VPN kialakítása

Szabad szoftver keretrendszer

Készítette a Közigazgatási és Igazságügyi minisztérium E-közigazgatási
Szabad Szoftver Kompetencia Központja
Budapest, 2013

Kódszám: EKOP–1.2.15

Tartalomjegyzék

VPN kialakítása

Ha már eljutottunk oda, hogy az SSH alagút és az egyéb megoldások nem kínálnak kellő komfortot, vagy nem jól központosíthatóak, vagy egész egyszerűen csak nem mindenki számára jelentenek könnyen kezelhető megoldást, akkor érdemes megfontolni egy olyan megoldást, amikor a teljes hálózati kommunikáció egy titkosított magánhálózat kiépítésével jár együtt. A VPN nem csak akkor jelent megoldást, amikor két céges alhálózatot akarunk összekapcsolni, bár valójában erre tervezték. Akkor is kiválóan használható, ha az otthoni gépünkkel akarunk rákapcsolódni egy céges hálózatra, illetve bizonyos esetben szerver és szerver közötti kommunikációra is használható, de arra talán nem a legideálisabb megoldás. A fő előnye, hogy az alhálózatokat típusától függően erős titkosítással biztosítja, valamint minden, az adott kapcsolaton átmenő forgalom az átjáró beállításainak függvényében a titkosított közegben utazik. A manapság használatos VPN kliensek olyan egyszerűen kezelhetőek, hogy egy átlagos tudású felhasználónak sem okoz gondot a kezelése, természetesen ha megfelelően automatizálta a rendszergazda a felépülési folyamatot. Segítségével nem csak a szervereinket érhetjük el, hanem az irodában lévő bekapcsolva hagyott klienseket is (például egy távoli asztal szoftver segítségével). VPN megoldásokból számtalan létezik, mint ahogyan a VPN-protokoll használata alapján is sokrétűek (PPP, PPTP, L2TP, IPsec)[1]. A leginkább elterjedt és széles körben használt az OpenVPN[2], amely széleskörű dokumentációval[3] rendelkezik. Az Ubuntu kliensek beépített lehetőséggel rendelkeznek a NetworkManager programon keresztül a gyakorlatilag pár kattintásos VPN kapcsolat kiépítésére. A VPN technológiára is érvényes az SSH-nál már bemutatott biztonsági irányelvek alkalmazása, azaz a lehetőségek tárházából használjuk ki a lehető legtöbb azonosítási, hitelesítési eljárást és módot annak érdekében, hogy biztosak lehessünk benne, csak az jut be a hálózatunkba, akit mi be is akartunk engedni. Ha az a cél, hogy néhány felhasználó számára biztosítsuk OpenVPN-nel a távoli elérést, akkor a következő lépésekre lesz szükségünk.

OpenVPN

Szerver oldalon telepítsük az OpenVPN csomagot, és hozzuk létre a fő konfigurációs fájlt, neve tipikusan /etc/openvpn.conf, de például Ubuntu 12.04 LTS alatt /etc/openvpn/szerverneve.conf. A lenti konfigurációban az alapértelmezett UDP helyett TCP-t használunk. (Nincs rá általános szabály, hogy TCP, vagy UDP a jó. Az OpenVPN dokumentációja az alapértelmezett UDP-t javasolja, viszont elég sok tűzfalon könnyebb átengedtetni egy titkosított IMAP-nak álcázott kapcsolatot[4], mint mindenféle szedett-vedett UDP-csomagokat. Ráadásul a tűzfalszoftverek jelentős része az SSL mi-

[1] http://en.wikipedia.org/wiki/Virtual_private_network
[2] http://openvpn.net/
[3] http://openvpn.net/index.php/open-source/overview.html
[4] Ebben a példában egy működő gyakorlatot mutatunk, viszont a helyi biztonsági házirendbe ütközhet, és az azért felelős társaság akár támadásnak is tekintheti. Erről mindenképpen bizonyosodjunk meg, mielőtt így állítanánk be. Az OpenVPN-hez a szabványosított szerverport az 1194-es, kétségek esetén próbálkozzunk inkább azzal (persze mind a szerver, mind a kliens oldalon azonos módon kell beállítani).

