Linux mint Ethernet híd

Úgy adja tovább a csomagokat, mint egy híd, és úgy szűri őket, mint egy tűzfal. Ha a szükség úgy hozza, bármilyen kiszolgálót vagy készüléket külön védelemmel láthatunk el úgy, hogy a beállításain semmit nem kell módosítanunk.

Kértek valaha arra, hogy erősítsük meg egy olyan forgalomirányító védelmét, amelyhez nincs rendszergazdai hozzáférésünk? Mit lehet tenni, ha nem mi felügyelünk egy hálózatot, de erőteljesebb védelmet akarunk az általunk használt szegmensre? Egy ehhez hasonló kérés kapcsán ismerkedtem meg a ***Bridge (híd)*** világával, a linuxos ***Ethernet*** híd kifejlesztésére irányuló tervezettel.

A ***Bridge*** webhelye szerint:
Az ***Ethernet*** áthidalás egyike a több hálózat összekapcsolásával egy nagyobb hálózat kialakítására alkalmas megoldásoknak. Az áthidalásra vonatkozó szabvány az ***ANSI/IEEE 802.1d***. Híd segítségével két különböző hálózati szegmenset tudunk protokollfüggetlen módon összekötni. A csomagok továbbítása ***Ethernet***-cím alapján történik, és nem ***IP***-cím alapján, ahogy a forgalomirányítók esetében. Mivel a továbbítás a második rétegben folyik, a hídon bármely protokoll átlátszó módon haladhat át.

A kód karbantartását jelenleg ***Stephen Hemminger*** végzi, a 2.4-es és a 2.6-os rendszermaghoz egyaránt. Az áthidaló kód a legtöbb korszerű, 2.6-os rendszermagra épülő terjesztésbe be van építve. Írásom alapjaként a ***Fedora Core 3*** szolgált, mely szintén a 2.6-os rendszermagra épül. Aki leragadt a 2.4-es rendszermagnál, annak sem kell kétségbe esnie, a ***Bridge*** webhelyén (lásd az internetes forrásokat) ugyanis ehhez is találhatók foltok.
A tűzfalhíd, vagy, ha úgy tetszik, áthidaló tűzfal tűzfalösszetevőjét egy másik tervezet, az ***ebtables*** keretein belül fejlesztik. Az ***ebtables*** egy szűrőréteg az áthidalást is végző tűzfalakhoz. A szűrés az adatkapcsolati rétegbeli Ethernet keretek tartalma alapján történik. A szűrés mellett az ***Ethernet MAC***-címek módosítására is van lehetőség. Az ***ebtables*** kód révén ***iptables*** szabályokat is lehet áthidaló módban használni, így a tűzfalat ***IP***- és ***MAC***-szintű szűrőkkel egyaránt elláthatjuk.

Mi az a híd?

A híd kettő vagy több, azonos hálózati technológiát alkalmazó hálózati szegmenset összekapcsoló készülék. A szegmensek topológiája lehet eltérő, akár optikai szálas és rézkábeles szakaszokat is összekapcsolhatunk, ám a technológiának azonosnak kell lennie. A legegyszerűbb az, ha a hidat linuxos hubként szemléljük. A géphez annyi kaput adhatunk hozzá, amennyit csak akarunk, mindegyik ugyanannak a hubnak lesz a része. Ami az egyik kapun beérkezik, az az összes többin kimegy, hacsak nem mást írunk elő a szabályokban. Ha a hub működik, ***iptables*** és ***ebtables*** szabályokkal pontosan úgy szűrhetjük a forgalmat, ahogy azt bármely más linuxos forgalomtovábbító rendszeren is megtehetjük.

A kezdés

Első lépésként vegyünk egy két hálózati csatolóval ellátott számítógépet.

