# Informatikai technológiák laboratórium 2.
(BMEVIMIA429)

# Komplex alkalmazási környezetek felderítése és menedzsmentje
(Mérési segédlet)

Szatmári Zoltán, Izsó Benedek
Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

2013. szeptember 15.

## Tartalomjegyzék

# 1. Bevezetés

Az informatika területén dolgozzunk akár szoftver-fejlesztőként vagy rendszerüzemeltetőként, naponta találkozunk komplex alkalmazási környezetekkel, többrétegű informatikai infrastruktúrákkal. Legyen szó egy összetett, üzleti szolgáltatást nyújtó rendszerről vagy webes alkalmazásfejlesztésre használt tesztrendszerről, ezek tervezése, megismerése és üzemeltetése igényli a korábbi félévekben megtanult ismeretek együttes, gyakorlati alkalmazását. Összetett rendszerek fejlesztése és üzemeltetése közben gyakorta szembesülünk általunk nem ismert vagy furán viselkedő konfigurációkkal, melyeknek a megismerése, és felderítése komoly feladat. Az együttműködő komponensek megfelelő összehangolásához, működés közben pedig azok monitorozásához, hiba esetén javításához, összetett alkalmazás környezet ismeret szükséges.

A mérés célja, hogy egy több szerverre elosztott, komplex szolgáltatást nyújtó infrastruktúrát a rendelkezésre álló eszközökkel felderítsünk, az ilyen környezetben felmerülő jellemző konfigurációs lépéseket elsajátítsuk, és megismerkedjünk a hibakeresés, diagnosztika eszköztárával. Célunk, hogy a korábbi félévekben elsajátított ismereteket a gyakorlatban alkalmazzuk *önálló problémamegoldás* során. A mérésben egy kezdetben ismeretlen felépítésű web- és adatbázisszolgáltatást nyújtó infrastruktúrával és annak monitorozásával kapcsolatos feladatokat tűzünk ki, melyeket a megismert eszköztár önálló alkalmazásával kell megoldani.

# 2. Szükséges előismeretek

A mérés során a következő tárgyakban elsajátított előismeretekre építünk, melyek gyakorlati alkalmazása szükséges a mérés elvégzéséhez.

- Számítógép hálózatok:
  a TCP/IP alapú hálózatok működése (IP címek, IP cím osztályok, alhálózatok, NAT működési elve, TCP/UDP protokollok különbségei, MAC címek használata) [1]

- Mérés laboratórium 4:
  Ethernet, TCP/IP mérés (hálózati diagnosztika eszközök), Alkalmazási réteg mérés (HTTP protokoll, Wireshark eszköz), UNIX/Linux mérés (alapvető Linux ismeretek)

- Intelligens rendszerfelügyelet:
  IT infrastruktúra modellezése, rendszermonitorozás témaköre

- Adatbázisok, Szoftver laboratórium 5:
  SQL, alapismeretek

# 3. Hálózatok menedzsmentje

Szolgáltatások összehangolásához nem csak magukkal az alkalmazásokkal, hanem az azt kiszolgáló infrastruktúra topológiájával, működésével és konfigurálásával is tisztában kell lenni. Ebben a fejezetben a méréshez szükséges alapvető hálózatmenedzsment ismereteket mutatjuk be.

## 3.1. Hálózati alapismeretek, útvonalválasztás

A hálózaton minden fizikai eszköznek van egy egyedi azonosítója, amit MAC címnek hívnak (például `00:50:56:c0:00:08`). Ha egy adott IP címen lévő gépet meg szeretnénk szólítani, akkor végeredményben annak MAC címére is szükség van, ahol az *IP cím - MAC cím összerendelésért* az ARP (Address Resolution Protocol) felel. Ez tipikusan „magától jól működik”, de azért van lehetőség ezen gyorsítótárnak a (kézi) kezelésére, a következő paranccsal:

| **arp** - ARP táblázat megtekintése és módosítása |
|---|

Amennyiben nagy méretű hálózatról van szó, akkor könnyen belátható, hogy a fenti módszer nem skálázódik. Éppen ezért bevezették az IP címek alapján történő *útvonalválasztást.*

A hálózatra való kapcsolódás vagy előre (kézzel) beállított IP paraméterekkel, vagy DHCP segítségével kiosztott IP paraméterekkel lehetséges. Ilyen IP paraméter az *IP cím*, az *alhálózati maszk*, az *alapértelmezett átjáró* és a *névszerver(ek).*

| **Paraméter** | **Decimális** | **Bináris** |
|---|---|---|
| IP cím | 172.16.219.138 | 10101100 00010000 11011011 10001010 |
| Alhálózati maszk | 255.255.255.240 | 11111111 11111111 11111111 11110000 |
| Átjáró | 172.16.219.142 | 10101100 00010000 11011011 10001110 |
| Hálózat | 172.16.219.128 | 10101100 00010000 11011011 10000000 |
| Broadcast | 172.16.219.143 | 10101100 00010000 11011011 10001111 |
| Névszerver | 8.8.8.8 | 00010000 00010000 00010000 00010000 |

A fenti példában egy gép IP paraméterei láthatóak, melynek címe 172.16.219.138, alhálózati maszkja pedig 255.255.255.240. A maszk és IP cím között elvégzett ÉS művelet megadja az alhálózat legkisebb IP címét,

amit gyakran nem osztanak ki géphez, hanem „hálózati címként” ismeretes (itt 172.16.219.128). Magát az alhálózatot az (al)hálózati cím és maszk azonosítja, ami CIDR jelöléssel a példában 172.16.219.128/28, ahol a 28 az a maszkban lévő 1 bitek száma. Az alhálózaton használt legnagyobb IP címet pedig úgy kapjuk, ha a hálózati címben 1-be állítjuk azokat biteket, ahol a maszk 0 bitjei vannak. Tipikusan ez a broadcast cím (jelen esetben 172.16.219.143), amely IP címre küldött csomagokat üzenetszórással mindegyik gép megkapja az alhálózaton.

Ha egy gép az adott alhálózatba szeretne csomagot küldeni, azaz a csomag cél címe 172.16.219.128-172.16.219.143 tartományba esik, akkor az ARP táblában eltárolt MAC című gépnek küldheti a csomagot. Ellenkező esetben pedig az átjárónak kell címeznie a csomagot, amit tehát az eredeti IP címmel, de az átjáró MAC címével kell kiküldeni. Ezek után már az átjáró feladata a kézbesítés, ami az IP cím alapján más alhálózatba továbbítja a csomagot. Ezt így folytatva (hiba mentes esetben) célba érkezik a kiküldött IP csomag.

Az emberek nem szívesen jegyeznek meg számokat (IP címeket), mert annál sokkal könnyebb nevekre emlékezni (pl. google.com). A nevek IP címmé fordítását a névszerverek végzik, így szükséges ezen gépek IP címének ismerete is (a példában 8.8.8.8). Megfigyelhető, hogy míg a gateway címe az alhálózatba kell, hogy essen, addig a névszerver bármely elérhető gép lehet.

