Paranoid Pingvin – Linuxos VPN Módszerek

Vajon melyik privát hálózat felel meg nekünk? Mick végigszalad a lehetőségeken és bemutatja a győzteseket, valamint kapunk néhány hasznos tanácsot is.

A ***virtuális magánhálózat (virtual private network; VPN)*** igen hasznos és kényelmes dolog. Az úton lévők biztonságosan csatlakoznak vele saját otthoni hálózatukhoz utazás közben; a földrajzilag megoszló társaságok nyilvános sávszélességet használó ***WAN*** kapcsolatként alkalmazzák; a drótnélküli ***LAN*** felhasználók pedig újabb védelmi rétegként használják WLAN kapcsolataik felett.

Linuxra számos ***VPN*** csomag létezik: ***FreeS/WAN***, ***OpenS/WAN***, ***PoPToP***, ***OpenVPN*** és ***tinc***, hogy csak néhányat említsünk. De hogyan választhatjuk ki a megfelelőt a megfelelő feladatra? Nos, ebben a cikkben éppen ezzel foglalkozunk.

VPN szerkezet

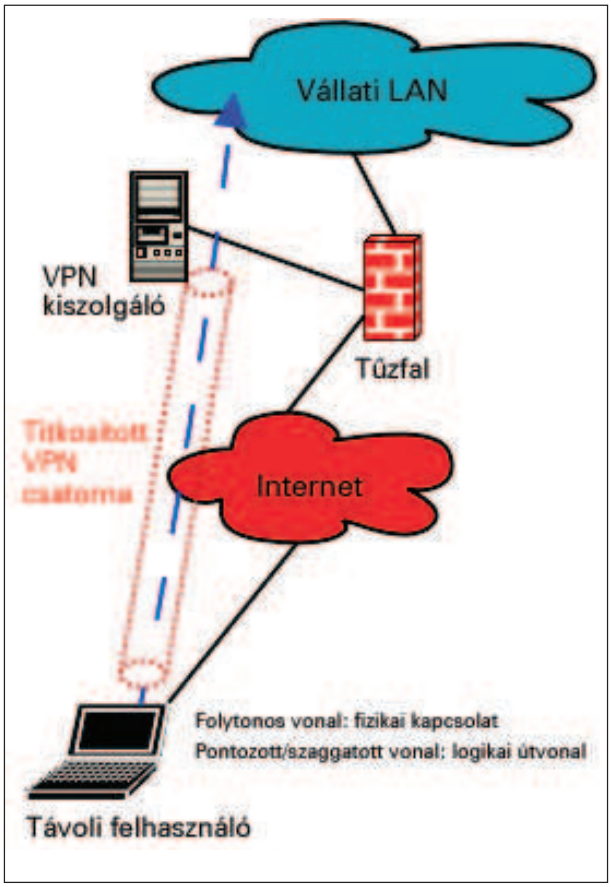
A ***VPN***-ek általában két különféle feladatot látnak el.

Az egyik feladat a felhasználóknak lehetővé tenni az otthoni hálózathoz történő titkosított csatlakozást egy megbízhatatlan médiumon, például az interneten vagy ***wireless LAN (WLAN)*** rendszeren keresztül. Az *1. ábra* a távoli-elérés összeállítást mutatja be.

Az 1. ábrán látható szaggatott kék adatfolyam a teljes üzleti ***LAN*** hálózat elérését jelképezi. A gyakorlatban a távoli elérésű ***VPN*** csatornák ***hozzáférési listák (Access Control List; ACL)*** vagy tűzfalszabályok alapján korlátozhatják ezt az elérést. Az elérést akár egyetlen gép egyetlen alkalmazására is korlátozhatjuk, ahogy az ***SSL-VPN*** rendszerekben ezt általában meg is teszik (az ***SSL-VPN*** rendszerrel hamarosan foglalkozunk).

