje samo njegovom vlasniku

Na primjer, skupini korisnika i vlasniku datoteke dozvole za čitanje i izvođenje datoteke pridjeljuju se naredbom:

```
$ chmod ug+rw <filename>
```

Sve dozvole uklanjaju se naredbom:

```
$ chmod a-rwx <filename>
```

kojom se svim korisnicima onemogućuje čitanje, izmjena i pokretanje datoteke imena *<filename>*. Dozvole nad datotekom postavljaju se naredbom oblika:

```
$ chmod ug=rx <filename>
```

kojom se vlasniku datoteke i datoteci pridruženoj skupini korisnika omogućuje čitanje i izvođenje datoteke. Ovlasti za izmjenu datoteke *<filename>* u ovom primjeru nema niti jedan korisnik.
Osim znakovnim nizovima *[mode]* oznaku moguće je navesti kao oktalni broj dobiven zbrajanjem odgovarajućih vrijednosti iz tablice *Tablica 1*.

Oznaka	Korisnik	Dozvola
400	vlasnik	čitanje
200		pisanje
100		izvođenje
040	skupina korisnika	čitanje
020		pisanje
010		izvođenje
004	ostali korisnici	čitanje
002		pisanje
001		izvođenje
4000	-	SUID
2000		SGID
1000		*sticky*

Tablica 1: Oktalne oznake *chmod* naredbe

Primjer tog koncepta je naredba:

```
$ chmod 0664 <filename>
```

Kojom se vlasniku i skupini korisnika dodjeljuju dozvole čitanja i izmjene datoteke *<filename>* dok se ostalim korisnicima omogućuje samo čitanje iste datoteke.

3.1.1. Zaštita dijeljenih direktorija

Ako je potrebno načiniti direktorij u kojem svaki korisnik može stvarati datoteke, ali u kojem ih samo vlasnik može mijenjati i brisati, potrebno je spomenutom direktoriju postaviti tzv. *sticky* oznaku naredbom:

```
$ chmod 1777 <dirname>
```

gdje je *<dirname>* ime dijeljenog direktorija. Bez postavljanja *sticky* oznake, odnosno postavljanjem dozvola pristupa na 0777, stvara se direktorij u kojem svaki korisnik može mijenjati i brisati sve datoteke (eng. *world-writable directory*). Navedena oznaka izmjene i brisanje datoteke omogućuje samo njezinu vlasniku.

3.1.2. Zabrana pregledavanja sadržaja direktorija

Kako bi se datoteke u određenom direktoriju zaštitile od neovlaštenog pristupa, a da ih ovlašteni korisnici istovremeno mogu nesmetano koristiti, moguće je zabraniti pregledavanje sadržaja tog direktorija naredbom:

```
$ chmod 0111 <dirname>
```

Korisnik u tom slučaju datoteci može pristupiti samo ako zna njezino ime.

3.2. ACL

Kod lista kontrole pristupa svakom objektu datotečnog sustava pridijeljen je skup pravila koja određuju koji korisnici mogu spomenutom objektu pristupiti i koje radnje su nad njim dozvoljene. Za svakog korisnika za kojega se određuju dozvole pristupa stvara se element u odgovarajućoj ACL listi koja se prema tome sastoji od minimalno tri elementa: za vlasnika objekta na kojega se odnosi, pridijeljenu skupinu korisnika i ostale korisnike. Proširene ACL liste sadrže i tzv. *mask* element koji može sadržavati proizvoljan broj imenovanih korisnika i imenovanih skupina korisnika.
Primjer ACL liste dan je tablicom *Tablica 2*.

