WLAN-ok védelme WPA és FreeRADIUS alkalmazásával (2. rész)

A vezeték nélküli hálózatok védelmi eszközeinek új generációja nem csupán a WEP hiányosságait tünteti el, de lehetővé teszi a vezeték nélküli felhasználók RADIUS kiszolgálóval végzett hitelesítését is.

A múlt hónapban ismertettem a vezeték nélküli *LAN*-ok új biztonsági protokollját, a *WPA*-t (***Wi-Fi Protected Access***, ***Wi-Fi védett elérés***). Megmutattam, hogy erőteljes és rugalmas hitelesítéssel, továbbá dinamikus titkosítással és kulcsegyeztetéssel hogyan foltozza be a *WEP* protokoll biztonsági réseit. Szó volt arról is, hogy a *WPA* összetevő-protokolljai, köztük a ***802.1x***, valamint az *EAP* és a *RADIUS* különféle változatai milyen kapcsolatban állnak egymással. Ebben a hónapban megmutatom, hogy *FreeRADIUS* és *OpenSSL* használatával hogyan helyezhetjük üzembe *WPA*-ra vagy egyéb ***802.1x*** alapú megoldásra épülő környezetünk hitelesítő kiszolgálóját.

Gyors áttekintés

A *WPA*, mint remélem, sokan emlékeznek, modulárisabb, mint a *WEP*. Míg *WEP* használatakor a hitelesítés és a titkosítás egy az összes ügyfél által közösen használt titkos kulcs segítségével történik, *WPA* használatakor a hitelesítés alapesetben a ***802.1x*** protokoll használatával folyik. A *WEP*-re sokban hasonlító előre megosztott kulcs (***pre-shared key, PSK***) mód alkalmazása egy másik lehetőség. A *WPA* esetében az egyes ügyfelek egyedi titkosító kulcsait a hozzáférési pont állítja elő, és rendszeresen frissíti.

A ***802.1x*** egy rugalmas, az *EAP*-ra (***Extensible Authentication Protocol***, ***bővíthető hitelesítő protokoll***) épülő hitelesítő protokoll. A *WPA*-képes termékek az *EAP* számos különböző változatát támogatják, köztük az *EAP-TLS*-t és a *PEAP*-t. Aki a ***802.1x*** elhagyását és a *WPA* egyszerűbb, *PSK* módban történő használatát választja, amely ugyan biztosítja a titkosító kulcsok dinamikus előállítását, ám a hitelesítő adatokat hozzáférhetően, nyílt szövegként továbbítja, annak csupán annyit kell tennie, hogy azonos előre megosztott kulcsot állít be a hozzáférési ponton és a vezeték nélküli ügyfeleken.

Ha viszont a ***802.1x*** jóval erőteljesebb hitelesítési eljárásainak alkalmazásával a *WPA* teljes tudását ki akarjuk használni, akkor szükségünk van egy *RADIUS* kiszolgálóra. Erre a célra léteznek kereskedelmi megoldások is, mint például a *Funk Software Steel Belted RADIUS* programja, de annak sem kell elkeseredni, aki a nyílt forrású programokat pártolja, a *FreeRADIUS* ugyanis az *EAP* minden fontosabb változatát támogatja, és ugyanolyan üzembiztos és biztonságos. Nézzük, hogyan bírhatjuk munkára.

A használati környezet

Természetesen nincs elég helyem arra, hogy a *FreeRADIUS* és a ***802.1x*** együttes használatának minden lehetséges módját ismertessem, sőt, még a vezeték nélküli alkalmazásokat sem tudom mindenre kiterjedően tárgyalni. Nézzünk tehát egy példakörnyezetet, amelyet az alábbi eljárásokkal fogunk létrehozni.