Az egyszerűség kedvéért az 1. ábra egyetlen ügyfelet mutat be; természetesen az ilyen felállás szinte mindig több ügyfelet tartalmaz. Más szóval, a távoli-elérés megoldás esetében ügyfél-kiszolgáló rendszerben dolgozunk, ahol egyetlen ***VPN*** kiszolgáló vagy gyűjtő távoli felhasználók százaival vagy akár ezreivel létesíthet kapcsolatot. (Ebben a cikkben az ügyfél-kiszolgáló kifejezést kibővített és nem a bizonyos programfejlesztési értelemben használom.)

Bár az 1. ábra a ***VPN*** kiszolgálót az üzleti ***LAN VPN*** végpontjaként mutatja be, tűzfalat is használhatunk erre a célra. Üzleti és ingyenes tűzfalak egyaránt megfelelnek, a ***Linux iptables/Netfilter*** is támogatja a ***VPN*** protokollokat. Fontos megjegyezni, hogy amikor a cikkben csatornát említek, mindig titkosított csatornára gondolok. Igen tudom,



1. ábra Távoli elérésű VPN egy távoli rendszert kapcsol a hálózathoz

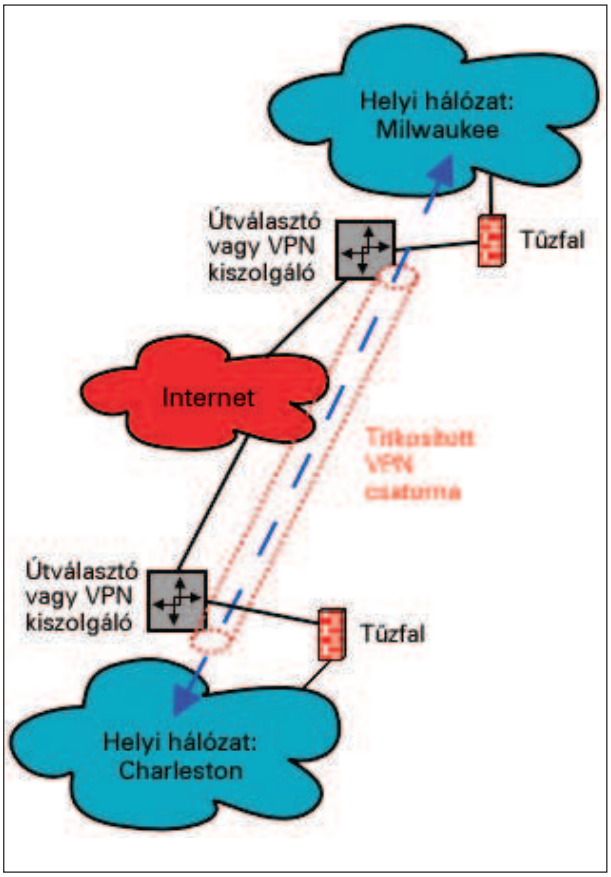
elméletileg a csatorna csak annyit jelent, hogy az egyik adatfolyamot a másikba csomagoljuk. Azonban az egész ***VPN*** lényege a titkosítás, ezért ebben a környezetben a csatorna titkosítást jelent.

A második ***VPN*** alkalmazási lehetőség, amikor két hálózat között titkosított ponttól-pontig kapcsolatot létesítünk, valamilyen nem megbízható médiumon keresztül. Míg a távoli elérésű ***VPN***-ek ügyfél-kiszolgáló modellt alkalmaztak, a ponttól pontig csatornák ***egyenrangú (peer-to-peer)*** szerkezetben dolgoznak. A ***ponttól-pontig típusú VPN szerkezetet*** a *2. ábra* mutatja be.