A hoszt hálózati (IP) rétegbeli információit a következő parancsok segítségével tudjuk lekérdezni, módosítani, vagy feltérképezni UNIX és Linux alapú rendszerekben. A dokumentum további részében is a következő jelölést követjük a szöveghez kapcsolódó eszközök leírására.

**ifconfig** - hálózati interfészek információinak megjelenítése és módosítása

**route** - az adott hoszt routing táblájának megjelenítése

**traceroute** - távoli hoszthoz vezető útvonal elemeinek meghatározása

**ping** - Távoli hoszt elérésének és válaszidejének vizsgálata

## 3.2. A DNS szolgáltatás

A DNS szolgáltatás egy internet méretű adatbázis, melyben különböző típusú rekordok (bejegyzések) találhatóak. Leggyakrabban az `A` (Address) típusú rekordra vagyunk kíváncsiak, mely név alapján megmondja az ahhoz tartozó IP címet. (Például: mi az inf.mit.bme.hu IP címe? 152.66.252.223.)

A névfeloldás visszafelé is megtehető: egy reverse DNS lekérdezéssel megkapható egy adott IP címhez tartozó valamely név. Ezen címhez nevet rendelő rekordoknak `PTR` a típusa.

A DNS struktúráját tekintve hierarchikus felépítést követ. Az egyes hierarchiaszintek a domain név ponttal elválasztott elemeinek felelnek meg. A hierarchia gyökere a ROOT domain, ennek gyermekei a legfelsőbb szintű domain (TLD, Top Level Domain) nevek, melyek a különböző országkódok és általános célú végződések (.com, .org, .nasa, .hu, .xxx, stb.) lehetnek. A fa csomópontjai a domain nevek logikai felépítését követik, de egy csomóponthoz a gyakorlatban több DNS szerver tartozik, így biztosítva a terhelés elosztását és a megbízhatóságot.

A *DNS névfeloldás* folyamata során a domain névhez tartozó IP címet a fa gyökerénél kezdjük el meghatározni és a hierarchiában lefele haladva folytatjuk egészen addig, amíg egy adott DNS szerver a kérdést megválaszolja. Ezek a köztes DNS szerverek *autoritatív DNS szerverek*, azaz csak olyan információkat adnak vissza, amiért ők a felelősek (a fa hierarchiában ahhoz a csomóponthoz tartozik). A feloldás folyamatát az inf.mit.bme.hu domain példáján keresztül mutatjuk be:

- A kliens valamely ROOT névszervertől megkérdezi, hogy az inf.mit.bme.hu domain milyen IP-re oldható fel. A ROOT szerver válaszol, hogy ezt a címet ő nem tudja feloldani, de a .hu végződésű domaineket az ns.nic.hu, ns2.nic.fr, b.hu, c.hu DNS szerverektől lehet kérdezni, mert ők hivatottak a .hu végződésű domain nevek feloldására. Természetesen a végtelen ciklus elkerülése érdekében a visszaadott névszerverek IP címei is kiderülnek, vagy azért mert a felelős DNS szerver hozzáfűzi a válaszhoz, vagy azért, mert más úton könnyen feloldható. A ROOT szerverek címei pedig közismertek, és nagyon ritkán változnak [2].

- A kliens ezután az egyik .hu DNS szervertől megkérdezi, hogy a inf.mit.bme.hu domain milyen IP-re oldható fel. Az DNS szerver válaszol, hogy ezt a domaint ő nem tudja feloldani, de a bme.hu tartományért a nic.bme.hu (152.66.115.1) vagy az ns.bme.hu (152.66.116.1) címeken elérhető DNS szerverek felelősek, ezen szerverek hivatottak a bme.hu tartományba tartozó domain nevek feloldására.

- A lekérdezés folyamata ebben a formában addig folytatódik, amíg végül eljutunk a mit.bme.hu DNS szerveréig, amely már képes feloldani a kért domain nevet, és visszaadja annak `A` rekordját, azaz a kért IP címet.

A *DNS lekérdezés* az a folyamat, amikor a kliens alkalmazás egy általa ismeretlen domain névvel találkozik, és kísérletet tesz az ahhoz tartozó IP

cím lekérdezésére. Ez annyiban tér el a feloldási folyamattól, hogy a korábbi DNS lekérdezések eredményei különféle ideiglenes gyorsítótárakba kerülhetnek, vagy fix összerendelés lehet előírva. Így az előbb írt teljes feloldási folyamat bizonyos esetekben nem is fut le. A lekérdezési folyamat általános váza a következőkben látható, ami egyből visszatér, amint egy fázisban megtalálta az eredményt:

- A kliens megkérdezi az operációs rendszertől, hogy fel tudja-e a domain nevet oldani, ezáltal delegálja a feladatot számára.
- Az operációs rendszer először ellenőrzi, hogy a helyi `hosts` fájlban statikusan megadásra került-e a domain név - IP párosítás, mert ha igen, akkor visszaadja. A `hosts` állományban minden operációs rendszeren előírhatunk fix domain név - IP cím kötéseket, ami Linux rendszerekben a `/etc`, míg Windows alatt a `%WINDIR%\System32\Drivers\etc` mappában található.
- Az operációs rendszer korábbi találat hiányában delegálja a kérdést a hálózati beállítások során megadott valamely *rekurzív DNS* névszerver felé (pl. 8.8.8.8). A rekurzív szerverek nem tárolnak csomópontra vonatkozó autoritatív információkat, hanem feladatuk a névfeloldás végigjátszása, illetve az eredmények gyorsítótárba helyezése, és ezáltal a kliensek gyors kiszolgálása.
- A delegált kérdésre ezért a rekurzív névszerver megvizsgálja a saját helyi gyorsítótárát, és ha benne van az érvényes válasz, visszaadja.
- Amikor a névszerver gyórsítótárában nem található meg a kért bejegyzés, akkor elindítja a *DNS feloldás* folyamatát.

Az internet méretű DNS adatbázis lekérdezésére a következő két parancs használható:

| **nslookup, dig** - DNS adatbázis lekérdezése (pl. névfeloldás kérése), opcionálisan megadva a feloldásra megkérendő szerver címét, vagy a lekért DNS rekord típusát |
|---|

## 3.3. Tűzfalbeállítások

A tűzfal általánosságban egy olyan hálózati alkalmazás (sokszor dedikált eszközön), amely a rajta áthaladó forgalmat „jólformálttá” teszi annak érdekében, hogy az egyes interfészeihez tartozó hálózatokat kölcsönösen megvédje

a többiből érkező "érvénytelen" (vagy legalábbis érdektelen) kommunikációtól [3].

