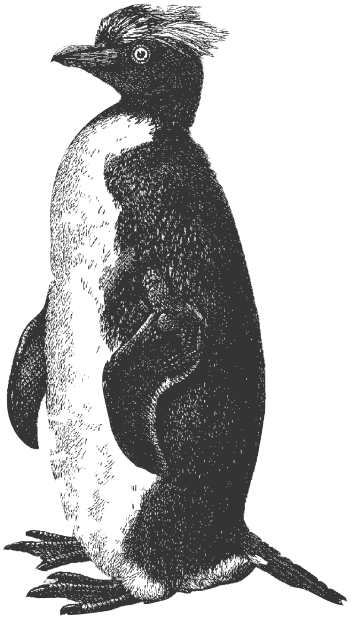
Ellenőrzés pásztázókkal: Nessus (2. rész)

Nessus: a magasabb szintre emelt biztonsági elemzés.



A múlt hónapban megkezdtük érdekes és veszélyes utunkat a pásztázás világába. Akkor az Nmap nevű sokoldalú kapupásztázó program állt írásunk középpontjában. Az Nmap segítségével a rendszergazdák, a biztonsági szakemberek (és bizony-bizony a leendő betörők is) megállapíthatják, hogy egy adott gépen milyen szolgáltatásokhoz lehet kapcsolódni.

A gép biztonsági elemzéséhez jó kiindulópontot nyújt, ha megvizsgáljuk, hogy milyen belépési pontokat kínál fel a rendszer. Vajon miként értelmezzük az Nmap által átadott ismereteket? A múlt hónapban például az egyik kipróbált pásztázás a *listán* látható kódot adta. Mit jelent ez? Rendben, azt már tudjuk, hogy a gép futtat webkiszolgálót (TCP 80), bizonyos RPC-szolgáltatásokat (UDP 111, UDP 1026), és valószínűleg Windows-megosztásokat is (UDP 137, TCP 138-139). Ezek közül a szolgáltatások közül melyik támadható? Ezen a ponton lépnek be a biztonsági pásztázók. Vállalva azt is, hogy a témában előreszaladunk, nézzük meg célpontunk Nessus-pásztázásának kimenetét (lásd *1. kép*).

Helyhiány miatt nem mutatjuk be az egész jelentést, de még ebből a rövidített változatból is jól látható, hogy a Nessus hét biztonsági lyukat, illetve kihasználható támadási pontot észlelt a célrendszerben. Ezenkívül négy további figyelmeztetést és két biztonsággal kapcsolatos megjegyzést is megjelenített.

A Nessus egyebek között megállapította, hogy ez a gép a Sambar webkiszolgálót futtatta, amelyen nem volt beállítva a rendszergazda jelszava, és a veszélyes mailit.pl CGI programot is el lehetett érni (az 1. kép nem mutat minden részletet, a hiányzó részeket el kell hinnünk). Az egész C:\ meghajtó a jelszó beállítása nélkül volt megosztva. Továbbá a Nessus még azt is felfedte, hogy a rendszer sebezhető, ugyanis jelszavas védelem esetén is védtelen maradt volna a *Null session* kapcsolattal és a *first-letter* jelszótámadásokkal szemben. Futott még egy FTP-kiszolgáló a TCP 1432-es kapun (ezt azonban az Nmap tévesen blueberry-lm szolgáltatásként értelmezte), és a TCP/IP-verem kiszámítható TCP-sorozatszámokat használt. Ezt sokféle módon ki lehet használni: például TCP-eltérítésre és IP-hamisításra.

Ez a rendszer megérett a feltörésre! Mi lehet ez a Nessus nevű halálos varázslat? És miért táncolja körbe az Nmapot a rendszer elemzése közben?

Biztonsági pásztázók

Amíg az Nmaphoz hasonló kapupásztázók feltárják, hogy mi figyel, a Nessus-szerű biztonsági pásztázók elárulják, hogy mi sebezhető. Mivel előbb érdemes megtudni, hogy mi figyel, és csak ezt követően lehet a gyenge pontokat kipuhatolni, a biztonsági pásztázók általában tartalmaznak kapupásztázót vagy kapcsolódnak egyhez.

A Nessus minden egyes pásztázás első lépéseként az Nmapot hívja meg, ezért volt félrevezető az előbb azt sugallni, hogy a Nessus sokkal jobb elemzést adott, mint az Nmap – valójában a Nessus az Nmaptól függ.

Miután a biztonsági pásztázó megállapította, hogy milyen szolgáltatások működnek, különféle ellenőrzéseket végez: meghatározza, hogy melyik programcsomag melyik változata fut, és ez sebezhető-e ismert módszerrel. Természetesen a nyomozásnak ebben a szakaszában szükséges egy jó adatbázis, amely a biztonsági lyukak leírását tartalmazza, és ezt az újabb hibák felbukkanásával egy időben frissítik. Eszményi esetben az adatbázis kézzel is szerkeszthető, azaz a felhasználó saját sebezhetőségi próbákat építhet be – az adott igényeknek és környezetnek megfelelően. Ezzel a szolgáltatással megvalósulhat az az igény, hogy a felhasználó maga is frissíthesse adatbázisát, ha a pásztázó fejlesztője ráérősen adja ki a frissítést. A Nessus magas szinten testre szabható, ellentétben számos más biztonsági pásztázóval.

A biztonsági másoló a megadott gépeken megtalálja, azonosítja és elemzi a figyelő szolgáltatásokat, ezután jelentést készít. A jobb pásztázók nemcsak a sebezhetőség tényét rögzítik, hanem részletesen elmagyarázzák az esetet és a javításra is javaslatot tesznek.

