



# **LOK 2009**

**követő kiadvány**

A rendezvény szervezője az [FSF.hu Alapítvány](http://FSF.hu)



A géptermi gyakorlat támogatója a [KÉK-SULI Oktatóközpont Kft.](http://KÉK-SULI Oktatóközpont Kft.)



A konferencia kiadvány a [Linux-felhasználók Magyarországi Egyesületének](http://Linux-felhasználók Magyarországi Egyesületének) és az [Internet Szolgáltatók Tanácsa](http://Internet Szolgáltatók Tanácsa) támogatásával valósult meg.



A kiadvány a konferencián készült hanganyagok szöveges átirata.

A szöveget Daczi László lektorálta.

Jelen kiadvány a Creative Commons *Nevezd meg! – Ne változtasd! 2.5*”

licenc alapján szabadon terjeszthető.



# Tartalomjegyzék

<b>Hagyományos LOK szekció.....</b>	<b>5</b>
Bánhidi Árpád: Az OpenOffice.org Impress funkcióinak megismerése, a 2008-as érettségi feladat megoldásával.....	5
Bánhidi Árpád: Az OpenOffice.org Base funkcióinak megismerése, a 2008-as érettségi feladat megoldásával.....	9
<b>EDU szekció.....</b>	<b>14</b>
Kodácsy Tamás: Elektronikus Tanulmányi Nyilvántartás.....	14
Erdélyi Gábor: E-learning keretrendszer I.....	23
Erdélyi Gábor: E-learning keretrendszer II.....	32
Németh László: A legcsodálatosabb oktatóprogram: a 40 éves Unix programozási környezet I.....	41
Németh László: A legcsodálatosabb oktatóprogram: a 40 éves Unix programozási környezet II.....	50
Szervác Attila: Zeneszerkesztés szabad szoftverrel.....	55
<b>Szabad szoftver nap szekció.....</b>	<b>60</b>
Höltzl Péter: Modern naplózás GNU/Linux alapokon.....	60
Kis Gergely: Google App Engine.....	68
Czakó Krisztián: OpenVPN.....	74
Makár Zénó: OpenSolaris.....	83
Erdei Zsolt: Green IT bevezetése egy multinacionális környezetben, egy általános iskolában ..	89
<b>Admin szekció.....</b>	<b>93</b>
Lajber Zoltán: Mi is az intranet? (fogalmak, célok, megkötések).....	93
Csillag Tamás: Virtualizáció.....	101
Lajber Zoltán: Hálózat és alap infrastruktúra.....	106
Lajber Zoltán és Csillag Tamás: Csoportmunka (webmail, címtárak, naptár, üzenőfal, nagy fájlok küldése, stb.).....	115
Lajber Zoltán: Fájl- és nyomtatószerver (SaMBa - tipikus esetek, alapproblémák).....	123
Csillag Tamás és Lajber Zoltán: Rakjuk össze az egészet (hardver kiválasztás, tippek, trükkök szerver üzemeltetéshez).....	131

# Hagyományos LOK szekció

## Bánhidi Árpád: Az OpenOffice.org Impress funkcióinak megismerése, a 2008-as érettségi feladat megoldásával

**Feladat:** az OpenOffice.org bemutatókészítője, az Impress-nek lehetőségeit megismerni. Elhoztunk egy példányt a középszintű gyakorlati vizsgafeladatokból, ha közben szeretné valaki nézni a vizsgakiírást és esetleg hozzá a javítási-értékelési útmutatót, az meg lehet tenni.

Ez a feladat annyiban érdekes, picit más hogy itt az elején már rögtön képszerkesztéssel kell indulni, ugyanis a bemutatóhoz szükséges képeket módosítani kell a forrásokhoz képest. Az első, amit meg kell csinálnunk az a hering.gif állomány, amit a vonalak.gif-ből kell előállítanunk. A mérete az úgy van megadva, „a méretet azt ne változtassa meg”, úgyhogy azzal nem kell foglalkoznunk.

**A lépések a következők:** tükrözésekkel és másolásokkal kell a feladatot elvégezni, ehhez pedig a legegyszerűbb, ha azt csináljuk, hogy az első réteget azt duplássuk, és utána a réteg menüpontból az átalakítások között kérjük a vízszintes tükrözést. Akkor látjuk, hogy átfordította a jobb oldalra ezt a képet, viszont a fehér rész az kitakarja az alatta lévő réteget, így az eszköztárak közül válasszuk ki az egybefüggő terület kijelölését és utána egy ctrl-x-szel vagy egy kivágással tüntessük el azt a fehér részt. Ezáltal a két rétegen lévő kép ezáltal egymás mellettinek látszik.

Viszont ez még mindig két réteg, ezért ezeket fésüljük össze. A réteg menüpontnál az összefésülés lefelé menüpontját válasszuk ki. Ugyanezzel az eljárással tudjuk az alsó részt is megcsinálni, tehát duplássuk először a réteget, majd egy függőleges tükrözést végezzünk, és ugyanúgy, az egybefüggő területeket kijelölve, és a fehéret törölve, akkor a kész képünk az meglesz. Ugyan még több rétegben, ezért össze kell őket fésülni.

Ha ez megvan, akkor ezt a képet viszont hering.gif néven kell elmenteni, tehát a fájl menüpontnál a „Mentés másként” opciót kell, hogy kiválasszuk. Ugye az eredeti kép az a vonalak.gif volt, és hering.gif lesz az új képnek a neve. A teljes képpel még nem végeztünk, ugyanis meg kell csinálni két párhuzamos piros vonalat kell húznunk a képre. Válasszuk ki a ceruzát, de az ecsetet is választhatnánk és ott egy 3 képpont vastagságú rajzolás az megfelelő lesz. Szintén nincs megadva a színnek a pontos kódja, úgyhogy tetszőleges piros színt tudunk kiválasztani. A színeket két kis négyzetben tudjuk kiválasztani, duplán tudunk kattintani akár az előtér, akár a háttér színre és akkor jön be az előbb látható színválasztó panel, ahol akár konkrét színt is megadhatunk. Menjünk át a képre. Hozzunk létre egy új réteget, amely átlátszó és ezen húzzunk egy egyenes vonalat. Ezt hogy tudjuk megtenni? Legegyszerűbbnek én azt találom, hogy a képen kívül kattintsunk egyet és utána nyomjuk le a shift-et és ctrl-t és utána egyenes vonalat tudunk húzni vagy 45 fokosat, ha arról van szó. Ezzel egy szem vonalunk megvan, ezt kell párhuzamossá tenni és ugyanúgy a képnek a másik részére elhelyezni. Duplássuk meg ezt a réteget, és ezen is végezzük el a függőleges tükrözésnek a funkcióját. Így a kívánalmaknak és a feladatkiírásnak megfelelő garantáltan párhuzamos vonalaink lesznek, annak ellenére, hogy vizuálisan tényleg nem annak látszik.

Ezek után egyszerűen a rétegeket fésüljük össze, vagy kétszer vagy a réteg menüpontban van egy, látható rétegek összefűzése, az a ctrl+n billentyűkombinációval ahány rétegünk van, az összes réteget egy képpé fogja összefűzni. Megvan a képünk, most már csak mentenünk kell. Ez lesz az, amit majd be kell illesztenünk a diasorba.

A feladat szerint az a következő, hogy egy serleg.gif nevű képből kell elkészítenünk a rubin1 és a rubin2.jpg képeket. Ha duplássuk a képet, akkor van két olyan képünk, amivel dolgozhatunk. A feladatkiírás szerint feketével kell kitölteni a részeket, úgyhogy megváltoztathatjuk az előtér szint csupa feketére és a vödör eszköztár, kitöltés előtér színnel, hasonló színűek kitöltése, arra figyeljünk

oda és átlátszó területek kitöltése is be lehet kapcsolva. Ilyen beállítások mellett hogyha először a bal oldali fehér területen kattintunk, majd a jobb oldalin, akkor a feladatkiírás szerinti első képet kapjuk meg, ahol mondhatjuk azt, hogy vagy egy fehér serleg vagy két fekete arc látszik. A másik képen, ugye az előbbieket tapasztalatán, a középen kattintunk a vödör eszköztár használatára, akkor a középső részt fogja kitölteni. És ezt kell elmenteni nekünk JPG formátumban rubin1 illetve rubin2 néven. Mivel fekete-fehér a kép, tehát nyugodtan a szürke árnyalatosba való konvertálást használhatjuk, nem kell feltétlenül nekünk az RGB. A tömörítésnél pedig ugyanazt a 85%-os tömörítési eljárást használhatjuk. Ő pedig a rubin2.jpg lesz.

**Ha ezzel megvagyunk, akkor ezután tudunk nekiállni a bemutató elkészítésének,** ez pedig az OpenOffice.org-ból az Impress irodai csoportból a bemutató készítő tudjuk használni. A feladatleírás meghatározza a háttérét, meg minden egyéb beállítást, ezért egy egyszerű üres bemutatóval indítsunk, ne menjünk végig a varázslónak a további lépésein. Az első dolog, amit meg kell tennünk, ugye hasonlóan a szövegszerkesztésnél is volt, hogy a megadott színt azt fel kell, hogy vegyük az OpenOffice.org-nak a színskálájába, ezért az eszközök, beállításoknál, a színeknél ezt a meghatározott színt, a 23416384-es RGB színkódolású színt elnevezzük, jobb híján most „erzekel” lesz, és hozzáadással felkerül a palettára. Okézzuk le. Más színt ebben a feladatban nem kell beállítanunk. Az első feladat, hogy a háttérszínt, az összes dia háttérszínének beállítása, azt szintén többféle módon tudjuk megtenni. A legegyszerűbb az, hogy ha a formátum->oldal menüpontjának háttér fülén kiválasztjuk, hogy színnel töltünk ki és kiválasztjuk az általunk már előbb beállított színt. Megkérdezi, hogy az összes diára? Igen, az összes diára akarjuk és ha most még 3 diát beszurunk, a feladat kívánja, hogy összesen négy diából álljon a bemutatónk, akkor látszik, hogy a következő 3 diánál is ezeket a háttérszíneket alkalmazza.

Ezen kívül a feladatleírás azt mondja, hogy a szöveg színének fehérnek kell lennie. Ehhez a legegyszerűbb, hogy ha nézet menüpontnál a minta és diasablon menüpontnál határozzuk meg, hogy a szöveg színe, akár legyen az a címsor vagy a vázlat szövege, az milyen színű. Jelöljük ki a teljes szöveget, és állítsuk be a formátum->karakter->betűhatásoknál van a betűszín. Érdemes továbbolvasni a feladatnál, ahogy a táblázatkezelésnél is volt egy ilyen példa, ugyanis a 7. pontban azt kéri, hogy a diáknak a címe, az 42-es méretű legyen, a szöveg az pedig 28-as méretű. Azt itt a 44-es méretűről 42-es méretűre állítsuk be, okézzuk le, így akkor a címdia minden egyes dián 42-es méretű lesz és utána a vázlat szöveget is ctrl+a-val az egészet jelöljük ki és akkor formátum karakternél egyrészt a betű színét fehérre állítjuk, és ez pedig 28-as betűméret. Ezzel gyakorlatilag létrehoztunk egy alapértelmezett diasablont szerkesztettünk át, amely az összes diára érvényes. Ezt zárjuk be, és akkor visszakerültünk a már meglévő 4 diánkhoz, ezt azért mentjük el a biztonság kedvéért. A feladatleírás az azt mondja, hogy „erzekel” néven mentjük el.

Ugye az első az egy címdia, azt kéri és oda az érzékelés, észlelés szöveget kell beírjuk. Válasszuk ki a „csak cím” típusú dia-elrendezést és akkor írjuk be azt, hogy „érzékelés, észlelés”. Ezt a feladat mintája szerint középre kell, hogy helyezzük. Ezt megtehetjük akár úgy is, hogy az igazításnál van egy olyan eszközünk, hogy függőlegesen és vízszintesen is tudunk középre helyezni vagy igazítani. Ezek közül kiválasztjuk a függőlegesen középre igazítást és így a pozícióval nem kell már többet foglalkoznunk.

A második diára pedig egy olyan elrendezést kell, hogy válasszunk, ahová már képet is tudunk beilleszteni. Ennek a diának a címe a „Laterális gátlás Hermann rács”, tehát ide a címbe ezt kell begépelni és ide fog bekerülni az a kép, amivel eddig nem foglalkoztunk. Ha ezt az elrendezést választjuk, akkor most jelen pillanatban a bal oldalon van egy objektumtároló, arra egyszerűen duplán kattintva kiválaszthatjuk, hogy melyik képet akarjuk és oda arra a pozícióra ezt be is fogja illeszteni. A mintadia szerint, ami a feladatleírásban van, a szövegnek alatta kell lennie, ezt az ellenőrzést majd később megcsináljuk.

Egyelőre gépeljük be a megadott szöveget: Számolja meg, hány fekete pontot lát a képen! Az első, amivel érdemes foglalkozni az az, hogy a képet igazítsuk be. Láthatóan a diaminta szerint

függőlegesen és vízszintesen is középre van igazítva. Úgyhogy jobb egérgombbal kattintsunk rá, és az igazításnál válasszuk ki, középre igazított vízszintesen és függőlegesen.

Ezután már csak az a dolgunk, hogy a szövegdobozt azt úgy pozicionáljuk, hogy a kép alatt a szöveg középen jelenjen meg. Felülről a méretét azt csökkentjük le, és utána a szélességén igazítsunk annyit, hogy egy sorba kerüljön a szöveg. Látszik, hogy alapértelmezetten mivel ez egy felsorolás szöveg volt, ezért balra igazított, tehát a szöveg igazítását mindenképpen középre kell, hogy igazítsuk és nem árt, hogyha nem tudunk pontosan szemmel igazítani, a szövegdobozt magát is középre igazítjuk, így biztos, hogy a szövegünk az jól fog megjelenni. Akkor ezzel meg is van ez a dia. Mentünk egyet.

És akkor nézzük a következőt, ez a hering vonalak lesz, tehát az előbb szerkesztett képet kell hogy majd beillesztjük. Ugyanaz az elrendezés, egy kép és egy szöveg. Itt a dia címe: „Hering vonalak”.

A kép beillesztésénél ugyanazt az elvet tudjuk követni, mint korábban, és a feladat szerint azt kell begépelni, hogy a két piros vonal görbe vagy egyenes? Ez a feladatsor nem túl bonyolult, bár a bemutató készítésénél sosem adnak olyan túlságosan bonyolult feladatokat, viszonylag egyszerűen lehet ezeket megoldani. Ugyanúgy ezeket az igazításokat, vízszintesen és függőlegesen igazítsuk középre a képet és ugyanúgy a szövegdoboznak a pozicionálásával a szöveget a kép alá igazítsuk.

Azt megtehetjük, ha a diára kattintunk jobb egérgombbal, van olyan hogy másolás és akkor egy beillesztés megjelenik a dia után, de azt meg törölhetjük. Az utolsónál kell a két serleget beilleszteni. Ennek a diának pedig az a címe, hogy „Figura és háttér Rubin serleg”. Mondjuk sokat nem spórolunk, mert itt például pont két képet kell beilleszteni. Beszúrás képfájlból, azzal le tudjuk cserélni az egyik képet, a rubin1-et. Viszont ebben az esetben az előző kép méretét fogja átvenni ez a kép, így az eredeti méret menüpontjára kattintva megkapjuk azt a méretet, ami volt. Most külön kell beszúrnunk fájlból másik a JPG állományt.

Azért, hogy azt az elrendezést kapjuk, mint ami a mintadián van, a legegyszerűbbnek azt látom, hogyha mindegyik képet először vízszintesen és függőlegesen is középre igazítjuk, majd, hogyha a shift billentyű lenyomása mellett vízszintesen elmozgatjuk az egyiket, akkor csak ebben az irányban tudjuk mozgatni, függőlegesen nem. Ezért nem lesz abból probléma, hogy a két kép egymáshoz képest úgy mozdul el, hogy függőlegesen néhány pixelnyi vagy milliméternyi távolság van a kettő között. „Két emberi arc vagy egy serleg”, ez a kép aláírása. Ezt mentjük el.

Ami hátra van, az a diavetítési beállítások, az alul van, az eszköztárban, a diaátmenetnél tudjuk megtenni, nincs megadva a feladtleírásban, hogy milyen áttűnést alkalmazzunk, úgyhogy tetszőlegesen válasszuk ki. Egy dolgot kell csak beállítanunk, azt, hogy 5 másodpercenként történjen ez a váltás az összes diára. A sebességet azt szerintem állítsuk gyorsra, jobb a szemnek. Alkalmazás az összes diára és akkor lépünk az első címdiát F5-tel nézzük meg a következő 20 másodpercben. Most a projektor vágja le a szélét a képnek, azért látjuk így, és elvileg pontosan 5 másodpercenként kell, hogy váltson. Két kattintás után lép csak ki, mentjük el azért, amit csináltunk.

Volt egy korábbi feladat, ha jól emlékszem a 2006-os bemutató készítésénél volt az a feladat, hogy folyamatosan, az Esc billentyű lenyomásáig vetítsen. Ebben a verziójában az OpenOffice.org-nak ezt biztos nem lehet beállítani. Tehát az sajnos pontvesztés, hogyha ilyen feladtkiírás van. 3.0-ban ezt még nem volt időm megnézni pontosan, de az a gyanúm, hogy még ott sincs ilyen funkció. Attól függetlenül, hogy maga a bemutató készítés is egy alacsonyabb pontszámú és egyszerű feladat, lehet egy-két olyan paraméter, ami pontvesztést eredményezhet. Az egyik lehet az konkrétan, hogy ha a vetítési beállításoknál olyat kérnek, hogy az folyamatosan, az Esc billentyű megnyomásáig legyen.

Egyébként ahogy végignéztem a középszintű feladatokat, olyan paraméter ezen kívül, ami nem volt teljesíthető, nem volt. Tehát egyszer volt egy ilyen egy pont vesztés. Ha azt nézzük, hogy ez nem a diák hibája, akkor végül is max. pontszámot adhatunk neki, mert nem tudta beállítani, de azért nem néz ki jól, hogy tulajdonképpen van egy feladat, ahol bizonyos paramétereket, akár csak egyet is, nem lehet beállítani. Úgyhogy ez részben a feladtkiírónak a felelőssége vagy feladata, hogy mivel a gyerek az stresszben lesz addig, amíg meg nem kapja a pontszámokat, hogy milyen lett az

értékelése, meg a szoftverfejlesztőké is, hogy ugyanazt a funkcionalitást tudják. Valahogy ezt a kettőt össze kell hozni. Maga a feladatmegoldás az ennyi volt.

Ami érettségihez kell, amit egy diáknak nagyjából, a funkcióit tekintve tudnia kell a bemutató készítésnél, azok ezek a feladatok. Itt különösebben többet nem lehet mutatni, persze van lehetőség sablonokat készíteni, tehát a nézet menüpontnál a Minta->diasablonra kattintunk, ott nem csak ezt az alapértelmezett mintáz készíthetjük el, hanem saját elrendezéseket hozhatunk létre, tehát egy új mintát készíthetünk. Ennek más lehet az elrendezése, más lehet a háttérszíne, előre megadhatunk szöveget benne. Gyakorlatilag egy előadáshoz a címdiát, a köztes diát meg a végső diát is ilyen sablonokból, mintanézetekből elkészíthetjük, hogyha a diára kattintva minta átnevezése alatt nevet adunk neki és a mintaoldalaknál meg is jelenik. Hogyha rákattintunk, akkor ezt a sablont hozza be. Ez az egyik hibája, hogy az alkalmazás után eltűnik, legalábbis a jobb oldali munkapanelben. Egyébként, ha jobb egérgombbal választjuk ki a diasorban a diaelrendezésnél, akkor megjelenik egy külön panel, innen nem tűnik el. Gyakorlatilag, ha ezt a diatervező paneljét használjuk, akkor a korábban beállított sablonjaink azok mind megmaradnak és akkor azokat alkalmazhatjuk. De ez eddig így még nem volt érettségi példa.

**Hallgató:** van-e olyan sajátossága a programnak, ami megkülönbözteti a PowerPointtól, valami, ami többet tud?

**Előadó:** végiggondolva nagyjából ugyanazt tudja. Nem tudok sem negatívumot sem olyan többletet funkcióban most mondani, ami miatt megérné bemutatót készíteni mindenképpen ezzel.

## **Bánhidi Árpád: Az OpenOffice.org Base funkcióinak megismerése, a 2008-as érettségi feladat megoldásával**

**Következő feladatunk az OpenOffice.org-nak az adatbázis kezelő funkcióját megismerni.** Itt lesznek érdekességek a korábbiakhoz képest, mert egyrészt több a lehetőség, másrészt meg több a probléma is. Az első dolog, amit el kell, hogy mondjak, hogy 3 lehetőségünk is van kapásból, ami tulajdonképpen csak kettő. Az egyik, hogy van egy beépített adatbázis a HSQL motor, ez az egyik lehetőség az adatbázis feladatok megoldására. A másik pedig az, ha valamilyen SQL szerveret használunk az OpenOffice.org háttereként vagy backend-jeként. Ekkor a táblákat és magát az adatbázist, azt ebben az adatbázisban tárolja el, ami ugye MySQL vagy PostgreSQL jellemzően Linux alatt. Illetve bármi olyan adatbázis kezelőt használhatunk, amihez van JDBC motor vagy JDBC osztály, és ez telepítve van a rendszerünkben. Illetve ODBC-n keresztül is van csatlakozási lehetőség.

**Nyissunk meg egy OpenOffice.org Base-t,** hogy lássuk, hogyan is néz ki egy adatbázisnak a létrehozása az elején. A „Mit kíván tenni?” Kérdésre, ha azt választjuk, hogy új adatbázis létrehozása, akkor ez a HSQL motor, HSQL adatbázis motort jelenti. Ha itt most a befejezésre kattintunk, akkor egyből a mentés jön, és rögtön megjelenik az a panel, amiben dolgozhatunk. Ez többé-kevésbé hasonlít az Access-hez, megvan a táblák fül, megvan a lekérdezések, űrlapok, jelentések. Egy baja van, hogy ez elég sűrűn le szokott fagyni. Volt már 2 évvel ezelőtt is egy előadás, ahol szintén az adatbázis kezelési feladatot egy érettségi példán mutatták be, ott azt hiszem kétszer fagyott le a 45 perc alatt. Nem biztos, hogy ezzel az adatbázismotorral érdemes foglalkozni, már én is többször belefutottam abba, hogy ez munka közben egyszerűen leállt és akkor mondtam, hogy megnézzük, mi van másik lehetőségként. Megnézem, hogy ha egy adatbázis SQL adatkezelőben van eltárolva, akkor mit tud. Jelentem, akkor legalább stabilan fut. Még ha szűkösek is a lehetőségek, legalább nem fagy le amiatt.

**Menjünk vissza az elejére.** Itt fogjuk majd látni ezen a panelen, hogy kapcsolódás létező adatbázishoz. Ez már egy kicsit gyanús kell, hogy legyen nekünk, ugyanis itt már nem azt mondja, hogy új adatbázist hozunk létre, hanem kapcsolódunk egy létezőhöz. Na most ha elővesszük a feladatot vagy bármilyen adatbázis kezelő feladatot kell megoldanunk, akkor azt mondja, hogy készítsen új adatbázist „vizállás” néven. Ha JDBC-n, ODBC-n keresztül kapcsolódunk egy ilyen külső adatbázishoz, akkor ez azt fogja jelenteni, hogy már létezőhöz akar kapcsolódni, ergo nekünk azt valamilyen módon létre kell hoznunk. Jelen esetben a legegyszerűbb az, ha valamilyen terminálban kapcsolódunk az adatbázis kiszolgálóhoz, és ott létrehozuk az adatbázist. Utána a táblákat a MySQL felületen belül létre tudjuk hozni, tehát az már nem jelent problémát. Magát az adatbázist az SQL, MySQL vagy PostgreSQL-en belül nem tudjuk megcsinálni.

**Akkor most az első változatban nézzük meg a PostgreSQL-t, „psql” paranccsal.** A környezet az úgy van beállítva, hogy alapértelmezetten azzal a felhasználóval jelentkezheti be, aki ezt kéri, a jelszó bekérése. „create database vizallas” néven a feladathoz tartozó adatbázist akkor hozzuk létre. Nézzük meg, hogy valóban létrejött-e az adatbázis, igen ott van, van egy vizállás nevű adatbázis, szöveges felhasználó a tulajdonos, UTF8-at tud, tehát ez így valószínűleg jó lesz nekünk. Lépünk ki ebből egyelőre, ez így bőven jó nekünk. A PostgreSQL kapcsolatot most megmutatom. Magát a feladatot majd minden lépésében azt a MySQL-en fogom megmutatni vagy azon fogom végigcsinálni.

A lényeg az, hogy a kapcsolódáshoz válasszuk ki a JDBC típusú kapcsolatot. Ha ott lenyitjuk, azért azt nézzük meg. Ha van Oracle, ahhoz is kapcsolódhatunk, nyilván ez nem érettségien érdekes, hanem máskor. Illetve akár DBase vagy munkafüzet, tehát jó néhány ilyen forrást választhatunk. Most a JDBC az jó, tovább, itt meg kell adni a forrás URL címét, ami úgy fog kinézni, hogy postgresql://localhost:5432/vizállás. Azt mondjuk, hogy a JDBC-nél PostgreSQL típusú

adatbázisunk van, nem kell, hogy hálózat legyen, a helyi visszacsatolt hurok interfészen, loopback-en van ez a localhost név, megadjuk egzaktan a port számot, ahol a szolgáltatás figyel, ugye „/” jellel elválasztva az adatbázis nevét adtuk meg. A JDBC illesztő program az pedig org.postgresql.Driver. Meg kell adni pontosan nekünk azt a JDBC osztályt, amit felhasználunk az adatbázis kapcsolathoz. A postgres SQL-nél itt látszik, hogy org.postgresql.Driver. Valahogy azonosítanunk kell magunkat az adatbázis kezelő felé, ez nálunk most „oveges”, mivel jelszó is kötelező, ezért kell azt a pipát beikszelni.

Hozzunk létre egy „A” nevű táblát, tábla létrehozása tervező nézetben valami egyszerű szövegtípussal, hogy ha belenézünk az adatbázisba, akkor valóban ott van az a tábla. Ezt mentjük el „A” néven, okézzuk le, zárjuk be és nézzük meg parancssorba, hogy valóban megvan a tábla. Ez „\c”-vel kapcsolódunk az adatbázishoz és egy „\d”-vel meg tudjuk nézni, hogy ott van az „A” nevű táblánk. „\d” után írjuk, ge, hogy szóköz „A”, és megmondja, hogy milyen típusú mezők vannak benne, amit az OpenOffice.org-ban létrehoztunk, azt itt megcsinálta. Akkor PostgreSQL-ről ennyit.

MySQL-hez a kliensnél egy „-p” kapcsoló kell, alapértelmezetten a kliens nem kérné a jelszót. Itt is létrehozuk a vízallas adatbázisunkat, itt a „show databases”-szel meg tudjuk nézni, hogy ott valóban létrejött, nem is adott hibaüzenetet.

**Feladat:** kapcsolódás létező adatbázishoz. Itt megtehetnénk azt is, hogy nem JDBC-t választunk, hanem mondjuk kiválasztjuk a MySQL-t. Ez azért jó nekünk, mert ha továbbra is kapcsolódás JDBC-vel, akkor a következő panelen már egy kicsit felhasználóbarátabb az a felület, amellyel a kapcsolatot meg tudjuk adni, mert megmondja, hogy ide ebbe a mezőbe az adatbázis nevét írd be, ebbe a mezőbe a kiszolgáló nevét írd be, a portszámot megadja, az osztály, a MySQL JDBC osztályát meg automatikusan ki is tölti. Itt látszik, hogy ez egy hosszabb paraméter: com.mysql.jdbc.Driver. Ennyivel könnyebb a kapcsolat létrehozása. Ha a JDBC-t használjuk, ott ezeket végig be kell gépelni, kicsivel többet. Teszteljük, hogy ez így jó-e? Tovább is léphetünk. A felhasználó azonosítás az ugyanaz, mint az előbb is volt. „vizallas” néven ezt el tudjuk menteni. Van adatbázisunk, most már a táblát is be kell, hogy illesszük. Ez sajnálatos módon nem megy olyan egyszerűen, mint Acces-ben, hogy a fájl menü, importálás külső adatforrásból vagy állományból menüpontját használom, hanem picit trükközni kell. A lényeg, hogy a forrásállományt, azt egy munkafüzet lapba kell, hogy betegyük és onnan tudjuk majd beilleszteni. Egyszerű szövegszerkesztővel megnyitjuk, ctrl+c-vel bemásoljuk és egy új munkafüzet lapba egyszerűen csak beillesztjük. Az elválasztójelre kell odafigyelni, az tabulátor és gyakorlatilag okéra kattintva hétezer valahányszáz sort így be is illeszt. Ha itt ctrl+a-val kijelöljük és ctrl+c-vel vágólapra bemásoljuk az adatunkat, akkor visszatérve az adatbázisunkba, jobb egérgombbal kattintva beillesztés menüpont kiválasztása után meg fog jelenni egy panel, ahol bekéri a tábla nevét. Ez a feladat kiírás szerint legyen „meres”. Itt a tábla adatainak beimportálásakor és a mezők létrehozásakor is már megadhatjuk azt, amit később jelez a feladat számunkra, hogy egy „id” nevű elsődleges kulcsot illesszünk be a táblába. Ez már rögtön az importálás lépésénél megtehetjük.

Megkérdezi, hogy azokból az oszlopokból, ami ebből az adatforrásból származik, melyiket akarjuk majd a mi adattáblánkban alkalmazni és eltárolni? Jelen esetben az összeset. Kijelöljük, áttesszük a jobb oldalra. Ennél a panelnél, a típusformázásnál meg kell adnunk azt, hogy ami egyszerű szövegformátumként átjött, az az adatbázisunk szempontjából milyen típusú mező lesz. Minden oszlopon kell valamit állítani, a dátum mező az „d” típus, dátum lesz, a vízallas egy egyszerű egész, tehát ott integer-t, „int”-et kiválaszthatunk. Én a városnál és a folyónál, nem írja elő a feladat, de mondjuk egy 50 karaktert, mondjuk az is sok állítanék be. Tulajdonképpen variable karaktertípusú a mező, nem volna gond a 255 sem, de adatbázis tervezés szempontjából azért ez lényeges. Kattintsunk a létrehozás gombra, ez hosszú ideig tart, mert több, mint 7000 rekordot kell, hogy beillesszen. Ugye „use vizallas” és „show tables”-szel nézzük meg, describe meres... megcsinálta. „select \* from”-mal meg tudnánk nézni, hogy valóban a tartalom is benne van. Lépünk ki „quit”-tal. A terminált is bezárhatjuk. Most már bezárhatjuk a táblázatkezelőnket is, importálás alatt nem célszerű bezárni, mert onnan olvassa be a rekordokat. Nagyjából 10 perc

kellett hozzá, hogy az adataink ott legyenek, ahol szeretnénk. Kicsit macerásabb, mint ahogy az Access-nél megszoktuk, de innentől kezdve azért nagyjából ugyanazt kell csinálni.

**Az első lekérdezés az a 2002 szilveszter napján mért vízállásoknak a megjelenítése.** Ennél a lekérdezésnél használjuk a tündért. Egyébként nem érdemes igazából használni, mert látjuk majd, hogy ügyis hozzá kell nyúlni a szerkesztő nézetben. A feladat azt mondja, hogy egyes településeken milyen vízállást mértek. Ebből az következik, hogy a város és a vízállás mező mindenképpen kell, viszont a dátumot meg kell, hogy adjuk, ezért azt a mezőt is szerepeltetni kell a lekérdezésben. Rendezés az most nincs meghatározva a feladatban, ezért ezt nem állítjuk be. Feltétel van, így a dátum mezőre meg kell adnunk a 2002.12.31-et értéként. Csoportosítani nem fogunk, mert négy darab értéket kell, hogy kapjunk, azok között nagyon nincs mit. Ha azt szeretnénk, hogy a mező nevek helyett más jelenjen meg, ezek mező álnevek, akkor itt még van lehetőség beállítani. Egy áttekintés van, ahol megadhatjuk a lekérdezés nevét egyrészt ez a kettő szilveszter és a végén dönthetünk, hogy megjelenítjük vagy szerkesztjük a lekérdezést. Válasszuk a szerkesztést most, itt látszik, hogy ez a szerkesztő nézet, gyakorlatilag olyan felületében, mint az Access qv rácsa és itt a dátumot azt a láthatóságból vegyük ki. Mentsük el. Van egymás fölött elhelyezkedő korongok ikon, a lekérdezések futtatását teszi lehetővé és látjuk, hogy meg is jeleníti mind a négy rekordot. Budapest, Majsz, Nagymaros, Szeged vízállásait ezen az adott napon. Ez egy viszonylag egyszerű lekérdezés, egyébként ebben a feladatsorban talán egy összetettebb van, a többi az viszonylag könnyen megoldható.

**A következő lekérdezés, hogy jelenítsük meg ABC sorrendben a táblázatban található városok nevét.** Ezt akkor lekérdezés létrehozása tervező nézetben, szokásos tábla hozzáadás panelben. Csak a város mező fog megjelenni, és itt a feladat az, hogy a város egyszer jelenjen meg. Ha most ezt lefuttatnánk a korongokra kattintva, akkor azt látjuk, hogy Budapest, Budapest aztán jön a következő város. 7000 rekordban jó párszor kell lefelé görgetni, míg másik várost találunk, tehát lehetővé kell tenni azt, hogy ezek az ismétlődések eltűnjenek, egyszer jelenjen meg valami. Egy SQL-ben mi ez? A distinct záradék, illetve kitétel. Itt van egy külön indíték arra, hogy a grafikus megjelenítésbe ezt be tudjuk kapcsolni. Ez a különböző értékek nevű, gyakorlatilag szűrő, ahhoz hasonlít az ikonja. Ha most ezt bekapcsoljuk, akkor a 4 darab dekódot kell, hogy kapjunk. Ahhoz, hogy grafikus felületen ezt el tudjuk érni, jobb oldalt felül a szűrő alatti 1,2,3 feliratú gombra kell kattintani és ez fogja eredményezni azt, hogy a distinct-et elhelyezi az SQL utasításban. Egyébként, ha kikapcsoljuk azt a vonalzós ikont, akkor alul megjelenik az SQL lekérdezés. Tehát azt egyből láthatjuk, hogy mi az, amit generál ebből, szabvány SQL lejkérdezéseket készít, nem úgy, mint az Access, tehát van egy kicsi sajátossága is például a having-nél.

**A következő feladat** szintén egy egyszerű lekérdezés, hogy 9 méternél nagyobb Tiszán mért vízállásokat kell megjeleníteni. Ugye megjelenítjük egyszer a vízállást, meg a folyó mezőt, illetve ezt nem is kell megjeleníteni viszont feltételként oda kell írni, hogy Tisza. És mivel ES kapcsolatról van szó ugyanabban a sorban kell a nagyobb, mint 900 feltételt beírni. Bocsánat, itt számolni kell. Ha ezt futtatnánk le, akkor kilistázná, hogy a Tiszán mely értékek voltak ezek. Válasszuk ki a meres.\* mezőt és alapértelmezetten ez a qv rács úgy jelenik meg, hogy mind a rendezés, mind a függvény sor is megjelenik. Alapból az Access-nél ezeket külön nézet menüpontból kell előcsalogatni, itt alapértelmezetten ezek már mind láthatók, így a függvény oszlopnál egyszerűen a darab függvényt ki tudjuk választani. Tulajdonképpen a másik két mező nem is kell, hogy látható legyen, és ezzel, hogy ha ezt lefuttatjuk, 13 darabot kell, hogy kapjunk. Itt a le-föl nem mondja meg, hogy mi legyen a mező neve, így count\* szerepel ott, de darab legyen az alias. Mentés. Látszik, hogy ugyanazokat a lehetőségeket a számolásra, illetve van átlagfüggvény, ezen a szinten még minden ugyanolyan, mint az Access-ben, tehát itt még nincs különbség.

**A következő,** itt már lesz egy apró eltérés, Budapesten meg kell mondani, hogy mely napon mérték a legmagasabb vízállást. Egyrészt itt a város mező kell, ahol Budapest lesz a feltétel, és kell a

vízállás, itt viszont a rendezésnek meg csökkenőnek kell lennie. Nem bonyolítjuk meg az életünket, csökkenő sorrendbe tesszük a vízállást és a megoldó kulcs azt mondja, hogy használjuk a top1 lehetőséget, amit Access-ben egy gombra kattintva, a menüből könnyen ki tudjuk vadászni és elő tudjuk hozni, hogy a lekérdezés soraiból, maga a lekérdezés, amit létrehozunk mennyit jelenítsen meg. Erre az OpenOffice.org adatbázis kezelőben nincs lehetőség, úgyhogy itt már az SQL szerkesztő nézetét kell, hogy használjuk és a limitet kézzel kell külön beírjuk. Ha csökkenő sorrendbe rendezem a vízállást, akkor az első lesz a legnagyobb ugye az adott településen. A dátumot meg kell jeleníteni. Abban az esetben, ha én a qv rácsot használom és van egy mező, ahol bekapcsoltam a rendezést, akkor annak a láthatóságát nem tudom kivenni. Az SQL nézetben kivehetem, hogy szeretném, hogy az az oszlop ne jelenjen meg és tudom utána a rendezést megadni az SQL végén, azzal nincsen probléma. Itt, ebben a grafikus felületben, ha már beállítottam rá a rendezést, akkor nem vehetem le a láthatóságot. Már csak ezért is kell az SQL nézet nekünk, ahol a vízállást kivesszük, mert csak a dátum érdekes és a végére kell beírni, hogy limit szóköz 1. Ez így még nem lesz jó, mert jobb oldalon megjelenik egy négyzet SQL felirattal, az SQL közvetlen futtatása. Csak enne a bekapcsolása esetén fog helyesen lefutni az OpenOffice.org-ból ez a lekérdezés. Ha nem kapcsoljuk be azt a kis négyzetet, akkor hibát fog adni. Belefutottunk két dologba is: ha rendezést adok meg egy mezőre, akkor nem vehetem ki a láthatóságot, illetve a limit 1-et azt külön menüpontból vagy valamilyen más beállításból nem adhatom meg, csak ha SQL-be teszem.

**A 6. feladat** vagy a lekérdezéssel működik, én magam kell, hogy megírjak egy „select”-et, amiben van egy al-lekérdezés és annak az eredményét használom fel, vagy segédtablát, segédlekérdezést készítek, és azt kell megmondani, hogy a nyilvántartott legmagasabb vízállás, az a 928 cm és ezzel egy napon a Duna mentén melyik településen, milyen vízállást mértek. Ez valószínű a Tiszán volt. Az első lekérdezésünk az arról fog szólni, hogy a dátumot kikeressük erre a legnagyobb vízállásra. Kell a dátum mező és kell a vízállás mező, ami nem kell, hogy látszódjon, most lényegtelen, ahol a feltételünk az lesz, hogy 928, ugye ezt az értéket keressük, ennek a láthatóságát ki lehet kapcsolni. És ha lefuttatjuk, akkor egy dátumot kell, hogy adjon. Ezt mentsük el altsegéd néven. Akkor most készítsük el azt a lekérdezést, ahol egyszer a mérés és a lekérdezések közül a ha segéd szerepel és grafikuson kössük össze a két dátummezőt. Ugye ez fogja számunkra lehetővé tenni azt, hogy ahhoz a dátumhoz tartozó rekordokat fogja csak megjeleníteni, ami az al-segédlekérdezés visszaad. Egyszer kell a folyó, mint mező, kell a város és a vízállás. És a folyóhoz meg be kell írni a Dunát a feltételhez. Szerintem a folyót kapcsoljuk ki, a város és a vízállás a lényeg. Ezt futtassuk le, azt fogja megmondani, hogy Budapesten ezen a napon 430 cm volt, másik két településen, meg amit ott éppen mértek. Ezt pedig 6 cm 928 néven mentsük el.

**A 7. feladat, az egy jelentéskészítés.** Ennek a jelentésnek a végrehajtása, ez is egy olyan feladat, amit az OpenOffice.org-nak a beépített jelentés kezelőjével maradéktalanul nem tudjuk megoldani. Ahhoz, hogy ez rendesen, a feladat kívánalmainak megfelelően jelenjen meg, és készüljön el a jelentés, egy lekérdezést kell létrehozunk. Mégpedig azért, készítsünk egy jelentést most lekérdezés nélkül. Az a feladat, hogy város és azon belüli hónap szerinti csoportosításban jelenítsük meg a dátumot és a hozzá tartozó értéket. Végül is az id kivételével és a folyó kivételével, pontosan a dátum, a vízállás és a város kellenek. Kiválasztottuk, hogy a mérés táblából kellenek ezek a mezők, most egyelőre, ennél a lekérdezésnél ne foglalkozzunk a mező nevekkal. Ugye egyszer város szerint kell csoportosítanunk, ugye az lesz az első, majd dátum szerint, illetve hónap szerint kéri tőlünk a feladat, de ha továbblépünk, gyakorlatilag csak a rendezést adhatjuk meg, tehát azt nem mondhatjuk meg, hogy dátum szerint hogyan legyen ez a csoportosítás. Így gyakorlatilag nem tehetünk olyan feltételt a jelentés készítésén belül, ami lehetővé tesz, hogy a dátum egészére, év, hónap vagy a hónapra vonatkozólag legyen egy csoportosítás, így most a teljes dátumra marad, a többi az meg jelentéstündér. Ha ezt befejezzük, és megnézzük, akkor gyakorlatilag mire végigpörgeti a végén, nem lesznek havi lebontásban a vízállásjelentésünk, hanem városonként ugyan csoportosítja, de hát egymás után fogja megjeleníteni az egyes rekordokat.

Látszik, hogy gyakorlatilag dátumonként létrehozott egy csoportot. Ugye nem ez a cél, hanem hogy havonta legyen, ehhez viszont mindenképpen kell, hogy a jelentés formátuma is nem biztos, hogy a legjobb. A lényeg, hogy mindenképpen kell egy lekérdezést csinálni, hogy ez megfelelően működjön.

**Menjünk vissza a lekérdezésekhez és készítsünk egyet.** Tehát a városnak kell benne lennie, a dátumnak, a vízállásnak és majd lesz egy negyedik mezőnk, ezt én hónapnak neveztem el. Az nincs benne, úgyhogy térjünk át az SQL nézetbe és a mező neve felsorolásához fogjuk beilleszteni ezt a mezőt. Van-e valakinek ötlete, hogy hogyan egészíthetnénk ki ezt a lekérdezést, hogy a hónap megjelenjen? Az a baj, hogy ha egy függvényt használunk, akkor nem biztos, hogy azt a függvényt mondjuk a diák is tudni fogja, mert nem gyakori, meg ha egyszer megtaníttuk, nem biztos, hogy eszébe jut akkor. Van egy concat függvény, ami a szövegek összefűzését teszi lehetővé és azzal manipulálok, hogy a year és a month függvényekkel kivesszem a dátumból az évet, a hónapot és ezt szöveggé összefűzöm ponttal elválasztva és így létrejön egy 2000. január vagy 2001. január érték stb. Ez alapján, ha csoportosítom, úgy fog megjelenni a lekérdezésben, hogy meg fog jelenni a város alatt az, hogy 2000. január, mint hónap és az alatt csak a januári értékek lesznek. Akkor a vízállás után, a legvégére szűrjük be oda vessző concat kerek zárójel year, az év, kerek zárójel datum, megadjuk, hogy ez a datum nevű mezőből kell nekünk, kerek zárójel zárjuk be, vessző és aposztrófok között ott legyen egy pont, ugye ez fogja elválasztani az évet a hónaptól, vesszővel elválasztva meg jön a month függvény, kerek zárójel datum és két bezáró kerek zárójel. Viszont így csúnya lesz az oszlop név és nehéz lesz rá hivatkozni, úgyhogy ez kulcsszóval hónap ékezet nélkül szerepeljen. Ezt így mentjük el praktikusán héthavi néven, ahogy egyébként a jelentést is el kell majd készíteni. Innentől kezdve már egyszerű a dolgunk, mert ahogy az előbb láttuk, annak megfelelően, csak erre a lekérdezésre alapozva kell jelentést elkészítenünk. Amikor a forrást kiválasztjuk, akkor a héthavi lekérdezést kell kiválasztanunk, viszont akkor már az összes mező kell, nem kell válogatnunk. Város és hónap, az a két csoportosítási alap, ami alapján a listánk elkészül. Azt hiszem a vázlat kiemeléssel az, ami jól olvasható lesz, és kattintsunk a befejezésre. Gyorsabban teszi be a rekordokon is valamiért. Tehát látszik: város, hónap. Megvan 2000. első hó és ott megvannak a januári értékek, aztán jön a második hónap, ott megvannak a februári értékek. Maga a feladatléírás ezt kéri tőlünk számon, azt így meg tudjuk oldani. Az Access-ben jobb lehetőség van e jelentéskészítésre. Ha itt megnézzük, maga a tervező nézet is bonyolultabb. Nem is nagyon szeretem a tervező nézetét, bár csináltam egy-két űrlapot meg jelentést is, de nem találtam olyan kezesnek, mint ahogy az Access-ben megszoktam vagy megszoktuk. Ez volt ebben a feladatsorban az utolsó példa, remélem látszik, hogy nem lehetetlen megoldani, csak picit több SQL ismeret kell hozzá, meg egy picit furfangosnak kell lenni, hogy ha a jelentés az nem hajlandó, akkor valahogy máshogy megoldjuk.

# EDU szekció

## Kodácsy Tamás: Elektronikus Tanulmányi Nyilvántartás

Az elektronikus tanulmányi nyilvántartásról szól az első előadás. Az elektronikus tanulmányi nyilvántartás az egy felsőoktatásban használt tanulmányi rendszer. A célja az, hogy a felsőoktatásban lévő folyamatokat, adminisztrációs folyamatokat menedzselje és ezentúl a felsőoktatási intézményeknek olyan eszközöket adjon, ami a mindennapjaikban hasznos lehet, tehát nem csupán tanulmányi adminisztrációt képes ellátni ez a szoftver, hanem intézményi honlapot generál, oktatói honlapot generál és még egyéb más dolgokra is képes. 2001-ben kezdődött a fejlesztése és mind a mai napig tart.

**Ez a harmadik legelterjedtebb felsőoktatási rendszer Magyarországon.** Az első kettő a Neptun és az ETR, azok .NET alapúak, tehát mondhatom, hogy a legelterjedtebb linuxos tanulmányi rendszerről van szó, amely jelenleg 8 intézményben fut. Ezek kis intézmények, tehát jellemző rájuk, hogy kis intézményekről van szó. A fejlesztés 2001-ben a Debreceni Református Hittudományi Egyetemen kezdődött, és meg is maradt ilyen teológiai síkon, illetve vannak nem olyan intézmények például a mozgássérültek Pető András Nevelő Képző és Nevelő Intézete, ami szintén ETN-t használ az Adventista Teológia Főiskola, Baptista Teológiai Akadémia, DRH Evangélikus Hittudományi Egyetem, Pető Intézet, Pápai Teológia, Püskösdi Teológiai Főiskola és a Sárospataki Református Teológiai Akadémia. Ezek használnak ETN-t.

**A rendszer az teljesen Linux alapú,** kísérleteztünk több disztribúcióval az elején, RedHat-tel indult, volt Mandrake, a vége az lett, hogy mindenhol Debian van. Tehát ma mindegyik ETN mögött Debian fut, PHP illetve most már PHP5 a kliensfelület illetve ami generálja a kliens felületet, az adatbázis szerver mögötte az mindenhol PostgreSQL, tehát nem a MySQL-t, hanem a PostgreSQL-t választottuk. És most már egyre több Javas alkalmazással bővült ki, tehát a technikai háttér ez. Mindenhol, minden intézményben egy darab szerver képes ellátni ezeket az összes komponens egy vason fut, és egy szerver szolgálja ki a kéréseket. 2002-ben az Edukáció Kht. auditálta ezt a rendszert és azóta is a másik két nagy rendszerrel szinte párhuzamosan megy át azokon a teszteken, amiket hallhatunk ma, mint például a felsőoktatási információ rendszerhez való kapcsolódás, oklevél melléklet készítés, tehát ez lényegében egy elfogadott rendszer.

**Főbb részei az ETN-nek:** a két nagy rész az a törzsadatoknak a kezelése, a másik pedig az oktatási és tanulmányi adatoknak a kezelése. A törzsadatokban az oktatók, hallgatók és munkatársaknak kezelését értjük, az utóbbi időben ez akkorára növekedett, mint maga az oktatási rész, tehát alapjában ez egy tanulmányi szoftvernek indult, de a felsőoktatási információs rendszer által támasztott követelményeknél ez a törzsadat rendkívül részletes és kiterjedt lett, úgyhogy az oktatók-hallgatók, adatainak rögzítése és az ETN-nel való integrációja az egy testes kis része lett a szoftvernek.

A másik ilyen nagy része az az oktatási és tanulmányi adatoknak a kezelése, a képzések nyilvántartása, a tantervek, tantárgyak, kurzusok nyilvántartása, hallgatói leckeönyvek nyilvántartása, vizsgák koordinálása. Ez ma majdnem minden intézményben használt és futó dolog. Végül is azt lehet mondani, hogy a Linux rendszeridőhöz mérik a maguk dolgait és az ütemezését. Megvan, hogy mikor nyílik a tárgyfelvétel, lezáródik, vizsgafelvétel lezáródik, mikor lehet vizsgát visszamondani. Az is a rendszeridőhöz visszaszámítva órákba. Tehát ez egy jól szabályozott folyamat.

Ezenkívül van egy olyan kisebb része, amit kommunikációnak hívhatunk. Mivel az ETN megköveteli az e-mailes regisztrációt ezért nagyon jól használható körlevelek írására, körleveleken belül az intézmények nagyon szeretik azt, hogy különböző csoportok mondjuk azok, akik bizonyos tárgyat felvettek, azoknak küldenek emailt, vagy azok, akik kint laknak, kollégisták, tehát ezek a körlevelek majdnem mindig ETN-en, az információ áramlás nagyon sokszor az ETN-en keresztül történik.

Létezik egy fórum is, a tanulmányi ügyeknek a megbeszélésére. Ezt akkor használták ki nagyon, amikor a kreditrendszert bevezették és rájöttek a tanulmányi osztályon, hogy a 45. ugyanolyan kérdést felteszik, sokkal jobb, ha a fórumhoz irányítják a hallgatókat és azon keresztül kommunikálnak.

Létezik naptár, hírek megjelenítése és létezik egy tudakozó is, ahol lényegében az intézmény munkatársai, dolgozói, oktatói, hallgatói tehetik közzé az adataikat. A törzsadat, oktatás és kommunikáción kívül léteznek más modulok is, mint ahogy említettem az ETN nem csupán a tanulmányi dolgok adminisztrálására van, hanem kihasználva azt, hogy lényegében beléptető rendszer, ezért nagyon sok minden más feladatot is el tud látni. Ezek közül két intézmény is használja és lehet, hogy most már egy harmadik is az intézményi honlapot. Egyszerűen az ETN-en keresztül töltik fel a saját intézményi honlap tartalmát. Egy menü oldaltérképet lehet az ETN-en keresztül szerkeszteni, és a szerkesztői jogosultságokat ki lehet adni akár oktatóknak akár hallgatóknak, és azon keresztül nagyon egyszerű dolgok, nem kell nagy dolgokra gondolni, a HTML szerkesztést viszonylag egyszerű szövegszerkesztő veszi át. Képfeltöltésére, táblázatbeszúrására, kép nagyítására, kicsinyítésére van lehetőség, színek, betűtípusok változtatására, ezáltal lényegében megszűnik az a sokszor hosszadalmas folyamat, hogy elküldi valaki a honlap szerkesztőjének az anyagot, hogy tegye már fel, és még mindig nincs fent. Ezzel ezt a dolgot ki lehet küszöbölni. Nagy előnye még az intézményi honlapnak, hogy ilyen makrók is vannak benne, ami azt jelenti, hogy az ETN-ből származtatott adatokat is meg tudja jeleníteni. Például egy boxban meg tudja jeleníteni az öt legfrissebb hírt, vagy az ETN-en keresztül bevitt organogramot, egy bizonyos oktatónak a tantárgyait, hogy miket tanít, tehát ilyen kis kapcsos zárójeles makrókkal az intézményi honlapon abszolút friss adatokat az ETN-ből meg lehet mutatni.

