Könnyű álom (7. rész)

A 2.4-es rendszermag védelmi lehetőségei (2. rész)

Cikkünk a Linuxvilág múlt havi számában megjelent írásunk folytatása. Elolvasása előtt javasoljuk az előző rész újraolvasását. A korábbi cikk sokadszori (sajnos már elkésett) átolvasása után ismertük fel (köszönet *Kadlecsik Józsefnek* és *Kis Szabó Andrásnak*), hogy a magyarítás kissé zavaróra sikeredett. A magyar *szűrő* kifejezés olyan sok értelemben tűnt föl a cikkben, hogy avatatlan olvasó elbizonytalanodhatott. E hibát igyekszünk most kijavítani. Idézzük fel röviden az előző cikk tartalmát (most már a helyes szóhasználattal)!

A Netfilter a Linux-rendszermag része. Keretrendszer, amelyet a hálózaton közlekedő csomagok kezelésére hoztak létre. Olyan egységekből áll, ezek a hálózaton közlekedő csomagok jellemzői alapján szükség esetén annak továbbítását megtagadják, vagy fejlécében módosításokat végeznek.

Az IP Tables a Netfilter-rendszer része, segítségével az IP és az arra épülő protokollok csomagjainak módosítására nyílik lehetőség. Több részből áll, melyek feladatai bizonyos jellemzőkkel rendelkező csomagok eltávolítása, így ezzel egyfajta tűzfalszolgáltatást adva szükség esetén a csomagok forrás- vagy célcímét módosítják.

Az egyes modulok (table) végzik a csomagok tulajdonságainak tulajdonképpeni figyelését és módosítását. Minden alapmodul tartalmaz beépített szűrőláncokat, de szükség esetén a felhasználó is létrehozhat ilyet. A szűrőláncok szabályokat tartalmaznak. Egy szabály két részből áll: feltételből (ezeket az előző alkalommal áttekintettük) és műveletből. A feltételek több elemi feltételből is állhatnak, melyek logikai ÉS kapcsolatban állnak egymással. Ha minden elemi feltétel teljesül, akkor a rendszer végrehajtja azokat a megadott műveletet, amelyekről a továbbiakban szó lesz.

A rendszernek szóló utasítások

Amennyiben az adott csomag minden feltételnek megfelel, akkor a szabályban megadott művelet hajtódik végre. A rendszerben négy egyszerű beépített művelet van: a DROP, az ACCEPT, a RETURN és a QUEUE.

ACCEPT esetén az adott csomag további vizsgálatok nélkül továbbításra kerül, DROP esetén a rendszer az adott csomagot visszajelzés és újabb vizsgálatok nélkül eldobja. A RETURN művelet a csomag vizsgálatát visszaadja a hívó szűrőláncnak, a QUEUE pedig a csomagot egy felhasználói programnak küldi további feldolgozásra.

A fenti beépített műveleteken kívül két dolgot tehetünk: használhatunk kiterjesztett műveleteket, amelyek külön egységként tölthetők be, vagy az ellenőrzést egy, a felhasználó által létrehozott szűrőláncra léptethetjük. A kiterjesztett műveletek nagy részének hatását további értékekkel lehet finomítani. A további alcímek alatt a használható műveleteket mutatjuk be.

LOG-művelet

A feltételeknek megfelelő csomag megadott jellemzőit a rendszermag naplózó alrendszerén keresztül naplózza. Nekem az a véleményem, hogy az ipchains-rendszerben szintén megoldás nagyon jól használható volt, így célszerű lenne azt is támogatni. Az ipchains-rendszerben a naplózás független volt az adott szabály műveletétől, így bármely végrehajtott művelet naplózható volt. Mivel az IP Tables-rendszerben ilyen jelenleg nem létezik, az ember kénytelen először naplózni, hogy mit szándékozik tenni az adott csomaggal, és csak azután térhet rá a tényleges műveletre. Ez több szempontból is kellemetlen. Mi történik akkor, ha a naplózás és a végrehajtott művelet feltételrendszere egymástól elcsúszik (élő rendszereknél ez könnyen előfordulhat)? Így naplózzuk, hogy egy adott csomagot kitiltunk, de valójában nem tesszük meg… További felmerülő gond a beállítás átláthatóságának csökkenése. Amíg az ipchains-rendszeren a naplózás csak egy beállítás volt egy szabály bevitelénél, addig itt egy további szabály. Ez magától értetődően csökkenti az átláthatóságot. A kellemetlenségek mellett van azonban egy vitathatatlan jó tulajdonsága az új rendszernek: a finomhangolás lehetősége. Így a szokványos események más naplószinten (`log-level`) naplózhatók, mint az azonnali beavatkozást igénylők. Az esetleges hibák felderítésére a naplók kiegészíthetők a felhasználó által megadott szöveggel – így annak keletkezési helyére akár a naplóban is utalhatunk. Szükség szerint a csomagok alacsony szintű jellemzőit is naplóztathatjuk.