A *WPA* használatba vétele kapcsán a legfontosabb döntés a használni kívánt *EAP*-változat kiválasztása. E tekintetben nemcsak a *RADIUS* kiszolgáló alkalmazás, de az ügyfelek képességei is jelenthetnek korlátozást. A vezeték nélküli hozzáférési pont – érdekes módon – *EAP*-független, feltéve persze, hogy támogatja a ***802.1x***-et és/vagy a *WPA*-t. Egyszerűen közvetíti az *EAP*-forgalmat az ügyfelek és a kiszolgálók között, az *EAP* egyik altípusának kifejezett támogatására sincs szüksége. Az, hogy az ügyfélrendszer pontosan mit képes támogatni, a rajta futó operációs rendszertől és a vezeték nélküli hozzáférést biztosító hardvereszköztől egyaránt függ. Egy ***Microsoft Windows XP*** rendszer ***Intel Pro/2100 (Centrino)*** lapkakészlettel például támogatja az *EAP-TLS*-t és a *PEAP*-t, de az *EAP-TTLS*-t nem. Ha ***Linuxot*** és ***wpa_supplicantot*** (lásd az internetes forrásokat) használunk, akkor jóval szélesebb körű választási lehetőséget kapunk.

Példámban az *EAP-TLS* használatát fogom feltételezni.

Az *EAP-TLS* alkalmazásához az ügyfeleknek tanúsítványokkal kell rendelkezniük, amihez viszont szükségünk lesz egy ***hitelesítő szervezetre (certificate authority, CA)***. A nehézségek ellenére mégis érdemes az *EAP-TLS* mellett dönteni. Először is, széles körben támogatott. Másodszor, a ***TLS (X.509 tanúsítvány)*** alapú hitelesítés erőteljes biztonságot nyújt. Harmadszor, valójában nem kell túlságosan sok munka ahhoz, hogy *OpenSSL* segítségével összeállítsuk saját *CA*-nkat. Példakörnyezetünkben tehát egy *EAP-TLS*-t alkalmazó, *Windows XP*-t futtató ügyfél fog csatlakozni egy *WPA*-képes hozzáférési ponthoz. A hozzáférési pont egy *Linuxon* üzemelő *FreeRADIUS 1.0.1* kiszolgálónak fogja továbbadni a hitelesítési feladatokat.

1. kódrészlet Az openssl.cnf módosítása optimális tanúsítványkészítéshez

```
# Először módosítjuk a CA a CA_default részben szereplő gyökér elérési útját
# a létrehozni kí ánt CA-nak megfelelően
[ CA_default ]
dir             = ./micksCA               # Mindent itt tárolunk
# Az alábbi sorok kicsit lejjebb  annak az openssl.cnf-ben:
countryName_default             = US
stateOrPro inceName_default     = Minnesota
0.organizationName_default      = Industrial Wiremonkeys of the World
```

A FreeRADIUS beszerzése és telepítése

SuSE 9.2, ***Fedora Core 3*** és ***Red Hat Enterprise Linux*** alá külön, saját ***FreeRADIUS RPM*** csomag létezik, ***freeradius*** névvel. A ***Debian Sarge (Debian-testing)*** ugyanilyen nevű ***DEB*** csomaggal rendelkezik. ***Red Hat***, ***Fedora*** és ***Debian-testing*** alatt további csomagok is rendelkezésünkre állnak, ha ***MySQL*** adatbázist szeretnénk használni a hitelesítésre. Emellett a ***Debian-testing*** néhány további szolgáltatást is kínál, ezek újabb csomagokba kerültek. Mind a négy terjesztésre igaz azonban, hogy magához a ***802.1x*** alapú hitelesítéshez csupán az alap ***freeradius*** csomagra van szükség. Ha kedvenc ***Linux***-terjesztésünkhöz nincs ***FreeRADIUS*** csomag, vagy annak változata nem elég új az igényeinknek, akkor töltsük le a legújabb ***FreeRADIUS*** forráskódot a fejlesztők webhelyéről (lásd a forrásokat).