Az oktatói profil, az egy viszonylag új modul, létezett előtte is egy ilyen, hogy oktatói honlap csak teljesen átírtuk. Az oktatói profil arra való, hogy az akkreditációs eljárásokhoz is, meg amúgy is, az oktatók be tudják mutatni a saját munkásságukat. Ebbe beletartozik az, hogy önéletrajzot tudnak feltölteni, amit szintén vagy egy ilyen szöveges dobozban megszerkesztik, de egy közvetlen linkkel meg feltöltési lehetőséggel europass önéletrajzokat tudnak szerkeszteni, tehát ez egy ilyen standard önéletrajz bemutató, többnyelvű bemutató lehet. A másik része azonkívül, hogy lehet ilyen kiegészítő információkat és egyebeket megmutatni, ami nagyobb rész és szerintem nagyon fontos, az az, hogy az oktatóknak a publikációs jegyzéke az egy szintén standard formátumban, BibTeX formátumban felvihető az ETN-en keresztül. És nyilván a BibTeX az egy elfogadott szabvány és ehhez többfajta export létezik, tehát ezt a BibTeX formátumot le is tudják tölteni az oktatók, és most döbbenek rá, hogy az OpenOffice-szal már lehet irodalom adatjegyzék bázist létrehozni. Ugye amikor valaki nem tudja azt, hogy „jajj, ezen a cetlin felírtam valamit, de nem tudom a pontos bibliográfiai adatokat.” Hogy ha van egy bibliográfiai adatbázisa és mondjuk meg tudja azt találni BibTeX formátumban, akkor elvileg a bibliográfiai adatbázis frissítéssel egyszerűen be tudja szűrni a hivatkozást és a végén ugyanúgy, mint egy tartalomjegyzéket, egy irodalomjegyzéket tud készíteni. A BibTeX az abszolút alkalmas erre, főleg, hogy ha valaki LaTeXben dolgozik akkor az meg egy alapvető, magától értetődő rész a BibTeX. Ez azért fontos, mert maguk az oktatók is, amikor megmutatják a saját publikációs jegyzéküket, már egy elfogadott szabványba tudják menteni és mások le tudják tölteni ezeket a bibliográfiai adatokat és használni tudják. Sőt ugye béta verzióban van most a Google tudós, a scholar.google.hu, ami arra szolgál, hogy tudományos publikációs jegyzéket vagy adatbázist állít össze a Google, ahol az idézettséget is mutatja, tehát ez is egy nagyon fontos a kutatói munkában, és ez az oktatói profil egy BibTeX formátumot készítve, lehetővé tudja azt tenni, hogy a Google robot beszipantsa azokat az adatokat, és egyből a Google

tudásban meg tudja jeleníteni. Tehát ezért is tetszik az oktatóknak ez, hogy a Google-ben, hogy lehetővé válik ezeknek az adatoknak megjelenítése.

**Hallgató:** a LaTeX mennyire elterjedt az oktatók körében? Az hogy a természettudományos területeken, kvázi a matematikában semmi más nem lehet, a természettudományos területekbe általában mondom azt, hogy van jobb szabvány. A LaTeX így bölcsész tudományok, egyéb tudományok, mennyire elterjedt, kedvelt, használt?

**Előadó:** Ugye a JabRef az egy olyan BibTeX szerkesztő, ami tud többet, exportot is pl. MODS-t vagy RTF vagy HTML exportokat, sőt tud OpenOffice táblázat exportokat is, tehát a régít is, meg az újat is, Microsoft Office-t is. Tehát a LaTeX az nem nagyon elterjedt a bölcsész területen, viszont mivel itt más exportot is lehet, inkább ezeket használják. Azt azért elmondhatom, hogy én például tartottam előadást bölcsészeknek LaTeX-ről. Nyelvészeknek, akik el voltak ájulva, hogy végre meg tudják különböztetni a kötőjeleket, meg el tudják választani a karórát, hogy kar-óra vagy ka-róra, mert az két különböző elválasztás meg nagyon sok olyan lingvisztikai eszközt, fa rajzolást mutatott nekik, ami nagyon tetszett nekik. Például a Károli Egyetemen most kerestek meg nemrég. Akarnak egy terminológia szakot beindítani a nyelvészek, amiben nagy igény van arra, hogy egy precíz kiadványszerkesztőt tudjanak használni és meg tudják valósítani egyébként azokat a dolgokat, amiket elméletben tudnak, hogy hogyan kellene nyomdailag vagy tipográfiailag helyesen megjeleníteni, de erre nem mindig volt, van eszközük. Nagy küzdelem volt egyébként Linuxokat felrakni, felrakatni a gépterembe, de egyébként teljesen normális bölcsész hallgatók jöttek, és így megszokták, és tetszett nekik.

**Hallgató:** Esetleg, aki nem ismerné a LaTeX-et, meg ezt az egész dolgot, annak mondjuk már el, hogy az az érdekessége neki, hogy tulajdonképpen nem kell sokkal nagyobb szakértelem hozzá, mint egy sima szövegszerkesztőhöz, viszont az eredmény az ezerszer jobb és szebb, mivelhogy egyből nyomdakész. Ezt egyből el lehet küldeni a nyomdába és a nyomdagépek egyből egy szép terméket tudnak belőle. Ami teljesen nem igaz az általános szövegszerkesztőre. Tehát soha nem lesz igaz sem egy OpenOffice-ra, sem egy Microsoft Office-ra, azok nem lesznek szépek, bármennyire szomorúan hangzik is.

**Előadó:** Vagy azt hisszük, hogy szépek, meg mindent meg tudunk csinálni, csak tipográfiailag helytelenek. A LaTeX meg egyszerűen a tipográfus szerepét átvállalja és kikényszeríti a szerkesztőtől a szövegstruktúrát. Egy rendkívül jó szoftverről van szó, és ilyen WYSIWYG, tehát ahogy szerkesztem azt látom körülbelül. Ilyen interfész is létezik, a LyX vagy nem tudom, hogy kell kiejteni, ami nagyon nagy segítséget adhat. Na, az oktatói profil próbálja ebbe az irányba terelni a bölcsész akadémiai oktató gárdát is.

**Irattár:** az semmi más, mint mappára felosztott dokumentum-feltöltés, persze írási és olvasási jogosultságok szerkesztésével, attól függően, hogy ki milyen csoportba tartozik, mert minden felhasználót ilyen linux szemlélettel csoportokba lehet rendezni.

**Kérdőív:** a kérdőívet is többen használják, egyszerűen kérdőív szerkeszthető az ETN-n keresztül. Meg lehet adni, hogy a kérdőívnek mi a tárgya, mondjuk nincs tárgya, mindenkinek egy kérdőívet kell kitölteni, vagy lehet a tárgya, amire ki kell tölteni a kérdőívet mondjuk egy személy, tehát az oktatókra vagy a hallgatókra, meg lehetnek tantárgyak illetve kurzusok is. Tehát minden kurzusra külön ívet lehet kitölteni. A kérdéseket pedig viszonylag egyszerűen lehet szerkeszteni. 4 fajta kérdés van: egyszerű választásos, többszörös választásos, szám vagy szöveges mező és ez alapján a kérdést megadva a megfelelő form jelenik meg, amivel ki lehet tölteni ezeket a kérdőíveket. Nagy előnye az, hogy miután kitöltötték egy gombnyomással ez a statisztikát el is készíti, tehát nem kell ezzel foglalkozni. És az anonimitás is garantált ebbe a kérdőív részben. Többen használják, szeretik. Ma is megdicsérte az egyik intézmény .

A hallgatói gazdasági modul az egy olyan modul, azt egy intézmény használja csak, ami arra szolgál hogy a bonyolult kredit index, korrigált kredit index alapján szétossza a pénzt, ami a hallgatóknak jár, és a szétosztott pénzt, a tanulmányi ösztöndíjat a szociális támogatást meg

akármilyen juttatást havonta kimutassa, egy táblázatban elkészítse és ami nagyon fontos, hogy egy edifach nevezetű formátumba exportálja, amit az intézmény betesz egy PC bankár, azt hiszem, így hívják a szoftvert, és egyszerűen átutalja a pénzt a hallgatóknak. Tehát ez egy gazdasági modulja az ETN-nek.

A beléptető-naplózó rész azt egyetlen egy intézmény kérte, az inkább adminisztrációs dolog. Arra szolgál, hogy az ETN-ben lévő felhasználó és jelszóval tudjanak belépni a hallgatók mind a kollégiumi szobákba mind a géptermekekben a terminálokra. És azt az internetes tevékenységet, amit csinálnak, azt a rendszer naplózza személyre szabottan. Na, ez elég rosszul hangzik, tehát egy ilyen Big Brotheres a dolog. Nyilván meg voltak a maga problémái az intézménynek ezzel kapcsolatban, tehát rendőrségi ügyig ment már a dolog, és egy jogi nyilatkozattal a hallgatókkal ezt aláírták, hogy ez oktatási célra van. Az internet ezt naplózza és ezt valósítja meg ez a beléptető-naplózó rendszer a Samba/iptables segítségével. Ezekről majd később, ha lesz idő még, majd beszélek.

A következő listák és statisztikák. Tehát különböző listákat és statisztikákat tud készíteni. Ezek között, ami úgy kiemelkedik az a dinamikus dokumentum szerkesztés, amiben egyrészt formázási lehetőségek vannak, abban a bizonyos egyszerű szövegszerkesztőben, HTML dokumentumokról van szó, és ebbe makrókat lehet beilleszteni az ETN-ből vett adatokkal. Tehát adóigazolást, hallgatói jogviszony igazolást, meg ilyeneket maguk szerkeszthetnek az intézmények és egy gombnyomással ezeket az adatokat meg is tudják jeleníteni. Nyílt forráskódú nyilván, tehát a PHP nem is ad más lehetőséget, minthogy ez legyen. Olcsók, jóval olcsóbbak, tehát maga a fenntartásról beszélek, a karbantartásról, mint a .NET-es társai. Megbízhatóságra nem lehet panaszkodni. 2001 óta nem volt olyan, hogy adatok tűntek volna el, vagy bármi probléma lett volna ezekkel. Hát nyilván nem az ETN érdeme, hanem a mögötte levő szerver technikán, a PostgreSQL, az Apache, a PHP, ezek nagyon jó háttérrel biztosítanak erre, és intézményi részről is fejleszhető. Most itt az alsón, nem tudom mennyire látszik, de megváltoztatták a Pető Intézetben a skinjét az egésznek és hozzáraknak, elvesznek belőle. Tehát a PHP erre lehetőséget ad. Negatívuma ennek a PHP-nak ugye az, hogy ez a szkript nyelv, tehát futási időben fordítja a szerver az utasításokat és ez azt eredményezi, hogy bizonyos nagyobb számításoknál, mondjuk egy egész évfolyam korrigált kreditindex, kreditindex súlyozott tanulmányi átlagát valós időben szeretném megjeleníteni, akkor ott bizony az execution-t, a végrehajtási időt fel kell venni mondjuk 4 percre, hogy megjelenjen. Tehát ez egy hátránya. Nyilván ezen dolgozunk, hogy próbáljuk leosztani a feladatokat, és nem hagyunk egyszerre nagy dolgot futni. Tehát annyira nagy probléma ezekkel nincs, kis intézményekről van szó, ez azt jelenti, hogy 2000 hallgatónál kevesebb jár oda. Nincs emiatt különösebb gond még az ilyen nagyobb a számításoknál is, de lehetne. Szóval ezek a performance problémák, ezekkel mindig hadilábon állunk.

**A kerete az ETN-nek.** Itt látszik, hogy hogy néz ki egy ETN felület. Az egész webes kliensű, tehát mind az adminisztrátori rész, mind a hallgatói, mind az oktatói rész kezdetektől fogva webes kliensű, tehát nem kellett külön kliens programokat letölteni. Csoportos jogosultságon alapul a használat. Jobb oldalon mindig van egy online súgó, ami megmutatja, hogy éppen ahol jár valaki, ott mit tehet vagy segítséget ad neki. Létezik egy tájékoztató felület nem ETN felhasználóknak is. Ezek a kis intézmények nem akartak például nagy vagy pontos hitelesítési eljárásokat kidolgoztatni, viszont szerették volna, hogy ha valami adatuk van és az publikus, megjelenjen. Tehát ez egy tipikusan, ez a szoftver úgy alakult, ahogy a kis intézmény igényei megjelentek, nem bánták, hogy ha lassabb egy kicsit az egész, de szerették azt, hogy ha valamit felvisznek, az meg is tudjon jelenni mondjuk angolul is, ha külföldön akarják megnézni vagy egy hallgató kimegy külföldre, hogy mit tanult. Tehát az elejétől fogva a tárgyleírások, azok magyarul és angolul is megjelennek, egy kis zászlóval, magyar-angollal. A tájékoztató felület az változtatható. Az egésznek a beléptetése az ilyen portárs stílusú, e-mailes autentikáció, regisztrációval működik, aminek a nagy előnye az, hogy ezek a körlevelek működnek. Tehát nem arról van szó, hogy valaki olyan e-mail címet ad meg, ami nem létezik, hanem arra kapja a jelszavát vissza. Ez ebben a portál-stílusban nem régóta van csak másfél éve, de nagyon bevált.

Nyomatatási kép ugye keretmentesen és ami nagyon fontos, hogy aláírt oldalt is tud produkálni. Ez azért lényeges, mert a hallgató azt mondja, hogy én úgy láttam, az ETN-ben, hogy felvettem a tárgyat és most pedig eltűnt, ezt be tudja bizonyítani egy egyszerű PGP aláírást tesz arra az oldalra, a nyomtatott oldalra, a hallgató azt el tudja menteni és be tudja mutatni, hogy hát neki ilyen van. Amióta ez a PGP aláírás van, a hallgatóknak nincsenek ilyen problémái, hogy így látták.

**A törzsadatoknál az egy nagy kihívás volt, a FIR-rel kapcsolatban, Felsőoktatási Információs Rendszerhez igazodjunk.** A FIR az maga egy IBM WebSphere message queue rendszerben működik, amit úgy kell elképzelni, mint egy hivatalt, hogy valaki, egy intézmény elküld egy kérelmet, aláírt kérelmet, egy digitálisan aláírt kérelmet, ők azt átveszik feldolgozásra és visszaküldik, hogy ők most azt elfogadták, hiba van vele, vagy mi a helyzet. Azt, hogy milyen adatokat küldhetünk el, azt egy szótár határozza meg, amit meg egyébként ők küldenek szintén XML-be. Tehát ez egy eléggé bonyolult folyamat, arról van szó, hogy a PostgreSQL, az egy SQL adatbázis, az egész kommunikáció egy XML alapú, tehát az összes üzenet XML alapú, az aláírt is XML alapú. Amikor adatot küldünk az egy SQL XML export, amikor adat jön, akkor XML SQL exportról van szó, hogy ők mit csinálnak arról csak sejtéseink vannak, de a lényeg az, hogy valószínűleg SQL-ben vannak ők is. Tehát az XML a kommunikáció alapja, és úgy néz ki a dolog, hogy ha belép egy hallgató vagy egy oktató az ETN-be, akkor a törzsadatait látja. Látja azt, hogy a FIR-ben milyen adatok tárolandók vagy tároltak, és ezeket, ha valami problémája van a törzsadatokkal, akkor itt kérelmezheti az adatváltoztatást.

Ez azt jelenti, hogy egészen odáig engedi az ETN az adatrögzítést, ameddig nem rögzíti az adatokat, tehát nagyon fontos, hogy olyan sok formai ellenőrzés megy végbe az adatok megadásánál például legördülő sávban, legördülő menüben jelennek meg a nyelvvizsga központok, amikor valaki nyelvvizsgát akar bevinni. Vagy megkapjuk a Felsőoktatási Információs Rendszertől a KSH adatbázist, tehát olyan felesleges dolgot is tudunk, hogy például nemcsak Magyarország összes helyiségét, annak az irányítószámát, hanem a lakos számot, meg hogy hány ház van abban a helyiségben. Tehát egyszerűen a település azonosítóval kell küldeni például a címeket. Tehát ezek nagyon fontos ellenőrzési folyamatok amiket maga a felhasználó beviszi, ezek megtörténnek csak a rögzítés helyett egy kérelmet küld a FIR adminisztrátorhoz, aki azt a kérelmet jóváhagyhatja, módosíthatja vagy törölheti. Nyilván, ha olyan kérelemről van szó, ami igazolást igényel, tehát mondjuk nyelvvizsgapapír a nyelvvizsgához, akkor azokat bekérheti és azokat rögzítheti az ETN-ben, és amikor FIR csomag vagy konténerküldésről van szó, akkor ezeket az adatokat vagy adatváltoztatásokat berakja egy konténerbe.

Itt jön egy újabb állomás, hogy ezeket alá kell írnia digitális aláírással, biztos ismerik sokan az edukációnak az aláíró kártyáját, a tanúsítványát, most már az közhiteles. Itt is egy olyan alkalmazást használunk, amit nem mi fejlesztettünk ki, FIRA-nak hívják az XML aláírására használható, a Polisis Kft. csinálta. Ez egy Java alapú alkalmazás, de nagyon jól együttműködik az ETN-nel. Egyszerűen átadja az aláírt XML-t, ott aláírja kártyával az adminisztrátor és visszatölti az aláírt XML-t és utána küldi el szintén Java alkalmazással az ETN ezeket az adatokat. A FIR két, tehát ez a WebSphere queue két fajta klienst ismer, a .NET-est és a Javast, tehát a három közül mi vagyunk akik Javast használunk a másik kettő .NET-est. Egyszer volt egy nagyon érdekes problémánk, arról később esetleg beszélek, de egyébként ez a Java alapú alkalmazás nagyon jól működik. És jól tudjuk exportálni meg követni az adatokat. Ha elküldjük az adatokat akkor az üzenetek bekerülnek a FIR-be, és a felhasználó oktató, hallgató az oktatási azonosítója alapján a magyarorszag.hu-n elvileg meg tudja nézni, ha van ügyfélkapus hozzáférése, hogy milyen adatokat tartanak róla nyilván. Tehát látja az ETN-ben, ahol módosítja, az adminisztrátor, FIR és ő maga ellenőrizni tudja a rá vonatkozó adatokat természetesen a magyarorszag.hu portálon keresztül.

Az ETN üzenetváltási filozófiája az, hogy minden, tehát az adminisztrátor is csinálja úgy, ahogy szokta, tehát módosít, töröl, beszúr adatokat és a végén ez egy nagyon szabályozott üzenetváltási rendszer, tehát külön üzenettípusa van a hallgatói jogviszony létrehozásának, a dátum módosításnak, a karbantartásnak, amit megpróbálunk, ahogy lehet elrejteni az intézményi felhasználók előtt tehát ez úgy néz ki, hogy használja az ETN-t és amikor rákattint arra, hogy FIR

konténer küldés, akkor végigpásztázza az ETN azokat, hogy miket és milyen típusú üzeneteket kell küldeni, azokat egy csoportba szedi és akkor automatikusan generálja ezeket az üzeneteket át és ugyanígy a viszont válasz is úgy néz ki, hogyha jönnek válaszok, azokat beépíti az ETN-be, tehát hogyha sikeres, akkor az ETN-be jóváhagyja azt az adatot, ha nem, akkor ott módosításra kötelezi. Ha adatterítés érkezik, akkor azt beszippantja. Ez azért fontos, mert aszinkron a kapcsolatunk, tehát az ETN-ben minden rekordnak van egy ilyen FIR státusza hogy az a FIR-ben ugyanúgy néz ki, az az oké, ez a legmegnyugtatóbb, a FIR-be létre kell hozni, a létrehozás alatt van ugye. Két nap a hitelesítési igazolása, 48 óra oda-vissza tehát addig például elküldött adatokat nem szabad módosítani, törölni vagy törölt adatokról van szó ilyen státusz alapján az ETN ezeket az adatokat kezeli és ezekkel a státusz változtatásokkal szabályozza.

**Oktatás:** Amíg a FIR egy nagyon jól...ez nem igaz..., de mindenesetre pontosan szabályozott specifikációja van, hogy hogyan kell adatokat küldeni, addig az oktatásban a Neptun, az ETR meg az ETN is azt csinál, lényegében, amit akar, tehát ott nincs egy ilyen specifikáció, hogy például mit jelent az, hogy kurzus. Vagy mit jelent az, hogy tantárgy. Így aztán nemrég volt egy ETR konferencia Szegeden, és ott direkt azért is hívtak meg, hogy erről elkezdjünk beszélni, mert mintatanárgy, mintaelem, elem, al-elem, ösvény, doboz, tehát olyan metaforák vannak a felsőoktatásban az oktatás struktúrájának a kifejezésére, amik rendszerenként, sőt intézményenként változnak. Mindenki mást ért ezalatt. Az ETN-nek is van egy ilyenje, eddig azt tudom mondani, hogy minden nyakatekert és különleges igényt sikerült modellezni ezzel a struktúrával, nyilván ez is hosszú idő, két éve működik elég hosszú idő kellett ahhoz, hogy ez letisztuljon. Nem az, hogy bonyolódjon, hanem, hogy letisztuljon valamennyire.

A kurzus az amivel végül is a hallgatók találkoznak. Kurzust vesznek fel egy félévben. Azt jelenti, hogy egymás után valamilyen előadást, anyagot hallgatnak tehát a kurzusok azok, olyan entitások, hogy a tananyagok a mikéntjéért, hogy ki tanítja, mit tanít, hogyan tanít és mikor tanít tartalmazza. A tantárgy az eggyel elvontabb szint, mint a kurzus ugyanúgy néz ki a táblázat, mint a kurzusba, kivéve azt, hogy nincs hozzárendelve terem, meg nincs hozzárendelve órarendi időpont, meg nincs hozzárendelve félév. A tantárgy a mit? kérdésre próbál válaszolni, mint ahogy a nevében is benne van. Ezeknek a manifesztálódásai lényegében a kurzusok azzal, hogy újra öröklődnek azok a tantárgyban lévő adatok és hozzákerülnek még az órarendi és a félévhez kapcsolódó adatok. A tantárgyak utána a következő szint az a minta tanterv, a dobozos mintatanterv, amiben megpróbáljuk szabályozni a kötelezőséget, a kötelezően választhatóságot meg a szabadon választhatóságot. Egy kurzusleírás például így néz ki, angolul is elérhető, kódja van, neve van, kreditje, meghirdetési gyakorisága, milyen modulba tartozik, szemeszter stb. Ezt mindenki láthatja. Természetesen ez magyarul ugyanígy megvan.

**Ami izgalmas, az a dobozos rendszer.** 4 illetve 5-féle doboz van,  $2 \times 2 + 1$ , vannak egyszerű dobozok meg összetett dobozok, és vannak kötelezőek meg választhatók és ezeknek a kombinációi. Az egyszerű kötelező doboz egy olyan Legó elem, vagy olyan keret, amely minden doboznak van egy kredit értéke, ez azt jelenti, hogy mondjuk egy 180 kredites tantárgy létezik. Abból hogy ha egy 20 kredites dobozt teljesít valaki, akkor a 20 kreditje megvan. Ha az egyszerű kötelező doboznak az értéke 20 kredit, az azt jelenti, hogy abban a dobozban 20 kreditnyi tantárgy van felsorolva és akkor teljesíti a dobozt, ha mind a 20-at teljesíti. Azért egyszerű, mert tantárgyak vannak benne és azért kötelező, mert pontosan annyi kreditnyi tárgy van benne, mint a doboznak a kreditértéke. Az egyszerű választható doboz annyiban különbözik ettől, hogy annak is van egy kreditértéke, mondjuk 30, és abban is tárgyak vannak, de 30 vagy annál több kreditnyi tárgy kell, hogy benne legyen, mondjuk 45. Akkor teljesíti az egyszerű választható dobozt a hallgató, ha a 45 kreditnyi tantárgyból 20-at teljesít. Az összetett kötelező doboz, és ez a nagyszerű vagy ezért lehet nagyon sok dolgot megoldani vele, az összetett kötelező doboz az dobozokból áll. Hogy ha egy összetett kötelező doboz 20 kreditnyi, az állhat két 10 kredites dobozból, akkor teljesíti valaki az összetett kötelező dobozt, hogyha mind a két dobozt teljesíti, összesen 20 kreditnyit teljesít. Az összetett választható doboz pedig azt jelezni, hogyha egy doboznak a kreditértéke, egy összetett választható

doboznak a kreditértéke 15 kredit, akkor akárhány, mondjuk 3 vagy 4 doboz van benne, de annak a kreditértéke mind 15 kreditnyi, és akkor teljesíti valaki ezt az összetett választható dobozt, ha az egyik benne lévő dobozt teljesíti. A szabadon választható pedig olyan, mint egy kosár, semmi mása nincs, mint kreditértéke, mondjuk 8 kredit. Minden olyan tárgyat oda fog belegyűjteni vagy olyan kreditet, ami a többi dobozból kimaradt. Itt van például egy részlet: ez egy szabad vallás bölcsész tudományi szak, az ókori nyelvek az egy 12 kredites doboz, van héber, görög 12 kreditnyi. Itt látszik, hogy egy nagyobb dobozban van benne, ami szintén dobozokban van benne. Azért jó ez, mert az összetett dobozokba akárhány dobozt, mint egy matrioska babát, akárhány dobozt lehet belerakni, és ezért sikerül lényegében minden tantervi igényt lefedni.

**A kurzusfelvétel:** időponthoz kötött, vizsgák számát nézi stb., erős és gyenge előfeltétel megszabható. Az erős az azt jelenti, hogy akkor veheti fel valaki a kurzust, ha már teljesített bizonyos kurzusokat, a gyenge pedig az hogy vagy teljesítette már, vagy éppen akkor veszi fel és egy egész boole algebra létezik az előfeltétel rendszerre. Egyszerűen az ÉS, VAGY, meg a NEM (negáció) is szerepelhet a feltételek megadásánál, meg a zárójel is, ez egy fontos. Nagyon egyszerű trükkkel csináltuk ezt meg és nagyon jól alkalmazható. Behelyettesítettük a kódokat, ugye kódokkal kell megadni az előfeltételeket, IGEN-re vagy NEM-re, true vagy false. És kilépünk a PHP-be és egy PHP-t futtatunk, amikor ezt a kifejezést kiértékeljük így minden, ami a PHP-ban alkalmazható logikai operátor szerepelhet ebben a boole algebrában, tehát a PHP kifuttat egy PHP-t és az értékét visszakapjuk, hogy az igaz vagy hamis és így tudjuk meg, hogy a hallgató felveheti-e a tárgyat vagy sem. Így néz ki a főmenü: törzsadatok, oktatás, intézmény, FIR kommunikáció és ott vannak a modulok, meg a listák. A kiegészítő modulok, azok pl. az irattár, struktúrája így néz ki. Ez pedig az oktatói profil, itt lehet különböző BibTeX-be vagy más formátumban exportálni a dolgokat.

**Hallgató:** az erős és gyenge előfeltételek ellenőrzése rögtön akkor történik meg, amikor a hallgató felveszi a tantárgyat? És nem is engedi felvenni? Mert ETR-nél ezt ki lehet játszani.

**Előadó:** láthatod ezt a táblázatot, nem tudom, mennyire olvasható, post-control típusú ez az ellenőrzés, tehát az összes felvehető tárgy megjelenik a hallgatónak akkor rákattint, hogy felvesz és akkor történik meg az ellenőrzés. Nyilván ez azért van, hogy amíg valaki böngészik, addig ne tudjon betelni a létszám. Tehát abban a pillanatban történik meg mindez. És így van, a feltétel akkor kerül kiértékelésre, tehát akkor, amikor rákattint, akkor nézi meg, hogy megvannak-e a feltételei. Kilép, futtat egy PHP-t zárójelezve és visszalép. És hogy ha mindegyik oké, akkor engedi meg, hogy felvegye a tárgyat. Tehát ilyeneket lehet, hogy időben történt-e, belefér-e a maximális létszámba ugye nem teljesítette, ha már teljesítette, akkor megint nem engedi felvenni, felvett-e már ilyen kódú tantárgyat vagy kurzust ebben a félévben, ha már felvett, akkor megint nem engedi. Feltétel pont itt nincs, de ha itt a feltételeknél szintén ez történik, hány vizsgát lehet összesen ebből a tárgyból tenni. Tehát azt is megnézi, hogy abba belefér-e. Nincs-e abban a tanblokkolt csoportban, akiknek egyszerűen nem engedik meg a kurzusfelvételt. Tehát igen, a régi ETN-ben úgy volt, hogy ki lehetett pipálni, hogy ezeket a tárgyakat akarom felvenni és akkor egyben felvette. És ott már a megjelenéskor ellenőrizte. Itt az volt a baj, hogy tényleg betelhetett a létszám, úgyhogy egy-két hallgatóval több volt, úgyhogy változtattunk arra, hogy a tárgyakat egyenként kell felvenni, és az egyenként való felvételnél történik meg ez a fajta ellenőrzés és ez kemény. Ha nem engedi, akkor nem engedi.

**Hallgató:** ezt a rendszert ki fejleszti? Ki a fő fejlesztője?

**Előadó:** én vagyok a fő fejlesztője, de nem egyedül csinálom. Tehát vannak, akik bizonyos dolgokban segítenek. A Codosoft Informatikai és Tanulmányi Bt.

**Hallgató:** és egy nagyobb egyetemet is ki tudnak szolgálni?

**Előadó:** ahogy mondtam ez a PHP szkript az egyetlen, ahol okosan kellene szétosztani a feladatokat, erre vannak, volt már nagy egyetemmel kísérlet, ez ilyen 4000 fős és meglepő módon nem fagyott le a PostgreSQL, nem voltak problémák a PHP-val, tehát a tárgyakat így fel tudták

venni a hallgatók. Tehát azért a PostgreSQL, PHP, Apache 3-as azért elég nagy dolgokat ki tud szolgálni. Tehát ennél jóval nagyobb dolgokat az üzleti világban, úgyhogy itt sincs ilyesmivel gond vagy nem volt a teszteléskor.

**Hallgató:** tehát igazából ezeket a performancia problémákat, igen ezt én is úgy gondolom, hogy PHP, Apache, PostgreSQL, MySQL, Linux itt egész óriási weblapokat, webshopokat, mindeneket elvisz. Nem hiszem, hogy informatikában, adatbázis kezelésben néhány ezer adatrekord gond...az semmi, az nem sok.

**Előadó:** a feltétel kiértékelés az azért durva, nem az adatbázis részen van a gond, hanem mikor a PHP-nak el kell kezdeni számolgatni adatbázisban.

**Hallgató:** és hogy ha egyszerre sok ember próbál felvenni tantárgyat és mindenkinek kiírja, hogy felveheti-e ugye akkor... Most köztünk szólva egész nagy tanulmányi rendszerekről is hallottam már, hogy adott esetben, mikor a hallgatók vették fel a tantárgyakat napokig nem volt elérhető.

**Előadó:** ez nem fordult még elő az ETN-nel ez.

**Hallgató folyt.:** tehát a legnagyobb a Neptunnal, én úgy tudom, hogy minden intézményben, minden félévben legalább egyszer előfordul. Legalábbis nekem egészen hajmeresztő tapasztalataim vannak ezzel. Tehát ahhoz képest nem hiszem, hogy ez annyira kritikus lenne ebben a rendszerben.

**Előadó:** nem. Tehát itt arról van szó, hogy azonnal vagy kell 10 másodpercet várni. Tehát itt erről a kategóriáról beszélünk.

**Hallgató:** mégiscsak válság téma. Gazdaságilag, anyagilag hogy éri ez meg egy egyetemnek, egy főiskolának, hogy adott esetben egy szabad szoftveres alkalmazást használ? Nyilván gondolom, nem ismerem az üzleti modellt, szeretnék rákérdezni. Hogy az üzleti modell az valami olyasmi, hogy maga a termékek azok nyilván ingyenesek, mivel szabad szoftverek, és a terméktámogatásért kell az intézménynek fizetni illetve tekintve, hogy ez egy jól bevált rendszer, tehát egy Linux mondanám, hogy LAMP, de az m betű helyett p, tehát LAPP, Linux, Apache, PostgreSQL, PHP, tehát mondhatnám azt nyilván. Tehát előfeltételezéssel élek: sokkal kisebb erőforrás igénye van ennek, mint egy .NET-es rendszernek. Ez teljesen biztos. Tehát azt szeretném megkérdezni, hogy gazdaságilag hogy működik ez egy főiskolán, egyetemen, ha ezt ezzel az eszközzel akarja megoldani. Van-e itt megtakarítási lehetősége?

**Előadó:** tehát a kitörési pont az akkor történt meg, amikor egy illetve két intézmény helyett 4 -5 kezdte el használni, aztán most már nyolc. Egy olyan intézmény sem volt a történetünkben, aki visszamondta volna. Tehát, aki váltott volna ETN-ről. Úgy történt a kitörés, hogy egy bizonyos tanulmányi rendszernek nem szabad szoftveres tanulmányi rendszernek az adatbázis szervere Oracle volt. És elfelejtették kifizetni a licence díjat, az Oracle licence díjat, amire a kormány minden intézménynek támogatási összeget adott. Nem olyan nagyról van szó, tehát ilyen 2 millió forint nagyságú, tehát hogy egyszerűen a licence díjat fizessék ki és mivel az ETN-nek nincs PostgreSQL licence díja, így hirtelen rájöttek, hát hogy akkor ez máskor is probléma lehet és ebből az összegből tudták ezt a karbantartási dolgot fedezni. Ez a karbantartási összeg ez 50.000. Forintos karbantartási havi karbantartási összegről van szó, amibe nem úgy tartozik bele a karbantartás, hogy mondjuk valakinek változtatnia kell egy adatbázison, akkor táblázatonként változtatunk, hanem a karbantartás az így hangzik: hogy minden, ami SQL utasítással megoldható, akárhány karbantartási kérés van, azt ebben az átalánydíjas karbantartási összegben csináljuk, tehát nem táblázatonként, kérésenként kell fizetni. Akkor kell, tehát akkor van fejlesztésről szó és a fejlesztés is úgy történik, hogy az intézmények összeállnak és elhatározzák, szavaznak egyszerűen, hogy melyik modult szeretnék, ha fejlesztésre kerülne. És a karbantartás az pedig így történik, hogy egyszerűen, ami SQL-lel megoldható az ebben az átalánydíjas karbantartásban benne foglaltatik. Vannak intézmények, akiknek nincs saját szerverük, akkor ott a hostolási díj még feljön rá. Tehát ilyesmi.

**Hallgató:** ez igazából azért érdekes szerintem, mert mikor egy intézmény, bármilyen intézmény akar venni egy szakalkalmazást, hívjuk ezt szakalkalmazásnak, akkor ugye hogy kezdődik a történet? A kereskedelmi szoftverek gyártói általában azt mondják, hogy hát igen ám, de az én termékem az fut egy Windows 2003 szerveren, meg kell hozzá egy Oracle adatbázis és akkor már elköltöttünk 1 millió forintnál többet és még egy fia szakalkalmazásunk nincs. Valójában még sehol sem tartunk. Majd csak innentől jön az. Még egy fontosnak érzett kérdés: migrálni, tehát vannak előírások az adatszerkezeten vagy nincsenek? Tehát mondjuk egy főiskola, egyetem azt mondja, hogy szeretné migrálni, szeretne átlépni erre, de a régi adatait, évekre visszamenőleg nyilván nem akarja veszni hagyni. Ezt a migrációt meg lehet-e oldani fájdalommentesen vagy kis fájdalommal? Én azt hiszem, hogy a migráció az egyik legborzalmasabb dolog az informatikai projekteknél. Ahol sok véráldozat szokott esni.

**Előadó:** legutóbb az ÁTF tért át ETN-re, akik már webmodulon keresztül a FIR-be bejelentették az adataikat. És azt csináltuk, hogy vannak ezek a szabványok, tehát ez a FIR XML szabvány. Elsőként voltunk azok, akik adatlekérdezést tudtunk csinálni, mert rákényszerített bennünket az ÁTF. És azt csináltuk, hogy egyszerűen ebből az adatbázisból lekérdeztük az adatokat és beépítettük az ETN-be, és most mi vagyunk a gonoszok, mert az adatlekérdezésből...nem tudom ezt fel lehet-e venni?...rájöttünk arra, hogy olyan triggerek vannak a FIR-ben, adatmódosító triggerek, aki tudja, hogy mi az, hogy trigger, az borzong is tőle, hogy egyszerűen bejelentenek aláírt digitális adatokat az intézmények, és a FIR-ben megváltoznak ezek az adatok. És amikor lekérdeztük vissza az adatokat, ott döbrentünk rá, hogy dugig vannak adatmódosító triggerekkel, amiket nem ismerünk. Úgyhogy most most a fejlesztők közül, a Neptun, ETR fejlesztők közül mi vagyunk, akik nagyon piszkáljuk a dolgot és a Neptun, ETR csodálkozva néz, mert még nincs meg ez az adatlekérdező mechanizmus náluk flottul. Tehát élesben igazán ezt nem látják, egyszer ők is fáznak azoktól, amiket látnak. De hát reméljük, hogy ezek változnak. Az Excel táblázat a másik. Tehát, ha Excel táblázatba ki tudják nyerni, akkor nyilván Excelből viszonylag jól lehet csinálni.

**Hallgató:** és akkor ez gyakorlatilag azt jelenti, hogy ennek a szférának egy általános szabadsága van. Azért hála ezeknek a központi megkötéseknek, XML-eknek, hiszen akkor gyakorlatilag bármikor tudnának váltani és jó eséllyel.

**Előadó:** a tanulmányira még nincs ilyen megkötés, a törzsadatokra van. Valószínűleg a tanulmányira is lesz egy ilyen.

## Erdélyi Gábor: E-learning keretrendszer I

Kétoldalú a dolog. Jelenlegi munkáltatómnak az egészségügyi szakképző és továbbképző intézetnek köszönöm, hogy itt lehetek, merthogy munkaidő van most elméletileg, de hát jó voltukból itt vagyok és beszélhetek erről a témáról. Illetve az ő rendszerükön keresztül fogom bemutatni élőben ezt az ILIAS keretrendszert. Illetve a másik oldalról gyakorlatban, én privátban is ezzel foglalkozom, ilyen e-learning rendszerek bevezetésével, telepítésével, menedzselésével. Valami informatikai szlenget is benne hagyhatok, de oktatásról bővebben, módszertanról is tudunk beszélni. Egy picit távolabbról fogom megközelíteni a témát, de ígérem, ez rövid lesz. Csak azért kezdem így, hogy valami felső nézetünk legyen erről az egész e-learning, távoktatás témaköréről.

**A lifelong learningről már biztos mindenki hallott**, a csapból is ez folyik már egy jó ideje, csak felírok rá egy definíciót, hogy mégis körülbelül ugyanarról beszéljünk. Talán a lifelong learningről érdemes két szót mondani ez a fogalom gyönyörű, szép, új, de az igény az nem új. Ugye ismerjük a „jó pap is holtig tanul” mondást. Tehát már régóta van ez az igény, de vannak olyan tényezők, melyek egyre inkább generálják, például az információ feleződési ideje, a használhatóságnak a feleződési ideje és az információrobbanás ugye összefüggő témakörök. Az informatikába is, ha belegondolunk, mennyi új információ jelenik meg a szakterületen, melyekkel nekünk lépést kell tartani, amit követni kell. Ezzel párhuzamosan régebbi ismereteink milyen ütemben avulnak el, ez azt fogja generálni, hogy nekünk folyamatosan, folyamatosan tanulni kell, ha ott szeretnénk maradni a szerten és normálisan szeretnénk végezni a munkánkat. Érdekes, ehhez kapcsolódó kérdés ez, mélységeibe nem fogunk belemenni, ez a munkanélküliség felfelé csúszása. Ha visszagondolunk, a szocializmusban ott egy érettségi az egy olyan dokumentum volt, ami biztosította azt, hogy énnekem egy jó munkám lesz, jó esélyt adott arra, hogy egy megfelelő helyen könnyen el tudjak helyezkedni. Hát ma már a diplomás munkanélküliség is egyre gyakrabban emlegetett fogalom. Tehát hiába megyünk egyre magasabb szintre a tanulmányokban, az még nem feltétlenül jelenti azt, hogy az én jövőmet garantálni fogja.

**Ha egy ábrán össze szeretnénk szedni ezt az élethosszig tartó tanulást, az valahogy így nézhetne ki.** Ebbe benne van informális, formális tanulás, pedagógia, andragógia, attól függ, hogy milyen szintéren történik, illetve ugye azok a tanulási módok, amiket a mindennapi gyakorlatunk során mi végzünk. Hiszen mindenki tudja, hogy az egy dolog, hogy ül az iskolapadban, bekerül az életbe és szembesül az élettel, nagyon sokat fog más kollégáktól, gyakorlatból megtanulni.

**Ha azt mondom, hogy távoktatás, e-learning, kinek mi az első koncepciója, ami eszébe jut ezzel kapcsolatban?** Távoktatással került már valaki kapcsolatba? Akkor én néhány kulcsgondolatot hadd engedjek meg magamnak. Gyakorlatilag aki távoktatással még nem találkozott, az a gond, hogy bizonyos fogalmi keretei hiányoznak, amit először meg kell beszélnünk, mert nem tudja ezt az egészet elhelyezni. Maga a távoktatás, ugye benne van a nevében, hogy táv-oktatás, ez általában azt jelenti, hogy a tanulót azt nem látjuk személyesen, ez az egyik megnyilvánulási formája, de ebből jön egy előny is, a távolságok áthidalása. Az egészségügyi szakképző és továbbképző intézet, ahogy a nevében is benne van főleg egészségügyi képzésekkel foglalkozik.

Nézzünk erről a területről egy példát. Vannak például olyan szakképesítések az egészségügy területén, ahol 3-4 évente kell kiképeznie 20-as létszámot. Na, most mondjuk egy ilyen laborasszisztensi szakmára egyszerűen ennyi az igény erre a szakterületre, viszont ennyi ember kell, ha azt nézzük, hogy ez a 20 ember szétszórta helyezkedik el az ország területén. Két lehetőségünk van, vagy hagyományos tanfolyamos képzést szervezünk, ugye ennek az árát a tanulók, munkáltatók erősen meg fogják fizetni, mert el kell engednie az embert, ki kell fizetnie az útiköltségét, sokkal hosszabb lesz a távolmaradás. A többnapos tanfolyamoknál vannak szállás és egyéb költségek felvetődnek. Egy távoktatási módszertannal ez a kérdés például tökéletesen

áthidalható. Viszont távoktatásnál fogunk nézni konkrét tananyagokat, beszélünk picit a tananyag fejlesztési elveknél is, hogy nem használhatjuk azokat a hagyományos tananyagokat. Ugye egy hagyományos képzésnél, amiről mindenkinek vannak fogalmi keretei, ott van egy előadó-oktató, aki interpretálja az anyagot és ott van mellette még egy tananyag, amit kiegészít az előadó vagy kezelő, de mindenképpen az előadóval egy együttélésben van. Ugye egy általános távoktatásnál a tananyagoknak olyannak kell lenni, hogy bizonyos szinten az előadó szerepét is átvegye, olyan funkciókat kell biztosítani, amit egyébként az előadó biztosít.

Hogy mondjak egy gyakorlati példát. Ugye tudjuk, hogy a figyelemnek vannak korlátai, 20 perc körül, és amikor egy rutinos előadó látja, hogy mikor bólogatnak a fejek, csukódnak le a szemek, vagy egyszerűen merednek a semmibe, akkor ilyenkor van az, hogy elő kell venni a kis rutinunkat, elterelni a témát, feldobni egy viccet, feltenni egy érdekes kérdést, hogy megújuljon ez a koncentráció és tudjunk érdemben tovább haladni. Ugye távoktatási tananyagnál nincs ott egy előadó, aki azt mondja, hogy „Hohó, te hagyd abba, mert látom, hogy a fejed leginkább előre bukik és nem kéne folytatni, mert így nincs értelme.” Hanem be kell építeni a tananyagokba olyan lazító jellegű funkciókat, hogy azt mondjuk, hogy gondolkodjunk 20 képernyőképes egységekbe és vagy ott van vége az egységnek, és akkor úgyis azt gondolja a tanuló, „Ha vége van, akkor úgyis felállok egy kávéra”, vagy ha egy nagyobb tananyagrészeről beszélünk, akkor nem meglepő, ha találkozunk egy olyan ábrával, ami azt mondja, hogy „Figyelj, te most már elég sokat foglalkoztál a tananyaggal, hogy érdemes lenne kinyújtóztatnia a lábadat vagy valami meghökkentő dolgot írunk. Ugye épp az a cél, hogy meghökkenjen a tanuló, hogy kerül ez ide? Hát úgy kerül ide, hogy el kell terelni a figyelmét. Ezt nem feltétlenül kell a tanulónak megmagyarázni, ez a célja annak a funkciónak.

**Speciális szerepek jelennek meg a távoktatásban.** Ugye míg egy hagyományos oktatásban van a képzést szervező intézmény, a tanuló, illetve az oktató, ebből az oktató az, ami átmegy tutor szerepbe optimális esetben a távoktatásban, tehát nincs egy nagyobb óraszám, ahol interpretálja az anyagot az előadó, hanem az anyagot megkapja készen, oktatócsomagban a tanuló és csak támogatni kell a tanulót a tanulásban. Hogy hogyan, ezt meg fogjuk nézni egy picit részletesebben.

**Említettem ugye, hogy milyen szerepkörök vannak a távoktatásban, de nem beszéltünk arról, hogy milyen kihívás az adott szerepkörnek ez.** Ha belegondolunk, a tanuló mihez van szocializálva? Mint ahogy mindannyian, alapvetően az iskolapadhoz. Azt ismeri. Ő ahhoz szokott hozzá, hogy odamegy, leül, végighallgatja, amit mondanak. Eldönti, hogy jegyzetel, nem jegyzetel, befogadja-e vagy nem fogadja, ott van fejben vagy nincs a fejben. Ahhoz van szokva alapvetően, hogy odamegy, meghallgatja, jegyzetel és majd abból tanul. Neki egy olyan távoktatási forma, mint a távoktatás, e-learning, kihívást jelent, hiszen sokkal nagyobb önállóságot kap. Nem biztos, hogy ő alkalmas arra, hogy távoktatásban tanuljon vagy sokkal többet kell segíteni, hogy tanulni tudjon. Hiszen sokkal több önfegyelmet kíván meg egy távoktatás. Ugye a tanfolyamnál ott van egy jelenléti ív, itt voltál, nem voltál, ott van egy motivációs erő, hogy ott legyen és hallgassa. Távoktatásnál nincs. Akkor foglalkozik vele, amikor ideje van. És ha nem tudunk egy magas motivációs szintet fenntartani a tanuló részéről ebben a távoktatásban, akkor nagyon könnyen kudarcba fulladhat. Ez az egyik kulcskérdés a távoktatásban, a motivációs szint, hogy milyen technikával tudjuk a motivációt magasan tartani a tanulóknál. Ez a motiváció egy érdekes kérdés, ugye mert alapvetően, amikor valaki megkezd egy távoktatást akkor elindul egy magas motivációs szintre. Azért van ott, hogy tanuljon, azért jelentkezett. De megbetegszik a gyereke, nem volt jó napja a főnökének és kettőt belerúgott, most csökkentik megint a családtámogatást, ezek olyan dolgok, amik látszólag nem kötődnek az adott tananyaghoz, nem is, de a motivációt azt nagyon tudja rombolni. Azért van ott a tutor és az intézmény, hogy különböző technikákkal magasan tartsa ezt a motivációt.

**Tutor: ki az akit általában tutorként alkalmazunk?** Ez ugye attól függ, hogy milyen intézményi berendezkedésünk van. Lehet a tutor egy korábbi tanár, tehát egy főállású oktató, de őt akkor is fel

kell készíteni, hiszen ha megnézzük a mai pedagógiai oktatást, leginkább még mindig a hagyományos, jelenléti oktatásra fókuszál és a tanárunk valószínűleg nincs képben a távoktatási módszertannal. Tehát neki egy felkészítésen mindenképpen át kell menni, hogy hatékonyan tudjon tutorálni, hatékonyan tudja támogatni a tanulót. A másik lehetőség, amit az intézményünk gyakorol, az egészségügyi oktatások terén, hogy az adott szakmában jártas szakembert kérünk fel. Tehát például egy egészségügyi menedzserképzésnél ápolási igazgatók a tutorok. Ők a szakmában otthon vannak, profik, azért lehetnek ott ahol, viszont a pedagógiai oldal az nekik jóval gyengébb, hiszen azt nem tanulták. Tehát őket pedig egy pedagógiai jellegű felkészítésben kell részesíteni, ahol megmagyarázzuk nekik, hogy mi ez az egész távoktatás és hogyan kell működni a távoktatásban.

**A szervező részéről milyen kihívás van?** Na most ha megnézzük egy e-learninges képzést, kell egy olyan rendszer, amit működtetünk, ami kiszolgálja ezt az egész e-learning feladatokat, tehát egyrészt technikai kihívásról beszélhetünk, másrészt szervezési oldalról is sokkal előrébb kell gondolkodnunk, mint egy hagyományos tanfolyamos képzésben. Amikor beindítunk egy e-learninges képzést, az elejétől a végéig le kell legyen szervezve, és a tanuló, amikor megkezdja a képzését, akkor az összes információt a teljes képzésre előre meg kell, hogy kapja. Tehát mindig több lépéssel előrébb kell járjunk. Amit eddig meséltem az a távoktatásra, e-learningre egyformán igaz. Evezünk most picit az e-learning vizeire. Itt is elkezdeném egy definícióval.

**Ha felmegyünk a netre, töménytelen mennyiségű e-learning definíciót találhatunk.** Személy szerint nekem ez az egyik kedvencem., amit itt látunk, mert nemcsak a tananyag, a technológia, hanem a módszertan támogatás is benne foglaltatik. Gyakran találkoztam olyan e-learningnek hívott megvalósításokkal, amik valójában nem azok, vagy távoktatási megvalósításokkal. Megfogalmazom másképpen, úgy egyszerűbb lesz. Levelező képzés: ha azt mondjuk, hogy levelező képzés, az távoktatás vagy nem távoktatás? Távoktatásban is belefér az, hogy központi konzultáció, tutori konzultáció. De vajon miért nem? Én sem gondolom azt, hogy távoktatás, de én azért nem, mert a tanuló nem kap megfelelő támogatást. Nincs külön tananyag általában a levelező képzéseken, ugyanabból kell felkészülni, mint akik ott vannak órán, mint amit interpretált a tanár, tehát semmivel sem kapnak többet tananyag oldalról. Nincs megfelelő módszertani támogatásuk, tehát a konzultációk között nem nagyon tudnak akárkihez is fordulni, hogy kérdezzenek. Maguk a konzultációk eleve nem olyan jellegűek, hogy konzultálni lehessen, mert odamegy egy nagyszámú levelező évfolyamhoz az előadó, megpróbálja ugyanazt ledarálni, mint a nappalisoknál csak éppen töredék időben. Arra esélye sincsen a tanulónak, hogy kérdezzen, én ezért inkább pedagógiai módszertan alapján mondanám azt, hogy ez nem távoktatás.