A ponttól pontig *VPN* felállásban gyakran alkalmazunk útvonalválasztókat. A *Cisco IOS* útvonalválasztó operációs rendszer például több különféle *VPN* protokollt is támogat. A tűzfalak és a dedikált *VPN* gyűjtők/kiszolgálók felhasználhatóak *VPN* végpontként.
A *VPN* szerkezet ezzel a két problémával foglalkozik. Ezen kívül két szerkezeti kérdéssel érdemes még foglalkozni, a ***hálózati címfordítással (Network Address Translation; NAT)*** és a teljesítménnyel.
A legtöbb *VPN* protokoll esetében a *NAT* gondokat okozhat. Ugyanis a *VPN* kiszolgálóknak általában nem lehet lefordított címe. Ez az oka annak, hogy az 1. és 2. ábrában egyik *VPN* végpont sincs az üzleti *LAN* hálózatokban, kivéve a 1. ábrán, hiszen a távoli elérésű ügyfélre mindez nem vonatkozik.
A *NAT* probléma megkerülésének egyik lehetősége, ha a tűzfalat használjuk *VPN* kiszolgálónak, ezzel azonban egy másik kérdés is felmerül: a *VPN* csatornák ugyanis jelentős *CPU* időt emészthetnek fel. Amennyiben a tűzfalunknak nincsen titkosításgyorsító kártyája és nem csak néhány *VPN* csatornát szolgálunk ki, jobb ha külön *VPN* kiszolgálót használunk és nem a tűzfalat alkalmazzuk *VPN*-re.
Az alapokkal megvolnánk, lássuk a *Linux VPN* programjait.

FreeS/WAN és OpenS/WAN

Az *IPSec* protokoll, amely valójában az *Internet Protokol (IP) v6* változatűban felbukkanó biztonsági fejléceinek megvalósítása az *Ipv4* rendszer keretein belül, a legnyitottabb, legerősebb és legbiztonságosabb *VPN* protokoll mind között. Egyben a leggyakoribb is. Az *IPSec* támogatása ma már lényegében az összes számítógépi és hálózati eszköz operációs rendszer része lett. *Linux* alatt a *FreeS/WAN* és *OpenS/WAN* projekteket használhatjuk.
A *FreeS/WAN* rendszerrel korábban részletesen foglalkoztam. Dióhéjban a *FreeS/WAN Linux* rendszerünket kibővíti néhány rendszermag modullal és felhasználói programmal. Abból, hogy az *IP* protokoll a rendszermag része, sejthető, hogy a kiterjesztéseknek is a rendszermagban kell lenniük.
A *Linux 2.6* rendszermag már tartalmazza a *26sec* nevű *IPSec* modult. A *Red Hat Enterprise Linux*-ban fellelhető *Linux 2.4* rendszermagok úgyszintén tartalmazzák, itt ugyanis a *26sec* visszahozott (backported) modulját alkalmazzák. Amennyiben már van *IPSec* rendszermag modulunk, csak a *FreeS/WAN* felhasználói térben futó programjait kell telepítenünk.
A *FreeS/WAN* tetszés szerinti *Linux* terjesztésre telepíthető (az én kedvencem, a *SuSE*, eleve tartalmazza). Ugyanakkor a *FreeS/WAN Projekt* mostanában alakul át, ezért ha a terjesztésünk nem tartalmazza a *FreeS/WAN* rendszert és forrásból kell telepítenünk, jobban járunk ha az *OpenS/WAN* rendszert telepítjük.
Az *OpenS/WAN* projektet a *FreeS/WAN* fejlesztőinek azon csoportja indította útjára, akik nem voltak elégedettek a *FreeS/WAN* projekt alakulásával. Így amikor a *FreeS/WAN* befejeződött, az utódja az *OpenS/WAN* lett. Valószínűleg a nagyobb *Linux* terjesztők hamarosan lecserélik saját *FreeS/WAN* csomagjaikat az *OpenS/WAN* változatra. A legfrissebb *OpenS/WAN* forráskódot az *OpenS/WAN* weblapról tölthetjük le (lásd a hálózati forrásokat).



2. ábra Ponttól-pontig VPN rendszerek két hálózatot kötnek össze

A *FreeS/WAN* és *OpenS/WAN* előnyei:

- ***Kiforrottság***: az egyik legöregebb *Linux VPN* technológia.
- ***Biztonság***: Az *IPSec* ellenálló, hatékony és jól megtervezett protokoll.
- ***Együttműködés***: Más operációs rendszerek valószínűleg rendelkeznek *IPSec* ügyféllel amely együtt tud működni a *Free/OpenS/WAN* rendszerünkkel.
- ***Rugalmasság***: Az *IPSec* egyaránt kiváló ügyfél-kiszolgáló és pontól-pontig *VPN* rendszerekhez.

