Biztonságos FTP-szolgáltatás üzemeltetése vsftpd használatával

Több webhostinggal foglalkozó rendszergazda egyik nagy dilemmája, hogy hogyan lehetne biztonságos, ám mégis felhasználóbarát fájlelérést biztosítani.

Így vagy úgy, de előbb-utóbb minden rendszergazda szembekerül azzal a ténnyel, hogy a sikeres szerverüzemeltetéshez nem elég, ha rendszerét szigorú biztonsági megfontolásokkal őrzi. Gyakran előfordul, hogy egyébként rajta kívül senki nem tudja kényelmesen igénybe venni a szolgáltatásokat. A webhosting egy tipikus példája annak, amikor viszonylag nagy ügyfélkörnek kellene biztosítani azt, hogy hozzáférjen a (saját) weblapjához. Feltölthessen, letölthessen; magyarul módosítani tudja annak tartalmát. Ráadásul mindezt a lehető legegyszerűbben.
Az ilyen helyzetekben egyre gyakoribb a webes fájlkezelő, vagy az ***SCP/SFTP*** bevezetése. Már-már meg is bontaná a sörét a rendszergazda (kényelmesen hátradőlve székében), ám a következő pillanatban csörög a telefon, hogy nem lehetne-e inkább mégis ***FTP***-t használni. Majd az első elégedetlenkedők után jönnek a többiek is… Ugye ismerős a helyzet?
A közhiedelem úgy tartja, hogy az ***FTP*** használata nem biztonságos. Nos, sok esetben valóban helytálló lehet ez a megállapítás, de azért korántsem igaz. Jelen cikkben azt próbálom meg bebizonyítani, hogy hogyan lehet ezt a sokak által hangoztatott feltételezést megcáfolni. Mégpedig azzal, hogy létrehozunk egy olyan FTP kiszolgálót, amelyben egyesül a biztonság, az egyszerűség, a gyorsaság, és nem utolsó sorban alkalmas arra, hogy felhasználóinknak barátságos hozzáférést biztosítsunk személyes fájljaikhoz.
Mick Bauertől, a ***Linux Journal*** biztonsági témákkal foglalkozó szerkesztőjétől remek cikket olvashattunk a ***Linux-világ*** múlt év szeptemberi számában arról, hogy miképp lehet a ***vsftpd***-vel egy igazán remek ***anonymous FTP*** szervert készíteni. Ezt a gondolatmenetet követve, mi most elkészítünk egy ***PAM*** segítségével ***MySQL*** adatbázisból hitelesítő, opcionális ***SSL*** támogatással felvértezett ***vsftpd***-t; mellyel így élvezhetjük a több szempontból is hasznos virtuális felhasználó-kezelés előnyeit (szemben az ***SCP/SFTP*** megoldással, ahol kényelmetlenebb lenne ezt kivitelezni).
Hogy miért is kellenek ezek? A ***MySQL*** alapú háttérre azért van szükség, mert nagyobb számú felhasználói fióknál jobb teljesítményt nyújt, mint ha például egy ***Berkley DB*** adatbázisból hitelesítenénk; továbbá adminisztrálni is jóval egyszerűbb. Az ***SSL*** segítségével pedig biztosítani fogjuk a felhasználó és a szerver közötti titkosított adatáramlást, ezáltal megvédve a klienst például az illetéktelen lehallgatástól. Segítségével a jelszavak (és az adat is, ha úgy akarjuk) nem ***nyílt szöveg (plain-text)*** formátumban fognak mozogni a hálózaton, mint ahogyan azt egyébként tennék.

Vágjunk is bele!

Szerencsémre ***Mick*** már ismertette a ***vsftpd*** telepítését többféle disztribúció szemszögéből is, így ha minden jól ment, akkor jelenleg alapbeállításokkal ott figyel számítógépünkön a szoftver. Állítsuk le, és tegyük biztos helyre a mostani konfigurációs fájlt:

```
debian:~# mv /etc/vsftpd.conf /dev/null
```

Mi most teljesen a nulláról fogjuk újra felépíteni, megtárgyalva minden egyes opcióját.