A lehetséges értékek az alábbiak:

`--log-level` *szint*

Ezzel a naplózási szint állítható be vele. A naplózás szintjeiről bővebb tájékoztatás kapható a naplózó alrendszer leírásánál (például *man 5 syslog.conf*).

`--log-prefix` *előtag*

A naplózott sorok előtagja határozható meg. Az előtag legfeljebb 29 betűs lehet. Nagyszerűen használható a fontosabb sorok kiemelésére és a nyomkövetésnél is jó hasznát vehetjük.

`--log-tcp-sequence`

Naplózza a TCP-sorozatszámot. Ha a felhasználók olvashatják a naplóállományt, akkor ez komoly biztonsági kockázattal jár. A naplóállományok felhasználók általi olvashatóságát nem javasoljuk, íly módon ez a gond elkerülhető.

`--log-tcp-options`

Naplózza a TCP-csomag *Options* (Tulajdonságok) mezőjét.

`--log-ip-options`

Naplózza az IP-csomag *Options* (Tulajdonságok) mezőjét.

REJECT-művelet

Mivel a beépített DROP minden visszajelzés nélkül dobja a padlóra a csomagokat, annak küldője nem tudja, hogy valami gond van az adatátvitellel. E nehézség megoldására született a REJECT-művelet, amely hasonló az ipchains-rendszerben megismertéhez, de annál jóval okosabb. Itt meghatározható ugyanis, hogyan jelezzen hibát a csomag küldőjének. Ha az ipchains-rendszer REJECT-műveletét használtuk, akkor az `port unreachable` ICMP-csomaggal utasította el a kérést. Ebből TCP-kapcsolat esetén nyilvánvalóan látszott, hogy valójában a csomagszűrő utasította el a kapcsolatot, és nem arról van szó, hogy az adott kapun nem figyel semmi. Az IP Tables REJECT-moduljával ez a gond megoldható, mindössze annyit kell tennünk, hogy a TCP-kapcsolatot RESET TCP-csomaggal utasítjuk el.

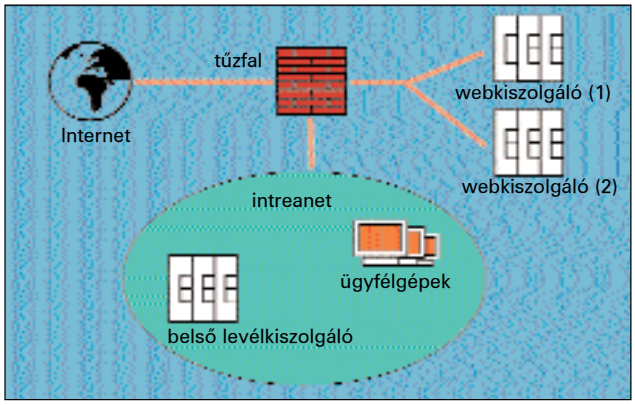
Alkalmazására csak az INPUT, FORWARD és az OUTPUT szűrőláncban vagy kizárólag az előbb említettekből hívott felhasználói szűrőláncban van lehetőség.

A válasz típusát az alábbi kapcsolóval tudjuk befolyásolni:

`--reject-with` *típus*
A lehetséges választípusok:
`icmp-net-unreachable`
`icmp-host-unreachable`
`icmp-port-unreachable` (ez az alapértelmezett)
`icmp-proto-unreachable`
`icmp-net-prohibited`
`icmp-host-prohibited`
ICMP echo request csomag esetén válaszul adható:
`echo-reply`
Csak TCP-kapcsolat esetén lehetséges válasz:
`tcp-reset`

MARK-művelet

Segítségével beállítható a csomaghoz kapcsolódó jelzés. Ezt a jelzést a későbbiekben további szűrésre használhatjuk (lásd mark feltétel) vagy `policy routing` használata esetén a csomagok előjelölésére. Ez utóbbi lehetőségről olvashatunk bővebben itt [1]. A MARK-művelet csak a mangle-modulban alkalmazható.
`--set-mark` *jelzés*



TOS-művelet

Az IP-fejléc 8-bites *Type of Service* (ToS) mezőjét állíthatjuk be a segítségével.
Használata lehetővé teszi a `policy routing` használatát összetett, több Linux-alapú útválasztóból vagy tűzfalból álló rendszereken. Csak a mangle-modulban használható.
`--set-tos` *tos*
Használhatók számok vagy a következő nevek:

- Minimize-Delay 16 (0x10),
- Maximize-Throughput 8 (0x08),
- Maximize-Reliability 4 (0x04),
- Minimize-Cost 2 (0x02),
- Normal-Service 0 (0x00).