Hátrányai:

- ***Összetettség***: Az *IPSec* viszonylag nehezen érthető, és digitális hitelesítésre van szüksége.
- ***Erő:*** ha mindössze annyit szeretnénk elérni, hogy a távoli felhasználók az egyik belső rendszerünk alkalmazását elérjék, az *IPSec*-et használni az tipikusan az ágyúval verébre lövöldözés esete. Az *IPSec*-et arra tervezték, hogy egész hálózatokat kapcsoljon össze.

Összefoglalva, ha a cikk elolvasása után még mindig nem tudjuk biztosan melyik *VPN* megoldást kellene választanunk, javaslom alapesetben maradjunk a *FreeS/WAN* vagy *OpenS/WAN* megoldásnál. Az *IPSec* messze a legkiforrottabb és legbiztonságosabb *VPN* technológia *Linux* alatt. Véleményem szerint ezek az előnyök bőségesen ellensúlyozzák az összetettség miatti hátrányt. A csomagok beállításával és használatával kapcsolatos további információkat a *FreeS/WAN* és *OpenS/WAN* weblapjain találunk.

OpenSSH

Az ember hajlamos azt hinni, hogy az *OpenSSH* egyszerűen csak egy távoli bejelentkezési eszköz. Csakhogy az *SSH* protokoll nem csak a héj (shell), hanem bármilyen, egyetlen *TCP*-kaput használó szolgáltatás átvitelét támogatja, mégpedig biztonságos csatornán, a `-L` és `-R` kapcsolók segítségével.
Például, tegyük fel van egy biztonságos héj (secure shell) kiszolgálónk a tűzfal mögötti de nyilvánosan elérhető *DMZ* hálózatunkon, valamint egy *Microsoft SQL* kiszolgálónk a belső hálózaton. Ha készítek egy tűzfal szabályt, amely a *MS-SQL* műveleteket engedélyezi az *SSH* kiszolgáló és a *MS-SQL* kiszolgáló között és az *SSH* kiszolgálóm engedélyezi a kaputovábbítást, létrehozhatok egy *SSH* csatornát egy tetszőleges távoli gép és az *SSH* kiszolgálóm között amelyen keresztül a távoli adatbázis ügyfelek lekérdezései először az *SSH* kiszolgálóhoz kerülnek, majd onnan a *MS-SQL* kiszolgálóhoz továbbítódnak. A távoli gépemen kiadott *SSH* parancs a következőképpen nézne ki:

```
bash-#> ssh -L 11433:ms-sql.server.name:1433
➥myaccount@remote.ssh-server.name
```

ahol az `ms-sql.server.name` a *MS-SQL* kiszolgáló neve vagy *IP* száma, a `remote.ssh-server.name` pedig *DMZ*-sített *SSH* kiszolgáló neve vagy *IP* száma.
Még *PPP*-t is küldhetünk *SSH*-n keresztül, ami lényegében azonos feladatot lát el, mint az *IPSec,* azaz a két hálózat között minden adatot továbbit. Mindazonáltal ez az egyik legkevésbé hatékony megoldás; Sokkal több adminisztrációt igényel mint a cikkünkben bemutatott egyéb eszközök és módszerek.
Összefoglalva, az *OpenSSH* leginkább egy adott gépen futó, adott alkalmazás adatainak átküldésére használható; ilyen felállásban távoli-elérésre és ponttól pontig *VPN* megoldásokban is használható. Kevésbé használható ugyanakkor a távoli hálózatok vagy felhasználók összes adatának továbbítására.
Az *ssh* és *sshd_config* kézikönyvoldalakon további információkat találunk az *OpenSSH* kaputovábbítási képességeiről.