A munkálatokhoz szükséges hozzávalók telepítése

Mielőtt belefognánk bármibe is, telepítsük a megfelelő szoftvereket. Szükségünk van tehát egy ***mysql*** szerverre (ha még nincs) a felhasználói nevek és jelszavak tárolásához, egy ***openssl*** csomagra az ***SSL*** támogatás biztosításához, valamint a ***libpam-mysql*** modulra a ***PAM*** és ***MySQL*** összekapcsolásához. ***Debian Sarge*** alatt ezeket nagyon egyszerűen össze is tudjuk szedni.

```
debian:~# apt-get install mysql-server openssl
➥libpam-mysql
```

Igazság szerint, akik már ismerik a ***MySQL***-t, és van egy megszokott kezelőfelületük az adminisztrálására, azok természetesen könnyedén fogják majd kezelni a felhasználókat. Akiknek viszont nincs, azoknak nem érdemes csak ezért telepíteniük egyet, hiszen a cikk végére látni fogjuk, hogy sokkal jobban járunk, ha megírjuk a saját szkriptünket az adminisztrációs munka megkönnyítésére.

A MySQL alapvető biztonsági beállításai

Ha most telepítettünk életünkben először ***mysql*** kiszolgálót, és még nem olvastuk el a dokumentációkat róla, akkor – bár a cikk alapvetően nem erről szól – leírom, hogyan tegyük meg a minimális biztonsági óvintézkedéseket.

```
debian:~# mysql -u root
mysql> USE mysql
mysql> DELETE FROM user WHERE User="";
mysql> UPDATE user SET Password=PASSWORD('ide
➥irjuk a jelszot') WHERE User='root'
mysql> FLUSH PRIVILEGES;
mysql> quit
```

Ez a néhány sor annyit csinál, hogy egyrészt kitörli az anonymous hozzáférést, másrészt megad egy jelszót a root felhasználó(k)nak, mivel telepítés után az üres. Sajnos elég gyakori, hogy ezt a figyelmetlen rendszergazdák így is hagyják. Ilyen esetben viszont sajnos hiába is dolgoznánk azon, hogy biztonságos FTP kiszolgálót hozzunk létre…

Az adatbázis elkészítése

A feladat tehát a következő: hozzunk létre egy `vsftpd` nevű adatbázist, egy `felhasznalok` nevű táblával, amiben a következő mezőkre lesz szükségünk: `nev`, `jelszo`. Természetesen más neveket is használhatunk, amennyiben ezek nem tetszenek.

```
debian:~# mysql -u root -p # bejelentkezünk
➥a MySQL szerverünkre root-ként,
➥immár jelszó használatával
mysql> CREATE DATABASE vsftpd; # elkészítjük
➥a vsftpd adatbázist
mysql> USE vsftpd; # majd ki is jelöljük, mivel
➥ezzel fogunk most dolgozni
mysql> CREATE TABLE felhasznalok ( nev char(16)
➥binary, jelszo char(16) binary ); # létrehozzuk
➥a táblát és a mezőket
mysql> INSERT INTO felhasznalok ( nev, jelszo )
➥values ( 'kisspista', password( 'a jelszo' ));
➥# felvesszük az első felhasználónkat
➥az adatbázisba
mysql> GRANT SELECT, INSERT, UPDATE, DELETE,
➥CREATE, DROP, INDEX, ALTER ON `vsftpd`.* TO
➥'vsftpd_mysql'@'localhost' IDENTIFIED BY
➥'a jelszavunk'; # egy vsftpd_mysql nevű
➥felhasználónak megadjuk a megfelelő jogokat
➥az adatbázisra (és létre is hozzuk egyúttal)
mysql> quit
```

Ezzel fel is készítettük rendszerünket arra, hogy a *vsftpd* gond nélkül információkhoz jusson az adatbázisból a továbbiakban.

