A DNS és a BIND biztonsága

Ha a DNS-kiszolgáló biztonságos, a hálózat is biztonságos és naprakész.

A SANS Institute nemrég kiadott, a tíz legfontosabb internetes biztonsági veszélyről szóló írásában első helyen foglalkozik a BIND sebezhetősége miatt előforduló hibákkal. A BIND nevű nyílt forrású programcsomag képezi a Világháló legtöbb DNS-kiszolgálójának lelkét. A SANS szerint a telepített BIND csomagok legalább fele a legismertebb támadási módszereknek sem képes ellenállni. A jó hír azonban az, hogy az itt ismertetésre kerülő, könnyen megérthető szabályok betartásával nagymértékben növelhetjük a BIND megbízhatóságát a linuxos (vagy más unixos) DNS-kiszolgálónkon. Mivel itt és most főleg a biztonsággal foglalkozunk, ajánljuk, hogy a BIND kezelésében kevésbé jártasak előbb olvassák el a csomag Interneten elérhető leírását, illetve az első két fejezetet az *Albitz–Liu* szerzőpáros *DNS and BIND* című híres könyvéből.

A BIND alapjai

Kezdjük a Domain Name Service (DNS) és a BIND működésének ismertetésével. Tegyük fel, hogy gépünk címe myhost.someisp.com (*1. ábra*) és a ➲http://www.wiremonkeys.org/ címen található honlapot szeretnénk megtekinteni. A DNS-kiszolgáló neve legyen ns.someisp.com. Mivel a www.wiremonkeys.org semmit nem mond az adattovábbításért felelős átjárók (gateway) számára, a felhasználó böngészőjének előbb meg kell tudnia a névhez tartozó IP-címet. Először a myhost az ns géphez fordul, hátha az tud valamit a címről. Mivel az ns.someisp.com nem felelős a www.wiremonkeys.org nyilvántartásáért, s az utóbbi időben nem érkezett hozzá a cím kiderítésére irányuló kérelem, ezért a felhasználó nevében kezdi meg a keresést. Az egyik kérelem kiszolgálása céljából intézett másik kérelmet rekurziónak hívjuk.

Az ns.someisp.com egy főkiszolgálótól (root server) tudja meg a www.wiremonkeys.org címet nyilvántartó DNS-kiszolgáló nevét. (Minden DNS-kiszolgálón találunk egy szövegfájlt, amelyben az Internet tizenhárom főkiszolgálójának neve és címe található. Ez a lista az ➲ftp://ftp.rs.internic.net/domain címen érhető el, neve „named.root”). Példánkban az ns az E.ROOT-SERVERS.NET főkiszolgálót kérdezi meg (címe 192.203.230.10), mely azt válaszolja, hogy a www.wirwmonkeys.org címet az ns-wiremonkeys.wiremonkeys.org kiszolgáló tartja nyilván, s ennek címe 55.100.55.100.

Az ns ezután az ns-wiremonkeys géptől érdeklődik a www.wiremonkeys.org IP-címe iránt. Az ns-wiremonkeys.org az 55.100.55.200 választ küldi vissza, s ezt az adatot az ns a myhost.someisp.com címre, azaz saját gépünkre továbbítja. Végül a myhost kapcsolatba lép az 55.100.55.200 címmel HTTP-n keresztül, és így már elkezdődhet az adatcsere.

Ez a névfeloldás legjellemzőbb példája, amit egyszerűen „lekérdezésnek” nevezünk. A lekérdezések számára az 53-as UDP kapu van fenntartva.

Azonban nem minden DNS műveletben egy gazdát kérdezünk le. Néha szükséges, hogy egész névtartomány- (vagy zóna-) adatbázisokat továbbítsunk. Ezt tartománytovábbításnak (zone transfer) hívjuk. Amikor a dig parancsot futtatjuk, vagy az nslookup program ls parancsát használjuk, akkor is ez a folyamat zajlik le. A tartománytovábbítás fő célja az, hogy az egyazon tartományért felelős névkiszolgálók összehangoltan működjenek. A tartománytovábbítás szintén az 53-as UDP kapun keresztül történik.

