Könnyű álmok (3. rész)

Az Interneten használt fontosabb protokollok.

A korábbi számok bevezetőnek szánt elméleti fejtegetései után ez alkalommal ismét vessük magunkat újabb elméleti fejtegetésekbe. Ahhoz, hogy elég mélyen átlássuk a rendszer támadhatóságának okait, részletesebb adatátviteli és programozás-módszertani ismeretekre lesz szükségünk.
A Linux rendszermag egy olyan erős védelmi alrendszert tartalmaz, amit egyes cégek tűzfal néven el is adnak: a csomagszűrőt. Ez a rendszermag része, és alkalmas mind a rendszer elérhetőségének, mind a rajta áthaladó (ha van ilyen) forgalomnak a szabályozására. Helyes beállításához a hálózat működésének alapos ismeretére van szükség. Jelen cikk célja az, hogy a csomagszűrő beállításához szükséges adatokat megismertesse, és ha már ilyen mélyen beleástuk magunkat a hálózatok rejtelmeibe, akkor tovább megyünk, és a később ismertetésre kerülő hálózati támadások megértéséhez szükséges „okosságot” is szétosztjuk. Azoknak a bátraknak, akik már ma este nekikezdenek a rendszermag csomagszűrőjének beállításához, van egy jó tanácsunk. Biztonsági szempontból jelenleg a 2.2-es rendszermag-sorozat tekinthető megbízhatónak. A cikk megírásának időpontjában már megszületett ugyan a 2.4-es sorozat első változata, mégis sok ellenőrzésre van szükség ahhoz, hogy érdemes legyen telepíteni egy biztonsági rendszerre. A jó tanács: amíg az időszerű sorozat utolsó számjegye (sublevel) nem megy 5 fölé, addig kiszolgálón nem érdemes a rendszermagot használni, mert olyan hiányosságok derülhetnek ki, amelyek a folyamatos üzemelést megzavarhatják. A tanács természetesen fokozottan érvényes a biztonsági rendszerekre. Ezzel kapcsolatban Solar Designer – az ismert és elismert biztonsági varázsló – azt írja, hogy az általa fejlesztett biztonsági kiegészítés 2.4-es rendszerhez használható változata (ez a rendszermag biztonsági lehetőségeit terjeszti ki) nem is várható addig, amíg a rendszermag el nem éri a 2.4.10-es változatot.
A továbbiakban a TCP/IP protokollcsalád alapjaival fogunk megismerkedni. Sajnos, minden részletre kiterjedő ismertetésre nincs lehetőségünk, hiszen a témakör rendkívül széles területet ölel át. Akik mélyebben szeretnének megismerkedni a TCP/IP protokollal, azoknak az [1.] és [2.] szakirodalmat ajánljuk tanulmányozásra. Feltételezzük, hogy az olvasó alapszinten ismeri a számítógépes hálózatokat és az OSI-modellt.

