Informatikai technológiák laboratórium 2.
(BMEVIMIA429)

Komplex alkalmazási környezetek felderítése és menedzsmentje
(Mérési segédlet)

Szatmári Zoltán
Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

2010. szeptember 20.

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, ezen rendszerek tervezése, megismerése és üzemeltetése igényli a korábbi félévekben megtanult ismeretek együttes gyakorlati alkalmazását. Az összetett rendszerek üzemeltetése közben gyakorta szembesülünk általunk nem ismert rendszerekkel, melyeknek a megismerése, és felderítése komoly feladatot jelent.

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 mérés során egy kezdetben ismeretlen felépítésű web- és adatbázisszolgáltatás nyújtó infrastruktúrával és annak monitorozásával kapcsolatos feladatokat tűzünk ki, melyek során a megismert diagnosztikai eszköztárat önállóan kell alkalmazni.

2. Szükséges előismeretek

A mérés során a következő tárgyak során elsajátított előismeretekre építünk, melyek gyakorlati alkalmazása szükséges a mérés elvégzéséhez. **Az alábbiakban zárójelben megnevezett konkrét technológiák ismeretét a beugróban is kérhetjük.**

- 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, PHP alapismeretek

3. Hálózatok menedzsmentje

Hálózat- és szolgáltatásmenedzsment tevékenységek során gyakran találkozunk számunkra ismeretlen infrastruktúrával, hálózati topológiával. Ilyen esetekben a menedzsment feladatok ellátásához szükséges információt a hálózat felderítése, a szolgáltatások megismerése során tudjuk megszerezni. Ebben a fejezetben a méréshez szükséges alapvető hálózatmenedzsment ismereteket mutatjuk be.

3.1. Hálózati alapismeretek

Amennyiben nyílt hálózatról van szó, akkor a hálózatra való kapcsolódás során vagy előre beállított IP paraméterekkel, vagy DHCP segítségével kiosztott IP paraméterekkel csatlakozunk a hálózatra. Ilyen IP paraméter az *IP cím*, az *alhálózati maszk*, az *alapértelmezett átjáró* és a *névszerver(ek)*.

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

ifconfig - Hálózati interfészek információinak és megjelenítése és módosítása

arp - ARP gyorsítótár megtekintése 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 teremt kapcsolatot az IP címek és domain nevek között. A szolgáltatást leggyakrabban a domain név feloldásra szokás használni, mely során egy megadott domain névhez keressük az őt kiszolgáló hálózati

hoszt IP címét. Kevésbé ismert, hogy a szolgáltatás fordítva is működőképes. A reverse DNS (rDNS) során adott IP címhez is visszakereshető egy olyan domain név, amit az azon a címen működő hoszt kiszolgál. A DNS segítségével tehát megoldódik, hogy ne kelljen IP címeket fejben tartani, hanem elegendő könnyebben megjegyezhető domain neveket megjegyezni.

A DNS strukturáját 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 nevek, melyek a különböző országkódok és általános célú végződések (.com, .org, .nasa, .gov, .net, .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 feloldás* 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. A feloldás folyamatát a www.mit.bme.hu domain példáján keresztül mutatjuk be:

- A kliens a ROOT névszerverektől megkérdezi, hogy a www.mit.bme.hu domain milyen IP-re oldható fel. Az egyik ROOT szerver válaszol, hogy ezt a domaint ő nem tudja feloldani, de a .hu végződésű domaineket a ns2.nic.hu, ns1.nic.hu, ns3.nic.hu stb. DNS szerverektől kell kérdezni, mert ők hivatottak a .hu végződésű domain nevek feloldására. A ROOT szerverek címei közismertek, nagyon ritkán változnak. [2]

- A kliens ezután az egyik .hu DNS szervertől megkérdezi, hogy a www.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 vagy az ns.bme.hu 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 egy alsó szintű DNS szerverhez, amely már képes feloldani a domain nevet, és visszaadja 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. Nem minden lekérdezés folyamat

fejeződik be DNS feloldással, hiszen ha a lekérdezési folyamat során bármelyik lépés eredményre vezet, akkor ott befejeződik. A folyamat általános váza a következőkben látható. Speciális esetekben adott szinten további lépések, további szerverek találhatók, de a séma minden esetben az alábbiakkal egyezik meg:

- A kliens megkérdezi az operációs rendszertől, hogy fel tudja-e a domain nevet oldani, ezáltal delegálja a feladatot számára.
- Az operációs rendszer ellenőrzi a helyi `hosts` fájlban, hogy statikusan megadásra került-e a domain - IP párosítás. A `hosts` állományban minden operációs rendszeren előírhatunk fix domain név - IP cím kötéseket, ezáltal letitva a DNS lekérdezés folyamatát. Linux rendszerekben a fájl a `/etc`, míg Windows alatt a `%WINDIR%\System32\Drivers\etc` mappában található.
- Az operációs rendszer megvizsgálja, hogy a helyi gyorsítótárban tárolódik-e a megfelelő érték.
- Az operációs rendszer delegálja a kérdést a hálózati beállítások során megadott valamely névszerver felé.
- A névszerver megvizsgálja a helyi gyorsítótárát.
- A névszerver elindítja a DNS feloldás folyamatát, mert egyik gyorsítótárban sem található meg a kért bejegyzés.

nslookup - DNS feloldás kérése opcionálisan megadva a feloldásra megkérendő szerver címét

3.3. Tűzfalbeállítások

A tűzfal általánosságban egy olyan hálózati eszköz vagy alkalmazás, 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 **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 elvileg akárhány kapcsolatot tud kezelni. Hátránya, hogy buta és rugalmatlan: pl.

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 akár DoS-támadás indítható ellene.

A mérési környezetben a Debian (Linux) operációs rendszeren általánosan használt Netfilter/IPTables [4] tűzfalat használjuk. Alapvető működésének megértéséhez a következő fogalmak ismerete szükséges:

- **Illesztés (match)**: egy feltétel, aminek a vizsgált csomag meg kell felelnie. Ilyen feltételt írhatunk fel többek között a forrás és cél IP cím, a protokoll, a forrás- vagy célport, a kapcsolat állapota és a hálózati interfész vonatkozásában.
- **Akció (target)**: a döntés arról, mi történjen a csomaggal. Ez többek között lehet engedélyezés, eldobás, visszautasítás vagy további vizsgálat előírása.
- **Szabály (rule)**: ÉS kapcsolatban lévő illeszkedési szabályok és egy akció együttese.
- További fogalmak a lánc (chain), tábla (table), melyek ismerete túlmutat a mérés célkitűzésein, így itt nem kerülnek részletezésre.

A könnyebb megértést elősegítendő nézzük a következő példát! Egy szerveren céges postafiókokat üzemeltetünk, melyeket POP3 protokoll segítségével kérhetnek le a tulajdonosaik. A céges szabályok értelmében ezt otthonról nem, csupán cégen belülről érhetik el az 1.2.3.x IP címekről. Ebben az esetben a levelező szerver tűzfalában vázlatosan fogalmazva a következő szabályt kell felvenni:

```
forrasIP=1.2.3.0/24, protokoll=TCP, celport=110 - engedelyez
```

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 betöltéséről és a futás közbeni menedzseléséről. A következő parancssori utasításokat célszerű megismerni:

- `ufw`: egyszerű súgó a paraméterezésről
- `ufw status`: a jelenleg beállított szabályok kilistázása
- `ufw allow 110/tcp`: engedélyező szabály felvétele, amely a TCP 110-es porton engedélyezi a bejövő forgalmat
- `ufw delete allow 110/tcp`: az előző szabály törlése
- `ufw deny 110/tcp`: tiltó szabály felvétele, amely a TCP 110-as porton tiltja a bejövő forgalmat

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

4. Hálózati szolgáltatások menedzsmentje

4.1. Hálózati szolgáltatások

Ebben a fejezetben a bemutatott elméleti ismereteket az SMTP (e-mail küldés) szolgáltatás példáján keresztül vizsgáljuk meg a könnyebb érthetőség kedvéért.

Egy SMTP szerver feladata, hogy megadott hálózati interfészen 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 rendelkezhet. A példa során a lehető legegyszerűbb, hitelesítés nélküli, csak helyi postafiókok részére kézbesítő szerverrel foglalkozunk.