A jó biztonsági pásztázó olyan terjedelmes jelentést hoz létre, hogy esetenként a súlyos pénzeket kereső tanácsadók is ezt nyújtják be teljes körű biztonsági ellenőrzésük fő végeredményeként. Ez a gyakorlat megkérdőjelezhető ugyan, de hangsúlyozza a tényt, hogy a jó pásztázás sok adatot szolgáltat.

A Nessuson kívül akad még néhány ingyenes biztonsági pásztázó, például a VLAD és a SAINT. A Nessus azonban kiemelkedik a sorból, mert kiváltható vele néhány nagy tudású kereskedelmi termék: az ISS Internet Scanner és a NAI CyberCop Scanner.

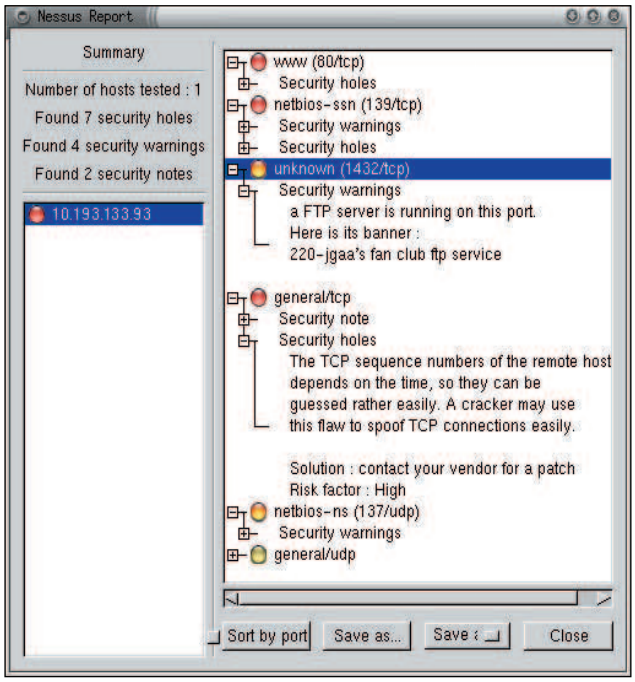
Az elsődlegesen *Renaud Deraison* és *Jordan Hrycaj* által fejlesztett Nessus a Gimp és az Apache programokkal áll egy szinten, olyan értelemben, hogy ezek az eszközök ugyanannyira vagy még jobban használhatók és rugalmasabbak, mint kereskedelmi megfelelőjük.

Mielőtt továbblépnénk, megismételjük előző havi figyelmeztetésünket: a tudás hatalom – mindent felelősségünk teljes tudatában használjunk! Az Nmap, a Nessus és a hozzájuk hasonló eszközök csak olyan rendszeren és hálózaton futtathatók, amelyek pásztázását engedélyezték számunkra. A kapupásztázás általában nem törvénytelen tevékenység, az engedély nélküli biztonsági pásztázás azonban komoly bajba sodorhat minket.

A Nessus felépítése

A Nessus két fő részből áll: a kiszolgáló végzi az összes pásztázást, az ügyféllel pedig vezérelhető a pásztázás és megnézhető a jelentés. Ez az osztott felépítés rugalmassá teszi a Nessust, így nem szükséges munkaállomásunk teljes processzoridejét a pásztázásra fordítani. Szintén lehetőség nyílik a felületek keverésére, például használhatjuk a kiszolgáló unixos változatát, és ehhez választhatjuk az X-, a Java- vagy az MS-Windows-ügyfelet. (A hirek szerint a Java-ügyfelet nem fejlesztik már tovább.)

A `nessusd` a TCP 3001-es és a TCP 1241-es kapukon várja a kapcsolódó ügyfeleket (az 1241-es kaput az Internet Assigned Numbers Authority nemrég rendelte a Nessushoz, ugyanis a 3001-es



1. kép Egy Windows 98 Nessus-pásztázása

előbb-utóbb „kimegy a divatból”.) Az ügyfélkapcsolatokat El Gamal-alapú nyilvános kulcsú eljárás alapján hitelesítik, és a titkosítást olyan titkosító kulccsal végzik, amelyet minden kapcsolatnál különkülön újra kicserél a két fél. E tekintetben a Nessus titkosító rétege (Jordan Hrycaj valósította meg a libpeks könyvtára segítségével) hasonlóan viselkedik az SSL-hez.

A Nessus ügyfélrésze, a `nessus` beállítható úgy is, hogy titkos kulcsunk jelszóval védett legyen, de akár azt is megadhatjuk, hogy ne kapcsolódjon hozzá jelszó. Előbbi esetben a nem engedélyezett felhasználók nem tudnak munkaállomásunkról csatlakozni a Nessus-kiszolgálóhoz.

Miután csatlakoztunk a kiszolgálóhoz, az felkínálja az általa támogatott bővítmények listáját (sebezhetőségi próbákat) és néhány egyéb beállítási lehetőséget. Ha belefordítottuk a Nessusba, az a lehetőség is megjelenik, hogy a pásztázás azután is folytatódjon, miután a kapcsolat megszakadt az ügyféllel (ez az úgynevezett leválasztott pásztázás – Detached Scan). Egy teljes oldalnyi lehetőségünk nyílik tudásbázis létrehozására és fenntartására, amely szintén belefordítható a programba, így ennek segítségével a pásztázások eredményei elrakhatók, valamint a gépek biztonsági állapota pásztázásról pásztázásra nyomon követhető (például lehet különbségi pásztázást futtatni).