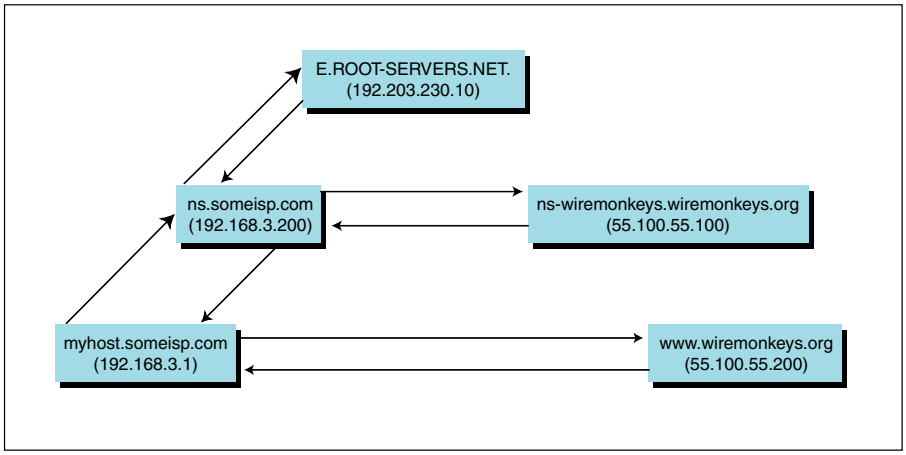
Az utolsó általános DNS tulajdonság a gyorstár (cache) alkalmazása. A névkiszolgálók a helyi tartomány minden fájlját gyorstárba helyezik, s ugyanígy a legutóbbi újraindítás óta hozzájuk intézett lekérdezési kérelmeket is. Tulajdonképpen ennyi lenne az egész: minden forrásrekordnak (resource record, RRs) élettartam-beállítása is van, ezek az adatok értékek azt határozzák meg, hogy az adott RR meddig tartózkodhat a gyorstárban, azaz mikor kell legközelebb frissíteni. Mindez természetesen csupán a töredéke annak, amit érdemes tudni a BIND-ről, hiszen a továbbítókat (forwarder) és a fordított lekérdezéseket nem is említettem. Remélhetőleg, ennyi is elég lesz a BIND biztonságával foglalkozó részek megértéséhez.

A DNS biztonsági elvei

A biztonságos DNS két alapvető szabálya a következő: mindig a választott DNS-programcsomag legfrissebb változatát használjuk, illetve soha ne engedélyezzük feleslegesen sok adat vagy szolgáltatás elérését idegenek számára. Röviden: legyünk naprakészek, gyanakvóak és zsugoriak!

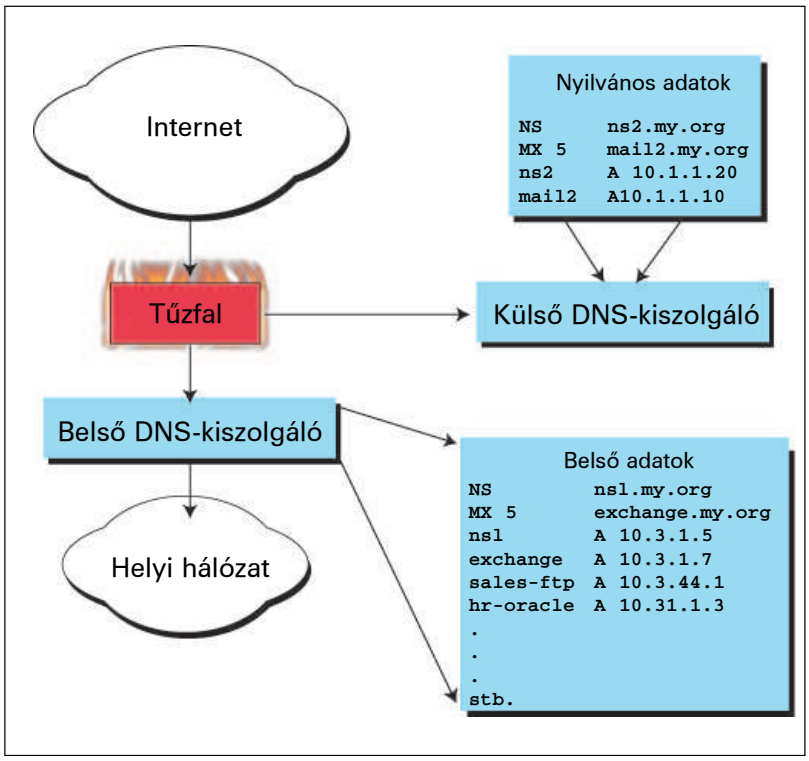
A fő elvekből több eljárás következik, amelyeket alkalmaznunk kell. Az első, hogy korlátoznunk, esetleg tiltanunk kell a rekurziót. A korlátozás egyszerűen elvégezhető a beállításfájl értékeinek megváltoztatásával; megtiltani viszont csak abban az esetben tudjuk, ha ez nem akadályozza a kiszolgáló munkáját.

Ha egy gép külső DNS-kiszolgálóként működik, melynek egyetlen feladata a saját hálózatához tartozó gépek felé irányuló kérelmek teljesítése, akkor nyugodtan megtilthatjuk a rekurziót, hiszen a gépnek így nem kell külső gazdaneveket keresnie. Másrészről, ha egy kiszolgáló DNS-névfeloldást ad a helyi hálózat gépei számára, akkor a helyi hálózatból érkező rekurziós kérelmeket mindenképpen teljesítenie kell, a kívülről érkezőket azonban nyugodtan figyelmen kívül hagyhatja.