A ***FreeRADIUS*** lefordítása egyszerű, a megszokott `./configure - make - make install` eljárással történik. Aki még nem nagyon fordítgatott programokat, az a forráscsomag ***INSTALL*** fájljában részletesebb útmutatást is talál. A `configure` és a `make` parancsot lehetőleg normál felhasználóként adjuk ki, root jogokra csak a `make install` futtatásához van szükség.

Megjegyzem, hogy alapesetben a parancsfájl a ***/usr/local*** könyvtár alkönyvtáraiba telepíti a ***FreeRADIUS***-t. Mivel a ***Makefile*** eltávolításra nem alkalmas, javaslom a telepítési könyvtár változatlanul hagyását, így ugyanis – ha valamiért szükségtelenné válna – később könnyebb eltávolítani a ***FreeRADIUS***-t.

Hitelesítő szervezet létrehozása

Mielőtt megadnánk a ***FreeRADIUS*** beállításait, létre kell hoznunk néhány tanúsítványt. A tanúsítványok létrehozásához viszont szükség van a hitelesítő szervezetre. ***Linux Server Security*** című könyvem 5. fejezete tartalmaz egy ***„How to Become a Small-Time CA" (Kisméretű CA üzembe helyezése)*** című részt, ami részletesen tárgyalja a témát; itt most csak néhány szóban foglalnám össze az eljárás menetét.

Először is, mi az a ***CA***, és hol kell elhelyezkednie? A ***CA*** egy a nyilvános kulcs infrastruktúra gyökereként üzemelő rendszer. Központi hatóság, amely digitális aláírások alkalmazásával jótáll a szervezeten belül kibocsátott tanúsítványok hitelességéért. Rendszeres időközönként ***tanúsítvány visszavonási listákat (certificate revocation list, CRL)*** bocsát ki, ezek azokat a tanúsítványokat sorolják fel, amelyekért a CA többé nem szavatol. Ilyenek például a vállalattól kilépett munkatársak vagy az üzemen kívül helyezett kiszolgálók tanúsítványai.

Egyik feladat ellátásához sincs szükség arra, hogy a ***CA*** ténylegesen kiszolgálóként üzemeljen, sőt, jobb is, ha nem az. Egy ***CA*** akkor megbízható, ha gondosan védjük az illetéktelen hozzáférésektől, visszaélésektől. Saját ***CA***-imat éppen ezért egyre inkább olyan rendszerekre telepítem, amelyekkel csak időszakosan lépek fel a hálózatra, például ***VMware*** virtuális gépekre.

Lehetséges, hogy már rendelkezünk ***CA***-val, segítségével webkiszolgálók, ***stunnel*** vagy egyéb, ***TLS***-t használó alkalmazásokhoz is létrehozhattunk már tanúsítványokat. Ha ez a helyzet, akkor a ***CA*** a ***WPA***-hoz is megfelel. Ha nincs ilyenünk, akkor lássuk, hogyan helyezhetjük üzembe a ***CA***-t. Először ellenőrizzük, hogy a kiszemelt rendszerre telepítve van-e az ***OpenSSL***. Az ***OpenSSL*** minden ismertebb ***Linux***-terjesztésnek része, ahogy a ***FreeBSD***, az ***OpenBSD*** stb. is tartalmazza. Az ***OpenSSL*** meglétét a legegyszerűbben a `which openssl` paranccsal ellenőrizhetjük, mely megadja, hogy hová van telepítve gépünkön az ***OpenSSL*** – ha telepítve van. Lépjünk át abba a könyvtárba, ahol rendszerünk az ***OpenSSL*** beállító és tanúsítványfájljait tárolja. ***SuSE*** alatt ez a ***/etc/ssl***, de a pontos könyvtár terjesztésenként változhat. Ha megkeressük az `openssl.cnf` fájlt, akkor valószínűleg jó helyre fogunk kerülni.