Fontosnak tartjuk megjegyezni, hogy az utóbbi két lehetőség csak kísérleti állapotban van, ezért kisebb üzemelési gondok miatt nem fordítódnak bele a programba, csak ha kifejezetten akarjuk. A Nessus 1.2 megjelenéséig ezeket a hibákat ki fogják javítani, és ez a két módozat is teljesértékű válik a programnak. Azért említjük meg ezeket itt, mert a leválasztott pásztázás (Detached Scan) lehetősége különösen jó példát szolgáltat a Nessus kiszolgálóalapú felépítésének értékes mivoltára.

Miután mindent beállítottunk és elkezdtük a pásztázást, a Nessus meghívja a megadott és/vagy szükséges modult vagy bővítményt, egy Nmap pásztázással bevezetve. Az egyes bővítmények futásának eredményétől függ, hogy kell-e további próbákat futtatni – a Nessus ebben meglehetősen okos. A pásztázás végeztével az eredményeket elküldi az ügyfélnek. Ha a munkafolyamat mentése (session-saving) be van állítva, az eredmények a kiszolgálón is tárolhatók.

A Nessus beszerzése és telepítése

A legtöbb nyílt forráskódú csomaghoz hasonlóan a Nessus is elérhető forráskódban és bináris formában egyaránt. A RedHat 7.0-hoz készített Nessus-csomag, melynek változatszáma 1.0.7a (a legfrissebb a cikk írásának idején), csak a ➲ http://redhat.aldil.org/rpm.html?id=73 címről tölthető le, *Matthias Saou*-nak köszönhetően (ezekbe a bináris okba nem fordították bele a kísérleti forráskódrészleteket).

Ha nem RedHat 7.0-n dolgozunk és az általunk használt terjesztésben nincs Nessus-csomag (a Debian 2.2-ben van), vagy a kísérleti lehetőségek érdekelnek, akkor a forrásból kell a Nessust lefordítani. Aggodalomra semmi ok: ha előbb telepítünk néhány szükséges dolgot és elolvassuk a Nessus telepítési leírását, a fordítás simán fog menni. A Nessus FAQ (➲ http://www.nessus.org/doc/faq.html) és a Nessus levelezőlista (http://list.nessus.org) bőséges ismeretanyagot szolgáltat a Nessus fordításáról és telepítéséről.

A Nessus a következő csomagok meglétét követeli: Nmap, a múlt hónapban ismertetett kapupásztázó, gtk, a Gimp eszközkészlet – beleértve a gtk+, gtk+-devel, glib+devel és XFree86-devel csomagokat, és az m4 parancsfájl-értelmező környezet, vagy a libgmp (ennek csomagját egyszerűen gmp-nek hívják).

Miután telepítettük ezeket, terjesztésünknek további szükségletei is lehetnek. Két ilyen esetről tudunk. Először, a gmp-2.0 szükséges a RedHat 7.0-hoz (amelyben általában a gmp-3.0 található meg és nem a 2.0; az RPM `--force` kapcsolóját kell használni, amennyiben a 3.0 már telepítve van a 2.0 telepítésekor). Ez a csomag letölthető a ➲ http://www.redhat.com/swr/i686/gmp-2.0.2-5.i686.html címről. Másodszor, a Nessus telepítéséhez vagy lefordításához SuSE Linux alatt szükség van a bison, flex, gtkdev és glibdev csomagokra. További részletekért olvassuk el a ➲ http://www.nessus.org/doc/faq.html oldalt.

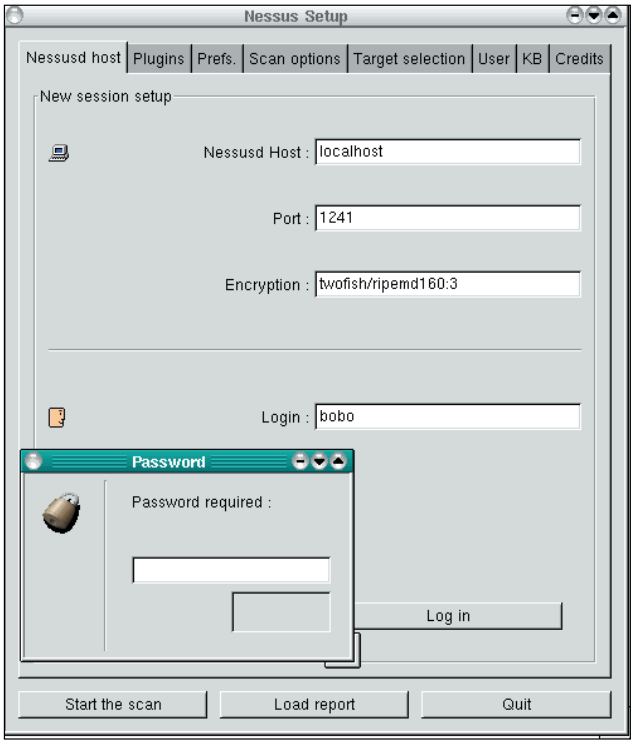
Ha már minden szükséges összetevő a helyére került, fordíthatjuk és telepíthetjük a Nessus-csomagokat. A fordítás egyszerű, mind a négy csomagnál a következő lépéseket kell végrehajtani: 1. kicsomagolás, 2. a csomag könyvtárában a `./configure` parancs kiadása, 3. `make`, majd 4. `make install`. A csomagokat a következő sorrendben fordítsuk le és telepítsük: nessus-libraries, libnasl, nessus-core és nessus-plugins.