att nem nagyon tudja megkülönböztetni a valódi IMAPS forgalmat, az annak álcázott OpenVPN forgalmától.) Az egyes klienseket tanúsítvány segítségével azonosítjuk. A klienseknek 1 hétig érvényes dinamikus címeket osztunk ki a 172.22.2.0/24-es tartományból. Lehetőség van az egyes klienseket ideiglenesen kitiltani a VPN-ből. Ehhez a Client-Config-Dir könyvtárban létre kell hozni a kliens tanúsítványában szereplő névvel megegyező nevű egyedi konfigurációs fájlt, amely fájlba a „disable” sort beírva, a tiltás érvényre jut. Ha valamely klienst véglegesen ki kell tiltani, akkor javasolt a disable helyett a tanúsítvány visszavonásával megoldani a dolgot.

A konfigfájlban megadandó paraméterek

Egy un. TUN eszközt használunk a kommunikációra, amikor a VPN szoftver elindul, akkor egy tunX nevű hálózati interfész jelenik meg.

```
dev tun
```

A fent említetteknek megfelelően, titkosított IMAP protokollnak álcázzuk magunkat – ezért a TCP beállítás, és ezért kommunikálunk a 993-as porton keresztül

```
proto tcp
port 993
```

A szerver ebből a címtartományból osztja a címeket a klienseknek.

```
server 172.22.2.0 255.255.255.0
```

Ezek a hitelesítés használatához szükséges fájlokat adják meg.
A Root-CA:

```
ca      /etc/openvpn/szerverneve/keys/ca.crt
```

A szerver tanúsítványa:

```
cert    /etc/openvpn/szerverneve/keys/kompkp1.crt
```

A szerver tanúsítványához tartozó kulcs:

```
key     /etc/openvpn/szerverneve/keys/kompkp1.key
```

A szervernek szükséges Diffie-Hellmann paramétereket tartalmazó fájl:

```
dh      /etc/openvpn/szerverneve/keys/dh2048.pem
```

Időnként szükség lehet az egyes kliensek számára egyedi paramétereket meghatározni – vagy mint fentebb már volt róla szó, ideiglenes tiltáshoz is nagyon kényelmes, ha minden kliensnek van saját konfig fájlja. Amennyiben egy kliens csatlakozik, akkor a tanúsítványa CN (CommonName) mezőjében szereplő nevű fájlt veszi figyelembe az OpenVPN szerver, ha ilyen nevű fájl nincs, akkor a DEFAULT nevű fájlt – és mindezeket a CCD paraméterben meghatározott könyvtárban keresi a szerver.

```
client-config-dir /etc/openvpn/szerverneve/ccd
```

Itt tárolja a szerver, hogy egyes klienseknek milyen IP-címet osztott ki, és az utolsó bejelentkezés után mennyi ideig tárolja ezt az információt. Ha több idő telik el, a bejegyzés törlődik, ily módon már nem biztos, hogy a kliens ugyanazt a címet kapja megint.

```
ifconfig-pool-persist /etc/openvpn/szerverneve/client_ips.txt 604800
```

Bizonyos esetekben SIGUSR1 megszakítás generálódik, de nem szeretnénk, ha ezek a megszakítások különböző paraméterek módosulását eredményeznék.

```
persist-tun
persist-key
persist-local-ip
```

```
persist-remote-ip
```

A működési állapotra vonatkozó információk ebbe a fájlba kerüljenek:

```
status /var/log/openvpn-status.log
```

A naplózás szintjének és helyének meghatározása:

```
verb 3
log-append      /var/log/openvpn.log
```

Megpróbáljuk fenntartani a hálózati kapcsolatot akkor is, ha nincs valódi adatforgalom az egyes kliensek irányában. Ez azért lehet érdekes, mert sok tűzfal bizonyos időtartam után, ha nincs forgalom egyszerűen lebontja a kapcsolatot.

```
keepalive 60 300
```