Amennyiben egy hoszton a hálózat felé nyújtott szolgáltatást futtatunk, akkor az a szolgáltatás a hoszt valamelyik hálózati interfészén, annak valamelyik portján várakozik bejövő kérésekre. Amennyiben az SMTP szerver

a hálózat hosztjaitól is fogad üzeneteket, akkor a hálózat felőli interfészen várakozik, jellemzően a 25-ös TCP porton. A gyakorlatban használt legtöbb szolgáltatás és port összerendelését az IANA weboldalán lehet megtalálni. [5]

netstat - A számítógépen futó szolgáltatások által foglalt hálózati portok és aktív kapcsolatok listázása

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. Kapcsolódás után a protokollban rögzített párbeszéd során tudják a szolgáltatást igénybe venni. SMTP kiszolgáló esetén a kliens lehet bármilyen levelezőprogram (pl. Mozilla Thunderbird vagy Microsoft Outlook), vagy egyszerű szöveges protokoll lévén a *telnet* alkalmazással is könnyen kezelhető.

telnet - Univerzális parancssoros hálózati kliens

A *telnet* nem más, mint egy univerzális hálózati kliens alkalmazás, melynek segítségével csatlakozhatunk valamely hoszt valamely portjához, és az ott várakozó szerverrel parancssoros interfészen eljátszhatjuk a protokollt.

Az SMTP (Simple Mail Transfer Protocol) protokoll e-mail (és spam) továbbításra (küldésére) létrehozott szabványos protokoll.

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 [6], vagy a „404” HTTP esetén, amit a webszerver akkor kommunikál, ha nem találja a kért erőforrást. [7] (Az „S:” és „K:” természetesen csak illusztráció, nem részei a protokollnak.)

```
root@desktop:# telnet smtp.itlab.hu 25
S: 220 smtp.itlab.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@itlab.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@itlab.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
```

Wireshark - Hálózati forgalom analizáló eszköz. Részletes információ a Mérés labor 4 [8] keretében használt dokumentációban található.

4.2. Hálózatfelderítés

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

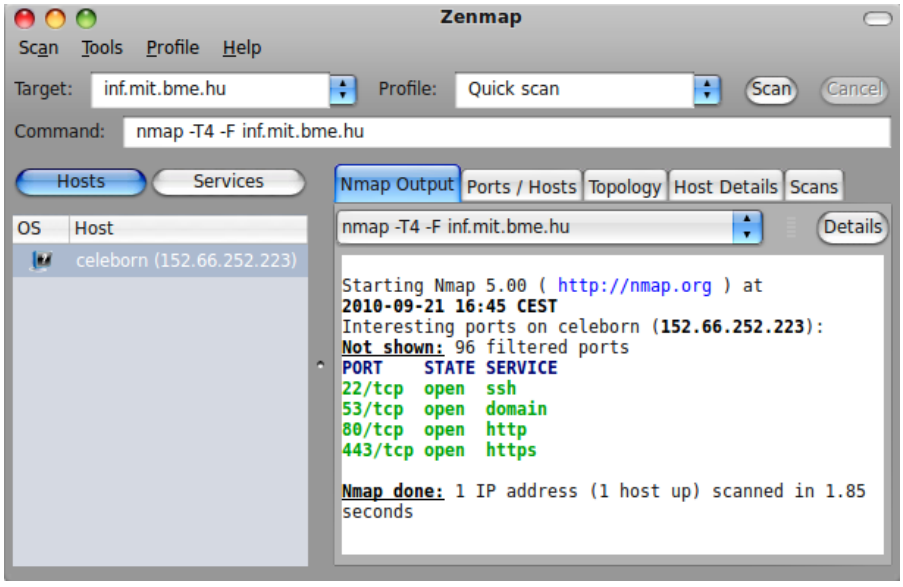
Egy hálózati hosztról kívülről vizsgálódva viszonylag kevés információt tudunk megállapítani: a hálózati kommunikációhoz használt címeit, így a *fizika címét (MAC)*, az *IP címét* és azokat a TCP és UDP *port*okat, melyeken hálózati szolgáltatásokat tud nyújtani. Az ilyen információkból különféle következtetéseket 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, valamint számtalan egyéb jellemzőre.

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 ismeretterjesztő célzattal érdemes az Nmap weboldalán utánanézni [9]. A mérés során csupán az alapvető hoszt és port felerítő technikákat alkalmazzuk.