Most nyissuk meg valamilyen szövegszerkesztőben az ***openssl.cnf*** fájlt. Néhány alapbeállítást át kell írnunk, így később gyorsabb lesz a tanúsítványok előállítása. Az *1. kódrészlet* az ***openssl.cnf*** módosítandó sorait tartalmazza. Ezután a ***CA*** tanúsítványkészítő parancsfájlját kell átírnunk, a ***CA*** alapértelmezett ***demoCA*** gyökérkönyvtára helyett az ***openssl.cnf*** fájlban a `dir` változónál választott könyvtárat kell megadnunk itt is. Én a ***CA.sh*** parancsfájlt használom, ez ***SuSE*** rendszereken a ***/usr/share/ssl/misc*** könyvtárban található; más rendszereken lehetséges, hogy máshova kerül. A módosítandó sor a következő:

`CATOP=./micksCA`.

Miután átírtuk a fájlt, lépjünk vissza az ***SSL*** beállítások könyvtárába, ami például a ***/etc/ssl*** lehet. Innen indítsuk el a ***CA.sh*** parancsfájlt a `-newca` kapcsolóval. Például: `/usr/share/ssl/misc/CA.sh -newca`. Ekkor módot kapunk új gyökértanúsítvány létrehozására és a hozzá tartozó titkos kulcs jelszavának megadására. Lehetőleg nehezen kitalálható jelszót válasszunk, és jegyezzük fel valamilyen biztonságos helyre; ha elveszítjük, képtelenek leszünk használni a ***CA***-t. Miután a parancsfájl végzett, az ***SSL*** beállításkönyvtárban lennie kell egy új könyvtárnak, példánknál maradva `micksCA` névvel. Ennek a könyvtárnak a gyökérszintjén

2. kódrészlet Az xpextensions fájl tartalma

```
[ xpclient_ext]
extendedKeyUsage = 1.3.6.1.5.5.7.3.2
[ xpserver_ext ]
extendedKeyUsage = 1.3.6.1.5.5.7.3.1
```

találjuk új *CA*-nk nyilvános tanúsítványát, a fájl neve alapesetben ***cacert.pem***. Mint később még lesz róla szó, ezt a fájlt át kell másolnunk a ***FreeRADIUS*** kiszolgálóra és minden egyes vezeték nélküli ügyfélre. Ha ***Windows XP*** ügyfelekkel dolgozunk, akkor a tanúsítványok létrehozása előtt egy további lépést is végre kell hajtanunk. A ***Windows XP*** bizonyos jellemzőket elvár az ügyfél- és a kiszolgálótanúsítványoktól egyaránt, ezért a 2. *kódrészletben* látható sorokkal létre kell hoznunk egy ***xpextensions*** nevű fájlt.
Az ***xpextensions*** fájlra a később szereplő ***OpenSSL*** parancsok egy része hivatkozni fog. Az ***openssl.cnf*** fájllal azonos könyvtárban kell lennie.

Az EAP-TLS működése

EAP-TLS használatakor a vezeték nélküli ügyfél és a ***RADIUS*** kiszolgáló kölcsönösen hitelesítik egymást. Mindkettő bemutatja saját tanúsítványát, majd kriptográfiai eszközökkel ellenőrzik, hogy a tanúsítványok rendelkeznek-e a vállalat hitelesítő szervezetének aláírásával. Tulajdonképpen ez a hitelesítés kezelésének egy egyszerű és elegáns módja. Miután telepítettük a *CA* nyilvános tanúsítványát a ***FreeRADIUS*** kiszolgálóra, az ügyfelek egyéb adatait, mint a felhasználónév és a jelszó, már nem kell kézzel megadnunk.
Persze szó sincs arról, hogy az ***EAP-TLS*** használata kevesebb munkával járna, mint a felhasználónév-jelszó alapú megoldásé, ugyanis az ***OpenSSL*** segítségével az összes felhasználóhoz létre kell hoznunk egy tanúsítványt, majd ezeket át kell másolnunk az ügyfélgépekre. Arra is ügyelnünk kell, hogy a megfelelő helyre telepítve mindenki rendelkezzen a gyökér CA tanúsítvány egy másolatával.