Ezzel a paraméterrel biztosítjuk, hogy a kliensek egymást is elérhetik, nem csak a szervert. Nyilván, ha szeparálni kell őket, ez az opció nem szükséges.

```
client-to-client
```

A paraméterek nélküli tömörítési opció azért jó, mert ebben az esetben akár kliensenként lehet kapcsolgatni, hogy legyen-e forgalomtömörítés. Általában érdemes bekapcsolni, de talán egy fokkal jobb, ha ezt a kliensek DEFAULT, illetve kliens-specifikus fájljában tudjuk testre szabni, nem pedig globálisan engedélyezzük vagy tiltjuk.

```
comp-lzo
```

A teljes VPN használatához ezen kívül szükséges a fent hivatkozott tanúsítványok (és a kliensek tanúsítványai) létrehozása. Ehhez a ma már szinte minden terjesztésben alapértelmezetten elérhető OpenSSL csomag szintén openssl nevű parancsát kellene közvetlenül használni, de használható helyette néhány könnyebben kezelhető segédprogram. Az OpenVPN terjesztés részeként elérhető az un. easy-rsa nevű parancssoros eszköz, de aki jobb szereti a grafikus eszközöket, az használhatja a Gnomint[5] nevű grafikus eszközt is, amely sok disztribúcióban szintén bináris csomagként elérhető, kényelmesen használható segédprogram.

Easy-rsa

Nézzük az easy-rsa-t. Először keressük meg, és lehetőleg másoljuk át a feltelepített könyvtárstruktúrát egy kényelmesebben kezelhető (és ráadásul frissítésektől nem bántott) helyre, mondjuk a saját könyvtárunkba. Például Ubuntu-12.04 LTS alatt a következő paranccsal:

```
cp -r  /usr/share/doc/openvpn/examples/easy-rsa $HOME/
```

Noha Ubuntun elérhető a régebbi 1.0-s verziójú easy-rsa is, javasolt az újabb verziót használni. Ehhez lépjünk a másolat könyvtárba, és annak 2.0 nevű alkönyvtárába:

```
cd ~/easy-rsa/2.0
```

Legelső lépésként az itt található vars nevű fájlt kell megszerkeszteni, és a különböző tanúsítványokhoz szükséges információkat beállítani. Ehhez át kell írni a fájl végén található export KEY_ kezdetű sorokat, hasonlóan az itt találhatóhoz:

```
export KEY_COUNTRY="HU"
export KEY_PROVINCE="Pest megye"
export KEY_CITY="Budapest"
export KEY_ORG="Szervezet Neve Ha Tartalmaz Szóközt Kötelező Az Idézőjel"
export KEY_EMAIL="hivatalos.emailcim@mail.example.hu"
export KEY_CN=openvpn-server
```

[5] http://gnomint.sourceforge.net/

```
export KEY_NAME=vpn.example.hu
export KEY_OU="szervezeti egység"
export PKCS11_MODULE_PATH=changeme
export PKCS11_PIN=1234
```

Ha a fájl ki van töltve, kezdhetjük. A legelső lépés, hogy az aktuális shellbe beolvastatjuk a szükséges beállításokat a következő, elég furcsa paranccsal:

```
. ./vars
```

A parancs neve „.” (azaz pont), paramétere pedig ./vars (pont-per-vars). Ezek után első lépésként futtassuk le a következő parancsot. Vigyázat, ezt a későbbiekben csak abban az esetben szabad használni, ha teljesen tiszta lappal szeretnénk kezdeni, minden korábban esetleg létrehozott kulcsot, tanúsítványt, egyebet töröl. **Még egyszer:** ezt a parancsot csak most kell futtatni, a későbbiek során már nem:

```
./clean-all
```

Miután (az egyébként tiszta) környezetünket kitakarítottuk, jöhet a munka hasznosabb része. Előállítjuk az un. root-CA-t. A parancs futása során kérdezget tőlünk, alapértelmezettként mindenhol megfelel az alapértelmezett válasz (amiket a vars fájlban kitöltött adatokból állít elő):

```
./build.ca
```