Stunnel

Az *Stunnel* nevű *SSL* csomagoló lényegében az *SSH* kaputovábbítással azonos képességeket nyújt. A legtöbb mai *Linux* terjesztésben alapcsomag.
Az *Stunnel* és az *SSH* közötti főbb különbségek, hogy az *Stunnel* sokkal korlátozottabb; kizárólag titkosított kaputovábbításra lehet használni. Ezen felül, mivel az *Stunnel* tulajdonképpen valamiféle előlap az *OpenSSL*-hez, az *Stunnel* használatához digitális bizonyítványokat kell telepítenünk, ami elég sokat ront az egyszerűségén. Egyéb tekintetben *VPN* eszközként az *Stunnel* az *OpenSSH*-hoz hasonló korlátozásokkal használható.
Az *Stunnel* beállításával és használatával kapcsolatos további információkat az *stunnel* kézikönyv oldalon, a *Stunnel* weblapon, no és a *„Kódolatlan szövegekkel dolgozó alkalmazások feltámasztása az Stunnel segítségével" (Linuxvilág 2004. október)* című korábbi cikkemben találhatunk.

OpenVPN

Az *OpenVPN* egy *SSL*/*TLS*-alapú felhasználói *VPN* eszköz, amely becsomagolja az összes forgalmat hagyományos *UDP* és *TCP* csomagokba két *VPN* végpont között (hagyományos szó itt olyan értelemben értendő, hogy a rendszermag *IP* veremben nincs szükség változtatásokra).
Az *OpenVPN* azért jött létre, mert a szerzője *James Yonan* szerint a világnak az *IPSec*-nél egyszerűbb megoldásra is szüksége van.
Minthogy semmilyen speciális rendszermag módosításra nincs szükségünk, az *OpenVPN* teljes egészében a felhasználói térben fut, így sokkal könnyebb az operációs rendszerek közötti átvitele mint az *IPSec* megoldásoké. Ezen kívül az *OpenSSL* könyvtárak alkalmazásával, az *OpenVPN*, akárcsak az *Stunnel*, a lehető legkevesebb felesleges újrafelfedezéssel oldja meg a problémát. A saját készítésű titkosítórendszerekkel szemben (ilyen a *CIPE*, *tinc* és *VPN* csomag, lásd alább), az *OpenVPN* valamennyi kritikus műveletét az *OpenSSL* végzi. Persze maga az *OpenSSL* sem hibátlan, de a biztonsági hiányosságok tekintetében folyamatosan a figyelem középpontjában áll és az *Open Source* közösség legkiválóbb titkosítási programozói tartják karban.
Az *OpenVPN* jó választás pontól-pontig *VPN* készítéshez, de a 2.0-ás verzió előtt (amely 2004 novemberében még béta változatban volt csak elérhető), az *OpenVPN* korlátozott képességekkel rendelkezik, ugyanis csak egyetlen csatornát tud átvinni egy adott kapunk. Amennyiben az *OpenVPN*-t távoli-elérésű *VPN* csatornaként szeretnénk használni tíz különféle felhasználóhoz, tíz külön *OpenVPN* fogadót kell indítanunk, mindegyiket saját *UDP* kapuval. Tehát fel kell használnunk az *UDP 10201*, *UDP 10202* és *UDP 10203* valamint még hét további kaput. Ezért ha a *OpenVPN*-t tényleg távoli *VPN* elérésre szeretnénk alkalmazni és nem csak egy két felhasználónk van, sokkal jobban járunk az *OpenVPN 2.0* változattal (még ha béta állapotú is).
Az *OpenVPN* beépítve megtalálható a *SuSE Linux 9.1* rendszerben és valószínűleg egyéb terjesztésekben is.
Az *OpenVPN* weblapján megtaláljuk a beállítási információkat és a legfrissebb *OpenVPN* programot.