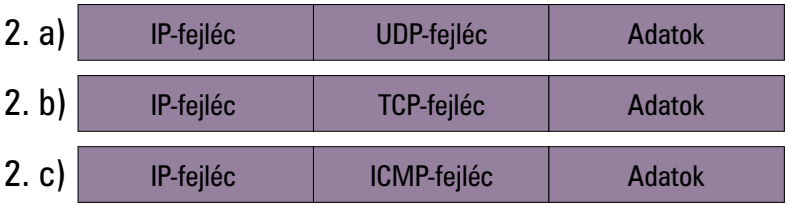
Az Internet anyanyelve

Az Internet adatátviteli rendszere a TCP/IP protokollcsaládra épül. A TCP/IP mind adatcsomag- (datagram) mind virtuális áramkör-szolgáltatást képes nyújtani a rá épülő alkalmazások számára. Az adatcsomag-szolgáltatás kapcsolatmentes, nem megbízható, viszont csekély felesleges adatátvitellel járó kapcsolatot nyújt. Előszeretettel alkalmazzák olyan esetekben, ahol a csomagok esetleges kiesése nem jelent komoly gondot, például üzenetszórással működő zenekiszolgálóknál. A virtuális áramkör-szolgáltatás kapcsolatalapú, nagy megbízhatóságú adatátvitelt tesz lehetővé. Ez teremti meg a lehetőségét annak, hogy a csomagok nem vesznek el, nem kettőződnek meg és sorrendjük sem cserélődik fel.
A protokollcsaládban a virtuális áramkör-szolgáltatást a TCP (Transmission Control Protocol), míg az adatcsomag-szolgáltatást az UDP (User Datagram Protocol) biztosítja. Az Internet legtöbb szolgáltatása TCP-re épül, mert az alkalmazásnak így nem kell kezelnie az adatátvitel közben fellépő hibák nagy részét. TCP-alapú protokoll például a webkiszolgálók és böngészők anyanyelve, a HTTP (Hypertext Transfer Protocol), vagy a leveleink megbízható továbbításáért felelős SMTP (Simple Mail Transfer Protocol).
Az UDP-re kevesebb, de nem kevésbé jelentős protokoll épül, például az Interneten használt nevek feloldásáért felelős DNS (Domain Name Service), vagy az órák összehangolását lehetővé tevő NTP (Network Time Protocol). Mind a TCP, mind az UDP egy közös rétegre, az IP-re (Internet Protocol) épül. Az IP mellett vezérlésre és hibajelzésre szolgál az ICMP (Internet Control Message Protocol).
A TCP/IP protokollcsaládnak még számos tagja van, de most csak a legáltalánosabban használt négy protokollal foglalkozunk.

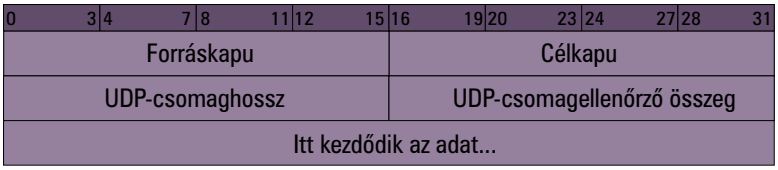
1. ábra Az IP fejlécének felépítése

IP protokoll

Az Internet Protocol feladata, hogy adatátvitelt tegyen lehetővé a hálózatba kapcsolt két számítógép között. Minden hálózatra kapcsolt rendszernek van egy egyedi azonosítója, ez az úgynevezett IP-cím, amelynek egyedinek kell lennie az egész Internetre nézve. Ez alól csak a kizárólag magánhálózatokban használható belső IP-címtartományok kivételek, ezeket viszont az Interneten nem lehet használni. Az IP-csomag egy fejlécből és a hozzá kapcsolt adat-