Ha ezzel készen vagyunk, állítsuk elő a szerverünk saját tanúsítványát. Amikor jelszót kér tőlünk a rendszer, ott is elegendő az Enter, ellenben ezt követően kétszer is egyértelműen IGEN választ *(„y”)* kell adni: előbb a tanúsítvány aláírásakor *(„Sign the certificate”)*, majd pedig a tanúsítvány elfogadtatásakor *(„1 out of 1 certifiate requests certified, commit”)* megjelenített üzenetnél.

```
./build-key-server server
```

Ha ez is megvan, gyárthatjuk a kliensek által használandó tanúsítványokat:

```
./build-key egyik-kliens
./build-key masik-kliens
```

Amikor a tanúsítványok már megvannak, még egy legenerálandó adat van hátra. A szerver számára szükséges un. Diffie-Hellman paraméterek előállítása. Ehhez szintén adott egy segédprogram:

```
./build-dh
```

Végeredményként előálltak a következő fontos fájlok, elvben a keys nevű alkönyvtárban:

A szerver számára szükségesek: dh*.pem (a fájl neve attól függ, hogy a vars fájlban mekkorára állítottuk a KEY_SIZE értékét – ezt az alapértelmezett 1024-ről javasolt megnövelni, legalább 2048-ra), a server.crt és server.key fájlok. Minden egyes kliens számára szükséges a kliensneve.crt és kliensneve.key fájlok. Végül mind a szerver, mind a kliensek számára szükséges a ca.crt. (Valamint a tanúsítványokat létrehozó – az eddigi példában a szerver – számára a ca.key is.) A *.key fájlokat erősen védeni kell, a klienseknek a sajátjukat javasolt valamilyen védett adatcsatornán eljuttatni (esetleg megfelelően titkosított – például GPG[6] – e-mailben). Miután mindent legeneráltunk, a ca.crt, server.crt, server.key és dh*.crt fájlokat helyezzük el ott, ahol a fenti openvpn-konfigurációs fájl szerint hivatkozunk rájuk. A szerver beállításával végeztünk, tulajdonképp a szerver elindítása és a kliensek bekonfigurálása van hátra.

Kliensek beállítása

A kliensekhez juttassuk el a ca.crt fájlt, és a hozzájuk tartozó client.crt és client.key fájlokat, majd pedig a kliens számára vagy létrehozunk egy szöveges konfigurációs fájlt, amit majd az im-

[6] Részletes GPG ismeretekért olvassa el GPG-fejezetünket.

portál[7], vagy rábízhatjuk magukra a felhasználókra is, hogy a kívánt adatokat adják meg kézzel. Egy tipikus kliens oldali konfiguráció, itt már csak a fentiekben nem tárgyalt paraméterek magyarázatával:

Ezeket a paramétereket a szerverrel összhangban kell beállítani:

```
dev tun
proto tcp
```

Ez a paraméter azt jelenti, hogy a kapcsolatban mi kliensként veszünk részt, TLS-t fogunk használni, és induláskor elkérjük a szervertől a ránk vonatkozó paramétereket.

```
client
```

Ehhez a szerverhez kapcsolódunk, a megadott porton.

```
remote 1.2.3.4 993
```

Véletlenszerű portot használunk a szerverrel való kommunikációhoz. Bizonyos tűzfalbeállítások esetén okozhat problémát, és helyette a tűzfal adminisztrátorával egyeztetett portot kell megadni (a port vagy lport paraméterek segítségével).

```
nobind
```

A következő paraméter egyfajta plusz biztonságot nyújt, a klienseink csak olyan OpenVPN-szerverhez fognak csatlakozni, amelynek a tanúsítványa szerver típusú. (Ehhez a szerver tanúsítványát megfelelő módon kellett generálni – szerencsére mi ezt tettük.) Nyilván a tanúsítványok használata eleve növeli a biztonságot, de ahogy fent már elhangzott: használjunk ki minél több biztonságot növelő lehetőséget, ha az nem megy a használhatóság rovására.

```
ns-cert-type server
```

Ezek pedig már ismerősek korábbról.