Tanúsítványok létrehozása

EAP-TLS használatakor a *CA* tanúsítványa mellett legalább kettő további tanúsítványra is szükségünk van, mégpedig egy kiszolgálótanúsítványra a ***FreeRADIUS*** kiszolgálóhoz, továbbá egy-egy ügyféltanúsítványra a hálózat minden vezeték nélküli ügyfeléhez. A tanúsítványok létrehozása három lépésből áll:

1. Aláírási kérvény létrehozása, ez lényegében egy aláíratlan tanúsítvány.
2. Az aláírási kérvény aláírása a *CA* kulcsával.
3. Az aláírt tanúsítvány átmásolása arra az állomásra, amelyik használni fogja.

A kiszolgálótanúsítvány aláírási kérvényét az ***OpenSSL*** `req` parancsával készíthetjük el:

```
$ openssl req -new -nodes -keyout
➥kiszolgalo_kulcs.pem -out kiszolgalo_kerveny.pem
➥-days 730 -config ./openssl.cnf
```

A parancs két fájlt hoz létre, a tényleges kérvényt, vagyis az aláíratlan tanúsítványt tartalmazó `kiszolgalo_kerveny.pem`-et, valamint a jelszó nélküli titkos kulcsot tartalmazó `kiszolgalo_kulcs.pem`-et. Előbb persze meg kell adnunk szervezetünk ***országkódját (Country Code)***, ***államát (State)*** stb., ezeknél sokszor az ***openssl.conf*** fájlban megadott alapértékeket is használhatjuk. A ***közös névvel (Common Name)*** már más a helyzet. Amikor a gép bekéri, írjuk be kiszolgálónk teljesen minősített tartománynevét, mint például ***kiszolgalo.wiremonkeys.org***.
Ezután a *CA* kulcsot használva, az ***OpenSSL*** `ca` parancsával aláírhatjuk a kérvényt:

```
$ openssl ca -config ./openssl.cnf \
-policy policy_anything -out
➥kiszolgalo_tanusitvany.pem \
-extensions xpserver_ext -extfile ./xpextensions \
-infiles ./kiszolgalo_kerveny.pem
```

A parancs beolvassa a `kiszolgalo_kerveny.pem` fájlt, bekéri a *CA* kulcsához tartozó jelszót, majd a `kiszolgalo_tanusitvany.pem` fájlba elmenti a kérvény aláírt változatát és a hozzá tartozó titkos kulcsot. Felhívnám a figyelmet a `-extensions` és `-extfile` kapcsolóra; ezek miatt hoztuk létre korábban az `xpextensions` fájlt.
Nyissuk meg valamilyen szövegszerkesztőben az aláírt tanúsítványt, és minden a `-----BEGIN CERTIFICATE-----` sor előtt lévő dolgot töröljünk belőle. Másoljuk össze egyetlen fájlba a tanúsítványt és a kulcsunkat:

```
$ cat kiszolgalo_kulcs.pem
➥kiszolgalo.tanusitvany.pem > \
kiszolgalo_kulcstanusitvany.pem
```

Van tehát kulccsal kiegészített kiszolgálótanúsítványunk, ezt kell átmásolnunk a ***FreeRADIUS*** kiszolgálóra. Titkos kulcsa nincs jelszóval védve, ezért miután minden a helyére került, minden felesleges másolatát töröljük.
Most az ügyféltanúsítvány aláírási kérvényét kell összeállítanunk. Az erre a célra szolgáló ***OpenSSL***-parancs hasonló a kiszolgálótanúsítvány létrehozására használthoz:

```
$ openssl req -new -keyout ugyfel_kulcs.pem \
-out ugyfel_kerelem.pem -days 730 -config
➥./openssl.cnf
```