Mielőtt lefuttatnánk a `configure` parancsfájlt a nessus-core-hoz, fontoljuk meg, hogy akarjuk-e használni munkafolyamat mentése illetve a tudásbázis lehetőségeket. A munkafolyamat mentése

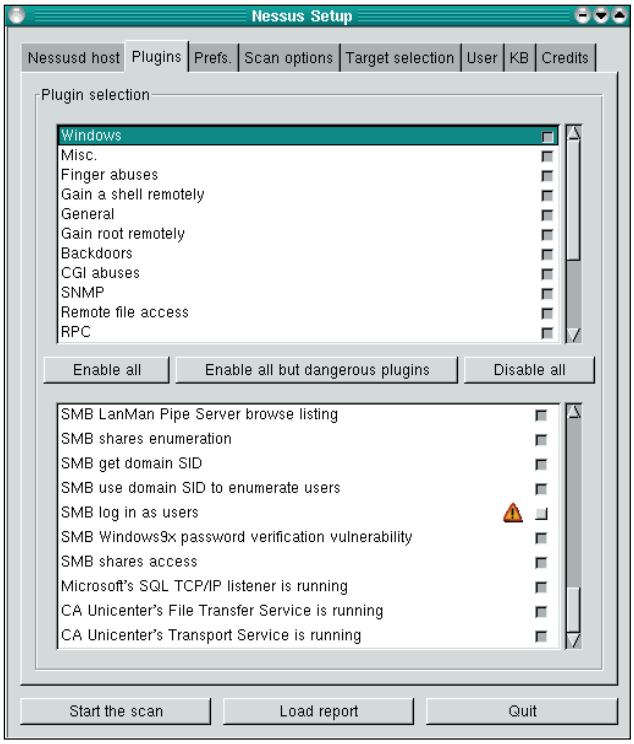
Nmap pásztázás TCP-kapcsolat, UDP- és RPC-modulokkal

```
Starting nmap V. 2.53 by fyodor@insecure.org
(www.insecure.org/nmap/)
Interesting ports on (10.193.133.93):
(The 3075 ports scanned but not shown below
are in
state: closed)
Port           State          Service (RPC)
80/tcp         open           http
111/udp        open           sunrpc (rpcbind
V2)
137/udp        open           netbios-ns
138/udp        open           netbios-dgm
139/tcp        open           netbios-ssn
1026/udp       open           (rpcbind V2)
1432/tcp       open           blueberry-lm

Nmap run completed-1 IP address (1 host up)
scanned
in 14 seconds
```



2. kép A „bobo” felhasználó első bejelentkezése a Nessus-kiszolgálóra



3. kép Bővítmények képernyője (A Windows-család látható)

lehetővé teszi a megszakított műveletek folytatását (például: a pásztázás folytatható az operációs rendszer vagy egy alkalmazás lefagyása után) és a leválasztott pásztázást (lásd fent). Ezt a lehetőséget a configure parancsfájl `--enable-save-session` kapcsolójával engedélyezhetjük.

A tudásbázis lehetővé teszi, hogy a pásztázások eredményeit a kiszolgálón tároljuk egy adatbázisban, amelynek felhasználásával később különbségi pásztázásra nyílik lehetőség. A tudásbázist engedélyező configure-kapcsoló a `--enable-save-kb`.

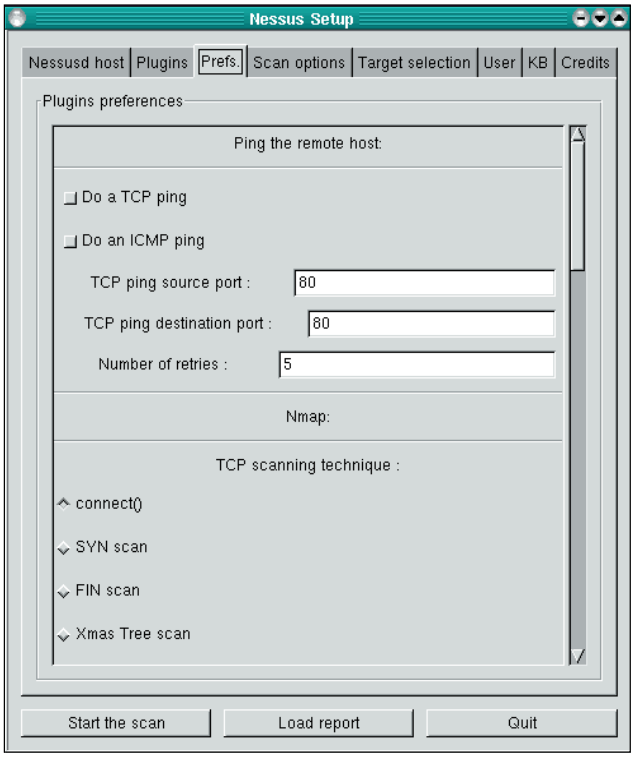
Ezek szerint ha a Nessus mindkét kísérleti lehetőségét ki akarjuk használni, a nessus-core fordítása előtt a következőképpen hívjuk meg a `configure`-t:

```
./configure --enable-save-sessions --enable-save-kb
```

A ➲ http://www.nessus.org/documentation.html címen részletesen olvashatunk ezeknek a lehetőségeknek a lefordításáról és használatáról. Mivel kísérleti kódról van szó, ebben a cikkben már nem ejtünk több szót róluk.