1. ábra A myhost.someisp.com lekérdezése

A DNS korlátozásának másik módja, hogy a szolgáltatásokat szétválasztjuk egymástól (*2. ábra*).
A „szétválasztott DNS” kifejezés azt jelenti, hogy minden egyes helyi résztartományhoz nyilvános és titkos adatbázist egyaránt fenntartunk. A nyilvános adatbázisban csak a legszükségesebb adatok találhatók meg: a nyilvános névkiszolgálók nevét tartalmazó NS bejegyzések; a külső SMTP (e-mail) átjárókat meghatározó MX bejegyzések; a nyilvános webkiszolgálók és más gépek nevei, amelyek létezéséről a külvilág is tudomást szerezhet.
A titkos adatbázis a nyilvános adatbázis bővített változata is lehet, de akár teljesen különbözhet is attól. Például sok helyen a Microsoft Exchange kiszolgálót használják belső levelezésre, de a külső levelezést egy másik SMTP átjárórendszerrel bonyolítják le, ez általában a hálózat tűzfala vagy egy, a tűzfallal kapcsolatban álló, de a belső hálózatról leválasztott szabad sávban (Demilitarized Zone) lévő levelező-kiszolgáló. Az ilyen felépítés előnye egyértelmű: az SMTP átjáró sebezhetősége nem jelenti azt, hogy egy behatoló az egész belső levelezést elérheti. Más, ehhez hasonló módon szétválasztott szolgáltatások között az alábbiakat találjuk: a WWW (a kívülről elérhető adatokat, illetve a belső hálózat adatait választják el), az FTP és gyakorlatilag minden olyan TCP/IP-alapú szolgáltatás, amelynél az adatokat „ez nyilvános”, „ez meg nem” elvek alapján lehet csoportosítani. A DNS szétválasztása talán a legkényesebb téma, hiszen a legtöbb TCP/IP-szolgáltatás alapját a DNS képezi.



2. ábra Szétválasztott DNS

A fentebb említett „gyanakvás” másik területe a tartományfájlok tartalma. Még a nyilvános adatbázis is feleslegesen sok adatot tartalmazhat. A gazdanevek túlságosan beszédesek lehetnek (ez megkönnyíti a behatolók dolgát), illetve túl sok vagy fontos adatot szolgáltathatnak. Néhány hálózat még az egyes rendszerek egyedi tulajdonságait is közzéteszi! Az ilyen jellegű adat a legtöbb esetben csupán a behatolók számára jelent valamit. A programok állandó frissítése és az ismert DNS-hibák kiszűrése legalább olyan fontos, mint annak eldöntése, hogy a DNS-kiszolgáló milyen kérelmeket teljesítsen. Ráadásul még egyszerűbb is, hiszen a BIND legújabb változatát ingyen letölthetjük az ➲ ftp.isc.org címről, és a BIND hiányosságairól szóló beszámolókat, vitákat több levelezési listán és hírcsoportban is figyelemmel kísérhetjük.
A DNS biztonságának harmadik alapvető elve nem csak a DNS-re vonatkozik: szánjunk időt a programcsomag biztonsági szolgáltatásainak átböngészésére és megfelelő beállítására! Szolgáltatónkra is figyeljünk oda: a DNS általános vezérléséért felelős Network Solutions és más hasonló cégektől gyakran levélben is kérhetünk értesítést a legújabb biztonsági frissítésekről.

A megfelelő BIND csomag kiválasztása és telepítése

E cikk megírása idején a legfrissebb változat a 8.2.2-es volt, az 5. hibajavítással (patch). A régebbi változatok egyik különösen alattomos hibája, hogy a behatoló a túlcsordulás kihasználásával rendszergazdai jogokat szerezhet (erre a CERT #CA-99-14 számú jelentése is kitér), ezért különösen fontos, hogy a 8.2.2p5 változatot használjuk. Megjegyzendő, hogy a BIND v.8.1 1997 májusi megjelenése után, megbízhatóságának köszönhetően sokan továbbra is a v.4-et használták (s így az új beállításokat sem kellett megtanulniuk...). Az Internet Software Consortium (ISC) továbbra is támogatja ezt a változatot; a v.4.9.7 a 8.1 kiadása után egy évvel jelent meg.
Az ISC sem javasolja azonban, hogy bárki ezt használja. Még egyszer ismétlem: ha BIND, akkor legalább 8.2.2p5.