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

Az alábbi ábrán az Nmap grafikus felületének egy futás közbeni állapotát mutatjuk be. Az ábrán látható, hogy az `inf.mit.bme.hu` címen elérhető szervert vizsgáltuk meg a „Quick scan” módszerrel, mely a közismert szolgáltatások portjait térképezi fel. Látható, hogy a kiszolgáló SSH, DNS és webszolgáltatást végez.



1. ábra. Az Nmap eszköz grafikus felülete

Idegen gépek scannelése éles infrastruktúrán nem megengedett, hálózat elleni támadásnak minősül, kitiltást vonhat maga után.

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 névfeloldás (DNS) 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 lekért oldal URL-je, a kliens böngésző típusa, karakterkódolási információk és további HTTP fejlécek [7]. Az alábbiakban egy egyszerű HTTP kérést és választ mutatunk be:

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

```
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>.........
```

3. A kiszolgáló szerveren a megfelelő porton hallgat a webszerver alkalmazás, és megkapja a HTTP kérést. A mérés során az egyik legelterjedtebb webszerverrel, az Apache2-vel foglalkozunk.

4. Egy webszerver alkalmas több különböző weboldal címen elérhető weboldal kiszolgálására is. Ezt a technikát nevezzük *virtualhost*nak. 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.

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 script lefuttatása után előálló tartalom kerül elküldésre.

   Ezen utóbbi kategóriába tartozik többek között a korábbi félévekben megismert PHP kiszolgálás is. Az lekért PHP scriptet a webszerver először átadja a PHP értelmezőnek, és a lefuttatott script eredményét (ami többnyire HTML tartalom) küldi el a kliensnek.

A PHP feldolgozás során a kért tartalmat a webszerver tehát átadja a PHP értelmezőnek, ami a PHP kódot lefuttatva a kimenetet visszaadja a webszervernek. A PHP értelmező is egy modulárisan bővíthető rendszer, melyhez a különböző modulok (extension) a konfigurációs állományban (php.ini) engedélyezhetőek. Felesleges például minden kérés során az Oracle adatbázis illesztő modult betölteni, ha soha sem készítünk olyan PHP scriptet, ami ezt felhasználja. Az egyes modulok engedélyezésével bővül a PHP eszközkészlet, és a modulban definiált függvényhívások is használhatóak.

A HTTP protokollnak egy biztonságosabb verziója a HTTPS, mely a közkeletű tévedéssel ellentétben nem alkot önálló protokollt, csupán egy titkosított csatorna fölötti HTTP kérésekről van szó. Abban az esetben, amikor egy weboldalt HTTPS kapcsolaton keresztül kérünk le, a fent leírt webkiszolgálás folyamat eleje picit módosul. Az első és második lépések között kialakításra kerül a titkosított csatorna, mely fölött a további kommunikáció zajlik.

1. A böngésző a kért URL alapján névfeloldás (DNS) segítségével kideríti a kiszolgáló szerver IP címét.

2. A kapott IP cím által jelzett kiszolgáló meghatározott portjához csatlakozik (alapértelmezetten ez a 443-as port), és a kiszolgálóval az SSL szabványban meghatározott protokoll szerint felépíti a titkosított adatcsatornát. A titkosított csatorna az IP rétegben épül fel, így a felsőbb alkalmazásszintű protokollokkal történő kommunikáció már titkosított csatornánk zajlik.

   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.

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

Az adatbáziskezelőhöz Windows rendszerről grafikus klienssel, Linux rendszerről parancssoros klienssel is tudunk csatlakozni.