A PAM beállítása MySQL hitelesítéshez

A *PAM (Pluggable Authentication Modules)* egy olyan függvénygyűjtemény, amelyben az a legnagyszerűbb, hogy egységes felületet biztosít a különböző programoknak az azonosítási procedúra elvégzésére. A folyamatot végző szoftver csak a *PAM*-mal áll kapcsolatban, a *PAM* pedig onnan veszi az információkat, ahonnan csak akarjuk (jelen esetben például egy *MySQL* adatbázisból). *Linuxunk* eme remek szolgáltatását használjuk például akkor is, amikor *SSH*-val bejelentkezünk szerverünkre. Ilyenkor az *SSH* démon elküldi a *PAM*-nak az általunk megadott felhasználónevet és jelszót, ami (többek közt) megvizsgálja a */etc/shadow* fájl alapján, hogy helyes adatokat adtunk-e meg. Amennyiben igen, visszajelez az *SSH* kiszolgálónak, hogy minden rendben, bejelentkezhetünk. Ma már a szolgáltatások legtöbbje a *PAM*-ot használja hitelesítésre.
Ha belépünk a */etc/pam.d* könyvtárba (bizonyos disztribúciók a */etc/pam.conf* fájlt használják), akkor látni fogjuk, hogy minden egyes szolgáltatáshoz más-más beállítófájl tartozik. A *vsftpd* például a `vsftpd` nevű fájlt hozza létre alapból (régebbi változatok esetén előfordulhat, hogy mást). Tegyük ezt is biztos helyre úgy, ahogyan a cikk elején az alapértelmezett *vsftpd.conf* fájlt. Egy teljesen újat fogunk írni. Adjuk ki a következő parancsokat:

```
debian:~# echo "auth required pam_mysql.so
➥user=vsftpd_mysql passwd=jelszavunk
➥host=localhost db=vsftpd table=felhasznalok
➥usercolumn=nev passwdcolumn=jelszo crypt=2"
➥> /etc/pam.d/vsftpd
debian:~# echo "account required pam_mysql.so
➥user= vsftpd_mysql passwd=jelszavunk
➥host=localhost db=vsftpd table=felhasznalok
➥usercolumn=nev passwdcolumn=jelszo crypt=2"
➥>> /etc/pam.d/vsftpd
```

Ezzel megadtuk a *PAM*-nak, hogy a *pam_mysql* modult használja az azonosításhoz. Természetesen a paraméterezés nagyon lényeges, így ügyeljünk arra, hogy helyesen adjuk meg például azt a jelszót, amit az adatbázis elkészítésekor létrehoztunk annak a *MySQL* felhasználónak (*vsftpd_mysql*), aki majd hozzáférhet az adatbázishoz.

Az SSL kulcs legyártása

Ahhoz, hogy *FTP* szerverünkön biztosítsuk az *SSL* szolgáltatást, szükségünk lesz egy *.pem* formátumú *RSA* kulcsra. Ezt a következőképpen tudjuk legenerálni:

```
debian:~# openssl req -x509 -nodes -newkey
rsa:1024 -keyout vsftpd.pem -out vsftpd.pem
```

Miután válaszoltunk az ez után feltett kérdésekre, létre is jön a `vsftpd.pem` fájl ott, ahol kiadtuk a parancsot. Másoljuk át a */usr/share* mappába, és állítsuk be a megfelelő jogosultságokat rajta.

```
debian:~# cp vsftpd.pem /usr/share
debian:~# chmod 400 /usr/share/vsftpd.pem
```

Problémák az SSL használatával

Az *SSL* használata nagyszerű dolog, azonban van egy hátulütője a dolognak. Sajnos nem minden *FTP* kliens támogatja. Így el kell döntenünk, hogy kötelezővé tesszük-e a felhasználóknak a használatát, vagy sem. Amennyiben maximális biztonságra törekszünk, akkor mindenképp kötelezzük el magunkat (és ügyfeleinket) mellette. Ennyi áldozatot bőven megér, hogy biztonságban tudhatjuk a kommunikációt is a hálózaton.

A vsftpd beállításának előkészületei

Végre elérkeztünk oda, hogy bekonfigurálhatjuk a *vsftpd*-t. No, de mit is akarunk pontosan csinálni? Mik a követelmények a beállításokkal szemben?