A következő kérdés, hogy előfordított bináris csomagot (például RPM-et) használjunk-e, vagy magunk fordítsuk le a forráskódot? A legtöbb felhasználó számára tökéletesen megfelel a lefordított csomag, feltéve, hogy megbízható helyről szerezte be. Gyakorlatilag a BIND minden Unix-változat része, de ellenőrizzük, hogy nekünk a legújabb van-e meg.
A Red Hat Package Managerben az `rpm -q -v bind8` parancsot kell használnunk, ha a csomagot már telepítettük, vagy az `rpm -q -v -p ./<a_fájl_neve>` parancsot, ha van, de még nem telepítettük. Az rpm-es BIND csomagneve általában „bind8” vagy „bind”.
Ha kiderül, hogy régi változatunk van, akkor sincs nagy baj: a legtöbb csomagformátum frissítési szolgáltatást is tartalmaz. A legújabb változat letöltése után a csomagkezelővel frissíthetjük a régit. Az RPM esetében az `rpm -U ./a_csomag_neve>` parancsot kell kiadnunk, ehhez természetesen még további kapcsolókat is fűzhetünk. Ha ez nem működne, akkor próbáljuk meg az `rpm -U --force ./< a_csomag_neve>` alakot.
Ha nem találunk megfelelő csomagot, akkor magunknak kell lefordítanunk a forrást. Ez sem nehéz, hiszen nincs beállítóprogram, és a BIND v.8x Makefile-jait sem kell módosítanunk. Csak kövessük a forrás INSTALL fájljában megadott útmutatást. A legtöbb esetben mindössze egy `make`, majd egy `make install` parancsra lesz szükségünk.

A named indítása: a lelakatolt cella

A BIND fő folyamatát, a namedet egyelőre még korai lenne elindítanunk. Arról azonban döntenünk kell, hogy mire akarjuk használni, hiszen ettől függ az összes beállítás. Itt az idő tehát, hogy szót ejtsünk néhány, a biztonságot növelő kezdeti beállításról.
Ahogy minden internetes szolgáltatást, a namedet is célszerű „lelakatolt cellában” futtatnunk, amelybe a behatoló szépen bezárja magát, ha a túlcsordulás kihasználásával támad kedve kísérletezni. Ezt a cellát három kapcsolóval hozhatjuk létre: `-u <felhasználónév>`, `-g <csoportnév>` és `-t <könyvtárnév>`.

Az első hatására a named az adott felhasználói néven, a második hatására pedig a megadott csoportnéven fut. A harmadik a named által hivatkozott könyvtárak elérési útvonalainak gyökerét változtatja meg (chroot). Figyeljünk oda arra, hogy ez utóbbi beállítás még a named.conf beolvasása előtt érvénybe lép, ezért a

```
named -u named -g wheel -t /var/named
```

parancs kiadásakor a named.conf fájlt a rendszer nem a /etc könyvtárban, hanem a /var/named/etc-ben keresi. Tehát a named.conf alapértelmezés szerinti helye mindig a /etc, de ha a könyvtárak gyökerét /más/útvonal-ra helyeztük, akkor a named.conf-nak a /más/útvonal/etc-ben kell lennie.
Hogy miként növeli ez a biztonságot? A három kapcsoló használatával a named jogait, környezetét, sőt, még az elérhető fájlrendszereket is komolyan korlátozhatjuk. Ha egy behatoló netán átvenné a vezérlést a BIND fölött, nem kapna rendszergazdai jogokat (a BIND v.8 előtt ezt megtehette, hiszen a BIND rendszergazdaként futott). Ehelyett egy közönséges felhasználó jogait kapja meg, ezenkívül (mivel a megadott gyökér fölött elhelyezkedő könyvtárak tartalma a named számára gyakorlatilag nem is létezik) szinte semmit sem láthat a fájlrendszerekből.

1. lista Példa a named.conf-ra egy külső DNS-kiszolgálón.

```
acl trustedslaves { 192.168.20.202; 192.168.10.30};

options {
     directory "/";
     listen-on { 168.192.100.254; };
     recursion no; fetch-glue no;
     allow-transfer { trustedslaves; };
};

logging {
     channel seclog {
          file "var/log/sec.log" versions 5 size 1m;
          print-time yes; print-category yes;
     };
     category xfer-out { seclog; };
     category panic { seclog; };
     category security { seclog; };
     category insist { seclog; };
     category response-checks { seclog; };
};

zone "coolfroods.ORG" {
     type master;
     file "master/coolfroods.hosts";
};

zone "0.0.127.in-addr.arpa" {
     type master;
     file "master/0.0.27.rev";
};

zone "100.168.192.in-addr.arpa" {
     type master;
     file "master/48.42.208.rev";
};
```

A named.conf

A fent ismertetett módszer már tökéletesen biztonságos, de ez még csak a kezdet! A BIND 8.x beállításfájlja (named.conf) rengeteg paraméterével a szolgáltatást egészen pontosan szabályozhatjuk. Vegyük az *1. listán* látható példát. Az ehhez tartozó gép egy külső DNS-kiszolgáló. Mivel feladata szerint a helyi tartományról (coolfroods.org) szolgáltat adatokat a külvilág számára, a rekurziót nem engedélyezi. Valójában nincs is `" . "` bejegyzése (ami egy listafájlra mutatna), tehát nem tud és nem is szerezhet adatot a helyi tartományon kívül található gépekről. A helyi tartomány adatbázisait csak a megbízható másodlagos kiszolgálók egy csoportja számára adja ki, s majd minden előforduló eseményről naplóbejegyzés készül.
Tehát mi kerül a named.conf-ba, amitől nyugodtabban alhatnak a legfélősebb rendszergazdák is?