1. kódrészlet A hálózat beállításainak megadása előtt ellenőrizzük, hogy mindkét Ethernet csatoló működik-e

```
#> ifconfig
eth0  Link encap:Ethernet  HWaddr 00:CC:D0:99:EB:26
      inet6 addr: fe80::2b0:d0ff:fe99:eb26/64 Scope:Link
      UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
      RX packets:86208855 errors:0 dropped:0 overruns:63 frame:0
      TX packets:77098217 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:3871506445 (3692.1 Mb)  TX bytes:266311184 (253.9 Mb)
      Interrupt:5 Base address:0xec00
eth1  Link encap:Ethernet  HWaddr 00:CC:03:D8:3A:1A
      inet6 addr: fe80::201:3ff:fed8:3a1a/64 Scope:Link
      UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
      RX packets:77087614 errors:0 dropped:0 overruns:0 frame:0
      TX packets:85110321 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:264995582 (252.7 Mb)  TX bytes:3672580334 (3502.4 Mb)
      Interrupt:9 Base address:0xec80
```

2. kódrészlet Két egyszerű beállító fájl IP-címmel nem rendelkező hálózati csatolókhoz

```
/etc/sysconfig/networking-scripts/ifcfg-eth0:
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
/etc/sysconfig/networking-scripts/ifcfg-eth1:
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=static
```

Mire végzünk, linuxos gépünk normál hubként fog működni, kapui között az igényeink szerint továbbítva a forgalmat. Ha az egyik csatolót a megszokott fali aljzatba csatlakoztatjuk, a másikhoz pedig hozzákötünk egy hordozható gépet, akkor úgy tudjuk majd róla használni a hálózatot, mintha közvetlenül az aljzatba dugtuk volna.

Célunk az, hogy a híd minden hozzá csatlakozó készülék számára átlátszó legyen. Érdekességként megjegyezném, hogy a hídnak nem muszáj ***IP***-címet adnunk, hacsak nem akarunk távolról csatlakozni hozzá, például karbantartás vagy a naplók áttekintése céljából. Természetesen manapság nem sok értelme van lemondani az ***IP***-címről, és ezt mi sem fogjuk megtenni.

A hardver esetemben egy régebbi gép volt – éppen egy hasonló feladatra várt. Processzora egy ***AMD K6-450*** volt, memóriája pedig 256 MB-nyi. Egyetlen 15 GB-os ***IDE*** merevlemez volt benne, valamint egy ***3Com*** 10/100-as ***Ethernet*** kártya. Volt egy másik ***3Com*** 10/100-as kártyám is, és mivel ezek elég jól működnek ***Linux*** alatt, ez lett a második csatoló. A hídprogramon túl néhány egyszerű tűzfalszabályt akarok csak futtatni, illetve behatolásészlelésre még talán a ***Snortot***. A forgalom várhatóan kicsi lesz, és valószínűleg a ***Snort*** sem fog nagy adatmennyiségeket kezelni, vagyis a 256 MB memóriának elégnek kell lennie. Ha valaki gigabites forgalmat akar kezelni, esetleg valós idejű szimatolást akar végezni, az értelemszerűen erősebb gépet lesz kénytelen üzembe helyezni.

Ha készen áll a gép, telepítsük fel a ***Fedora Core 3***-at, illetve az általunk igényelt kiegészítő programokat. Ha a lehető legmagasabb szintű biztonságot kívánjuk elérni, törekedjünk a lehető legkevesebb szoftver telepítésére. Ha később mégis szükségünk támadna valamire, a ***YUM*** segítségével könnyedén telepíthetjük. Egyelőre elégedjünk meg egy egyszerű ***Linux***-telepítéssel, illetve ellenőrizzük, hogy felismerte-e a hálózati csatolókat. Az ***ebtables*** kód lefordításához szükség lesz a rendszermag forrására és a szokásos fordítóeszközökre is, vagyis ezeket ne hagyjuk ki. Ha a gép telepítésével végeztünk, és elhelyeztük termelési környezetébe, ne feledjük majd eltávolítani róla a felesleges szoftvereket. A telepítés befejeződése után indítsuk újra a gépet, majd jelentkezzünk be rootként.

Most létre kell hoznunk egy virtuális hálózati eszközt. Annak nevezzük el, aminek akarjuk, nálam br0 lett, mint az első ***bridge***, híd eszköz:

```
#> brctl addbr br0
```

Futtassuk le az `ifconfig`-ot. Láthatók a hálózati csatolók? Az *1. kódrészletben* látható, hogy két hálózati csatolónk van, és nincs ***IP***-cím rendelve hozzájuk. Aki esetleg már rendelt ***IP***-címet a csatolókhoz, az az egyszerűség kedvéért most törölje azt. ***Fedora*** alatt a ***/etc/sysconfig/networking-scripts/ifcfg-X*** fájlt kell átírni, ahol az ***X*** a csatoló azonosítója. Az én rendszeremen a két csatoló az `eth0` és az `eth1` volt, az ezekhez tartozó, ***IP***-címet tartalmazó sorokat kellett törölni vagy megjegyzésbe tenni. Fontos, hogy rendszerindításkor mindkét csatoló feléledjen. A *2. kódrészlet* egy alapszintű beállításegyüttest tartalmaz, ennek jól kell működnie.