Vrsta elementa	Tekstualni oblik
vlasnik	`user::rwx`
imenovani korisnik	`usetr::name::rwx`
pridijeljena skupina	`group::rwx`
imenovana skupina	`group::name::rwx`
maska	`mask::rwx`
ostali korisnici	`other::rwx`

Tablica 2: Primjer ACL liste

3.3. RSBAC

RSBAC (eng. *Rule Set Based Access Control*) je sustav za kontrolu pristupa (eng. *access control framework*) otvorenog programskog koda koji djeluje kao nadogradnja jezgre Linux operacijskog sustava. Osnovne značajke ovog sustava su:

- distribucija pod GPL licencom,
- veći broj sigurnosnih modela,
- detaljna kontrola nad pristupom mreži pojedinih korisnika i programskih paketa,
- upravljanje dozvolama pristupa na razini jezgre operacijskog sustava,
- mogućnost proizvoljnog kombiniranja različitih sigurnosnih modela,
- jednostavna nadogradivost i
- podrška aktualnih inačica jezgri operacijskog sustava.

RSBAC sustav građen je modularno pri čemu je uobičajeno da pojedini modul implementira jedan sigurnosni model. Sustav je moguće nadograditi korisnički kreiranim modulima. U tablici *Tablica 3* navedeni su svi raspoloživi moduli:

Naziv modula	Skraćenica	Upotreba	Kratak opis
Authenticated User	AUTH	uvijek	pruža osnovne funkcionalnosti ostalim modulima
User Management	UM	proizvoljna	na razini jezgre nadopunjuje ili zamjenjuje podsustav za upravljanje korisnicima i grupama
Role Compatibility	RC	uobičajen	upravljanje dozvolama pristupa temeljeno na ulogama pridijeljenim korisnicima
Access Control Lists	ACL	proizvoljna	određuje koji korisnik, RC uloga ili ACL grupa može pristupiti pojedinom objektu određenom vrstom zahtjeva
Mandatory Access Control	MAC	nije uobičajena	povezuje proces sa sigurnosnom razinom korisnika koji ga je pokrenuo
Pageexec	PAX	uobičajena	onemogućuje izvođenje neželjenog programskog koda
Dazuko	DAZ	proizvoljna	traženje virusa tijekom pristupa
Linux Capacities	CAP	uobičajena	upravljanje tzv. *Posix Capabilities* mogućnostima jezgre
Jail	JAIL	uobičajena	ovijanje pojedinih procesa
Linux Resources	RES	proizvoljna	upravljanje zauzimanjem resursa od strane pojedinih procesa ili korisnika
File Flags	FF	proizvoljna	podešavanje karakteristika datoteka i direktorija
Privacy Model	PM	proizvoljna	osigurava osobne podatke

Tablica 3: Moduli RSBAC sustava

4. Enkripcija i potpisivanje datoteka i direktorija

Kontrola ovlasti pristupa pruža ograničenu zaštitu podataka. Ako zlonamjeran korisnik dođe u posjed korisničke zaporke ili ako iskorištavanjem nekog sigurnosnog propusta neovlašteno stekne povišene, tzv. *root*, korisničke ovlasti imat će nesmetan pristup potencijalno osjetljivim podacima. Slična je situacija i u slučaju krađe fizičkog podatkovnog medija, npr. magnetske trake za pohranjivanje pričuvnih kopija.

Iz navedenih razloga datoteke je tijekom prijenosa i pohrane preporučeno zaštititi njihovim kriptiranjem i potpisivanjem. To je moguće učiniti pomoću neke od implementacija OpenPGP protokola koji definira enkripciju podataka, digitalne potpise i certifikate za razmjenu javnih enkripcijskih ključeva (eng. *public key*). Spomenuti protokol temelji se na *PGP Encription* (eng. *Pretty Good Privacy*) programskom paketu kojega je izvorno razvio Philip Zimmerman 1991. godine.

Izvorno je PGP enkripcija korištena za kriptiranje poruka elektroničke pošte i datoteka slanih u privitku pošte, ali su od 2002. godine razvijene različite primjene tako da PGP enkripcija danas obuhvaća:

- elektroničku poštu i privitke,
- digitalne potpise,
- potpunu enkripciju diskova prijenosnih računala,
- osiguravanje pojedinih datoteka i direktorija,
- zaštitu IM (eng. *Instant Messaging*) sjednica,
- osiguravanje podataka pohranjenih na mrežnim poslužiteljima i
- zaštitu HTTP prometa.