Hasznos named.conf beállítások: acl{}

Az ACL (Access Control List) listák segítségével IP-címek egy csoportjához neveket rendelhetünk. Erre szükség is van, hiszen nyilvánvalóan tiltani szeretnénk a hozzáférést bizonyos IP-címekről. Egy ACL-t elvileg a named.conf bármely részén meghatározhatunk, de mivel a fájlt a rendszer felülről lefelé olvassa be, ezért első használata előtt minden ACL-t meg kell határoznunk. Ebből következik, hogy az ACL-meghatározásokat a named.conf tetejére kell tennünk. A beállítás formája egyszerű:

```
acl név { IPcím1; IPcím2; ...};
```

Figyeljük meg, hogy minden IP-címlista teljes (x.x.x.x alakú) és hálózati (x.x.x.24, x.x./16 stb. alakú) címeket is tartalmazhat. A fájl beolvasásakor az ACL neve helyére az általa meghatározott címlista kerül.

Általános beállítások: options{}

Nézzük át az általános beállításokat. Néhány ezek közül a tartományokra vonatkozó szakaszokban is használható. Figyeljünk arra, hogy, ha egy beállítás az options{} részben, és a tartományok szakaszában is szerepel, akkor az utóbbi változat felülbírálja az előbbit. Tehát a leíró részben megadott beállítások az általános beállítások kivételeit határozzák meg.
A *2. listán* az options{} részben használható néhány hasznos értéket találunk.

Naplózás

Az általános beállítások mellett a naplózási szabályokat is meg kell határoznunk. Alapértelmezés szerint a named csupán néhány indítási üzenetet (a hibákat és a betöltött tartományokat) naplóz a syslogd démon segítségével, az pedig a /var/log/messages-be vagy máshová írja azokat. A biztonsággal kapcsolatos események naplózásához a `logging{}` szakaszt is be kell illesztenünk a named.conf fájlba. A `logging{}` két részből áll: egy vagy több `channel{}` (ezek a csatornák határozzák meg, hogy hová kerüljenek a naplóadatok), illetve egy vagy több `category{}` (itt a nyomon követett eseményekhez rendelhetünk csatornákat) szakaszt tartalmaz. A csatornák általában fájlokra vagy a helyi syslogd démonra mutatnak. A tárgy-

körök legtöbbször előre meghatározottak, tehát ezek közül választhatunk, és megadhatjuk, hogy az adott tárgykörre vonatkozó naplóadatok hová kerüljenek.
A csatornákat az alábbiak szerint határozhatjuk meg:

```
channel csatornanév {
       [ fájlnév | syslog syslogtípus | null ];
       print_time [ yes | no ];
       print_category [ yes | no ];
       };
```

Ne felejtsük el, hogy a fájlnév alapértelmezés szerint a named munkakönyvtárába kerül, de teljes útvonalat is megadhatunk, mely (ha lehetséges) a módosított gyökérhez képest kerül értelmezésre.
A tárgykörök meghatározása jóval egyszerűbb:

```
category kategória_neve {
       csatornalista ; };
```

Az IP-címlistákhoz hasonlóan a csatornalista elemeit is pontosvesszővel választjuk el egymástól, s az csak egy fentebb szereplő `channel{}` beállításban meghatározott csatornaneveket tartalmazhat. A BIND útmutatójában (BOG) a támogatott tárgykörök teljes listáját megtaláljuk, itt és most elég annyit megemlítenünk, hogy az `xfer-out, security, load, os, insist, panic` és `maintenance` minden biztonságmániás rendszergazdát érdekelhetnek.

Csak gyorstárazó névkiszolgálók

A csak gyorstárazó (caching-only) névkiszolgálók nem felelnek a tartományok nyilvántartásáért, így biztosításuk is jóval egyszerűbb, s a velük végzett munka során az alábbiak közül csupán néhány dologra lesz szükségünk.
A named.conf szakaszai közül utolsóként a `zone{}` kerül terítékre. Az `options{}` részhez hasonlóan a lentebb ismertetetteken kívül rengeteg további beállítás használható. A BOG-ból mindent megtudhatunk róluk.
A tartományonkénti biztonsági beállítások közül az alábbi három a legfontosabb:

```
allow-update { IP_vagy_ACL_lista ; };
allow-query { IP_vagy_ACL_lista ; };
allow-transfer { IP_vagy_ACL_lista ; };
```

Az allow-update beállításnál a tartományra vonatkozó dinamikus DNS-frissítéseket küldő gazdákat sorolhatjuk fel. Az allow-query határozza meg, hogy mely gépektől fogadhatjuk el a DNS-kérelmeket. Az allow-transfernél állíthatjuk be, hogy kik tölthetnek le egész fájlokat. Jegyezzük meg, hogy mindhárom paramétert használhatjuk bármelyik, vagy akár mindkét `zone{}` szakaszban, illetve az `options{}` szakaszban.