Ha a fentiekkel végeztünk, ne feledjük el újraindítani a hálózatkezelést; ezt a `service network reload` paranccsal tehetjük meg.

Ez után – az alábbi módon – közöljük a rendszerrel, hogy mely eszközök tartoznak egy csoportba. Az utolsó sorral egyben a virtuális eszközt is üzembe helyezzük:

```
#> brctl addif br0 eth0
#> brctl addif br0 eth1
#> ip link set br0 up
```

Linuxos gépünk ettől a pillanattól kezdve alapszintű hubként működik. Aki nem bírja a kíváncsiságot, csatlakoztassa a csatolókat az ***Ethernet*** hálózatra, majd próbálja ki az új hubját. Természetesen a gép egyelőre vakon továbbítja a forgalmat, és nem rendelkezik ***IP***-címmel sem. Én szeretem, ha telepítés után távolról is tudok csatlakozni a gépeimhez, ezért hozzárendeltem egy ***IP***-címet a `br0` csatolóhoz, illetve forgalomirányítási adatokat is csatoltam.
A hídcsatoló ***IP***-címét a következő paranccsal lehet megadni:

```
#> ip addr add 10.1.1.18/16 brd + dev br0
```

A cím mellett meg kell határozni az alhálózati maszkot is (/16), illetve azt, hogy melyik hídeszközhöz kell hozzárendelni. Ennek elsősorban akkor van szerepe, ha egynél több virtuális eszközünk is van a gépben. Igaz, esetünkben csak egy ilyen volt, ám a parancs írásmódja szükségessé tette ennek az egynek a kiválasztását. Aki más nevet adott a hídnak, annak itt azt a nevet kell használnia.
A híd távoli elérésének lehetővé tételéhez már csak a forgalomirányítást kell elintézni:

```
#> route add default gw 10.1.1.1 dev br0
```

Itt a szokásos forgalomirányítási parancsokat és szabályokat alkalmazhatjuk, és a `br0` eszközt is pontosan úgy használhatjuk, mint az összes többi linuxos hálózati csatolót.

Tesztelés

Minden a helyén van, próbáljuk ki a rendszert. Először ellenőrizzük, hogy az összes beállítás rendben van-e:

```
#> brctl show
bridge name   bridge id          STP enabled    interfaces
br0           8000.0030843e5aa2  no             eth0
                                                eth1
```

Mint látható, egyetlen `br0` nevű hídeszközünk van, amely az `eth0` és az `eth1` csatolót használja. A jelek szerint minden rendben van.

Telepítés

Ideje nekilátnunk a fizikai telepítésnek. Az egyik hálózati csatolót csatlakoztassuk a hálózati kapcsolóhoz, pontosan úgy, mint ahogy bármilyen más számítógéppel tennénk. Az összeköttetés mindkét végén ki kell gyulladniuk a jelzőfényeknek. Egy keresztkötésű kábellel csatlakoztassunk egy asztali vagy hordozható gépet a linuxos gép másik csatolójához. Várjuk meg a jelzőfények bekapcsolását, számoljunk el tízig, majd a most csatlakoztatott gépről pingeljük meg a hálózat valamelyik gépét. A linuxos hub túloldalán található hálózatot úgy kell látnunk, mintha közvetlenül csatlakoznánk hozzá.

Túlélni az újraindításokat

Az, hogy a rendszert hogyan vesszük rá az újraindítások túlélésére, a mi választásunk. Talán a legegyszerűbb az, ha az eddig kiadott parancsokat hozzáadjuk a rendszerindítás végén lefutó */etc/rc.local* fájlhoz, ekkor a híd indítás után azonnal működőképes lesz.