2. ábra Az Internet leggyakoribb csomagformátumai



3. ábra Az UDP fejléc felépítése

részből áll, ezek együttes hossza nem lehet több, mint 65535 bájt. Az IP-fejléc szerkezetét az *1. ábra* mutatja. Mivel a TCP/IP protokollcsaládot akár határozottan különböző jellemzőkkel bíró hálózatok összekapcsolására tervezték, ezért meg kellett oldani az esetleg nagyméretű IP-csomagok átjuttatását kisebb csomagméret továbbítását lehetővé tevő alhálózatokon. Ha az alhálózatba belépő csomag mérete nagyobb az adott hálózaton megengedettnél, az útválasztó a túlméretes csomagot feldarabolja. Ezt a folyamatot hívjuk darabolásnak (fragmentation). Az egyes darabok ezek után önálló csomagként utaznak a cél felé. A darabokat a célgép állítja össze az *Azonosító* az *MF jelző*, valamint a *Darabeltolás (Fragment offset)* segítségével. Szándékosan hibásan darabolt csomagokkal bizonyos támadások is kivitelezhetők, ha a célgép azok összeszerelésekor hibázik. Erről a későbbiekben lesz még szó.
A *Változat (Version)* az IP-csomag formátumára utal, jelenleg az IPv4 az általánosan elterjedt, de ez elvileg csak 2^{32} számítógép egyedi címzését teszi lehetővé (gyakorlatilag lényegesen kevesebbet). Mivel maholnap a fürdőszobamérlegnek is szüksége lesz IP-címre, így az Interneten új szabvány bevezetését tervezik: az IPv6-ot, ez nagyságrendekkel több rendszer címzését teszi lehetővé.
Ezt követi a *Fejléchossz – FH* mező, amely a fejléc hosszát adja meg 32 bites szavakban. Legkisebb értéke 5 (az ábrát áttanulmányozva jól látszik, hogy 20 bájtnál rövidebb IP-csomag nem létezik), legnagyobb értéke pedig 15, ami legfeljebb 60 bájt hosszú IP-fejléchez vezet. Így a kapcsolók (options) legfeljebb 40 bájtot foglalhatnak el. A *ToS* (Type of Service) mező az adatátvitel jellemzőit befolyásolja. Ennek ismertetésére most nem térünk ki [3.]. A *Csomaghossz* mező a teljes csomag hossza a fejlécet is beleértve, bájtokban megadva. Mivel ez 16 bit, így egy csomag legfeljebb 2^{16}-1, azaz 65535 bájt hosszúságú lehet. Az *Azonosító* mező célja, hogy lehetővé tegye a feldarabolt IP-csomag későbbi összeszerelését. Minden darab az eredeti IP-csomag azonosítóját tartalmazza. A *Jelzők (Flags)* mező két jelzőbitet tartalmaz. A *DF* (Dont Fragment) bit a csomag darabolását tiltja, míg az *MF* (More Fragments) bit mutatja, hogy a csomag darabokból áll, és nem ez az utolsó rész. A *Darabeltolás (Fragment offset)* mező az adott darab címe az eredeti IP-csomagban 8 bájtos osztásokkal. A *TTL* (Time to Live) mező a csomagélettartam korlátozását szolgálja. Értéke másodpercenként, de legalább útválasztón áthaladásonként eggyel csökken. Ha lenullázódik, a csomagot el kell dobni. Célja az, hogy az útválasztók hibájából hurokba kerülő csomag ne keringhessen örökké a hálózaton. A *Protokoll* mező jelzi, hogy milyen magasabb szintű protokollhoz tartozik az adott csomag, például TCP vagy UDP. A *Fejlécellenőrző összeg* a fejlécet hivatott védeni az adatátvitel közbeni hibáktól. A *Forráscím* a feladó, míg a *Célcím* a címzett(ek) azonosítására szolgál. Ezek után következnek az egyes *Kapcsolók*, amelyek a csomag továbbítását befolyásolhatják (ilyenek például a Record Route, Loose Source Routing, Strict Source Routing). Jelenleg ezekre sem térünk ki. A *Kitöltés* célja, hogy a fejléc után következő adatrész 4 bájttal osztható címre kerüljön, hiszen a *Fejléc hossz* mező alapegysége is ez.

UDP protokoll

A User Datagram Protocol célja, hogy adatcsomag-szolgáltatást (datagram service) nyújtson a felsőbb rétegek számára. Az adatcsomag-szolgáltatás kapcsolatmentes, így rövid üzenetek továbbítása során nem kell számolni a kapcsolat felépítéséből és lebontásából származó felesleges hálózati forgalommal és időveszteséggel.
Hátránya azonban, hogy az adatcsomag-szolgáltatás nem megbízható, azaz a csomag elveszhet, többszöröződhet vagy a csomagok sorrendje felcserélődhet. Ezért az adatcsomag-alapú protokollokat használó alkalmazásoknak fel kell készülniük e gondok kiküszöbölésére. Sokszor mégis előszeretettel használnak adatcsomag-szolgáltatást, mivel ez rendelkezik egy hasznos lehetőséggel a kapcsolatalapú protokollokkal szemben, ez pedig az úgynevezett üzenetszórás (broadcast) vagy csoportcímzés (multicast). E lehetőséggel több gép számára küldhetjük el ugyanazt az üzenetet, egyetlen csomag felhasználásával.
Egy UDP-csomag három részből áll *(2. a) ábra)*. Az első rész maga az IP-fejléc, melyet közvetlenül az UDP-fejléc követ. Ezek után jönnek a tényleges adatok. Az UDP-fejléc szerkezete a *3. ábrán* látható. Az UDP-nél az IP-hez képest egy új fogalom jelent meg, ez a kapu. A kapu célja, hogy a megcímzett gépen belül azonosítsa a küldő vagy a fogadó szolgáltatást. A küldőnek a címzett kapu címét ismernie kell. Erre két lehetőség is rendelkezésre áll:

- az alkalmazás szabványos (well-known) címet használ (pl.: DNS, NTP),
- dinamikusan foglal kaput, és a lefoglalt kapu címét egy szabványos címen levő brókerrel közli (pl.: RPC szolgáltatások, ahol a bróker a portmapper vagy rpcbind).

Az UDP fejléc első mezője a *Forráskapu*, ez a feladó kapucímét tartalmazza. A feladónak nem kell megnyitott kapuval rendelkeznie. A következő mező a *Célkapu*, amely a szolgáltatást azonosítja a célgépen belül. Ezt követi az *UDP csomaghossz* mező, mely megadja a csomag hosszát az IP-fejléc nélkül. Értéke 8-tól 65 515-ig terjedhet (hiszen bele kell férnie egy IP-csomagba, ahol az IP-fejléc legalább 20 bájt). Az *UDP-csomagellenőrző összeg* az UDP-fejléc, adatrész, valamint az IP-fejléc legfontosabb részeiből számolódik, és az átvitel közbeni esetlegesen fellépő sérülések észlelése a célja. Az UDP-fejléc után következik az adatrész, ennek hossza legfeljebb 65 507 bájt. UDP-csomagok esetében az IP-fejléc *Protokoll* mezője 17 értékű.
Mivel sem az IP, sem az UDP nem ad biztosítékot a feladó hitelességére vonatkozóan, így ezek a csomagok könnyen hamisíthatók. Ironikusan fogalmazva, UDP-csomag esetében csak egyvalamiben lehetünk biztosak: a címzettben (később látni fogjuk, hogy néha még ebben sem).

TCP protokoll

A Transmission Control Protocol az egész protokollcsalád legbonyolultabb tagja, így ismertetésében csak a legfontosabb tulajdonságaira térhetünk ki [5.]. Mint az előzőekben említettük, a TCP célja, hogy kapcsolatalapú, megbízható szolgáltatást nyújtson a magasabb protokollrétegek felé. A TCP a küldendő bájtokat szakaszokba (segment) csomagolja, amelyeket eljuttat a megcímzett gép megadott kapujára (a kapu fogalma itt is létezik az UDP-hez hasonlóan, de az UDP és TCP kapuknak nincs semmi köze egymáshoz). Adatátvitel előtt a kapcsolatot fel kell építeni, az átvitel végén pedig le kell bontani. A kapcsolattartó gépek TCP protokollrétegei teszik lehetővé, hogy az adatok megfelelő módon átvitelre kerüljenek. A forgalommentes időszakban a TCP képes folyamatosan ellenőrizni a kapcsolat működőképes állapotát (keepalive).

4. ábra A TCP fejléc felépítése

Az adatátvitel biztonságáért a vett adatot a fogadóoldalnak nyugtáznia kell a küldő felé. Ahhoz, hogy az átvitel csatornáját minél jobban kihasználhassa, az egyszerű *megáll és vár* protokoll helyett egyfajta csúszóablakos protokollt használ forgalomszabályozással [1.]. A forgalomszabályozás célja, hogy egy gyors adó ne áraszthassa el a lassabb vevőt. A csúszóablakos protokollok lényege, hogy az adás és a nyugtázás egymással átlapolva zajlik, így a kapcsolat-tartásra használható csatorna kihasználtsága nagy késleltetésű vonalak esetén is közelít az eszményihez.