Ha meg szeretnénk nézni kicsit vizuálisabban, hogy hol helyezkedik el az e-learning, akkor ennek a keretét az egésznek a lifelong learning adja meg, annak egy területe, egy módszere lehet a távoktatás, amit ennek a keretében tudunk használni. És ugye a távoktatás és az informatikai kommunikációs technológia keresztmetszetében helyezkedik el. Konkrétan ez az e-learning.

Tételezzük fel, hogy van egy multimédiás CD, ahol van egy önellenőrzés a végén. Az én fogalmi kereteim szerint még ez sem e-learning. Akkor, ha visszaevézünk a definícióig, akkor ugye nincsen tutori, mentori támogatás az egészben, pedagógiai oldalról hiányzik a támogatás. Csak azért lovagolok az elején, hogy nagyjából hasonló fogalmi kereteket használjunk.

E-learning, b-learning, t-learning, számtalan ilyen egybetűs learninggel találkozhatunk jártunkban-keltünkben, akár újságban, akár interneten, ha nézelődünk. Igazából két elv szerint szokták ezeket elnevezni. Az egyik James Bond kedvenc mondatával élve: ez a keverve nem rázva, azaz, hogy milyen módon keveredik a jelenléti oktatás és a tiszta, natív e-learning, amikor nem is látjuk a tanulót. Ugye ezt hívják blending learningnek, amikor valamilyen keveredés van. Ha megnézzük ezt a skálát, ugye az egyik végén van a tanfolyam klasszikus jelenléti oktatás, semmi e-learninggel elemmel, a másik végén a klasszikus e-learning, ahol semmiféle jelenléti elem nincs, és a kettő közötti összes felület az a blending learningnek a területe, ahol valamilyen arányban keveredik a jelenléti oktatás és az e-learning. Én azt mondom, hogy inkább blending learning hívó vagyok, mint klasszikus e-learning, mert tapasztalatom alapján a legtöbb tanuló valamilyen mértékben azért még

igényel egy személyes kapcsolatot. De ha eltekintünk ettől ugye, akkor is van olyan dolgok, amiket e-learningben nem lehet megtanítani. Például elsősegélynyújtás. Ha valaki nekem e-learningben az elsősegélynyújtást megtanítaná azon nagyon csodálkoznék. Annak kell lenni mindenképpen egy gyakorlati részének, amikor ott vagyunk, megnézzük hogyan kell egy kötést feltenni például. Hogy kell egy újraélesztést kivitelezni. Ezt is meg lehet mutatni, szemléltetni valamilyen módon e-learningben, de a gyakorlatot nem pótolhatja értelemszerűen.

A pedagógia itt elég erős itt a teremben, tehát a De Block taxonómia nem ismeretlen a jelenlevők számára. Felosztotta De Block az ismeret szintjeit, a tudás szintjeit ismeret, megértés, alkalmazás, integrálásra. A klasszikus e-learning, a tiszta e-learningnél mondjuk az ismeret megértés szintig lehet eljutni, ha alkalmazás integrálás kell, akkor mindenképpen kellene bele azok a jelenléti elemek is. Tehát ez az egyik rész, ez a blending témakör ahonnan származhat a név, a másik rész, egyszerűen az, hogy milyen eszközt használunk. Tehát például t-learning néven szokták azt emlegetni, azokat a módszereket, amiket televízióra alapoz. Számítalan ilyen újat látok naponta én is. Találnak egy új eszközt és csinálnak hozzá egy új learninget, és innen származik a neve. Így lehet rendet tenni ezen a területen.

**Néhány szót az e-learning keretrendszeréről.** Ugye ha mi e-learninget szeretnénk folytatni, kell valamilyen keretrendszer, amire alapozva azt megtesszük, ami a technikai háttérrel fogja nekünk biztosítani. Két nagy részre lehet osztani a keretrendszert, egy learning menedzsment system és egy learning content menedzsment systemre, tehát igazából az a rész, ami egyrészt a tanulók adatait kezeli a másik az a rész, ami az anyagot, ami fenn van a rendszerbe kezeli. Ez igazából csak technikai elkülönítés.

Amiről érdemes beszélni ennek kapcsán, hogy nézzük meg a szabványossági szempontokat. Számítalan e-learning keretrendszert lehet a piacon találni, fizetőset, nem fizetőset, de már az e-learningben is vannak olyan kvázi szabványok, amit, ha komolyabban szeretnénk vele foglalkozni, nem szabad figyelmen kívül hagyni. Erről lesz egy dia később, ebbe most nagyon nem ugrok bele. Funkciók tekintetében általánosan ki merem jelenteni, hogy az e-learning rendszerek funkcionalitás tekintetében kb. 80%-ban megegyeznek, a különbség az a maradék 20%-ban rejlik. Vannak olyan funkciók, amik minden e-learning rendszerben megvannak egy chat, egy fórum, például egy naptárfunkció.

Harmadik, ha piaci szempontból nézzük meg, ahogy felmerült az előző előadónál is. Ha a klasszikus kereskedelmi rendszerről beszélünk, ott általában meg kell vennem egy Windows operációs rendszert, meg kell hozzá venni valamilyen fizetős adatbázis rendszert és meg kell venni magát ezt a keretrendszer szoftvert. Tehát elég nagy befektetéssel lehet elindulni. Másrészről a keretrendszereknél érdekes kérdés, hogy hogyan licencelik. Például a Magyarországon legelterjedtebb hazai fejlesztésű e-learning keretrendszerrel azt mondják, hogy sávós licenceket adnak. Mondjuk 1-10-ig, 10-100-ig. Ez online felhasználót jelent, aki egyszerre a rendszerben tartózkodik. Ha nekünk van terveink szerint durván 20-30 ember, aki online egyszerre tartózkodik a rendszerben, akkor is meg kell venni a 10-100-as sávot, mert ott van és pont. Ez egy újabb kérdéskör. Így ez az ILIAS rendszer, amiről beszélünk, maga a keretrendszer is ingyenes, open source szoftver. Ingyenesen letölthető, üzembe állítható, nulla forintos szoftver oldali befektetéssel el lehet indulni.

**Beszéltünk egy kicsit a rendszerről, ami szép szép, de mégis hát mégis a kulcs a tananyag, hogy mit teszünk fel ebbe a rendszerbe, mennyire elsajátítható önállóan.** Így első körben csak néhány kulcsmondatot szeretnék megosztani, mert sajnos az előadás keretei erre adnak időben lehetőséget. Javasolom mindig azt, hogy gondolkodjunk képernyőképekben. Amit egyszer látunk, amit egyszer be tudunk fogadni. Az a tananyag az én fogalmaim szerint, ami görgetni kell lefelé oldalakon keresztül és e-learning tananyagoknak hívunk, az már zaccos, az nem igazán jól feldolgozható. Mindig egy egység egy képernyőkép. Ha egy dolgot én úgy tudok elmondani, hogy egy mondat és két képernyőkép, akkor csak az szerepeljen azon az oldalon. Ha egy darab videó és egy kérdés egy gondolati egység, akkor annyi szerepeljen. Ugye nincs papír, amivel nekünk

spórolni kellene, és úgy kellene tördelni az anyagunkat, hanem egy-két lép van, amit egyszerre be tudunk fogadni.

**Speciális nyelvezetet igényel az e-learning épp amiatt, hogy könnyen és önállóan elsajátíthatónak kell lenni.** Ugye gondolom saját szakterületén mindenki találkozott már olyan nagynevű professzortól, akárkitől származó, a szakma bibliájának tartott könyvvel, aminek általános jellemzője az, hogy akkora, hogy talicska kell hozzá, hogy eltolja az ember. Másrészt olyan apró betűvel íródott, hogy nagyító kell hozzá, hogy elolvassa. És olyan óriási információsűrűség van rajta, hogy egy oldalt legalább négyszer-ötször el kell olvasni, ahhoz, hogy legalább a fele lecsengjen. Ugye ez az, ami e-learningben teljesen megengedhetetlen, olyan nyelvezetet kell alkalmaznunk, ami világos, jól érthető, de természetesen a tartalomnak meg kell maradnia.

**Előtesztelés, bejárás utak.** Mire gondolok én itt? Nem tudom, tudja-e valaki, hogy Japánban a parkokat hogy szokták építeni? Először befűvesítik, aztán amerre járnak a gyalogosok, ott betonoznak. Nem véletlen, ha a magyarországi parkokat megnézzük, ott vannak az utak, és ott vannak keresztül rajta az ösvények. Nem biztos, hogy a tanulók arra akarnak menni, amerre mi kigondoltuk, hogy menni akarnak. Tehát, ha megvan egy anyagunk mindig érdemes tesztelni először és aszerint módosítani, hogy hogyan használják azt a való életben.

**Milyen kötelező elemek vannak pedagógiai szempontból, amit nekünk használnunk kell a távoktatásban?** Cél és követelmény megjelölés. Aki foglalkozott egy kicsit andragógiával, az tudja, hogy a felnőtt tanulónak általában az az igénye, hogy célzottan azt tanulja, amire neki szüksége van. Tehát, ha az elején megjelöljük a célt, sokkal motiváltabbá tehetjük, de ha csak egy középiskolai szinten gondolkodunk, akkor sem árt, hiszen az ember szereti tudni, hogy mi az, amit ő tudni fog, ha elsajátította azt az anyagot. Ugye ez egy sokkal másabb megközelítés, minthogy tanulja a tantárgyakat és majd egyszer valami összeáll belőle. Így egy kézzel foghatóbb valami van, ami motiválhatja.

**Tanulási útmutató.** Miért szükséges ez? A tanulási útmutató azért kell, hogy valamilyen szinten helyettesítse a tanárt. A tanulási útmutató az a rész, ami el fogja mondani, hogyan érdemes azt a tananyagrészt elsajátítani, felhívja esetleg bizonyos részekre, amik nehézséget szoktak okozni a figyelmet. Megmondja azt, hogy kb. mennyit érdemes foglalkozni azzal a tananyagrésszel.

**És végül az önellenőrzés.** Ami megint csak kulcsfontosságú a távoktatásban, hiszen az önellenőrző kérdések, az önellenőrzés lehetősége az, ami visszajelzést adhat nekem. Ugye egy hagyományos oktatásban ott van az oktató, aki visszakérdez, illetve a tanuló is kérdezhet az oktatótól, hogy jól értette-e azt a részt. Önellenőrzésnél ezt meg tudjuk a rendszeren belül oldani, ott van egy önellenőrző teszt sorunk, kérdőívünk és ezen lemérheti a tudását a tanuló, hogy elég időt fordított-e arra a tananyagrésze, jól megértette-e azt a tananyagrészt.

**Picit a divathullámról a web 2.0-ról, ami azért az e-learningben is elég nagy változásokat hozott.** Az egyik a kollaboratív tanulás előtérbe kerülése. Ugye a hagyományos modellben, hogy működik a dolog? Én állok itt, az oktató a katedrán, én vagyok az okos, én mondom a nagy tudást és a hallgatóság csak befogadó. A klasszikus weblapoknál is hasonlóképpen működött, én feltettem a tartalmat, mindenki más nézhette. Kollaboratív tanulás például a wiki, a wikik megjelenése. A wiki egy olyan műfaj, amit bárki szabadon szerkeszthet. Én hozzászólhatok, hozzátehetem az én tudásomat, tehát nem egy személy van innentől kezdve, aki a nagy jó és mondja az igazat, hanem egy nagyobb közösség rakja ezt össze. És épp ez működteti ezt a rendszert, a közösség.

**Egy picit az ipari modellről.** Mit értek én ipari modell alatt? Ha megnézzük a mai oktatást, az hasonlít egy kicsit az iparhoz. Van egy alapanyagunk, egy inputunk, a gyerek, aki bejön az iskolába.

Érik őt mindenféle hatások, nem kalapács, hegesztés és egyéb történetek, de ott vannak mindenféle emberek, akik hatnak rá. És a végén kijön egy késztermék, elméletileg egy olyan tanuló gyerek, aki sikeresen elvégezte azt, felkészült. Az iparral csak egy gond van, hogy ott mindig van selejt hányad. Ez igazából a mai oktatásra is elmondható, de ez az a terület, ahol nem lenne megengedhető ez a történet. Tehát nem véletlenül megyünk egy kicsit más irányba. Ami még jellemző ma, ha a felnőttoktatás részét nézzük meg: én mindent most akarok, gyorsan akarom. Én most akarok elkezdni tanulni, nem akarok két hónapot várni, míg elindul egy tanfolyam, mert nekem most van szükségem arra a tudásra, hogy például el tudjak helyezkedni. Ez, ha megnézzük az ETI modelljét az e-learning szempontból, ezért van az, hogy a tanulók bejelentkeznek, ha éppen szombaton talált rá egy képzésre, akkor szombat este, internet bankon átutalja a képzés költségét, és akkor hétfőn tanulhat. Nem csoportok mozognak, egyedek mozognak a rendszeren belül, egyedi tutorálás van. Ez adja meg igazából a tanulás szabadságát, azzal együtt, hogy a tanulmányi időt se nagyon korlátozzuk. Tehát, ha mondjuk egy képzésnek az átlagos elsajátítási ideje egy átlag tanulóknál egy év, akkor a tanuló kap rá két évet. Ha neki sok ideje van és sok időt szán rá, nosza rajta, tanulja meg fél év alatt. Ha valami közbejön, és nem ér rá tanulni, tanulja meg másfél év alatt. Tehát szabadságot kap ezzel a tanuló.

**Az ígért szabványok:** igazából néhány szabványrövidítést tettem ide fel, kvázi szabvány, inkább így mondanám, mert igazi, technikai értelemben értett szabvány nincs, de ezek az elterjedt kvázi szabványok, amik meghatározzák most az e-learninget. Ebbe igazából nem gondoltam részletesebben belemenni, de ha kérdés van, nagyon szívesen beszélek még erről a részéről is. Emlegettem már azt a helyet, ahonnan most én jöttem az Egészségügyi Szakképző és Továbbképző Intézetet. Mi folytatunk e-learning képzéseket is, ugye említettem, hogy nem csoportok, hanem egyedek mozognak, egyének ebben a rendszerben és ezért is egyéni tutorálás van. Egy tutorhoz maximum 6-8 fő tartozik, nem véletlenül, mert ez teszi lehetővé, hogy nagyon szorosan nyomon lehessen követni a tanulókat és motiváltan tartani őket illetve említettem ezt a keretidőszakot is.

**Néhány szót a tutorálás, mentorálás kérdésköréről.** Ki a tutor? A tutor az az ember, aki egyrészt szakmai szempontból fog engem támogatni, hiszen ő ért ahhoz a tananyaghoz, amit én tanulok, tehát hozzá bármilyen kérdéssel fordulhatok, emberileg támogat. Nagyon fontos ez motivációs szempontból. Ugye említettem, hogy egy tutorhoz nálunk 6-8 fő tartozik. Ugye 6-8 fő egy olyan mennyiség, amit nagyon szorosan nyomon lehet követni. Általában a tutor és a tanulók között barátság is szövődik a képzés folyamán és ezt, ha csak szigorúan pedagógiai szempontból nézzük és eltekintünk az emberi oldalától, ami szintén nagyon jelentős tud lenni. Ha valakivel én olyan viszonyban vagyok, akkor sokkal jobban tud rám hatni, sokkal jobban tud motiválni, sokkal jobban tud segíteni. A tutor értékeli is, hiszen ez benne van a funkcióiban az értékelés, ez esetünkben nem egyenlő az osztályozással. Az értékelés egy sokkal tágabb fogalomkör, az is hozzátartozik, hogy visszajelzést adunk a tanulóknak a teljesítményével kapcsolatban, anélkül, hogy őt bekatégorizálnám, hogy egyedül ebbe adok neki segítséget. És amit említettem, nagyon fontos, hogy a tutorunkat pedagógiai szempontból is felkészítsük, hiszen ha pedagógus akkor sem biztos, hogy rálát a távoktatás módszertanára, ha nem pedagógus, akkor nagyon fontos, hogy kapjon egy ilyen nézőpontot.

**Kommunikációs csatornák címén mit értek itt:** azt szeretném elmondani, hogy milyen módon kommunikálhatunk mi a tanulókkal távoktatásban, e-learningben? Milyen kommunikációs csatorna van a tanuló, a tutor és a szervező intézmény között. Ugye akár a személyes megközelítéstől kezdve, ami a legritkább az e-learningben, távoktatásban, de mondjuk itt is lehet egy központi konzultáció, egy gyakorlat, rendelkezésünkre áll a hagyományos telefon, különféle internet telefonok, Skype, azonnali üzenetküldők, e-mail, fórumok, chat, a rendszeren belüli körlevél funkciók. Lényeg az, hogy minél több csatornát halmozzunk fel, hogy úgy mondjam, mert minél több csatornánk van, annál nagyobb az esély, hogy azonnal elérjük a tanulót és annál nagyobb az esély, hogy a tanuló minket meg fog találni, hogy ha ránk van szüksége, mint tutorra.

**Kicsit e-learning módszertan vizekre evezve.** Milyen információforrások állnak rendelkezésre a képzőintézménynek és a tutornak, hogy szorosan nyomon tudja követni a tanulót. Az egyik maga a tanuló, tehát mi folyamatosan kapcsolatban állunk, kérdezhetünk, a tutornak az a dolga, hogy ha azt látja a rendszerben, hogy nem lépett be a tanuló mondjuk 3 hete, akkor küldjön neki egy emailt, hogy „Szia, valami probléma van, hogy nem tudtál foglalkozni az anyaggal?”

A rendszer statisztikái: mikor lépett be, mennyit töltött ott, mennyit foglalkozott az anyaggal, az önellenőrző kérdéseket milyen sikerrel töltötte ki. Ezt meg fogjuk konkrétan nézni a rendszerből, hogy ezek hogy néznek ki. Illetve lehetnek, mint bármilyen képzésben különféle mérési pontok, amelyek hozzájárulnak az információhoz.

**És az elméleti nagy ívű részből az utolsó rátekintés: ez a hogyan tutoráljunk?** Hogyan érdemes nekünk tutorálni, ha távoktatásban, e-learningben gondolkodunk? Nagyon fontos a gyors reakció. Minél gyorsabban tudunk reagálni, a tanulónak, annál inkább motiválja arra, hogy többet kérdezzen tőlünk. Ha feltesz egy kérdést és mi két hét múlva válaszoljuk meg, addigra lehet, hogy már ő is megtalálta az anyag egy más részében, kitérte az internetről. Lehet, hogy ezzel egy életre elvettük a kedvét, hogy tőlünk kérdezzen.

**Gyakori kapcsolat:** hogy ha nem keres minket a tanuló, akkor a tutor dolga, hogy megkeresse a tanulót. Ugye említettem, hogy például látjuk, hogy nem jelentkezett be régóta a rendszerbe. Ilyenkor vegyük a fáradtságot és keressük meg. Ennek számtalan oka lehet: éppen úgy jött ki az életében a lépés, hogy beteg volt a gyerek, ő kifáradt, nem ért rá foglalkozni az anyaggal, de az is lehet, hogy kicsit lazára eresztette a gyepelőt és tudunk húzni egy kicsit rajta, hogy ne álljon meg ebben a tanulási folyamatban. Érdemes a tutornak mindig különféle információkkal színesíteni az anyagot. Ugye egy oktató, ha bármelyikünket is megnézzük, sose csak azt szokta elmondani szigorúan, ami ott van a tankönyvben. Mindenkinek megvannak a saját tapasztalatai, a saját példái, a saját élete, amivel kiegészíti azt a tananyagot. Ezt meg tehetjük az e-learningben is. A keretrendszer erre ad lehetőségeket, meg tudjuk nézni, hogy konkrétan mit. Ez megint egy motiváló erő, mert ha heti rendszerességgel felteszünk mi valami új információt a tananyaggal kapcsolatban vagy valami módosítást végzünk, kiegészítést adunk. Ezzel megint arra motiváljuk a tanulót, hogy nézzen fel sűrűbben a rendszerbe, használja, járjon vissza.

**Kihívások, lehetőségek, jutalmazások: mit értek én ezalatt?** A tutori pályafutásom alatt csináltam egyszer egy olyat, hogy én az egészségügyi alapismereteket természetgyógyászok számára szép kilométeres nevű képzésben tutorálok, és volt egy 10 kérdéses kérdéssorozat a tanulóknak. És akik legjobban válaszolták meg, legrövidebb idő alatt ezt, azok nyertek egy anatómiai atlaszt ennek az egésznek a végén. Ez egy olyan érték volt nekik, amit ők tudnak használni a képzés során, tehát érdemes volt hajtani érte, másrészt ezeket az eredményeket valamilyen szinten közzétettem a tanulóknak és megint azt látták, hogy vannak emberek, akik húznak és akkor ők is megpróbálnak felzárkózni, ők is megpróbálnak bekapcsolódni ebbe, tehát megint motivációs jellege van.

Én röviden, első körben módszertani rátekintést ennyit gondoltam mondani. Az az igazság ez az a téma amiről kb. két napot áttudnék beszélni és úgy, hogy értelme legyen. Most erre volt lehetőségünk. Itt a rendszerbemutatás során természetesen fogunk még érinteni módszertani kérdéseket, illetve az a kérésem, akinek kérdése van nyugodtan tegye fel azonnal, nagyon szívesen megválaszolom illetve mehetünk abba az irányba egy kicsit. Az az igazság, hogy elég sok irány van amerre el tudunk kalandozni ennek kapcsán. Nem tudom, most első körben kérdés ezzel kapcsolatban felmerült-e?

**Hallgató:** milyen technikai háttér kell hozzá?

**Előadó:** tehát technikai háttérből klasszikus LAMP (Linux, Apache, MySQL, PHP), tehát PHP, MySQL, ami kell hozzá. Van benne egy kis Java is, tehát JavaScript szinten, tehát az is szükséges hozzá. Tanulói oldalról meg egy böngésző gyakorlatilag és kb. ennyi.

**Hallgató:** van arra példa Magyarországon, nyilván te ismersz-e, hogy ez az e-learning ez egyszerűen furakszik a hagyományos oktatásba is. Nyilván nem a középiskolákra gondolkodom leginkább, meg az alapképzésre, de mondjuk egy főiskolán, egyetemen pont említettük például a levelező képzést ahol az ember mondjuk hetente 2-300 km-t utazgat.

**Előadó:** gyakorlatilag a felsőoktatási intézmények nagy része, sőt talán majdnem mindegyike most már használ valamilyen e-learning rendszert. Hogy milyen e-learning tevékenységet fejtenek ki, az egy más kérdés, de van rendszerük. Tehát így nyilatkoznék.

**Hallgató:** én még nem hallottam ilyet, attól lehet, hogy létezik, hogy mondjuk van olyan tantárgy, amiből e-learning módszerrel le tudnák vizsgázni.

**Előadó:** az a gond, hogy hiába van elméletileg olyan módszertani tudás a felsőoktatásban, az oktatóknak, a legtöbb oktatóknak van egy csőlátása. Nem tudom a felsőoktatásból ki jött, hogy mennyire tud megerősíteni vagy cáfolni, de például előadókat rávenni arra, hogy használják az e-learninget nagyon nehéz. És a vezetés meg általában nem mernek felvállalni olyan mértékű konfrontációt, hogy elinduljanak nagyobb lépésben az e-learning felé. Nekem ez a tapasztalatom. Illetve a másik az, hogy ott is kéne egy módszertani megerősítés mert ott általában a hagyományos jelenléti oktatáshoz szoktak és nem biztos, hogy bevállalják, de ha valaki módszertani segítséget tud nekik nyújtani, ennél persze részletesebbet, mint ami itt elhangzott, az azért fel tudja nyitni a szemeket és elindítani.

**Hallgató:** csak azt akartam hozzátenni, hogy ha nem is egész képzésben, de tantárgyakban észrevettem az e-learninges oktatás történetét is.

**Előadó:** még nem olyan széles körű.

**Hallgató:** tehát semmiképpen nem gondolom azt, hogy az e-learningnek legyen szerepe a nappali képzésben, mert ott azért egy professzor meghatározó lehet. Egy levelező vagy egy posztgraduális képzésben ott meg mindenképpen úgy gondolom, hogy ott meg elvitathatatlan dolognak kéne lennie.

**Előadó:** igen, ezen keresztül kellene, hogy működjön, de nappali képzésben is van, mert az ETI például az összes tanfolyamos képzésnél támogató szerepben használja az e-learning rendszert. Mert hiába van egy tananyag. Ha egy hagyományos tananyag előállítását megnézzük, gyakorlatilag mire kiadásra kerül a könyv gyakorlatilag elavult lesz. Utána azt nem frissítik, mert először el kell adni a raktárkészletet tehát ha csak azt mondjuk, hogy a frissítéseket az előadó csak a rendszeren keresztül érheti el, felteszi a tanulóknak, vagy az előadás vázlatait, prezentációt felteszi a tanulóknak. Vagy ott van az internet, ami óriási erőforrás és információforrás, de azt tudjuk, hogy nagyon sok szemét is van közte. Ha az oktató azt megteszi, hogy szűri az internetet és felteszi a linkeket, amik hasznos és valid adatokat tartalmaz az ő anyagával kapcsolatban, már az óriási segítség lehet, nem is beszélve a kommunikációs funkciókról.

**Hallgató:** egy dél-afrikai professzorral beszéltem nem is olyan régen. 160 ezer diákja van az egyetemen, és e-learning az egész. Azt akartam kérdezni, hogy egy összehasonlítást a Moodle és az ILIAS rendszerrel kapcsolatban kaphatnánk-e?

**Előadó:** a Moodle az azt mondanám, hogy svájci bicska tehát sokkal modulárisabb jellegű, az ILIAS kicsit monolitikusabb. Technikai szempontból mind a kettő PHP, MySQL, tehát technikai szempontból nagy csoda nincs. A Moodle-höz azt mondom, hogy több technikai szakértelem is szükséges, hogy igénybe vegye az ember és csatasorba állítsa, épp a moduláris jellegéből adódóan. Az ILIAS-nál azon kívül, hogy monolitikus, bizonyos funkciók kikapcsolhatóak, de ilyen, hogy hozzá modulokat telepítünk még külön, ilyen nem jellemző ILIAS szinten. Más egy kicsit a gondolkodásmódja az ILIAS-nak, mint a Moodle-nek, de igazából ezt prezentálva tudnám jobban bemutatni.

**Hallgató:** egy fontos különbség. Én a rendszerek között keresgélve próbálom meghatározni, hogy mi lenne az én céloknak a legmegfelelőbb, és ami nekem egy fontos különbség volt, elhangzott a SCORM szabvány, amit most már nevezhetünk nyugodtan szabványnak. Ahogy a neten keresgélve próbáltam utána olvasni a Moodle-be, hegeszteni lehet SCORM-nak csak egy régebbi verzióját adja ki úgy ahogy, az ILIAS pedig megeshi a legújabb, legeslegújabb verziót is.

**Előadó:** a SCORM 2004-es a legújabb, az 1.2 az előző, a mostani aktuális ILIAS mind az 1.2-vel, mind a 2004-el level 3 szinten kompatibilis. Tehát teljesen kompatibilis vele. A Moodle-t nem tudom, tehát arról nem tudok nyilatkozni.

**Hallgató:** az 1.2-t, hogy ha ügyesen trükközöl, akkor ez például nekem fontos szempont, mert úgy gondolom, hogy érdemes szabványanyagot csinálni, tehát a netrendszerek között hordozni és a Moodle-lel ez probléma.

**Előadó:** amit én hallottam, ott ez nem merült fel de én ilyen mélységében nem mentem bele, az az igazság, de amit olvastam, a Moodle-ről aszerint kompatibilis. Én csak ennyit tudok. Az 1.2-vel biztosan. De ha ilyen tapasztalat van, az egy más kérdés. A Moodle-be ilyen mélységében nem mentem bele.

## Erdélyi Gábor: E-learning keretrendszer II

A rendszer, amit most látni fogunk, az az egészségügyi szakképző és továbbképző intézet rendszere lesz. Ebben fogjuk megnézni azokat az elveket, funkciókat, gondolatokat, amiről röviden, felvezetőleg beszéltem az előző előadásban. Ez a rendszer a [www.etitav.hu](http://www.etitav.hu) címen érhető el. Ha valaki először erre az oldalra tallóz, két lehetőség van. Pontosabban, ha először tallóz ide, akkor valamit meg szeretne nézni, meg szeretné nézni, hogy mit tud kínálni ez az intézmény, akkor el tud menni magának az ETI ILIAS rendszerének a publikus felületére, ahol a publikus felületen ugye különféle információkat kaphat.

**Például, ha megnézi az e-learning képzéseket,** mert éppen ezek érdeklik, legyen ez az egészségügyi alapismeretek természetgyógyászok számára, akkor ugye van itt egy kurzus, amiről már a publikus felületen bejelentkezés nélkül információkat kaphat. Láthatja, hogy milyen nyelvű, ki az oktatás szervezője a képzésnek, hogyan történik a konzultáció, mi kell ahhoz, hogy hozzáférjen. Anyagokat lehet ide feltölteni, amit le tud tölteni erről az információs oldalról, akár tananyagmintát lehet már publikussá tenni. Lényeg az, hogy megkapja, azokat az alapinformációkat, ami számára ahhoz szükséges, hogy adott esetben érdekelje tovább a dolog és bejelentkezzen.

**Hogyan használjuk mi?** Saját életem megkönnyítése érdekében, mindenhová, ahol lehet kiteszem a technikai segítségnyújtás fórumát, mert ha valaki elakad, akkor ugye itt kérdezzen lehetőség szerint először. Maga a rendszer alapértelmezetten 22 nyelvet támogat, tehát ebből nekünk nincs mindegyik támogatva, csak az, ami számunkra érdekes lehet. Gyakorlatilag a publikus felületre mi bármit kipakolhatunk, amit szeretnénk. Az ILIAS-nak van egy nagyon komoly szerepalapú jogosultsági rendszere, azt majd egy villanás erejéig meg fogom mutatni. Tehát nagyon egyszerűen szabályozható jogosultságból mi látszódjon, mi ne látszódjon mihez férjenek hozzá akár a publikus felületen, akár a rendszeren belül. Maga az ETI a képzésen kívül általános információkat is kitett magáról erre a publikus felületre, például a rendszer felhasználói kézikönyvét. Egyébként az a tapasztalat, hogy ha valakinek van egy alapvető minimális internet rutinja a legtöbb funkciót zsigerből kezeli, hogy úgy mondjam, és csak néha-néha szükséges neki egy kis támogatás. Sőt van egészségügyi menedzser képzésünk is internetes e-learninges formátumban is. Na most, akik egészségügyi menedzsernek tanulnak, azok általában nem a legfiatalabb korosztály és egy minimális lökés után amikor elindult még anno ez az első képzés, nekik sem okozott gondot a rendszer használata. Vagy például az ETI kitette erre a publikus felületre a felhasználói szabályzat letölthető különféle információit. Tehát gyakorlatilag bármilyen elemet, amit a rendszer tud kezelni, ide ki tudunk tenni.

**Ha valakinek megtetszett valamilyen képzésünk, hogyan tovább?** Ugye először magába a rendszerbe kellene bejutnia. Mi azt a regisztrációs módot választottuk az ILIAS-ból, hogy bárki szabadon regisztrálhat a rendszerbe, ugye ez még csak arra jogosítja fel, hogy belépjen egy felhasználónév-jelszóval a rendszerbe, tudjon képzésekre jelentkezni, ugyanúgy, mint a publikus felületen, használhassa a fórumot, de a saját felhasználói nevével, de semmi más jogosultságot nem ad. Választhatunk mi olyan regisztrációs módot is, hogy szabad regisztráció nincs, valamilyen más tanulmányi intézményből veszi át a rendszer az adatokat. Akár egy Active Directory-val integráljuk, ha mondjuk vállalati képzésről beszélünk, és akkor onnan történik az autentikáció. Lehetséges az, hogy egy e-mail címet kell megadni és a megadott e-mail címre kerül továbbításra egy felhasználónév-jelszó, tehát csak valid e-mail címen lehet regisztrálni. Számptalan mód van. Az itt látható adatokat, tetszőlegesen szerkeszthetjük. Olyan mezőket veszünk fel, amiket szeretnénk, azt teszünk kötelezővé, amit szeretnénk. Tehát először a tanuló átmegy egy ilyen regisztrációs folyamaton. Utána már saját felhasználónevével, jelszavával be tud lépni a rendszerbe.

**Nézzünk be és meglátjuk, hogy hogy néz ki.** Ami először fogadja a tanulókat, amikor belépnek a rendszerbe, az a munkaasztal. A munkaasztal az a terület, amit teljes mértékben testre szabhat a tanuló. Olyan elemeket tesz ki, amit szeretne, azt vesz le róla, amit szeretne. Az itt látható blokkokat úgy variálja, ahogy akarja, kikapcsolhatja, áthelyezheti. Tehát saját magának egy egyedi munkakörnyezetet tud kialakítani ezen a munkaasztalon. Ide kipakolhat mindent, amit ő gyakran látogat, és innen gyakorlatilag egy kattintással közvetlenül ahhoz a részhez mehet. A munkaasztalról bármit le lehet venni, ugye az a levéltár gombra egy kattintás, vagy ha ki szeretnének tenni bármit, ugye ezen minden elem, ami a tanuláshoz kötődik a taneszköz tárolóban van. Minden elem mellett van egy olyan gomb, hogy munkaasztalra, egyszerűen rákattintva kikerül az adott elem a munkaasztalra.

**Milyen elemek lehetségesek ezen a munkaasztalon?** Egyrészt itt megjelenhet a tanulónak a saját határidőnaplója, erről fogunk még egy picit többet beszélni, jegyzeteket készíthet magának, megjelenhetnek itt az e-mailjei, beállításaitól függően láthatja, hogy éppen aktívan kik tartózkodnak a rendszerben, külső hírforrásokot vehet fel erre a munkaasztalra, webes hivatkozásokat készíthet, az általa a rendszerben elhelyezett TAG-ek megjelennek ezen a munkaasztalon, ezek a blokkok szabadon mozgathatók, a részletessége variálható. Látható ezek a blokkok alatt különféle részletességi nézetek. Tehát, ha mondjuk az aktív felhasználóknál én csak arra vagyok kíváncsi, az egyes részletesség, akkor azt látom, hogy hányan vannak bent. Kettes részletességnél már látok egy kicsivel több adatot. A hármas részletességnél pedig minden elérhető adatot.

**Felhasználói adatok alatt van arra lehetőség, ha lehetőséget adunk rá, hogy a tanuló megváltoztassa az adatait.** Mi erre lehetőséget adunk, hiszen, ha megváltozik az értesítési címe, lakcíme, akkor itt várjuk elsősorban, hogy adminisztrálja a tanuló, itt tud jelszót változtatni vagy bármilyen módosítást végezni a profiljában. Látható, hogy megadhatunk különféle üzenetküldő klienseket. Na, most az ILIAS egy szolgáltatáson keresztül monitorozni tudja, hogy az itt megadott Skype account él-e vagy nem él. Tehát a tanuló bejelentkezik a rendszerbe, én is ott vagyok a rendszerben és a munkaasztalon, például az én nevemnél megjelenik a kis Skype ikon, arra kattintva kapcsolatba lehet lépni velem és most látható, hogy éppen nem vagyok bejelentkezve. Tehát a státuszomat is lehet látni.

**Mit lehet itt még megtenni?** Vannak különféle körértesítési, körlevél funkciók a rendszeren belül. Itt a tanuló eldöntheti, a rendszeren belüli üzeneteket a rendszer által generált üzeneteket csak itt kapja meg, külső postafiókjába szeretné megkapni vagy mindkét helyre menjen. Amit kevésbé használnak, de azért megmutatom, hogy ilyen is van, hogy egy location-t, egy helyet be lehet állítani, hogy éppen hol tartózkodunk, vagy általában hol érhető el az ember.

**Határidőnapló:** minden tanulónak rendelkezésére áll határidőnapló, ez egy szabványos iCal alapú határidőnapló. Ugye egy embernek ugye attól függően, hogy hány csoportban, kurzusban van benne, számtalan határidőnaplója lehet, hiszen alapértelmezetten minden kurzushoz létrehoz egyet a rendszer. Amit látunk normál esetben, amikor be van pipálva minden, az egy aggregált nézet, tehát az összes naptár tartalma megjelenik, de ha csak egy bizonyosat szeretnének, azt is megtehetjük itt. Miért jó ez oktatásszervezési szempontból? Arról ugye nem beszélek, hogy megvannak ugye a szokásos funkciók, hogy különféle nézetekben (részletességgel) lehet ezt a naptárt megnéznünk. Azért jó nekünk, mert eseményeket hozhatunk létre egy naptárba és azt az összes tanulónak egyszerre el tudjuk küldeni. Tehát, ha létrehozunk egy új eseményt, vagy létrehoz valaki egy új eseményt, ami a mi naptárunkba be kell, hogy kerüljön, az itt a beérkezett időpontok közé meg fog érkezni, eldönthetjük, hogy betesszük-e vagy nem. Például legyen egy jelenléti képzést támogató helye, ahol azt mondjuk, hogy órarendváltás van és azt szeretnének minél gyorsabban kiértesíteni az összes tanulót. Az adott képzés naptárjában megteesszük ezt a módosítást, egy kattintás és az összes tanuló, aki arra a képzésre jár, megkapta ezt az információt.

**Jegyzetek, webcímek:** gyakorlatilag egy egyszerű felületet kapunk, ahol a munkaasztalra tudunk webes hivatkozásokat illetve ilyen virtuális sárga cetliket elhelyezni.

**Van egy szabványos kereső funkció értelemszerűen a rendszeren belül.** Ennek van egy egyszerűsített illetve egy bővített kereső nézete. Ez az ILIAS-ban megint szabályozható, hogy indexelten történjen a keresés vagy ne, ez az adatmennyiségtől függ, ami benne van, ezt az admin részről lehet állítani. Bővített keresésnél gyakorlatilag bármire kereshetünk, ami elképzelhető, ide értve a metaadatokat is. Értelemszerűen van egy beépített e-mail rendszer is az ILIAS-on belül. Alapértelmezetten minden tanulónak a felhasználóneve egyenlő az e-mail címével a rendszeren belül. Természetesen ez is kereshető, itt is tudunk körleveleket írni adott csoportnak, vagy az összes felhasználónak egyszerre. Valamint egyedi beállításokra, címjegyzékre is van lehetőség.

**Kukkantsunk be a taneszköz tárolóba, ahol a legtöbb érdekesség lesz számunkra.** A taneszköz tároló gyökerében egyetlen egy típusú elemet tudunk létrehozni, az a kategória. Ez a kategorizálás gyakorlatilag egy linket jelent a rendszeren belül semmi más. A kategórián belül, ha beléptünk egy kategóriába és megvan hozzá a megfelelő jogosultságunk, akkor kitárul a lehetőségeknek a tárháza. Akkor az összes elem feltárul előttünk, amire a rendszer képes. Első körben megmutatom a mi koncepciókat, hogy csináljuk mi, ennek kapcsán megnézzük az egyes funkciókat, de bárhol bármi kérdésetek van, állítsatok meg nyugodtan, és akkor megbeszéljük azt a részt. Első körben nézzünk meg egy teljesen e-learninges képzést. Legyen az, amit legjobban ismerek, mert én is tutorálok ez az egészségügyi alapismeretek. Ugye a rendszerbe belépve, miután felhasználónévvel, jelszóval léptünk be, rögtön megjelenik a csatlakozás lehetősége. Ezt átjavítjuk jelentkezésre, bármire amire szeretnénk és erre kattintva adhatjuk le a jelentkezésünket a rendszeren belül. Kurzusra is számtalan regisztrációs mód lehet, lehet ingyenes, lehet jelszóval védett, lehet kérni kell tagságot és majd az oktatószervező kurzus adminisztrátor engedi be. Illetve adatvédelmi okokból a rendszer tájékoztat, hogy a kurzus adminisztrátor számára milyen adatok lesznek láthatóak, amit a regisztráció során megadtunk, és elfogadjuk, hogy ezek az adatok láthatóvá válnak.

**Milyen struktúrát alakítottunk mi itt ki?** Egyrészt van egy egyszerű mappa struktúra, ahol különféle anyagok érhetőek el. Tananyagok alatt értelemszerűen minden, ami tananyag. Illetve itt még mondjuk egy első generációs eBook formátumú tananyag van, nem online, úgyhogy én már egy online-t fogok mutatni. Általában lehetőséget adunk a tanulóknak, hogy egymással cserélgessenek anyagot. Ez e-learningben kevésbé jellemző, ez függ a kritikus tömegtől, de mondjuk egy hagyományos tanfolyamos képzés támogató helyénél, amit szintén meg fogunk nézni, ott eléggé nagy mennyiségű anyagforgalom van, hiszen, ha belegondolunk, hogy valaki hiányzik egy előadástól, mi szokott azt első gondolata lenni? Az, hogy elkéri mástól. Na most, ha a saját gyakorlatából belegondol mindenki és a saját tapasztalatába, általában évfolyamonként egy-két ember az, akire rájár a rúd, aki értelmesen tud jegyzetelni, meg még olvashatóan is. Mindenki az övéket nyúlja le és hagyja el, meg hát nem tartózkodik nála a jegyzete, hogy tanulni tudjon belőle. Úgyhogy én már voltam olyan képzéseknél, ahol az az egy-két ember, akire rájárt a rúd rutinszerűen szkennelte be az anyagait és toltta fel ide a rendszerbe, hogy addig is hagyják őt békén az évfolyamtársai. De például hagyományos képzéseknél ide szoktuk mi feltenni az előadóknak a prezentációit, az órai prezentációit is, ami segíti a tanulókat. Itt kellene lennie egy olyan mappának, hogy tutori csoportok amiben az összes tutornak az összes tutori csoportja benne van a hozzárendelt tanulókkal és anyagokkal. Tehát itt vannak a tutori csoportok, az itt látható tutorok tutorálnak most nekünk a képzésünkben. A mi koncepciónk az, hogy nem kurzus tutor van, aki az összes tanulót tutorálja az adott képzésen belül, mert ahhoz főállású tutor kellene, hanem mi külsősöket foglalkoztatunk, megbízás alapján és hozzájuk rendelt néhány tanulóról beszélünk. Ha megnézzük az én tutori csoportomat, akkor azonkívül, hogy itt van a tanulók lecke könyve, én feltettem még nekik egy online elérhető anatómiai atlaszt, ami egy webhivatkozás-gyűjtemény és megszürttem azokat az atlaszokat, amik elérhetőek, szerintem ezek jól használhatóak, ezt érdemes használniuk.

Illetve változott a tananyag azóta, mióta felkerült a rendszerbe, változott az újraélesztés protokollja, azt is ilyen módon juttattam el a tanulónak. Hogy ennek kapcsán milyen csiri-birik lehetségesek.

**Ha megnézem a verziókat, itt egyetlen verziót fogok találni, de magában a rendszerben van egy beépített verziókövetés.** Ha frissítem ezt az állományt, az összes további verzió, ami előtte volt, itt marad, itt sorakoznak szépen sorba, bármikor elérhető. Ez mondjuk ha belső képzéseknél használják a rendszert és mondjuk esetleg minőségbiztosítási célokra, oktatásra is, a minőségbiztosítósok ezt nagyon szokták szeretni. Bármikor bármilyen dokumentum korábbi verzióját el lehet érni, láthatjuk, hogy mikor módosult, ki módosította. Az előző példánál maradva, a link gyűjteménynél. Magában a rendszerben van egy olyan funkció, hogy link ellenőrzés. Ez automatikusan is beállítható, tehát a rendszer automatikusan ellenőrzi, hogy élnek-e ezek a linkek, amiket én felvittem, és ha van egy törött link közöttük, ami már nem működik, erről automatikus értesítést tud küldeni és meg tudjuk szüntetni ezeket a törött linkeket.

**Nézzük meg, hogy egy e-learninges képzésnél nálunk mik vannak.** Ugye ezt a csoporttársaimnak segédanyagok kategóriája, már mondtam. Önellenőrzés. Említettem, hogy nagyon fontos, hogy önellenőrzési lehetőségeket adjunk a tanulónak. Ez egy moduláris szerkezetű képzés, amit nézünk, és eszerint vannak kialakítva az önellenőrző feladatlapok is. Azért van egy 1-5 modulós aggregált teszt is, mert az ötödik modul után van egy mérési pont a tanulónak, egy köztes vizsga, és ezért adunk lehetőséget, hogy a teljes öt modulból egyszerre tudják ellenőrizni a felkészültségüket. Még nem beszéltünk arról, hogy milyen típusú kérdéseket támogat a rendszer. Számtalan típusú zárt kérdés van, egyszeres, többszörös választás, mondat kiegészítés, párosítás, sorba rendezés akár képpel, akár szöveggel, de ezt meg fogjuk még nézni konkrétan. Amikor én úgy gondoltam, hogy mivel csak önellenőrzésről van szó, nem vizsgáról, szeretném felfüggeszteni azt a tesztet, mert most éppen nem érek rá többet. Ami a tutornak érdekes, hogy különféle statisztikai adatokat tudok ebből kinyerni. Van egy áttekintő nézet, amikor én látom, hogy ki töltötte ki, milyen átlagos eredménnyel. Megnézhetem az összesített teszteredményeket, amiből minimális statisztikát kapok itt az elején átlagos kitöltési idő, sikeresség, nem sikeresség szempontjából és látok egy kérdésszintű statisztikát, miért fontos ez nekünk? Ha találunk egy olyan kérdést, például van itt egy 33%-os, amire ilyen eredménnyel tudták átlagosan kitölteni, akkor gondolkodjunk el. Ezt nem tanítottuk meg, ezt rosszul kérdeztük meg, vagy netán valamilyen technikai hibát vétettünk felvitelkor, mert ha csak 33%-os válaszolták meg, ott biztos valami gubanc van, az a kérdés nem jó. Ha van olyan kérdés, amire 100% válaszolt, akkor megint gondolkodjunk, hogy érdemes azt a kérdést benne hagyni? Vagy mondjuk azt a kérdést egy feladatlapnak az elejére tegyük, mint motiváló kérdést, de nem biztos, hogy annyira pontosan mér. Tehát visszacsatolási lehetőséget ad nekünk úgy általában az anyagra vonatkozóan.

**A tutor szempontjából, ha én szeretném megnézni egy tanulómat, hogy ő konkrétan hogy töltötte ki azt a feladatlapot, természetesen arra is van lehetőség.** Ilyenkor látok egy, a tanulóval kapcsolatos összefoglalást az adott feladatlap kapcsán, mivel ez egy ellenőrző teszt, ezt többször kitöltheti, tehát ezt összesítetten, az összes kitöltésre vonatkozóan látom, és látom, hogy melyik kérdést, hogy válaszolta meg. És ha szeretném ezeket gyakorlatilag egy nézetben feladatlap szinten is meg tudom jeleníteni, és látom azt, hogy milyen sorrendben jöttek a kérdések, mi volt a helyes válasz és ő mit válaszolt rá, hány pontot kapott.

Vannak olyan kérdések, hogy nyitott kérdések, amit csak manuálisan lehet pontozni, tehát ha úgy hozunk létre egy feladatlapot, hogy nyitott kérdés is van, azt a manuális pontozásnál kell a feladat javítónak felmennie és kijavítani és utána ugyanúgy bekerül értelemszerűen az értékelésbe. Minden elemhez, azaz a feladatlapokhoz is metaadatok adhatóak meg. Említettem, hogy van a metaadatokra vonatkozó szabvány, ez megfelel neki, illetve az ILIAS kezeli a szerzői jogi kérdéseket is. Alapvetően a rendszerbe ezek a licencek vannak felvéve, a creative common-ok különféle változatai, nem véletlenül, mert open source-ról beszélünk, de bármilyen licence konstrukciót felvihetünk, az adminisztratív részen és az akkor választósként itt megjelenhet. Számtalan

korlátozást életbe léptethetünk mi egy tesz kapcsán. Például korlátozhatjuk azt, hogy mennyi időt lehet kitölteni, anonim vagy nem anonim, összességében mi az az időintervallum, amiben ki lehet tölteni. Random kérdéssorrend vagy nem random kérdéssorrend, mit lásson a kérdésekből a tanuló. Ha vizsgát szervezünk, akkor korlátozhatjuk, hogy a tanuló egyszerre csak egy ablakban lehessen bejelentkezve. Az az igazság, hogy ha az összes funkciót fel akarnám az összes funkció kapcsán sorolni, akkor akár napokig is itt ülnék, igazából itt is ízelítőket tudok adni.

**Ugorjunk vissza a kurzusba, amiben tallóznak és ennek kapcsán nézzük a lehetőségeket.** Lehetőség van felmérések készítésére a rendszerben. Ugye ami kifejezetten angolul survey, magyarul nem is tudom, talán kérdőívként tudnám lefordítani. Például képzés kapcsán kiváló példa erre, egy tanulói elégedettséget mérő. Ezek egy szintén lehetnek anonimak vagy nem anonimak, és ezekről is részletes statisztikai kiértékelést kapunk, ennek már van egy grafikus kiértékelése. Azt még nem mondtam, de bármilyen adatot a rendszerből általában CSV vagy XLS formátumban ki tudunk nyerni ki tudunk exportálni, ha bárhol máshol szeretnénk használni. Felépítettünk még képzésenként egy külön fórumot az oktatásszervezésre, külön egy fórumot a tutori konzultációról, a szakmai formációk mozognak.

A kávéházat ajánlom mindenki figyelmébe, aki esetleg e-learning rendszert használ. Ez egy olyan fórum, amit nálunk kávéháznak hívnak és megjelenik minden sarkában a rendszerben. Ebbe a rendszerbe jelenleg van 2500 tanulónk összesen. Ez már egy olyan kritikus tömeg, aki szeretne egymással adott esetben gondolatokat megosztani, megvitatni és itt, a kávéház fórumban adunk erre teret.

Ha online kommunikációra van szükség, arra a chat áll rendelkezésre értelemszerűen a rendszeren belül. És a weboldal-gyűjteményt azt már mondtam. Nézzünk meg mondjuk egy tananyaggyűjteményt, hogy mondjuk hogy néz ki ebben a rendszerben egy online tanulható tananyag. És ez pedig egy elsősegélynyújtás tananyagminta lesz.

**Tananyagnál is többfajta prezentációs módot választhatunk, egy ablakos, két ablakos, három ablakos.** Az a gond, hogy itt a részletekbe nem nagyon tudunk belenézni. Nézzük meg azokat az elveket, amikről beszéltünk. Maga a rendszer egyébként liquid layout-os, tehát alkalmazkodik a felbontáshoz attól függően, hogy milyen felbontásba nézzük, úgy látjuk. Mondjuk 1024x768 az a minimális felbontás, amiben érdemes használni ezeket a rendszereket. És a tananyagot is úgy optimalizálni, hogy minimum annyiban nézzék. Alapvetően képernyőképekben gondolkodunk, ami belefér egyszerre, azt az információt juttassuk el, és ami az e-learning savát-borsát adja, a különféle média elemek. Ha nem teszünk médiaelemeket a tananyaghoz, akkor minek használunk e-learninget? Akkor semmilyen pluszt nem adtunk jóformán. Ezek a média elemek megjelenhetnek beágyazottan, mint láthatjuk, érdemes feltenni egy nagy felbontásban megnézhető verziójukat, ha van annyira részletgazdag a kép.

Mint mondtam nem kell spórolnunk nekünk a papírral, tehát ha 3 mondat és 2 kép mutat be nekünk valamit jól, akkor csak annyit kell odatennünk, nem érdemes többet. Természetesen nem csak a képekben merül ki nekünk e rendszer tudása, hanem más médiaelemet is beágyazhatunk, például itt adott esetben egy videót. És ilyenkor a videó automatikusan elindul nekünk, ezt ugye számtalanszor vissza lehet tekerni, meg lehet nézni, itt a lehetőség rá. Tehát ilyen elveket követünk, de az elvek az egy más kérdés, másfajta koncepciót is követhetnénk.