Tűzfal emelése

Minden hálózati forgalmat továbbító vagy átengedő linuxos rendszer esetében lehetőségünk nyílik az áthaladó adatfolyam szűrésére. Az áthidaló tűzfalak esetében sincs ez másként. A tűzfalszabályok létrehozására és karbantartására több lehetőség is kínálkozik. A legegyszerűbb tűzfal az, amikor tiltunk mindent, kivéve azt, amit kifejezetten engedélyeztünk – az alábbiakban egy ilyen összeállítását ismertetem.
A tűzfal beállításainak megadásához le kell töltenünk és telepítenünk kell a felhasználói térben futó ***ebtables*** segédeszközöket, melyeket az ***ebtables*** webhelyéről érhetünk el (lásd a forrásokat). Írásom születésekor a legújabb változat a 2.0.6-os volt. A számos tükröoldal valamelyikéről töltsük le, majd a kezdő `configure` lépés kihagyásával végezzük el a szokásos kibontás-telepítés eljárást:

```
#> tar -xzf ebtables-v2.0.6.tar.gz
#> cd ebtables-v2.0.6
#> make
#> install
```

Ha minden rendben ment, már használhatjuk is az ebtables parancsot. Gépeljük be a parancssorba az `ebtables-t`, ennek hatására valami ilyennek kell megjelennie:

```
#> ebtables -V
ebtables v2.0.6 (November 2003)
```

Először azt kell biztosítani, hogy az ***iptables*** fogadásra legyen állítva. Mivel ***Fedora Core 3*** alatt dolgozunk, ehhez elég csak leállítanunk a szolgáltatást:

```
#> service iptables stop
#> chkconfig --level 35 iptables off
```

Hasonló hatást érhetünk el a `flush` (kiürítés) parancs alkalmazásával is. Listázzuk ki a meglévő láncokat, majd ürítsük ki őket:

```
#> iptables -L
#> iptables -F INPUT
#> iptables -F OUTPUT
#> iptables -F FORWARD
#> iptables -F RH-Firewall-1-INPUT
```

Most a tűzfalon áthaladó forgalmat teljes egészében le kell tiltanunk, érkezzen az a hálózat bármelyik részéről is. Az alábbi szabályok csak a jelenlegi példahálózatban használhatók, ha más környezetben akarjuk alkalmazni őket, akkor az alhálózatokat és az állomásokat értelemszerűen módosítanunk kell:

```
/sbin/ebtables -A FORWARD -p IPv4 \
--ip-source 10.2.0.0/16 -j DROP
/sbin/ebtables -A FORWARD -p IPv4 \
--ip-source 10.7.0.0/16 -j DROP
/sbin/ebtables -A FORWARD -p IPv4 \
--ip-source 10.4.0.0/16 -j DROP
```

```
/sbin/ebtables -A FORWARD -p IPv4 \
--ip-source 10.5.0.0/16 -j DROP
/sbin/ebtables -A FORWARD -p IPv4 \
--ip-source 10.6.0.0/16 -j DROP
/sbin/ebtables -A FORWARD -p IPv4 \
--ip-source 10.1.0.0/16 -j DROP
```

Aki szerzett már gyakorlatot az ***iptables*** használatában, annak ismerős lesz az írásmód. Közöljük az ***ebtables*** programmal, hogy amikor továbbít valamit (FORWARDING) az ***IPv4*** protokoll használatával, akkor dobjon el (DROP) minden a 10.2.0.0/16 alhálózatból származó csomagot. A többi paranccsal ugyanezt írjuk elő a többi alhálózathoz is.
A következő lépés magának a tűzfal mögött lévő eszköznek az engedélyezése. Ha ennek IP-címét nem vesszük be a továbbításra engedélyezettek közé, akkor semmi nem fog működni. Ne feledjük, ha maga a tűzfal is kap IP-címet, akkor ezt is engedélyeznünk kell:

```
/sbin/ebtables -I FORWARD 1 -p IPv4 \
--ip-source 10.1.1.5 -j ACCEPT
/sbin/ebtables -I FORWARD 1 -p IPv4 \
--ip-source 10.1.1.18 -j ACCEPT
```

Az alábbiakkal a hordozható gépem elérésére jogosult hálózati eszközöket adtam hozzá:

```
/sbin/ebtables -I FORWARD 1 -p IPv4 \
--ip-source 10.1.10.30 -j ACCEPT
/sbin/ebtables -I FORWARD 1 -p IPv4 \
--ip-source 10.1.10.19 -j ACCEPT
/sbin/ebtables -I FORWARD 1 -p IPv4 \
--ip-source 10.1.10.87 -j ACCEPT
```

Ellenőrzésképpen elsétáltam ahhoz a géphez, amelynek címét a fenti ACCEPT (elfogadás) szabályokban megadtam, majd megpróbáltam megpingelni a 10.1.1.5 címet használó hordozható gépemet. Most üljünk át egy másik, a fentiekben nem szereplő csomóponthoz – a `ping` nem fog működni.