A kapcsolat felépítésének két lehetséges módja van. Passzív felépítéskor a futó folyamat létrehoz egy kaput, amelyen képes bejövő kapcsolatokat fogadni. Egy bejövő kapcsolatépítési kérés elfogadásakor a kapcsolat a két végpont között létrejön. Aktív kapcsolatépítéskor a kezdeményező kapcsolatépítési kérést küld a túloldalnak. Ha a túloldal a kérést elfogadja, a kapcsolat felépül.

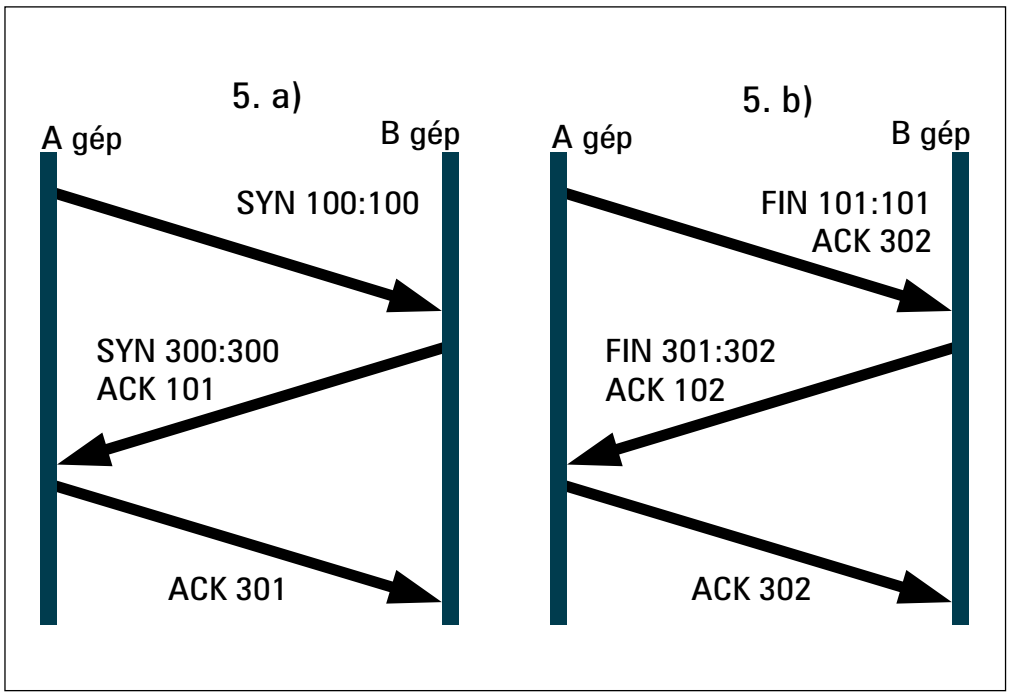
A TCP szakasz felépítése a *2.b) ábrán* látható. A csomag elején az IP-fejléc található. Ezt követi a TCP-fejléc, ez látható a *4. ábrán*, ezután következnek az adatok. Az első mező a *Forráskapu*, ezt követi a *Célkapu*. Utánuk a *Sorszám* jön, amely a szakasz legelső bájtjának sorszáma, amennyiben a *SYN* bit értéke 0. Minden bájt egyesével sorszámozott. Ha a *SYN* bit értéke 1, akkor a *Sorszám* a *Kezdő sorszámot* (ISN, Initial Sequence Number) jelenti, és a legelső adatbájt sorszáma a *Sorszám+1*. A következő mező a *Nyugtasorszám*. Ha a fejlécben az *ACK* bit 1 értékű, akkor ez a mező azt a *Sorszámot* tartalmazza, amelyet venni szeretne (az ez alatti bájtokat modulo 2^{32} már vette). Az *Adateltolás* (Data Offset) mező az adatok elhelyezkedését mutatja a TCP-fejléc elejétől 4 bájtos egységekben (ez egyébként a TCP-fejléc hossza). Mivel a TCP-fejléc legkisebb hossza 20 bájt, így e mező értéke legalább 5. Az egész TCP-fejléc hosszát szintén ez a mező korlátozza legfeljebb 60 bájtra.