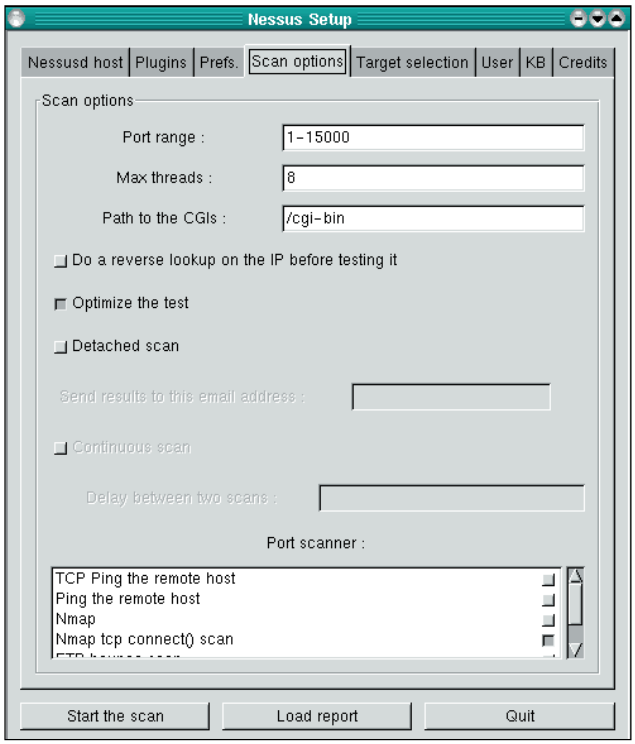
Miután mind a négy csomagot lefordítottuk és telepítettük, győződjünk meg róla, hogy a /etc/ld.so.conf fájl tartalmazza a /usr/local/lib bejegyzést. (Ha nem így lenne, adjuk hozzá kedvenc szövegszerkesztőnkkel.) Ezután futtassuk az `ldconfig` parancsot, hogy az ld (dinamikus csatoló) gyorstára frissüljön.

Végül, mivel a Nessus egyik erőssége, hogy fejlesztői rendszeresen új sebezhetőségi parancsfájlokat adnak ki, jó ötlet a munkát egy teljes sebezhetőségi adatbázissal kezdeni. A `nessus-update-plugins` parancsfájl segítségével az összes olyan bővítmény letöltődik a lynx program segítségével, amelyek a Nessus kiadása óta jelentek meg. Javasoljuk a `nessus-update-plugins -v` használatát, mert a `-v` kapcsoló nélkül a parancsfájl nem írja ki, hogy milyen bővítményeket telepít. Az új parancsfájlok letöltése, kicsomagolása és mentése után a `nessus-update-plugins` újraindítja a `nessusd`-t, hogy a rendszer az új bővítményeket használni tudja



4. kép Beállítások képernyő

(feltéve, hogy a nessus démon fut). Jelenleg ez a parancsfájl nem készít MD5 vagy másféle ellenőrzőösszeget, ezért a folyamat sokféleképpen befolyásolható. Ha ez zavar minket, a bővítményeket kézzel egyenként is letölthetjük a ➲ http://www.nessus.org/scripts.html weboldalról, de még ekkor sem lehetünk teljesen biztosak abban,



5. kép A Pásztázás beállításai képernyő

hogy minden rendben van – hacsak nem olvassuk el az összes parancsfájlt (a /usr/local/lib/nessus/plugins könyvtárban vannak), mielőtt a pásztázást megkezdenénk.

Nessus-ügyfelek

Hacsak nem egy gépen futtatjuk a Nessus-kiszolgálót és az ügyfelet, további telepítéseket kell elvégeznünk azon a gépen, amelyet ügyfélgépként kívánunk használni. A Nessus-kiszolgálóval (amelyen a nessusd fut) ellentétben, amely csak Unix-alapú gépen működhet, az ügyfelek Unixon és MS Windowson egyaránt futhatnak. A Nessus lefordítása és telepítése Unix-ügyfélgépen nem különbözik a kiszolgálótól (lásd fent).
A windowsos ügyfelek telepítése (WinNessus, NessusW és NessusWX) még ennél is egyszerűbb, mindhárom megtalálható bináris formában. A WinNessus hasonlít legjobban a Unix-ügyfél felhasználói felületére, aki ehhez van hozzászokva, annak ez a legjobb választás. Mindhárom Windows-ügyfél letölthető a ➲ http://www.nessus.org/win32.html címről.
Mielőtt a Nessus-ügyfelek használatáról beszélnénk, indítsuk el a démont.