4.1. GnuGP

GnuPG (eng. *GNU Privacy Guard*) popularna je implementacija OpenPGP protokola. Riječ je o besplatnom programskom paketu otvorenog programskog koda koji predstavlja zamjenu PGP skupine enkripcijskih alata, uz mnoštvo dodatnih mogućnosti. Distribuira se pod GNU GPL (eng. *General Public*

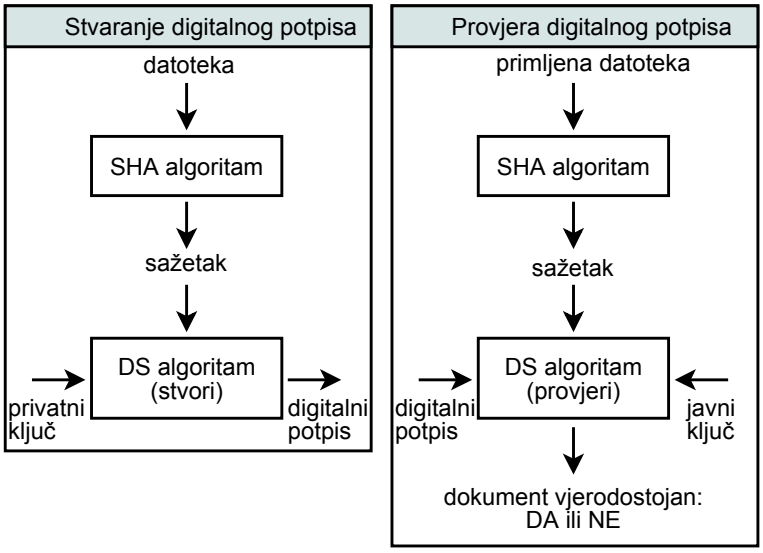
License) licencom, a u njegovu razvoju sudjelovala je njemačka vlada značajnim financijskim sredstvima. Često dolazi s besplatnim operacijskim sustavima pa je tako uključen u *FreeBSD*, *OpenBSD*, *NetBSD* te gotovo sve *GNU/Linux* operacijske sustave, a na raspolaganju su i inačice namijenjene *Windows* i *Mac OS X* sustavima.
GnuPG posjeduje tekstualno korisničko sučelje, a na raspolaganju su i brojni dodaci (eng. *front-end*) koji omogućuju njegovo korištenje putem grafičkog korisničkog sučelja. Tako, na primjer, *KMail* i *Evolution*, popularni grafički alati za rukovanje elektroničkom poštom, podržavaju GnuPG enkripciju.
Datoteke se kriptiraju pomoću asimetričnog para enkripcijskih ključeva kojega pojedinačno stvara svaki korisnik. Par ključeva sastoji se od privatnog ključa, koji je poznat samo korisniku, i javnog ključa, kojega je moguće distribuirati na različite načine, npr. putem Interneta pomoću posebnih poslužitelja. Pri tome je potreban oprez kako bi se izbjegla krađa identiteta neispravnim povezivanjem korisnika s javnim ključem.
Postupak stvaranja para enkripcijskih ključeva započinje naredbom:

```
$ gpg --gen-key
```

koja stvara *.gnupg* direktorij unutar kojega se pohranjuju ključevi, datoteke s postavkama isl. Ponovnim pokretanjem iste naredbe započinje interaktivni postupak stvaranja ključeva. Odabir predloženih postavki pruža zadovoljavajuću razinu sigurnosti, ali je u slučaju potrebe moguće izmijeniti pojedine postavke, na primjer:

- duljina ključa (eng. *key size*)
  - 1024 bita – predložena postavka
  - 2048 bita – teže je probiti zaštitu duljim ključem
- trajnost ključa (eng. *expiry*)
  - „0“ – vremenski neograničeno trajanje ključa, najčešće se odabire kod osobne upotrebe
  - kod zaštite posebno osjetljivih i vrijednih podataka preporuča se korištenje ključeva ograničenog vremenskog trajanja i njihovo izmjenjivanje, npr. mjesečno ili godišnje

Najznačajnija postavka koji je potrebno postaviti u postupku stvaranja ključeva je zaporka (eng. *passphrase*). To je znakovni niz koji bi se trebao sastojati od minimalno deset znakova te sadržavati slova (mala i velika), brojeve i interpunkcijske znakove. Ovom zaporkom kontrolira se pristup privatnom enkripcijskom ključu. Napadač koji dođe u posjed privatnog ključa može, metodom pokušaja i pogreški, doći do zaporke te provesti krađu identiteta potpisivanjem poruka pomoću kompromitiranog ključa ili pregledavati datoteke kriptirane istim ključem. Ako zlonamjeran korisnik nema pristup privatnom ključu, prisiljen ga je pogađati, što je u slučaju ključa dugog 1024 bita značajno dugotrajniji postupak od pogađanja zaporke duge deset znakova. Iz navedenih razloga privatni je ključ potrebno držati u tajnosti.
Digitalnim potpisivanjem dokumenta omogućuje se utvrđivanje njegove vjerodostojnosti, odnosno identificiranje njegova autora i dokazivanje kako nije neovlašteno izmjenjivan. U postupku stvaranja digitalnog potpisa za dobivanje sažetka datoteke koristi se sigurna jednosmjerna funkcija, tzv. SHA (eng. *Secure Hash Algorithm*) algoritam. To su funkcije koje se matematički jednostavno izračunavaju, ali im je vrlo teško pronaći inverz. Iz tako dobivenog sažetka stvara se digitalni potpis. Datoteka se, zajedno s pripadnim potpisom, šalje primaocu koji pomoću pošiljateljeva javnog ključa utvrđuje vjerodostojnost datoteke i samog digitalnog postupka. U postupku provjere potrebno je koristi SHA algoritam jednak onom korištenom prilikom stvaranja potpisa. Na slijedećoj slici shematski su prikazani opisani postupci stvaranja i provjere digitalnih potpisa.



Slika 1: Shematski prikaz postupka stvaranja i provjere digitalnog potpisa

Datoteku je pomoću GnuPG paketa moguće potpisati naredbom:

```
$ gpg –b <filename>
```

koja stvara posebnu datoteku unutar koje se nalazi digitalni potpis *<filename>* datoteke. Vjerodostojnost datoteke i pripadnog potpisa provodi se naredbom:

```
$ gpg –-verify <filename>.sig <filename>
```

gdje je *<filename>* potpisana datoteka, a *<filename>.sig* datoteka koja sadrži digitalni potpis. Ako su datoteka i potpis valjani navedena naredba rezultirati će odgovarajućom porukom, na primjer:

```
$ gpg –-verify <filename>.sig <filename>
gpg: Signature made Mon 01 Jan 2007 08:30:15 AM CET using DSA key ID
47D0D9A8
gpg: Good signature from “Hrvoje Horvat <hrvoje@mail.hr>”
```

Ako su potpis ili datoteka neovlašteno izmijenjeni provjera vjerodostojnosti neće uspjeti:

```
$ gpg –-verify <filename>.sig file
gpg: Signature made Mon 01 Jan 2007 08:30:15 AM CET using DSA key ID
47D0D9A8
gpg: BAD signature from “Hrvoje Horvat <hrvoje@mail.hr>”
```