SNAT-művelet

A forráscím átírására használható. Kizárólag a nat-modulban alkalmazható, a POSTROUTING szűrőláncban. Egyetlen kapcsolója van:
`--to-source` *IP-cím*`[-`*IP-cím*`][:`*kapu-kapu*`]`
Meghatározható vele egy IP-cím, IP-címek egy zárt tartománya, illetve lehetőség van egy kaputartomány megadására (ezt csak TCP- vagy UDP-protokoll esetén lehet használni). Ha a kaputartomány nincs megadva, akkor a rendszer az 512-es alatti kapukat lehetőség szerint a továbbítás alatt is 512 alatt tartja, az 513 és 1023 közötti kapukat 1024 alatt, az efölöttieket pedig 1023 fölött továbbítja. Amennyiben lehetséges, a kapuk a továbbítás alatt nem változnak meg.
Ha IP-címtartományt adunk meg, akkor minden kapcsolat azzal az IP-címmel tart kapcsolatot a továbbiakban, amivel a kapcsolat felépült.

DNAT-művelet

A csomagok célcímének átírását teszi lehetővé. Csak a nat-modulban használható, a PREROUTING és az OUTPUT szűrőláncokban vagy csak a belőlük hívott felhasználói szűrőláncokban. Egyetlen kapcsolója hasonló a SNAT-hoz:
`--to-destination` *IP-cím*`[-`*IP-cím*`][:`*kapu-kapu*`]`
Megadható vele egy új cél-IP-cím, egy zárt címtartomány (itt is követi a rendszer a már felépült kapcsolatokat, és a továbbiakban a kezdeti jellemzőkkel tart kapcsolatot), és lehetőség van a célkapu megváltoztatására is.
Amennyiben a célkapu nincs megadva, akkor nem változik meg.

MASQUERADE-művelet

Hasonló feladatot lát el, mint a SNAT, csak itt nem kell a kimenő IP-címnek állandónak lennie. Ez a művelet használandó tehát dinamikus IP-címkiosztású rendszereknél (például modemes behívás, ISDN, ADSL) a rendszer mögött lévő hálózat forgalmának továbbítására. A lehetőség sokak előtt ismert az ipchains-rendszerből.
A hálózati szolgáltatók őszinte bánatára itt már gyerekjáték átállítani a forráskapu-tartományt, így az ilyen címfordítást használó rendszert nem olyan egyszerű felismerni. (A szolgáltatók gyakran adnak el egygépes széles sávú szolgáltatást olcsóbban. A felhasználók szeretik olcsón megvenni az egygépes szolgáltatást, majd több gép hálózati forgalmát azon keresztül bonyolítani. Mivel az ipchains alapértelmezés szerint a 61000-es és a 65096-os kapu közül indította az ilyen kapcsolatokat, ami általában nem szokása semmilyen operációs rendszernek, viszonylag könnyű volt a turpisságot érzékelni. Természetesen a rendszermag kis módosításával a dolog áthidalható volt, de ezt nem mindenki ismerte, illetve hibát is okozhatott.
Egyébként más, kifinomultabb módszerek is léteznek az ilyen rendszer felfedezésére – ez esetleg egy későbbi cikk tárgya is lehet.
A MASQUERADE annyival több az adott kimenő IP-címen történő SNAT-nál, hogy a hálózati csatoló lekapcsolásakor önműködően elfelejti addig rögzített kapcsolatait. Így egy következő behívás nem okozhat gondot. Kapcsolója:
`--to-ports` *kapu*`[-`*kapu*`]`
Ez határozza meg a fent említett kaputartományt, felülbírálva a SNAT-nál már ismertetett eljárást. Csak TCP- és UDP-protokolloknál alkalmazható.