PoPToP and the Linux PPTP Client

Az *IPSec* nem az egyetlen alacsony szintű *VPN* protokoll az Interneten. A *Microsoft Point-to-Point Tunneling Protocol (PPTP)* megoldásnak is vannak követői, leginkább azért mert ez a rendszer vált a *Microsoft* kiszolgáló operációs rendszer szabványává a *Windows NT 4.0* óta, valamint mivel az *IPSec*-el ellentétben amely csak *IP* csomagokon keresztül tud csatornázni, a *PPTP* nem csak *IP* csomagokon de más protokollokon például *NETBEUI* vagy *IPX*/*SPX* rendszeren keresztül is működik.
A *Linux* alatt két *PPTP* változat létezik, a kiszolgáló oldali *PoPToP* és az ügyfél oldali *Linux PPTP*.
Bár ha nem *IP* protokollokon szeretnénk csatornázni igen hasznos lehet és általános minden *Windows* kiszolgáló környékén megtalálható, a *PPTP*-nek van egy nagy hátránya. Amikor *Bruce Schneier* és *Dr Mudge* megvizsgálták a *Windows NT 4. PPTP* megoldását 1998-ban, komoly biztonsági hiányosságokat fedeztek fel, amelyeket csak részben orvosolt a nem sokkal később kiadott *MSCHAPv2* javítás.

Az *MSCHAP* azonosítási protokoll, amelyre a *PPTP* épül, *Schneier* és *Mudge* által talált legrosszabb biztonsági hibák forrásának bizonyult. *Schneier* honlapján megtekinthetjük a vizsgálatuk eredményét (lásd a forrásokat).
Schneier és *Mudge* a *Windows NT 4.0* rendszert vizsgálták; de mi a helyzet a *Linux PoPToP* kiszolgálóval? A *PoPToP* weblap szerint (a *„PoPToP Kérdések és Válaszok”* fejezetben): *„A PoPToP ugyanolyan biztonsági problémákkal küszködik, mint az NT kiszolgáló (ez azért van mert Windows ügyfelekkel dolgozik).”*
Nem javaslom a *PPTP* használatát, hacsak nem tudjuk a *PPTP* kiszolgálónkat és valamennyi *PPTP* ügyfelünket rábírni az *MSCHAPv2* használatára (sajnos nem minden *Windows* változat támogatja a *MSCHAPv2*-t) és akkor is csak abban az esetben ha valami olyasmit csinálunk amit egyszerűen nem lehet megoldani *IPSec*-el. Az *IPSec* sokkal jobban megtervezett és bizonyítottan biztonságosabb. Ezen kívül a nem *IP* alapú hálózati protokollok ma már nem olyan létfontosságúak mint valaha; a *Windows* és a *Novell Netware* bármit meg tud csinálni *IP*-n keresztül.
Senki ne értse félre, nem állítok olyasmit, hogy „ne használd a PPTP-t mert béna”. Mint előző hónapban kifejtettem, a biztonság , a kockázatelemzésről szól, és nem valamiféle utópisztikus, tökéletes biztonság kereséséről. Miután elolvastuk *Schneier* és *Mudge* vitáját, a *Microsoft* válaszát és a *MSCHAPv2*-t, majd gondosan megvizsgáltuk a vállalatunk igényeit és képességeit, elképzelhető, hogy úgy döntünk, hogy a *PPTP* elfogadható kompromisszumot jelent a biztonság és a funkciók terén – csak aztán senki ne mondja hogy ***én*** javasoltam!