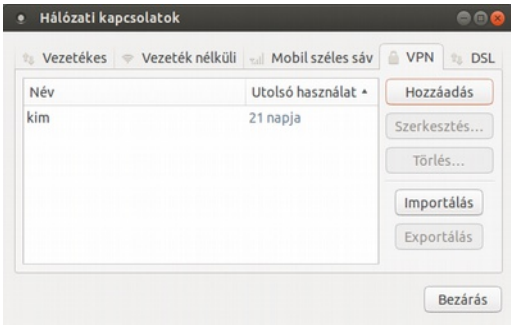
```
persist-key
persist-tun
ca ca.crt
cert kliens1.crt
key kliens1.key
comp-lzo
verb 3
```

Fenti beállítások szerint a szerver az 1.2.3.4 IP-című gép 993-as portján érhető el. Az azonosításhoz ahogy korábban már elhangzott, tanúsítványt használunk, ez a kliens1.crt fájlban található. A fenti VPN kliens konfiguráció állományba mentése után (*.ovpn kiterjesztéssel) célszerű egy könyvtárba helyezni a kulcsokkal (crt és key fájlok), majd Ubuntu Desktop rendszer esetén az .ovpn állományt megadni a NetworkManager alkalmazás VPN kiegészítőjének. Előtte azonban Ubuntu alatt szükséges az NM OpenVPN kiegészítésének a telepítése a következőképpen:

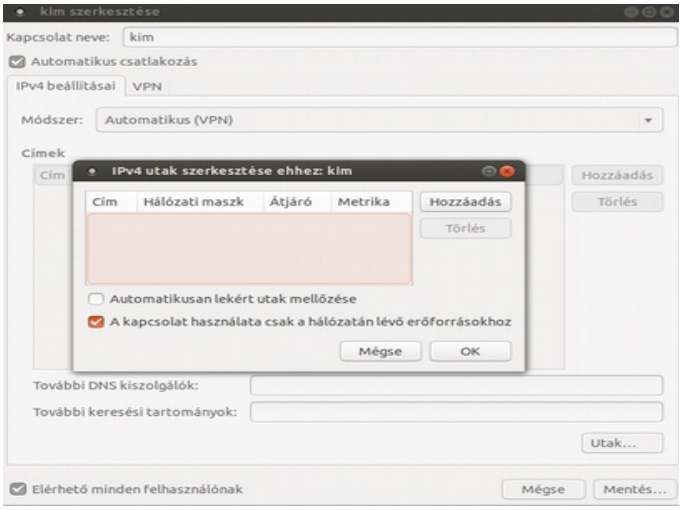
```
sudo apt-get install network-manager-openvpn
```

Ezek után a NetworkManager már felismeri az opvn állományt, amit egyszerűen az importálás menüpontban tudunk felolvastatni vele:

[7] Természetesen kliens konfigot nem kell létrehozni abban az esetben, ha a felhasználók nem csak egy viszonylag egyszerűbb importálást, hanem egy bonyolultabb, pár paraméter kitöltését igénylő műveletet is képesek elvégezni a NetworkManager-ben.



Ha szeretnénk, hogy a VPN elindítása utána, a default route-ot a VPN csak kiegészítse, és ne állítsa át a VPN irányába teljesen, akkor a következő kapcsolót alkalmazzuk:



Tehát, ha az „A kapcsolat használata csak a hálózatán lévő erőforrásokhoz” pipát használtuk, egyszerre fog működni a publikus internet felé a routing és a VPN felé is. Persze sok esetben pont az a cél, hogy ha a VPN van életben, akkor a publikus internet felé a felhasználó ne tudjon forgalmazni, ebben az esetben célszerű ezt a kapcsolót alapállapoton – azaz kikapcsolva – hagyni.