A kívánt hatás a következő különböző tűzfalmegoldásokkal érhető el:

- A **állapotmentes csomagszűrő** minden egyes hálózati csomagról önmagában dönti el, hogy áthaladhat-e a tűzfalon. A döntéshez felhasználhat mindent, amit az adott csomagról tud, akár a tartalmát is, de általában a fejléc alapján születik a döntés. Előnye, hogy állapotmentes, így sokkal kevesebb információt kell karbantartania, sokkal gyorsabb működésre képes. Hátránya, hogy rugalmatlan, és csak egyszerű feltételek alapján tudja szűrni vagy manipulálni a csomagokat. Például nem tudja megállapítani, hogy egy új TCP-kapcsolat egy már fennálló FTP-kapcsolathoz tartozó adatkapcsolat-e.

- A **kapcsolatkövető csomagszűrő** olyan tűzfal, amely nyomon követi a rajta keresztül felépített hálózati kapcsolatok állapotát, és ezt az információt is fel tudja használni a döntés során. Előnye, hogy állapottal rendelkező kapcsolatokat is tud kezelni, így sokkal rugalmasabb. Hátránya, hogy az állapot-információ nyilvántartásához memóriára és többlet műveletekre van szüksége, ami miatt könnyebb sikeres DoS-támadást indítani ellene.

A mérési környezetben az Ubuntu Linux operációs rendszeren Netfilter/IPTables [4] tűzfal fut. Alapvető működésének megértéséhez a következő fogalmak ismerete szükséges:

- **Illesztés (match)** történik, ha a megfogalmazott feltételt kielégíti egy hálózati csomag. A feltételekben kényszerek írhatóak fel többek között a forrás és cél IP címre, a protokollra, a forrás- vagy célportra, a kapcsolat állapotára és a hálózati interfészre vonatkozóan. Például a 172.16.219.138 IP cím 143-as portjára, TCP protokollon érkező csomagokkal szeretnénk valamit kezdeni.

- **Akció (target)** írja le a döntést arról, hogy mi történjen egy illeszkedő csomaggal. Ez többek között lehet engedélyezés (allow), eldobás (deny), visszautasítás (reject) vagy további vizsgálat előírása.

- A **szabály (rule)** egy illeszkedés és egy akció együtteséből áll. Például a 143-as portra érkező csomagok bejövetelét engedélyezzük.

- A **lánc (chain)** szabályok sorozatából áll. A csomag sorban halad a szabályokon, és amint a tűzfal egy illeszkedést talál, alkalmazza annak a szabálynak az akcióját.

- **Alapértelmezett akció (default policy)** írja le, hogy egy láncon végig ért, egyik szabályra sem illeszkedő csomaggal mi történjen. Például dobjuk el (deny), és naplózzuk ennek tényét.

A könnyebb megértést elősegítendő nézzük a következő példát! Egy szerveren céges postafiókokat üzemeltetünk, melyeket IMAP protokoll segítségével (143-as port) kérhetnek le a tulajdonosaik. A céges szabályok értelmében ezt otthonról nem, csupán cégen belülről érhetik el az 172.16.219.128/28 alhálózat IP címeiről. Ebben az esetben a levelező szerver tűzfalában a következő szabálynak kell szerepelnie:

```
Default: deny (incoming)

To         Action      From
--         ------      ----
143/tcp    ALLOW IN    172.16.219.128/28
```

Az IPTables önálló használata esetén ezen egyszerű szabály megadása is komoly szakértelmet, részletes paraméter ismeretet követel meg a felhasználóktól. A rutin jellegű tűzfalbeállítások elvégzéséhez több alkalmazás is rendelkezésünkre áll, melyek elfedik előlünk az IPTables bonyolult szintaktikáját, és egyszerű felületet biztosítanak a tűzfalbeállítások megtételéhez.

A mérés során az UFW (Uncomplicated Firewall) alkalmazást használjuk, mely gondoskodik a szerver indulásakor a tűzfal szabályok betöltéséről és a futás közbeni menedzselésről. A következő parancssori utasításokat célszerű megismerni:

- `ufw help`: Egyszerű súgó a paraméterezésről.
- `ufw status [verbose|numbered]`: A jelenleg beállított szabályok kilistázása, amihez kiegészítésként lehet kérni bővebb információt (verbose) ami az alapértelmezett akciót is kiírja, illetve a szabályok láncban elfoglalt sorszámát is ki lehet íratni (numbered).
- `ufw allow 143/tcp`: Új szabály felvétele, amely a TCP 143-as porton engedélyezi a bejövő forgalmat.
- `ufw delete allow 143/tcp`: Az előző engedélyező szabály törlése.
- `ufw deny 143/tcp`: Új szabály felvétele, amely a TCP 143-as porton tiltja a bejövő forgalmat.

A tűzfalbeállítások távoli szerkesztése során vigyázzunk, hogy téves módosítás esetén akár saját magunkat is kitilthatjuk, például ha SSH kapcsolaton keresztül dolgozunk, és letiltjuk az SSH (22) portot. Ilyen esetekben legtöbbször csak a konzolnál tudjuk ezt a tévedést kijavítani, ami egy távoli gép esetén valóságban több száz kilométer utazást is jelenthet.

# 4. Hálózati szolgáltatások menedzsmentje

Ebben a fejezetben először azt fogjuk áttekinteni, hogy hogyan lehet a hálózat topológiáját, illetve más gépek szolgáltatásait, egyéb tulajdonságait feltérképezni. Ezután a gépen belüli szolgáltatások feltérképezését ismerjük meg (Linux környezetben), majd két, szöveges protokollt használó alkalmazást (SMTP és HTTP kiszolgálót) is megvizsgálunk. Bízunk abban, hogy ezen elterjedt szolgáltatások működési elve, az azokban felhasznált ötletek, hozzájárulnak a későbbi tanulmányok/munka során írt hálózati alkalmazások színvonalas tervezéséhez.

## 4.1. Hálózatfelderítés

Hálózatfelderítés során feladatunk a csatlakoztatott eszközről elérhető további eszközök felderítése, azok hálózati paramétereinek és a rajtuk futó szolgáltatások felderítése.

Egy hálózati hosztról kívülről vizsgálódva viszonylag kevés információt tudunk megállapítani: a hálózati kommunikációhoz használt címeit, így az *IP címét* (esetleg a *fizikai (MAC) címét*), és azokat a TCP és UDP *port*okat, melyeken hálózati szolgáltatásokat nyújt. Az ilyen információkból különféle következtetéseket tudunk levonni, de további vizsgálatokra lehet szükség. Például, ha egy hoszton a 22-es port elérhető, akkor valószínű, de nem biztos, hogy SSH szolgáltatás fut rajta.