**Nézzünk egy olyat, hogy nyomtatás nézet, megmondom, hogy miért.** Amikor elindultunk mi egy első generációs rendszerrel, nagy megrökönyödésünkre a tanulók első kérdése az az volt, hogy hogy lehet kinyomtatni a tananyagot? Ugye nem igazán ez a cél. Az a cél, hogy online tanulják meg, mert akkor kapunk mi információt az előrehaladásról, viszont azt kell mondjam, hogy az egy jogos igény, hogy ha mindennap utazik egy órát a munkahelyére, akkor ki tudjon nyerni a rendszerből valamit, amit útközben tanulhat. Tehát ebben van ráció. Lehetőséget adhatunk, hogy a tanuló kijelöl akár a teljes tananyagot, akár kijelölt részeket és abból kér egy nyomtatási képet, amely értelemszerűen már úgy lesz tördelve, hogy az egy A/4-es papírra normálisan nyomtatható legyen.

Mondjuk a videót az A/4-es papíron nem fogja tudni megnézni, de ebbe a tördelt nézetbe át tud kerülni.

**Ha megnézek egy hagyományos képzést támogató helyet egy villanásra.** Mondjuk egy gyógyszerügyi asszisztens. Milyen élet lehet a rendszeren belül? Ugye itt van az óravázlatok mappája, ahová felkerülnek azok az anyagok, amit az előadók adnak át. Itt az előadó nevével fémjelzett mappába letölthetők azok az anyagok, amiket az előadó prezentált az előadás során. Illetve, ha megnézzük azt a könyvtárat, ahol az emberek tudnak egymás között cserélgetni, hát itt speciálisan nem volt nagyobb élet, de tudok olyat mutatni, ahol százas mennyiségűt cserélgettek már ki egymás között a tanulók. Az az igazság, hogy csoportfüggő. Ha az elején rájön a csoport ennek a használatára, akkor kötetlenül használják.

**A rendszer képességeiről.** A tananyagszerkesztő rész gondolom az többeket érdekel. Nézzük meg azt, hogy lecke-könyv. Van egy elég jól használható, egyszerű beépített szerkesztő felülete a rendszernek. Ezek a paraméterek, amiket mondtam, hogy hogy jelenjen meg, milyen formátumban. Magát a tananyag megjelenését CSS szabályozza, arra is van egy szerkesztő felület, de bármilyen más CSS-t is feltölthetünk hozzá. Attól függően, hogy szeretnénk-e Javat használni vagy nem használni, azt beállíthatjuk, a Java a drag and drop-hoz szükséges. Gyakorlatilag a fogd és vidd művelet bele van építve a rendszerbe. Tehát ha én ezt képet át szeretném helyezni máshová, azt drag and drop-pal is megtehetem.

Be tudunk szűrni szöveget, médiát, listákat, táblázatokat tudunk ilyen módon kezelni, vágólapról betenni. Ha egy médialemez szeretnénk betenni, annyi, hogy kiválasztjuk, hogy média beszúrása, oké, majd kapunk egy egyszerű párbeszéd panelt, ahol ki tudjuk választani azt az állományt. Vagy, ha egy webes, máshol érhető elegy webes hivatkozást tudunk megadni, át tudjuk méreteztetni, illetve ha meg tudjuk adni a szabványnézetet, ami alapértelmezetten megjelenik illetve azt a nézetet, ami a kis nagyító ikonra kattintva meg fog itt nekünk nyílni. A rendszeren belül van itt, hogy választás média gyűjteményből.

Tudunk definiálni média gyűjteményeket, ahol ezeket a média gyűjteményeket felhalmozzuk és csak hivatkozva vannak a tananyagokba. Miért jó ez? Mondjuk van egy képünk, egy szív ábránk, ami szerepel 23 tananyagban. Ha valamit javítottunk ezen az ábrán, kijavítjuk a média gyűjteményben, onnantól kezdve mind a 23 tananyagban frissült.

**Nézzük meg, hogy még milyen elemeket tudunk létrehozni és milyen módon.** Ha itt a gyűjteményes elemek közé bemegyünk, itt van a legtöbb. Például nézzünk meg egy kérdés-gyűjteményt. Úgy épül fel az ILIAS logikája, hogy először létrehozunk egy kérdés-gyűjteményt, vagy kérdés-gyűjteményeket, amiből majd fognak táplálkozni akár az önellenőrző tesztek, akár az online vizsgák. Egyre több kapcsolat van, tehát egyetlen gyűjteményből ki tudunk szolgálni számtalan tesztet, vagy egy teszt számtalan gyűjteményből táplálkozhat. Például vagyunk egy iskolában és vannak a szaktanárok. Minden szaktanár a saját témakörében készíti el a gyűjteményeket. Mondjuk a biológia tanár azt mondja, hogy állatok, növények, ezeken belül is még csoportosíthat a több szakra is. És ha érettségire akarunk felkészíteni, akár az összesből összerendezhetjük, de ha az adott tanár csak egy röpdolgozatot szeretne írni, akkor ő csak a saját kérdés-gyűjteményéből fog dolgozni. Egy ilyen kérdés-gyűjteményben itt egymás alatt sorakoznak a kérdések. Látjuk a kérdés nevét, látjuk a típusát, látjuk, hogy hány pontos, illetve egy összesített statisztikát is megnézhetünk arról a kérdésről, hogy akár hány helyen szerepel és akárhányan kitöltötték az itt fog szerepelni benne, ez nekünk visszacsatolás lehet a fejlesztéssel kapcsolatban.

A szerkesztés az megint egy nagyon egyszerű felületen történik. Gyakorlatilag vannak kötelező paraméterek, amiket megadhatunk, van egy beépítve szövegszerkesztőnk a rendszerben, megadhatunk hogy keverték legyenek-e a válaszok vagy ne. Ez egy kiváló funkció. Mindenki ismeri azokat a teszt-gyűjteményeket, ahol már így kitöltik a tanulók, már tudják, el se olvassák a kérdést, csak tudják, hogy a harmadik. Két olvasás után simán megoldható. Itt kevert válaszokkal ez kiküszöbölhető. Illetve egyszerűen hozzáadhatunk egy új definíciót, ha szeretnénk. Ami a varázslat,

inkább azt mondom, az az, hogy az önellenőrző kérdést, vagy egyáltalán a kérdést össze lehet rendelni tananyaggal, a tananyag egy lapjával, fejezetével, a fogalomtár egy magyarázatával, és ha a tanuló rosszul töltött ki mondjuk egy önellenőrző kérdéssort, akkor a kérdéssor végén már rögtön ott van neki a link, arra a helyre, ahol a jó választ megtalálja, rögtön meg tudja tanulni, hogy mi lett volna számára a jó válasz.

**Milyen típusú kérdéseket támogat az ILIAS?** Ugye ezek a lehetőségeink. Az esszé kérdés egyértelmű, a Java outlet kérdés, a Java-val gyakorlatilag bármit meg tudunk csinálni, tehát ha valami nagyon csili-vilit kell csinálni, azt Java programozással tudjuk megoldani. A képtérképes kérdés az azt jelenti, hogy feltölthetünk egy képet, amin kijelölünk aktív területeket és azt kell választania a tanulónak. Például azt mondjuk, hogy feltöltünk, hogy a saját szakmámról beszéljek, egy képet a koponyáról és azt mondjuk, hogy válassza ki a homlokvarratot, és ha jó helyre kattint, akkor megkapta a pontot, ha nem, nem. Mondat kiegészítés, párosítás ez lehet képet képpel, képet szöveggel, szöveget szöveggel. Sorrendberakásnál is ugyanúgy kép-szöveg felmerülhet. Egyébként bármilyen kérdéshez tudunk csatolni médiaelemet. Tehát csinálhatunk egy olyan egyszerű választásos kérdést mondjuk egy nyelvtanfolyamnál, hogy hallgassa meg azt a hanganyagot, ami ott van a kérdésnél és utána válassza ki a jó megoldást, hogy mit mondott. Számszerű kérdésnél egy intervallumot vagy egy konkrét számot várunk, az egyszeres, többszörös választás szintén lehetséges. Még két dolgot mutatnék meg.

**Tudunk létrehozni azonkívül, hogy kurzusokat, csoportokat, chateket, tudunk RSS feed-et generálni a rendszerből.** Az RSS feed-re rá lehet ültetni különféle média elemeket, média cast-okat, akár hangot, akár videót, beépített flashes lejátszó segítségével Youtube-szerűen tudunk videókat eljuttatni a tanulónak. Tananyag szempontjából van egy szabvány tananyagformátuma az ILIAS-nak, amit láttunk, de ugyanúgy HTML, form-mail CC anyagokat is tud fogadni. És ami a nagy varázslat, hogy ezt egymás között tudja konvertálni, tehát, ha az ILIAS formátumába megcsinálunk egy tananyagot, egy gombnyomásra SCO csomagot exportálhatunk belőle, egy gombnyomással HTML anyagot exportálhatunk belőle. Ez miért jó? Csinálunk mondjuk egy offline verziót, megcsináltuk az online tananyagot, azt mondjuk, hogy HTML tananyag export, rádobjuk egy CD-re, csinálunk egy előlapot és ott van az a verzió, amit elvihet a tanuló magával oda, ahol nincs internet kapcsolata.

**Fogalomtárakat definiálhatunk, ahol megmagyarázzuk a fogalmakat.** Egy fogalomhoz több magyarázat tartozhat. A rendszeren belül bármit linkelhetünk. A tananyagba linkeljük a fogalom magyarázatát, arra kattintva azonnal megjelenik. Ugyanúgy webes külső-belső hivatkozásokat létrehozhatunk, tesztek, gyakorlatokat, kérdőíveket.

**Egy képzésnek az adminisztratív felületére vessünk egy pillantást.** Láthatjuk azt, hogy pillanatnyilag kik vannak a képzésen belül, kik a jelentkezők, ha ilyen regisztrációs módot választottunk, azonnal e-mailt küldhetünk, az egyes embereknek innen, erről a felületről. Ha többet jelölünk ki, úgyis tudunk kör e-mailt küldeni az összes embernek egyszerre, ah ki szeretnénk értesíteni őket. Ezeket az adatokat természetesen ahogy mondtam ki lehet exportálni. Definiáltunk bejárési utakat. Ha azt szeretnénk, csak akkor mehessen a második modulra, ha az előző tesztet sikeresen kitöltötte, megköthetjük. Tehát akár konkrét bejárési utakat lehet így létrehozni, ha nem akarunk olyan szabadságot adni.

A tanulási célkitűzéseknél lehetőségünk van arra, hogy létrehozzunk célkitűzéseket, azokat összerendeljük önellenőrző kérdésekkel, tesztekkel és tananyagokkal. Itt pontosan nyomon lehet követni, hogy ki hol tart, ki mennyit sajátított el miből.

Az adminisztratív részből pedig csak egyetlen egy dolgot mutatok meg, ami azt mondom az egyik nagyon nagy előnye, ennek a rendszernek, a szerepek. Gyakorlatilag szerep alapú jogosultsági rendszere van ahol létrehozhatunk mi globális szerepeket, lokális szerepeket, amik csak bizonyos területekre érzékenyek, ezek érvényesek lehetnek automatikusan generáltak. Így lehet azt nagyon

egyszerűen megcsinálni, ha valamit el szeretnénk dugni a tanulók elől, de az adminisztrátoroknak látniuk kell, akkor azt mondjuk, hogy például itt van ez a háttér rendszer, amit én látok, mint adminisztrátor, de a tanuló nem, így azt mondtam, hogy mint adminisztrátor én mindent látok, vannak olyan felhasználók, akik benne vannak ebben a helyileg generált szerepben, és ők is láthatják, olvashatják, a többiek pedig nem látják ezt, mert nincs rájuk szükségük. De ez gyakorlatilag a rendszer összes szintjén működik, én még nem talákoztam olyan vad igénnyel amit ne lehetett volna megoldani ezzel a jogosultsági rendszerrel. Ezáltal lehetőség nyílik arra, mivelhogy nem beégetett szerepek vannak teljesen testre lehessen szabni bármilyen szervezetre és annak a működéséhez igazodjon.

**Hallgató:** módszertani, felhasználói tanári kézikönyvnek, felhasználói kézikönyvnek elérése létezik-e ebből a rendszerből?

**Előadó:** mármint módszertani útmutató?

**Hallgató:** tehát olyan, hogy installálás, telepítés, használat.

**Előadó:** van angol és német nyelvű kézikönyv, amit a fejlesztők csinálnak, ez ILIAS formátumú, tananyag formátumban a fejlesztők honlapjáról le lehet tölteni és be lehet tenni. Magyar nyelvre ez a kézikönyv nincs lefordítva sajnos, viszont, ha megengeditek, akkor annyit hadd mondjak, láttam itt egy-két szórólapot már az embereknél. Én foglalkozok ezeknek az ILIAS alapú rendszereknek a telepítésével, vezetésével és nem csak technikai szempontból, hanem teljes módszertani palettát is tudunk nyújtani hozzá. Módszertani képzéseket tudunk biztosítani. De konkrét, írásos, nagyobb terjedelmű anyag erre nincs pillanatnyilag magyar nyelven.

**Hallgató:** a hallgatók monitorozására ezek szerint működik, mennyit, mikor lépett be, meg hogy áll. Az érdekel, hogy mondjuk egy felsőoktatás e-learninget szeretne, és egy megbízásos szerződéssel foglalkoztatott oktatót foglalkoztat, arra van-e lehetőség, hogy az oktatót monitorozza?

**Előadó:** a felhasználó az felhasználó. Az, hogy oktatóként lép be csak egy szerepet jelent. Az az igazság, hogy számtalan olyan funkció van, amit nem érintettünk, de úgy gondoltam, hogy inkább a fő csapást nézzük meg első körben esetleg ezt majd jövőre folytathatjuk, de ez már más kérdés.

**Hallgató:** a Moodle bizonyára elég elterjedt a felsőoktatási intézményekben?

**Előadó:** gyakorlatilag az a legelterjedtebb e-learning rendszer a felsőoktatásban. Kereskedelmi terméket is használnak egy-két helyen, de annak olyan óriási költségei vannak, hogy nem nagyon mennek bele. ILIAS-ra is van példa a felsőoktatásban, egyébként, ha megnézzük egy picit a pénzügyi oldalát ennek a történetnek, ennek az e-learning rendszereknek, hogy mit lehet most felvenni a piacon. Vannak a nagy multi gyártók által gyártott e-learning rendszerek, Oracle, IBM, Microsoft. Közös jellemzője mindegyiknek, hogy óriási szoftver és kapcsolódó szoftver költsége van licenc szinten, ha ezt valaki be akarja ezt vezetni, ezt csak az igazán nagyok engedhetik meg maguknak. Vannak a magyar fejlesztésű kereskedelmi e-learning rendszerek, itt is meglehetősen, hát sokat nem, de egy 4-5-öt könnyedén lehet találni a magyar piacon. Ez egyrészt amellet, hogy fizetős vannak a járulékos szoftverköltések, a Windows, az adatbázis, egyéb, amit említettem. Nyíltan és kimondottan ezek a cégek sem tudják követni tudásban azt, amit az open source rendszerek tudnak. Miért? Ha megnézzük a magyar piac kicsi piac. Egyik magyar cég sem tud akkora bevételt realizálni, hogy ugyanazt a tudást, hozzá tudja fejleszteni a saját rendszeréhez, mint amit az open source rendszerek tudnak. Ezért gyakorlatilag mindegyik választott valamilyen specializációs irányt és abban az irányban fejleszt.

A másik koncepció, amellet, hogy fejlesztik a rendszert kisebb gözzel, főleg a tananyagkészítésből élnek meg nagyobb részt ezek a cégek. És a harmadik, ami fellelhető ugye azok az open source, nyílt forráskódú rendszerek, ahol gyakorlatilag szoftver szinten befektetés az elején nem szükséges. Befektetés annyi szükséges, hogy általában ezeknek az üzembeállításához nagyobb szakértelem szükséges, tehát van egy egyszeri költség, viszont hosszabb távon megint nincsenek licence költségek. Ugye bármilyen kereskedelmi rendszerrel megjelennek tovább a licence költségek. Frissíteni kell az operációs rendszert, az adatbázisrendszert, meg kell venni az új verziót az

e-learning rendszerből. Az open source-nél ezek a hosszú távú költségek abszolút nem fognak jelentkezni, csak egy üzemeltetési költség, ha külsőssel csináltatja az ember.

# Németh László: A legcsodálatosabb oktatóprogram: a 40 éves Unix programozási környezet I

40 éve annak, hogy elindult a UNIX programozási környezet a maga útján, mégpedig azzal, 1969-ben egy hobbi fejlesztésként létrejött az alapja, a UNIX rendszermag és utána a következő években fokozatosan kiterjedésedett ez a rendszer. Arról majd el fog hangzani, hogy 1979-ben jelent meg a 7-es változata a UNIX-nak és erre mondták, hogy ez jobb, mint, ami korábban az összes UNIX volt, amik utána születtek, azoknál is jobb, tehát ez az etalon, amit azóta sem sikerült utolérni, sőt inkább távolodnak tőle a UNIX-ok. Hogy miért? Azt meg fogjuk nézni az előadás során.

Kérem a jelentkezőt, akinek felvesszük a hangját, ezzel azt akarom szemléltetni, hogy milyen örömet okoz a Linux operációs rendszer. Az lett volna a trükk, azt szemléltette volna ez a rendszer, hogy mire képes a Linux egy drága hi-fi meg elektronikai berendezések helyett. Azt a bemenő hangcsatornát én át tudom alakítani és elvileg az előadók, akik nagyon odafigyeltek és képesek voltak ezt legyűrni. Nem grafikus felületen vagyok, akkor megmutatom, hogy miről is lett volna szó itt. Az látszik, hogy a „rec” parancs, itt meg a másik a „play”, meg tudja csinálni, hogy a bemenő hangcsatornát átkapcsolom a kimenőre, átjátszom, egy kis átalakítással és elvileg ez lesz itt a következő és megmutatom, hogy mit hallottak az előadók és mit tūrtek itt hősiesen. Ha lassabban beszélek, akkor talán jobban lehet hallani, hogy az egyszerű magnó PC jobban megbirkózik azzal a feladattal, hogy a bemenő hangjelet egy digitális szignálfeldolgozó algoritmussal felbontja frekvenciákra azt átalakítja, majd újra kiteszi a gombot, ezért van látencia, késedelmi idő, mert ezt ablakkal oldja meg a feldolgozó algoritmus. Majd nézünk a következőkben egy példát, ahol nem lesz ilyen szünet, viszont nem lesz ilyen szép sem.

Azt szerettem volna ezzel demonstrálni itt, hogy igazából nagyon egyszerű módszerekkel lehet érdekes kísérleteket vagy látványos informatikai feladatokat megoldani a Linuxszal, egy UNIX típusú operációs rendszerrel. Maga ez a parancssor ránézésre bonyolult, de ahogy mondtam de igazából csak az van, hogy elindítjuk a „rec” nevű kis programot a sox nevű fejlesztésben, programcsaládban található ez a két alkalmazás. Felveszi a hangot, ezt alapesetben kiadja a standard kimenetén, amit vagy át lehet irányítani valahova, vagy akár lehet fájlba lehet irányítani. Mi most ezt nem irányítottuk sehova, illetve mi adjuk meg, hogy WAV formátumban, ez a típusa a hangfájlnak, küldje ki a standard kimenetre. Ez általában UNIX alatt ez sima, igazából ez azt jelenti, hogy a névnek a hiánya az alapértelmezett kimenetet szokta takarni a különféle programoknál. Tehát arra a kimenetre fogja átnyomni a felvett hangot. Megkapja a „play” program, aminek szintén megmondjuk, hogy a bemenete a standard bemenet legyen, tehát onnan átveszi ezt az adást. Itt van egy kis segédprogram, hogy minél kisebbre vegyük azt az ablakot, hogy ne legyen akkora szünet ahogy én beszélek és közben megjelenik a hang a kimeneten. Ez a legkisebb megadható ennek a sox alkalmazásnak, itt a trükk az, hogy azt mondjuk csökkentjük a frekvenciát, legyen mélyebb az adott hang, ha éppen meg elhagyja a mínuszt a hang, akkor magasabban lehet hallani.

Tehát itt van egy olyan sor, amivel szerintem nagyon jól lehet vagy látványosan lehet demonstrálni, hogy milyen ereje van a UNIX-ban az egyszerű kis alkalmazások összekapcsolásának. És itt a lényeg, amiről beszélni fogunk, hogy az összekapcsolás ezzel az egyszerű jellel, a „cső” jellel történik, ezzel a pipe jellel és én ezt gyakorlatilag tovább fűzhetem, adhatok még további alkalmazásokat is egymásnak, amelyek itt szűrőként feldolgozzák az adatfolyamot, amit megkapnak.

Ez annyira beleépült a UNIX programozási környezetbe, hogy az eredeti fejlesztők közül sokan azt mondják, hogy tulajdonképpen ez a legfőbb ismérve a UNIX rendszereknek, az a UNIX filozófia lényege, hogy ilyen kis pici programok, ha együtt képesek működni, és itt nagyon együtt képesek működni, akkor valami egészen különleges dolgokat lehet vele létrehozni. Ez csak egy példa volt,

hogy meg lehet ezt játszani a hanggal, ez elég könnyen elérhető alapértelmezetten része minden Linux disztribúciónak ez a sox rendszer, lehet telepíteni egy pillanat alatt.

Először a Linuxsal ismerkedtem meg, de rögtön kiderült, hogy, ami jó a Linuxban az megvolt korábban is, és ez volt a UNIX programozási környezet. A háttérben egy archív fotó látható, ez a név is utal arra, hogy negyven év az elég régen történt. A szereplők azok még most is élnek, ők a UNIX alkotói, igazából, akit itt ki szoktak kiemelni, az Ken Thompson, aki egy ilyen Teletype rendszernél ül, és éppen a UNIX-ot írja, annak nem a legkorábbi, hanem egy későbbi változatát. Itt pedig Dennis Ritchie látható, aki mosolyogva nézi. A gép pedig, ami ott áll mögöttük az egy Digital gép a PDP 11-es változata. Nem ezen jött létre a UNIX, hanem egy kistestvérén a PDP-7-esen, ez egy '73-as fotó azt hiszem.

Ezen a fotón is látszik, hogy nem voltak még nagyon ellátva monitorokkal, meg ilyen képernyőkkel, ez a Teletype annyit csinál, hogy amit begépel az ember ilyen írógép billentyűzeten, azt elküldi ennek a szekrénynek, az elvégzi a megfelelő műveleteket, igazából meg fogjuk látni, hogy egy alkalmazás kapja meg a jeleket, amit héjnak vagy shell-nek hívnak itt a UNIX-ban, és az fogja azt majd továbbadni, ha szükséges más programoknak. És a héj az, ami visszaszolgáltatja, visszaküldi az üzeneteket, az üzenet kinyomtatódik egy papírhengerre, amit pénztárgépeknél gyakorta láthatunk, talán ugyanazzal az automatikával. Nem tudom digitális pénztárgépek megjelentek-e akkor már, de azzal megkapja az üzenetet és egy 30 karakter/perc sebességgel, talán még az első időkben ilyen volt '69-ben, kinyomtatja a választ, ha éppen van válasz. Mivel ez olyan csiga-lassú volt, ezért a UNIX-ot még úgy fejlesztették még a legelején egy még lassabb gépen, hogy ha minden jó, akkor nincs válasz, ami éppen ellentétes a grafikus felületeknek a filozófiájával, ami azt mondja, hogy mindig erősítsük meg a felhasználót. Hogyha valami működik, ha ott nem kapunk választ, akkor ott esünk kétségbe. UNIX-nál az van, hogy ha nincs üzenet, akkor a dolgok rendjén mennek. Tehát ez mintha ellentéte lenne, de meglátjuk, hogy igazából nem ellentétes a két megközelítés egymással.

Dennis Ritchie-nél nem említettem meg, hogy miről híres, igazából a C programozási nyelvről. Amellett, hogy a UNIX-nak a fejlesztésből is alaposan kivette a részét.

**Nézzük 1969-et. Ahol szerepel három pont, a legfontosabb ismérve a UNIX-nak.** Az, hogy több feladatos, tehát egyszerre képes több programot párhuzamosan futtatni, ezt időosztással oldja meg. Felosztja a számítási időt a programok között, a futó programokat itt a UNIX alatt folyamatoknak vagy processzeknek (processzus magyarul) nevezzük. Ezzel a rendszerrel azt is meg lehet oldani, hogy a programok más-más felhasználóhoz fordulnak, más-más felhasználótól érkező bementet dolgoznak fel, akkor gyakorlatilag azt fogják látni, hogy meg lehet oldani, hogy több felhasználós legyen a rendszer, többen tudják egyszerre használni a gépet. A PDP-7-es esetében ez már megvalósult, hiszen hozzá lehetett csatolni kettő Teletype-ot. Maximum kettő Teletype-ot, igazából ez két felhasználós rendszer volt, azt lehet mondani, de persze úgy írták meg a rendszert, úgy írta meg Ken Thompson, hogy ne kottyanjon meg több felhasználó sem. Amit hozzá lehet tenni, hogy korántsem volt ez triviális feladat, olyan operációs rendszert fejleszteni, ami megbirkózik több felhasználóval, és ezt kellő hatékonysággal teszi. Pár évvel előzte meg a DTSS rendszer a UNIX-ot, ami annyiban érdekes, hogy ott a felhasználó egy olyan héjat, illetve nem nevezhetjük ezt héjnak, mert a héj, az egy külön folyamat, egy processzus, amit indít az operációs rendszer. Az operációs rendszer biztosított egy olyan keretet, hogy úgy érezte a felhasználó, hogy csak vele foglalkozik a rendszer, ami minden ilyen több felhasználós rendszerre igaz, viszont lehetőséget nyújtott arra, hogy egyszerűen lehessen programozni. Tehát a DTSS rendszer, az még rendelkezett egy ilyen keretrendszerrel, ami programok írását tette lehetővé, igazából egy egyszerű, de barátságos szövegszerkesztő volt. Amivel talán találkoztak sokan, az nem más, mint ez a felület volt a BASIC-nek a programozási interface-e. Tehát a BASIC az úgy nézett ki, számokkal kellett szépen beírni a sorokat, látom ez ismerős, kilistázhatjuk a programunkat, megvan a program, elindíthatjuk, és akkor kiírja, hogy „Helló világ!” És lehet trükközni, tehát mondhatjuk azt, hogy „20 goto 10”. A BASIC régen a DTSS rendszernek a felhasználói felülete volt.

Onnantól kezdve, hogy a BASIC-et elkezdtek a mikroszámítógépekbe beépíteni, illetve leutánozni, átvették ezt a felhasználói felületet, mert kényelmesnek találták. Gyakorlatilag a DTSS-en ugyanilyen formátumba írták be a Fortran programokat is. És csak akkor éppen úgy indították be, hogy a Fortran fordította le a programot és utána elindította a lefordított programot. Láttuk, hogy ez jó, ha egyszerre többen tudják használni a számítógépet. A DTSS esetében még a környékbeli középiskolákba is bekötötték a rendszert, azok is hozzáférhettek, programozhattak, tehát már a kezdetek kezdetén benne volt ez a pakliban, illetve az is kifejezett cél volt, hogy oktatási rendszer legyen, tehát meg lehessen tanulni a számítógép programozást. Kemény János nevéhez fűződik ez, az egyik fő fejlesztője magának az operációs rendszernek illetve magának a BASIC programozási nyelvnek is. Tehát az kifejezetten egy kezdő számára készült. A UNIX az viszont nem annyira oktatási rendszernek indult, hanem olyan rendszernek, amit szerettek volna, hogy igazából működjön.

**Gyakorlatilag egy kudarcból született a UNIX.** Egy sokkal nagyobb szabású projekt a Multics nevű projekt félig-meddig kudarca, ha kudarcnak nevezzük azt, hogy úgy tervezték a rendszert, hogy 300 felhasználó is tudja használni az adott gépen, amire fejlesztették. Sokkal komolyabb volt a cél gép, mint a PDP-7-es, és a végén örültek, ha 30 felhasználó tudta használni, de már a '70-es évek elején jutottak el oda, ha jól tudom. Tehát három nagy, konzorciumként fejlesztette ezt a Multics rendszert, és az egyik fejlesztő volt Ken Thompson, az AT&T Bell laboratórium részéről, amit ő csinált a Multics-ban, az a fájlrendszer. Egyike az első hierarchikus fájlrendszereknek, tehát amint az előbb beléptem a „cd” parancsal munka/lock könyvtárba, az még a Multics rendszernek a parancsa, vagy a listázás, az „ls” parancs, az is a Multics rendszerben jött létre. Volt egy-két olyan dolog, mint például a pl1 magas szintű nyelven tervezték, hogy elkészítik a Multics operációs rendszert, ami jó ötlet volt, de igazából az implementációs problémák miatt nagyon erőforrás igényes volt ez a megvalósítás, szemben az addig bevett fejlesztési nyelvvel, ami nem más, mint a majdhogynem a gépi kód, az Assembly, ami az olvashatóvá tett gépi kód tulajdonképpen. A DTSS is abban készült, így nem lehet nagyon csodálkozni, hogy nem volt valami hordozható a rendszer. A BASIC-et nagyon sok helyen megtaláljuk még, az utóélete megvan még. A DTSS-t sokáig működtették még, de nem nagyon élt meg sok hardvert, tehát ott is maradt. A UNIX sikere az lett, hogy megoldották, hogy hordozható legyen, portábilis a rendszer.

Meg szokták vádolni a Ken Thompsont, hogy azért készített egy operációs rendszert, hogy tudja futtatnia a játékprogramjait. Egy space travel programot írt, amivel gyakorlatilag a Naprendszerben lehetett utazni, és a bolygónak a gravitációját modellezte le. Látszott a képernyőn, hogy forognak a bolygók a Nap körül, egy kis űrhajóval le lehetett szállni a különböző objektumokra, a Holdra, a Marsra. Az egyik ilyen érdekesség, hogy a Marsnak a holdjára, a Phobosra, ott egész trükkös volt leszállni, mert gyakorlatilag a lebegőpontos aritmetikát is neki kellett megvalósítani. A PDP-7-esen ez volt az egyik ilyen feladat, hogy egyáltalán tudjon ilyen számításokat végezni, tehát fizikai modellezést létre tudjon hozni. Tehát úgy nézett ki ez a Space Travel játék is, hogy a képernyőn futott a levegő pontos aritmetikának a debugere is pontosan oda volt irányítva és látszott, hogy mit számol a program miközben a bolygók is ott keringtek és lehetett irányítani billentyűvel a játékot. Lehet, hogy rájuk is szóltak, hogy ne nagyon játszanak, mert korábban komolyabb gépeken is futtatta Ken Thompson és úgy hívták ezt a játékidőt, hogy funny-money, mivelhogy komoly pénzekbe került, hogy vidám játékkal töltötték az időt. Valami 50 dollár volt a gépidő kiszámolva és ennyit tudtak eljátszani. Aztán ahogy felfedezték a PDP-7-est, amit annyira nem használtak, először assemblert készített rá, ami lefordította az Assembly kódot, megismerte magának a gépnek a lelkét Ken Thompson.

Utána, amint elkészült a játék meg egy-két kisebb könyvtár, akkor arra gondolt, hogy átviszi a fájlrendszert, amit fejlesztett, ezt a Multics fájlrendszert, és akkor lehetne csinálni egy operációs rendszert és meg lehetne csinálni azt, ami nem sikerült nagyban sok ember által, egy egyemberes fejlesztésként. Nyáron, amikor a felesége hazautazott a szüleikhez, mert várandós volt, addig úgy

gondolta, hogy neki a legjobb helye, hogy ha nem zavarja most a feleségét, aki az utolsó hónapokban van, helyette elkészíti a UNIX-ot. Tehát már nem is olyan szép.

**1970-ben ez a UNIX epocha a UNIX korszak 1970. Január 1-jén kezdődött.** Gyakorlatilag az eredeti UNIX egy szó hosszan tárolta, 32 biten tárolta az eltelt másodperceket, illetve előjeles volt ez a változó. Ezért 31 bit volt, ez a '70-től eltelt másodperceket mutatja, elvileg korábbi időpontokat is lehet írni, nem tudom ez mennyire volt kihasználva a rendszeren.

Aztán ebben az évben kifejlesztettek egy komolyabb gépet a PDP-11-est, ami már nem 18 bites gép volt, hanem 32 bites és nem csak két Teletype-ot lehetett hozzáakasztani, tehát ilyen szempontból tényleg komoly volt az előrelépés. De ezt csak úgy tudták elérni, hogy mondogatták, látták, hogy gond van a szövegfeldolgozással a vállalatnál, ők majd készítenek egy komoly szövegszedő programot, ami mennyivel termelékenyebbé fogja tenni a munkát és így is lett. Ahogy 3 gépirónó gépelte és szerkesztette a ezen a rof nyelven a különféle szabadalmakat, a nyomtatás, az archiválás előkészítéséhez, látta a vezetőség, hogy ez jó dolog és szükség van rá, úgyhogy még egy PDP-11-est kaptak, amivel már 256 Mbyte memóriát tudtak megcímezni. Ez egészen komoly érték volt. Lehet, hogy ez nem Mbyte, hanem kilobyte. Valószínűleg kilobyte lesz, mert a PDP-7-esnek volt olyan korlátjai, hogy a futó program nem lehetett 8 kilobyte-nál hosszabb.

**A UNIX Assembly nyelven készült, gyakorlatilag gépi kódban, akkor mitől beszélhetünk mégis operációs rendszerről?** Onnan, hogy volt egy meghatározott programozási felülete, és ezeket rendszerhívásoknak hívjuk és ezeknek egy olyan készletét biztosította, ami lehetővé tette, hogy utána könnyebben fejlesszen az ember Assemblyben szintén hozzá mondjuk alkalmazói programokat. Amint elkészült a rof szövegszedő program, az egy Assemblyben készült program volt, de maga mondjuk a fájlrendszer, amit használt a gép, viszont már nagyon is olyan volt, mint a mai. Tehát a mai UNIX-oknak a fájlrendszerére emlékeztetett. Ahol egyes fájlok inode számmal, vagy itt a UNIX operációs rendszerkönyvben i-csomó-nak magyarították, nem tudom, úgy gondolták, hogy ez megér egy külön fogalmat és így hivatkoznak erre a számmra. Ezek már megfigyelhetőek voltak ezekben a Thompson-féle fájlrendszerekben.

**Kevesen gondolnák, hogy a rof szövegszedő program, ez később a troff nevet vette fel, amikor már átírták C programozási nyelvre.** Tulajdonképpen egy jelölő nyelvre hasonlított, a parancsokat, ezt tulajdonképpen kétbetűs parancsai voltak, nagybetűsek, a sor elején ponttal kezdődött a parancs, akkor tudta a feldolgozó program, hogy most egy parancs következik és annak megfelelően mondjuk egy felsorolást csinált, új bekezdést kezdett és így tovább. Tehát ezt lehetett egy szövegszerkesztővel leírni. Mi köze ennek a mához? Ez a rendszer szolgált arra, hogy megjelenítse a rendszernek a dokumentációját, a sűgőjét, amit kézikönyvnek hívnak a UNIX-ban, és az egyes részeket, amik egy-egy parancsokhoz kapcsolódtak illetve rendszerhíváshoz vagy egyéb szabványleírásokhoz, a kézikönyvben oldalként hivatkozunk rá. Manuál a neve és ez a manuális, ilyen troff jelölőnyelvvvel volt leírva és amikor az ember lekérte, akkor elindította a rendszer a szövegszedő programot, formázta, nem egy nyomtatóra formázta, hanem mondjuk a Teletype-ra vagy a képernyőre azt látta az ember, azt tudta elolvasni kényelmesebben, mintha magát a jelölőnyelvvvel leírt forrásszöveget nézné. Amikor a Linux kialakult, akkor egyre több ilyen eredeti UNIX alkalmazást vettek át, de valahogy senkinek nem volt kedve troff-ot átírni szabad változatra. A Linux az egy ilyen szabad rendszer és ilyen alkalmazásokat tettek rá vagy próbáltak írni. Egy ember volt talpon a vidéken, James Clarknak hívják, egy oxfordi egyetemista, nem tudom, hogy milyen életkorú volt, aki vette a fáradságot, úgy gondolta, hogy egy jó kihívás, és megcsinálta az eredeti még UNIX dokumentáció alapján valamennyire szabványosították ezt a troff nyelvet, és az alapján elkészítette a szabad változatát a gnutroff vagy röviden gtroff szedőprogram. Ez a James Clark nagyon megszerette ezt a szövegszedést illetve a jelölő nyelveket. Aztán ránézett egy ilyen DSSSL szabványra, „déeszeszeszel”-nek ejtik, ami már ISO szabvány volt és gyakorlatilag nem más, mint a HTML-nek az elődje. Ez a James Clark, mikor látták, hogy ez a DSSSL,

ez a jelölő nyelv igazából gépi feldolgozásra nem olyan kényelmes, részt vett egy szabványosító bizottságban, sőt ő volt a technikai vezetője, ami létrehozott egy új jelölő szabványt, ez volt az XML. Annyit kell róla tudni, hogy a legújabb irodai csomagok is, azok átállnak már az XML használatára. Tehát egy adattárolási formátum, hogy hogyan definiáljunk saját magunknak illetve saját rendszereink részére adatformátumot. Amellett ez a James Clark nem csak ezt csinálta. Thaiföldön él most, de ettől függetlenül az összes olyan alapalkalmazás, ami az XML feldolgozásához készült, mondjuk az XPath elemző, XML elemző, de maga a stílusnyelv feldolgozó is, nagyon sok minden a nevéhez kötődik. Nagyon sok mindent megvalósított, nem csak a szabványt készítette el, hanem meg tudta csinálni magát a programokat, a rendszereket, ami azért nem utolsó dolog. Az egy dolog, hogy a szabvány, mint a Multics-nál, hogy nekiálltak a szabványnak, de a megvalósításba beletört az ember bicskája.

Egy másik szintén elég komoly fegyvertény, ez pedig a Unicode kódolás, ennek is elég sok köze van a UNIX-hoz, mégpedig a következő. Amikor kiderült, hogy nem csak az Egyesült Államokban vagy Nyugat-Európában szeretnék használni a rendszert, a számítógépeket, hanem más helyen is, és esetleg használnák úgy is, hogy egyszerre több nyelv jelkészletét is egy-egy adatállományban, fájlban vagy egy adatfeldolgozó rendszerben, szövegszerkesztőben. Akkor jó volna egy új szabvány, ami mondjuk nem az eredeti ASCII kód, ahol 7 biten tárolták 128 karakteren, a vezérlő karakterekből le volt foglalva jó pár, tehát limitált volt a mennyiség. A kibővített ASCII, gondolunk itt a latin 1,2,3 az ISO 8859-es szabványra. Ezek se voltak elégségesek egy komolyabb leírásra, ilyen több nyelvű szövegeknek a tárolására és maga a Unicode szabvány, az ISO 10642-es szabvány ezek külön léteztek. Azt tűzték ki, hogy létrehoznak valami újat, a legtriviálisabb az az, hogy kibővítik a kódolást, hogy egy karakter, azt milyen byte hosszan, milyen szó hosszan tároljanak. Rögtön azt mondták, hogy 8 bit, az nem elég, 16, ne szűkösködjünk, legyen 32 bites, ez volt az eredeti ISO szabványnál a megközelítés.

A probléma csak az volt, hogy elég nagy áttérést kívánt volna, eléggé erőforrás igényes volt, hogy négyszer akkora lett a feldolgozandó adat mennyisége azzal, hogy 4 byte-on tárolták. Próbáltak köztes megoldást keresni, ami mondjuk kevesebb byte-on tárolja a dolgokat, és végül is ez volt az egyik legfontosabb fegyverténye majd. Itt meglátjuk, hogy kinek és mikor. 1992-ben, ekkor volt ez a szabványosítási kísérlet, létrejött egy olyan Unicode kódolás, ami kellően hatékony volt és sok jó tulajdonsággal rendelkezett. Sejtették, hogy úgy volna érdemes, hogy megőrizzék az eredeti ASCII kódokat, tehát valamennyire olvasható legyen a szöveg és mondjuk azt tárolják 1 byte-on és jó volna, ha esetleg speciális, ékezetes betű van, akkor hozzáadunk még egy-két byte-ot. Voltak ötletek, hogy hogyan lehetne ezt tárolni. Kialakult egy munkaterv, a munkatervben az szerepelt, hogy itt tárolják az adatokat, a munkacsoport vezetője, az ISO szabványosításért felelős csoport vezetője úgy gondolta, hogy egy hiba van vele: ha megszakad az eleje vagy a közepébe nézünk, akkor nem lehet elolvasni az adatfolyamot, tehát magyarul szólva érzékeny a kezdőpontra a kódolás, a kezdeti kódolás. Ezen ő sem tudott segíteni. Az nem tetszett neki, hogy máshol is előfordulhatnak ASCII karakterek a kódban, csak azok éppen nem ASCII betűk és zavaró, hogy nem tudja elolvasni ránézésre az ember ezt a kódot. Annyiban módosított a szabványon, hogy ne szerepeljenek az a,b,c,d, tehát a latin betűk, az ékezet nélküli betűk ne szerepeljenek az ékezetes betűk kódolásában, és utána írt mindenkinek egy körlevelet, aki rajta volt egy listán, hogy valaki csináljon már ebből egy szabványt, mert megakad a szabványosítási folyamat. Amikor olvasta a levelet Ken Thompson, ez 1992-93-ban lehetett, gondolkodott, hogy az sem jó, ha a program érzékeny a kezdőpontra, hagyjuk ki belőle ezt. Ne legyen a kezdőpontra érzékeny a rendszer, a többit azt hagyjuk meg, próbáljuk lekódolni úgy, hogy megosztjuk valahogyan vagy leírjuk a megosztást, hogyan tároljuk több karakteren az ékezetes betűket. Legyen az, hogy van olyan tartomány, ahol két karakterrel fogjuk rögzíteni a karaktereket van ahol 3-mal, 4-gyel, 5-tel vagy 6-tal, attól függően, hogy mondjuk az alapkiosztásban, karaktersíkon hol helyezkedik el az adott karakter. Ezt kollégájával Rob Pike-al (akkor már nem a UNIX-on dolgoztak, hanem a Bell laboratóriumban, a Plan 9 nevű, „kilences terv” nevű operációs rendszeren, ami kapcsolódik a mai operációs rendszerekhez, a Vista-hoz meg az újabb Windowsokhoz) összeültek és megcsinálták a

szabványt, ez a szabvány lett az UTF8. Gyakorlatilag ez az, ami sikeressé tette az Unicode kódolást és ha megvolt a szabvány, akkor azt be is küldték, utána szabványosították is egy részét. Maga az ISO 10642-es szabvány csak ezt írja le, hogy van több UTF kódolás, a legfontosabb az UTF8, és van még az Unicode szabvány, ami kicsit többet ír le, a feldolgozással meg egyébbel kapcsolatban is sok mindent leír.

Mi köze van a UNIX-hoz az Unicode kódolásnak, illetve az XML-nek? Gyakorlatilag az, hogy nem jött volna létre a UNIX fejlesztők nélkül egyik sem, illetve azt láttuk, hogy Ken Thompson maga csinálta az UTF8 karakterkódolást. 23 évvel a UNIX után, az XML-t pedig James Clark szabványosította, aki megcsinálta a szabad troff szövegszerkesztőt a gtroff-ot, azt a szövegszerkesztőt, ami nélkül nem jött volna létre a UNIX, hiszen csak annak fejében kaptak megfelelő gépeket a szerkesztők. Valahogy mindennek a UNIX a gyökere, minden a UNIX-hoz vezet.

**1971: megjelent a UNIX első kiadása, így is hivatkoznak rá.** Az első kiadású UNIX és érdekes, hogy a kézikönyvben is az első kiadású, nem véletlen, hogy a kézikönyvről nevezték el a kiadásokat. A kézikönyv gyakorlatilag a rendszer dokumentációja és annyiban érdekes, hogy mindenki, aki komolyabban benne volt, kihangsúlyozza, hogy a kézikönyv írás az egy nagyon furcsa dolog volt, mert akkor tisztázta le az ember a rendszert. Ahogy nekiállt írni az új parancsoknak a dokumentációját, akkor rájött, hogy ezt nem lenne jó beleírni, mert elég égő dolog, hogy ilyen kapcsolókat, meg ilyeneket beírunk, ezt inkább vegyük ki vagy módosítsuk. Gyakorlatilag nem csak a kézikönyvet írták meg, hanem átírták a programokat hozzá, hogy az valami kerek egészet alkosson és átláthatóbb legyen.

Maga a kézikönyv oldalakat a man paranccsal lehet előhívni, látható, hogy a man man parancs dokumentálja saját magát. És akkor leírja mondjuk a rendszernek a nevét, hogy „man”, egy rövid összefoglalás itt az első sorban, tehát, hogy egy interface az online kézikönyvhöz, tehát a referenciához. Itt látható egy összefoglalás, hogy mennyi kapcsolóval rendelkezik. Ha megnézünk egy régebbi UNIX-ot, ott kevesebb a kapcsolók száma, az elburjánzás az megfigyelhető ezekben a rendszerekben. De nem kell kétségbe esni, egy mai beágyazott UNIX-hoz éppen olyan parancsok szükségesek, amik kicsik, tehát van ilyen jellegű fejlesztés is, hogy ezt levigye minél kisebbre. 60 parancsot találunk ezen a rendszeren, ugye még a '71 az azt jelenti, hogy Assemblyben íródott a rendszer a segédprogramok legtöbbször is, bár ekkor már elkészült a héj, elkészült a legelső változat, ezt Thompson héjnak hívják, ami feldolgozta a parancsokat. Ez képes volt, hogy elindítsa a programokat, átirányítsa a kimenetét fájlba vagy onnan olvassanak a programok. Illetve maga a héj, az egy programozási nyelv is egyben, tehát lehetett írni, úgymond kötegelni lehetett a parancsokat és feldolgozni akár egy háttérprogramként.

**1973:** sok olyan dolog történt, ami elválaszthatatlan a mai Linuxoktól. Megjelentek a csövek (pipe). Például a hang adatokat fel lehet dolgozni csőhálózat segítségével és menet közben képes arra, hogy módosítsa vagy átalakítsa a beszédhangnak a frekvenciáját egy más tartományba, egy csőhálózat segítségével a felvételt és a lejátszott programot összekötve egy ilyen csővel. Abszolút elméleti megfontolásokból nőtt ki. Douglas McIlroy nevéhez kötődik a csövek kitalálása és papíron a megvalósítása is, aki egy matematikus, aki éppen szeretett programozni, és gyakorlatilag a Bell laboratóriumban a fejlesztési munkáknak a vezetője volt. Már a '60-as évek végétől ezen törte a fejét, hogy hogy lehetne megoldani azt, hogy a szoftverkrízisnek az egyik ilyen fő problémáját, hogy nincsenek újrahasznosított kódok. Újra és újra az ember megcsinálja ugyanazt, ha éppen van valami, amit fel tud használni, de kicsi az esély rá, és már a '70-es években ő azon gondolkodott konkrétan, hogy hogyan lehetne szimbolizálni, leírni magát a csővezetéknek. Amikor kezdett kristályosodni a kép, hogy már lehetne a UNIX-nak a standard be/kimenetelével valamit csinálni, ha felírta a beleírási jelet, azt a rókaszájba kacsacsőr jelet, egy pontot, meg még egy beleindítási jelet. Ez pont '73-ban történt, egy isteni szikra, hogy rájött, hogy van még egy karakter a billentyűzeten, egy csőjel, ami igazából nincs kihasználva és ezt megmutatta Ken Thompsonnak, aki még aznap este nekiállt és átírta a rendszert, átírta a segédprogramokat is, hogy képesek legyenek ezt használni, a legalapvetőbb segédprogramoknál. Gyakorlatilag ez azóta jelképe a

UNIX-nak, más kérdés, hogy utána az MS-DOS-ban is találkozott vele az ember, az olyan, de nem (az ideiglenes állományokat ment el a háttérben, nem egy többfeladatos operációs rendszer volt az MS-DOS, ezt nyilván így kellett elvégeznie).

**C nyelv. Ennek is megvan a története.** Tehát egy magas szintű nyelvből, még a BCPL-ből, az alapján és még több más ihletésből Ken Thompson csinált egy olyan nyelvet, a B nyelvet, ami értelmező volt az eredeti UNIX-on, gyakorlatilag arra szolgált, hogy mondjuk kevésbé erőforrás igényes dolgokat meg tudott csinálni a B nyelven. Bár eredeti Multics terv volt az, hogy magas szintű nyelven valósítsák meg az operációs rendszert, hogy a PL/I-et választották. Meg van az előnye, hogy egy rendszer magas szintű. A legnyilvánvalóbb előnye, az nem is az, hogy átláthatóbb a kód, mert ők átlátták az Assemblyt is, a gépi kódot is átlátták, hanem a hordozhatóság, hogy át lehetett tenni egy másik gépre. Mert a gépek azok újabbak, jobbak születtek és mindig kicsit variálódtak.

A C nyelvre jellemző, hogy már '72-ben elkezdte átírni. Akkor már a C nyelvnek megvolt az elődje, megvolt a C korábban. A '73-ban kijött UNIX változatban volt benne a C nyelv, tehát hivatalosan a kézikönyv oldalakra bekerült, tehát így kötődik '73-hoz. Illetve 1973-ban jött létre egy új nyelvi szerkezet, ami nélkül '72 nyarán nem tudta, feladta Ken Thompson, hogy átírja a UNIX-ot magas szintű nyelven, és ami hiányzott belőle, az a struktúrák volt, azt tette még be Dennis Ritchie, és azzal már meg lehetett azt csinálni. '73 februárjában már kijött az a változata a UNIX-nak, a v6 talán vagy v4-es változat, és valamikor november táján már átírták C nyelvre magát a UNIX-ot. Ez a háttérkép akkortájt készült.

A kódméret 10%-kal több, a futási idő sem mindig több, átlagolva mindenféle alkalmazásra, nyilván ez függ attól, hogyan kódol le az ember, milyen algoritmussal valamit. A lényeg az, hogy nem vesztettek sok mindent vele, viszont nagyon sok mindent nyertek. Az egyik ilyen az, hogy át lehetett tenni a rendszert más gépekre.

A C nyelvnek is szép utóélete van. A UNIX-nak ott a helye az iskolai oktatásban, ugyanúgy a középiskolai oktatásban, az informatika órákon, ha más nem bemutatni egy-két ilyen látványos programmal. Onnantól kezdve, akit megragad ez, az tudja, hogy van ilyen, aminek utána lehet nézni. Az oktatásnak is ez a lényege, hogy megmutasson sok dolgot, elindítsa az embereket valamilyen irányba. Nyilván nem lesz mindenki UNIX programozó, de ha mondjuk már évente lesz egy-kettő, aki különben nem lett volna, akkor már érdemes erről beszélni. Nagyon sok hatása van és nagyon sok dolgot tanít meg, amit az ember más programozási nyelv, más operációs rendszer alatt nem lát. Mivel nyitottak a mai UNIX-szerű rendszerek, a UNIX forráskódja is nyitott, az eredeti UNIX-é valamennyire eredetié. Gyakorlatilag ott áll az ember előtt, jó rettenetes 40 év tudása, tapasztalata, csak éppen hozzá kell fordulni, tudnia kell az embernek és lehet vele élni.