A vsftpd.conf

```
# Beállítjuk, hogy daemon-ként induljon el
# a vsftpd
listen=YES
# Tiltjuk a névtelen bejelentkezést
anonymous_enable=NO
# Viszont engedélyezzük a "nem-névtelen"
# felhasználókat (akiket a PAM hitelesít nekünk)
local_enable=YES
# Engedélyezzük a vendég felhasználót.
# Ez nagyon fontos, mivel ez aktiválja
# a virtuális felhasználó-kezelést.
guest_enable=YES
# Beállítjuk, hogy minden bejelentkező felhasználót
# a vsftpd_user felhasználóra mappeljen a szerver
guest_username=vsftpd_user
# Megadjuk a felhasználók egyéni beállításait
# tartalmazó mappa helyét a fájlrendszeren
user_config_dir=/usr/share/vsftpd/users
# Beállítjuk, hogy a vsftpd nevű PAM beállítófájlt
# használjuk az azonosítási procedúrához
pam_service_name=vsftpd
# Engedélyezzük az írást. feltöltést, könyvtár
# létrehozását
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES
anon_other_write_enable=YES
# Beállítjuk, hogy mindenki csak a saját
# könyvtárján belül mozoghasson
chroot_local_user=YES
# Itt adhatjuk meg az ftp root könyvtárat,
# de mi most elővigyázatossági okokból az
# /usr/share/vsftpd/empty -t adjuk meg, később
# rájövünk, hogy miért…
local_root=/usr/share/vsftpd/empty
# Engedélyezzük az SSL támogatást
ssl_enable=YES
# Meghatározzuk a .pem fájl helyét
rsa_cert_file=/usr/share/vsftpd.pem
# Kényszerítjük-e a felhasználókat, hogy
# a bejelentkezési információkat SSL-el titkosítsák?
force_local_logins_ssl=NO
# Kényszerítjük-e a felhasználókat, hogy
# az adatfolyamatot SSL-el titkosítsák?
force_local_data_ssl=NO
# Beállítjuk a megfelelő jogosultságokat
# a feltöltött fájlokra..
anon_umask=022
# Ha akarunk megadni globális sávszélesség
# korlátot, akkor itt megtehetjük. Ha 0-án hagyjuk,
# akkor korlátlanra állítjük. Az értéket byte / sec
# formában értelmezi a program.
anon_max_rate=0
```

Egyrészt ne engedélyezzünk semmilyen anonymous kapcsolódást, csakis azok csatlakozhassanak, akik a *MySQL* adatbázisunkban szerepelnek. Szerepeljen benne az *SSL* lehetősége (jelen esetben nem kényszerítve), a virtuális felhasználó-kezelés, a különböző felhasználók egyéni beállításainak engedélyezése, valamint az, hogy mindenki csak a neki beállított könyvtárban mozoghasson. Érdemes külön figyelmet szentelni a felhasználók egyéni beállításainak lehetőségére. Ez egy remek szolgáltatás, ami a *vsftpd 1.1.0*-s verziójában jelent meg először. Segítségével minden egyes felhasználónévhez más-más beállításokat rendelhetünk, kezdve a sávszélesség korlátozástól, a *home* könyvtár helyéig.

Előkészületként szükségünk lesz egy vsftpd_user nevű felhasználóra a rendszeren, egy /usr/share/vsftpd/empty, egy /usr/share/vsftpd/users, valamint egy ftp root könyvtárra (ami legyen most például a /var/www).

```
debian:~# mkdir -p /usr/share/vsftpd/empty &&
➥chmod 500 /usr/share/vsftpd/empty
debian:~# mkdir /usr/share/vsftpd/users
debian:~# useradd -d /usr/share/vsftpd/empty
➥vsftpd_user
debian:~# passwd vsftpd_user
```

A felhasználók egyéni beállításai

Emlékezzünk csak vissza, az adatbázis létrehozásánál készítettünk egy `kisspista` nevű felhasználót. Hogyan hozzunk neki létre egyedi beállításokat, mint például, hogy hol legyen a *home* könyvtára?

A működési elv nagyon egyszerű. A konfigurációs fájlban megadott helyen (*/usr/share/vsftpd/users*) létre kell hozni egy fájlt a felhasználó nevével megegyezően, majd szépen beleírni azokat a beállításokat, amiknél azt szeretnénk, hogy eltérjen az alapértelmezett értéktől. Hozzuk is létre neki!

```
debian:~# echo "local_root=/var/www/kisspista.hu"
➥> /usr/share/vsftpd/users/kisspista
debian:~# echo "anon_max_rate=20000"
➥>> /usr/share/vsftpd/users/kisspista
```

Ezek után készítsük el *Kiss Pista* tetszetős weboldalát, hogy ne tátongjon ott üresen az a könyvtár. Ne felejtsük el megfelelően beállítani a könyvtár tulajdonosi jogát!

```
debian:~# mkdir -p /var/www/kisspista.hu
debian:~# chown vsftpd_user /var/www/kisspista.hu
debian:~# echo "<HTML><HEAD><TITLE>Kiss Pista
➥weboldala</TITLE></HEAD><BODY BGCOLOR="green"
➥TEXT="red"><CENTER><H1>Felújítás
➥alatt…</H1></CENTER></BODY></HTML>"
➥> /var/www/kisspista.hu/index.html
```