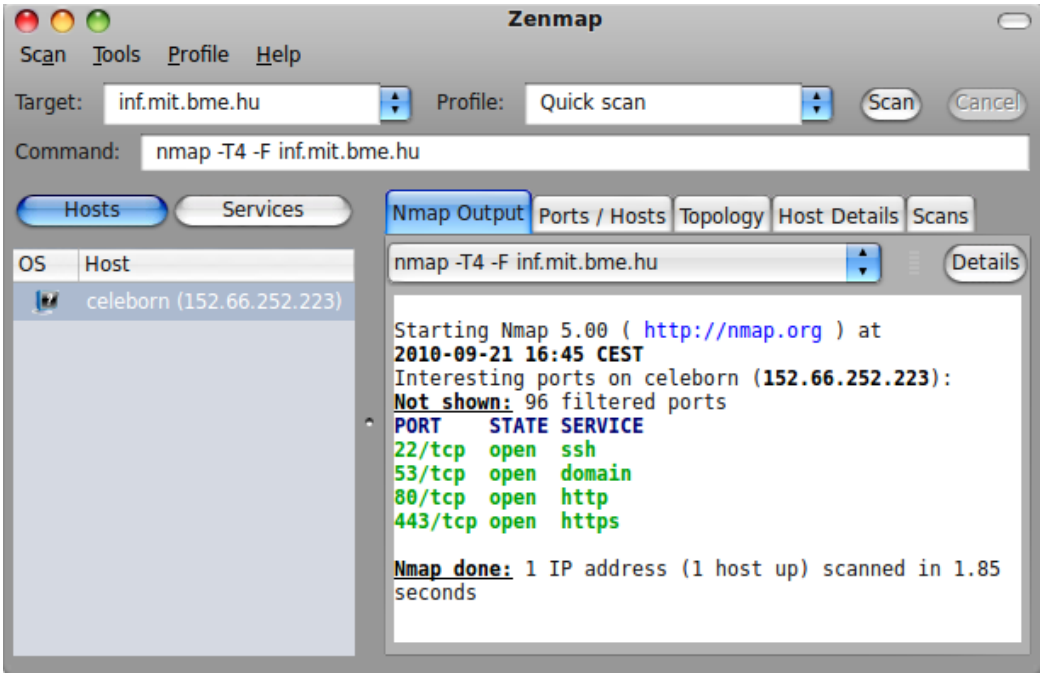
Az Nmap (Network Mapper) egy hálózatfelderítésre és biztonsági ellenőrzésre használható, nyílt forráskódú eszköz. Nagy hálózatok gyors feltérképezésére tervezték, ennek ellenére jól használható egyetlen számítógép ellenőrzésére is. Az Nmap a nyers IP csomagokat használja annak kiderítésére, hogy mely számítógépek érhetők el a hálózaton, ezek a számítógépek milyen szolgáltatásokat (alkalmazás neve és változata) kínálnak fel, milyen operációs rendszert futtatnak (és annak melyik változatát), milyen csomagszűrőt/tűzfalat használnak, illetve milyen egyéb jellemzői vannak.

Az Nmap tucatnyi különböző technikát használ az egyes felderített nyitott portok mögötti szolgáltatások meghatározására, a cél hoszton futó operációs rendszer azonosítására és a tűzfalak felderítésére. Ezen technológiák részletes

ismertetése nem a mérés célja, de néhány irodalom megismerésére szükség van, hogy értsük a mögöttes technikákat. Az nmap manuálja (`man nmap`) és a hivatalos honlap [5] elegendő információt ad, hogy megértsük a *különböző letapogatási módokat* (-sP, -sS, -sA, -PS, -PA), illetve az *operációs rendszer felderítés* működését [6][1].

**Nmap** - Hálózat feltérképező és port letapogató (portscan) eszköz.

A Zenmap az Nmap-hez ad grafikus felületet, futás közbeni állapota az 1. ábrán látható. Leolvasható, hogy az `inf.mit.bme.hu` címen elérhető szervert vizsgáltuk meg a „Quick scan" módszerrel, mely a gyakran használt szolgáltatások portjait térképezi fel. Látható, hogy a kiszolgáló SSH, DNS és webszolgáltatást végez.




1. ábra. Zenmap, az Nmap eszköz grafikus felülete

**Idegen gépek szkennelése éles infrastruktúrán nem megengedett, hálózat elleni támadásnak minősül, kitiltást vonhat maga után.**
**Így mérés közben a virtuális laboron kívüli egyetemi infrastruktúra, illetve egyetemen kívül lévő gépek ilyen módon való felderítése tilos.**

[1] A legnagyobb segítséget a módszer megértéséhez a Fingerprinting Methodology fejezet adja.

## 4.2. Hálózati szolgáltatások

Ebben a fejezetben először a gépen futó hálózati szolgáltatások feltérképezését mutatjuk be, majd a szöveges protokollt használó SMTP (e-mail küldő) szolgáltatást vizsgáljuk meg.

Amennyiben egy hoszton a hálózat felé szolgáltatást nyújtó alkalmazást futtatunk, akkor az a szolgáltatás a hoszt valamelyik hálózati interfészén, annak valamelyik portján várakozik bejövő kérésekre. Ezeket a netstat parancs listázza ki Linux környezetben.

**netstat** - A számítógépen futó szolgáltatások által foglalt hálózati portok és aktív kapcsolatok listázása.
Fontosabb opciók:
-a összes (hallgatózó és nem hallgatózó) socket kilistázása
-t tcp socketek listázása
-u udp socketek listázása
-p azon program nevének és azonosítójának megjelenítése, amihez a socket tartozik

A kliens alkalmazások a szolgáltatásokat a hozzáférési pontjukon érik el. Ez hálózati alkalmazás esetén tipikusan egy meghatározott cím (IP cím vagy domain név) és port. A `nentstat -atp` parancs kiírja, hogy a helyi gépen mely alkalmazások, mely portokon használnak TCP socketet. Amennyiben a 25-ös TCP porton hallgatózik egy program, akkor az valószínűleg egy SMTP szerver. A gyakorlatban használt legtöbb port és szolgáltatás összerendelését az IANA weboldalán lehet megtalálni [7]. Ha ez a 25-ös porton hallgatózó program nem csak a localhost-on, hanem hálózat felőli interfészen (is) hallgatózik, akkor azt távoli gépről is igénybe vehetjük.

A továbbiakban feltételezzük, hogy egy 25-ös porton hallgatózó SMTP szervert találtunk, amit ki szeretnénk próbálni. Egy SMTP szerver feladata, hogy a konfigurációban megadott hálózati interfészen és porton várakozzon, és bejövő kapcsolat esetén a protokollban meghatározott párbeszéd során a kliens által küldendő e-mail üzenetet átvegye, majd a megfelelő postafiók részére kézbesítse. Az SMTP szolgáltatás rendkívül összetett, gondoljunk csak az hitelesítés folyamatára vagy arra, hogy a címzett helyi vagy távoli postafiókkal is rendelkezhet. A példa során a lehető legegyszerűbb, hitelesítés nélküli, csak helyi postafiókok részére kézbesítő szerverrel foglalkozunk.