A nessusd futtatása és karbantartása

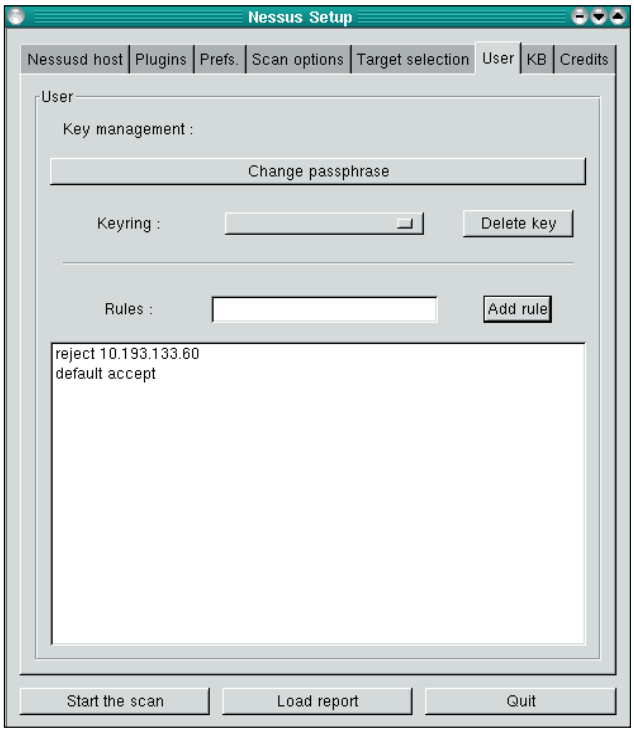
Rendben, térjünk vissza a Nessus-kiszolgáló konzoljához és készüljünk fel az első indításra. (Remélem, mindenki izgatottan várja ezt a percet és csakis jóban sántikál!) A nessusd sok más démontól különbözik abban, hogy elindítható démonként (azaz a háttérben) fut, valamint kapcsolóiban is, amelyek befolyásolják a viselkedését.
A démonmódban való indításhoz adjuk ki a `nessusd -D` parancsot. Ahogy az egy kiszolgálóalapú alkalmazástól elvárható, szükségünk lesz néhány Nessus-felhasználói fiókra a kiszolgálón. Ezek függetlenek a kiszolgáló helyi Unix-felhasználói fiókjaitól. A Nessus-fiókok két különböző módon hozhatók létre. Az első módszer a nessusd meghívása a `-P` kapcsolóval, amelyet közvetlenül a felhasználónév és az egyszer használatos jelszó követ. Ez nem zavarja a nessus démon működését, és nem is indít egy újat; valójában a Nessus felhasználói adatbázisát frissíti, és észrevétlenül újraindítja a démont. Például a „bobo” felhasználót „scuz00DL” jelszóval így adjuk hozzá:

```
nessusd -P bobo,scuz00DL
```

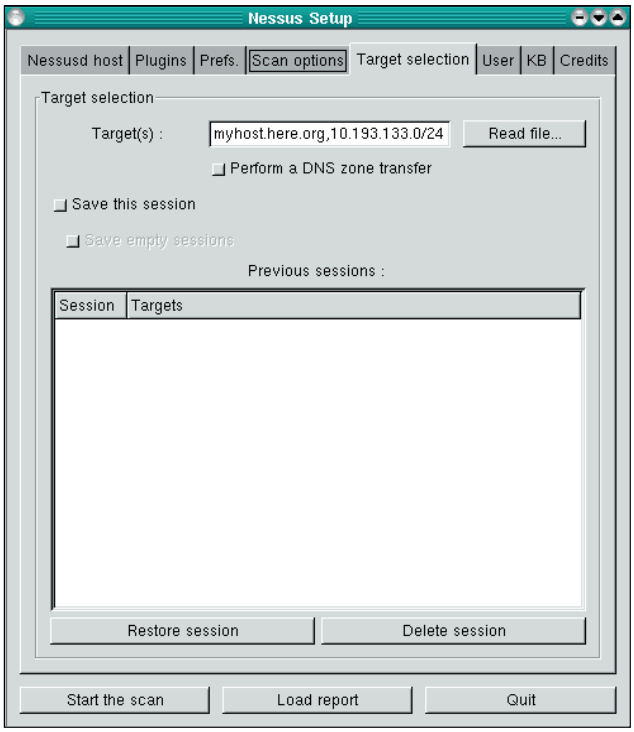
A jelszót egyszer használatosnak neveztük, mert alapesetben miután bobo először bejelentkezett és megadta a jelszavát, nyilvános kulcsa feljegyzésre kerül a Nessus-kiszolgálón, így a további bejelentkezések alkalmával nem kell újra beírnia a jelszót (a hitelesítés észrevétlenül történik egy SSL-szerű felszólítás illetve válasz művelettel).
A második és sokkal hatékonyabb módszer, hogy új felhasználói fiókokat hozzunk létre a kiszolgálón: a `nessus-adduser` parancs. Ez a parancsfájl gyakorlatilag minden varázslatát a nessusd hívogatásával végzi el, de kényelmes felületet ad a felhasználók kezelésére, ha a `nessusd -D` parancsnál nagyobb részletességgel szeretnénk őket szabályozni. Nemcsak a felhasználónevet és a jelszót kéri el, hanem azt az IP-címet is, ahonnan a felhasználó kapcsolatot kezdeményezhet, illetve azokat a szabályokat is, amelyek meghatározzák, hogy a felhasználó mely gépeket pásztázhatja a Nessusszal.
Ha ilyen mélységben érdeklődünk a felhasználói fiókok kezelése iránt, érdemes végigolvasnunk a nessus-adduser leírását. A cikk hátralevő részét most inkább arra szánjuk, hogy megbeszéljük a Nessus-pásztázás felépítését, futtatását és értelmezését.
Mielőtt azonban elhagynánk a hitelesítés témakörét, meg kell említeni egy másik hitelesítési fajtát is, amelyet a Nessus használ: az ügyfél helyben is végez hitelesítést minden egyes ügyfélkapcsolathoz. A Nessus-ügyfél első elindításakor egy jelszót kérdez. Ez a jelszó védi a titkos kulcsot a Unix-fiók `home` könyvtárában, amelyre a Nessus indításakor bejelentkezünk. Ezt mindig meg fogja kérdezni. Ezután ha a Nessus-kiszolgálóhoz csatlakozunk, a titkos kulcs segítségével jön létre az előbb említett észrevétlen felszólítás/válasz tranzakció, amely tulajdonképpen a távoli nessusd folyamat számára hitelesít minket. Ha ez most homályosnak tűnik is, semmi baj, a lényeg, hogy ne feledjük: ennek a jelszónak, amelyet mindig elkér az ügyfélprogram, semmi köze ahhoz a jelszóhoz, amelyet a Nessus-kiszolgálóhoz való első csatlakozásunk alkalmával használtunk.