Indítás, bejelentkezés

Ezzel el is készültünk minden beállítással, elindíthatjuk a *vsftpd*-t. Tapasztalataim szerint nem nagyon reagál a konfigurációs hibákra, ha valami nem tetszik neki, akkor egyszerűen nem indul el. Ha viszont semmi probléma nem adódott, akkor itt az idő letesztelni, hogy működik-e a hitelesítés. Valami hasonlót kell látnunk:

```
debian:~# ftp localhost
Connected to localhost.localdomain.
220 (vsFTPd 2.0.1)
Name (localhost:rusty): kisspista
331 Please specify the password.
```

1. lista

Titkosítatlan bejelentkezés

```
04:04:32.660997 IP (tos 0x0, ttl 122, id 36801,
➥offset 0, flags [DF], length: 56)
➥3e44b710.adsl.enternet.hu.4480 >
➥programfiles.hu.ftp: P [tcp sum ok] 1:17(16)
➥ack 21 win 65515
E..8..@.z..&>D...F2=....B..Z-.WpP...q...USER
➥kisspista
04:04:32.687190 IP (tos 0x0, ttl 122, id 36807,
➥offset 0, flags [DF], length: 56)
➥3e44b710.adsl.enternet.hu.4480 >
➥valami.hu.ftp: P [tcp sum ok] 17:33(16)
➥ack 55 win 65481
E..8..@.z.. >D...F2=....B..j-.W.P....PASS pisti9234
```

Titkosított bejelentkezés

```
04:09:27.227510 IP (tos 0x0, ttl  64, id 50447,
➥offset 0, flags [DF], length: 1053) valami.hu.ftp
➥> 3e44b710.adsl.enternet.hu.4509: P 52:1065(1013)
➥ack 143 win 6432
E.....@.@....F2=>D.....@...g9.@P.. `T......J...F..A.
➥.....wq.].....gq...i........
04:09:27.286248 IP (tos 0x0, ttl 122, id 37542,
➥offset 0, flags [DF], length: 230)
➥3e44b710.adsl.enternet.hu.4509 > valami.hu.ftp:
➥P 143:333(190) ack 1065 win 64471
E.....@.z...>D...F2=....g9.@@...P..................
➥vE.I....f`...TqSxK.z...^d....R.
```

```
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 0    0     122 Jan 18 05:17 index.html
226 Directory send OK.
```

Ha nem sikerül bejelentkeznünk, akkor nézzük át ismét minden konfigurációs fájlunkat, valamint olvassunk bele a */var/log/mysql/mysql.log* és a */var/log/syslog* naplófájlokba.

Az SSL támogatás tesztelése

Az *SSL* hasznosságára valószínűleg akkor fogunk rájönni, ha a saját szemünkkel látjuk a különbséget a titkosítatlan, valamint a titkosított bejelentkezés között.
Tegyük is ezt meg. A megfigyeléshez szükségünk lesz a `tcpdump` nevű programra, amit ***Debian Sarge*** alatt egyszerűen az `apt-get install tcpdump` paranccsal tudunk beszerezni. Ha megvan, adjuk ki a következő utasítást:

```
debian:~# tcpdump -A host ipcím and port 21 -v
```

A parancsba természetesen helyettesítsük be az `ipcím` helyére azt az IP címet, ahonnan csatlakozni fogunk az *FTP* szerverre a tesztelés ideje alatt. Figyeljük meg a program kimenetét akkor, amikor egy *SSL*-t nem támogató kliens programmal, valamint, amikor egy azt támogatóval jelentkezünk be. A különbség szemmel látható (*1. lista*)
A probléma persze nem ott kezdődik, hogy a titkosítatlan bejelentkezésnél az *FTP* szerverünkön ki tudjuk szűrni a hálózati forgalomból a felhasználói nevet és jelszót, hanem ott, hogy a helyi hálózaton – ahonnét a kliens csatlakozik – kis túlzással bárki meg tudja ezt tenni.