**1975:** ez már a hordozhatóságnak az egyik szimbolikus éve. A 6-os változata jelent meg a kézikönyvnek, meg magának a UNIX-nak is, és Ken Thompson kivette egy éves szabadságot, elment a Berkeley egyetemre, hogy ott vendégtanárként egy éven keresztül okítsa a hallgatókat. Itt tette át a rendszert egy 32 bites gépre a UNIX-ot, nem korábban, '71-ben. Itt már sok egyetem meg is kapta magát a rendszert, sőt korábban megkapta. '74-ben már olyan kapcsolata volt a Berkeley Egyetemmel Ken Thompsonnak, hogy amikor felhívták, hogy valami nem megy, vagy az operációs rendszer nem rendesen viselkedik, akkor kérte, hogy a telefonon keresztül férjen hozzá a kódhoz és gyakorlatilag onnan tudta megkeresni a hibát az ottani fejlesztőket segítve ezzel.

Említettem, hogy milyen történeti szálak kapcsolódnak itt a UNIX-hoz, például '74-ben a Berkeley Egyetemen az Ingres adatbázis-kezelőt fejlesztették szintén még igen elméleti megközelítésből, gyakorlatilag ott az SQL-nek az alapjai, az alapvető adatbázis kezelőknek az alapvető működési dolgai a UNIX-on alakultak ki. Használtak gépeket, amik VAX gépek, amik VMS operációs rendszert futtatták arra tették át a UNIX-ot. Viszont, akik már megszokták a másikat nem akartak egy új alapértelmezett operációs rendszert használni a gépen, hanem azt sikerült elérni, hogy megosztva, egy „vetésforgóban” felosztották a gépeket a UNIX és a VMS vagy annak az elődje operációs rendszer között. Először a UNIX kapott 8 órát és a hátralévő időt, a 16 órát megkapta a

másik rendszer. Aztán szóltak a fejlesztők, hogy ne este meg hajnalban kapják meg ezt a 8 órát, hanem más időben, aztán a végén kiderült, hogy a UNIX jobban működik, hatékonyabb, termelékenyebb a rendszer, úgyhogy neki legalább ugyanannyi idő jutott itt.

**1975:** Stephen C. Johnson Bell fejlesztő módosította a C fordítót illetve elkészítette egy új változatát. Annyit csinált, hogy legyen a C fordító is hordozható, teljesen automatikus programíró rendszereket fejlesztett, ebből a YACC a legismertebb, ami nem jelentett más, mint, Yet Another Compiler Compiler (Egy Másik Fordító Fordító program), ami nem csinált mást, mint hogy az ember leírja a fordítónak a nyelvtanát, amit megért, amit fel fog dolgozni, és ez lefordította C forráskódra ez a rendszer és aztán csak be kellett illeszteni, hogy mi történjen akkor, ha olyan utasításra talál C fordító. Csinált egy lexikai elemzőt is, ami felismeri ezeket az egységeket. A szintaktikai elemző generátor volt a YACC, a másik program pedig a LEX, lexikai elemzőt állított elő, ami felismerte, hogy a karaktersorozat az egy szám és ez a számegységet át tudta adni a szintaktikai elemzőnek, hogy az már annak megfelelően kezelje, hogy mit is akart ez a program jelenteni.

A Johnson C nem volt más, mint egy hordozható C program. Ez a LEX meg a YACC, még mindig tanítják, használják újabb és jobb eszközök, de még ha az ember kényelmesen akar egy Linux alatt egy fordítót fejleszteni, vagy éppen az OpenOffice-ban is használják ezeket a részeket, akkor a LEX-et tölti le vagy telepíti a gép, a YACC-nak ez a Bison vagy bölény, GNU fejlesztés azt fogja használni, és ezt használják is a különféle fejlesztők rendszeresen.

BSD UNIX, ez a Berkeley Egyetemről kikerült fejlesztés. Az az érdekessége, hogy ez már kereskedelmi fejlesztés lett, és elindult egy versengés az eredeti Bell fejlesztés és e között. Sok olyan dolog például, mint a terminálok kezelése, az egyetem rendelkezett már különféle terminálokkal, lehetett teljes képernyős szövegszerkesztőt csinálni, mint például a vi (ejtsd: viáj), ami már ki tudta használni a különböző dolgokat. Itt van a C programozási nyelvről '78-ban íródott könyv. Erről azt kell tudni, hogy gyakorlatilag ma is egy ilyen Bibliaként lehet forgatni. Ez már egy második kiadás, beleírták már a szabványosítás után, különféle C szabványok voltak '89-ben, meg egyéb szabványok.

**1979-ben megjelent a 7-es változat UNIX, amire azt mondtam, hogy jobb volt, mint az összes korábbi előző, meg mint, ami később született UNIX változat.** Ebben jelent meg a Bourne héj, amit ma is Linuxokon, mondjuk éppen vagy a Bash változatban, vagy van egy Dash változat, amit Ubutuban talán arra cserélték le. Ez a Bourne héjnak az utánérzése. Az awk programozási nyelv is itt jelent meg. Melyik a két alapértelmezett unixos nagyon magas szintű nyelv, szkript nyelv? Az egyik maga a héj, a másik az awk program. Ez egy kicsit C nyelvű szintaxissal rendelkezik, de annál többel, reguláris kifejezéseket is lehet benne használni. Ebből nőtt ki valamennyire a Perl, a Python meg a Ruby nyelvek is. Az awk szövegszerkesztőben is voltak olyan megfogalmazási beépített lehetőségek a reguláris kifejezések felhasználására.

**Elindítottam ezt a 7-es változatú UNIX-ot, megnézzük, hogy mi az ami benne található.** Ez a PDP-11/SIMH emulátor, ami különféle korai '70-es évekbeli, de megjelentek a '60-as években is megjelentek már azok a PDP-k, azoknak a képeknek az emulátora. Ha ezt az ember egy kicsit beizzítja, hogy 18 bites legyen a rendszer, vagy mit csináljon. Ezzel el lehet indítani egy lemezképet, amit szalagról mentettek fel és olyan felhasználói voltak, mint Dennis Ritchie.

Ami most megjelent az már maga a UNIX-nak a promptja tehát már fut a UNIX rendszer, ami jó hír nekünk, mert talán azt jelenti, hogy el is tudjuk indítani. Boot paranccsal indul, vannak különféle eszközök, megírjuk, hogy milyen eszközt, al-eszközt indítson el és milyen fájlrendszer van ott, unixos fájlrendszer. Most újra be fogom írni, mert a visszalépést azt nem támogatja. Elindult a 7-es változatú UNIX. Ezt a SIMH projektnél lehet letölteni ezeket a régi lemezképeket, meg 5-ös változatú UNIX is elérhető. Ezt valamennyire szabaddá tette a Caldera legutóbbi birtokosa, a Caldera UNIX, ami SCO volt korábban majd egyesült a Caldera Linuxszal. Talán valahol Utah

államban fejlesztették a mormonok a Caldera Linuxot és nagyon jól prosperált. Utána meg tudták venni az SCO vállalatot.

Amikor hiba van, akkor bőbeszédű a rendszer. Ha az „ls” parancsot kiadom, akkor végigfut a fájlrendszer a gyökérben, látható, hogy van egy /bin könyvtár, ami a bináris futtatható elemeket tartalmazza, a /boot magát az operációs rendszert. Ahhoz, hogy tudjak törlést használni stty paranccsal beállíthatom a terminált és akkor mondhatom, azt, hogy legyen a ctrl+h a törlés. Beállítottam a terminált, hogy minden lapdobás után hagyjon egy kis szünet ezt a cr3 paranccsal lehet elérni, stty cr3, ezt elfogadja, most visszaállítom cr1-re, elvileg most nem fog várni. Annyit kell tudni, hogy a cr3 parancsnak van egy sokkal újabb változata, ami ezt csinálja, tehát kicsit megáll minden egyes oldal után, a nyomtatást befolyásolja ez és mivel ez többet tudott a BSD-s fejlesztő elnevezte „more”-nak. Ezt később a UNIX gyűlölközők kézikönyve úgy idézte, mint a legrosszabb példa arra, hogy milyen titkos, érthetetlen parancsai vannak a UNIX-nak. Mi a fenének írják meg azt a programot, amibe, ha beleírnyítunk megáll egy oldal kinyomtatása után és egy billentyű lenyomására vár? Ebben a rendszerben, a 7-esben ilyen nem volt, tehát oldják meg a fejlesztők maguk, ha ilyet akarnak. Ha ctrl+s-t nyomunk, akkor megállítja a terminál kimenetet és el tudjuk olvasni a manuált, utána ctrl+u-val meg továbbindítom. Tehát ezek alapból benne vannak ebben a könyvben, ezért mindenkinek tudom ezt javasolni és az órán is ezt kell mutogatni a gyerekeknek, hogy ezzel kezdjék az ismerkedést.

Elindítok egy sakkjátékot, egy üres sorra kiírja magát a rendszert, mármint a sakktáblát. Azt mondom, hogy B2B4, akkor elindul a játék. Megoldották, hogy a bábuk felismerhetők, megkülönböztethetők, a kisbetű, nagybetű különbözteti meg, és aki megismerkedik a UNIX-szal mindenhol ott van, minden szinten a zsenialitás szikrája.

Forgattam Tannenbaum „Az operációs rendszerek” című könyvét, de annyira nem ástam bele magam, de gyakorlatilag az eredeti UNIX-nak az Assembly forráskódja, vagy ahogy megoldja a rendszerhívásokat, miket biztosít a futó programok számára, hogyan kezeli az eszközöket, az mind zseniális. Távol áll attól az elméleti megközelítéstől, hogy tökéletes legyen, az a fontos, hogy legjobban használható legyen. A praktikus megoldás meg a tömörség az a rendszernek a sajátja. Ctrl+d-vel lezárom ezt a sakkjátszmát.

### **Hallgató: Naptár volt benne?**

**Előadó:** Igen volt. Azért nem mindent kezel ez a naptár része. Szerettem volna megmutatni mennyire továbbfejlesztették a „cal” programot, meg tudja azt csinálni, hogy a naptár el is férjen ezen a képernyőn. Az máshogyan sorrendezi és pontosan elfér egy 80x25-ös képernyőn, optimalizálják a régi linuxos programokat. Elvileg nem kéne már formázni, az lpr elvileg egy fejléccet rádob, én közvetlenül szoktam mátrix nyomtatóra „átutalni”, amit manapság is szívesen használ az ember, főleg, ha közvetlenül van csatlakoztatva egy mátrix nyomtató.

Elvileg rajta van a C fordító és elvileg forráskódot is találunk. Mutatok még egy játékot, az akasztófa játék, annak a forráskódját meg lehet nézni.

Csodálatos maga a rendszer. Ha valaki gyerekeknek megmutatja, fakultáción remek. Megmutatni nekik a password állományt, nézzük meg az /etc/passwd-t, ami a rendszerben a felhasználókat tartalmazza, ott lesz Dennis Ritchie a C nyelv kidolgozója. Ez úgy lett lementve, hogy Dennis Ritchie-nek volt egy azonosítója ezen a rendszeren és ez úgy lett szalagra elmentve ez a rendszer, ez a kép.

## Németh László: A legcsodálatosabb oktatóprogram: a 40 éves Unix programozási környezet II

Úgy dokumentálták a rendszert, hogy először bootolja az ember a UNIX-ot, akkor egy egy felhasználás üzemmódba kerül rendszergazdaként és utána, amikor kilép, többválasztóként lehet bejelentkezni, akkor ad logint legelőször. Itt most én tudtam a root logint és beléptem rendszergazdaként, tehát ezt valahogy ki is írja. Sok minden dolog nem működik, ami meglepő, ezt a „more” parancsot eleve nem találja, ez már a BSD UNIX-hoz köthető. Itt hagyom ezt a háttérben, hátha még szükség lesz rá.

**Xenix:** még a Microsoftnak is volt UNIX rendszere, gondolja az ember. Licencelték ezt a v7-est, amit az előbb láttunk és akkor megkértek egy céget ez az SCO (Santa Cruz Operation) és az megcsinálta nekik 16 bites mikroszámítógépre vagy a nem túl gyakori az IBM PC, ami megjelent 79-80-ban. Átírták neki és ő licencelte különböző cégeknek. Azt mondták, mivel a mikroszámítógépek tényleg sikeresek lettek, sokkal több helyen, több számban fordultak elő, ez volt az a UNIX, az a Xenix, igazából a v7-es átdolgozva mikroszámítógépre, ami a legtöbb gépen futott ebben az időben. A Xenixet a végén megvette maga az átíró cég, az SCO, ez az SCO UNIX, nagyon jó nevű UNIX lett, aztán utána a Caldera Linux vette meg, utána rossz fény vetült rá, amikor Linuxszal perlekedett.

**A SUN Microsystems,** ami 82-ben jött létre, a Berkeley egyetem hallgatói hozták létre. A fejlesztők berkeiben a legismertebb az Bill Joy, aki nem csak a „vi” szövegszerkesztőt készítette, hanem sok olyan dolgot, ami az első igazán hatékony TCP/IP kezelő kernel rész megvalósítás, ami gyakorlatilag sikeressé tette a hálózatot, magát a hálózat koncessziót az akkori egyetemek között, ez korábban született valamivel. 1975-ben tanított Ken Thompson az egyetemen, Bill Joy is talán hallgatója volt és Ken Thompson egy ráérős idejében a gépteremben írt egy Pascal fordítót, ami oktatási célra nagyon jó és gyakorlatilag a BSD UNIX így jött létre, hogy ezt a Pascalt karbantartották. Küldték az egyetemeknek szalagon és Bill Joy volt a felelős, aki másolgatta. Egy hónapban 25 kazettát elküldött egyetemekre, aztán kiderült, hogy javítani kell a Pascalon, tehát átvette a Pascal karbantartást és utána beleásta magát a programozásba és a végén termcap fejlesztés, a különböző terminálokat be lehessen állítani a UNIX rendszerhez és azt használja. Aztán a vi szövegszerkesztő, a C-shell, a C-héj a BSD Linuxon szintén alapértelmezett volt, ezt Bill Joy készítette. Valaki nyilatkozta róla, aki ismerte, hogy ő is tudna úgy programozni, ahogy a Billi Joy, de a Bill Joy tízszer annyit csinál ugyanannyi idő alatt. Valaki ezt vetette a szemére, hogy éppen ez a baj, csak írja, írja és nem olyan tisztázott a kód.

Mindenesetre itt ez a 1983-as, mégiscsak akkor jelent meg ez a TCP/IP megvalósítás, ez is érdekes, mert a szabványosító szervezet gyakorlatilag rákényszerítették, hogy UNIX-ot használjanak az eredetileg az VMS helyett, ugyanis amikor elküldték, hogy hogyan képzelik a szabványt, elküldték Berkeley egyetemnek és Bill Joy jobban megcsinálta, mint ahogy azt meg tudták volna csinálni eredetileg a tervezők. Úgyhogy aztán a UNIX lett az alaprendszere az internetnek.

Aztán a UNIX System V-ös, a kereskedelmi korlátozások megszűntek, ők is árulhattak UNIX-ot. Nem lett annyira ismert, ez annyiban lett ismertebb, hogy ez képezné majd az alapját a Solaris operációs rendszernek. A GNU projekt még ebben az évben jött létre, amikor ezeket a szabad szoftvereket így külön hangsúlyozták. Itt van ez a könyv, amit többször meg fogok mutatni, melyik a legalapvetőbb műve a UNIX operációs rendszernek? A UNIX operációs rendszer.

**Aztán Tannenbaum „Operációs rendszerek tervezése és implementáció”** című kötete jelent meg társszerzővel és mellékletként csinált egy operációs rendszert, egy UNIX típusút, a MINIX nevű operációs rendszert. Ez gyakorlatilag teljesen átvette az eredeti 7-es kiadású UNIX-nak a 40 rendszerhívását és maga a kód is 40 kilobyte-os volt, tehát elég tömör volt a rendszermag, ezt átvette és írt hozzá egy mikrokernels, mikro rendszermagos rendszert. Ami a MINIX-ből még

érdekes, hogy ezen lelkesült fel Linus Thorvalds néhány évvel később, hogy nem akarta a MINIX-et továbbfejleszteni Tannenbaum. Ez egy oktatórendszer volt és az volt a lényege, hogy kicsi legyen és be lehessen mutatni, hogy hogyan kell egy operációs rendszert írni.

**1988-ban a POSIX szabvány jött létre.** A névadója a szabványnak az Richard Stallman. A neve a UNIX alapján lett, mivel azt szabványosította ez, ha volt szabvány is, leírás, meg tankönyv is, akkor a Linux-ot meg lehetett csinálni egy 386-os AT számítógépre, ami egy preventív ütemezést már támogatta, meg tudta szakítani a rendszer mag a futó programot.

**Solaris:** volt egy saját UNIX-uk, a BSD UNIX a fejlesztőknek, a SUN-nak, de igazából megvették, egyesítették, de legfőképpen a System V-re alapították az ő operációs rendszerüket. Ezt is, a Solarist, talán 100 dollárért megkaphatták az egyetemek, tehát meg lehetett ismerni a forráskódját ennek is. A MAC OS X, amiről annyit kell tudni, hogy a match mikrokernel, amit kutatott a Darwin operációs rendszer, az ugyan egy korábbi más jellegű fejlesztésből született, mint a UNIX, de hogy igazán sikeres legyen, az volt, hogy átvette a UNIX-ból a rendszerhívásokat és beillesztették egy UNIX rendszernek a helyébe, kicserélték a monolitikus kernelt. Erre épül a Darwin operációs rendszer, ami nem más mint egy teljesen UNIX programozási környezet. És ez fut például az iPhone, iPod, iTouch-on.

**2003: „A UNIX programozás művészete” Eric S. Raymond tollából.** Alapmű ez is valamennyire, ha valaki meg akar ismerkedni a programozással.

**2005: OpenSolaris, megnyitották a System V-ös forráskódot,** ez az egyetlen létező szabad operációs rendszer, ami az eredeti UNIX forráskódra épül, a System V-ös, a Bell laboratóriumi UNIXra. Ideális rendszer oktatási célokra, legalábbis megmutatni, hogyan programoztak a nagyok.

**PowerShell:** gyakorlatilag ez a Plan 9-ben megjelent, nem szöveg koncesszióban, hanem gyakorlatilag egy magasabb szinten objektumokat ad át csővezetéken egymásnak a rendszer. Ezt csinálták meg valahogy a Microsoftnál, rendszer programozási nyelvnek. Lehet vele adminisztrálni, illetve programozni ezzel a PowerShell-lel, úgyhogy elértek oda, mint a Plan 9-cel mondjuk a 70-80-as években.

**2009:** fejlesztik folyamatosan, itt a segédprogramokat azt még mindig fejlesztik. Most március végén jött ki a GNU core utilities-nek, ami a Linuxnak, meg a többi rendszernek a legújabb segédprogram változata. A busybox, ami beágyazott rendszerekhez készült, az február 26-án volt az utolsó változat. Nagyon is fejlődik a rendszer.

**2038 probléma:** a 31 biten tárolt idő az már lefut, tehát ha én azt mondom neki, hogy „date %s200904032, bocsánat a 7-es változatú UNIX alatt voltam és az kicsit más. Még nem ismeri az ISO8660-as szabványt, ezt a fajta dátumjelölést, amit most beírok, ez kicsit később született. Ez kiírja az eltelt másodperceket. Fel lehet figyelni, hogy „123” ez nem messze van az „1234567890”-tól, ez most volt valamikor februárban, amikor átütötte ezt a szép dátumot a UNIX. Ezt számon tartják és ünnepelnek az emberek, akik azt érzik, hogy van valami köztük a UNIX-hoz. Nézzük a 2038-at, január 19, itt már nagyon ott van a tartományánál, ez a két milliárdos, nézzük a másnapi napot. Ajaj, érvénytelen a dátum, ez rosszabbnak tűnik, kritikusabb rendszerek futnak UNIX-on, a 2000 év problémájánál sokkal súlyosabb lesz.

Ha ezt az bitet felhasználjuk, akkor 2106-ra fellélegezhetünk picit. Más kérdés, hogy már most, pár éve jelentkezett egy 2038 év probléma, az AOL szervernek volt egy ilyen hibája, hogy nem tudom pontosan, de egymilliárd évre előre dátumozott, ezt a UNIX 31 biten csinálta és ezért lefagyott a szerver. Egyre több ilyen probléma lesz. 64 biten tárolva az időt elvileg már nem lesz probléma. Linuxnál már talán így van, a 64 bites operációs UNIX-okon már így van szerintem, tehát ezek a jövőbe néznek. A Windowsnál, nem tudom a Vistanál is így van-e, de ott azt hiszem 64000 évet lehet még előre lépni.

**40 évnyi tudás:** a filozófiáról, amiről már a csővezeték kapcsán már beszéltem, reguláris kifejező eszközök, kifejezések, rendszermag, héj, egyéb segédprogramok. A Linux filozófiáról röviden. McIlroy-ról már meséltem, matematikus volt. Már 1967-ben tartottak itt valahol Európában egy konferenciát, ahol először a szoftver krízis elhangzott, talán Distra holland informatikusnak a szájából. Már akkor is ezt érezték az emberek, hogy egyre gyorsabbak a gépek és meg kell osztani a rendszert, több felhasználósnak kell lennie, hogy ki tudja használni a gépidőt, de úgysem tudják kihasználni, mert nem tudtak olyan gyorsan programozni az akkori eszközökkel, hogy ki tudja használni a gépek nyújtotta teljesítményt. És ezt úgy élték meg, mint egy ilyen frusztrációt és gyakorlatilag utána évekig ez a McIlroy azon törte a fejét, hogy hogyan lehetne ezt betenni egy UNIX-ba. A '73-as év volt, amikor megtörtént a nagy áttörés. A „tee”, ez gyakorlatilag t-csövet jelent az angolban, annak a rövidítése, így hívják a vízszelők az elágazást és a „tee” parancs, az azt csinálja, hogy az ember beilleszti valahova vagy megad paraméterként egy fájlnevet, a csővezetékéből, amin megy az adat, azt még elmenti közben egy másik állományba, mintha megcsapolnánk közben a csővezetékét. Annyira ez a koncepció, hogy nem csak a cső jelnek, hanem az alapprogramnak is ez a neve. Minden fájl koncepció nagyon praktikus tud lenni.

Tehát itt a végszó a „UNIX az iskolában”, hogy aki nem ismeri a történelmet, az arra van kárhóztatva, hogy megismételje, ez nagyon igaz. Ezt úgy kölcsönöztem a SIMH szimulátor fejlesztőinek az előadásából, ezt az idézetet. Ez azt hozza fel, hogy gyakorlatilag egy 8 kbyte-ra optimalizálni valamit, mondjuk egy PDP 7-esen egy folyamatot, az gyakorlatilag olyan szintű feladat lehet, mint mondjuk egy modern beágyazott rendszerben egy 8 kbytes cache-re optimalizálni valamit. Nem változnak a dolgok, újra előkerülnek, nem árt, ha az ember ismeri ezt.

A UNIX nem történelem. Ez a Western Digital Mybook egy Linuxot futtat, az újabb változatok, a word változatban van egy Ethernet port rajta és az ember rá tudja kötni magát az internetre. Nem tudom mennyire szabályos, de itt letölthető egy olyan kiegészítés, hogy az ember be tud rá jelentkezni és akkor felmásolja rá a torrent szerverét és akkor lekapcsolva a gép a winchester tölt le az internetről magától, meg oszt meg, mert megy a torrent és osztja meg az adatokat. Itt az ember nem látja, ha lenne itt egy iPhone, akkor mutogatnám, de egyre inkább ez lesz. Egyre több helyen most már a mobiltelefonokban az Android platform megy.

Itt van az amit az előbb láttunk, egy BASIC értelmező. Ez egy 20 soros program, ami azt csinálja, hogy be tudom írni a BASIC parancsokat, ki tudom listázni, megőrzi, ha kell elmenti a save paranccsal, betölti. Ha el tudom indítani a programot, akkor az meghívja az internetet itt egyből. A program futását szabványos interrupt szignállal meg tudom szakítani, ctrl+r-rel. Tehát azt csinálja, hogy elindítja magát a BASIC-et, igaz, hogy nem C-re fordítja le, hanem awk nyelvre fordítja le awk programmal és utána, amint elindul, itt látható a második sorban, hogyha kap egy kettes szignált, itt gyakorlatilag írhattam volna „int”-et vagy „sigint”-et is. Régebbi hír, hogy a belső az kezeli az „int”-et is. Ha kap egy ilyen szignált a program, lenyomom a ctrl+c-t, ami megállítás lenne, azt elkapja, csapdába ejti a „trap” paranccsal. Meghívja ez a stupid függvényt, ami kiírja azt, amit az előbb láttunk a BASIC programnál. Szerintem nem ezt írja ki a BASIC fordító eredetileg, de mindegy, kiírja. Kétsoros programmal, ha nem is ilyen rendszerszintű, de nagyon komoly dolgokat valósítok meg két sorban. Gyakorlatilag itt látható egy sorban, hogy mit tud a program. Ez a program nem más, kiír egy tömböt és rendezi a számok alapján, hogy sorrendben legyenek a program sorok. Ezt elküldöm a compiler-nek, az lefordítja, ami egy fájl eredményez és azt futási időbe be tudom forgatni, az awk-t megadom paraméterként, ez egy ilyen átmeneti fájl lesz, ez az a szintaxisra és utána ez gyakorlatilag lefut. Ha lefut akkor kiírja, hogy „ready”, ha nem fut le, akkor megszakítom mondjuk a ctrl+c-vel. Gyakorlatilag már csak azt kell megmutatnom, hogy milyen az interpreter, amit compilernek hívtam. Ez egy kicsit hosszabbnak tűnik, azért mert meghagytam itt egybe a sorokat, gyakorlatilag azt kell nézni, hogy a belső utasításokat a „goto” vagy éppen „print”, „return”, ezeket átalakítja és kiír egy awk sort. A végén amit kapunk, indítsunk példa programot, pl. az f2, valamilyen bas program. Itt látható egy kettős „for” ciklusban egy program, kiírja a szorzótáblát vagy valami hasonlót, ezt átírányítom ennek a compilernek, akkor kapok egy olyan programot, lefordítja awk programra, ami egy végtelen ciklusban gyakorlatilag nem csinál mást,

egy hosszú feltételes utasítást csinál. Megadja, hogy mi a következő programsor, mondjuk a 10-es, akkor visszafut a végtelen ciklusban, megkeresi a 10-es sort, végrehajtja a 10-es sort. Ezt az awk programot, ami most kijött, átmentem az x.awk állományba, utána ha az awk programmal lefuttatom, akkor lefut a szorzótábla.

Mennyire UNIX-os ez a megközelítés? Az abszolút UNIX-os, az hogy programokkal készítünk programot. Sőt itt gyakorlatilag programmal készíték egy olyan BASIC programot, amit egy másik program lefordít szintén egy újabb programra, és azt a programot a legelső program elindítja. Gyakorlatilag azt lehet látni, hogy komoly rendszereket lehet alkotni, itt egy BASIC interpretert, pár sorral.

Futtathatóvá teszem a „chmod” paranccsal, ez is egy régi parancs. Ez gyakorlatilag egy héjban írt életjáték program. Lassú a gép, ahhoz, hogy futtassam, de látni fogunk azért néhány dolgot. Ez a Conway féle életjátéknak szabályai és gyakorlatilag héjból fut az egész. A gnuplot nevű program fogja az egészet kirakni. És a gnuplot az nem más, mint egy diagram rajzoló program. Képes arra, hogy azt mondom neki, hogy plot, és a plot-nak idézőjelek között megadok egy kötőjelet, akkor a standard bemenetről fog adatsorokat olvasni. Nem csinállok mást, kiszámítjuk az awk programmal különféle fázisait, optimalizálva, hogy csak az életjáték pontjait tárolja el. Átméretezi magát a root-ot, tehát nagyon barátságos ez a rendszer. Így egy sorba ez az életjáték demonstrációnak. Meg maga a diagramrajzolás is ezért kényelmes, hogy átméretezi magának. Az awk program azt csinálja, hogy minden egyes ilyen képernyőképet átad a gnuplot-nak és azt mondja, hogy ezt most rajzolja ki, az e-vel lezárja az adatszakszót, azt kirajzolja, addig kiszámolja a következő fázist, megint átadja, azt kirajzolja. Igazából ez volt a hiányosságom, hogy nem tudtam grafikus üzemmódban futtatni a héjt, nem tudtam a programozást bemutatni. Sikerült a gnuplot-tal egy jó megvalósítást találni ilyen módon.

Ebben a hónapban készítettem egy-két olyan héjprogramot, ami abszolút UNIX-os gyökerű a megközelítése is és olyan céljaim vannak vele, az egyik egy olyan függvény, amit táblázatkezelőbe lehet beilleszteni és arra szolgál, hogy átírja a megadott számot egy számnévvé, vagy pedig hozzáírja a pénznemet is, ha úgy akarjuk megkérni. Annyiban UNIX-os ez a megközelítés, hogy a példaprogramot awk-ban megcsináltam, be se fejeztem, mert láttam, hogy ez működni fog, aztán átírtam Python nyelvre és amint ez megtörtént gyakorlatilag készítettem az OpenOffice.org-hoz egy ilyen kiegészítést, kiterjesztést, amit megnézzük, hogy itt telepítve van-e? Hozzá fogjuk adni, mert azt hiszem ez itt nincs telepítve, bemegyünk a lock könyvtárba és itt kéne lennie egy „number textnek”. Most kiderül, hogy elindul-e. Most elindítom újra, mert telepítettem, elindítom rögtön a táblázatkezelőt. Elvileg azt tudná ez a kis program, hogy ha beírom, hogy „number text”, akkor átírja valamilyen szöveggé. 454676. Ha most megadom, hogy alapértelmezett dokumentum nyelvet használ, nem tudom el fogja-e ezt fogadni. Pontosvevesszöt kell még írni és kiírja olaszul. \$a2 ide beírom, hogy 45. Nézzük az angolt. A franciát, azt nem csináltam meg, de megcsináltam a thaiföldit. Van a manitex függvény, gyakorlatilag a pénznemet is hozzáírja.

Mi köze ennek a UNIX-hoz? Az, hogy ennek a modelljét egy egysoros awk programba írja be, gyakorlatilag egy nyelv független szöveggenerátor, ami a szám alapján átalakítja tetszőleges, természetes nyelvnek a nevére. Meg kéne mutatnom, hogy mi áll e mögött. Ez gyakorlatilag regionális kifejezéseket használ. Nagyon egyszerű táblázatban úgy néz ki ez, hogy írom be a számokat, ez rekurzívan meg fogja hívni magát ez a függvény, be tudom írni, ha egy vessző kell, szóköz vagy éppen nagybetű vagy másképp kell írni. Nagyon egyszerű a szintaxis, azt remélem, hogy ez OpenDocument szabvány lesz. Azért mert gyakorlatilag az Excel 2003-ban megjelent egy bahttext nevű függvény, ami a thaiföldi változatot megcsinálja. A Wikipédia és részben a program alapján néztem meg, hogy a nagyobb thaiföldi számokat hogy írják illetve hogy írják pénznemmel kombinálva. Tehát megvan ez a bahttext és a szabványosító bizottság, aki ezt az open formulát csinálja, azon törte a fejét, hogy mit csináljon ezzel a bahttext-tel, mert iszonyú csúnya, hogy most van egy függvény, ami megcsinálja Excelben a thai átalakítást, a megadott számot átírja thai nyelvre, mikor más nyelveken is szükség volna ilyesmire. Az OpenOffice.org is kompatibilis akar lenni, ott is megcsinálták ezt a pat teszt függvényt. A fejlesztő azt nyilatkozta, hogy ez egy rémálom, nem lehet ilyet megcsinálni. Meglehet, csak UNIX-os megközelítés kell hozzá. Úgy kell

hozzáfogni, hogy ha programozok, akkor írjunk valami olyat, ami programozható. Itt például programozható egyszerű szöveges fájlal adatbázis, kiegészítés és megcsinálja azt, amit különben nem lehetne.

**A másik ami készül, az a nyelvhelyesség-ellenőrző az OpenOffice.org-hoz.** Gyakorlatilag ugyanilyen módon azt fogja tudni csinálni, hogy egy szövegfájlal programozható, egész összetett kifejezéseket is le fog futtatni, azért, mert a Python nyelv, amit használ, az eredeti héj utasításoknak megfelelően rendelkezik az „eval” nevű függvénnyel vagy képességgel. Az „eval”-al megadok egy programsort és azt lefordítja és végrehajtja a rendszer. Tehát nem magát a programsort adom meg, hanem szöveggként megadom a programot, azt lefordítja és beilleszti. Itt rettenetesen hatékony ez a megközelítés, mert a reguláris kifejezéseket C nyelvű modulok futtatják le a Pythonban is. Egy-két másodperc alatt kiszámol nekem ezer ilyen hosszú számnevet, úgy, hogy nincs optimalizálva a Python program. Úgy lenne optimalizálva, hogy a lefordított reguláris kifejezéseket, amiket automatán fordít le a rendszer, azt én elmenteném, de azt már nem csináltam meg.

## Szervác Attila: Zeneszerkesztés szabad szoftverrel

A mindennapi zenei munkámhoz szerettem volna olyan szoftvereket találni, amikkel egy rugalmasabb munkafolyamatot lehet összeállítani, mint a tulajdon-korlátolt szoftverek világában. Néhány nagyon alapvető példát szeretnék bemutatni azon a világon keresztül, melyikkel én foglalkozom, tehát a közoktatásban például egy általános iskolán belül egy énekórán előfordulhat. Ha abból indulunk ki, hogy a fiatalokban megvan az az igény, hogy kifejezzék önmagukat zeneileg, és aztán szeretnék átadni, kérdés az, hogy milyen lehetőségeik vannak mondjuk az oktatásban. Ezt egy analógiával tudnánk elképzelni, ha azt vesszük, hogy verseket írni a legtöbb gyerek szeret, vagy valamikor megpróbálkozik vele, vagy amikor megtanul beszélni ilyen mondókákat, halandzsa mondókákat mond magának stb. Akkor hogyan tudja ezt a történetet, hogyan, milyen formái vannak ennek, hogy ezt terjesszük. Ez lenne itt az első kérdés. Ha átgondoljuk, hogy egy mondóka terjesztésének mi a formája, és azt is végiggondoljuk mondjuk egy történeti vonalon, illetve, hogy egyáltalán milyen technikai lehetőségek vannak erre, akkor kezdődik a dolog a szájhagyománnyal, és aztán befejeződik azzal, hogy manapság egészen könnyen hozzájut egy fiatal a hangot is rögzíteni képes videokamerához vagy be van építve a telefonjába stb. És egy történetet, amit ő kitalál akár úgy is fel tud venni és tudja terjeszteni.

Kezdjük az alapoktól a legalapvetőbb technika, amit az általános iskolában is tanulunk, a zenének az írásbelisége, vagyis a kotta a leguniverzálisabb, nagyon jó kiinduló alap, mert egy nagyon kis kompakt állományba le tudjuk képezni ezt a dolgot tulajdonképpen szöveges alapon. Ez mint egy kiinduló eszköz nagyon jó lehet, ezek a kicsi fájlok, amiket így el tudunk kezdeni gyártani, utána ezeket lehet megosztani, lehet exportálni mindenféle hangmintákkal lehet párosítani, és utána lehet exportálni hangzóanyaggá, ilyen-olyan opciókkal össze lehet keverni stb. Az első dolog, amit szeretnék megmutatni, hogy hogyan lehet szabad szoftverekkel kottákat írni, és aztán abból hangot csinálni. Biztos többen közületek találkoztak már olyan feladattal, hogy mondjuk képet kell előállítani, egy rajzot. Egy rajzot azt milyen formátumban lehet leírni?

**Hallgató:** képpontos.

**Előadó:** a képpontos az igazából nem rajz, hanem kép. A rajz, aminek vannak kis elemi dolgai, egy kör vagy valami stb., azt fogalmilag le lehet írni. És ahogy azt leírtuk, mondjuk egy SVG fájl, az mit tartalmaz? Senki nem nézett még bele?

**Hallgató:** koordinátákat.

**Előadó:** és milyen formátumban? Hogy néz ki egy Postscript fájl? Egyébként ezt az Adobe Illustrator fájlról is el lehet mondani ezt, mert az gyakorlatilag egy Postscript fájl bizonyos fejléc elemekkel.

Az egy egyszerű szöveg. Tehát ez egy gyakorlatilag TXT fájl, ami leírja szöveges alapon, hogy innen ide húzom a vonalat stb. Ez egy példa arra, hogy egy képet egy szöveges állományba le lehet írni, lehet definiálni. És ez a legtömörebb formája annak, hogy a legkisebb adatállomány jön ebből. Végülis mikor azt mondjuk, hogy zene, egy dallamot szeretnénk csinálni, egy dallamot hogyan tudunk leírni? Ha a hangokat megnevezzük, az is egy szöveg. Ma már elmondható majdnem minden területről, persze vannak kivételek, a videó állományokról elmondhatjuk, hogy nem szöveges alapúak, egyébként majdnem mindent szöveges alapon írunk le. Ha azt mondjuk, hogy az OpenOffice Wordban sima XML meg stílusleíró fájlok adják ki végül is azt a képet vagy annak a rajzprogramjába vagy prezentációs programjába, akkor ez máris egy megközelítése. Vagy egy WEB oldalnak, tehát annak is egy szöveges leíró fájlba van. Tehát tulajdonképpen gyakorlatilag mindennek. Tehát nyilván egy zenét is nagyon jól lehet erre írni.

**Erre van egy rendkívül egyszerű leíró nyelv egy de facto standard, ez pedig nem más, LilyPond nevű kottagrafikai nyelv, ami egy LaTeX alapú valami, és rendkívül rugalmas, igazából bármelyik stílustörténeti korból tehát akár beat zenét akarunk vele kottázni, akár népzene,**

akár egészen modern zenei érzéseket, ezt mind meg lehet vele csinálni és rendkívül egyszerűen meg tudjuk ezt tenni vele. Egész egyszerűen fogunk egy közönséges szövegszerkesztőt, az én esetemben ez a nano lesz és elkezdünk vele szerkeszteni egy fájlt. Ki az aki dolgozott már TeX-szel? Az egyes egységeket azokat kapcsos zárójelbe tudjuk belefoglalni, ez tulajdonképpen ez olyan mint az SGML nyelvekben felvesszünk egy elemet, van egy nyitótag és zárótag, az körbefog egy egységet, az egységeket itt a kapcsos zárójelek képzik, Tehát, akarunk csinálni egy dallamot, a kapcsos zárójelbe beírunk két hangot, mondjuk A és C, ezzel gyakorlatilag már alkottunk is valamit, aminek nézzük az eredményét.

Itt van egy forrásunk, ami logikus zenei hangokat tartalmaz. Ebből el tudunk indulni mindenféle irányba: le tudunk belőle gyártani egy kottát, egy MIDI fájlt, mindenféle kimenetet tudunk belőle csinálni, akár kép, akár hang. Mindezt a LilyPond szoftverrel fogjuk elvégezni a forráskódból a tárgykód előállításával, átadjuk neki a fájlnevet, és egész egyszerűen lefordítjuk. És megnézzük hogy mi történik. Az alapértelmezett kimenet az egy kotta kép. Ha akarunk csinálhatunk olyat is, hogy egy zenefájlt, esetleg MIDI fájlt állítunk elő, ez PDF fájlt generál. Nagyon nem vagyunk meglepődve,

ha azt mondjuk hogy ez egy TeX alapú nyelv. A kimeneten látjuk, hogy írhattunk volna mellé egy normális fejléctet. Itt az Evince PDF nézegető. Megnyitjuk benne a keletkezett PDF fájlt, és itt van a két hang amit beirtunk, de mi nem írtuk mellé semmit, sem azt, hogy milyen oktávban legyenek ezért egy kotta gyönyörűen előállt belőle valamilyen értékekkel. Ha azt akarom igazolni hogy ebből lehet valami bonyolultabb dolgot is csinálni, ezt is viszonylag könnyen meg lehet tenni. Írok még bele néhány hangot, tudok bele nyolcadokat is írni, illetve több szólamot is lehet csinálni. Alapvetően abszolút ábécés hangokkal dolgozik, ami azt jelenti, hogy én amikor azt írtam be, hogy A és C és nem írtam hozzá oktávot, akkor ez a „a,c” hangoknak felel meg. Ha át akarom rakni egy vonalas oktávba egyszerűen teszek egy vonalat, és akkor a hang egyből egy magasabb oktávba lesz. Vannak különböző módjai ennek a nyelvnek. Legtöbb zenére az igaz, hogy nagyon ritka benne a nagy ugrás, általában a tercek a legnagyobb ugrások, ezért van ennek egy relatív módja ennek a szövegnek. Mint minden trackerben a rep karakterrel kezdődő kulcsszavakkal tudunk bizonyos parancsokat átadni. Azt mondjuk, hogy az itt beírt hangok szintén egy egységbe foglaljuk, és azokat egy relatív módban fogjuk tárolni. Akkor fordítsuk le még egyszer, és megnézzük az eredményt. Ezt egy Postscript sablon tárolja, hogy hogy fog kinézni az egész oldal tördelés, és utána megcsinálja belőle a PDF fájlt. Rá is frissít, mivel észreveszi, hogy változott a fájl. Tehát a kezdeti „a” után nem lefelé ugrott mert a közelebbi lépés irányt választotta. Nem írtunk ütemmutató sem, ezért alaphoz 4 negyedre feltételez, és látható, hogy kiszámolja hova esik az ütem. Ettől függetlenül van rá lehetőség, hogy az ütemvonalat be is szűrhetem, maximum sikítani fog, ha olyan helyre szűröm ahol épp nem ott tart az ütem. Ez igazából magának egy jelzés. És ha van megjegyzés akkor a TeX nyelvekben „%” jellel jelöljük.

**Még egy dolgot szeretnék megmutatni, ami látványos és életszerű. Tudunk több szólamot is csinálni.** Ha szeretnék minden olyan dolgot ami egy időben történjen, azokat „<< >>” operátorok közé tesszük, és az abban lévő kifejezéseket egy időben fogja ütemezni a program. Most van nekünk egy kifejezésünk, egy együtemű, szöveg, a „[ ]”-ben és hozzáteszünk egy másikat. Miben előírunk egy új vonalrendszert, és abba is beírunk egy pár hangot, ha minden jól megy akkor meg is csinálja,

és generál egy összevissza kottát. Ezzel szerettem volna szemléltetni, hogy ez egy rendkívül egyszerű eszköz, amivel pillanatokon belül komplett zenét lehet szerkeszteni. Amivel akár több oldalas szövegeket tudunk bevinni, amit a szoftver automatikusan megcsinál, amiből egy rendkívül szép olvasható kimenetet kapunk. Tehát ezért is érdemes az oktatásban használni, mert nagyon egyszerű ezzel előállítani a kis példákat, amiket a hallgatók kezébe oda tudunk adni. Látszik hogy ezt megtanulni nem nagy tudomány, és bármilyen olyan gyerek aki ismeri a betűket és tudja hogy vannak zenei hangok, pár perc alatt meg tudja tanulni. Ez a tudományos része a dolognak. Most áttérünk a populáris részre, hogyan tudjuk mindezt kattintással elérni, azaz mindent grafikus felületen csináljunk.

A grafikus zeneszerkesztők amik vannak a Linux alatt, ezek a szabad szoftverek mind kompatibilisek a LilyPond program nyelvel, legalább olyan szinten, hogy lehessen őket exportálni tudnak bele. Tudjuk hogy minden grafikus szoftver mögött van egy szöveges háttér, egy leíró nyelv, nem is beszélve arról, hogy a LilyPond-ban vannak olyan utasítások amik messze túlmutatnak a leíró nyelv képességein. A grafikus szoftver mögötti leíró nyelvet könnyen lehet minden elemét paraméterezni, egymásba ágyazni tetszőleges mélységben, stb. A grafikus felület mindig le van maradva ehhez képest. Fejlesztjük a háttérrel, és fejlesztjük hozzá a monitort. Tehát a grafikus felületet nem lehet úgy fejleszteni, hogy valahol ne legyen korlátja, hogy korlátlanul egymásba ágyazni valamit, mert egyszerűen nem tudná máshogy megjeleníteni. De amikor a forrásfájl írom akkor akármilyen mélységben tudok dolgozni. Tehát ezek után el tudom mondani, hogy a Linux alatti szabad grafikus zeneszerkesztő programok mindegyike kompatibilis a LilyPond-dal, legalább olyan szinten, hogy exportálni tudunk bele. De ha ez kevés, akkor az alapot a grafikus felületen összedobom, és forráskódban további finomításokat tudunk elvégezni.

**Egy olyan szoftver, ami ezt valamilyen szinten vállalja is,** hogy még olvasni is tudja ezt az adatot, szerencsére még olyan is van, ez pedig nem más, mint a kb. 2 éve viszonylagos fejlődésnek indult, MuseScore. Ezt a Werner Schweer nevű úriember, aki ezt csinálja nagyon régóta ebben nyomul, az ilyen zenei programok készítésében. Elsősorban ilyen MIDI alapú dolgokat szokott csinálni. Amikor az ember ezt először telepíti és elindítja, akkor ezt csinálja, hogy az első tételt iderakja nekünk, a sima zongorás változatot, és akkor lehet látni, hogy ezt, amit írunk, amikor a LilyPond közvetlenül dolgozunk van egy olyan parancs, hogy `\midi` parancs azzal elő lehet írni neki, hogy ne csak a PDF fájl, gyártsa le vagy esetleg azt egyáltalán ne, viszont gyárt egy MIDI fájlt, és ott meg lehet mondani, hogy a vonalrendszerhez milyen hangszert párosítunk és akkor ezt előállítja.

Tapasztalatból mondom, hogy stílusa válogatja, ez mennyire jön ki jól. A MIDI fájlok nagyon limitáltak abban, hogy mit tudnak, például a ritmikai ábrázolásuk nagyon durva. Ha összetettebb ritmusokat ír az ember, akkor már se füle, se farka az egész történetnek. Ezzel csínján kell bánni, mindenestre ilyen hétköznapi feladatokra használható, és itt a grafikus felületen is csak egy kattintás. A lényeg az, hogy ez így működik. Nézzük meg akkor, hogy mi hogyan alkotunk ezzel. Ez egy rendkívül fejlett felületű program, keresztplatformos tehát Windowson, és más Posix jellegű rendszeren OS X-en is fut. Úgyhogy, ha ezt egyszer felfedezik maguknak azok a buta zenészek, akik olyan buták, hogy még az OS X-et is el lehet nekik adni, akkor ez tetszeni fog nekik, mert ez egy nagyon kényelmes, buta zenész szemléletnek. Tehát az ember azt mondja, hogy itt egy új kotta, beírja a címet, meg az alcímét meg beírja a szerzőt, hogy xy szerzőt és gyakorlatilag ilyen projekt jellegű. Nyilvánvaló, ha vokális műről van szó, akkor a szöveget is el lehet sütni LilyPond fejlesztésben is vagy akármilyen speciális, tulajdonképpen mindenre fel van készülve a gitártól kezdve nagyon sok modern zenei eszköze is van. És akkor választunk hangszereket, mondjuk választunk magunknak egy fuvolát, meg egy csellót, ez egy ilyen klasszikus összeállítás, honosítva még nincs a szoftver.

Megmondom őszintén vannak olyan ambícióim, hogy megcsináljam, elég nagy munka és nagyon sokat fogok foglalkozni a honosítással, de valószínűleg előbb-utóbb be fog ez következni. Remélem, hogy valaki megelőz ezzel a dologgal, de jó pénzért mondjuk van itt 50.000 Ft, egy hét alatt megcsinálom. Előbb-utóbb úgyis kénytelen leszek. Szóval ezeket is hozzáadjuk, nyilván egy vonalrendszerben több szólamot is lehet kezelni, általában ez nem cél, de mindenestre erre is lehet gondolni. Lehet előjegyzést, amennyiben ilyen klasszikus zenével szeretnénk dolgozni. Elő lehet írni az ütemmutatót, felütés egyéb dolgokat lehet csinálni. Az a jó, hogy nem kell trükközni. Dolgoztam még régen, 6-8 évvel ezelőtt kereskedelmi kottairó programokkal is, folyton trükközni kellett, ha az ember úgy akarta, hogy szaggatott ütemvonal legyen, egy felütés vagy valami, tehát ha nem ilyen szögletes volt a zene, mint a szimfóniáé, akkor folyton valamit trükközni kellett a programmal.

Nem könnyű találni egy olyan zenei programnyelvet, ami leírja a zenét, mert a zene nem ilyen merev hierarchia, hogy vannak nagyobb egységek, azon belül kisebbek stb. Ennek ellenére van

music XML nevű formátum is, amiről tudni kell, hogy az szigorúan egymásba ágyazós. A zenében van ilyen, hogy az egyik dolog átmegy a másikba stb., meg azért nyilván látható, hogy nem lehetetlen. Hozunk egy ütemszámot az automatic 12 megcsinálja nekünk és akkor nagyon egyszerűen, mindig szoktam mondani, hogy a menüből mindent megtalálunk, ha valaki kérdezi, hogy mi hol van? Mindenesetre van két módja ennek, egy kijelölő mód meg egy beviteli mód gyakorlatilag. Tehát először a beviteli módra váltunk és onnantól kezdve, hogy ezt megtettük akkor lesz itt egy kis kurzorunk és akár a billentyűvel, gondolom én, ki kell választanunk egy ritmusértéket, amiket be fogunk vinni és akkor elvileg működni kéne.

Ez alapvetően egy kotta orientált dolog, a kották nagyon jó dolgok, mert a legrugalmasabban fel lehet használni mindenféle kimenet és minden ilyen komolyabb zeneszerkesztő keretrendszer tud kottákkal dolgozni, leképezni kottákra a dolgokat. Viszont vannak másféle megközelítései is a zenének, ilyen a hang orientált megközelítései. Amikor nem is feltétlenül MIDI fájlokat akarunk gyártani, mert azok ugye buták. Erre van egy klasszikus technika, ami szerencsére nagyon aktívan él a mai napig is, mondhatni a legősibb számítógépes ilyen hangzózenét előállító modell. Ez pedig nem más, mint a trackerelés.

**A tracker programok használata.** Itt úgynevezett hangmodulokat lehet előállítani, közvetlenül megszólal, rendkívül élvezetes velük dolgozni. A használt és preferált Debian disztribúcióban több ilyen tracker program is van. Tulajdonképpen az ilyen tracker program az arról szól, hogy látszólag teljesen más, mint az előző, hogy kottát írunk, de nem. Végül is mindig arról van szó, hogy van nekünk egy időfolyamatunk, elindul egy adott időpillanatban, lezárul valamikor és valamilyen egységekben mérjük közben az időt és bizonyos időpillanatokban megszólal egy hang, végül is a zenének a ritmus az alapja és tulajdonképpen erről szól az egész dolog.