Enkripcija datoteka provodi se jednosmjernom funkcijom pomoću korisnikova javnog ključa i rezultira naoko besmislenim skupom podataka. Korisnik pomoću privatnog ključa, koji odgovara javnom ključu korištenom za enkripciju, obnavlja izvorne podatke, odnosno dekriptira datoteku. Javni ključ korisnika kojemu se šalju kriptirane datoteke ponekada je moguće pronaći na njegovoj osobnoj web stranici (ako ju dotični korisnik posjeduje) ili na nekom od brojnih poslužitelja javnih ključeva, npr. na http://pks.aaiedu.hr/dohvat.html poslužitelju javnih ključeva kompatibilnih s OpenPGP standardom. Kako bi mogao primati kriptirane datoteke korisnik dakle treba objaviti svoj javni ključ, a do njega je moguće doći naredbom:

```
$ gpg --armor --export --output filename.asc <userID>
```

gdje je *<userID>* jedinstveni znakovni niz koji određuje korisnika i mora biti zatvoren navodnicima. Parametar *--armor* nije obavezan, a omogućuje stvaranje datoteke kodirane prema ASCII standardu koju je moguće uklopiti u elektroničko pismo ili web stranicu čime se olakšava distribucija ključa. Bez spomenutog parametra navedena naredba rezultira kriptiranom binarnom datotekom koju je jedino moguće otvoriti pomoću GnuPG paketa.
Dohvaćeni javni ključ primatelja datoteke pošiljatelj uključuje u vlastitu bazu podataka javnih ključeva (eng. *keyring*) naredbom:

```
$ gpg --import <filename>
```

Prije korištenja uvezenog javnog ključa potrebno ga je potpisati:

```
$ gpg --sign-key <userID>
```

Datoteku je tada moguće potpisati i kriptirati naredbom:

```
$ gpg --encrypt --sign --recipient <userID> <filename>
```

koja rezultira kriptiranom datotekom punog naziva *<filename>.gpg* koju pročitati može samo korisnik određen *userID* oznakom, naredbom:

```
$ gpg --decrypt <filename>
```

nakon koje je potrebno potvrditi identitet unošenjem korisničke zaporke.

5. Skrivanje datoteka

Kriptiranje datoteka može rezultirati neželjenim privlačenjem pažnje zlonamjernog korisnika. Postojanje kriptiranih datoteka napadaču može ukazati na važnost podataka koje sadrže pa napad može koncentrirati na njih. Zbog toga je kriptirane datoteke preporučeno dodatno zaštititi skrivanjem. Takav pristup zaštiti podataka skrivanjem naziva se steganografija i temelji se na činjenici da multimedijalne datoteke u pravilu sadrže neupotrijebljene ili nevažne podatkovne prostore koje različite steganografske tehnike koriste tako da ih popune tajnim informacijama. Osnovni nedostatak ovog pristupa je potreba za većom količinom nevažnih podataka kojima se prikrivaju osjetljivi podaci. Tako za skrivanje jednog megabajta podataka može biti potrebna multimedijalna datoteka velika nekoliko desetaka ili stotina megabajta.

5.1. *StegHide*

StegHide je steganografski programski paket koji omogućuje skrivanje podataka unutar zvučnih i video datoteka, a distribuira se pod GNU GPL licencom. Omogućuje skrivanje datoteke naredbom:

```
$ steghide embed -cf <multimedia_file> -ef <secret_file>
Enter passphrase:
Re-Enter passphrase:
embedding “<secret_file>” in “<multimedia_file>”… done
```

gdje je *<multimedia_file>* ime multimedijalne datoteke unutar koje se skrivaju tajni podaci, npr. JPEG (eng. *Joint Photographic Experts Group*) slikovne datoteke, a *<secret_file>* je ime tajne datoteke, npr. GPG (eng. *GnuPG*) datoteke.
Ukoliko datoteka sadrži tajne podatke skrivene pomoću ovog paketa, moguće ih je izdvojiti korištenjem sljedeće naredbe:

```
$ steghide exstract -sf <multimedia_file>
Enter passphrase:
```

```
wrote extracted data to "<secret_file>"
```