REDIRECT-művelet

A bejövő vagy kimenő kapcsolatok helyi kapuba való irányítását teszi lehetővé. Mi ennek az értelme? Ez teszi lehetővé átlátszó tűzfalak és proxyk létrehozását. Megadhatjuk, hogy az áthaladó webkérések legyenek egy squidba irányítva. Kapcsolója:
`--to-ports` *kapu*`[-`*kapu*`]`
A célkaput vagy kaputartományt határozza meg. Alapértelmezésben a célkapu nem változik.

Csomagszűrés a gyakorlatban

Az eddig említett lehetőségek szemléltetésére létrehoztunk egy példahálózatot. Az *ábrán* látható az elképzelt rendszer felépítése. A közepes cég logikai hálózata három alapvető részből áll: az Internetből, a cég belső hálózatából (intranet) és egy kevésbé védett, úgynevezett szabad területből (DMZ), amelyben két webkiszolgáló található. A webkiszolgálókat azért célszerű leválasztott hálózatrészbe tenni, mert ezek a legerősebben támadott rendszerek, gyakran változó tartalommal. A változó tartalmat kiszolgáló programok tervezésénél általában nem a biztonság a legfőbb szempont, így ezek sajnos gyakran válnak a betörők áldozataivá. Ha egy webkiszolgáló külön hálózatban van, akkor esetleges feltörése esetén sem különösebben veszélyezteti a belső hálózat biztonságát.

iptables.conf

```
# a nat-modul beállításai
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]

# 1. példa
-A PREROUTING -p tcp -i eth0
              ➥-m tcp --destination-port www
              ➥-j REDIRECT --to-ports 3128
-A PREROUTING -p tcp -i eth0 -m
              ➥tcp --destination-port https -j
              ➥REDIRECT --to-ports 3128
# 2. példa
#-A PREROUTING -p tcp -d <interIP> -m
               ➥tcp --destination-port www -j
               ➥DNAT --to-destination
               ➥10.0.1.11-10.0.1.12
# 3. példa
-A POSTROUTING -p tcp -o comx0 -m
               ➥tcp --destination-port telnet
               ➥-j MASQUERADE

COMMIT

# a mangle-modul beállításai
*mangle
:PREROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
COMMIT

# a filter-modul beállításai
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
:spoof - [0:0]
:interIN - [0:0]
:interLO - [0:0]
:intraIN - [0:0]
:intraLO - [0:0]
:dmzIN - [0:0]
:dmzLO - [0:0]

-A INPUT -i lo -j ACCEPT
-A INPUT -j spoof
# a saját lábunkra érkező forgalom
-A INPUT -i eth0 -d 10.0.0.1 -j intraLO
-A INPUT -i eth1 -d 10.0.1.1 -j dmzLO
-A INPUT -i comx0 -d <interIP> -j interLO
-A INPUT -j LOG --log-prefix "input catch all "
-A INPUT -j DROP

-A FORWARD -j spoof
# az átmenő forgalom
-A FORWARD -i eth0 -j intraIN
-A FORWARD -i eth1 -j dmzIN
-A FORWARD -i comx0 -j interIN
-A FORWARD -j LOG --log-prefix "forward catch all "
-A FORWARD -j DROP

################## SPOOF szűrőlánc a címhamisítás
##megakadályozására ##############################
################## Embereknek, akik nem bíznak a
##magfejlesztőkben :) ############################

-A spoof -i eth0 -s 10.0.0.0/24 -j RETURN
-A spoof -i eth0 -j LOG --log-prefix
       ➥"spoof cache all eth0 "
-A spoof -i eth0 -j DROP

-A spoof -i eth1 -s 10.0.1.0/24 -j RETURN
-A spoof -i eth1 -j LOG --log-prefix
       ➥"spoof cache all eth1 "
-A spoof -i eth1 -j DROP

# Internetről jönne nem szabályos forráscímmel
-A spoof -i comx0 -s 10.0.0.0/8 -j DROP
-A spoof -i comx0 -s 172.16.0.0/12 -j DROP
-A spoof -i comx0 -s 192.168.0.0/16 -j DROP
-A spoof -i comx0 -s 224.0.0.0/4 -j DROP
-A spoof -i comx0 -s 240.0.0.0/5 -j DROP
-A spoof -i comx0 -j RETURN
-A spoof -j LOG --log-prefix "spoof catch all "
-A spoof -j DROP
COMMIT

############### A lábakra bejövő és az áthaladó
##forgalom irányítása ###########################

# if_intranet helyi lábra érkező forgalma
# 4. példa
-A intraLO -p tcp -s <in_mail> -m
           ➥tcp --source-port 1023:65535
           ➥--destination-port smtp -m mac
           ➥--mac-source <in_mail:mac_addr>
           ➥-j ACCEPT
-A intraLO -p tcp -m tcp --source-port
           ➥1023:65535 --destination-port
           ➥3128 -j ACCEPT
-A intraLO -p udp -m udp --source-port
           ➥1023:65535 --destination-port
           ➥domain -j ACCEPT
-A intraLO -p udp -m udp --source-port ntp
           ➥--destination-port ntp -j ACCEPT
-A intraLO -j LOG --log-prefix
           ➥"intraLO catch all "
-A intraLO -j DROP

# if_intranet átmenő forgalma
-A intraIN -p tcp -m tcp --destination-port ftp
           ➥-j ACCEPT
-A intraIN -j LOG --log-prefix
           ➥"intraIN catch all "
-A intraIN -j DROP

# if_internet helyi lábra érkező forgalma
-A interLO -p tcp -m tcp --source-port
           ➥1023:65535 --destination-port
           ➥smtp -j ACCEPT
```