A tracker programok is erről szólnak. A múltjuk egészen abba az időbe nyúlik vissza, amikor már először elég erősek voltak a hardverek, ez például a Commodore, Amiga és társai, ezek a nagyobb teljesítményű egykártyás gépeknek a kora. 4 MHz-en, 8 MHz-cel hasonlóval órajellel működő processzorokkal működtek. Ezek, amik már képesek voltak arra, hogy kicsi fájlokban leírt hangmintákat, amik valamilyen hangszínnel megszólalnak és kiadnak valamilyen hangot az egyik egy ilyet, a másik egy olyat, azt valós időben szoftveresen összekeverjék. Tehát mindenféle hardveres támogatás nélkül, pusztán a szoftver kap egy ilyen hangmintát, meg egy olyat és azt mondta neki valaki, hogy ezt a kettőt most egyszerre kell megszólaltatni, és ezt képesek voltak a szükséges idő alatt olvasni ezeket a hangmintákat és ezeket a szükséges idő alatt összekeverni és a kevert eredményt kiadni a gépeknek. Tehát magyarul amit látunk nagyon egyszerű. Vannak sávok, ugye trackek, különböző formátumok különböző mennyiséget tudnak kezelni ebből, a legtöbb program által használt xm kiterjesztett hangmodult. A formátum az nagyon sok, akár 64 ilyen szólammal is tud dolgozni, az a lényeg, hogy egy sávban egyszerre egy hang szól, praktikusán egy hangszert szoktunk egy sávhoz összerendelni, de ez nem kötelező. Tehát lehet ott váltogatni, hogy éppen melyik sávon, melyik hangszert használom, egyszerűbb esetben az átláthatóság kedvéért. Az a feladatunk, hogy be kell töltenünk nem is közvetlenül hangmintákat, különböző hangszereket leíró hangminták, hanem azoknál kicsit összetettebb hangszerfájlok, de lényegében hangmintákat tartalmaznak. Ezekből egyre több helyen van, freesound, ahonnan le lehet ilyen hangmintákat tölteni. Megpróbálok egy fuvola hangmintát megnyitni. Amikor kiválasztottam egy ilyet, hogy ez nekem tetszik, akkor választok egy sávot, és abban a sávban bizonyos időpillanatokban elkezdem bevinni a hangokat, a space-t kell megnyomni és ettől ilyen szerkesztő módba vált át a dolog. A billentyűkiosztást lehet válogatni, de alapban egy ilyen zongoraszerű billentyűkiosztásként fogja értelmezni a PC billentyűzetet, úgyhogy itt zongorázhatok a billentyűzettel, vihetek be hangokat. A bevitt hangokhoz különböző egyéb értékeket is lehet rendelni, tehát itt az elején megmondja, hogy milyen hang. Tehát így be lehet vinni ezeket a hangokat. A lényeg az, hogy amikor bevitem ilyen hangokat, akkor utána ezt le tudom játszani, lehet ismételni, a bevitt motívumokat újra és újra felhasználni. Annyit csinálunk, hogy kiválasztunk egy hangszert, az adott hangszer hangszínével valamilyen sávban elkezdek rögzíteni hangokat. Az hogy mi a ritmus, azt kiadja az, hogy milyen távolságban vannak egymástól ezek a hangok, tehát ha tá-ti-tit akarok, akkor kihagyok először 4-t,

aztán, 2-2-t stb. Fölveszem a hangot aztán átmegegyek egy másik sávba, mondjuk praktikusán azt mondom, hogy ott hozzáadok egy másik hangszert. Ebben a fájlban ezeket a hangmintákat hozzáadom, az eredmény, ami keletkezik gyakorlatilag tartalmazza a zene logikáját, a zenei szöveget, hogy milyen hangmagasságok és milyen ritmusok illetve, hogy melyik csatornán szól, tehát ezzel is lehet játszani, hogy most jobbról szól, most balról szól stb., nagyon érdekes effekteket lehet ezzel csinálni. Ez egy nagyon kicsi adatmennyiség, ezen kívül még hozzáadok hangmintákat, az egy nagyobb adatmennyiség, viszont a lényeg az, hogy egy kicsi hangmintát kell csak egyszer hozzáadnunk például egy fuvola az egy mp-ig szól, azt én tudom bármilyen hangmagasságban alkalmazni. Nagyon kicsi hangminták vannak csak benne, amikkel mégis ki tudom tölteni az egészet, és eredményként azt kapom, hogy egy nagyon kicsi fájlom keletkezik, ami akár egy egész zenekarnyi, zenekarokban használt különböző hangszereknek a kicsi hangmintáit tartalmazza. Gyakorlatilag pár száz kilobyte-os méretekre kell gondolni, viszont tetszőleges méretű zenei szövegekhez tudja alkalmazni ezeket a kicsi hangmintákat, nagyon kicsi, kompakt kis fájlom lesz. Olyan hangmintát tudok választani vagy csinálni, ami nekem szimpatikus, és tetszőlegesen összetett hangzást elő tudok vele állítani. Tehát van kijelölve a fájlban egy hangbank amiben ki van jelölve az egyesnél egy fuvola, most átmegegyek a hangbanknak egy másikra pozíciójára és hozzáadódik a második csatornára egy újabb vonós hangszert, amit most fogok kicsomagolni egy RAR fájlból. Szerkesztő módba váltok és akkor itt is be tudok vinni a hangokat. Lejátszva az idáig szerkesztett dolgot eléggé kortárs hangzást sikerült produkálni.

**Tulajdonképp más megközelítésből ugyanazt adja mindkettő.** Elindulhatunk egy kotta irányból, ami vizuálisabb, a gyerek részére érthetőbb, és ha nem kell a grafikus felület, de ha először a hangszínekkel szeretne játszani akkor használja ezt a tracker csomagot, de lényeg mindig ugyanaz lesz különböző hangsávokban visszaadni hangokat, valamilyen ritmusokkal, és akkor ha kész ezt el tudja menteni és utána hallgatni. Ez kétfajta megközelítése annak hogyan alkossunk zenét. Meg lehetne nézni mi van akkor ha több eszközöm van nagyobb multimédiás eszközökkel rendelkezünk, és végignézünk egy ilyen teljes folyamatot hogy saját magam valamilyen programnyelvben generálok egy burkológörbét, hogy milyen legyen az én hangszínem én magam állítom elő ezeket a hangfájlokat az egészet beteszem egy hangszerkesztőbe, és ott effektelem és visszhangot rakok rá, stb. Ennek van egy szép folyamata. De egyszerűen és játékosan bármilyen életkorban, ezekkel a programokkal el lehet kezdeni. Talán sikerült meggyőzőnöm mindenkit hogy könnyű elkezdeni a zeneszerkesztést.

# Szabad szoftver nap szekció

## Höltzl Péter: Modern naplózás GNU/Linux alapokon

Felépítés: a naplózás alapelvei, syslog-ng felépítése, trükkök, példák, tanácsok

A naplózás az egy elég öreg dolog, az első rendszernapló 1951. Naplózni kell. Az első syslog daemon-t Eric Allman készítette 1980 körül. Ő volt az az ember, aki a sendmail projektet is csinálta annak idején. Amikor csinálta, azt vette észre, hogy a Unix rendszereken a naplózás az eléggé mostoha dolog és ezért írt egy külső naplógyűjtő alkalmazást, ezt hívták úgy, hogy syslog daemon és azt találta ki, hogy majd az alkalmazások nem foglalkoznak a napló tárolással, hanem majd csak a /dev/log-ba vagy ahová akarja, alapvetően a /dev/log-ba küldik az üzenetet, és akkor majd ez a daemon fogja és letárolja az üzeneteket, és nem kell vele foglalkozni. Ez annyira jól sikerült, hogy egy szabvány is lett belőle, úgy hívják, hogy RFC 3164, de ez az RFC elavult, én négy példát hoztam fel, de ennél jóval több hibája is van. Az egyik például az, hogy nagyon elavult az időpecsét formátuma, nincsen benne év és nincsen benne időzóna, ami 2009-ben kissé kellemetlen. Eric Allman amerikai volt és elképzelhette volna, hogy van egy cég, aminek két külön telephelye van, mondjuk a keleti parton meg a nyugati parton, teljesen más időzóna, még sincs időzóna kezelés benne. A hálózati forgalmat UDP alapon kezeli és nincsen benne rejtjelezés, ez megint csak nagy hátránya a hagyományos syslog protokollnak. Nem kezeli az üzenetvesztést, nem kezeli az üzenetduplikátumokat, több soros üzeneteket sem kezeli és nem tudja sorba rendezni pontosan emiatt a bejövő üzeneteket, ami korreláció elemzésnél megint csak kellemetlen lehet. Tehát volt ez a régi típusú protokoll, ez a 3164, egyébként még mindig ez a hivatalos és ez a stabil, ugyanis van egy új RFC, illetve nem is egy, hanem rögtön öt, az 5424-től 5428-ig. Ez az új syslog protokollt váltja le és ez már kezeli az összes hibát, igazából nincs is neve, IETF syslog protokollnak szoktuk hívni alapvetően. Ez már kezeli a TCP-t, kezeli, ha SSL-ezünk vagy TLS-en küldünk, kezeli a normális időpecsétet, a többsoros üzenetet, sőt a unicode-os üzeneteket is kezeli, nem úgy, mint a régi syslog.

**A régi syslog:** nagyjából három részből áll a syslog üzenet, az elején van egy PRI nevezetű mező, a priority rövidítése, ez úgy néz ki, hogy van minden üzenetnek egy facility-je, meg minden üzenetnek van egy severity-je és úgy számoljuk ki az értékét, hogy a facility-t megszorozzuk 8-cal és hozzáadjuk a severity értékét. Ezzel a mezővel kezdjük. Utána következnek a fejlécek, headersnek hívják. Így néz ki egy header: van benne egy dátum, van benne egy időpecsét, óra, perc, másodperc egy hostnév, egy programnév, utána jön a PID, a végén pedig szabad szöveges naplóbejegyzés. Így kéne kinéznie hivatalosan egy log üzenetnek.

**Az új protokoll:** az IETF-nél évek óta marják egymást, és dolgoznak azon, hogy hogyan nézzen ki ez az új protokoll, ugye itt most már nem csak Unixok vannak, hanem a picipuhának is vannak kívánságai, és hát évek óta működik, nagyon büszke vagyok rá, hogy az egyik kollégám tagja annak a csoportnak, akik ezt az új protokollt kitalálták, egyébként ő írja a syslog-ng-t nagyjából, nézzük, hogy néz ki az új formátum. Annyi a különbség, hogy a PRI-t meghagyták ugyanúgy és a headert kicsit kibővítették. A header úgy néz ki, hogy most már van benne egy verziószám, az a bizonyos 1-es az elején, utána következik egy időpecsét, ez úgynevezett isodate formátumban van. Ugye van év, hónap, nap, utána jön egy T betű, óra, perc, másodperc és utána az időzónának a jele. Utána jön a hostnév, ami most már végre FQDN, ha megnézzitek a régit, az csak sima hostnevet tartalmaz, az új pedig végre tud FQDN-t. Utána jön a programnak a neve, utána esetleg a PID-je, ha van, utána jön az üzenet ID, utána jön a strukturált adat és utána következik maga az üzenet. Az üzenet úgy néz

ki, hogy jön egy ilyen BOM, ez speciális karakter, azt jelenti, hogy innentől kezdve Unicode-ban jön az üzenet, és akkor érkezik meg maga az üzenet unicode-os formában. A Unixok még nem támogatják ezt a strukturált adatot, illetve elég kicsi a támogatottsága ennek az új protokollnak, mert hogy még draft állapotban van, tehát nem elfogadott, ettől függetlenül a syslog-ng már támogatja, meg pár más alkalmazás is, rajtunk kívül. Nézzük meg, hogy miből áll ez a strukturált adatmező. Ez egy mező lesz a headerben, opcionálisan metainfókat fog tartalmazni az üzenetről. Úgy kéne kinéznie, hogy ilyen adatblokkok vannak, minden egyes blokk szögletes zárójelek között van, a blokkok között nem lehet szóköz, de a szögletes zárójelen belül lehet. Azon belül úgy néz ki a dolog, hogy ilyen név=érték-vel lehet beállítani mindenfélét, például a sorszámát az üzenetnek stb.

**Az a kérdés, hogy miért van szükség egyáltalán logolásra?** Az egyik legfontosabb az, hogy szeretnénk tudni, hogy mi történik a rendszerben. Log elemzés. Ennek két oka lehet, az egyik az, hogy ha jól működik a rendszer és szeretnénk azt látni, hogy milyen olyan események történnek, amik nem igazán odavalóak például: hibás bejelentkezés, próbálkozások, mindenféle rossz szándékú aktivitás, ez az egyik fele. A másik, hogyha szeretnénk hibát keresni. Én felírtam még az egészséges paranoiát, ami minket biztonság technikusokat jellemez. Én azt szoktam mondani, hogy nem vagyok paranoiás, engem tényleg üldöznek. Nagyon fontos feladat még a logolásnál a központi archiválás. Az utóbbi években a politika is ráugrott erre a dologra, ezért külső szabály, külső kötelezettség lett a logolás. Ez iskolában nem igazán fordul elő, de minden máshol igen, akár a PSZÁF-tól a PCI-ig meg van még sok más szabályozás, amivel megpróbálják ránk kényszeríteni, az informatikai rendszereket üzemeltetőkre, hogy igenis naplózzunk és archiváljuk is a naplóinkat.

**Pár szó a syslog-ng-ről.** A syslog-ng 1998-ban született, most már 11 éves. A legelterjedtebb syslogd alternatíva per pillanat és kb. 300 ezer gép van a világban, felméréseink alapján, akik syslog-ng-vel logolnak. Ha ránéztek a levelezési listánkra van egy ilyen, egy mailman-os listánk, ahol bizony szoktak érdekes email címről válaszolni emberek, tehát ilyen army.mail meg nasa.gov, meg ilyenek, néha meglepődünk. 2007-ben kicsit kettéválasztottuk a dolgot, a fejlesztést. Egyrészt kettéágazott OSE edition-nek nevezett valami, ami igazából a marketingeseinknek a csalása, mert full GPL-es, tehát én inkább azt mondanám, hogy GPL edition, és lett egy premium edition, ami egy fizetős verzió, amiben általában fejlesztjük az új funkciókat és aztán utána csorgatjuk át a GPL-be. Most is 4 nagyon fontos dolog csorgott át a 3.0-ba, ahogy ezt az Ubuntu konferencián már megígértem. Most például nem olyan sok van a premium edition-ben. Erre azért volt szükség, mert nagyon sok helyen az az elvárás, hogy csak akkor használják a terméket, hogyha van hozzá support, meg van mögötte cég.

**Nézzük meg, hogy mi a különbség és mit tudnak ezek az új verziók.** Az OSE változat, az ugyanúgy, mint a premium edition, helyi és távoli naplógyűjtésre képes. Megbízható protokollon, tehát TCP-n tud küldeni. Van egy eléggé fejlett szűrési és osztályozási rendszer, természetesen ipv4-et és ipv6-ot is támogatunk, és ami most lecsorgott, az egyik, az a bizalmas napló továbbítás, az SSL csatornák támogatása, ez kiment GPL-be illetve kiment a közvetlen adatbázis kapcsolat támogatása is, tehát most már tud a syslog-ng adatbázisban logokat tárolni, és majd még mondok hozzá egy-két trükköt. A premium edition változatban pedig két funkció van per pillanat, az egyik az a diszk buffer, ez arra van kitalálva, ha nincs elég sávszélességünk, de túl gyorsan jönnek az üzenetek, akkor le tudjuk tárolni egy diszk buffer-ben a logokat, és amikor majd lesz sávszélességünk és időnk rá, mondjuk például éjszaka, akkor majd kiküldjük az üzeneteket. A másik pedig a rejtjelezett napló tárolási formátum, ez is per pillanat egy premium edition-ös dolog, ez egy titkosított fájl, amiben tároljuk az üzeneteket, van ahol erre szükség van.

**Nézzük meg, hogy hogyan naplóznak a programjaink per pillanat egy tipikus Linuxon.** A legegyszerűbb a saját naplófájlnak a megnyitása és írása, ilyen például az Apache vagy a Squid, ők azt csinálják, hogy megnyit egy fájlt vagy többet, és oda hányja bele az üzeneteit. A másik az megint nagyon elterjedt, hogy a /dev/log-ba elkezdünk írogatni. Ez egy device a /dev alatt,

azt megnyitjuk és arra kezdünk el írogatni. Aztán lehetőség van még a logger nevezetű paranccsal logokat küldözgetni. Az is lehet, hogy egy alkalmazás csinál egy hálózati kapcsolatot vagy egy adatbázis kapcsolatot és oda logol. Miért jelent ez számunkra problémát? Azért, mert nem egységes. Ha megnézel egy feltelepített Debiánt, én Debian és Ubuntu hívő vagyok alapvetően, ha megnézed fel van telepítve több száz, akár ezres nagyságrendben csomagok, ahol mindenféle alkalmazások futnak. Szerencsénk van, ha 10%-a gyárt logokat és mindegyik a saját módszere szerint, azt nem nagyon lehet központilag karbantartani, ha mindenki hasraütésszerűen naplóz. Tehát nem egységes a módszer, nem egységes a hely. Általában /var/log alá, de mindenfelé szoktak küldözgetni logokat. Én láttam kár /tmp-ben is logokat, ami elég hajmeresztő. Virtualizáció és chroot-ok esetében még ott is van mindenfelé mindenféle log küldés. Az utolsó legfontosabb, hogy nem egységes a formátum. Össze-vissza formátumokat használnak és ezért nehéz kezelni, nagyon nehéz korrelációt elemezni, ilyen együttállásokat.

**Hogy is működik a syslog-ng belülről?** Alapvetően úgy néz ki a syslog-ng, hogy vannak forrásai, source-ok, és a source-ökbe érkeznek az üzenetek. Igazából source driverek olvassák, és a syslog-ng ezeket a source-öket folyamatosan pollozgatja, ellenőrzi, hogy érkeztek-e bejövő üzenetek. Ha beérkezett az üzenet, akkor fogja és átrakja a destination-ökbe. Tehát vannak nekünk céljaink, ahol megírogatjuk a logokat és oda rakjuk át. Az, hogy melyik forrásból melyik célba azt az úgynevezett log path-ban tudod összeállítani. Én igazából log routingnak szoktam mondani. Van még egy pár apróság. Vannak filtereink, tudunk szűrni mindenfélére és vannak template-jeink. A template arra van kitalálva, hogy üzeneteket tudjunk átformázni. Például neked nem tetszik a hagyományos időpecsét formátum, akkor fogod és átírod. És ami a 3.0-ba jött be, az a két új funkció, az egyik az úgynevezett parserek. A parserek arra vannak kitalálva, hogy a beérkező üzeneteket tudjuk értelmezni. Ha jól strukturált egy üzenet, akkor fel lehet vágni kisebb darabokra. Ugye az nagy probléma volt, hogy alapvetően azt mondja a syslog protokoll, hogy van egy stringünk, úgy hívják, hogy \$msg és abban van benne az üzenet és azt mondja, hogy ez egy üzenet és azzal kell valamit kezdeni. És a 3.0 előtti időkig a syslog-ng azt mondta, hogy ez egy string. Ha csináltál egy adatbázist, akkor fogtuk és bele tudtuk rakni egy darab mezőbe ezt az egy stringet, csak hát ebben keresni nem nagyon kényelmes. Ha ez egy normálisan strukturált üzenet lenne, mondjuk az Apache meg a Squid csinál ilyeneket, és az tényleg normális, akkor fel lehet parse-olni. Megmondod, hogy hol vannak a delimiterok, külön apró részekre vágjuk és akkor így be lehet rakni normálisan egy adatbázisba, sokkal könnyebb és gyorsabb benne keresni. És van még egy másik funkciónk, úgy hívják, hogy rewrite, arra van kitalálva, hogy üzenet tartalmat tudjunk átírni. Ez mondjuk kicsit hajmeresztően hangzik, de lehet rá szükség például, ha neked nem az a struktúra tetszik, amiben az eredeti alkalmazás küldi vagy azt szeretnéd, hogy anonimizáljalj üzeneteket, mert azt szeretnéd, de az alkalmazás ezt nem teszi lehetővé. Sőt az egyik leghajmeresztőbb az az, hogy a pppd képes arra, hogy password-öket tegyen bele log üzenetekbe, amit aztán elküld UDP-n valahova, az kicsit félelmetes. Nem kéne.

**Konfigurálás(1):** van a source driver, ami arra van kitalálva, hogy egyfajta metódussal tudjon logokat olvasni. Rengetegféle source driverünk van. Ezeket össze kell gyűjteni egy source-be, ez egy nevesített gyűjteménye a source drivereknek. Mindegyiknek van kis címkéje és megmondod, hogy hol kell olvasnia. Az írás ugyanerre a rugóra jár, vannak destination drivereink, amik írni tudnak és ezeket ugyanúgy össze kell fogni egy úgynevezett destinationbe. Filterben tudod meghatározni a kereső vagy a szűrő kifejezésedet, ez szintén egy nevesített objektum. A log path az egyébként egy nem nevesített dolog, ott pedig össze tudod rendelni, hogy melyik forrást melyik cél felé kell továbbítani. Vannak template-jeink, ez szintén nevesített, ahol meg azt tudod megmondani, hogy hogyan kéne átírni, milyen formátumra kellene átírni egy ilyen üzenetet. Vannak még opcióink, amik a globális paramétereket tudják megmondani a syslog-ng-nek. A parser arra van kitalálva, hogy darabolgassuk szépen, és ahogy szétdaraboltuk, az egyes darabjait. Először is mi határozzuk meg, hogy hol kell szétdarabolni, másrészt makrókat tudunk ráállítani és attól fogva hivatkozhatunk rá. A rewrite üzenet átírásra, átstrukturálásra való.

Van még egy újdonság a syslog-ng 3.0-ban, hogy most már lehet elágazni a log path-okban, régen nem lehetett, úgy volt, hogy voltak source-ok, filterek meg destination-ok és kész. Most már lehet azt mondani, hogy van egy bejövő forrásom, odaadom egy filteren keresztül egy destination-be, utána rakok még egy filtert és továbbküldhetem, de akár el is ágazhatunk belőle, újabb source-okat lehet hozzá rakni. A 2.1-es ágból milyen funkciókat csorgattunk át az OSE ágba? Most az SQL támogatás került át, átkerült az SSL, TLS támogatás, és ez egy új dolog, az IETF-nek az új syslog protokolljának a támogatása is. Ez is ott van GPL-ben és szabad használni. Ha valaki azt szeretné, hogy az új protokollban szeretne küldeni vagy az új protokollban szeretne fogadni, akkor van végre valaki, aki támogatja.

A következő az időzóna kezelés. Nagyon rossz a hagyományos syslog protokollnak az időpecsétje. Ha a kliens alkalmazás, aki küldi, beállítja, akkor természetesen mi is fel fogjuk használni ezt az információt és tudjuk kezelni. Előfordulhat, hogy maga a küldő nem kezeli az időzónát és az év információt, azt a source driverben is meg tudod mondani, hogy bindoljuk rá egy IP-re meg egy port-ra, és aki erre az IP:portra küld üzenetet, annak megmondod explicit, hogy ez az időzónája és akkor ezt fogjuk beállítani. Ha sehol nincs meghatározva, sem a source driverben sem a küldő nem határozza meg, akkor mi se kezeljük, természetesen. Még egy, hogy vannak rá makróink, elég sok az időre és az időzónára.

**Konfigurálás(2):** az új syslog-ng-nek kicsit megváltozott a dolga, úgyhogy most az /opt/syslog-ng alá települünk. Ez újdonság, mert nagyon sok platformon és operációs rendszeren fut a syslog-ng, a legextrémebb: AS/400-on is van most már syslog-ng, de AIX-tól kezdve a legrégebbi HP Unix-on keresztül az összes Linux disztribúcióban van syslog-ng. Hogy néz ki egy konfigurációs fájl? Vannak opciók, source-ok stb. Ha konfigurálunk valamit, akkor egy objektumot kell előállítanunk. Ez az objektum úgy néz ki, hogy megmondjuk a típusát, megmondjuk az ID-ját, ha ez egy nevesített objektum és akkor kapcsos zárójelek között a paramétereit és a végén egy pontosvessző kell. Kivétel volt a globál és a log path, mert azok nem nevesítettek, akkor ott hiányzik az ID. Így néz ki egy forrás meghatározás: azt mondod, hogy source, megmondod a nevét és a source driver típusát, ami lehet egy fájl egy TCP kapcsolat egy syslog protokoll és utána a paramétereit, amit rewrite-ból ki lehet szedni. Egy darab source-be lehet több source drivert is felsorolni. Kétféle iskola van alapvetően, az egyik azt mondja, hogy ahány forrásom van, annyi source objektumot hozok létre, a másik azt mondja, hogy egy source objektumom van és az összes drivert belerakom és szűrök. Ez ízlés kérdése, egy nagyon fontos, hogy egy source drivert csak egyszer használjunk, például hálózati kapcsolat esetében, ha két helyen használod, akkor gáz, mert nem tud elindulni, azt mondja, hogy már foglalt ez az IP:port és nem tud rá bindolni. Egy source drivert csak egyszer használjunk és utána tegyük bele annyi log pathba amennyibe szeretnénk. Lehet két driverrel ugyanazt a fájlt olvasni, csak az a baj, hogy ebből a szempontból a syslog-ng nem bolondbiztos. Lehet 2-3-szor olvasni ugyanazt a fájlt, ha szükség van rá, csak ezzel el fogod érni azt, hogy üzeneteket fogsz duplázni, triplázni. Előfordulhat, hogy egy üzenetet beír valaki és megszázaszorozzuk, nagyon sok helyet elfoglalva.

**A következő driverek vannak,** például az internal, az a syslog-ng saját belső üzeneteihez, de tudunk UNIX stream-et vagy datagramot olvasni, fájlt, named pipe-ot, TCP-n, UDP-n tudunk kommunikálni, ez a hagyományos protokollt jelenti per pillanat. Tudunk SUN-on SUN stream-et olvasni. Syslog-nak hívják az IETF-nek a protokollját. És van egy újdonság, ez a program nevezetű dolog, ez azt csinálja, hogy elindít egy programot, a standard outputját felolvassa és beteszi a logok közé. Megmondod, hogy van egy destination, van egy ID-je és belerakod a drivereket, így például lehet továbbküldeni egy hálózati kapcsolatban, UDP-n. Lehet akár konvertálni is UDP-n olvasunk, TCP-n küldünk tovább, ez teljesen tipikus felhasználás. Sajnos egyes hálózati eszközök, a routerek, switchek meg ilyenek kizárólag UDP-n logolnak. Ha mégis az van, hogy te szeretnéd látni az összes aktív eszközödnél a logjait, akkor azok UDP-n fognak logolni. Például a PIX, ASA, azok is csak UDP-n logolnak. Azt mondja a gyártó, hogy mi az RFC szerinti használjuk. Ilyenkor egy megoldás van, nagyon közel le kell rakni egy un. relayt, ez a relay syslog-ng semmi mást nem csinál, csak

fogadja UDP-n az üzeneteket, esetleg átalakítja, rak rá tisztességes időpecsétet és TCP-n továbbítja. Ezzel meg lehet oldani, hogy biztosan megérkezzenek ezek a log üzenetek. Természetesen a destinationben ugyanúgy lehet több driver.

**Támogatottak:** támogatjuk a fájlba írást, a pipe-ba küldést, UNIX datagramot, TCP, UDP, userTTY, ez a belépett usernek a termináljára én ezt szoktam legelőször kiírni belőle, mert rettenetesen zavaró. Alapvetően a rootnak szokták küldeni a nagyon fontos üzeneteket, csak ezzel az a baj, hogyha beléptél, és mondjuk valami pity-puty történt és sok a nagyon fontos üzenet, akkor tönkreteszi a munkádat. Lehet a program standard inputjára írni, mondjuk log elemzésnél.

A Swatch egy online log elemző alkalmazás, mintákat lehet neki megadni. Itt a programból a Swatch-nak küldöd a standard inputjára. Én általában a signal 11-et szeretem figyelni vele és küldjön róla e-mailt vagy SMS-t. Logot elemezni lehet egyrészt periodikusan, utólag, mondjuk óránként egyszer átnyalazzuk az üzeneteket, de mondjuk az nem biztos, hogy elég gyors, meg van, amikor ritkán van a dolog, és akkor lehet olyat is csinálni, hogy azonnal küldjön riasztást mondjuk e-mailban.

Az SQL adatbázisba írás igazából a libdbi-t használjuk, amit az támogat, azt mi is támogatjuk. SQL, MS SQL, PostgreSQL, MySQL, Oracle, azt hiszem ezek vannak most.

Syslog protokollon is tudunk továbbküldeni, ez is egy jó lehetőség, mondjuk, ha UDP-n fogadod a hagyományossal, akkor tudod syslog protokollal továbbítani, ha szükséges. Így néz ki egy log parse-nak a leírása, nincsen ID-ja, felsorolod az összes forrást, lehet egynél több, felsorolod a filtereidet, parsereidet, ha kell, a rewrite szabályaidat, ha kell, fontos lehet a sorrend, mert lehet, hogy a parserrel értelmezett mezőn akarsz rewrite-ot csinálni és azt csak így lehet. Aztán felsorolod a célokat, hogy hová kéne továbbítani, meg lehetnek bizonyos flagek. Nagyon egyszerű syslog-ng konfiguráció, tehát van valamilyen forrásunk, simán a loopback-en figyelünk és küldjük tovább TCP-n valahova. A log path az láncolható. Tehát van egy forrásunk, van egy filterünk és ez belemegy egy célba, és mondjuk, tovább szeretnéd szűrni, akkor beraksz az útvonalba még egy filtert és egy másik célt. Olyat is tudunk csinálni, hogy itt nem egy destination következik, hanem egy másik ilyen log üzenet. Itt látszik, hogy egy elágazás történik, mert ami a destinationre megy, azt ráküldjük mint forrás mind a két log sorunkra, amit ide beleraktunk és itt csak egy további szűrőt, parsert, destinationt tettem. Idáig még lehetett volna az előző módszerrel, csak mivel itt még szeretnék hozzáadni plusz forrásokat, ezért egy ilyen elágazást gyártottunk bele, ahol ennek az utolsó sornak két forrása van. Egyrészt, ami a filteren keresztül megy, azt is megkapja, ide még másik forrásokat lehet belerakni, ha szükség van rá.

**Beszéljünk a szűrőkről:** a szűrők megint csak nevesített objektumok, ahová ilyen kifejezéseket lehet bepakolászni, például a hostnévre lehet szűrni. Rengeteg szűrőnk van, lehet szűrni a facility-re és az üzenet prioritásra, de ezt nem ajánlom senkinek. Lehet szűrni a hostnévre, a programnévre, a match-el bármilyen regexp-szel lehet szűrni. A netmask, az egy speciális szűrő, ez a feladó IP címére szűr, kicsit olyan access control jellege van, bár én ezt inkább csomagszűrőből csinálnám meg. És a filternek lehet része másik filter. Úgy néz ki egy szűrőkifejezés, hogy van benne egy szűrőfunkció, megmondod azt, hogy mire kell rászűrni és utána lehet bele többet is rakni, közé pedig logikai operátorokat lehet rakni. Ez ugye azt mondja, ha az sshd program az example nevezetű hostról jön, és benne van az üzenetben az, hogy deny, akkor fog matchelni. Természetesen ki lehet ezt váltani három külön filter bejegyzéssel, és utána belerakjuk mind a hármat az útvonalba és „összefésélednek”, de itt fixen AND. Lehetne használni NOT-ot meg OR-t, ha szükségünk lenne rá. Nyilván, ami itt bejön, azt beleöntjük ebbe a filterbe, ami abból kijön, azt beleöntjük a következőbe és így tovább. Tartalmazhatnak regexeket, de ezt sem ajánlom használni, mert nagyon lelassítja, tehát tudsz belerakni a hostnévre, azt mondod, hogy router.\*1 és akkor úgy. Nyilván a speciális szimbólumokat, azokat escape-elni kell, sőt a backslash-et is escape-elni kell.

**Template:** arra van kitalálva, hogy egy üzenetet átformáljunk. Az üzenet átformálás az úgy néz ki, hogy a template-ekben makrókat tudunk használni. Van egy elég hosszú leírása, de például facility,

priority, date, isodate, év, hónap, hét, nap, óra, perc stb., ilyeneket lehet összeállítani és magunknak csináljuk ezeket. Tehát ha szükséged van rá, akkor magadnak tudod ezeket összerakni. Például azt mondod, hogy isodate-tel kezdődjön, jöjjön utána a hostnév és az üzenet, amit küldött a kliens. Utána a template-et magánál a destination-nél lehet használni. Tehát azt mondod, hogy fájlba szeretnék írni, ez a fájl a /var/log/messages lesz és a t\_demo template szerint átalakítva magát az üzenetet. Természetesen nem fontos külön objektumban definiálni a template-et, azt lehet inline is definiálni, itt nem a nevét mondod meg, hanem magát az üzenetet is belerakhatod. Én ezt azért nem ajánlom, mert sokkal jobb kívül megcsinálni magát a template-et, és utána mindenhol használni, ahol kell. Ha kell változtatnod, akkor mindenhol át kéne írni.

**Időpecsét kezelés:** ez megint egy nagyon érdekes dolog. Kétféle időpecsét makrónk van: az egyik „s”-sel kezdődik, a másik meg „r”-rel. Ez nagyon fontos, az „s”, az mindig az időpecsétből, tehát kapunk egy logüzenetet, abban van egy időpecsét az elején azt szépen parse-oljuk, azokra lehet hivatkozni az „s” betűvel kezdődővel. Ha „r”-rel kezdődővel, akkor meg annak a szervernek a rendszerórája lesz az etalon, amelyiken éppen fut a syslog-ng. Ugye az a kérdés, hogy mit akarsz tárolni? Én egyébként mind a kettőt. Azt szeretném tárolni, hogy mit hazudott a kliens, amikor küldte az üzenetet, de az is érdekel, hogy a syslog szerveren konkrétan mi volt az idő, szerintem mind a kettőt érdemes eltárolni. Tehát az „s”, az a küldő oldal, az „r”, az meg a fogadó oldal. Én azt szeretem, hogy mindig „r”-eseket használjak és utána az üzenettörzs végére, zárójelbe vagy akárhogy odacsapom azt, hogy ő eredetileg mit mondott magáról.

**Tanácsok:** Az első, hogy ne szűrjünk facility-re! És egyáltalán a PRI mezőre ne szűrjünk, mert az teljesen ad hoc. Van, aki használja, van, aki nem használja, a legtöbb alkalmazás a local0-val küld. Az nem igazán megbízható.

A másik az, hogy van egy olyan opciónk, hogy ts-format, ha opcióba beírod, hogy ts-format(iso), akkor fixen mindenhol isodate-re konvertál a syslog-ng.

A következő probléma, hogyha névfeloldást szeretnétek, akkor az nagyon lassítja, visszafogja a rendszert, mert mindent fel kell oldani. Ha a névszerverek elérhetetlenek, akkor megint csak hatalmas gáz van. De lehet azt mondani, hogy a syslog-ng egyrészt DNS-t cache-eljen, a másik pedig, meg tudod mondani, hogy mennyi legyen a lejárata. Én alapvetően IP címeket szoktam eltárolni, mert az a biztos vagy eggyel biztosabb talán, és azokat visszaoldjuk, ha szükség van rá. Sőt azt is lehet, hogy host fájl alapján fogjuk és nem DNS feloldással, ha valaki mindenképpen host neveket szeretne látni a logokban.

**Chroot-ból hogy érdemes logokat összegyűjteni?** Én nagyon sok chroot-ot használok, az egyik megoldás, ha a chroot-ban egy /dev/log-ot hozunk létre. Nem kell létrehozni kézzel, mert azt a syslog-ng megcsinálja induláskor nekünk, ha azt látja, hogy nincs, akkor legyártja. Megmondod, hogy hol van a chroot-od és a /dev/log-ot csinálja meg. A chroot-ban futó alkalmazások pedig írnak a /dev/log-ra ügyesen. A másik lehetőség az, hogy a chroot-ban van egy /var/log/messages, azt fogjuk és letöröljük és csinálunk mknod-dal ügyesen egy named pipe-ot és kívülről meg olvassgatjuk. És ez nagyon jó lesz, mert a benne futó alkalmazás azt hiszi, hogy egy fájl ír, holott nem, és magán a chroot-on belül nincsenek üzenetek. Ennek az a hátránya, hogyha megáll a syslog-ng valamiért és nem olvassa ezt, akkor a benne lévő alkalmazást blokkolhatja és megáll, ami egy kritikus alkalmazásnál okozhat problémát. Vannak vállalatok, ahol vannak security-sek, akik azt mondják, hogyha nem tudjuk logolni, hogy mit csináltunk, akkor nem szolgáljuk ki a klienst. Az Apache például ilyen, ő azt mondja, ha nincs log alrendszer és nem tudok logot írni, akkor nem szolgál ki. De ha azt mondjuk, hogy fontos, hogy ne álljon meg a rendszerünk, mert mondjuk egy lélegeztető gépet üzemeltet, akkor azt csináljuk, hogy fogunk egy fájl és felolvassuk. És lehet még a bind mounttal olyat is csinálni, hogy a /dev-be mountoljuk a /var/chroot4 alatt a /dev alá.

**A klogd-t is ki tudjuk váltani,** ha szükség van rá, ez ugye a kernel log daemon, azt így tudjuk megcsinálni. Azt mutatja, hogy a log prefix szépen belerakja az üzenet elé, hogy „kernel:”. Ez egy

példa arra, hogyan kell SQL-ben tárolni PostgreSQL adatbázist, megmondod, hogy hol van, milyen user, milyen passworddel, melyik adatbázisba, szépen definiáljuk a táblákat, az oszlopokat, belerakjuk makrókkal az értékeket, az indexeket beállítjuk és kész. Ez volt a program. Így a cat-nek a standard inputjára küldünk üzeneteket vagy a Swatch nevezetű alkalmazásnak is tudunk küldeni üzeneteket.

**Logrotate:** gondolom, mindenki használja, a Debian apt-get install-lal felpakolja. A logrotate úgy működik, hogy van egy alkalmazásunk, aminek megmondjuk, hogy mely fájlokat kell rotálnia, megmondod, hogy hány generációt szeretnél, milyen gyakran és akkor ő fogja és rotálja ezeket a fájlokat, azért, hogy a log fájljaink ne nőjenek az égig. Azt csinálja, hogy a messages-ből alaphelyzetben csinál egy messages.1-et, de előtte, ha volt .1, akkor abból .2-t stb. A .365-ből, azt meg fogja és letörli. Annyit tud még, ha ide berakod, hogy compress, csak ennek az a baja, hogy ki az, aki csinál napi mentést? Hogy ha rsync-kel vagy bármivel napi mentesz, akkor ugye a rotálás következtében ugyanazt a fájlt fogod áthozni minden nap, csak más néven, ez meg nem teljesen optimális. Megfogjuk és átnevezzük, mondjuk postrotate-ből lehetne olyat csinálni, hogy a .1-et mindig átnevezed év, hónap, napra, de szerintem sokkal szebb megoldás, ha a syslog-ng-nek a fájl útvonalába azt mondod, hogy /var/log/archive/\$HOST/\$YEAR/\$MONTH/\$DAY.log.

**Logrotálás SQL esetében,** ha azt mondod, hogy ha tábla névben használsz hónap, napot ilyeneket, ugye akkor automatikusan létre fogja azokat a táblákat hozni. Ugye az se jó nekünk, ha egy nagyon nagy táblában van minden. SQL-es tárolásnál arra készüljétek fel, hogy nagyon lassú maga a tárolás. Sokat ne küldjétek SQL-re, mert nagyon lassú lesz. Kb. ötöde a sebessége a fájl íráshoz képest vagy annyi se. Ellenben visszakeresni mennyivel kényelmesebb és gyorsabb.

**Teljesítménnyel kapcsolatban:** úgy működik a syslog-ng, hogy ha bejött az üzenet, akkor a beérkezés pillanatában feldolgozzuk, ami okozhat egy csomó problémát, túl sok erőforrást fog felzabálni maga a syslog-ng, de mivel ezeknek a bejövő oldalaknak van némi buffere, ezért lehet olyat csinálni, hogy az opciók közé ezt a time-sleepet beállítjuk. Ez azt jelenti, hogy az összes source végigolvasása után várunk 20 millisekundumot és utána kezdjük el. Ennyit az összes buffernek ki kell bírnia. Így viszont töredékére lehet csökkenteni a CPU terhelést, ellenben okoz némi latency-t, de azt gondolom 20 millisekundumot mindannyian kibírunk.

**Nagy mennyiségű üzenetre hogyan lehet optimalizálni még?** Kapcsoljuk ki a DNS feloldást mindenképpen, maximum használjuk az /etc/host-ot, kapcsoljuk be a flow-control engedélyezését. Ez a flow-control azt jelenti, hogy megmondjuk kifelé mennyi üzenet mehet, meg bejövő oldalon és ha elfogyott a windownk, akkor le tudjuk lassítani a küldő oldalt. Semmiképpen ne használjunk usertty destination drivert, ne használjunk regex-eket vagy legalábbis csökkentjük, és az UDP forgalom esetén a receive buffert azt növeljük, ha lehet.

**Kérdések:** A template az csak azt tudja neked megcsinálni, hogy van az üzenet és magát a PRI mezőt, meg a header mezőt meg az msg mezőnek a részeit felismeri és ott sorrendeket tudsz variálni illetve az időpecségeket tudod konvertálgatni. A rewrite az meg a programmezőre, a hostnévre és a message tartalomra vonatkozik, tehát tartalmon belül tudsz vele átírni dolgokat. Még hozzá úgy érdemes, hogy előtte ráeresztesz egy parsert, mondjuk van egy olyanunk, hogy CSV parser, megmondod, hogy hol van a delimiter, az szeparátor mentén darabokra vágja, elnevezed és megmondod, hogy a 8-as oszlopot szeretném átírni.

Logon belül még egy log és azon belül egy source. Erre példa: azt lehet csinálni, hogy ide rakod a hálózati forrásokat, leszűrsz belőle valamit és utána még a local. Mondjuk hálózati, leszűröd csak az sshd-t és ide rátolod, és utána azt mondod, hogy a lokális sshd-ről és erről leszűröd csak a violationokat és oda átrakod.

Microsoft windowsos eseménynapló feldolgozása: az a helyzet, hogy van a neten csomóféle ilyen megvalósítás, a nagy része alapvetően pénzes, például a nagy konkurenciánk az rsyslog, ő például azt mondja, hogy az rsyslogd az ingyenes, ez a default logger Debianban is, meg Redhat ágon is, viszont neki a kliense fizetős. Nekünk is van egy syslog-os fizetősünk és vannak ilyen tervek, hogy az szabad szabad legyen, ha nem akkor lehet keresni a neten olyat, aki összegyűjti a windowsnak az event logjait és tudja syslog protokollon tovább küldeni. Tele van vele a háló. Arra figyeljetelek, hogy az új 2008-as szerver ágon megszűnt az event log, illetve nem megszűnt, hanem már nem nagyon logolnak bele, mert van valami saját log API-ja vagy akármije és azt is kell támogatni. Utána neked már teljesen mindegy, mert TCP-n vagy UDP-n jön egy üzenet.

## Kis Gergely: Google App Engine

Ez egy alkalmazás platform, amit a Google csinált és az benne az érdekes, hogy a saját alkalmazásainkat a Google-nek közismerten jól skálázott infrastruktúrájában tudjuk futtatni. Ehhez készítettek egy fejlesztő környezetet, ami Python nyelvre van alapozva. További nyelvek támogatását is tervezik, de azt nem árulják el, hogy mi lesz a következő nyelv. A pletykák arról szólnak, hogy a Java lesz, de ez még később derül ki. Kisebb alkalmazásoknál ingyenesen használható, itt a kisebb az havi 5 millió oldalöltést jelent illetve 1 gigabyte tárhelyet. Azt hiszem sokan örülnének annak, ha a kis alkalmazásunk havi 5 millió oldalöltést tudna magához húzni.

**Mit keres ez az egész a szabad szoftver opcióban?** Maga az App Engine SDK, az szabad szoftver az Apache 2.0 licence alatt elérhető, le lehet tölteni. Ez egyrészt magának a programozási felületnek a megvalósítása. Nyilván nincs benne az a nagy megbízhatóságú adattár, amire az egész Google infrastruktúra épül, de elvileg hasonló adattárakat, például a typer-table-t vagy a Doop-ot mögé rakva lehetne saját Google App Engine klónt építeni ebből az SDK-ból kiindulva. Egyébként miután ez megjelent kb. 1-2 héttel utána az Amazonnak az ec2 rendszerére idézőjelben portolták, tehát nem nagy megbízhatóságú rendszert építettek, csak annyit, hogy ec2-es gépekre felrakták az SDK-t és azzal demózták, hogy milyen jó. Maga a rendszer szabad szoftverekre épül, Python nyelvű, angol alkalmazás keretrendszer, valamint az adatok tárolására illetve a backend rendszerekkel történő kapcsolattartásra pedig a Protocol Buffer szabványt használják, ami szintén a Google-tól érkezett és szintén nyílt forráskódú. Maga az a futatókeret, amin a Google futtatja az alkalmazást az zárt, tehát ahhoz nem férünk hozzá, hogy az hogyan működik, nem is biztos, hogy erre szükség lenne.

**Milyen szolgáltatásokat indít az App Engine?** Ad egy Python nyelven írt alkalmazás keretrendszert, egy nagy teljesítményű elosztható naptárat, egy olyan biztonságos és zárt környezetet, hogy a többi alkalmazás az ne tudjon belenyúlkálni a miénkbe, automatikus skálázódást, tehát nem kell külön azzal foglalkozni, hogy skálázódjon. Ez a marketing része, a másik oldala az, hogy megvan a lehetőség arra, hogy jól skálázódó alkalmazást írjunk, de ha ügyesek vagyunk, akkor ezen az infrastruktúrán is tudunk rosszul skálázódó alkalmazást létrehozni. Ezenkívül integrálódik a Google-nek a felhasználó kezelésével, illetve egyéb segédeszközöket is ad még a fejlesztéshez.

A Python nyelvet, azt sokan ismerik, a legtöbb mai Linux disztribúcióban a jelentős része a különböző beállító programoknak az Pythonban van írva, nyilván disztribúciófüggő, hogy mennyi része. A Python talán leginkább arról ismert, hogy a szintaxisában meg van határozva, hogy hogyan kell kinéznie a forráskódnak.

Itt van a Django keretrendszer, ami egy hagyományos modell-view-controller alapú keretrendszer, szintén ismert, sokan használják és megkönnyíti a webes alkalmazások fejlesztését. Az alapl működése az úgy néz ki, hogy a böngészőből jönnek a kérések, hogy meg akarunk nézni egy oldalt vagy valamit szeretnénk csinálni, az eljut a controllerhez. A controller az adatainkat betölti, az adatainkat manipulálja és utána a kigyűjtött megjelenítendő adatokat átadja a view-nak, ami pedig a böngészőnek visszaküldi azt a HTML, JavaScript kódot, amit meg szeretnénk jeleníteni, ez az alapl működése a modell-view-controller alapú rendszereknek. Ilyen típusú alkalmazás fejlesztésre ad támogatást az App Engine API.

**Az adattár szerintem ez a legérdekesebb része az App Engine-nek,** mert minden mást már láttunk máshol. Az adattár a Google-nek a saját BigTable megoldását használja. Ebben az a trükk, hogy nagyon sok számítógép tárolja az adatokat elosztva és ezekhez az adatokhoz nagyon gyors hozzáférést tud biztosítani. Nem egy hagyományos relációs adatbázis, emiatt másképpen kell kezelni, teljesen más módszereket kell alkalmazni, mint a relációs adatbázisoknál. Például a normál formákat azonnal el lehet felejteni, ha valaki idáig főleg relációs adatbázisokkal foglalkozott.

**Hogyan működik az adattár?** Az összetartozó adatokat egy csomagban tárolja, ezeket hívjuk entitásoknak, például egy diák entitás vagy egy személy entitás aminek az összes adatát megpróbáljuk egy csomagban tárolni. Minden egyes tárolt entitásnak vagy objektumnak van egy egyedi azonosítója, ez alapján lehet megtalálni az adattárban. Az entitások között kapcsolatokat is definiálhatunk például minden diák egy adott osztályba tartozik. Az adatokat persze le is kérdezhetjük, nem csak a kulcs szerint, nem csak az egyedi azonosítója szerint, hanem lekérdezhetjük a különböző adatmezők szerint, például kérem az összes fiú diákot, az 5. b-ből névsor szerint, hasonló lekérdezéseket lehet csinálni.

**Van egy SQL szerű leíró nyelv is,** de ugyanakkor van egy ilyen szűrő API, ami gyakorlatilag teljesen megfelel egymásnak. Valószínűleg az SQL-szerű lekérdező nyelvet csak azért rakták oda, hogy amikor van az összehasonlító táblázat, akkor ki lehessen pipálni, hogy van lekérdező nyelv. Lehet tranzakciókezelés illetve maga az adattár tranzakcionális, ami azt jelenti, hogy ha módosítunk benne valamit, akkor az azonnal megtörténik, atomi módon történik meg, ahogy az SQL adatbázisoknál megszoktuk. Ugyanakkor ez a tranzakció kezelés csak korlátozásokkal érvényes, ez az elosztott jellegéből fakad az adattárnak. Konkrétan egymással erősen összefüggő objektumokat tudunk egy tranzakcióban kezelni, ez az egyik, a másik, hogy lekérdezéseket nem tudunk tranzakción belül végezni. Ez azért van, mert a lekérdezéseket, azokat kicsit másképp kezeli az adattár. A lényeg az, hogy az elosztott jellege azt jelenti az adattárnak, hogy előfordulhat, hogy az egyik diákot az egyik szerver fogja tárolni, a másik diákot meg a másik szerver. Ezt az alkalmazásból alapvetően nem tudjuk befolyásolni, hogy hova kerül, kivéve, ha azt mondjuk, hogy az entity groupba tartoznak, ez azt jelenti, hogy nagyon közeli kapcsolatot definiálunk közöttük, szülő-gyerek relációt, és ebben az esetben a rendszer biztosítani fogja azt, hogy ők egy szerveren tárolódjanak.

**A tranzakciókezelés az úgy van megoldva,** hogy azokat az objektumokat tudjuk egy tranzakcióban kezelni, akik egy szerveren tárolódnak. Érthető okokból, mert ha megpróbálnánk nagyon elosztott szerverek között átívelő tranzakciókezelést megvalósítani, akkor rögtön a skálázódás egyik legnagyobb akadályát sikerülne behoznunk. Csak az egy entity groupban levő objektumokat tudunk egy tranzakcióban kezelni, illetve mint mondtam, mikor egy tranzakción belül vagyunk, akkor lekérdezéseket sem tudunk végrehajtani. A demóban majd megmutatom ennek a forrását, hogy hogyan működik.

**A lekérdezések másképp működnek, mint ahogy egy SQL adatbázisban megszoktuk.** Külön előre meg kell mondanunk, hogy az alkalmazásunkban milyen lekérdezéseket fogunk használni, ez azt jelenti, hogy meg kell határoznunk például a fenti lekérdezéshez, hogy egy olyan lekérdezésünk lesz, ami adott osztályból a fiúkat névsor szerint kéri. És erre, ha ezt definiáljuk, akkor a rendszer egy külön indexet épít és lekérdezéseket kizárólag indexekből tud kiszolgálni a rendszer. Ez azért van mert maga az adattár alapvetően kétféle műveletet tud, egy kulcs alapján megtalál egy adott sort, illetve egy kulcs prefixe alapján tud keresni, tehát egy kulcs eleje alapján tud keresni. Ennyit tud csinálni az adattár, viszont ezt nagyon gyorsan tudja és óriási adatmennyiségre. És ezért, amikor eltárolunk egy objektumot, mondjuk egy diákot, akkor egyrészt eltárolja magát az adatokat egy adott kulccsal, illetve külön indextáblákban eltárolja a megfelelő index sorokat. Ezeknek az index soroknak az azonosítója vagy a kulcsa, az gyakorlatilag az itt megadott feltételeknek egy szabályok szerint összeállított értéke, és erre az értékre tud aztán a lekérdező rendszer keresni, illetve ezt a prefix keresést végrehajtani.

**Milyen egyéb szolgáltatásokat nyújt még a rendszer?** Tud más weboldalokról letölteni adatokat, URLfetch szolgáltatás, levelet küldeni, ad egy átmeneti tárolót, a memcache-t, amivel a memóriában tudunk úgy tárolni adatokat, hogy azt a későbbi lekérdezések során elérhessük. Képek kezelésére ad még néhány egyszerű funkciót, hogy át tudjuk méretezni, elforgathassuk, illetve egy ilyen mágikus képminőség javító algoritmust tudunk még ráfuttatni. Röviden ezeket tudja az App Engine.