Mint látható, az aláírási kérvényt és a kulcsot rendre az `ugyfel_kerveny.pem` és az `ugyfel_kulcs` fájlba írjuk. A kiszolgáló aláírási kérvényével ellentétben a `-nodes` kapcsoló itt elmaradt, ezért a parancs futtatásakor meg kell adnunk a tanúsítvány titkos kulcsának titkosításához használt jelszót.
A következő lépésben aláírjuk az ügyféltanúsítvány aláírási kérvényét:

```
$ openssl ca -config ./openssl.cnf \
-policy policy_anything -out
➥ugyfel_tanusitvany.pem \
-extensions xpclient_ext -extfile ./xpextensions \
-infiles ./ugyfel_kerveny.pem
```

Ismétlem, a parancs hasonló a kiszolgáló esetében használthoz, kivéve azt, hogy a `-extensions` parancs az

`xpextensions` fájl egy másik szakaszára hivatkozik. Ha az ügyfeleken ***Linux*** fut, akkor a `kiszolgalo_tanusitvany.pem` fájlnál látott módon törölni kell belőle a felesleges részeket. A tanúsítványt és a kulcsot külön fájlban is hagyhatjuk, de össze is fűzhetjük. Ezután másoljuk az ügyféltanúsítvány fájlját vagy fájljait a linuxos ügyfélgépre.
Ha egy tanúsítványt ***Windows XP***-t futtató ügyfélen szeretnénk használni, még egy lépést el kell végeznünk: ***PKCS12*** formátumúra kell hoznunk a tanúsítványfájlt. A szükséges parancs a következő:

```
openssl pkcs12 -export -in ugyfel_tanusitvany.pem \
-inkey ugyfel_kulcs.pem -out ugyfel_tanusitvany.p12
➥-clcerts
```

Meg kell adnunk az `ugyfel_kulcs.pem` jelszavát, majd az új fájl jelszavát. Ha gondoljuk, akár ugyanazt a jelszót is használhatjuk. Csábító lehetőség, hogy a jelszó begépelése helyett csupán lenyomjuk az ***Entert***, különösen, ha azt nézzük, hogy a ***Windows XP WPA***-kérvényezője csak akkor működik, ha a tanúsítványait jelszavak nélkül tároljuk. Nagyon, nagyon rossz ötlet ez, a titkos kulcsokat nem szabad védtelenül másolgatni a hálózaton keresztül. Nyomatékosan javaslom mindenkinek, hogy a jelszavakat csak akkor távolítsa el, ha a fájlokat már biztonságosan átmásolta a ***Windows XP***-t futtató ügyfelekre.
Gondolom, szinte kínálja magát az alkalom, hogy ennek kapcsán szidjuk egy kicsit a Microsoftot, de el kell árulnom, hogy a linuxos `Xsupplicant` és `wpa_supplicant` szintén csak úgy működik, hogy üres jelszót adunk meg, vagy elmentjük a jelszót nyílt szövegként egy beállító fájlba. Természetesen mindez ellentétben áll az igazán biztonságos tanúsítványkezelés elveivel. Remélem, hogy hamarosan elkészülnek azokat a ***WPA***-kérvényezők, amelyek indításkor képesek bekérni a felhasználótól a jelszavát.
A létrejött fájl, példánkban az `ugyfel_tanusitvany.p12` az aláírt tanúsítványt és a hozzá tartozó titkos kulcsot egyaránt tartalmazza. Ezt kell átmásolni a ***Windows XP*** alapú ügyfélgépre.

Összefoglalás

sTelepítettük a ***FreeRADIUS***-t, létrehoztunk egy hitelesítő szervezetet, előállítottuk a kiszolgáló és az ügyfelek tanúsítványait, majd átmásoltuk őket a megfelelő gépekre. Természetesen messze nem végeztünk még. Meg kell még adnunk a ***FreeRADIUS***, a hozzáférési pont és a vezeték nélküli ügyfelek beállításait. Erre a következő alkalommal kerítünk sort. Addig is mindenkinek biztonságot kívánok!

Linux Journal 2005. május, 133. szám

A cikk forrásai: ➲ ***www.linuxjournal.com/article/8134***

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.