A rendszer határvédelmi tervét a Linuxvilág február-márciusi számának 58. oldalán találjuk. A táblázat a korábban tárgyalt [4.] módszerrel írja le, hogy a rendszer egyes részei milyen erőforrásokhoz férhetnek hozzá. Ne felejtsük el: ennek a példarendszernek egyetlen feladata a csomagszűrés lehetőségeinek bemutatása – a gyakorlatban kissé finomabban hangolt védelmi rendszerekre van szükség.

A csomagszűrő betöltése

A csomagszűrő beállításait kétféle módszerrel érdemes az operációs rendszer elindulása után visszatölteni. Az egyik módszer szerint írunk egy parancsfájlt, amely `iptables` parancsok kiadásával egyenként beviszi az előre elkészített csomagszűrő-beállításokat. Egy ilyen parancsállomány tartalmazhat további beállításokat (a szükséges rendszermagmodulok betöltése, `sysctl`-beállítások stb.). Több ilyen letölthető az Internetről, ahol csak a csomagszűrő hangolását kell elvégeznünk.

Elegánsabb módszer az `iptables-save` és az `iptables-resore` programok használata. Ezek segítségével egy memóriában beállított csomagszűrő egy állományba menthető, illetve onnan visszatölthető. Mivel formátuma meglehetősen egyszerű, akár eleve ebben a formátumban megírhatjuk a csomagszűrő beállítóállományát, így pedig az iptables-restore programmal könnyen betölthető. Mi az utóbbi lehetőség mellett döntöttünk. Beállítóállományunk a *listán* látható.

Egy dolgot érdemes megemlíteni ezekkel a parancsokkal kapcsolatban. Képesek mind a szűrőláncokhoz, mind az egyes szabályokhoz tartozó csomag- és bájtszámlálók mentésére és visszaállítására.

A rendszer szűrési lehetőségei

A 4. példa összetettebb szűrést mutat be. A cél az, hogy az intranetről közvetlenül kizárólag a belső levelezőlkiszolgáló tudjon levelet továbbítani a rendszeren keresztül. Ennek érdekében a rendszer MAC-címét is felvettük a feltételek közé. A megoldással kapcsolatban két gond merülhet fel. Egyrészt a hálózati kártyák MAC-címeit programból át lehet állítani, így a levelezőkiszolgáló leállása esetén a helyére állhat valaki. A másik gond az, hogy ez a megoldás csak a helyi ethernethálózaton működik. Tetszőlegesen bonyolult szűrések állíthatók össze a beépített és a kiterjesztett feltételek alkalmazásával.

Példaműveletek

A levéltovábbító protokoll (SMTP) egy sajátosságán keresztül mutatjuk be az egyik legjobban használható műveletet, a REJECT-et. Levél továbbításakor a levelezőkiszolgálók egy része a feladó gép *auth*-kapujához fordul. Erről korábban már volt szó [4.], ott azonban a rendszer a *port-unreachable* ICMP-üzenettel tért vissza, mivel az ipchains REJECT művelete nem volt hangolható. Az üzenet miatt a figyelmes szemlélő azonnal rájöhetett, hogy csomagszűrő utasította el, és az adott szolgáltatás bizonyos feltételek mellett akár el is érhető. Mivel az IP Tables által adott REJECT-műveletnek meg lehet mondani, hogy mit adjon vissza, a rendszer tökéletesen tud szimulálni egy olyan gépet, amelyen az adott kiszolgálóprogram valóban nem figyel. Ezt láthatjuk az 5. példában. (➲ http://www.linuxvilag.hu/konnyualom)

Mint azt már korábban is említettük, a naplózás továbbfejlődött az ipchainsben megszokotthoz képest, ez azonban néhány kellemetlenséget is magával hozott.