Dodatne informacije o primljenoj datoteci, bez izdvajanja tajne datoteke iz datoteke u koju je ugrađena, moguć je dobiti naredbom:

```
$ steghide info received_file.wav
"received_file.wav":
   format: wave audio, PCM encoding
   capacity: 3.5 KB
Try to get information about embedded dana ? (y/n) y
Enter passphrase:
   embedded file "secret.txt"
      size: 1.6 KB
      encrypted: rijndael-128, cbc
      compressed: yes
```

Nakon ispisa općenitih podataka o primljenoj datoteci (u gornjem primjeru *received_file.wav*) *StegHide* paket nudi ispis podataka o skrivenoj datoteci. Ako se odabere ova mogućnost i upiše ispravna zaporka skrivena datoteka (u primjeru *secret.txt*) se izdvaja te se ispisuju određeni podaci o njoj.

5.2. *StegFS*

StegFS (eng. *Steganographic File System*) datotečni sustav pruža visoku razinu zaštite datoteka objedinjujući njihovu enkripciju i skrivanje. Skrivenim datotekama može pristupiti samo korisnik koji zna ime datoteke i odgovarajuću zaporku. Napadač koji ne posjeduje spomenute informacije ne može utvrditi postojanje skrivene datoteke na napadnutom računalu čak i u slučaju stjecanja potpune kontrole nad njim.
Kako bi se osigurala tajnost skrivenih datoteka spomenuti datotečni sustav dozvoljava njihovo slučajno prepisivanje. Zbog toga je potrebno pohraniti više kopija iste skrivene datoteke kako u slučaju prepisivanja podaci ne bi bili trajno izgubljeni. Nedostaci ovakvog pristupa skrivanju datoteka su:

- potreba za većim diskovnim prostorom zbog zapisivanja višestrukih kopija,
- usporavanje rada računalnog sustava iz istog razloga, te
- mogućnost slučajnog prepisivanja svih kopija skrivene datoteke.

5.3. *OutGuess*

OutGuess omogućuje skrivanje datoteka unutar PNM (eng. *portable anymap*) i JPEG slikovnih datoteka. Na temelju podataka dohvaćenih iz zaglavlja slikovne datoteke ovaj programski paket iz nje izdvaja neiskorištene bitove, pohranjuje u njih tajne podatke i zapisuje ih natrag u istu slikovnu datoteku. Paket je moguće nadograditi tako da podržava skrivanje podataka unutar različitih formata datoteka uz uvjet da je raspoloživo zaglavlje koje opisuje strukturu takve datoteke.
Ovaj programski paket tijekom skrivanja podataka unutar slikovne datoteke ne mijenja statističke podatke o broju frekvencija izvorne slike. Posljedica toga je nemogućnost otkrivanja skrivenih podataka statističkim testovima temeljenim na brojanju frekvencija.
Datoteka se skriva naredbom:

```
$ outguess -k <passphrase> -d <secret_file> <source_img> <output_img>
```

gdje je *<passphrase>* zaporka kojom se ograničuje pristup datoteci, *<source_img>* je naziv slikovne datoteke unutar koje je potrebno skriti tajnu datoteku *<secret_file>*, a *<output_img>* je naziv rezultirajuće slikovne datoteke sa skrivenim podacima.
Za izdvajanje skrivenih podataka iz slikovne datoteke potrebno je znati zaporku korištenu prilikom skrivanja:

```
$ outguess -k <passphrase> -r <output_img> <secret_file>
```

U gornjem primjeru se tajni podaci izdvajaju iz *<output_img>* slikovne datoteke i pohranjuju u datoteku imena *<secret_file>*.