A fájlok biztonsága

DNS-szolgáltatásunk most már jó úton halad a tökéletes biztonság felé. De mi legyen az adatbázisokkal?
A jó hír az, hogy mivel jóval kevesebb lehetőségünk van, mint a named.conf esetében, ezért kevesebb feladatunk lesz. A rossz hír viszont az, hogy legalább egy RR idejétmúlt és használata veszélyes, így legjobb messze elkerülni. A *3. listán* a boneheads.com fájlját láthatjuk.
Az első érdekes elem a Start-of-Authority (SOA) bejegyzés. A fenti példában a sorszám az éééehhnn## formában van megadva. Ez így kényelmesen használható és a biztonságot is növeli, továbbá csökkenti annak az esélyét, hogy véletlenül régi adatfájlt töltünk be. A frissítési időközt három órára állították, ami megfelelő középút a sávszélességgel való takarékoskodás és az üldözési mánia között. Minél gyakrabban frissítünk, annál kevesebb kárt okozhat egy esetleges „cache-poisoning” (gyorstárfertőzés) típusú támadás, hiszen a behatoló által szaporított rossz bejegyzések az adatok minden frissítésekor javításra kerülnek.
A lejárati idő két hét, ez az időtartam az, amíg a fájl érvényesnek tekinthető. Egy biztonságmániás rendszergazda kétféleképpen tekinthet erre az értékre. Egyrészt a hosszú időtartam megengedi, hogy ha az elsődleges kiszolgálót egy bizonyos időszakban sorozatosan érnék szolgáltatásmegtagadás (denial-of-service) típusú DOS támadások, a másodlagos kiszolgálók a gyorstárban elhelyezett adatokkal

2. lista {} Options {}

```
listen-on [port #] { a helyi IP-k listája ; };
# Meghatározza, hogy a DNS-lekérdezésekre és a
# kérelmekre mely csatolókon keresztül válaszoljunk.
# Megadása nem kötelező. Ennek és minden más
# címlistának az elemeit pontosvesszővel kell
# elválasztanunk egymástól.

# allow-recursion { a rekurzió, mely
                    IP-címek felé legyen
                    engedélyezett ; };
# Rekurzív lekérdezéseket végez a meghatározott
# IP-listán,
# Ez a "none;" szót is tartalmazhatja.

allow-transfer { azon IP-k, amelyek számára
                 engedélyezzük az adatok fogadá-
                 sát vagy "none" ; };

allow-update { IP- vagy ACL-lista ; };
# Ezen IP-k, ACL-ek és hálózatok felől
# engedélyezett a dinamikus DNS frissítés
# (vagy "none")

allow-query IP- vagy ACL-lista ; };
# Ezen helyek számára engedélyezett az
# egyszeres DNS-lekérdezés (vagy "none")

version " [a változatszám mellett megjelenő üzenet]";
# Ezt senkinek nem kell kiszolgáltatnunk.
# A legtöbben ide humoros üzenetet írnak.

recursion [yes | no];
# Az általános rekurziót kapcsolhatjuk ki-be.
# Ha kikapcsoljuk, a fetch-glue paramétert is
# "no"-ra kell állítanunk (lásd lejjebb).

fetch-glue [yes | no];
# A "glue" bejegyzések olyan RR-ek, amelyek más RR-ek
# értelmezéséhez szükségesek (például minden olyan
# névnek, melyre egy "CNAME" bejegyzés hivatkozik,
# lennie kell valahol egy "A" bejegyzésnek is.
# Alapesetben a glue bejegyzéseket a hagyományos
# lekérdezések során is átvihetjük, hacsak itt ki
# nem kapcsoljuk e lehetőséget.
```