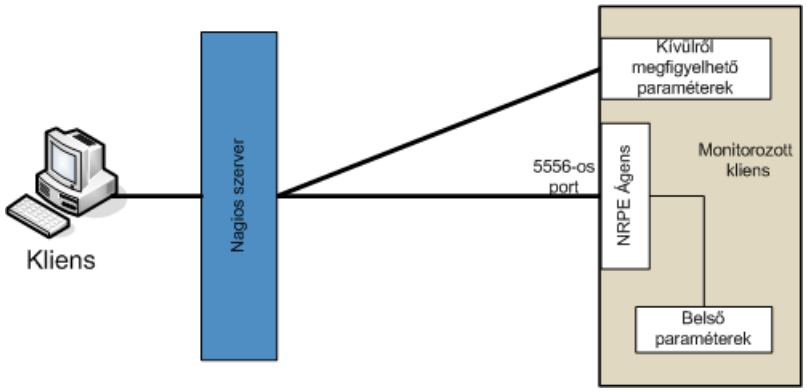
A webszerverek teljesítményét számos paraméter meghatározza. Ilyen paraméterek lehetnek például a párhuzamos kiszolgáló folyamatok száma, egy folyamaton belüli párhuzamos szálak száma vagy egy folyamat által kiszolgálható kérések maximális száma. Az külön tudománynak számít, hogy egy webalkalmazáshoz megfelelőn beállítsuk a webszervert. A teljesítmény teszteléséhez az ApacheBenchmark eszközt szokás használni. Részletesebb információ a különböző Benchmark technikákról a Rendszermodellezés tárgy keretében hangzik el [10].

ApacheBenchmark - Webszerver teljesítmény mérésére használható eszköz

4.4. Rendszermonitorozás

A mérés során rendszermonitorozásra az előző félévben bemutatott Nagios [11] rendszert használjuk. A Nagios monitorozó rendszer központi adat-

gyűjtő komponense rendszeresen lekérdezi a megfelelő hosztok és szolgáltatások állapotát, azok paramétereit, és megjeleníti azokat egy webes felületen. A Nagios rendszer vázlatos felépítését az alábbi ábra szemlélteti.



2. ábra. A Nagios monitoring architektúra felépítése

Távoli hosztok monitorozása esetén az ágens alapú monitorozást használja. Ennek során a Nagios NRPE (Nagios Remote Plugin Executor) ágens telepítve van a megfigyelt hosztra, és adott porton (alapértelmezetten az 5666-as 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 központi rendszer konfigurációs állományai a következő fogalmakkal dolgoznak:

- Host: definiálja a monitorozandó hosztot
- Hostgroup: csoportosítást tesz lehetővé a hosztok között
- Service: monitorozandó szolgáltatást definiál adott hostgroup-hoz tartozó hosztokon

A monitorozott hoszton az NRPE ágens beállításait kell megtenni. A beállítások közül kiemelendőek a következők:

- **Allowed hosts**: azon hosztok listája, akiktől monitoring kérdéseket fogadunk. 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 melyik beérkező monitoring kérdésre milyen parancs végeredményét kell megadni

A Nagios beállítások és azok szintaktikája nem egyszerű, de a beállítások logikájának megértése után a meglévő beállítások alapján újabb hosztok, szolgáltatások felvétele már egyszerű.

Saját pluginek fejlesztése is könnyen kivitelezhető, hiszen a Nagios pluginek egyszerű futtatható programok vagy scriptek, melyek előre definiált kimenettel rendelkeznek. Működése során a plugin elvégzi a szükséges vizsgálatokat, majd egy egysoros kimenetet ír a standard kimenetre, és annak megfelelő visszatérési értékkel lép ki. Ez a plugin ezek alapján lehet bármilyen bináris program vagy akár shell script is.

Az egysoros kimenet formátuma 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

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 paracssori 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 az `mc` használata, beépített szerkesztőjével könnyen módosíthatóak a beállításokat tartalmazó fájlok.

- Távoli fájlok letöltése ftp vagy http protokoll segítségével a `wget` parancs segítségével történhet.

- Konfiguráció módosítás után a szolgáltatások újraindítandóak, hogy a beállítások érvényre jussanak. Ehhez a megfelelő scriptek a `/etc/init.d` mappában találhatóak. Minden script egy paramétert vár, ami legtöbbször a „start”, „stop”, „restart” és „reload” karakterláncok közül kerül ki.

  Például az apache2 webszerver a `/etc/init.d/apache2 restart` utasítással indítható újra.

- A különböző szolgáltatások beállításait tartalmazó fájlok jellemzően a `/etc` könyvtárban helyezkednek el.

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. A mérés első felében ezen esettanulmány végigjátszása is a feladatok közé fog tartozni.

Az esettanulmány során 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 (smtp.itlab.hu) időtúllépés miatt nem válaszolt. A levélkiszolgáló szervernek mi vagyunk a rendszergazdái, így a hibát saját kezűleg tudjuk kijavítani. Mit tudunk tenni?