5.4. *RubberHose*

RubberHose programski paket omogućuje kriptiranje i skrivanje većeg broja posebno organiziranih cjelina podataka, tzv. aspekata, na jednom tvrdom disku ili nekom drugom podatkovnom mediju. Prije zapisivanja tajnih podataka spomenuti paket prepisuje cjelokupni medij nasumičnim znakovima koje je nemoguće razlučiti od naknadno zapisanih kriptiranih podataka.
Na primjer, ako je na tvrdom disku veličine 1 GB potrebno pohraniti dva aspekta veličina 400 i 200 MB, *RubberHose* usitnjava oba aspekta i nasumično zapisuje fragmente preko cijelog diska čime se stvara privid kako su oba aspekta veličine 1 GB. U slučaju krađe diska matematičkom analizom niti fizičkim testiranjem nije moguće utvrditi koliko aspekata je na njemu pohranjeno.
Svaki aspekt sadrži posebnu podatkovnu strukturu koja je zaštićena zaporkom, a omogućuje mu razlikovanje vlastitih podataka od nasumičnih znakova zapisanih na mediju. Pojedini aspekt posjeduje minimalne podatke o ostalim aspektima prisutnim na istom mediju kako ne bi došlo do njihova međusobnog prepisivanja. Zbog toga je prije pohranjivanja podataka na *RubberHose* disk potrebno ispravno unijeti zaporke svih prisutnih aspekata.
Ostale karakteristike *RubberHose* programskog paketa su:

- mogućnost odabira enkripcijskog algoritma (DES, 3DES, IDEA, RC5, *Blowfish*, *Twofish*, CAST),
- kompatibilnost s različitim datotečnim sustavima (UFS, ext2fs, FAT, FAT32),
- moguće je podesiti ponovno zahtijevanje zaporke nakon određenog vremena te onemogućavanje sjednice nakon perioda neaktivnosti korisnika,
- dekriptiranje jednog bloka podataka ne omogućuje neovlašteno dekriptiranje ostalih blokova i
- učestalo premještanje blokova podataka kako bi se onemogućila analiza površine diska temeljena na učestalosti uporabe pojedinih blokova.

6. Sigurno brisanje datoteka

Brisanjem datoteka nije nepovratno uništena. Obnavljanje sadržaja izbrisanih datoteka ponekad je moguće čak i slučaju njihovog prepisivanja ili formatiranja particije na kojoj su pohranjene . Kako bi se podaci potpuno uklonili s tvrdog diska potrebno je provesti neku od procedura sigurnog brisanja datoteka koje se temelje na opetovanoj izmjeni magnetskih bitova, odnosno njihovom uzastopnom prebacivanju iz vrijednosti *1* u *0* i obrnuto. Na ovaj način potpuno se uklanjaju svi tragovi datoteka izvorno prisutnih na disku.
Kako bi se osigurala dugoročna zaštita podataka poželjno je koristiti enkripciju gdje god je to moguće. Zaštita enkripcijskih ključeva i korištenje sigurnih tehnika brisanja datoteka zlonamjernom korisniku uvelike otežavaju neovlašteno pristupanje potencijalno osjetljivim podacima. Kod korištenja kombinirane zaštite enkripcijom i sigurnim brisanjem treba na umu imati mogućnost „curenja“ podataka, npr. putem privremenih datoteka pohranjenih na tzv. *swap* particijama koje nisu kriptirane.

6.1. wipe

Wipe je alat za sigurno brisanje datoteka koji između uzastopnih prijelaza preko podataka koji se uklanjaju zahtijeva mogućnost zabrane pisanja (eng. *write barrier*). Ako su dostupne, *wipe* programski paket implementira ovu zabranu pomoću *fdatasync(2)* ili *fsync(2)* funkcija. Ako spomenute funkcije nisu dostupne datoteka se otvara s postavljenom *O_DSYNC* ili *O_SYNC* oznakom.
Kako bi uklanjanje datoteke bilo uspješno pisanje preko podataka u svakom prolazu mora biti potpuno. Zbog toga tvrdi disk treba podržavati neki oblik zabrane pisanja, pražnjenje priručne memorije za pisanje (eng. *write cache flush*) ili njezino onemogućavanje (eng. *write cache disabling*).