Biztonsági pásztázások a Nessusszal

Most kezdődik az igazi móka! Miután a Nessust telepítettük és legalább egy felhasználót is beállítottunk, kezdhetünk pásztázni.
Először indítsuk el az ügyfelet és írjuk be az ügyfél titkos kulcsának jelszavát (mellesleg ezt bármikor megváltoztathatjuk a `nessus -C` paranccsal, amely bekéri a jelenlegi jelszót és az újat, amire módosítani szeretnénk).
Ezután írjuk be annak a nessusd kiszolgálót futtató gép nevét vagy IP-címét, amelyhez csatlakozni szeretnénk; a kaput, amelyen a kiszolgáló figyel; a használni kívánt titkosítási módszert és a Nessus felhasználónév-jelszó párost (*2. kép*). A kapu és titkosítás (*Port és Encryption*) alapértelmezett értékei általában megfelelők.
Ha elkészültünk, kattintsunk a *Log in* (Bejelentkezés) gombra.
Ha ez az első alkalom, hogy a kiszolgálóhoz kapcsolódunk, akkor meg kell még adnunk az egyszer használatos jelszót is (legközelebb már nem kéri a program). Ezután létrejön a kapcsolat és elkezdhetjük a pásztázást.
Ha a *Plugins* (Bővítmények) fülre kattintunk, megjelenik az összes a Nessus-kiszolgálón elérhető sebezhetőségi teszt listája, családok szerint csoportosítva (*3. kép*). Kattintsunk a család nevére (ezek az ablak felső felében jelennek meg), hogy megnézhessük, milyen bővítmények érhetők el az adott családhoz. A család melletti jelölőnégyzet használatával választhatjuk ki az összes bővítményt.
Ha nem tudjuk, hogy egy adott bővítmény mit csinál, kattintsunk



7. kép Felhasználó képernyő



6. kép Célpont kiválasztása képernyő

a nevére, ilyenkor egy tájékoztató ablak bukkan fel. Ha az egérmutatót a bővítmény neve fölé húzzuk, egy buborék jelenik meg a bővítmény rövid ismertetésével. Az olyan bővítmények, amelyek jelölőnégyzete mellett sárga háromszög található, különösen veszélyesek: az általuk végrehajtott tesztek megszakíthatják vagy akár le is fagyaszthatják a célponton (áldozaton) futó szolgáltatásokat. Csakis nagy körültekintéssel használjuk ezeket!

Gond nélkül kiválaszthatunk nagyszámú bővítményt, akár az összeset is. A Nessus elég okos ahhoz, hogy például a Windows-próbákat átugorja a nem Windowst futtató gépeken. Általában a Nessus hatékonyan eldönti, hogy milyen próbát kell futtatnia az adott körülmények között.

A következő képernyő a *Prefs* (Beállítások – *4. kép*). Ellentétben azzal, amit ennek láttán elsőre gondolnánk, ez a képernyő mégsem az általános, hanem a bővítményekkel kapcsolatos beállításokat tartalmazza, némelyik kötelező az adott bővítmény helyes működéséhez. Menjünk végig a listán, és adjunk meg annyi adatot, amennyit csak tudunk.

Figyeljünk a *Ping* részre (a legtetején), elég sokszor előfordul, hogy ha bármelyik pingmódszert (TCP vagy ICMP) választjuk, a Nessus arra a hibás következtetésre jut, hogy a célpont nincs bekapcsolva. A Nessus nem hajt végre semmilyen próbát olyan gépen, amely nem válaszol a pingre, ezért ha nem vagyunk biztosak benne, ne pingeljünk! Figyelem: az Nmap részben a Linux-felhasználók csak a `tcp connect()` lehetőséget válasszák, az összes többit a libpcap hibája miatt ne, ugyanis az befolyásolja a Nessus kapupásztázási tevékenységét.

A Prefs után jön a Scan Options (Pásztázás beállításai – *5. kép*). Vegyük észre, hogy az 5. képen szereplő Nessus-program a munkafolyamat mentése lehetőséggel lett fordítva, ennek bizonyítéka a *Detached Scan* és a *Continuous Scan*, amelyek egyébként nem jelentek volna meg. Akárcsak a Prefs képernyőn, itt is csak az Nmap `tcp connect()` lehetőséget válasszuk ki a *Port scanner* beállításnál, a fent említett hiba miatt.