továbbra is elérhetővé teszik a tartományt, persze a fő DNS-kiszolgáló kivételével. Azonban az adatok a támadás idején is változhatnak, s a régi adatok néha több bajt okoznak, mint ha nem is léteznének.
Tehát az élettartamot elég rövidre kell állítanunk ahhoz, hogy az esetleges támadást követő helyreállítás gyorsan megtörténjen, viszont elég hosszúra ahhoz, hogy a sávszélességet ne pazaroljuk feleslegesen. (A TTL határozza meg, hogy az egyes tartományrészek RR-jei meddig maradhatnak más névkiszolgálók gyorstárában, lekérdezésük után.)
Másrészt a biztonság fontos tényezője még az is, hogy minél kevesebb felesleges adatot szolgáltassunk ki. A lehető legkevesebb nevet („A record”) és másodnevet („CNAME record”) tartsunk nyilván, csak azok a gépek legyenek jelen, amelyekre feltétlenül szükség van. Valójában a DNS-t szeretnénk szétválasztani, de ha ez valamilyen okból nem kivitelezhető, akkor minél „szűkszavúbb” adatfájlokat kell készítenünk.
Ez akkor következik be, ha egy kérdezett bejegyzés olyan nevet tartalmaz, melynek IP címe („A” típusú bejegyzésben) nincs jelen a kiszolgálón. Más szóval, ha az X kiszolgáló tudja azt, hogy Y felelős a WUZZA.com tartományért, de nem ismeri Y IP-címét, akkor kezd csak igazán bonyolódni a dolog a rendszer a lehető legjobb úton halad afelé, hogy valaki támadással törjön be. Ezért, ha minden rekurziót ki szeretnénk szűrni, győződjünk meg arról, hogy egyik RR sem igényel többszintű feloldást (glue-fetching), s ezután állítsuk a „fetch-glue” értéket „no”-ra.
Az RP és TXT típusokat ne használjuk, vagy csak körültekintően. A HINFO típusok azonban soha (!) nem tartalmazhatnak fontos adatokat! Az RP (Responsible Person) alapértelmezés szerint a kiszolgálóért felelős személy levélcímét adja meg. Ide legjobb valami teljesen semmitmondó címet írni, például: information@wuzza.com, vagy hostmaster@wuzza.com. A TXT bejegyzés további szöveges adatot tartalmazhat a kapcsolattartó személyről (telefonszám stb.), de lehetőleg tényleg csak ezt írjuk ide. A legjobb, ha az egészet figyelmen kívül hagyjuk.
A HINFO a régi szép idők hagyatéka: egyesek itt annak idején a használt operációs rendszert, annak változatát, sőt, még a kiszolgálóhoz tartozó számítógépek tulajdonságait is megadták! Akkoriban az Internethez kapcsolódó hálózatok legnagyobb része egyetemi rendszer volt, a számítógépekre még csodálkozva tekintettek (kevesebb bajkeverő ügyködött...), és semmi sem indokolta, hogy ezt az adatot eltitkolják. A HINFO-nak manapság már nincs valódi haszna, hacsak az nem, hogy hamis adatok megadásával félrevezethetjük a betörőket.
A 3. listára visszatérve: láthatjuk, hogy a három utolsó bejegyzés teljesen felesleges, a behatolók számára viszont valóságos aranybányát jelent. S bár úgy fogalmaztunk, hogy a SOA bejegyzésekkel minden rendben van, az utána következő NS bejegyzéssel együtt egy másik tartomány gazdájára mutat. Mivel az ilyesmit nem szeretjük, ezért az ns.otherdomain.com-hoz egy „A” bejegyzést is be kell illesztenünk.

3. lista Példa egy adatfájlra.

```
@ IN SOA cootie.boneheads.com. hostmaster.boneheads.com. (
              200000215       ; sorszám
              10800             ; frissítés (3 óra)
              1800              ; újrapróbálkozás (30 perc)
              1209600           ; lejárat (2 hét)
              432000 )          ; RR TTL (12 óra)
          IN     NS       ns.otherdomain.com.
          IN     NS       cootie.boneheads.com.
          IN     MX 5     cootie.boneheads.com.
blorp     IN     A        10.13.13.4
cootie    IN     A        10.13.13.252
cootie    IN     HINFO    MS Windows NT 3.51, SP1
@         IN     RP       john.smith.boneheads.com.
dumb.boneheads.com.
dumb      IN     TXT      "John Smith, 612/231-0000"
```

A BIND további biztonsági kérdései: a TSIG

A dolog legnehezebb részén túl vagyunk, de még nem is érintettük a titkosítás vezérlését. A „Secure DNS” (biztonságos DNS) protokoll (az RFC 2535-ben leírt DNSSEC) ismertetésének akár külön cikket is szánhatnánk. A DNS e bővítésének segítségével titkosíthatjuk a tartományrészek között továbbított adatokat, beleértve a szükséges kulcsadatok forgalmát is. Mivel a DNSSEC még nem terjedt el széles körben (a BIND v.8x még nem is támogatja teljesen), így most csak a TSIG-ek (Transaction Signature, csomagaláírás) használatával foglalkozunk.