Egyéb Linux VPN csomagok

Három további *Linux VPN* eszközt érdemes még megemlíteni itt, hiszen néha láthatunk rájuk hivatkozásokat. Kettő használatát nem szívesen javaslom, a harmadikban nem vagyok biztos.
A *CIPE* és a *vtun* lényegét tekintve azonos az *OpenVPN* rendszerrel. A forgalmat *UDP* vagy *TCP* csomagokba zárják. Az *OpenVPN*-el ellentétben azonban házilag fejlesztett titkosítási rendszereket használnak az *OpenSSL* helyett. Pontosabban olyan hagyományos titkosítási eljárásokat használnak mint a *Blowfish* vagy az *MD5*, de saját megvalósítással (folyamat-kulcs készítés, felhasználó azonosítás és egyebek). Minthogy a titkosítás programozásban éppen a megvalósítás a legnehezebb rész, igen veszélyes lehet, és lám, *Peter Gutmann* titkosítási szakember súlyos biztonsági hibákat talált a *CIPE* és *vtun* rendszerekben.
Amennyire tudom, egyik esetben sem javították *Gutmann* azonosította hibákat. Ráadásul úgy tűnik sem a *CIPE* se a *vtun* rendszert nem fejlesztik már aktívan (a *CIPE*-t biztosan nem), ami önmagában éppen elég indok, hogy messziről elkerüljünk egy biztonsági alkalmazást, kivéve, ha egy *Linux* terjesztés része, ahol a csomag karbantartói maguk készítenek foltokat. Ezekből az okokból kifolyólag nem javaslom se a *CIPE* se a *vtun* használatát.
A *tinc*, akárcsak a *CIPE* és a vtun, saját kódolási megoldást használ a *VPN* forgalom titkosított *UDP* csomagokba zárására. És akárcsak a fenti csomagokban, *Gutmann* a *tinc*-ben is talált hibákat, a korábban említett vizsgálata során. A *tinc* fejlesztői azonban a *CIPE* és *vtun* csapataival ellentétben hitelt érdemlő módon válaszoltak Gutmann felfedezéseire; legalábbis az én szemszögemből nézve. Ami az illeti, részemről IANAC! (IANAC = „I Am Not A Cryptographer” ; „Én Nem Vagyok Kriptográfus”.) Úgy tűnik, van valami fogalmuk róla mit is csinálnak.
Önökre bízom a *tinc* weblapjának felderítését, és *Gutmann* lapjának elolvasását (amely komoly kutatási jelentés lévén elég súlyos darab), no és pár *Google* keresés elvégzését *Gutmann* megállapításainak sorsát illetően. Ezek alapján mindenki eldöntheti, hogy a *tinc* éppen az amire szüksége van, vagy inkább nemkívánatos biztonsági kockázatot jelentene a könnyen elérhető *OpenS/WAN* és *OpenVPN* helyett.

SSL-VPN

Végül, ejtsünk néhány szót a számos üzleti *VPN* termékben megjelenő népszerű új megközelítésről: az *SSL-VPN* rendszerről. Az *SSL-VPN* gyakorlatilag pontosan úgy működik mint az *Stunnel* és az *SSH* kaputovábbítás. A hálózati tranzakciókat szolgáltatás- és kiszolgáló-alapon viszi át, és nem felső kapcsolati szinten. A többi megközelítéssel szemben az *SSL-VPN* termékek a végfelhasználónak központosított webes felületet nyújtanak ahol a *VPN* rendszerben kezelt összes kiszolgáló/szolgáltatás hivatkozásként megjelenik. Amikor a felhasználó rákattint egy hivatkozásra, általában egy *Java* kisalkalmazás töltődik le, amely az alkalmazás ügyfélprogramjaként működik.
Valamennyi *SSL-VPN* kiszolgáló termék amellyel találkoztam, üzleti megoldás volt, de mivel az ügyféloldal általában *Javaban* íródott és rendszerfüggetlen, a *Linux* rendszerek is dolgozhatnak *SSL-VPN* ügyfelekként.

Összefoglalás

A *FreeS/WAN* és *OpenS/WAN* (lehetőség szerint az utóbbi) és az *IPSec* nyújtják valószínűleg a legbiztonságosabb és leghatékonyabb *VPN* megoldást *Linuxos* eszközökön.
Az *OpenVPN* egyszerűbb, ugyanakkor kevésbé alaposan vizsgált alternatívát jelenthet. Az *OpenSSH* és az *Stunnel* jó csomagoló megoldást jelenthetnek amikor az előzőek használata túlzás lenne. Más *Linux VPN* eszközök is elérhetőek, de egyesek bizonyítottan veszélyesek, mások esetében pedig a zsűri még nem döntött. Melyik *VPN* eszköz lesz a legmegfelelőbb számunkra? Természetesen ezt nem tudom megmondani anélkül, hogy az adott igényeket és erőforrásokat ismerném. De remélem ez a kis összefoglaló legalább segít elindulni.

Linux Journal 2005. február, 130. szám

A cikk forrásai: ➲ ***www.linuxjournal.com/article/7923***

Mick Bauer (mick@visi.com)
Biztonsági szakember, a Linux Journal biztonsági témákkal foglalkozó szerkesztője, biztonsági tanácsadó a Minnesota állambeli Minneapolisban található Upstream Solutions LLC Inc.-nél.