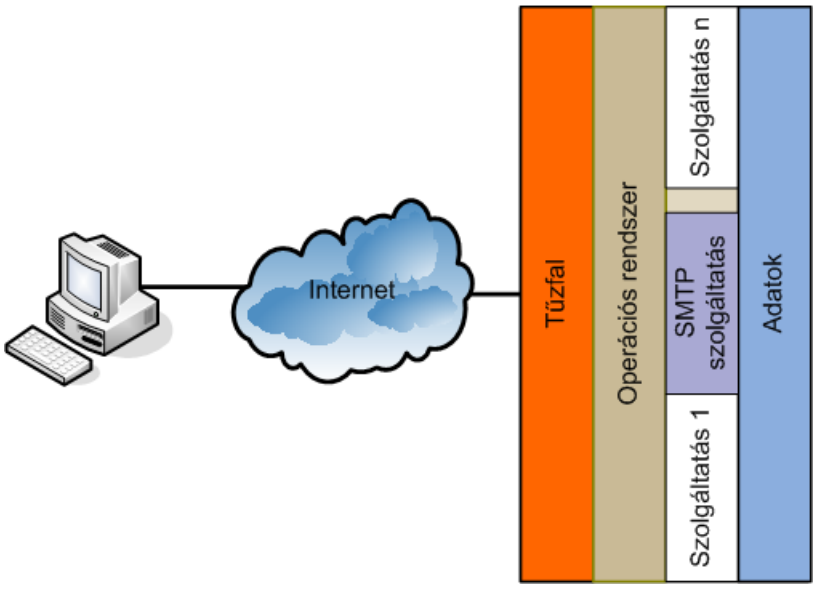
A probléma felderítés során szisztematikus diagnosztikai módszereket alkalmazunk, melyek segítségével közeledünk a probléma forrásához.

- Ábrát készítünk, melyen ábrázoljuk 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 a konkrét példa keretében bemutatjuk a diagnosztika lépéseit. Az alábbi ábrán a rendszernek a probléma szempontjából releváns komponeneset ábrázoltuk.



3. ábra. Az SMTP szolgáltatás elérése

Az ábra alapján a következő lépéssorozatot tehetjük meg:

- Nulladik lépés gyanánt 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.

```
root@desktop:# ping smtp.itlab.hu
PING smtp.itlab.hu (1.2.3.4) 56(84) bytes of data.
64 bytes from smtp.itlab.hu (1.2.3.4): icmp_seq=1 ttl=57
time=1.06 ms
64 bytes from smtp.itlab.hu (1.2.3.4): icmp_seq=2 ttl=57
time=1.06 ms
64 bytes from smtp.itlab.hu (1.2.3.4): icmp_seq=3 ttl=57
time=2.68 ms
-- smtp.itlab.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
```

- Amennyiben a szerver elérhető, akkor ellenőrizzük, és ha szükséges, akkor kinyitjuk a tűzfalon a 25-ös, levélküldésre használt portot.

- Ellenőrizzük, hogy várja-e a szerver a kapcsolatokat a 25-öson. Ezt többféleképpen ellenőrizhetjük. Megvizsgálhatjuk az nmap segítségével, hogy kívülről milyen nyitott portokat látunk:

  ```
  root@desktop:# nmap smtp.itlab.hu
  ```

  De talán ennél is egyszerűbb, ha belépünk a szerverre, és a következő utasítással kilistázzuk az éppen aktív szerveralkalmazásokat és kapcsolatokat a gépen.

  ```
  root@smtp.itlab.hu:# netstat -at
  ```

  A kimeneten láthatjuk a sort, mely jelzi, hogy a szerver az SMTP porton hallgat a localhost interfészen. Ez azt jelenti, hogy a levelező szerver csak helyi gépről származó kapcsolatokat fogad, távoli kapcsolódást nem tesz lehetővé.

- A levelezőszerver beállításait módosítva beállíthatjuk, hogy a publikus külső IP címén is hallgasson, így lehetővé téve 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`).

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.

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

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

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

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

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

[10] Rendszermodellezés tantárgy weboldala. `https://sauron.inf.mit.bme.hu/Edu/Remo/remo.nsf`.

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