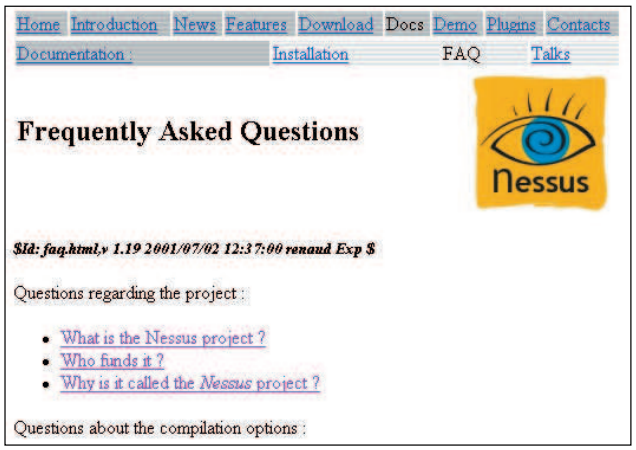
Az *Optimize the test* lehetőség választásával elkerülhetjük a látszólag felesleges próbákat, de ez hamis negatív eredmény keletkezéséhez vezethet, legalábbis elméletileg. Gondoljuk meg, mennyire aggaszt ez minket ahhoz viszonyítva, amilyen gyorsan megkapjuk az eredményt. Ha már a sebességről beszéltünk: amennyiben gyorsan szeretnénk eredményt kapni, kerüljük a *Do a reverse (DNS) lookup…* használatát, ez ugyanis minden pásztázott IP-címhez megpróbálja megtalálni a gépnevet.

Itt az ideje megnevezni „áldozatainkat”, vagyis a célpontokat. Ezeket a *Target(s):* mezőben adhatjuk meg a *Target Selection* (célpont kiválasztása) képernyőn (*6. kép*). Ide írhatjuk be vesszővel elválasztva a gépneveket, IP-címeket és hálózati címeket x.x.x.x/y alakban (ahol az x.x.x.x a hálózat száma és y az alhálózati maszk bitjeinek a száma, pl. 192.168.1.0/24).

A *Perform a DNS zone transfer* bejelölésével a Nessust arra utasítjuk, hogy minden elérhető DNS-információt megpróbáljon megszerezni a Target(s): mezőben megadott valamennyi tartománynévről és altartománynévről. Megjegyzendő, hogy a legtöbb internetes DNS-kiszolgáló úgy van beállítva, hogy megtagadja az ismeretlen gépektől érkező zónaátviteli kéréseket. A többi beállítás ezen a képernyőn a fentebb említett kísérleti munkafolyamat mentésének lehetőségével van összefüggésben. További tájékoztatásért olvassuk el a ➲ http://www.nessus.org/documentation.html weboldalt a kísérleti lehetőségek használatáról.

Végezetül még egy képernyőn kell túljutnunk a pásztázás megkezdése előtt, ez a *User* (Felhasználó) képernyő (*7. kép*). (Átugrottuk a KB képernyőt, amely csak akkor van jelen, ha a tudásbázis-lehetőséget is használni akarjuk, és ezért azt is lefordítottuk.) Ezen a képernyőn adhatjuk meg az ügyfél jelszavát (ugyanaz a hatása, mint a `nessus -C` parancsnak), és a Target Selection képernyőn beírt célpontok közül jelölhetünk meg kivételeket (valójában a célpontlista finomhangolása ez).

Ezeket a kivételeket szabályoknak hívják és egyszerű alakban felírhatók: cím elfogadása, cím elvetése, alapértelmezett elfogadás vagy visszautasítás. A 7. képen a felsorolt szabályok azt jelentik, hogy ne pásztázza a 10.192.133.60-at, de pásztázzon minden mást



➲ http://www.nessus.org/doc/faq.html

az előző képernyőn megadottak közül.
Lássuk az eredményt! Kattintsunk a *Start the Scan* (Pásztázás megkezdése) gombra a képernyő alján. A pásztázás időtartama eltérő hosszúságú lehet, főként a kiválasztott gépek és a végrehajtandó próbák számától függ. A végeredmény hasonló az 1. képen láthatóhoz. A *Report* (Jelentés) ablakból menthetjük is a jelentést egy fájlba azon kívül, hogy nézegethetjük és különböző részleteit megjeleníthetjük. A támogatott formátumok között szerepel a HTML, az ASCII, a LaTeX és természetesen a Nessus saját jelentésfájl-formátuma, az NSR. Ha valaha is jelentést szeretnénk visszatölteni a Nessusba, az NSR formátumot kell használnunk.

Olvassuk el alaposan a jelentést, minden + jelre kattintsunk rá, és javítsuk a Nessus által jelzett hibákat. A Nessus megtalálja a hibát, sokszor megoldást is javasol, de nem javítja ki helyettünk. Az is igaz, hogy a Nessus nem feltétlenül talál meg minden biztonsági rést a rendszerünkön.
Ez a helyzet a biztonsági pásztázókkal: ennyit képesek megtenni, ráadásul nem minden bővítmény hatékony egyforma mértékben a hibák megtalálásában. Még ha hatékonyak is lennének, a Nessus érthető okokból nem találhatja meg azokat a biztonsági réseket, amelyekre nincs bővítménye, ezért bővítményeinket rendszeresen frissítsük.

Néhány záró gondolat

A Nessus sokoldalú, rugalmas, kereskedelmi szintű, de teljesen ingyenes és szabad biztonsági pásztázó. A helyesen létrehozott és értelmezett Nessus-jelentések segíthetnek minket abban, hogy a jól ismert hibákat elkerüljük. Nem beszéltünk a saját bővítmények írásáról, amelyek lehetővé teszik, hogy ne csak a közismert biztonsági réseket ellenőrizhessük, hanem új, ez idáig ismeretlen hibákat fedezzünk fel. Még egyszer kérjük, hogy mindenki felelősen használja ezt az eszközt!
Remélve, hogy így lesz, jó szórakozást kívánunk!

Mick Bauer (mick@visi.com) hálózati biztonsággal foglalkozó szaktanácsadó. 1995 óta a Linux elkötelezett híve, 1997 óta pedig OpenBSD prófétaként tevékenykedik. Mick szívesen fogad minden kérdést, és megjegyzést.