6.2. *fwipe*

Fwipe uklanja datoteke tako što ih nekoliko puta uzastopno prepiše nulama i jedinicama (izvorna postavka je pet prepisivanja) te ih nakon toga izbriše. Ovaj paket je iznimno siguran, uspješno uklanja datoteke koje u imenu imaju posebne znakove i prikladan je za korištenje u programskim skriptama za čišćenje sustava (eng. *cleanup script*).

7. Zaključak

Linux operacijski sustavi pružaju različite mogućnosti osiguravanja datoteka. To se prije svega odnosi na upravljanje dozvolama pristupa `chmod` naredbom ugrađenom u ove sustave. Kako bi se nadišla ograničenja prisutna kod spomenute naredbe moguće je nadograditi jezgru operacijskog sustava nekom od implementacija listi za kontrolu pristupa, npr. postavljanjem RSBAC sustava koji pored ACL listi implementira i druge sigurnosne modele.

Korištenjem enkripcije datoteke je moguće zaštititi od neovlaštenog pristupa i u slučaju kada napadač stekne povišene korisničke ovlasti. Programski paket otvorenog programskog koda *GNU Privacy Guard* popularna je implementacija OpenPGP enkripcijskog protokola koja se često distribuira u sklopu besplatnih operacijskih sustava. Kriptiranu datoteku preporučeno je potpisati digitalnim potpisom čime se omogućuje utvrđivanje identiteta njezinog autora i otkrivanje neovlaštenih izmjena.

Dodatnu razinu sigurnosti kriptiranih datoteka moguće je postići njihovim skrivanjem unutar drugih datoteka ili korištenjem posebno oblikovanih datotečnih sustava. Prvi pristup naziva se steganografijom, a implementiran je kod *StegHide* i *OutGuess* programskih paketa. *RubberHose* i *StegFS* paketi, s druge strane, datoteke skrivaju na razini datotečnog sustava.

Kada osjetljive datoteke postanu suvišne potrebno ih je sigurno obrisati kako zlonamjerni korisnik ne bi naknadno obnovio njihov sadržaj. To je moguće učiniti pomoću *wipe*, *fwipe* ili nekog sličnog alata koji brisanje provode uzastopnim prepisivanjem datoteke.

8. Reference

[1] Kurt Seifried: Linux File System and File Security, http://www.seifried.org/security/index.php/Linux_File_System_and_File_Security, svibanj 2007.

[2] Wipe, http://wipe.sourceforge.net/, svibanj 2007.

[3] Fwipe, http://www.securityfocus.com/tools/1704, svibanj 2007.

[4] chmod, http://en.wikipedia.org/wiki/Chmod, svibanj 2007.

[5] Andreas Grünbacher, POSIX Access Control Lists on Linux, http://www.suse.de/~agruen/acl/linux-acls/online/, svibanj 2007.

[6] RSBAC Security Modules, http://www.rsbac.org/documentation/rsbac_handbook/security_models#access_control_lists_acl, svibanj 2007.

[7] Pretty Good Privacy, http://en.wikipedia.org/wiki/Pretty_Good_Privacy, svibanj 2007.

[8] GNU Privacy Guard, http://en.wikipedia.org/wiki/GNU_Privacy_Guard, svibanj 2007.

[9] GNU Privacy Guard Tutorial, http://legroom.net/howto/gnupg, svibanj 2007.

[10] Steghide, http://steghide.sourceforge.net/, svibanj 2007.

[11] Ross Anderson, Roger Needham, Adi Shamir: The Steganographic File System, http://www.cl.cam.ac.uk/ftp/users/rja14/sfs3.ps.gz, svibanj 2007.

[12] OutGuess, http://www.outguess.org, svibanj 2007.

[13] Steven Baum: Rubberhose review, http://iq.org/~proff/rubberhose.org/current/src/doc/review.html, svibanj 2007.

[14] Daniel J. Barrett, Robert G. Byrnes, Richard Silverman: Linux Security Cookbook, O'Reilly, 2003.