A *Vezérlőbitek* mező a fejléc részeinek érvényességét biztosítja, illetve vezérlési feladatokat lát el. Az *URG* bit a *Sürgős adat mutató* érvényességét jelzi. Az *ACK* bit a *Nyugtasorszám* mező érvényességére utal. A *PSH* bit szerepe, hogy utasítsa a vevőoldalt az eddig fogadott adat eljuttatására a felsőbb rétegek felé. Az *RST* jelző a kapcsolat törlésére szolgál (pl.: a kapu nincs nyitva). A *SYN* jelző a kapcsolatfelépítési, míg a *FIN* a kapcsolatbontási kérelmet jelzi. Természetesen ezen jelzőknek csak bizonyos kombinációja használható, egyebek tiltottak (pl.: *SYN+FIN*). Az *Ablakméret* mező a *Nyugtasorszámtól* kezdve venni szándékozott bájtokat jelöli. Ennek célja, hogy az adó oldal ne küldjön több adatot, mint amennyi pufferrel a vevő rendelkezik e kapcsolat kiszolgálására. A következő mező a *TCP-ellenőrző összeg*, amely az UDP-csomagéhoz hasonlóan számítható. A *Sürgős adat mutató* csak az *URG* bittel együtt értelmes, szerepe az általános adatfolyamon kívüli fontos adat jelzése. Ezt követhetik a *TCP kapcsolók*, majd a *Kitöltés* mező. A *Kitöltés* mező célja megegyezik az IP-fejlécnél látottakkal, hiszen a fejléc méretét itt is 4 bájttal osztható méretben lehet csak megadni. TCP-csomagok esetében az IP-fejléc *Protokoll* mezője 6 értékű.

A TCP a kapcsolat felépítésre háromszintű kézfogást használ, amint az az *5.a) ábrán* látható. A kapcsolatot kezdeményező fél egy *SYN*-es csomagot küld a túloldalnak, amelybe elhelyezi a saját *Kezdő sorszámát* (a kezdő sorszám választása mindkét oldalon roppant fontos, mint azt majd a következő részekben látni fogjuk). Amennyiben a

Az ICMP protokoll típusai

Típus	Leírás
0	echo reply
3	destination unreachable
4	source quench
5	redirect
8	echo request
9	router advertisement
10	router solicitation
11	time exceeded
12	parameter problem
13	timestamp request
14	timestamp reply
15	information request
16	information reply
17	address mask request
18	address mask reply



5. ábra TCP kapcsolat felépülése és lebomlása

túloldal elfogadja a kapcsolatot, akkor erre egy *SYN+ACK* csomaggal válaszol, ahol a *Sorszám* a kapcsolatépítési kérést elfogadó gép *Kezdő sorszáma*, a *Nyugtasorszám* mező pedig a kezdeményező *Kezdő sorszám+1* értéket tartalmazza. A harmadik és a kapcsolatépítést befejező lépésként a kezdeményező gép egy *ACK* csomagot küld, amivel visszaigazolja a fogadó gép által választott *Kezdő sorszám+1* értéket. Amennyiben a túloldal nem kívánja a kapcsolatépítést, akkor egy *RST* csomagot küld, amelyben a *Nyugtasorszám* a kezdeményezési kérés *Kezdő sorszámának* eggyel növelt értékét tartalmazza. A TCP-kapcsolatok full-duplexek (az adatok egymástól függetlenül közlekedhetnek mindkét irányba), így a kapcsolat csak akkor záródik le, ha mindkét adatirány lezárult. A kapcsolat lebontásának menete az *5.b) ábrán* látható. A lezárást kezdeményező fél egy *FIN* csomagot küld, amelyet a túloldal visszaigazol, ha minden eddig küldött adatot sikeresen vett. A kapcsolat lezáródását jelzi az alkalmazás felé. Erre a helyesen megírt alkalmazás lezárja a kapcsolatot, amire szintén egy *FIN* csomag lesz a válasz, de most ellenkező irányba. A bontást kezdeményező oldal a vett *FIN* csomagra szintén egy *ACK* csomaggal válaszol. Ennek hatására a kapcsolat bontása lezárul. Mivel a TCP kapcsolatalapú protokoll, és a kapcsolatépítés háromszintű kézfogást használ, így a címek hamisítása jóval behatároltabb és nehezebb. A későbbi részekben még kitérünk a TCP/IP protokollcsalád biztonsági kihatásaira is, ott majd részletesebben is megvizsgáljuk ezt a témakört.