Az SMTP (Simple Mail Transfer Protocol) protokoll e-mail (és spam) továbbításra, küldésére létrehozott szöveges protokoll. Kapcsolódás után az SMTP szabványban rögzített párbeszédet folytatva tudják a kliensek a szolgáltatást igénybe venni. SMTP kiszolgáló esetén a kliens lehet bármilyen

levelezőprogram (pl. Mozilla Thunderbird vagy Microsoft Outlook), de egyszerű szöveges protokoll lévén akár a *telnet* is használható.

| **telnet** - felhasználói program a TELNET protokollhoz |
|---|

A *telnet* nem más, mint egy interaktív, két irányú, szöveges kommunikációt lehetővé tevő program és protokoll. A programmal bármely hoszt bármely portjához csatlakozhatunk, amivel így kézzel eljátszhatjuk a protokollt. Régebben a telnet programmal leginkább a 23-as porton hallgatózó telnet daemonhoz csatlakoztak, amivel sikeres autentikáció után távoli parancssoros hozzáférést lehetett szerezni. Ezt azonban a titkosítás hiánya miatt felváltotta a (tipikusan 22-es porton működő) ssh.

Ez viszont nem jelenti azt, hogy más szöveges protokollokat (HTTP, SMTP) ne lehetne az eszköz segítségével végigjátszani. SMTP esetén az alábbi kis példa illusztrálja egy e-mail üzenet elküldését. Figyeljük meg a szerver (S:) és kliens (K:) által küldött üzeneteket, és képzeljük el, hogy ugyanezt a párbeszédet folytatja a levelező program is az SMTP szerverrel. Érdemes megfigyelni még azt is, hogy a szerver a válaszaiban a protokollban meghatározott „return code" értékekkel is kommunikál. Ilyen kód például a „250", mely SMTP esetén a helyes paraméter átadást jelenti [8], vagy a „404" HTTP esetén, amit a webszerver akkor kommunikál, ha nem találja a kért erőforrást [9]. (Az „S:" és „K:" természetesen csak illusztráció, nem részei a protokollnak.)

```
$ telnet mail.konyv.hu 25
S: 220 mail.konyv.hu ESMTP Exim 4.69 Wed, 15 Sep 2010
10:39:14 +0200
K: HELO kliens.vagyok.hu
S: 250 Hello kliens.vagyok.hu, I am glad to meet you
K: MAIL FROM:<mereshallgato@example.org>
S: 250 Ok
K: RCPT TO:<root@mail.konyv.hu>
S: 250 Ok
K: DATA
S: 354 End data with <CR><LF>.<CR><LF>
K: From: "XY" <mereshallgato@example.org>
K: To: "Root user" <root@mail.konyv.hu>
K: Date: Wed, 15 Sep 2010 16:02:43 +0100
K: Subject: Teszt üzenet
K:
K: Hello root,
K:
K: tesztelem a telnetet, írj, ha megkaptad!
K: Én vagyok a méréshallgató
K: .
S: 250 Ok: queued as 12345
K: QUIT
S: 221 Bye
```

## 4.3. Webkiszolgálás

A webszolgáltatás felhasználói szemmel a böngészőbe írt URL-nek megfelelő weboldal megjelenítése. Ez az egyszerűnek tűnő művelet a gyakorlatban többlépéses folyamat eredményeképpen hajtódik végre:

1. A böngésző a kért URL alapján DNS lekérdezést hajt végre, aminek segítségével kideríti a kiszolgáló szerver IP címét.

2. A böngésző a kapott IP címre egy HTTP kérést küld a meghatározott portra (alapértelmezetten ez a 80-as port). A kérésben szerepel a GET parancs, a lekért oldal URL-je, a tartalom elfogadott kódolása, illetve egyéb információk, mint a kliens böngésző típusa, preferált nyelv, és további HTTP fejlécek [9]. Az alábbiakban egy egyszerű HTTP kérést mutatunk be:

```
GET / HTTP/1.1
Host: mitnick.konyv.hu
Accept-Language: Hu
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows)
```

3. Ha a kiszolgáló szerveren a megfelelő porton és interfészen hallgat a webszerver alkalmazás, és a tűzfal sem tiltja a csomag beérkezését, akkor megkapja a HTTP kérést. A mérés során az egyik legelterjedtebb webszerverrel, az Apache2-vel foglalkozunk.

4. Egy webszerver alkalmas több különböző webcímen elérhető oldal kiszolgálására is. Ennek a technikának a neve *virtualhost*ing. A HTTP kérés `Host` mezője, a bejövő TCP kapcsolat interfészének IP címe és port száma alapján a webszerver eldönti, hogy melyik virtualhost tartalmát szolgálja ki. Az egyes virtualhost-ok tartalma a számukra létrehozott mappában található (tipikusan /var/www alatt).

   Az Apache webszerver a virtualhost konfigurációit a /etc/apache2/sites-available mappában tárolja, azonban csak azokat üzemelteti aktívan (szolgálja ki), melyekre link mutat a /etc/apache2/sites-enabled mappából. A virtualhostok ilyen módon való engedélyezésére az `a2ensite`, tiltására az `a2dissite` parancs használható.

5. Végezetül a webszerver a kérésnek megfelelő tartalmat összeállítja, és a HTTP válaszcsomagban eljuttatja a böngésző számára. A kért tartalom jellegét tekintve két típusú lehet:

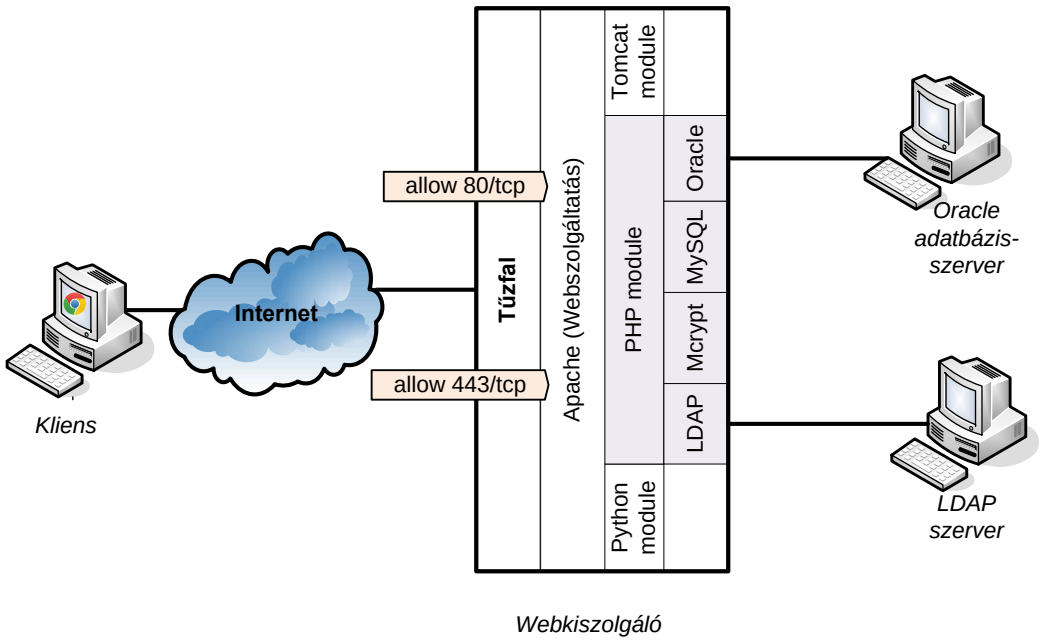
   - Statikus tartalomtípus esetén legtöbbször egy fájl lemezről történő felolvasása és annak elküldése jelenti a kiszolgálást.
   - Dinamikus tartalomtípus esetén egy külső program, egy Apache beépülő modul vagy egy szkript lefuttatása után előálló tartalom kerül elküldésre.