2 .lista

```
#!/usr/bin/perl
use strict;
use DBI;

my $host = “localhost”; # a MySQL elérhetősége
my $database = “vsftpd”; # az adatbázis neve,
➥ahol tároljuk a felhasználókat
my $table = “felhasznalok”; # a tábla neve
➥az adatbázison belül
my $user = “vsftpd_mysql”; # a MySQL felhasználó
➥neve, aki jogosult hozzáférni az adatokhoz
my $passwd = “jelszavunk”; # a MySQL felhasználó
➥jelszava
my $dir = “/usr/share/vsftpd/users”; # a felhasználók
➥egyeni beállításait tartalmazó könyvtár

my $sql = “”; # hagyjuk üresen
my $val = “”; # hagyjuk üresen

if ($ARGV[0] eq “del”) {
    $sql = “delete from $table where nev =
    ➥‘$ARGV[1]’”;
    if ($ARGV[1]) {
        unlink (“$dir/$ARGV[1]”);
    }
} elsif ($ARGV[0] eq “add”) {
    $sql = “insert into $table (nev, jelszo) values
    ➥(‘$ARGV[1]’, password(‘$ARGV[2]’))”;
    open(FH, “>$dir/$ARGV[1]”);
        print “A felhasználó home könyvtára: “;
        $val = <STDIN>;
        print FH “local_root=”.$val.””;
    close(FH);
} else {
    print “Nem megfelelő paraméterezés!\n”;
}

if ($sql) {
    my $dsn =
“DBI:mysql:host=$host;database=${database}”;
    my $dbh = DBI->connect($dsn, $user, $passwd)
        or die “Nem tudtam kapcsolodni
        ➥a szerverhez!\n”;
    my $s = $dbh->prepare($sql);
    $s->execute();
    $s->finish();
    $dbh->disconnect();
}
```

Automatizált felhasználókezelés

Kétségtelen, hogy az első gondolat, ami az emberben felmerül ezzel a megoldással kapcsolatban, az az, hogy milyen jó lenne, ha nem két külön helyen (a *MySQL* adatbázisban, és a */usr/share/vsftpd/users* könyvtárban) lehetne konfigurálni a felhasználókat.
A legegyszerűbb, ha elkészítjük hozzá saját szkriptünket, ami leveszi a vállunkról ezt a terhet is. Íme egy egyszerű *Perl* szkript a feladat megoldására (*2. lista*).
Töltsük ki a megfelelő részeket a beállításainkkal, majd mentsük el például ***user.pl*** néven. Ne felejtsük el beállítani, hogy csak nekünk legyen jogunk hozzáférni ehhez a fájlhoz, mert esetleg kellemetlen meglepetések érhetnek minket. Ha mindezzel megvagyunk, próbáljuk ki, hogy működik-e:

```
debian:~# ./user.pl add kisjozsi jozsi123
```

Válaszoljunk a feltett kérdésre, és ha nem kaptunk hibaüzenetet, akkor nagy valószínűséggel sikerrel jártunk. Próbáljunk bejelentkezni `kissjozsi` felhasználóval.

Törölni a következőképpen tudunk:

```
debian:~# ./user.pl del kissjozsi
```

A szkript természetesen bővíthető, így más paramétereket is bekérhetünk a felhasználó *home* könyvtárán kívül.

Összegzés

A rengeteg pozitívum mellett szóljunk néhány szót ennek a kivitelezésnek a hátulütőiről is. Kétségtelen, hogy nem a legszebb és legkényelmesebb a *vsftpd* virtuális felhasználó-kezelése. Továbbá abban is biztos vagyok, hogy nem a döbbenetesen precíz sávszélesség-menedzsment funkciója miatt fogják imába foglalni a nevét a közeljövőben. Vagy a józan paraszti ésszel is érthető konfigurációs beállítása miatt…
Arra a feladatra viszont, amire mi most beállítottuk, úgy gondolom, tökéletes. Ha egy kicsivel kényelmesebb lenne a program, az már sajnos a biztonság és a gyorsaság rovására menne. Bár tegyük hozzá, tulajdonképpen egy kis kreativitással a program gyengeségei könnyedén ellensúlyozhatóak (mint azt iménti kis *Perl* szkript is mutatja).
További jó kísérletezést kívánok a *vsftpd*-hez, és amennyiben valakinek kérdése, vagy javaslata lenne az itt leírtakkal kapcsolatban, örömmel veszem, ha megírja azt nekem.

Maróti Tamás (tamas.maroti@async.hu)
18 éves, az ASYNC Magyarország Kft.-nél dolgozik informatikusként, ahol a hálózatbiztonság üzletágat vezeti. Emellett a Leövey Klára KSZKI tanulója. Szabadidejében ha teheti zenél, ír, könyveket olvas, illetve színházi előadásokban statisztál.