ICMP protokoll

Az Internet Control Message Protocol lehetővé teszi az IP-csomagok továbbítása folyamán felmerült hibák kezelését, vagy egyéb vezérlőüzenetek továbbítását. Az ICMP-csomagok szerkezetét a *2.c) ábra* mutatja. A *Típus* mező az üzenet tárgykörét határozza meg, a *Kód* mező a *Típus* finomítására szolgál. Az I*CMP-ellenőrző összeg* a teljes ICMP-üzenetből számolt. Az adatrész tartalma a *Típus/Kód* páros értékétől függ. ICMP-csomagok esetében az IP-fejléc *Protokoll* mezője 1 értékű.

A *táblázat* a legfontosabb *Típusokat* határozza meg. A jól ismert ping parancs egy `ECHO REQUEST` ICMP-üzenetet küld a megcímzett gépnek, amely erre egy `ECHO REPLY` típusú üzenettel válaszol. A `DESTINATION UNREACHABLE` típusú üzenetek jelzik, ha a csomag valamely okból nem érhette el célját. A lehetséges kódok finomítják az üzenet jelentését, például a hálózat vagy a számítógép nem érhető el, darabolni kellene a csomagot, de a darabolás tiltott (az IP-csomag *DF* bitje 1 értékű), érvénytelen protokoll stb. A `PARAMETER PROBLEM` típus a csomag hibás szerkezetére utal. A `SOURCE QUENCH` csomagot torlódásvezérlésre szánták, viszont sportszerűtlen viselkedése miatt egyre ritkábban használják. A `TIME EXCEEDED` típus jelzi, hogy a lehetséges csomagélettartam lejárt, vagy a darabolt csomag néhány darabja nem érkezett meg. A `REDIRECT` típus célja, hogy helyesbítse a küldő gép útvonaltábláját. Miután megismertük a hálózatokon közlekedő csomagok felépítését, sorozatunk következő része a hálózati védelem tervezéséről és kivitelezéséről szól majd. Megismerjük a csomagszűrők beállításának ökölszabályait és azokat az eszközöket, amelyek segítségével az esetleges beállítási hibákat meg tudjuk találni.

Hivatkozásjegyzék

[1.] Andrew S. Tannenbaum: Számítógépes hálózatok (ISBN 963-545213-6)
[2.] W Richard Stevens: TCP/IP Illustrated, Volume 1 (ISBN 0-201-63346-9)
[3.] RFC791: Internet Protocol
[4.] RFC768: User Datagram Protocol
[5.] RFC793: Transmission Control Protocol
[6.] RFC792: Internet Control Message Protocol

Mátó Péter (atya@andrews.hu), informatikus mérnök és tanár. Biztonsági rendszerek ellenőrzésével és telepítésével, valamint oktatással foglalkozik. 1995-ben találkozott először linuxos rendszerrel. Ha teheti, kirándul vagy olvas.

Borbély Zoltán (bozo@andrews.hu), okleveles mérnök-informatikus. Főként Linuxon futó számítógépes biztonsági rendszerek tervezésével és fejlesztésével foglalkozik. A 1.0.9-es rendszermag ideje óta linuxozik. Szabadidejét barátaival tölti.