   Az előbbihez hasonló HTTP kérdésre adott válasz eleje a következőképpen nézhet ki:

```
HTTP/1.1 200 OK
Date: Mon, 12 Sep 2010 19:12:16 GMT
Server: Apache/2.0.0 (Unix) Debian/GNU mod_perl/1.24
Last-Modified: Fri, 01 Sep 2010 14:16:18
Content-Length: 3369
Content-Type: text/html

<html><head>.........
```

A tartalom dinamikusan áll elő PHP alapú weblapoknál is. Ekkor a 2. ábrát követve, a kliens elküldi a kérést a webszervernek, ami továbbítja azt a PHP értelmezőnek. A PHP értelmező lefuttatja a programkódot (PHP szkriptet), ami (mint bármely CGI program) futása eredményeként előállítja a HTML formátumú eredményt. Ezt a webszervernek visszaadja, ami az eredményt már csak továbbítja a felhasználónak.




2. ábra. Apache webszerver felépítése PHP programok futtatásakor

A PHP értelmező is egy modulárisan bővíthető rendszer, melyhez a különböző kiterjesztések (extension) a konfigurációs állományokban (php.ini, conf.d alatti .ini fájlok) engedélyezhetőek. Felesleges például minden kérés során az Oracle adatbázis illesztő modult betölteni, ha soha sem készítünk olyan PHP szkriptet, ami ezt felhasználja. Az egyes modulok engedélyezésével bővül a PHP eszközkészlet, és a modulban definiált függvényhívások is

használhatóak. Így ha az MCrypt modul be van töltve, akkor például egy regisztráció során annak függvényeivel titkosítható a jelszó. Ha címtár lekérdezésére van szükség a programból, akkor az LDAP kiterjesztést lehet használni, ami belül valóban felveszi a kapcsolatot egy távoli LDAP .

A HTTP protokoll biztonságosabb használatát teszi lehetővé a HTTPS, mely a közkeletű tévedéssel ellentétben nem alkot önálló protokollt, csupán egy titkosított csatornát épít ki, ami fölött a hagyományos HTTP kommunikáció zajlik. Abban az esetben, amikor egy weboldalt HTTPS kapcsolaton keresztül kérünk le, a fent leírt webkiszolgálás folyamat eleje a következőképpen módosul:

1. A böngésző a kért URL alapján a hoszt IP címét lekérdezi a DNS adatbázisból.

2. A kapott IP cím által jelzett HTTPS kiszolgáló meghatározott portjához csatlakozik (ami alapértelmezetten a 443-as port), és a kiszolgálóval az SSL szabványban meghatározott protokoll szerint felépíti a titkosított adatcsatornát.

   Az SSL alapú kapcsolatnak alapvetően két célja van:

   - titkosított adatcsatorna biztosítása a kommunikáló felek között,
   - a szerver hitelesítése annak tanúsítványa alapján. Ennek céljából a kapcsolat kiépítése során a kiszolgáló bemutatja a tanúsítványát, melynek segítségével hitelesíti magát a kliens (böngésző) számára. Amennyiben a hitelesítés nem kielégítő (pl. self signed certificate esetén), akkor a böngésző a felhasználóhoz fordul a jól ismert „nem megfelelő tanúsítvány” kérdéssel.

     A tanúsítvány elfogadásához a böngésző többek között a következőket ellenőrzi:

     – a tanúsítvány érvényességi ideje
     – megfelelő hosztnévre van-e kiállítva
     – megfelelő hitelesítő szervezet írta-e alá

3. A böngésző a titkosított csatornán egy HTTP kérést küld, melyben szerepel a lekért oldal URL-je, ...

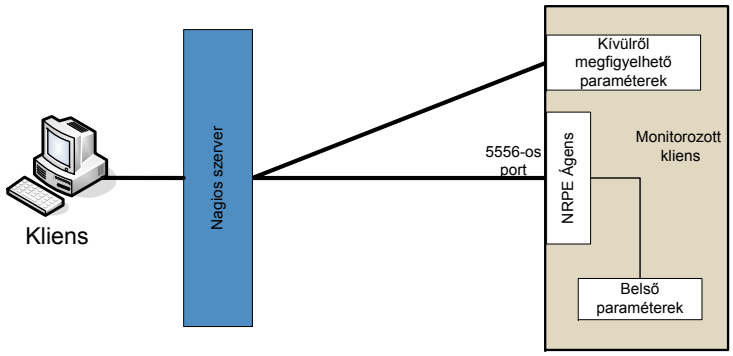
   Innen kezdve pedig minden megegyezik a webkiszolgálás menetében leírtakkal azt hozzátéve, hogy minden kommunikáció egy titkosított csatornán keresztül zajlik.

**Wireshark** - Hálózati forgalom elemző eszköz. Rögzíti a hálózati forgalmat, és megjeleníti azon részeit, melyeket a *Capture Filter* nem szűrt ki. Részletes protokoll információk nyerhetőek ki, illetve a *Follow TCP Stream* egy TCP kommunikációt is össze tud illeszteni, és szövegesen megjeleníteni. Részletes információ a Mérés labor 4 [10] keretében használt dokumentációban található.

A mérés során webalkalmazásokat fogunk használni, melyek háttéradatbázisban tárolják az adataikat. Ehhez MySQL adatbáziskezelő rendszert alkalmazunk, mely egy kliens-szerver architektúrájú adatbáziskezelő rendszer.

## 4.4. Rendszermonitorozás

A mérés során rendszermonitorozásra az előző félévben bemutatott Nagios [11] rendszert használjuk. A Nagios monitorozó rendszer központi adatgyűjtő komponense rendszeresen lekérdezi a megadott hosztok és szolgáltatások állapotát, azok paramétereit, és megjeleníti azokat egy webes felületen. A Nagios rendszer vázlatos felépítését a 3. ábra szemlélteti.




3. ábra. A Nagios monitoring architektúra felépítése

Távoli hosztok monitorozása esetén az ágens alapú monitorozást használja. Ennek során az NRPE (Nagios Remote Plugin Executor) ágens telepítve van a megfigyelt hosztra, és adott porton (alapértelmezetten az 5666-os TCP porton) várakozik a központi monitoring rendszer kéréseire. A kérések beérkezése esetén lefuttatja a kéréshez definiált parancsot, és az eredményt visszaküldi a központi rendszernek.