Megvalósítás éles környezetben

Nemrég elhívtak egy felhasználó telephelyére, hogy javítsam fel egy pénzügyi kiszolgáló védelmét. A kérés egyszerű volt: kell egy tűzfal a rendszer elé, de az *IP*-címét nem lehet megváltoztatni. Két hálózati kártyával és ***Linuxszal*** felszerelkezve néhány perc alatt megoldottam a feladatot. A telepítés sem okozott gondot. A tűzfalat és a kiszolgálót egy keresztkötésű kábellel kapcsoltam össze, a tűzfal másik csatolója és a fali aljzat közé pedig normál kábel került. Ennyi volt az egész. Nem kellett újratervezni az *IP*-címzési séma egyik részét sem, egyszerűen csak össze kellett dugni a gépeket. Miután megírtam néhány szabályt, amelyek szerint az összes csomagot el kellett dobni, kivéve az engedélyezett *IP*-címekről és kapukról érkezőket, a munkát befejezettnek nyilváníthattam.
A ***Linux*** egyik szépsége, hogy egyszerre akár nagy számú szolgáltatást is képes futtatni. Vegyük a fenti példát. Adva volt egy érzékeny kiszolgáló, ami elé rövid idő alatt fel tudtam állítani egy tűzfalat. A ***Linux*** révén kapott idő- és pénzbeli megtakarításon túl azonban arra is módunk nyílt, hogy ***Snortot*** futtassunk a tűzfalon. Elég volt egy pillanatra belenyúlni a szimatolóprogram beállító fájljába (*/etc/snort.conf*), közölni vele, hogy a `br0` csatolón vizsgálódjon, és a ***Snort*** azonnal megkezdte munkáját az áthidaló csatolón.
Ezen a ponton lehetett ráérezni az áthidaló kód igazi erejére. A hálózat egyik szegmense feltűnően lassú, de nem tudni miért – ismerős a helyzet? A következő alkalommal elég fogni egy linuxos gépet, feltenni rá a ***Snortot*** és a többi felderítőprogramot, majd üzembe helyezni a hidat. Fogjunk tehát egy keresztkötésű kábelt, és már indulhatunk is a helyszínre. Mivel a híd hubként viselkedik, a linuxos gépet a hálózat tetszőleges pontjára beiktathatjuk. Elég a fizikai kapcsolatokat biztosítanunk, a gépet bárhol letehetjük, és a következő pillanatban már vizsgálhatjuk is a forgalmat. Utolsó munkánknak része volt több átlátszó ***Squid*** gyorsítótár-kiszolgáló is – ezek valóban átlátszóak, az IP-címzési sémában, az ügyfelek oldalán vagy a böngészőknél semmilyen módosítást nem igényelnek. Elég a ***Squidet*** futtató gépet a forgalomirányító elé helyezni, majd minden a 80-as kapun folyó forgalmat ráirányítani.
A ***Linux*** azon képessége, hogy átlátszó módon tud beépülni a meglévő hálózati rendszerbe, újfajta, korszerű szolgáltatások biztosítását teszi lehetővé. Mivel különböző hálózati készülékeket is egyesíthetünk egyazon virtuális elemben, ugyanazzal géppel tűzfalat is üzemeltethetünk, miközben tetszőleges szempontból figyelhetjük a hálózatot. Az egyetlen korlátot a vas sebessége és a rendelkezésünkre álló kártyahelyek száma jelenti.

Köszönetnyilvánítás

A szerző köszönettel tartozik azoknak a fiúknak és lányoknak, akik az áthidaló kódot készítették, valamint az ***ebtables*** parancskészlet szerzőinek, amiért üzembiztos, jól használható eszközt állítottak össze és tettek elérhetővé *GPL* hatálya alatt.

Linux Journal 2005. július, 135. szám

Jim Robinson a Maconben, Georgia-ban található Linux Solutions Provider, Inc. tanácsadó vállalkozás elnöke. Örömmel játssza férji és apai szerepét, gitározik és persze linuxozik.

KAPCSOLÓDÓ CÍMEK

- bridge.sourceforge.net
- ebtables.sourceforge.net
- www.snort.org
- www.squid-cache.org