Tegyük fel, hogy a tartomány fő- és másodlagos kiszolgálója között továbbított adatait szeretnénk titkosítani. Ehhez a következőket kell tennünk:

1. kulcsot kell készítünk a tartományhoz;
2. minden egyes kiszolgáló named.conf-jában létrehozunk egy, a kulcsot tartalmazó `key{}` bejegyzést;
3. minden kiszolgáló named.conf-jába beillesztünk egy `server{}` bejegyzést, melyben a 2. pontban létrehozott kulcsra hivatkozó kiszolgáló neve található.

Az első lépést a BIND dnskeygen programjával végezhetjük el a legegyszerűbben. Egy 512 bites, a fő- és a másodlagos kiszolgáló által egyaránt használható kulcsot a
`dnskeygen -H 512 -h -n <kulcsnev>`
paranccsal hozhatunk létre. A kimenet a `K<kulcsnev>.+157+00000.key`, illetve a `K<kulcsnev>.+157+00000.private` nevű szövegfájlba kerül. Itt a kulcs mindkét fájlnál azonos lesz, s körülbelül így fog kinézni: “ff2342AGFASsdfsa55BSopiue/-2342LKJDJlkjVVVvfjweovzp2OIPOTXUEdss2jsdfAAlskj==”.
A másik két lépéshez a named.conf fájlokat kell szerkesztenünk mindkét kiszolgálón (a kulcsnévnek mindkét gépen azonosnak kell lennie!):

```
key kulcsnev {
    algorithm hmac-md5;
     secret "<ide kell beilleszteni a kulcsot>";
}

server <a másik gép IP-címe> {
    transfer-format many-answers;
     # (a válaszokat kötegelve és nem egyenként
     # küldi)
    keys { kulcsnév; };
};
```

Figyeljük meg, hogy a `key{}` parancsnak meg kell előznie a rá hivatkozó többi parancsot (például a `server{}`-t). A kulcs és a kiszolgáló megadásának legjobb helye az `option{}` és az adatok között van. Most már csak újra kell indítanunk a namedet (a `kill -HUP` vagy az `ndc restart` paranccsal) mindkét kiszolgálón. Ezek után elmondhatjuk, hogy DNS-kiszolgálónk szinte tökéletes biztonságban van!

Kapcsolódó címek

A SANS Institute által kiadott tíz legnagyobb internetes biztonsági rész:
➲http://www.sans.org/topten.htm

Információk a DNS biztonságáról
BIND és a DHCPD honlapja:
➲http://www.isc.org/
Cricket Liu, a DNS Security Slides szerzőjének és a DNS and BIND társszerzőjének oldaláról letölthető PDF-állomány címe:
➲http://www.acmebw.com/papers/securing.pdf
A comp.protocols.tcp-ip.domains hírcsoport leggyakoribb kérdései:
➲http://www.intac.com/~cdp/cptd-faq/
„DNS Security Paper" (Craig Rowland):
➲www.psionic.com/papers/dns/

Néhány érdekesebb RFC
➲http://www.rfc-editor.org/
1035 A DNS-ről általában
1183 A forrásbejegyzések (RR) formája
2308 Negatív gyorstárazás
2136 Dinamikus frissítések
1996 DNS Notify (értesítések)
2535 DNS biztonsági bővítések

Néhány DNS/BIND biztonsági tanács
➲http://www.cert.org/
CA-99-14: Multiple Vulnerabilities in BIND
CA-2000-03: Continuing Compromises of DNS Servers
CA-98.05: Multiple Vulnerabilities in BIND
CA-97.22: BIND

Összefoglalás

Az itt ismertetett irányvonalak és módszerek jó kiindulópontként szolgálnak DNS kiszolgáló(i)nk biztonságos kialakításához. E módszerek tökéletesebb megértéséért javaslom, hogy olvassuk el a BIND felhasználói útmutatóját (a legtöbb bináris csomagban megtaláljuk, de a ➲http://www.isc.org/ címről is letölthető). Egy másik hasznos segédanyag Liu DNS Security című bemutatója (ez PDF-formátumban is hozzáférhető).
Ugyanilyen fontos, hogy minden BIND-üzemeltető iratkozzon fel legalább egy levelezési listára, mely a felfedezett biztonsági hibákkal foglalkozik, illetve jó tanácsokat ad mindenféle biztonsági kérdésben. Az én kedvencem a CERT, hiszen elég régóta működik ahhoz, hogy megbízhassunk benne, viszont kis terjedelmének köszönhetően könnyen kezelhető. A CERT legfrissebb jelentéseit is olvassuk el, hiszen a gondok ismerete elengedhetetlen egy biztonságos rendszer kiépítéséhez.

Michael D. Bauer (mick@visi.com) az ENRGI nevű hálózati tanácsadó cég minneapolisi képviseletén dolgozik. 1995 óta a Linux elkötelezett híve, az OpenBSD-nek 1997 óta szerelmese. Hírhedt ama „perverziójáról", hogy imád tökéletes rendszereket felépíteni a leghasználhatatlanabb gépekből is. Mick örömmel várja olvasóink kérdéseit.