**Lássuk, hogy lehetne ezt az oktatásban használni, mire lenne ez jó?** Először is van a hagyományos programozás oktatás, amikor elindulunk Pascállal vagy C-vel, benyaljuk a szintaxist és utána elkezdjük a kedvenc rendező algoritmusokat, meg a különböző faredezést, ciklusszervezés stb.

Egy másik vonulat, amit főleg középiskolai oktatásban használnak, MS Access-szel összedobálunk egy ilyen adat alapú alkalmazást és akkor azzal készül el valamilyen adatkezelő eszköz. Ha ma ezt egy átlag középiskolásnak próbálnánk megmagyarázni, akkor kb. 5 percig figyelne ránk és akkor még érdeklődő volt az elején. Ha ezzel próbáljuk az átlag diákot megfogni, akkor ez abszolút unalmas lesz.

**Szerintem hogyan oktassunk programozást?** Válaszunk egy olyan környezetet, amit a diákok nap mint nap használnak, amit jól ismernek, tehát gyakorlatilag ebben élnek és ez manapság, hogy ha a platformokat nézzük, akkor ez a WEB, mindenki internetezik. Fent lógnak a Facebookon, iwiweznek, játszanak, fórumoznak, tehát ez az, amit minden nap használ. Olyan eszközöket tanítsunk, amik ezeken a platformokon jól működnek, ehhez a platformhoz kapcsolódnak és azokat a problémáikat, ötleteiket tudják megvalósítani, amik őket tényleg foglalkoztatják. Hogyan segít ebben az App Engine? Az App Engine-nel webes alkalmazásokat tudunk fejleszteni úgy, hogy ezeket ingyenesen, minden diáknak létre lehet hozni, külön egy App Engine accountot és teljesen ingyenes, mindenki 10 db alkalmazást regisztrálhat, felrakhatja, játszhat vele.

**Mik az előnyei?** Könnyen tanulható. Mondható az is, hogy kb. egyforma nehéz minden nyelvet megtanulni vagy nagyon különbözik. Azt értem itt a könnyen tanulhatóság alatt, hogy maga az App Engine nem ad túl sok eszközt. Tehát nem az van, hogy három vastag könyvet meg kell tanulnod ahhoz, hogy jól tudjál benne fejleszteni, hanem viszonylag rövid tanulási időszak alatt már elég jó dolgokat össze lehet rakni benne. Jól definiált mintákat kell követni. Ha bármilyen fejlesztő eszközt nézünk, akkor általában a fejlesztő eszközök nagyon megengedőek, mondjuk az ember összedobál MySQL-ben, PHP-ban vagy C++-ban vagy akármiben egy kis alkalmazást, akkor az nagyon sok esetben soha nem derül ki, hogy rossz mintákat követett és a rossz minták idegződtek be.

Az App Engine-ben, ha rosszul használja az adattárat, rossz mintákat követ, akkor nagyon gyorsan el fogja érni a kvótákat, nagyon gyorsan ki fog derülni, hogy valamit rosszul csinál. Ez a tanulás során nagyon fontos, hogy a hibás megoldások viszonylag hamar kiderülnek és látszik a különbség a hibás és a helyes megoldás között.

**Teljesítmény:** először megtanulunk gazdálkodni az erőforrásokkal. Az App Engine adminisztrátor felületén folyamatosan lehet látni, hogy mennyi erőforrást használnak az egyes lekérések, hol lehetne javítani a szoftveren. Konkrét lehetőségek, lehet hagyományos webes alkalmazásokat fejleszteni, de ami érdekes, az a másik kettő, egyrészt mashup-okat létrehozni, mikor létező komponenseket dobálunk össze például a Google Maps-et és csak hozzá rakunk valamit, hogy például a Google Maps-en jelenítsük meg a közös élményeinket és erre csinálunk egy kis alkalmazást, ami a mi adatainkat a saját Google App Engine alkalmazásunkban tárolja és magára a megjelenítésre pedig a Google Maps-nek a felületét tudja használni. Például egy ilyen mashup jellegű.

A másik ami még érdekes, hogy a Facebook és Iwiw alkalmazások fejlesztésére a Google App Engine kifejezetten ideális megoldást tud nyújtani. Természetesen vannak hívások is. Tehát ahhoz, hogy valójában webes alkalmazást tudjon fejleszteni valaki a Google App Engine-n, ahhoz két nyelvet kell megtanulni, egyrészt a Pythont, másrészt a HTML-t és a JavaScriptet. Erre mondhatnák akár, hogy három nyelv, és ha még hozzá vesszük a CSS-t, akkor négy.

Vannak itt is kihívások, de jól megválasztott keretrendszerrel, nem az alapeszközöket adjuk oda, hanem előre kiválasztott, összerakott keretrendszereket gyorsan lehet haladni. Ugyanakkor bonyolultabb, egész féléven átívelő projekteknél érdemes gondolkodni, mert ez szerintem általában is igaz a programozás oktatására. Ha úgy oktatjuk a programozást, hogy mindig csak egy 10 vagy

20 soros kis programmal mutatunk meg valamit, akkor pont azok a bonyolultabb összefüggések nem látszanak, amik csak a nagyobb alkalmazásoknál jönnek elő. Tehát ez nyilván több energiát igényel az oktatótól egy ilyen típusú féléves anyagnak az összeállítására, mint hogy ha mindig csak a jól bevált buborékrendezést kéne újra megtanulni. A kis óraszám megint probléma, tehát egy átlagos középiskolában, már ha eljutnak a programozásig, emelt szintű érettségire készülve, akkor is csak néhány óra van arra, hogy a programozás alapjait bemutassuk.

**A felsőoktatásban használni App Engine-t amiatt lenne nagyon hasznos,** hogy egy valóban skálázható architektúrát lehet megismerni. Egyszerűen informatikus mérnökök úgy kerülnek ki az egyetemről, hogy soha nem láttak skálázható rendszert. Nem tudják azt, hogyan kell azt építeni, nem is tananyag sok helyen, illetve ha tananyag, akkor csak távolról, hogy igen tudjuk, hogy valakik már csináltak ilyet. Elméletben el lehet mondani, hogy milyen elveket kell követni, de valóban kipróbálni egyet vagy használni egyet, megismerni a tényleg működő mintákat, azokat nagyon kevés helyen oktatják most. Ez az egyik. A helyes tervezési mintákat rögzítik, piacképes a tudás, mind a Python nyelv, mind maga az App Engine, önmagában mind a webes fejlesztés egy teljesen piacképes tudás, azzal biztosan el lehet helyezkedni. Ugyanakkor természetesen önmagában az App Engine nem helyettesíti a hagyományos „technológiák” megismerését, ugyanis a feladathoz kell az eszközt választani.

Tehát nem állítom azt, hogy minden feladatra a Google adattára a legjobb megoldás, vannak olyan feladatok, ahol a relációs adatbázisok sokkal jobban működnek. Például olyan feladatok, ahol szeretjük azt, hogy ha maga az adatbázis képest viszonylag bonyolult számításokat végrehajtani. Mert ebben az esetben, ebben az adattárban ilyen lehetőségünk nincsen, az App Engine ilyen lehetőségeket nem is nyújt, hogy háttér folyamatokat indítsunk el rajta, kizárólag a böngészőből vagy egy kliensből beérkező kéréseket tudjuk kiszolgálni. Ez egyébként a „legnagyobb kritikája” is az App Engine-nek, amit megfogalmaztak vele kapcsolatban, hogy pontosan a háttér folyamatokat nem lehet benne automatizálni. Különböző hack-ek létrejöttek, vannak ilyen időzítő szolgáltatások, amik egy adott URL-t meghívogatnak egy másik szerverről. Van ebből ingyenes fajta is, meg természetesen mindenki csinálhat magának, egy szerverre felrak egy kis programot, ami percenként 10x meghív egy URL-t az App Engine. Ezek működő megoldások, olyannyira, hogy a Google Androidnak a code review eszköze az első változat, azt Gerritnek hívják, az App Engine alkalmazás volt. Az úgy működött, hogy a git forráskód repositorykat kezelő rendszer az egy Javában írt rendszer volt, egy háttér szerver a Google tűzfalon belül, és az így hívogatta percenként vagy még gyakrabban magát ezt az App Engine-es alkalmazást és kérdezte, hogy van-e újabb feltöltött csomag, amit le kell tölteni. Ez az App Engine-nél ez most működik.

**Írtam egy kis demo alkalmazást, ami egy üzenőfal alkalmazás nyílt forráskódú.** Mit mutat be? Egyrészt az adattárnak a helyes kezelését, egyszerű bejelentkezést, hogy lehet egyszerűen cookie-kal bejelentkezni, template-ek használata. Látható, hogy van egy kis üzenő fal, ha utána elküldjük, akkor utána megjelenik, hogy köszönjük az üzenetet, akkor hamarosan megjelenik a felelet, visszaírunk és akkor hamarosan már ott is van. Ennyit tud ez a kis alkalmazás.

**App Engine-re a legkényelmesebb módon Eclipse-ben lehet fejleszteni.** Az Eclipse-nek van Python fejlesztő kerete, ezt hívják Pydevnek, ha ezt felrakjuk, az Eclipse-t és a Pydevet, letöltjük az App Engine-nek az SDK-ját, akkor utána az interneten rengeteg helyen le van írva, hogy hogyan lehet ezeket kényelmesen összelőni.

**Lássuk, hogy hogyan néz ki egy App Engine-re írt alkalmazás.** Itt van a forráskód könyvtár, akkor kezdjük talán a leíró fájlal. Ez a leíró fájl adja meg az alkalmazás nevét, hogy ez hányas verzió, milyen runtime-ot használ. Ugye jelenleg az egyetlen runtime-unk az a Python, reménykedünk, hogy később lesz más is. Ugye 1-es API verzió és utána meg kell adni gyakorlatilag a feldolgozó URL-eket, hogy melyik URL-t azt melyik modul fogja feldolgozni. Megadhatjuk,

hogy ezek adott sima könyvtárak illetve van unit test framework is, amit az App Engine-n keresztül tudunk kezelni, illetve van maga a főprogramunk.

Menjünk is be a főprogramba. Ha kinézetre akarjuk megnézni, akkor minden egyes oldalt, azt egy külön osztályként tudjuk definiálni és a különböző típusú webes lekérése, a get illetve post lekéréseket, azokat egy-egy külön metódusként le tudjuk írni. Van egy „Not found page”, ami akkor fog megjelenni, hogy ha nem talál semmi értelmes oldalt. Van a főoldal, akkor, ha itt elolvassuk ami ide van írva, gyakorlatilag ellenőrzi, hogy valaki be van-e jelentkezve, ha nincs, akkor átdob minket a loginra. Ha be van jelentkezve, akkor pedig megyünk tovább és megjelenítjük a falnak az aktuális állapotát. Ami trükközés itt megy, az az, hogy ez a fal lapozható legyen, tehát mindig csak az utolsó 10 darabot javítjuk meg és attól függően, hogy hány darab üzenet van, attól függően rakjuk ki a navigációs gombokat. És a végén rámegy egy ilyen render hívásra, ami átadja itt a változóban összerakott adatokat magának a view-nak, ugye ez a view egy HTML template. Aztán van a post page, ami pedig akkor hívódik meg, amikor elküldünk egy üzenetet, az ugye még egyszerűbb, itt csak el kell mentenünk magát az üzenetet.

Aztán van a login, amivel bejelentkezünk. A login az annyit csinál, hogy egy cookie-ban eltárolja azt, hogy milyen nevet választottunk magunknak. A logout pedig törli ezt a cookie-t és végül itt ez az URL-s tömbben rakjuk össze, hogy a főoldal az az alapértelmezett oldal és akkor az egyes oldalakat hova milyen URL-re kötjük rá.

Ezek után már csak az alkalmazásnak az indítása van hátra. Ez minden alkalmazásban ugyanúgy néz ki, ha akarjuk, meg van írva, hogy átválthatunk profile módra és nézhetjük, hogy milyen gyorsan futnak az egyes függvények. Ez a főprogram, ennyi az algoritmus mögötte és ami még érdekes benne, az maga az adattárnak a kezelése.

Egyrészt van a fal, definiáltuk a falat és a falban egyetlen egy attribútumot írunk le, hogy hány darab üzenet van benne és mi az utolsó üzenetnek a száma. Aztán van maga az üzenet, amiben pedig leírjuk, hogy ki a szerző, mi a címe, mi a törzse, melyik falhoz tartozik, hogy ha tovább bővítenénk és ez egy sokfalas üzenőfal rendszer lenne, az rögtön bele van tervezve, hogy több falat lehet definiálni, illetve, hogy hányadik üzenet az adott falon, az van még leírva.

A következő segédfüggvény az azt mutatja meg, hogy ha lekérdezzük egy falat úgymond az adattárból, ezt egyszer ezzel a hívással lehet, és a falnak a nevére, ha megfigyeltétek, nem adtam meg külön attribútumot, ugyanis a kulcsnak megadhatok saját értéket és a kulcsnak itt a fal nevét adom meg.

Ha megyünk tovább, akkor itt kezd érdekessé válni a dolog. Egészen konkrétan ez az increment index és a felette levő increment index tx-n, ez mutatja meg, hogy a tranzakciókezelés hogyan működik. Úgy tudunk egy tranzakcióban több műveletet végrehajtani, hogy ha ezt a db running transaction hívást használjuk és annak átadunk egy függvényt, ami azokat az utasításokat tartalmazza, amit a tranzakcióban akarunk végrehajtani. És itt gyakorlatilag annyi történik, hogy egy tranzakción belül lekérjük a falat, megnöveljük az értékét az utolsó üzenet számának és beállítjuk a memcache-ben, hogy amikor csak megnézzük, ne kelljen elmenni az adattárig és utána visszaadjuk. Ugyanez ugye fordítva, amikor az utolsó indexet a megnézésnél, amikor csak az utolsó üzenetnek a száma érdekel, vagy pedig először megpróbáljuk lekérdezni memcache-ben. Ha memcache-ben megvan, akkor nem megyünk el az adatbázisig, ha nincs, akkor pedig lekérdezzük, beállítjuk és visszakapjuk. Magának az üzenetnek a mentése egyszerű. Nem is kell egy darab tranzakciót használni, mert jobb, hogy ha rövidebbek a tranzakcióink és maga az üzenet nem feltétlenül lesz ugyanazon a szerveren eltárolva, mint a fal maga. Tehát csak egyszer lekérjük az új sorszámot és letároljuk magát az üzenetet. Ezen kívül még egy metódus, függvény van, amit használunk, ez pedig a lekérdezés. És itt látszik, hogy én nem az SQL-szerű szintaxist használtam, hanem ezt a kis lekérdező függvényhívásos megoldást.

Egy olyan lekérdezést használok, aminél az adott nevű fal, egy adott indexnél kisebb vagy egyenlő indexű üzeneteket adja vissza, mégpedig az index érték szerint csökkenő sorrendben. Ennyi az adatkezelés. Még egy dolog kell, hogy ez az egész működhessen, az pedig az index definíció, mi leírja, hogy az üzenettípusoknál milyen indexeink vannak. Ez talán látszik az autogenerated szóból, hogy ezt a fejlesztő eszközt, amikor futtatjuk itt a helyi gépen, ezt mindjárt meg is mutatom,

akkor automatikusan észreveszi, hogy milyen lekérdezéseket használunk és ezt az index.jano fájlt elkezdki kitöltögetni. Ezért a fejlesztés nem olyan kényelmetlen, hiszen csak beírom a kódba és ez az index.jano fájl automatikusan frissülni fog. Lehet olyan módot is használni, hogy ez ne így legyen. A template-eket bármilyen más eszközben hasonlóan kell, egy megadott template nyelv, az angolnak a template nyelvét kell használni, ebbe túl sok izgalom nincs.

## Czakó Krisztián: OpenVPN

Amiről ma beszélni szeretnék, az az OpenVPN, illetve a VPN-ek haszna úgy általában, és hogy mire jó az OpenVPN. Miről szól ez az egész? Bizalmasan kommunikálunk. Mi a célja a VPN-nek, mit várunk egy VPN rendszertől. OpenVPN esetén, ha gyorsan, egyszerűen akarunk csinálni valamit, akkor 3 lépésben hogyan húzhatunk fel egy működő VPN csatornát. Ha nem csak ilyen egyszerű dolgot akarunk, hanem komolyabban akarjuk, nagyobb biztonságot akarunk és nagyobb manager lehetőséget, azt hogyan megy. Azok számára, akik nem rendszergazdák, hogyan adhatjuk ezt oda a felhasználóknak, hogy ők is tudják használni anélkül, hogy úgy éreznék ez túl bonyolult számukra.

**Az első és legfontosabb, az a bizalmas kommunikáció.** A rejtjelezés meg a titkosítás igazából egy eléggé ősi mesterség, amit gyerekkorunkban mi is elkezdünk tanulni. Ki ne ismerné ezt a rövid, egyszerű kérdést és beszédstílust? Ez sem szólt másról gyerekkorunkban, minthogy úgy beszéljünk, hogy az nehezen legyen érthető és akkor jót szórakozunk rajta, aki még nem ismeri, mert még csak most jött az óvodából, nem érti. De nagyon hamar megtanuljuk és elég könnyű visszakódolni, tehát nem biztos, hogy ez a legjobb és legbiztonságosabb módja a kommunikációnak, de valahonnan innen indul az egész és az igény arra, hogy ez megtörténjen.

**Persze az informatikai rendszerekben számos különböző válfaja létezik annak, hogy a bizalmasságot és hitelességet megőrizzük,** illetve ellenőrizni tudjuk, hogy ki jön, honnan jön, mit mond, és tényleg ő mondja-e. Van néhány 3 betűs csodaszó, amit mindig emlegetnek a biztonsági szakemberek, 7 darab ilyen szó van, amiből van egy kakukktőzés, ha valaki megnézi, akkor gyorsan felismeri. SSL, TLS. Az SSL ismerős mindenkinek, alkalmazásszintű biztonságot fog nyújtani. Itt a kliensprogram, amivel kommunikálok és a szerveren futó alkalmazás egymás között biztosítja ezt a funkciót, de gyakorlatilag ugyanarról a funkcióról beszélünk. Mind a szerver, mind a kliens képes ellenőrizni azt, hogy tényleg azzal beszélget, akivel szeretne, és utána az adatokat titkosítja oly módon, hogy más lehetőleg ne férjen hozzá. Nyilván ennek megvan a megfelelő technikája, ezt lehet rosszul is csinálni, meg jól is csinálni. Mikor szól a böngészőnk, hogy a tanúsítvány nem megfelelő, folytatod-e? És nyugodtan az igenre kattintasz, na, onnantól kezdve a biztonság nem működik, odáig tartott a biztonság, míg a böngésző szólt, hogy ez nem biztonságos. És itt jön a hamis biztonságérzet, azt hiszed, hogy most biztonságos, mert a HTTPS az biztonságos, de nem az a HTTPS, ha nem győződtél meg arról, hogy tényleg azzal beszélsz, akivel akarsz. Attól kezdve egyáltalán nem biztos, hogy az az adat, amiről azt hiszed, hogy titkosítva megy át, tényleg nem fogja látni más, a man-in-the-middle támadás tipikus esete. Ne felejtjük el, hogy a világban már háborúkat buktak el azért, mert azt hitték, hogy biztonságos a kommunikáció, de mégsem volt az.

**A TLS az egy kicsi csavar ugyanezen, amikor az eredeti induló kommunikációhoz nem kell eleve a biztonság,** hanem elindulunk a nem biztonságos, megszokott csatornán és majd a túloldalán megbeszéljük, hogy folytatjuk biztonságosan. A végeredmény ugyanaz, csak az egyiknél az SSL-nél előtte itt utána. A VPN az, ami ezt egy alacsonyabb szintre viszi. Az előbbi szintnél egy alkalmazás szinten voltunk. Az alkalmazás szintnél az is egy fontos dolog, hogy aki használja azt a bizonyos szoftverkliens, legyen az egy web-böngésző vagy levelező, hogyan reagál a program által feltett kérdésekre. Ugye mondtam, hogy HTTPS-nél bemegyek az oldalra, közli, hogy nem jó a biztonsági tanúsítvány, jó a Firefoxnál öt kattintás legalább mire egy kivételt most már hozzá tudok adni, de régen megkérdezte, hogy akarom-e folytatni vagy sem? Rákattintottam, hogy igen, de jó, csak kapott egy nagy fenékkerűgást a biztonság, de még ma is a Firefoxszal néhány kattintással ki tudom kerülni.

**Ha ezt az egészet lejjebb viszem a VPN szintjére, akkor megkaphatom ugyanezt a biztonságot** úgy, hogy először: a felhasználónak nem lesz lehetősége kikerülni azt; másodsor: egyáltalán nem

kell foglalkoznia neki ezzel a résszel, mert alatta az operációs rendszer, el van intézve az egész hálózati szinten. A túloldali másik géppel való teljes kommunikáció biztonságos. Majd jönnek a kivételek is. A VPN lényege ez, hogy egy alacsonyabb szintre helyeztük ezt az egészet, nem egyes programok fognak dönteni a biztonságról, többnyire nem a felhasználó interakciói döntenek a biztonságról, hanem sokkal alacsonyabb réteg, tehát jobban fogjuk tudni kényszeríteni azt, hogy ez az egész biztonságosan működjön.

**Jön másik három érdekes rövidítés: DES, AES, stb.** Szokták az emberek keverni a dolgokat, hogy akkor most az SSL, TLS, a VPN az a titkosítás. NEM. Az nem titkosítás, az egy protokoll, ami lehetővé teszi a hitelesítést, az azonosítást és azt, hogy a két oldal megbeszélje egymással azt, hogy milyen titkosítási algoritmust fog használni. A titkosítási algoritmus a DES, a 3DES, az AES a Twofish, a Blowfish. Ezek olyan titkosítási algoritmusok, aminek megvan a matematikai alapja arra, hogy jelenlegi tudásunk szerint vagy nem visszafejthetők, vagy olyan erőforrással fejthetők vissza, ahol nem éri meg feltétlenül, illetve ne felejtjük el, hogy ezek az algoritmusok nagyon sok esetben tudottan visszafejthetők. A visszafejtésre olyan erőforrásra van igény, ami adott esetben rengeteg pénz vagy rengeteg idő. Kisebb pénzbefektetéssel tovább tart, nagyobb befektetéssel, több géppel kevesebb idő. Tehát arra figyeljünk oda, hogy ha így titkosítottan viszünk át adatokat, azt tudjuk, hogy x idő múlva vissza tudják majd fejteni.

**Ne higgyük azt, hogy most ezt biztonságosan átküldtük és senki nem fogja látni,** lehet, hogy csak 5 év múlva fogja tudni megnézni azt, hogy mit kommunikáltunk most, de ha 5 év múlva ugyanaz a jelszavunk, ami akkor átment a hálózaton, akkor azt biztosan fogja tudni. De ugyanez, hogy ha bizalmas adatokat viszek át, a cégnek a bizalmas adatait, 5 év múlva hozzá fognak jutni. Ezt ne felejtjük el, akár HTTPS-sel megyünk, akár VPN-en megyünk. Nem biztos, hogy 5 év, lehet, hogy egy év, attól függ, hogy hány bites titkosítást és milyen algoritmust használtunk. Ezt tessék fejben tartani. Ne hidd azt, hogy soha nem fogja valaki ezt megnézni. Ezért kell változtatni a kódot. Általában ezek kétkulcsos rendszerrel dolgozó hitelesítési módok, ami után még egy session key megy, aminek megint nem mindegy, hogy hány bites és hogy azt mikor tudják visszafejteni. Az ugye egy egykulcsos rendszer, a kétkulcsosnak még túl nagy az erőforrás igénye ahhoz, hogy folyamatos kommunikációra alkalmas legyen. Az stb. a kakukktojás a többi a rövidítés.

**Honnan is jött a VPN?** Jött az ipsec nevezetű fejlesztés, ami az IPv6-os rendszernek egy alapértelmezett tartozéka, aminek az volt a célja, hogy minden gép a hálózatban egymással, kliensek, szerverek, mindenki, aki kommunikál egymással, az biztonságosan tegye ezt. Felismerték azért az IPv4-nél, hogy ha minden kódolatlanul megy, az nem egy szerencsés dolog. Ez azt feltételezné, hogy minden számítógépnek megvan a maga tanúsítványa, és van egy olyan központi tanúsítványkiadó, mint a weboldalaknál vannak tanúsítványkiadók, nem egy központi, sok, akit tárol az én gépem. Eredetileg az operációs rendszerrel együtt jönnek azok a tanúsítványok, amiket hitelesnek tartunk. Ilyen módon hitelesítjük, de nem a böngészőt és a szerveret, hanem egymás között a gépeket és mindenki mindenkivel biztonságosan fog kommunikálni, nem lesz kódolatlan forgalom. Az IPv6 manapság még mindig nem tartozik a nagyon elterjedt dolgok közé. Közben rájöttek az emberek, hogy milyen jó ez a dolog, hogy titkosítottan, biztonságosan lehet kommunikálni. Átültették a gyakorlatba az IPv4-re portolva, csakhogy nem egészen azzal a céllal, amire eredetileg készült. Hanem arra, hogy távoli rendszereket elérjek, ott konkrétan, adott esetben a vállalati, intézményi rendszert elérjem biztonságosan. Gyakorlatilag a világ azt mondta, hogy nekem a VPN ezért kell, és nem azért, hogy mindenki mindenkivel, és nem foglalkozunk vele, így mind a mai napig nem terjedt el. Gyakorlatilag nem létezik az internet szintű általános biztonság. Így a gyakorlatban a VPN esetében kialakultak a host-host, host-net, net-net VPN megoldások, ugye a host-host, amikor két gép kommunikál egymással és ez lett volna az eredeti cél, hogy bármikor, amikor két gép kommunikál egymással, akkor hitelesítik egymást és így kommunikálnak. Ehelyett jött a host-net, amikor én a saját kis notebookommal vagy otthoni gépemmel bemegyek a céges VPN-be, vagy a net-net, amikor a cég két telephelyét VPN-en összekötöm. De ezenkívül már

nincs titkosítás. És itt jön a probléma, hogy oké, hogy én a notebookommal a tűzfalig vagy a VPN gateway-ig titkosítottan kommunikálok, onnan a helyi hálózaton megy tovább kódolatlanul az információ. És itt jön, hogy mégse dobhatjuk ki azt, amikor a böngésző HTTPS-en, vagy a levelező TLS-en keresztül megy.

**Mit is várunk a VPN-től?** Egyrészt azt, hogy elérjünk távoli hálózatokat biztonságosan, másrészt azt, hogy ezt egyszerűen és könnyen tudjuk használni. A felhasználó nem fog olyat használni, ami neki macerás. És itt jön az egyik, hogy nagyon szép, hogy a böngésző ott van a HTTPS, de az emberek nem értik. Hiába tudja azt az informatikai szakember, hogy ha azt mondja az a böngésző, hogy ez a tanúsítvány nem arra a névre szól, akivel kommunikálok, leokézza csak menjen már tovább. Nagyon jó, adathalász rendszereknél ugyanúgy okézzák le, amikor azt hiszi, hogy a bank weboldalára megy be, ugyan a böngésző szólt neki, hogy a tanúsítvány nem stimmel, mert nem hiteles tanúsítványkiadó adta ki. Megnéz, hát ugyanarra a névre szól, hiába szól a böngésző, hogy nem is ezzel a névvel kommunikálunk, nem baj, megyünk tovább. Ezt nem nevezem az egyszerű felhasználhatóságnak, ettől kezdve a biztonság, csak egy álca, egy látszat, arra jó, hogy a bank mondta, hogy ő nem tehet róla, hogy a felhasználó hülye volt. A felhasználó nem érti, nem tudja ezeket a dolgokat.

A felhasználónak az kell, hogy könnyen és egyszerűen tudjon dolgozni és egy VPN esetében ugyanez. Nem fogja érdekelni az egész akkor, ha neki öt percet kell vacakolni vele, hogy elindítsa, akkor fogja tudni ezt használni, ha neki a számítógépe vagy magától csinálja és ő nem is tud erről az egészről semmit és akkor az a számítógép, notebook mindig megy be a vállalati VPN-be és a felhasználónak nincs is joga ezt módosítani, ez az ideálisabb változat egyébként.

A kevésbé ideális, de inkább gyakorlati változat nyilván azért vannak úgymond okosabb felhasználóink, aki otthon is akarja használni, nem akarja, hogy a VPN-be menjen, ő majd eldönti, hogy mikor akar a VPN-be menni. Ezzel már csinált egy átjárót a külvilág és a belső hálózat között, amit védjen ki, aki bír, tűzfalon vagy bárhogy, hiszen néha a nem biztonságos hálózaton kommunikál, nem tudni, mit szed ott össze, néha meg bejön a vállalati hálózatba, ki tudja, mit nyom ott be. Vagy fordítva: mit szed ki onnét, amit majd valaki egy trójai programon keresztül a másik irányba leszed a géperől.

Mindegy, akkor is egyszerű kell. Egyszerű, ha egy vagy két kattintással a VPN-t fel tudja építeni, ha egyáltalán kattintania kell hozzá. Menedzsment oldalon is viszonylag egyszerűség kell, kevés az olyan képzett rendszergazda vagy általában drága és nem tudják, vagy nem akarják megfizetni, aki vi-jal meg akármivel VPN szerveret fog adminisztrálni. Főleg a felhasználói adminisztrációra gondolok itt, nem a kialakítás üzemeltetési részére, illetve az üzemeltetés részére, hogy a felhasználónak jogot adunk, elveszünk, jelszót beállítunk. Ehhez nyilván értelmes, egyszerűen kezelhető menedzsment kell. Illetve a megfelelő biztonság, olyan VPN megoldást kell tudnunk alkalmazni, amely megfelelően garantálja a biztonságot, tehát be lehet állítani, hogy megfelelő minőségű hitelesítési algoritmusokat, kulcsokat használjon. Nem hátrány, ha brute force betörés elleni védelmet ad a szerver, tehát ha felhasználónév-jelszóval azonosít a VPN-gateway és lehet próbálkozni a végtelenségig, akkor az se szerencsés.

**A világban van néhány VPN megoldás,** az említett ipsec, ami elvileg a standardnak szánt megoldás. Nagyon jó, majdnem olyan jó standard, mint az openh323, ha ez mond valakinek valamit. Olyan standard, amit mindenki úgy tudott implementálni, hogy a különböző implementációk nem igazán együttműködők egymással. Hiába tud ipsec-et az én Linuxom, meg hiába ipsec-et valamelyik gyártónak az ipsec rendszere, azt ha egyáltalán egymással kommunikációra bírom venni, akkor is vért izzadok hozzá. Bár ma már egész jól állunk, pár évvel ezelőtt ez tragédia volt. Ez az egyik része, a kompatibilitási gond.

**A másik része az, hogy ahány gyártó, annyi ipsec és plusz extra megvalósítás.** Ugyanis az ipsec az arra szolgál, hogy két gép hitelesítse egymást és utána biztonságosan kommunikáljon egymással.

Nem arra, hogy egy távoli hálózatot elérjek ezen keresztül. Ma már implementálták az ipsec-ben a host-net tunnelinget, de eredetileg nem volt.

Jöttek gyártók, például a Microsoft és az L2TP implementálása ipsec alatt. Ez layer2 tunneling protokoll az L2TP, Linux alatt is szépen működik. A célja az, hogy kommunikáljak egy másik géppel egy tunnelen keresztül, és ott egy egész hálózatot el tudjak érni. Ezt a Microsoft például úgy hozta össze, hogy fölépít a windowsos gép a Microsoft VPN szerverével egy ipsec kapcsolatot, majd az ipsec-en belül titkosítva egy L2TP-t, ami nem autentikált, nem biztonságos, de mivel fölötte van egy ipsec, és így éred el a belső hálózatot, mert maga az ipsec-et nem implementálták úgy, hogy azon keresztül menjen, mert amikor implementálták, nem tudott ilyet az ipsec. Bár szerintem hivatalosan még ma sem szabvány ez a része. Tehát, ha azt akartam, hogy a Windows munkaállomásba beépített gyári VPN megoldás kommunikáljon egy linuxos VPN gateway-en keresztül, akkor kellett egy ipsec-et raknom rá, egy L2TP-t és ezen keresztül működött. De ha Linuxsal akartam bemenni, akkor a kliensen nem kis fájdalom volt az, hogy a kliens L2TP-n akarjon bemenni. Ahány gyártó, annyi megoldást kell implementálni a gateway-en, hogy működjön, felejtjük el.

Ugyanez mondjuk a Cisco VPN megoldása, ami az ipsec-ből még extrákat rakhat be, jó, Linuxra ott van a Cisco-nak a VPN kliense, tehát gyakorlatilag ott is egyszerű, meg Windowsra is megvan a Cisco-nak a VPN kliense, akkor legalább egységes az egész. Megint meg van erőszakolva egy protokoll.

**Még egy probléma van az ipsec-kel és megint az eredeti tervezésre utalnék vissza: az a célja, hogy host-host kapcsolatot valósítson meg, ráadásul IPv6-on.** Az IPv6-nak van egy nagyon fontos jellemzője, annyi IP cím van benne, hogy szerintem a földön minden molekulának jut egy IP cím, de legalábbis a hűtőszekrénynek jut egy pár ezres címtartomány, hogy minden ételt külön tudjon azonosítani. Ettől kezdve az ipsec szabvány szerint azt fogja csinálni, hogy hitelesíti az IP címet, hogy ki vagyok cím szerint meg a másikat. IPv4-en mi a problémánk ezzel szemben? Kicsit kevés az IP cím. Na, most mit csinálunk? Címet fordítunk különböző helyeken, privát tartományok. De mit csinál az ipsec? Hitelesíti az IP címet. Mit hitelesít? Magát az IP csomagot, minden tartalommal fejlődéstül. Na most ha én elindítok privát címmel egy csomagot hitelesítve, majd jön a router, az kicseréli az IP címet benne egy másikra, címet fordít magyarul, és eljut a címzetthez a csomag, a címzett megnézi és nem lesz jó a digitális aláírás, hiszen már más a forrás IP címe a csomagnak. Ennyit arról, hogy az ipsec-et címfordítással lehet használni, nem lehet.

Az emberek kitalálták, hogy lehessen, az ipsec ugye IP szintű protokollként, ESP (Encapsulating Security Payload) nevű dologba megy az adat. A hitelesítést az AH, de ez általában egy csomagba megy. Mert eredeti implementációs ötlet szerint az ESP és az AH az külön IP protokoll, de az AH-t már az ESP csomaggal együtt szoktuk küldeni, és ESP-be megy az egész. ESP csomagot küldünk csak, nem használunk AH csomagokat, a mai implementációk sehol. Az ESP-t és az AH-t ugyanarra a szintre kell tenni, mint az TCP-t vagy az UDP-t. Nem egy olyan protokoll az ipsec, hogy TCP-n vagy UDP-n vagy ICMP-n megvalósított adatforgalom, mint az alkalmazások csinálnak, de ugyanazon a szinten megy. Az ESP csomag meg a TCP csomag ugyanaz a szint, és még ebbe jönnek bele a dolgok.

**A lényeg, hogy a címfordítás az egy problémás dolog, ezért kitalálták, hogy felejtjük el az eredeti implementációt, tegyük át UDP-re.** Akkor attól függ, hogy van-e címfordítás vagy nem, vagy bekapcsolom a kliensem, hogy az ESP-vel fog vagy UDP-n, de akkor az ESP csomagokat becsomagolja UDP csomagokba és úgy küldi át, meg az AH is (Authentication Head). De kell még hozzá az IKE (Internet Key Exchange) is, nyilván az a kulcs csere mechanizmus.

Megemlítem a Windowsosok által oly kedvelt PPTP-t, amit VPN-nek csúfolnak, ezt inkább felejtjük el, a biztonsági színvonal nem hozza azt amit egy VPN-től elvárunk, bár tény, hogy az egyszerűsége megvan, legalábbis, ha Windowsból indítja a kommunikációt a kliens oldalon.

**Az OpenVPN egyszerű és biztonságos.** Nyílt forrású program, minden operációs rendszeren ugyanazt a programot fogom tudni használni, tehát nincs olyan problémám, hogy különböző op. rendszereket, különbözőképpen kellene kiszolgálnom vele. Nyílt forrású jól ellenőrizhető kód stb. Telepíteni, beállítani rendkívül egyszerű, grafikus menedzsment faxokat is találni hozzá, de egy vi-jal sem kell öt sornál többet írni egy átlagos OpenVPN konfigurációba. A szokásos X.509-es rendszerrel dolgozik ez is, az ipsec is tudja egyébként, az OpenVPN-nek ez természetes része. Ha ezzel dolgozunk például, ha windowsos munkaállomásokat akarunk bekötni, találunk egy jó kis MyCert windowsos binárist, meg gyorsan és egyszerűen lehet kulcsot és tanúsítványt szerezni hozzá. Az OpenVPN számos titkosítási algoritmust is implementál, gyakorlatilag az összes elterjedtet DES, AES, Blowfish és TwoFish-t is, szóval szabadon válogathatunk, alapesetben meg is beszél a túloldallal, hogy milyen kommunikáció legyen, de kényszeríthetjük is a megfelelőt.

A cím fordítható, sőt még proxy-zható is az OpenVPN, korlátozásokkal persze. UDP-t használ alaphelyzetben, tehát ő az IP csomagjait UDP-be fogja csomagolni eleve, nem használ semmiféle szabványos dolgot, tehát azt ne felejtjük el, hogy az OpenVPN nem szabványos. Az OpenVPN csak OpenVPN-nel fog kommunikálni, de azzal tökéletesen. Tud TCP-felett is kommunikálni, nyilván egy kicsit lassabb és nehezebb, mert a TCP nem erre való, az UDP alkalmasabb erre a feladatra. De ha már TCP-n, akkor egy átlagos proxy-n, ami megengedi connect metódussal történő közvetlen TCP kapcsolat létrehozását, át fog menni. Nyilván, ha TCP-n használom akár 443-as, tehát a HTTPS portjára is tehetem, tehát ha a hálózat korlátozása csak annyiból áll, hogy egy Squid típusú proxy-n át kell menni, azon át fog tudni jutni. A Squid alapértelmezetten tilt minden mást connect metódussal, mint a 443-as portot, de ha odatesztek egy OpenVPN-t menni fog. Tehát egész sok mindenben át tudunk gyalogolni vele, mindaddig, amíg nincs applikációs szűrés egy tűzfalon, addig OpenVPN-nel nagy eséllyel át fogunk tudni jutni. Mi kell ahhoz, hogy egy OpenVPN elinduljon? Nagyjából 3 lépés. Itt egyszerű osztott kulccsal dolgozunk most, majd jön később az, ha nem így. Debianból indultam ki, mert az nekem jobban kézre esik, de Ubuntu is hasonló, minden deb csomag alapú disztribúciónál az „apt-get install openvpn” után nagyjából az „openvpn -genkey -secret <kulcsfájl>” generál nekünk egy kulcsot, ezt a kulcsot nyilván át kell másoljuk valami megbízható módon a másik gépre is, hiszen osztott kulcsról van szó. Az /etc/openvpn könyvtárban csinálunk egy config fájlt, megadom a saját IP címemet, a másik oldalnak az IP címét. Az OpenVPN alapértelmezésben az 1194-es portot használja UDP-vel. Az ifconfig mondja meg az OpenVPN-nek, ez nem Linux parancs az ifconfig OpenVPN opciója, hogy mi az én IP címem, mi a másiké, meg mi a secret, az osztott kulcsunk. A konfiguráció különbség a másik gépen annyi lesz, hogy az ifconfigban a két cím fordítva lesz. Ezek után, a három lépés után működő OpenVPN kapcsolatunk van. Pontosabban van még egy negyedik lépés, ami innen kimaradt, el is kell indítani az OpenVPN-t, /etc/init.d/openvpn start. Ezzel egy osztott kulcsos VPN már működik. A biztonsága annyi, hogy osztott kulcs, tehát az annak megfelelő szintű biztonságot adja, de titkosításban meg fogja hozni nekünk azt, amit bármelyik más OpenVPN kapcsolat tud. Ez kifejezetten akkor jó, ha két olyan gépet kötök össze, amihez mind a kettőhöz biztonságosan hozzá tudok férni és egy stabil tunnelt akarok csinálni a kettő között. Gyakorlatilag ugyanaz a kategória, mint egy ip-ip tunnel, csak biztonságosabban. Hiszen ott nincs titkosítás.

**Még egy dolog, az OpenVPN az nem csak tunnel módot, hanem bridge módot is tud.** Tehát tunnel módnál a defaultja azt fogja csinálni, hogy mind a két oldalon lesz egy IP címem, a Linux esetén tun eszköz fog létrejönni, mondjuk tun0, ha adott esetben egy darab VPN csatornám van. Ennek lesz egy IP címe, attól kezdve a Linux belső routingja fogja elintézni, hogy merre menjenek a csomagok. Tehát adott esetben egy ilyen tunnelnél is, ha át route-olok másik hálózatot, abba az irányba fog menni a forgalom.

A másik a tap interfész használata Linux esetében. Ez egy kvázi ethernet-szerű interfész lesz, nem kell, hogy IP címe legyen, de lehet. Beletehetem egy bridge-be, ha össze bridge-dzselem, mondjuk, fogok két routert OpenVPN-en egy osztott kulccsal összekötöm őket úgy, hogy tap interfésszel dolgozzak, a tap interfészt meg a belső hálózati interfészt bridge-be teszem, úgy kötöttem össze a két hálózatot az interneten keresztül, hogy azon a címtartományon összebridzseltem a két hálózatot

teljesen transzparens módon. Nem ez a legjobb megvalósítási példa erre, de nagyon jól lehet ilyesmikre használni.

**X.509**, amikor beírom a böngészőbe HTTPS oldalt és ellenőrzi a böngésző, hogy a tanúsítvány tényleg azé, akivel beszélgetek, az a név, a tanúsítványkiadó stimmel-e? Erről szól az X.509. Gyakorlatilag van egy tanúsítványkiadó, nyilván adott esetben egy cégnél egy al-tanúsítványkiadó, ami kifejezetten VPN kapcsolatokra ad ki tanúsítványokat. Ezzel intézem el. A tanúsítvány le tud járni, adott időre adom ki a tanúsítványt, amit néha meg kell, hogy újítson, a tanúsítványt vissza tudom vonni. Tehát központilag menedzselhető az, hogy ki az, aki egyáltalán bejöhét. Ha az adott alkalmazottat ki akarom zárni, pár kattintás, visszavonom a tanúsítványát, a VPN gateway nem engedi be többet.

**Az OpenVPN csomagban van egy easy-rsa kis szkriptkészlet, amivel ha nincs lehetőségünk arra, hogy korrekt, komplett tanúsítványkiadó rendszert implementáljunk a cégnél.** Lássuk be egy 5-10 fős kis vállalkozásnál, ennek az erőforrás és költségigénye túl nagy általában ahhoz, hogy érdemben implementálni lehessen. Akkor, ha más nem, valamelyik szerverükön, lehetőség szerint azért egy titkosított könyvtárba, amihez csak korlátozottan lehet hozzáférni, felrakjuk, egyszerű szkriptekkel regenerálhatjuk könnyedén a tanúsítványkiadót, tanúsítványokat is, sőt kulcsokat is, bár az szerencsés eset, ha a kulcs a kliensen generálódik, és onnan nem távozik el. Aztán persze lehet bonyolítani azzal, hogy smartcard stb, de ez nem az iskola szintje, ahol ezt meg lehet valósítani.

Tud ezen kívül előazonosítást osztott kulccsal. Osztott kulccsal egyszerűen, gyorsan VPN-t felépítettünk, de mi van akkor, ha az X.509-et és az osztott kulcsot kombináljuk? Ugyanazt az osztott kulcsot szétesztjük sok embernek, de minek teszünk ilyet, mert ha már többeknél van, akkor hol bizalmas meg titkos? Ennek az a poénja, hogy így is sokkal kevesebb helyen lesz osztott kulcs, mintha nem lenne egyáltalán osztott kulcs. Ez arra jó, hogy a vakvilágból jövő robotizált automatikus támadások ott bukna el, hogy már nincs osztott kulcsuk. Nem jut el egyáltalán az X.509-es tanúsítvány-ellenőrzésig. Nem tudom floodolni az X.509-es tanúsítvány ellenőrzését mindaddig, amíg az osztott kulcshoz hozzá nem jutottam. Az életemet erre nem bíznám, de arra nagyon jó, hogy tipikus DOS támadások ellen megvédi az OpenVPN gateway-t. Az osztott kulcs, ha normális felhasználóink vannak, nem fog az interneten terjedni. Tehát egy minimális szűrést már ad. És utána jön a normális autentikáció X.509-cel, ha ezen túltette magát a kliens. A harmadik variáció a plug-in-ek. Az OpenVPN-hez lehet plug-ineket írni, egy van, egy van, ami, legalábbis a debianos disztribúcióban gyárilag benne van, az egy PAM kiegészítő, ettől kezdve bármilyen PAM autentikációs rendszeren autentikálni tud a felhasználó. OpenVPN protokollban implementálva van a felhasználónév-jelszó alapú autentikáció is, használhatjuk ezt. Ezeket kombinálhatjuk szabadon, csinálhatom azt, hogy csak felhasználónév-jelszóval autentikálom a usert, nincs tanúsítvány-ellenőrzés, se osztott kulcs. Vagy osztott kulcs és felhasználónév-jelszó autentikáció, vagy azt, hogy X.509 után jön egy felhasználó autentikáció vagy mind a hármat, osztott kulcs, X.509 és felhasználó azonosító.

A tanúsítványkezelés teszi picit macerásabbá, de azt nem hagynám ki a dologból. Ha éves lejárátú tanúsítvánnyal dolgozom, akkor azt évente egyszer meg kell, hogy újítsa a delikvens. Ennyi a macera vele, az osztott kulcsot egyszer kellett, hogy megkapja, a felhasználónév-jelszó a legmacerásabb, mert azt szeretik elfelejteni vagy primitív jelszavakat adni a dologhoz. Nyilván, ha felhasználónév-jelszó autentikáció van, akkor akár központi autentikációs rendszere, bármire például LDAP-ra, de lehet RADIUS-on megy mind, amihez PAM modul van, arra autentikálni fog tudni az OpenVPN a PAM-on keresztül.

Az előbb az osztott kulcsnál azt mutattam, hogy host-host kommunikáció van, két pont kommunikációja egymással és látszhatott az is a konfigon, hogy ez többet nem tud. Van egy úgynevezett szerver módja az OpenVPN-nek. Ez esetben minden egyes kliensnek dinamikusan ad IP címet, dinamikusan adja oda a hálózati paramétereket, gyakorlatilag, mint egy DHCP jellegű információadás történik, de nem DHCP, nem kell mögé DHCP szerver, az OpenVPN-en

bekonfiguráltam, hogy mit adjon át, ebbe átadhatok gyakorlatilag IP címet, bármit át tudok adni, amit a DHCP, ugyanazokkal az opciókkal dolgozik. Ugyanazokat küldi le például a windowsos implementáció a Windowsnak úgy emulálja, mintha DHCP-n keresztül kapta volna az információkat és így adja át a hálózati résznek az információkat. Szerver módban több kliens tud jönni, mindenki adott esetben felhasználónév-jelszó-val, tanúsítvánnyal stb. autentikálja magát. Ugyanarra ez egy OpenVPN processzre több kliens rá tud kapcsolódni és ezen keresztül elérni a hálózatot.

Itt nyilván a dinamikus kiosztás mellett nagyon egyszerű módszerrel tudunk fix információkat is átadni az adott klientsztől függően. Ha X.509 van, akkor a tanúsítvány common név alapján, aminek egyedinek kell lennie definíció szerint, tudunk egyedi konfigurát írti illetve a felhasználónév-jelszavas autentikációnál mondhatjuk, hogy tekintse common névnek a felhasználónevet, ami megint csak technikailag egyedi lesz és ennek megfelelően tudunk egyedi konfigurát adni az adott kliensnek, tudunk neki IP címet adni, vagy egyéb egyedi hálózati beállításokat ez alapján. Ez esetben gyakorlatilag egy gateway-en összpontosul mindenkinek mindenki, és itt beállítható, hogy ez a mindenki tud-e egymással kommunikálni vagy sem. Mondhatom azt, hogy a kliensek, akik bejönnek a VPN központba, azok lássák egymást vagy ne. És mivel tudok egyedileg kliens szinten konfigurát adni, így kliensekként is megmondhatom, hogy ki igen, ki nem.

**Még egy fontos és hasznos dolog, mivel egy userspace-ben futó dolgról van szó, tehát nem kernel szintű implementáció az OpenVPN,** ellenben az ipsec-vel, ami Linux alatt kernel szinten van implementálva. Kedvemre futtathatok több OpenVPN processzt, egyetlen dolog, hogy ugyanazon a porton nem figyelhetnek. A default 1194-en vagy 1195-ön vagy 20500, ahová kedvem van rakni, egyiket TCP-re, másikat UDP-re. Tehát ugyanazon a gateway-en több különböző, teljesen más konfigurátal működő, más célt szolgáló VPN-t tudok csinálni.

Tipikus példa, amikor egy cégnek van két vagy több telephelye, a két telephelyet összekötöm, egyik OpenVPN-nel egy egyszerű net-net összekötéssel, ahol route-olom a két netet egymással, és adott esetben mindkét helyen lehet egy olyan OpenVPN processz, ami arra jó, hogy a felhasználók be tudjanak lépni, akár mindkét helyre ugyanazzal a névvel-jelszóval. Mondjuk egy központi LDAP autentikáció van, a központban van a címtár központja, a másik helyen, a másik irodában egy replika, bárhova ugyanazzal a névvel-jelszóval be tudok lépni, csak azt választom ki, hogy melyik OpenVPN gateway-t használom.

Az OpenVPN, mint minden rendes VPN a leszakadó kapcsolatot automatikusan megpróbálja – ha úgy konfigurálom – újraindítani, újraépíteni. Különböző ellenőrzések vannak beépítve, tehát ilyen ping ellenőrzések, ahol próbálja pingelni a túldalt az OpenVPN a saját csatornáján belül, ha erre nem kap választ, újraépíti a kapcsolatot. Az ipsec-es megoldásokhoz képest, legalábbis Linuxszal és Openswan vagy Freeswan régebben az volt a tapasztalatom, hogy az Openswan, FreeS/WAN nem annyira szereti minden esetben újraépíteni a leszakadt kapcsolatot, különösen akkor, ha dinamikus IP címen van valamelyik oldal. Nem igazán van erre tervezve, mert az macerás. Az OpenVPN-nel ezzel szemben azt tapasztaltam, hogy hiánytalanul, szinte az estek 99,9%-ban megteszi ezt, ha egyszer leszakadt, akármilyen IP címről indul újra a kliens, tehát ilyen kliens-szerver megoldásban, akkor újraépíti a kapcsolatot. Tehát meglehetősen stabil. Tipikusan, ha ADSL-en lóg egy gép és ott az ADSL leszakad és újraépül, mert szoktak a szolgáltatók ilyet csinálni, az OpenVPN-en is azonnal újraépíti a kapcsolatát, mint kliens. Szervernél ott macerásabb, mert neki ott bind-olnia kellett vagy listen-elnie kellett a megfelelő porton. Ha ppoe-t csinállok, akkor az if-up/if-down szkriptekkel illik leállítani, újraindítani az OpenVPN-t ahhoz, hogy a szerver is ezt vegye.