A *Nagios szerver* konfigurációs állományai a következő fogalmakkal dolgoznak:

- Hosts: definiálja a monitorozandó hosztokat

- Hostgroup: az monitorozandó hosztokat csoportokba sorolja
- Service: szolgáltatásokat definiál adott hostgroup-hoz vagy hoszthoz, ami azt írja le, hogy az adott gépeken mely monitorozó parancsokat *(command)* kell lefuttatni

A monitorozott hoszton az *NRPE ágens* beállításait kell karbantartani, melyek közül kiemelendőek a következők:

- **Allowed hosts**: azon hosztok listája, akiktől monitoring kérdéseket fogad. Amennyiben ez a paraméter nem tartalmazza a központi monitoring szerver címét, úgy az NRPE ágens nem fogja kiszolgálni azt.
- **Command**: parancs definíciók, melyek megadják, hogy egy adott beérkező monitoring kérdésre milyen parancsot kell végrehajtani.

A Nagios beállítások és azok szintaktikája nem egyszerű, de a beállítások logikájának megértése után meglévő konfiguráció módosítása (például újabb hosztok, szolgáltatások felvétele) már nem jelenthet akadályt.

Egy újabb monitorozási feladat definiálásához (főleg NRPE ágens számára) saját *command-ot*, vagy más néven *plugint* kell létrehozni. Ezek fejlesztése is könnyen kivitelezhető, hiszen a Nagios pluginek egyszerű futtatható programok vagy szkriptek, melyek *kimenetének formátuma rögzített*, ezáltal a központi Nagios szerver azokat egységesen fel tudja dolgozni és meg tudja jeleníteni a webes felületen.

A plugin tehát működése során elvégzi a szükséges vizsgálatokat, végül pedig *legalább egy sort* ír a standard kimenetre, és annak megfelelő visszatérési értékkel lép ki.

Az egysoros kimenet formátuma általában a következő:

```
SERVICE STATUS: Information text
```

- „SERVICE”: a monitorozott szolgáltatás neve
- „STATUS”: a vizsgálat eredménye, mely lehet OK, WARNING, CRITICAL és UNKNOWN
- „Information text”: bármilyen személyes információ, ami megjelenik a szolgáltatás mellett a webes felületen

Például:

```
PING OK - Packet loss = 0%, RTA = 0.33 ms
```

A plugin visszatérési értéke pedig a státusznak megfelelően a következőképpen alakul:

- 0: OK
- 1: WARNING
- 2: CRITICAL
- 3: UNKNOWN

# 5. Linux alapismeretek

A mérés során a szolgáltatásokat Linux alapú kiszolgálók üzemeltetik. Az alapvető Linux ismereteken túl a következőkben foglaljuk össze a méréshez szükséges információkat.

- Bármilyen Linux parancssori utasításról, annak paraméterezéséről a `man utasítás` vagy a Google ad bővebb felvilágosítást.
- A kiszolgálókra SSH kapcsolaton keresztül érdemes bejelentkezni, melyhez Windows rendszeren a Putty, Linuxon pedig az `ssh` program használható.
- Fájlok felmásolásához javasolt az SCP használata, melyhez Windowson a WinSCP alkalmazás ajánlott, Linuxon pedig van parancssori `scp` kliens. A fájlok felmásolása után mindig figyeljünk a megfelelő jogosultságokra, hogy például a megfelelő szolgáltatás felhasználója is tudja olvasni, írni a megfelelő állományt.
- Fájlok böngészésére javasolt a Midnight Commander (`mc`) használata, beépített szerkesztőjével pedig könnyen módosíthatóak a beállításokat tartalmazó fájlok.
- Távoli fájlok letöltése ftp vagy http protokollon keresztül a `wget` parancs segítségével történhet.
- Konfiguráció módosítás után a szolgáltatások újraindítandóak, hogy a beállítások érvényre jussanak. Ehhez a `service` kulcsszót lehet használni, aminek paramétere legtöbbször a „start”, „stop”, „restart” vagy „reload”, értelemszerű jelentéssel.

  Például az Apache2 webszerver a `service apache2 restart` utasítással indítható újra.

- A különböző szolgáltatások beállításait tartalmazó fájlok jellemzően a `/etc` könyvtárban helyezkednek el. (A hivatalos dokumentációktól eltérően a /usr/local/etc alatti konfigurációs fájlokat lehet, hogy Ubuntuban a /etc alatt kell keresni.)

- Rendszer szintű beállítások módosításához, illetve szolgáltatások (újra)indításához csak a root felhasználónak van joga. Ehhez `sudo bash` paranccsal szerezhetünk rendszergazda parancssort.

# 6. Esettanulmány

Az elméleti bevezető jobb megértése érdekében egy egyszerű esettanulmány leírása következik. Az esettanulmány során a mérés feladataihoz hasonló problémák fordulnak elő, és a mérésen használatos eszközöket mutatjuk be.

Az esettanulmányban egy, a kimenő levelezés működése során felfedezett hiba felderítése és kijavítása kerül bemutatásra.

> Munkaállomásunkon észleljük, hogy nem tudjuk elküldeni az éppen megírt e-mailt, mert a távoli levelező kiszolgáló (mail.konyv.hu) időtúllépés miatt nem válaszolt. A levélkiszolgáló alkalmazásnak mi vagyunk az üzemeltetői, így a hibát saját kezűleg tudjuk kijavítani. Mit tudunk tenni?

A probléma felderítés során szisztematikus diagnosztikai módszereket alkalmazunk, melyek segítségével közeledünk a probléma forrásához.

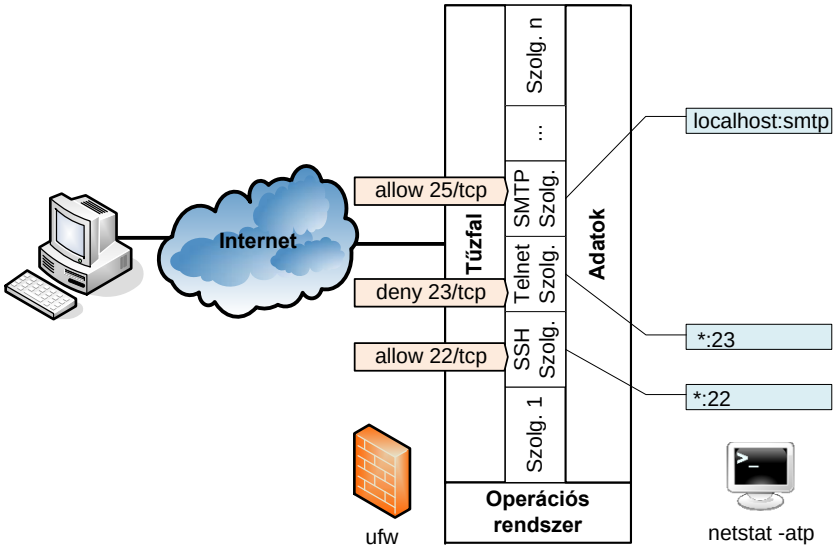
- Ábrát készítünk, melyen feltüntetjük a rendszernek a probléma szempontjából releváns komponenseit. Esetünkben az ábrán megjelenítjük, hogyan jut el a kérés a kiszolgálóhoz. Ha a struktúra egyszerű, ezt fejben is megtehetjük.