Minthogy a LOG művelet nem köthető más valóban érdemi műveletekhez, a naplózni kívánt csomagok két szabály bevezetését kívánják meg az IP Tables-rendszerben (6. példa ➲ http://www.linuxvilag.hu/konnyualom). Ha az ilyen érdemi változást nem okozó műveletek összevonhatók lennének a valóban tiltó vagy engedélyező műveletekkel, akkor a beállítások kissé átláthatóbbak lennének.

A címátírás használata

Első három példánk olyan címfordítási helyzeteket mutat be, amelyek gyakran előfordulnak a gyakorlatban. Az 1. példa egy átlátszó webgyorstár csomagszűrő részét mutatja be. A csomag a belső hálózat felől érkezik és egy Interneten lévő webkiszolgálóhoz megy. Mivel a hálózat terheltségét erősen csökkentheti egy webgyorstár, de minden gépen beállítani sok munkát jelentene, célszerű valamilyen átlátszó megoldást létrehozni. Az IP Tables nat-modulja a PREROUTING szűrőláncban átírja a csomagok célcímét a helyi gép címére, a célkaput pedig 3128-ra. Így a kapcsolatot a helyi webgyorstár szolgálja ki.

A 2. példa terheléselosztást mutat be a DMZ-területen lévő két webkiszolgáló között. Az Internetről bejövő kapcsolatot a rendszer az egyik webkiszolgálónak adja és követi a kapcsolatot annak bontásáig. Így az egyszer felépült TCP-kapcsolathoz tartozó csomagokat minden esetben a megfelelő kiszolgáló kapja meg. A harmadik példaként kiemelt sor pedig az IP-álcázás (masquerading) használatát mutatja be.

Miközben ismerkedtünk a csomagszűrő rendszer sajátosságaival, néhány alul- vagy nem dokumentált dolog is kiderült. A nat-modul használata során a csomagszűrésnél az alábbiak igazak: DNAT és REDIRECT használata esetén a filter-modul szűrőláncai a módosított célcímet kapják meg, az SNAT és MASQUERADE műveleteknél azonban a filter-modul szűrőláncaiban az eredeti forrás látszik.

A címátírás használata esetén a tcpdump (bővebben lásd a következő számban) használatával érdekes dolgokat tapasztalhatunk… Az átírt csomagok kimenő és bejövő jellemzői nem minden esetben egyeznek meg, így előfordulhat, hogy a célcím-átírás után a tcpdump nem tudja a kapcsolatot egységként látni.

Összegzés

Mindent egybevetve az IP Tables rendkívül hatékony csomagszűrő rendszer. Hely hiányában csak egy kis ízelítőt tudtunk nyújtani lehetőségeiből, de nem is a kimerítő tárgyalás volt a cél, hanem ezeknek a puszta bemutatása. Az IP Tables és a 2.4-es rendszermag más hálózati szolgáltatásainak segítségével szinte kimeríthetetlen lehetőségek állnak a rendelkezésünkre.

Hivatkozásjegyzék

[1.] *Bert Hubert*, *Gregory Maxwell*, *Remco van Mook*, *Martin van Oosterhout*, *Paul B. Schroeder* és mások: Linux 2.4 Advanced Routing HOWTO

[2.] *Paul 'Rusty' Russel*: Linux 2.4 Packet Filtering HOWTO ➲ http://netfilter.samba.org/

[3.] *Paul 'Rusty' Russel*: Linux 2.4 NAT HOWTO ➲ http://netfilter.samba.org/

[4.] Linuxvilág (II évf. 2-3. szám, 54. oldal): Könnyű álmok (4. rész) A hálózati határvédelem alapjai

Mátó Péter (atya@andrews.hu), informatikus mérnök és tanár. Biztonsági rendszerek ellenőrzésével és telepítésével, valamint oktatással foglalkozik. 1995-ben találkozott először linuxos rendszerrel. Ha teheti, kirándul vagy olvas.

Borbély Zoltán (bozo@andrews.hu), okleveles mérnök-informatikus. Főként Linuxon futó számítógépes biztonsági rendszerek tervezésével és fejlesztésével foglalkozik. A 1.0.9-es rendszermag ideje óta linuxozik. Szabadidejét barátaival tölti.