- Definiáljuk az egyes komponensekhez tartozó hibamódokat, bonyolultabb esetben hibafát rajzolunk.

- Végiglépkedünk az ábrán (a klienstől a szerverig vagy fordítva), és ellenőrizzük, hogy az adott komponens hibája fennáll-e.

Ezt az általános elvet követve, konkrét példa keretében mutatjuk be a diagnosztika lépéseit. A 4. ábrán a rendszernek a probléma szempontjából releváns komponenseit jelenítettük meg.

A 4. ábrán az SMTP kliens gépet az internet felhőtől balra lévő számítógép szimbolizálja, a szerver oldalt pedig az attól jobbra elhelyezkedő rendszer.




4. ábra. Az SMTP szolgáltatás elérése

Azt, hogy a kliens miért nem tudja az SMTP kiszolgálót igénybe véve elküldeni a levelet, a következő lépéssorozattal deríthetjük ki, illetve javíthatjuk:

- Nulladik lépésként ellenőrizzük számítógépünk internetkapcsolatát pl. más, független weboldalak betöltésével. Felesleges a levelezésben keresni a hibát, ha az internetkapcsolatunk nem működik.

- Első lépésben megvizsgáljuk, hogy a levelező kiszolgáló elérhető-e a saját gépünkről.

```
meres@desktop:$ ping mail.konyv.hu
PING mail.konyv.hu (1.2.3.4) 56(84) bytes of data.
64 bytes from mail.konyv.hu (1.2.3.4): icmp_seq=1 ttl=57
time=1.06 ms
64 bytes from mail.konyv.hu (1.2.3.4): icmp_seq=2 ttl=57
time=1.06 ms
64 bytes from mail.konyv.hu (1.2.3.4): icmp_seq=3 ttl=57
time=2.68 ms
-- mail.konyv.hu ping statistics --
3 packets transmitted, 3 received, 0% packet loss, time
2002ms
rtt min/avg/max/mdev = 1.060/1.600/2.680/0.764 ms
```

- Ellenőrizzük, hogy elérhető-e a szerver, és ha szükséges, akkor kinyitjuk a tűzfalon a 25-ös, levélküldésre használt portot. Az ábra szerint

ez a port nyitva van, de például a titkosítás nélküli távoli parancssor eléréséhez (telnet) külön ki kellene nyitni a 23-as portot (vagy törölni a tiltó szabályt). A tűzfal állapotának lekérdezését, illetve manipulálását legegyszerűbb az `ufw` paranccsal megtenni.

- Ellenőrizzük, hogy várja-e a szerver a kapcsolatokat a 25-ös porton. Ezt úgy lehet megtenni, hogy belépünk a szerverre, és a következő utasítással kilistázzuk az éppen aktív szerveralkalmazásokat és kapcsolatokat:

```
root@mail.konyv.hu:# netstat -atp
```

  A kimeneten láthatjuk a sort, mely jelzi, hogy az Exim4 szerver alkalmazás az SMTP porton hallgat a localhost interfészen (`localhost:smtp`). Ez azt jelenti, hogy a levelező szerver csak a helyi interfészen figyeli a bejövő kéréseket, az internet felé néző interfészen nem. Azaz nem fogja fogadni az internetről jövő kéréseket, még nyitott tűzfalnál sem. Azt, hogy egy szolgáltatás melyik interfészen hallgatózzon, a szolgáltatás konfigurációs fájlaiban kell beállítani.

  Azt érdemes megjegyezni, hogy van, amikor a szolgáltatás elérését a kliens gépről próbáljuk tesztelni magával a szolgáltatás igénybevételével, vagy portjainak letapogatásával. Ekkor viszont sokszor csak együtt lehet tesztelni, hogy az adott port nyitva van-e, és hogy a publikus interfészen hallgatózik-e az alkalmazás.

- A levelezőszerver beállításait módosítva beállíthatjuk, hogy a publikus külső IP címen is hallgasson, így lehetővé téve a távolról történő csatlakozást. A szerveren Exim4 levelezőszerver fut, ennek konfigurációjában kell beállítani azt, hogy melyik interfészen hallgasson. Módosítsuk a `/etc/exim4/update-exim4.conf.conf` fájlt úgy, hogy a `dc_local_interfaces` paraméter üres string legyen. Ezzel a beállítással adhatjuk meg, hogy mely interfészeken hallgasson a levelezőszerver, üresen hagyva pedig alapértelmezetten minden létező interfészt használni fog. Az exim újraindítása után a beállítás érvényre jut, és `netstat` segítségével ellenőrizhetjük, hogy most már minden interfészen hallgat a szerver (`*:smtp`). A levélküldést kipróbálva azt tapasztaljuk, hogy a rendszer működik.

Ezen esettanulmányon keresztül láttuk, hogy amennyiben ismerjük a hálózatot és a rendszerünk komponenseit, akkor szisztematikus diagnosztikai módszerekkel képesek voltunk elhárítani a hibákat. A mérés célja, hogy ehhez hasonló szituációkban önálló problémamegoldás keretében alkalmasak legyünk a hibákat felderíteni és elhárítani.

Fontos, hogy a komplex problémát visszavezetjük egyes konkrét komponensek és technológiák meghibásodására, így elegendő a hibakeresést azokon belül elvégezni. Természetesen nem elvárás a bemutatott Exim4 levelezőszerver konfigurációjának ismerete, de internetes források (Google) segítségével konfigurációs módosításokat meg kell tudni tenni.

# Hivatkozások


[1] Andrew S. Tanenbaum. *Számítógép-hálózatok*. PANEM, 2006.

[2] Root névszerverek. `http://en.wikipedia.org/wiki/Root_nameserver`.

[3] UNIX/Linux kiszolgálók üzemeltetése választható tantárgy. `http://unixlinux.tmit.bme.hu`.

[4] Netfilter weboldal. `http://www.netfilter.org/`.

[5] NMap eszköz. `http://nmap.org/man/hu/`.

[6] Fyodor. Remote OS detection via TCP/IP Stack FingerPrinting. `http://nmap.org/nmap-fingerprinting-article.txt`, 1998.

[7] IANA Portnumbers. `http://www.iana.org/assignments/port-numbers`.

[8] SMTP Specifikáció. `http://tools.ietf.org/html/rfc2821`.

[9] HTTP Specifikáció. `http://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html`.

[10] Wireshark protokollanalizátor használata. `http://www.mit.bme.hu/oktatas/targyak/vimia315/jegyzet/ml4_1_wireshark.pdf`.

[11] Nagios weboldal. `http://www.nagios.org/`.