**Grafikus felület, felhasználók.** Linuxnál nagyon egyszerű dolgunk van, a legtöbb Linux disztribúció közismert, hírhedt network manager-rel dolgozik, mármint hálózati konfigurációs alrendszerrel, ez azért jó, mert már lassan eléri az alfa állapotot. Pár évet várunk, szerintem bétát is kiadnak belőle, mindenesetre ma mindenki ezt használja. Nagyjából ezzel minősítettem a

megbízhatóságát, nagyjából kezdi hozni a Windowsok színvonalát a konfigurálhatóság, lehet klikk-klikk, csak nem mindig működik.

Ennek van egy OpenVPN plugin-ja, ennek a network manager-nek, ami nagyon szépen kezeli az OpenVPN tunneleket ilyen felhasználói szinten. Ez azért jó, mert gyönyörűen kihagyja az implementációból a leszakadt csatorna újjáépítését. Tehát az előbb elmondtam, hogy milyen jól újraépíti az OpenVPN-t, ha leszakadt, na, nem ekkor. De legalább felhasználóbarát. Ha nem akarom, hogy a felhasználó foglalkozzon, ugye ideális esetben az OpenVPN fut a háttérben és a felhasználónak nincs módja ezt manipulálni, ő bárhol felmegy a netre, bármikor, a rendszere nem kommunikál a helyi hálóval, kizárólag az OpenVPN-t enged ki, ezt megfelelő tűzfal policy-vé és egyéb biztonsági policy-vel előírjuk. Akár Linuxon, akár Windowson ez előírható és automatikusan megy minden, automatikusan fog menni az újracsatlakozás is, fut az OpenVPN processzor a háttérben, szuper.

Hát itt a network managernél nem annyira, a network manager észleli, ha az OpenVPN leszakadt, akkor az OpenVPN-t leállítja és vár arra, hogy a felhasználó újra kattintson, hogy az OpenVPN-t indítsuk újra. Cserébe viszont megadja azt a lehetőséget, hogy katt, lista, VPN kapcsolataim, – na, hova megyek –, de ez nem tipikus.

Van az OpenVPN-nek egy GUI változata, ez az <http://openvpn.se>-ről letölthető, egyébként az OpenVPN a <http://openvpn.org>-ról tölthető le az eredeti, ott is van windowsos változat, de az a szerver, itt van egy GUI, ami a tálcára lehelyezi a megfelelő kis ikont, klikk-klikk és fel lehet építeni a kapcsolatot. Az a különbség a linuxos megvalósítás és a network manager-es linuxos pluginnal szemben, hogy míg az előző a network manager grafikus részéhez is adja a plug-int, tehát ott a teljes OpenVPN-t, klikk-klikk wizardos módszerrel tudom konfigurálni, addig a Windowson a konfigurációt kell kézzel írni. A Windows még nem annyira felhasználóbarát, mint a Linux.

Ellenben, és itt a Windowsosok vannak előnyben picit mint felhasználók, ez a myset.exe. Ez egy nagyon aranyos kis program, aminek elsődleges célja az, hogy X.509-es kulcsot és tanúsítványkérelmet generáljon nekem és melleleg az OpenVPN-hez írták, képes template-ből OpenVPN konfigurációt gyártani. Mit csináljak? Fogom a myset.exe-t odateszem mellé a CA-nak a tanúsítványát, ami kell a kliensen is, odateszem a template-t, ezt odaadom a felhasználónak. A template-ben azt is leírom, hogy a tanúsítványkérelemben milyen mező, hogyan legyen kitöltve, mi az, amit kitölthet a felhasználó, mi az, amit nem. Ezt zip-ben odaadom neki, ebben semmilyen bizalmas adat különösebben nincs, de nem árt odafigyelni arra, hogy az ember küldött CA-t, akkor az tényleg az a CA lesz az ő gépén, amivel ellenőriznie kell a kliensnek és nem valaki másét, tehát itt is jó vigyázni, de ott az egy fingerprinter ellenőrzéssel ellenőrizhető lesz. Ezt elindítja, két kattintás, feljön, kitölti a nevét, mondjuk, ha csak ennyit engedélyeztem, mert a szervezet neve, ország stb., beírtam fixen, azt nem tudja átírni, beírja a nevét, kulcsgenerálás, a kulcsot, a konfigurációt leteszi a megfelelő C:\Program Files\OpenVPN\config könyvtárba, kiadja a képernyőre a tanúsítványkérelmet, oké, ezt e-mail-ben beküldi, e-mail-ben kapja a választ, a tanúsítványt. Utána illik fingerprint ellenőrzést elvégezni, hogy tényleg azt kapta-e, amit mi küldtünk, ha nem volt biztonságos az átviteli csatorna. Bemásolja a megfelelő fájlt, amit ő visszakapott, és máris egy kattintással a tálcájáról tudja indítani az OpenVPN csatornáját.

Van még egy pár GUI OpenVPN-hez, olyan is, amelyik szerver oldalon menedzseli, tehát tudom nézni, hogy kik vannak rákapcsolódva, tudom módosítani futás közben a konfigurációt, illetve kikapcsolni, kliens visszaengedni. Meg kliens oldalon is van olyan GUI, amivel lehet menedzselni, nemcsak a network manager Linux alatt. Ebből lehet válogatni.

A Windows szerver az ipsec-et kattintásokkal implementálja és a Windows kliensből pillanatok alatt be lehet menni és az oktatási intézményeknek tudom, hogy ez ingyen van, vagyis az adónkból van. Talán, ha szabad szoftvert használunk, akkor kicsit kevesebb lesz az, amit a Microsoft ki tud szedni az állam zsebéből. Nem állítom, hogy a Windowst nem lehet használni, csak fölösleges pénzt kiadni olyanra, amiből van jobb ráadásul. Az open source-ról még sokat lehetne mesélni.

**Hallgató:** távmenedzsmentre OpenVPN vagy SSH?

**Előadó:** szándékosan hallgattam róla. Az SSH alkalmazás szint. Az SSH arra, hogy belépjél távoli gépekre, teljesen jó. Apró problémák szoktak lenni, ha open SSH-t használsz, a biztonsággal. Az SSH egyes protokolljai, mint protokoll, tervezési hibás. Én azt mondanám, hogy OpenVPN és SSH.

**Hallgató:** az SSH tud layer2 tunneling-et. Az OpenVPN és SSH-nál arra kell odafigyelni, hogy ugyanazt a protokollt ne használd egymásba ágyazva, mert gyengítheti egymást. Ha linuxos szerveret üzemeltetsz, ami mondjuk a VPN koncentrátor és a kliensek windowsosok, akkor nagyon kényelmes, hogy nem kell kattogatni a beállításhoz. A klienst beállítod egy konfigurációs fájl, a tanúsítvány, a privát kulcs meg ilyenek, ezt az egészet így odaadod pendrive-on a felhasználónak, azt mondd, hogy telepítsd fel ezt a GUI-t, az összes fájlt hányd be ebbe a könyvtárba, indítsd el, menni fog.

**Előadó:** teljesen igazad van. Én ezt azzal bővítettem ki, hogy ott van az a myset.exe. Én nem szeretem azt, hogy az a privát kulcs ki tudja milyen úton jutott el ahhoz a szegény felhasználóhoz. Láttam olyat, hogy a helyi operátor elküldte e-mail-ban a felhasználónak.

Egyébként sokszor teljesen jó az SSH, nem kell mindenhol VPN. Amikor csak belépsz távolról egy gépbe, azért, hogy ott dolgozzál, arra nagyon jó az SSH, az azért nem árt, ha legalább IP címre leszűröd, hogy honnan engeded az SSH próbálkozásokat és SSH kulccsal autentikálsz és nem jelszóval.

**Hallgató:** ha dinamikusan kapom az IP-t, akkor milyen lehetőségem van a szűrésre?

**Előadó:** VPN.

**Hallgató:** VPN-nél ott a configban azt láttam, hogy a local illetve a távoli IP is rögzített.

**Előadó:** itt rögzítettem, nem muszáj illetve a távoli IP-t lehet név szerint is írni. Ez az egyik része. A másik, nem muszáj a localt, ha nem rögzítem a localt az OpenVPN szerveren – tehát ahová kapcsolódom –, akkor minden interface-en figyelni fog a megadott porton. Tehát adott esetben az ADSL interface-en is. Egy korlátozási lehetőség, így van. Azt meg, hogy mi a távoli neve? Lehet egy win-dns név, név szerint is mehetnek. Teljesen mindegy, az OpenVPN nem fog IP címet és nevet hitelesíteni a tanúsítvánnyal. A tanúsítvány csak arra fogja ellenőrizni, hogy helyes-e, nem ellenőrzünk nevet. Nyilván a kliens is ezen a szerver tanúsítványon, amit kiad, sőt van a netscape-certificate-type, a felhasználás célja. A cél, hogy szerver vagy kliens célú a tanúsítvány. A szervernek szerver célút, a kliensnek kliens célút adok és még konfigurálom az OpenVPN kliensben, hogy csak akkor fogadja el a tanúsítványt, ha az szerver célú, a szerverben csak akkor, ha az kliens célú, és így megakadályozom a kliens-kliens közötti véletlen tunneleket, meg szerver-szerver közötti véletlen tunneleket ugyanazzal a CA-val is.

**Hallgató:** a network manager-en kívül van más linuxos kliens?

**Előadó:** A kliens az úgy néz ki, hogy van maga az OpenVPN bináris, ami a szerver processz és a network manager az, ami ezt elindítja, de mondom, van pár, ami tudja indítani, leállítani meg menedzselni, az <http://openvpn.org>-on van egy pár. Az Debianban, Ubuntu-ban is vannak, ha nem is a hivatalos részeként az Ubuntu-nak, de ilyen multiverse vagy universe környékén vannak különböző GUI-k. Ami a szerveret is tudja menedzselni, mert ez kliens menedzsment a network manager-en keresztül, de van olyan is, amelyik szerveret is tud menedzselni, klienst is tud menedzselni

## Makár Zénó: OpenSolaris

OpenSolaris alatt lehet érteni az OpenSolaris közösséget, ami 45-50 csoportból áll és ezek a csoportok állnak az egyes projektek mögött, ilyen projektből van kb. 230. Az OpenSolaris alatt érthetjük még az OpenSolaris disztribúciókat is, ami csak egy-egy projekt ebből a kb. 230-ból. Van sok disztribúció, kettőt emelnék ki, az egyik a projekt Indiana, erre szoktak önmagában is, mint OpenSolarisra hivatkozni, ez az egy CD-s, ami egyben live CD és telepítő CD, ezt lehet ingyenesen megrendelni az opensolaris.org-ról. Kb. félévente van új kiadás belőle, a legfrissebb a 2008.11, előtte egy verzióval, csak a 2008.05. Amit érdemes tudni róla, hogy alpból ZFS root fájlrendszerrel települ és már az új IPS csomagkezelőt használja nem pedig a hagyományos system fájl pkg-t. Több repository van, tehát ezt a csomagkezelést úgy kell elképzelni, mondjuk például a debianos apt csomagkezelőt.

**A másik, amit fontos disztribúció az a Nevada**, ez most jelenleg a 110-es build-nél tart, két-három hetente jelenik meg belőle új, ez inkább a fejlesztőknek tesztelésre szánt disztribúció, ebben vannak bent mindig a legfrissebb újítások. Alapértelmezetten ez még UFS root fájlrendszerrel települ, sőt amikor legutóbb néztem, akkor még csak úgy lehetett ZFS root-tal telepíteni, ha a konzolos telepítést választjuk, ebben még a hagyományos csomagkezelés van és a hagyományos Solarisnál megszokott userland. Van még sok más kisebb disztribúció is, ezek elérhetőek az <http://www.opensolaris.org/os/downloads> oldalon.

**Miért érdemes kipróbálni illetve használni az OpenSolarist?** Az első két pontról már sokan hallottatok. Az egyik az ZFS, a Zettabyte File System, az egy 128 bites fájlrendszerrel egybeépített volume menedzser, tehát már nem válik úgy szét élesen a kettő, illetve a DTrace, ami dinamikus szoftver és kernel instrumentációs alrendszer, ezekről majd bővebben is fogok beszélni. Illetve az SMF, a Service Management Framework, ami a systemfájl init-et hívatott leváltani, valamint a Solaris Zónák, ami egy izolációs technika, tehát az operációs rendszer szintű virtualizáció. A Solaris xVM egy Xen alapú, tehát egy hypervisor alapú virtualizációs technológia. Tehát itt is megtalálható az Apache, a MySQL, a PHP, ezt WEB Stack-nek hívják.

**A ZFS.** Mint mondtam nem válik szét élesen a kötetkezelés és a fájlrendszer, ennek olyan előnyei adódnak, hogy a fájlrendszerről készíthető egy pillanatfelvétel, ami lényegében atomi. A fájlrendszer mindig konzisztens állapotban van, ugyanis az egész fájlrendszer működése tranzakcionális. Jól skálázódik, tehát minél több diszk van alatta, annál gyorsabb és jobb lesz, hatékonyabb. Ezek a fájlrendszerek online méretezhetőek, online lehet a mount pontjukat is megváltoztatni, tehát nem kell az, hogy a nyitott fájl deskriptorokat bezárjuk, és csak utána lehet átsatolni valahova máshova a rendszeren belül. Maximális mérete az 256 quadrillió Zebibyte, ez nagyon-nagyon sok. Pár napja hallottam egy előadást, ahol azt mondták, hogy ha az egész Föld felszínét beborítanák lehetőleg nagyobb tárolókapacitású diszkekkel, akkor sem lehetne ekkora adattároló felületet kialakítani.

**Hogy is működik ez, hogy tranzakcionális?** Fastruktúrában tárolódnak az adatok eredetileg, legfelül van egy überblokk, azalatt olyan blokkok találhatóak, ezek az indirekt blokkok, amibe pointerok vannak és végül a legalsó blokkokban találhatóak maguk az adatok. Minden írás úgy történik, hogy az alsó adatblokkból készül egy copy-on-write és oda beírjuk az új adatot, ezután lépésenként felfelé haladva a pointerokat tartalmazó blokkokat is lemásoljuk és módosítjuk bennük a pointerokat. Majd a végén a legfelső, az überblokk is módosul, ez a művelet atomi, ezáltal, ha közben elhal a gép, elmegy az áram nem lesz olyan, hogy nem konzisztens a fájlrendszer, mert akkor még a régi überblokk van és az a régi pointerokra fog mutatni, tehát nincs olyan, hogy valahol már átírtuk az adatot és az mégsem látszik. Illetve ugyanezen az elven működik a snapshot készítés

is, ott annyi a különbség, hogy az überblook módosításánál nem hagyjuk elveszni a régi überblookot, hanem azt megtartjuk és onnantól kész is a snapshot.

**DTrace: lehet érteni az egész felhasználói szoftver és kernel instrumentációs alrendszer**, de DTrace-nek hívják már magát azt a szkript nyelvet is, amivel az alrendszer tudjuk elérni. Régebben ezt d-nyelvnek hívták, de aztán szóltak nekik, hogy már van d-nyelv, így átnevezték DTrace-nek. Általános eszköz, sok mindenre lehet használni, tehát szoftverekben, hibakeresés, teljesítményelemzés, monitorozási feladatokra nagyszerűen használható, és amit fontos elmondani, hogy futási időben módosítja az instrumentális szoftvert bizonyos szondák mentén és nincs jelentős hatása az instrumentált rendszer teljesítményére. Valamint hibakeresésnél is egy fontos tulajdonság, hogy a legtöbb olyan debugger eszköz, ami futtatási időben tud hibákat keresni, egyrészt sokat elvesz az adott teljesítményéből, másrészt esetleg plusz hibákat injektálhat a szoftverbe. A DTrace ezeknek a valószínűségét a minimálisra csökkenti és nincs hatása az instrumentális rendszer teljesítményére.

**A zónák: operációs rendszer szintű virtualizáció**, tehát hogy egy kernel felett több elkülönített userland fut, és belülről mindegyik zóna egy önálló Solaris instance tűnik, de ennek természetesen vannak hátrányai, mivel egy kernel fut, zónákon belül nem tudunk minden kernel opciót elérni, például zónán belül nem tudunk DTrace-t futtatni, ugyanakkor a kernel bizonyos részeit mégis el tudja érni a zóna, tehát adhatunk neki akár egy saját IP stack-et, amit a hasonló technológiák többségénél nem tehetünk meg. Mondjuk ezt úgy tudjuk csak megtenni, hogy az egész hálókártyát dedikáljuk a zónának. Meg kell említeni a Branded zónákat, ezzel lehetőség nyílik, hogy más kernelt emuláljunk a zónában. Egyrészt régebbi Solaris verziókat, másrészt van egy LX Brand, amivel Linux 2.4-es kernelt tudunk, illetve régebbi RedHat verziókat tudunk zónán belül futtatni illetve most folyik a tesztelése a 2.6-os Linux kernel emulálására képes Branded zónának. Valamint, ha már az OpenSolaris-nál és az Indiana projektnél tartunk, a SUNWipkg brand-et is meg kell említeni. Hagyományosan a zónák a Live Upgrade nevű régebbi solarisos rendszerfrissítő metódust használták az új zóna létrehozására, míg az Indiana-ban ez a régebbi fajta csomagkezelés már nincs meg, ezért egy új zóna telepítése az Indiana-ban úgy történik, hogy az IPS csomagkezelő rendszerrel letölti az internetről a csomagokat és azt telepíti bele a létrehozandó zónába.

**SMF (Service Management Framework):** adódik rengeteg probléma a system fájl init-tel. Például nem kapunk rendes szolgáltatás menedzsmentet, tehát ahhoz, hogy ne csak leállítsuk az adott szolgáltatást, hanem azt akarjuk, hogy következő újraindításnál se induljon el, akkor az /etc/rcX.d-ből is ki kell szedni az adott symlink-et. Nem tartja nyilván az egyes szolgáltatások közötti összefüggéseket, hanem csak az /etc/rcX.d symlink nevében lévő kétjegyű számból állapít meg egy sorrendet, hogy hogyan indítsa el az egyes szolgáltatásokat. Esetleg még benne van pár sleep, meg hasonló csúnya dolgok az init szkriptekben, illetve a szolgáltatások leállítása is problémás system fájl init esetén, mivel nem pontosan tudni, mi a PID-je az egyes processzeknek. Mondjuk érdemes megnézni egy Apache init szkriptet, hogyan próbálja leállítani az egyes szolgáltatásokat. Az SM-ek használatával ezek kiküszöbölhetőek, ez egy központi konfigurációs menedzsment megoldás, egyrészt a szolgáltatásokról adatokat XML-ben tárol. Le van írva a szolgáltatás neve, milyen paraméterekkel, hogy kell elindítani illetve, hogy milyen függőségei vannak. Egységesen az svc kezdetű parancsokkal tudjuk őket kezelni és gondoskodik arról, hogy ha egy szolgáltatás valamilyen hiba miatt leáll, akkor újraindítja, és nyilvántartja a függőségeket a szolgáltatások között, ezáltal az egész boot folyamat párhuzamosítható.

**A Solaris xVM Xen alapú virtualizáció**, amivel több, mint a Xen, hogy a domain0-ban egy OpenSolaris fut, ennek olyan előnyei vannak egyrészt, hogy ZFS volume-okat használhatunk a virtuális gépeink tárolására, valamint DTrace szondák vannak a hypercall-ok nyomon követésére, ezáltal bármilyen guest rendszerben lévő hibák sokkal könnyebben kideríthetőek. Mivel Xen alapú, képes paravirtualizációra is és full virtualizációra is. Paravirtualizáció, amikor a guest operációs

rendszer tud arról, hogy alatta egy hypervisor van, full virtualizációnál nem, tehát ott bármilyen módosíthatatlan operációs rendszert lehet futtatni, viszont ahhoz hardveresen támogatottnak kell lennie a rendszernek Intel VT-x vagy AMD-V processzor kell a szerverbe.

**Web Stack:** ez a CoolStack nevű csomagot utódja, megtalálhatóak benne kiszolgáló programok, úgy mint az Apache és a lighttpd webserverek, Squid proxy szerver, adatbázisszerverekből MySQL és PostgreSQL, illetve fejlesztői környezetek Perl, PHP, Python, Ruby támogatott illetve a Memcached is benne van.

**A legfrissebb verzióban a 2008.11-ben lévő újdonságok, amit érdemes megemlíteni.** Egyrészt a Time Slider, említettem, hogy a ZFS fájlrendszerekről pillanatkép készíthető. A Time Slider egyrészt áll egy automatizált snapshot készítő rendszerből, szolgáltatásból, ami bizonyos időközönként pillanatképeket készít a bekonfigurált fájlrendszerekről, illetve egy GNOME fájlkezelő, amiből ezeket az elkészített snapshot-okat tudjuk megjeleníteni.

**Amit még érdemes megemlíteni az a COMSTAR,** ez a Common Multiprotocol SCSI Target. A WebStack kapott egy saját repository-t, ahol már elérhető a Drupal, Django, phpMyAdmin és egy-két webfejlesztéshez hasznos program, illetve pár szoftver verziója frissült. Ugorjunk a COMSTAR-ra. A SCSI protokollnál megemlítem, hogy van target és initiator. Az initiator az aki parancsokat küld, a target pedig aki fogadja, tehát read és write parancsok. Azáltal, hogy ez a program bekerült az OpenSolarisba, ezáltal tud SCSI target-ként viselkedni, jelenleg FC, SAS protokollok felett. Ezáltal egy olcsó, sok diszkes szerver, egy SAN storage-é tudunk alakítani, és nem kell megvenni a drága készüléket hozzá. Készülőben van az iSCSI target, az iSCSI Extensions for RDMA illetve az FC over Ethernet protokollok feletti COMSTAR is. Az iSCSI Extensions for RDMA a SCSI-n keresztül egyből DMA parancsokat küld. Az a nagy előnye, ha nagyon nagy az adatforgalom, akkor a szoftveres iSCSI rettentő mód lelassulhat, és ezáltal sokkal kevesebb CPU terhelést kap a rendszer.

Nézzük meg a Time Slider-t. A System->Administration->Time Slider alatt tudjuk beállítani, egyrészt megadhatjuk azt, hogy az összes fájlrendszerrel készítsen időnként mentéseket vagyis snapshot-okat, vagy beállíthatjuk, hogy konkrétan melyik fájlrendszeréről akarjuk. Most beállítottam, hogy a home könyvtárról készüljenek ilyen mentések. Megadhatjuk azt, hogy az egész tárolókapacitás hány százalékát használja a rendszer, ha ezt eléri az elkészített snapshot-ok mérete, akkor többet már ne csináljon.

Biztos felmerült bennetek a kérdés, hogy milyen időközönként készülnek a snapshot-ok? Ezzel kapcsolatban megmutatom, hogy a Service->Management->Framework hogyan is működik. Van egy olyan parancs, hogy svcs, ezzel az SMF-fel menedzselte szolgáltatásokról státusz információt tudunk lekérni, a „-a” kapcsolóval megmondjuk neki, hogy az összes szolgáltatás érdekel minket, majd rá grep-pelünk arra, hogy snapshot. Láthatjuk, hogy van 5 ilyen szolgáltatás, mindegyik online állapotban. Van itt ilyen, hogy hourly, weekly, montly, frequent és daily. A hourly óránként készít snapshot-okat és ebből 24 darabot tárol el, a weekly hetente és ebből 4 darabot tárol el, a montly az havonta és 12 darabot tárol el, és a daily az naponta és 31 darabot tárol el, természetesen, ha még nem érte el a beállított százalékát a szabad tárolókapacitásnak. Illetve itt van még a frequent, ami negyedóránként készít snapshot-okat és ebből 4 példányt tárol el, de tudjuk módosítani, hogy a frequent milyen időközönként készítsen snapshot-okat. Az svccfg paranccsal megadjuk neki, hogy az auto snapshot frequent-et változtassuk, és a ZFS/period legyen mondjuk 30 (svccfg -s zfs/auto-snapshot:frequent setprop zfs/period=30). Ezután megmondjuk a szolgáltatásnak, hogy frissüljön (svccfg -s zfs/auto-snapshot:frequent refresh), szintén svccfg-vel és újraindítjuk a szolgáltatást (svcadm restart zfs/auto-snapshot:frequent). A szolgáltatások elindítását, újraindítását az svcadm paranccsal tudjuk végrehajtani. Restart-tal mondjuk neki azt, hogy induljon újra. Ha azt szeretnénk, hogy ne készüljenek naponta snapshot-ok, mert úgy gondoljuk, hogy ha valami hiba van, azt úgyis 1-2 órán belül észrevesszük, vagy pedig sokkal később lesz az, és emiatt mondjuk feleslegesnek

tartjuk, akkor ezt a szolgáltatást kikapcsolhatjuk az „svcadm disable zfs/auto-snapshot:daily” paranccsal, de ugyanígy vissza is lehet kapcsolni az enable paraméterrel.

**Térjünk rá egy kicsit az ZFS-re.** Az „zfs list” paranccsal tudjuk megnézni az elérhető fájlrendszereinket, ha látni akarjuk milyen snapshot-ok vannak, akkor „zfs list -t snapshot”, itt egyébként felsorolásszinten lehet több mindent is megadni, nem csak a snapshot-ot, hanem mondjuk filesystem-et, volume-ot, de minket most csak a snapshot-ok érdekelnek. Látom az elkészített snapshot-okat, amik a fájlkezelőben elérhetőek. Itt megjelent egy ikon, ezzel lehet a létrehozott snapshot-okat böngészni, és ha visszamegyünk egy régebbi állapotba, és van ott egy fájl, ami jelenleg nincs meg nekünk, akkor jobb klikk és „restore to desktop”-pal helyre tudjuk állítani. És meg is jelent a desktop-on ez a fájl.

Fájlrendszer szinten is be lehet állítani azt, hogy készüljenek-e snapshot-ok, ahogy a menüben is látható volt, ez parancssorból is megtehető, de először nézzük meg, hogy miket lehet állítani egy fájlrendszeren. A „zfs get all” parancssorral lekérdezhethetjük az összes attribútumát egy adott fájlrendszernek. Most lekérjük a home könyvtámról, látható, hogy vannak benne ilyenek, hogy „com.sun:auto-snapshot:weekly true”, ezen van engedélyezve a snapshot, de ha azt akarjuk, hogy ne legyen az, hogy mind az ötféle időközönként, tehát naponta, óránként, hetente, havonta, meg negyedóránként készüljön, hanem csak naponta, akkor beállíthatjuk egyrészt azt, hogy „zfs set com.sun:auto-snapshot:daily=true rpool/export/home/zeno”. Beállíthatjuk azt is, hogy „zfs set com.sun:auto-snapshot=false rpool/export/home/zeno”, így már csak hetente fognak snapshot-ok készülni az adott fájlrendszerről.

Most megnézzük, „zfs get all rpool/export/home/zeno”, akkor láthatjuk, hogy csak a hetente készítendő snapshot-ok vannak engedélyezve. És akkor hogy is működnek ezek, hogy snapshot visszaállítás? Hozzunk létre egy új fájlrendszert, ezt a zfs create paranccsal (zfs create rpool/a) tudjuk megtenni és láthatjuk, hogy létrehoz egy rpool/a, ami alapértelmezetten az rpool könyvtár alá van felcsatolva. Állítsuk át a csatolási pontot „zfs set mountpoint=/a rpool/a”, megadjuk, hogy melyik fájlrendszerre és már át is íródott a csatolási pont. Hozzunk létre egy fájlt, írjunk bele valamit, készítsünk erről a fájlrendszerről egy snapshot-ot, ezt a „zfs snapshot” paranccsal tudjuk megtenni

(zfs snapshot rpool/a@snapshot1), úgyhogy beírjuk a fájlrendszer nevét és utána kukaccal elválasztva a snapshot nevét. Listázzuk ki az eddigi snapshot-okat és látható, hogy létrejött ez a snapshot, töröljük le mondjuk ezt az „a” fájlt, de aztán rájövünk, hogy ez esetleg fontos lehet nekünk, és vissza akarjuk állítani, ezt úgy tudjuk megtenni, hogy „zfs rollback rpool/a@snapshot1”, és újra ott van az a fájlunk.

A snapshot csak olvasható, ha azt akarjuk, hogy legyen az egésről egy írható változatunk is, akkor ezt le kell klónoznunk, azt úgy tudjuk megtenni, hogy zfs clone (zfs clone rpool/a@snapshot1 rpool/b), megadjuk a snapshot nevét és itt is létrejött az a fájl és látható a ZFS résznél, hogy az különálló fájlrendszer, és hogy 0 a mérete. ZFS-ben csak egyszer van eltárolva az, amit létrehoztunk, az a fájl, csak a pointerok ugyanoda mutatnak. Ha megnézzük jobban, látható, hogy az „a”-ról volt ez a snapshot, és hogy ha olyat akarunk, hogy a „b” legyen a magasabb prioritású, tehát képzeljük el azt, hogy még frissítjük a rendszerünket, nem akarjuk azt, hogy ha valamelyik csomag mégsem tetszik, gond legyen a visszaállítással, ezért csinálunk a rendszerről egy snapshot-ot, azt leklónozzuk, feltelepítjük a klónba az új rendszert, ha mégis tetszik nekünk és azt akarjuk, hogy az maradjon. Ezt meg tudjuk tenni azzal, hogy van egy olyan parancs, hogy zfs promote, megadjuk a klón nevét (zfs promote rpool/b), és ha most nézünk egy „zfs list -t snapshot”, akkor láthatjuk, hogy a snapshot már a „b”-ről készült el, tehát az van magasabb hierarchia szinten. Ha nézünk egy „zfs list”-et (zfs list -t snapshot), akkor láthatjuk, hogy az összes fájlrendszerünknek van még 43 Gigabyte szabad helye, az rpool/a 15 k-t foglal, mondjuk a pool/b az 35 k-t, és be akarjuk korlátozni, hogy mekkora méretű lehet egy fájlrendszer, akkor azt a „zfs set quota” paranccsal tudjuk megtenni. Mondjuk korlátozzuk 1 Gigára az rpool/b-t (zfs set quota=1G rpool/b), akkor láthatjuk, hogy az elérhető területeknek már csak 1024 megája maradt, de ha azt akarjuk, hogy ezzel a területtel mindig rendelkezzen, akkor azt a „zfs set reservation”-nal lehet megtenni

(zfs set reservation=1G rpool/a). Most láthatjuk, hogy az összes többi fájlrendszernek az elérhető területe csökkent 1 Gigabyte-tal. A ZFS-ben nem csak fájlrendszert hozhatunk létre, hanem volume-ot is, „zfs create -V”-vel (zfs create -V 1G rpool/c). Létrehozunk egy „c” nevű volume-ot, a volume-oknál pedig le is foglalja azt, amekkorára létrehozuk a volume-ot.

**Milyenek a zónák?** Egyrészt a global zóna az az aktuálisan futó OpenSolaris, látszik, hogy van nekem egy master zóna, amit létrehoztam. A zone paranccsal tudjuk kezelni a zónákat, mondjuk indítsuk el ezzel a „zoneadm -z master boot” paranccsal, mert a master zóna is running állapotba került. A zlogin paranccsal tudunk belépni a zónába (zoneadm -z master boot), belülről ez teljesen különálló OpenSolaris-nak tűnik. Ha megnézzük jobban a fájlrendszerünket, ez a zóna is egy ZFS fájlrendszeren van, tehát azt jelenti, hogy erről is csinálhatunk snapshot-ot, klónozzhatjuk és akkor pillanatok alatt csinálhatunk magunknak egy új zónát.

Ez a legfrissebb verzióban még tovább egyszerűsödött, ugyanis a zone parancs kapott egy olyan lehetőséget, hogy clone és ez az egész ZFS-es műveleteket megoldja helyettünk, annyit kell tennünk, hogy a meglévő zónáknak a konfigurációját exportáljuk, átszerkesztjük, átírjuk a zóna útvonalát, létrehozuk az új könyvtárat és beállítjuk, hogy csak a root-nak legyen mindenféle joga hozzá. Fontos még, hogy a zónának leállt állapotban kell lennie, tehát nem lehet running-ban. Ezután a módosított konfigurációs fájlal létrehozunk egy új zónát, zonecfg paranccsal, láthatjuk, hogy a beállítások elkészültek és zon -ad-mínusz z, új zóna, klón és az eredeti zóna paranccsal hozhatjuk létre a klónt (zoneadm -z zona clone master). Látható, hogy elkészült az új zóna és az installált állapotban van, boot-oljuk be. Először pár konfigurációs beállítást kell elvégezni, meg kell adni a terminál típusát, beállítjuk a host nevet, beállítjuk az időzónát és a root jelszót.

Ezzel létre is jött az új zónánk, láthatjuk, hogy létrehozta neki az új ZFS fájlrendszereket, vagyis a klónokat és láthatjuk, hogy míg az eredeti zóna 240 MByte-ot foglal, addig az új zóna összesen 1,29 MByte-ot. ZFS-nél még kihagytam, a ZFS-hez kapcsolódik még egy parancs, ez a zpool, ezzel az úgynevezett pool-jainkat tudjuk menedzselni. Ezek ilyen nagy tárolók, amivel allokálunk az egyes ZFS fájlrendszereknek területet, ezzel a paranccsal tudunk létrehozni egy új pool-t vagy állíthatunk be különböző RAID szinteket, ha több diszkontálás, tükrözés, RAID-Z-t, ami RAID 5-höz hasonló leginkább.

**DTrace:** a rendszer teljesítményének elemzéséhez hogyan lehet ezt használni? Van itt egy ilyen szkript, hogy syscalls.d, kis rövid szkript. A rendszerhívások be és kilépési pontjainál felírja azt, hogy konkrétan milyen processz hívta azt a rendszerhívást és ezeket elkezdí számolni. És ha ezt lefuttatjuk, várunk, amíg tetszik, addig számolja a rendszerhívásokat, nyomunk egy ctrl+c-t és láthatjuk, hogy milyen program, hány rendszerhívást hajtott végre. Valamint konkrétan read vagy write vagy konkrétan milyen rendszerhívás volt. Ennek egy kicsit fejlettebb változata ez a második szkript, ami azt csinálja, hogy egy eloszlást számol abból, hogy melyik program mennyi időt töltött rendszerhívásban, és hogy kb. milyen időközönként. Skálán ábrázolja, hogy nagyjából milyen időt igénylő rendszerhívásokból hány darab volt az adott mérési periódusban és erről egy szép hosszú listát kapunk processzenként. Ezt akkor érdemes használni, ha az előző szkripten megállapítottuk, hogy kb. melyik processzek lehetnek azok, amik nagyon leterhelik a rendszert és párra le tudjuk szűkíteni és azokat megvizsgáljuk.

**Fejlesztésnél hogyan használható a DTrace?** Írtam egy egyszerű kis python szkriptet, ez ennyit csinál, hogy kiírja, hogy „Helló LOK!”, csak kicsit körülményesen sikerült megírni. Ez a DTrace pedig azt csinálja, hogy vizsgálja a konkrét Python szkriptnek a belépési és kilépési pontjait és ezeket aztán a felére, hogy konkrétan melyik függvény hívta meg melyiket. Látszik amellet, hogy a Python fut még rengeteg más is elindul nem csak ez a hello.py, látszik, hogy a func\_hello meghívta a func\_hello-ot, azon belül a func\_lok-ot, valamint szépen a visszatérési értékek is láthatók.

Érdemes megnézni, ha valaki kipróbálja az OpenSolaris-t, a /opt/DTT könyvtárban van egy hatalmas DTrace gyűjtemény, amit érdemes megnézni. Rengeteg mindent lehet vizsgálni egy adott rendszerben.

**Mi várható a következő verzióban?** 2009 júniusában, a legfontosabb újítás a Crossbow lesz, ez egy hálózatvirtualizációs technológia. Adatfolyam szinten tudjuk garantálni a sebességeket, az adatfolyam lehet szolgáltatás protokoll vagy virtuális gép. Tudunk majd olyat csinálni, hogy van 1 gigabites hálókártyánk, azt mondjuk, hogy az egyik zónának abból 500 megabit/sec sebessége mindig legyen. Tegye egy másik zónába a HTTP-nek legyen mondjuk 300 megabit/sec, és egy FTP-re még 400 megabit/sec, ugye ez már több, mint 1 gigabit, meg lehet mondani, hogy az első zóna előnyt élvezzen, tehát ha ő éppen nem használja ki a megabitjét, akkor a többiek használjanak amennyit tudnak, de hogyha ő kihasználja, akkor a többiekét automatikusan csökkenti.

## Erdei Zsolt: Green IT bevezetése egy multinacionális környezetben, egy általános iskolában

Egy monitort hoztam be és nem értettem, hogy miért van három helyen rámatricázva ez az ENERGY STAR? Akkor még nem volt Google sem, hogy begépeljem, enter, és megkapjam a választ. Elfogadtam egyszerűen, hogy ez jó. Aztán pár év múlva hirdették azt, hogy ez a monitor kevés áramot fogyaszt, ez nekem még jó is, de akkor nem érdekelt. Anyám fizette a villanyszámlát, nekem az kellett, hogy szépek legyenek a színek. Most viszont már én fizetem a villanyszámlát, most már csak olyan monitort veszek, ami ENERGY STAR matricával el van látva és biztosan spórol nekem áramot. A 92-es kormányötlethez csatlakozott az USA-ban, Új-Zéland, az európai unió és az összes nagyobb ország, még az oroszok is.

Aztán jött az eco. Még a dzsunka PC-t is valami egyszerű hardverből összerakjuk, csak ha én azt kidobom az utcára, az alaplapom az ott elrohad, szennyezi a környezetet. Az Eco-s cuccokat meg arra nyomják rá, aminek a gyártásánál, a felhasználásánál és a megsemmisítésénél nem keletkeznek egyéb káros anyagok.

A 80+ a legjobb, szintén amerikai ötlet. Vannak a tápegységek, most nem akarok nagyon villamosmérnöki babérokra törni, de abba bemegy az áram és kijön a kevesebb áram, a kettő között meg hőt termel. Na, ezt a hőtermelést szeretnék egy picivel csökkenteni és effektíve kevesebb veszteséggel működjenek ezek a tápegységek, ez is egy cél. Teljesen logikus cél, hiszen nem mindegy, hogy 40 wattal többet kell, felvegyen a tápegység, hogy két winchesterrel többet meghajtson, vagy elég ehhez 20.

Egyik egy ilyen kész tanulmány, ezt egy német kollégám küldte el nekem. Tisztázzuk először, mi az hogy flops? Nálunk egy datacentert konszolidáltak. Ez annyit jelentett az első körben, hogy volt cirka 600 szerverünk, meg két, elég tetemes kb. félszobányi storage-unk. Azt hiszem HP. Ez 127 megaflop/watt, ugye most már így számolunk. Aztán kicsit racionalizálták, ugyanazon a storage-on, de már 320 gépen futottak ezek az alkalmazások, szándékosan nem mondom hogy hogyan, majd kitérek rá. Aztán fél év múlva jött egy drága öltönyös főszer, menedzser titulussal, és azt mondta: uraim, ebben a rendszerben nincs elegendő redundancia. Ezért egy storage-ot kivágtak. Viszont a másikba belepakoltak egy rahedli lemezt és megduplázták a számukat, kvázi clusterbe tették. Így elérték a 131 megaflop/watt mérőszámot. Mi a lényeg? Nyertek 4 megaflop/watt-ot egy csomó pénz elköltésével. Erről szól a green IT, erről szól a green computing a nagyvállalatoknál. Erre akartam körülbelül kilyukadni. Tök mindegy, hogy mit csinálunk, hogy szervezzük át. Az átszervezés, az a legkirályabb, nincs annál jobb, mint amit egy nagyvállalat tehet. Ez egy valós példa, ti is használjátok a szolgáltatásait, csak a magyar leányvállalatát.

**Másik példa**, velük én vagyok kapcsolatban napi szinten, nekik még mindig SuSE Linux 10-es szervereik vannak, én azért még mindig kiteszem a BSD logót a jobb alsó sarokba, biztos, ami száz. Szóval SuSE 10-es szervereik voltak, olyan 100 darab. Ezen 127 megaflop/watt, ezt számoltuk egy este egy kollégámmal, nagyságrendileg. Aztán kitalálták, hogy virtualizálunk, ez a jövő uraim, irány ezen az úton. SuSE 10-es szerverek 186 megaflop/watt lett a teljesítmény. Megérte? Igen, sokkal többet dolgozik a gép azért az energiáért, de nem csak ez a poén, hanem írtam ide zárójelbe a számot. Ezek az alkalmazások, a virtuális szerverek, gyakorlatilag meg is kétszerezték a virtuális szerverek számát. Nem mindegy, hogy egy gépen futtatok egy adatbázist, fut, mint az örült, futtatok kettőt, elfut, mint az örült, de tesztek bele még két CPU-t szétbontom és futtatok négyet. Nyilván később ezek a cégek is rájönnek arra, hogy egy határig éri meg nekik az, hogy ezt játsszák, de lássuk be, van olyan cég, ahol ez megéri így mind pénzügyileg, mind teljesítményügyileg. Ebben a történetben nagyon nagy tartalékai vannak, ha neki 5 perc múlva szüksége van 10 clusterre, mert ő most SAP-t akar futtatni irdatlan adatbázisokkal, akkor felhívja az IT-s gyereket, hogy kérek 2x5 gépet clusterbe holnap reggel, és a tartalékomból kiveszem. Ezekbe a történetekbe ez is benne van, hogy ha egy nagyvállalatot nézünk.

De nézzünk általános iskolát is. Az okok lehetnek: pénz, mert azért a magyar általános iskolák, nem a legjobbak, én, mint a munkahelyemen dolgozó ember, csoportvezető vagyok, szeretnék végre kvalifikált embereket is felvenni rendszeresen. Az iskolának érdeke, hogy egy szélesebb spektrumot oktasson és a szélesebb spektrum viszont a virtualizációban rejlik, mert az iskola nem vehet meg 640 gépet, hogy mind a 640-et bemutathassa. Ok 4 primary partíció van, értem én 260 PC-t sem vehet meg az iskola erre a történetre. Az iskola törekedjen arra, hogy az a diák, aki kilép az ajtón végzés után, nem az élelmiszer-iparira gondol rögtön – a pékek nem biztos, hogy jó programozók lesznek – , de az aki informatikai iskolában végez, hiszem azt, hogy Linuxon, BSD-n egyéb Unixokon tanuljon, dolgozzon, fejlesszen, csak használja a mindennapokra. Lehet, hogy 10 év múlva, ha visszamegyek a jelenlegi cégemhez dolgozni, nem egy Microsoft Windowst kapok a laptopomra image-ből, hanem valamilyen operációs rendszert. Ezeket írtam direkt kötelességnek.

Van egy buktató, az iskolatitkár. Anélkül, hogy fikáznám az iskolatitkárt, a középiskolába, ahová én jártam, ott volt egy rendszergazda, még a Potatoe volt a feature, így nekiment, hogy minden Linux, ami van, az Linux lesz. Jó, jó, de az iskolatitkárnak van egy DOS-os órarendtervező programja, semmi gond, majd fut DOSEMU-ban FreeDOS-szal. Körbe kell gondolni, a programot nem tudjuk lecserélni, illetve újra lehet írni azt is, az sem egy agysebészlet. Auditáltatni kell, elfogadom, fusson DOSEMU-ban.

Munkahelyen a szabadságtervező rendszer csakis Internet Explorer 7.0-val működik. És csak akkor tudtam elmenni szabadságra, amikor ezt el tudtam indítani, mert nekem mindig lefagy. Win XP tök standard, befagy az Internet Explorer 7.0, ezt úgy abszolváltam, hogy felhívtam a HR-es kolleginát, hogy írja be, hogy ekkor és ekkor szabadságon vagyok. Te miért nem tudod beírni? Mert nekem van Firefoxom és nem érdekel.

Ekkora blokkoló tényező kb. az iskolatitkár is. De meg lehet oldani, mert neki teljesen mindegy, mit futtat, hol érdeklí őt? Induljon el az a csodaprogram, amit ő használ, csak kétszer kelljen odakattintania. Egy shell szkript megindítja a DOSEMU-t, jól felparaméterezve, abban meg egy batch fájl, ami elindítja. Ha valaki ezzel foglalkozni akar, állítom, hogy az iskolatitkár néni 15 év múlva elmegy innen nyugdíjba és biztos, hogy ő egész életében Microsoft DOS-t használt. Aztán lehet, hogy nem, sőt lehet, hogy még HDD sem lesz a gépében. Pont ezért érdekes a történet a diákok szempontjából, mert nekik is tanulniuk kell. A BSD-ek nem véletlen ördögök. Kolléga, például Guska értelemszerűen pingvint tenne...

Praktikusan a diákok azok, akiknek minél több, minél érdekesebb dolgot meg kell mutatni, és itt egy hatalmas kanyarral áttérnék arra, hogy minél több olyan operációs rendszert, ami nem Windows. Magának, ennek a konferenciának a célja, hogy a Linux az oktatásban történetet népszerűsítsük. Aztán a kollégák azt mondták, hogy ez egy open source igazából, szabad szoftver. Népszerűsítsük őket? Ez ugye a BSD is. Ennek a diáknak nem teljesen mindegy az, hogy most Linuxot, FreeBSD, OpenBSD, Debian Linux stb. A NAT-ban az van meghatározva, hogy mit kell megtanulni, de nincs az meghatározva, hogy mit nem tanulhat meg.

Visszaulalnék 15 évvel ezelőttre, amikor valaki azt mondta egy nagy kontinensen, hogy márpedig ez a nyomtató inert fog nekem küldeni, azt sem munkaidőben tette, jó tőle elvárom, hogy munkaidőben tegye, de délután is hozzáadott valamit. Megjön a csomag, értem, hogy ezt kell feltelepíteni és tanítani. Azt szeretném, hogy megtörténjen az, hogy Linux/Unixokkal foglalkozunk az iskolában.

**A másik téma a virtualizáció.** Az teljesen világos, hogy a nagyvállalat hajlandó VMWare-t vásárolni, ennek két oka van, egy, mert van rá pénze, kettő, mert ahhoz van support. Valami nem megy, akkor jön a supportos, a bejáratnál meggyepálják, beküldik, megcsinálja, kiengedik. Az iskola nincs ennyire elengedve, ha valaki ide kijön, azt, mint a messiást, de még pénz sincs rá. Ugye erre van a Qemu, nincs csili-vili felülete – de van hozzá –, de be lehet gépelni a parancssorba, hogy Qemu, diszknév és nagy enter, és innentől kezdve, amit te mondtál, hogy van mindenkinek

egy kis rack-je ötletre, kicsit fejlesszük tovább. Mi van akkor, ha van egy szerverünk, legyen FreeBSD – az én kedvemért –, de mivel ti Linux fanok vagytok, legyen Ubuntu. A diákoknak van egy ilyen kis dzsunka PC-jük, monitor hozzá, van valamilyen tápegység, billentyűzet, egér meg van benne egy hálókártya.

Egy olyan hálókártya, ami ezt a három betűs mágiát ismeri, hogy PXE. Szépen boot-ol, ez egyébként egy IBM blade center, beszélek a top500-ról információ szinten. Boot-ol egy FreeBSD-t, de mondjuk, legyen Linux, boot-olja ezt a FreeBSD-t, majd XDMCP majd rácsattan annak a grafikus szerverére, legyen ez egy X.Org. Elkészültünk, onnantól van egy nagy tárhelyünk, amit beléptettünk a szerverbe, ne adj isten külső storage, és van egy local gépünk, amit igazából semmire nem használunk, ezért elvárásaink sincsenek vele szemben. Akkor egyszer a tanár azt mondja, hogy kattintsatok a nagy X betűre. Nagy X, megnyílik az xterm, írjátok be a következőt: Qemu, Debian/GNU Linux, enter, és elindul egy Linux, tud vele dolgozni.

Mi több! Ez a Linux egy image-ben fut. Mi történik, ha tönkreteszi a gyerek, hát akkor másoljuk át az /etc/passwd-t, rm/etc/passwd, enter. Semmi gond, image-ből visszamásolom az image-t. Elindít egy Windowst, lehet nekik Microsoft Word-ben kiadványszerkesztést tanítani, amivel engem büntettek, mert azt mondta a nő, hogy kiadványt szerkesztünk, ha valakinek havária van a gépén, szól. Mi havária, milyen gépén, mi az hogy kiadványszerkesztés? Én programozó leszek. Lehet akár Windowst is tanítani Microsoft Office-szal, másnap lehet neki Windowson OpenOffice-t tanítani, ha eléggé úgy tűnik, hogy már összekeveri a kettőt, akkor be lehet csempészni mögé egy FreeBSD-t OpenOffice-szal, csak jól kell megalkotni. És ez a lényege.

Tegyük fel, hogy a Qemu nem egy image-re cuppan rá, hanem egy device-re iSCSI target, nevezzük nevén. Azalatt is egy fájlrendszer lesz, nem nagy mágia a történet, de a FreeBSD-ben van egy olyan szó vagy valami, hogy Vinum fájlrendszer, nem tudom kinek ismerős? Veritas Filesystem (VxFS)? A Vinum fájlrendszer az, ahol szépen tudsz lemezeket részegységekre bontani, csoportosítani, snapshot-olni és odacsatolni, ahová te akarod. Ezek után a Qemu image-k mögött levő tartalmat gyakorlatilag a tanár úr az asztal mögül szabályozza, és amikor a tanuló „enter”-t üt a gépénél, Windows, Linux, BSD, Solaris elindul és boldoggá teszi a diákot, aki aztán ebből tanul. Éppen ezért gyakorlatilag az összes lényeges operációs rendszert meg lehet neki tanítani.

Green IT, green computing, miről beszéltünk eddig? Arról, hogy gépeket kidobálnak, vesznek helyettük meg teljesen újraszerveznek. Mit gondoltok, egy iskolában mi lenne a hathatós irány? Vékony kliens, nincs helyi géped, csak megjelenítik a képet, egy kihelyezett monitor billentyűzettel, egérrel, de hálókártya van benne, szükségképpen. Ezért volt jó, amit mondtál, hogy mindenkinek van egy kis rack-je. Miért? Ha van egy központi gépem, abba az összes HDD-t beleteszem és azok a kis gépek, távolról őket eléri. Azoknak a vékony klienseknek, vagy mondjuk, vegyünk egy általános iskolát, általánosságban egy iskolát, nekik nincs pénzük rendes, valódi vékony kliensre, nevezzük nevén, dzsunka PC. Éppen tökéletes rá.

**Hallgató:** ehhez hasonló megoldást mi is alkalmazunk, csak mi a hálókártyával sem molyoltunk, ősrég gépek voltak, kétszeres CD meghajtó volt benne, beleraktunk a CD-kbe a live CD, boot-olt a live CD és ugyanúgy az X-et átváltottuk hálózatra.

**Előadó:** Az is egy megoldás. Floppy-ról is lehetne, csak azt tönkreteszik a diákok. Igazából teljesen mindegy, hogy azon a dzsunka PC-n, thin client-en, vékony kliensen, nagyon vastag kliensen, szürke dobozon, ami a diák előtt van, hogyan indítjuk el a távoli rendszert. Technikailag ez nem lehet sem probléma, sem kérdés. Mindenki maga el tudja dönteni. De talán egy szofisztikáltabb megoldás arra, mint amikor esti oktatásban, mint amikor a 35 éves kollegina beteszi a táskájába a fél kilós winchestert, felszáll a 4-6-osra és így tartja, annál rosszabb nincs.

Csináljatok egy virtualizált szerveret! Csináljatok egy szerveret, amire rácsattannak távolról, legyen rajta egy emulátor, VMWare Player, legyen Qemu. Legyen XEN supervisor, legyen akármilyen, teljesen más a történet, mert onnantól te szabad meg a lehetőségeidet és nem a lehetőségeid szabják meg az órádat.

Itt vannak a források, ennek az elkészítéséhez nem használtam mást csak a laptopomat, meg ezt a három weboldalt, két kollégával meg egy barátommal beszéltem telefonon.

# Admin szekció

## Lajber Zoltán: Mi is az intranet? (fogalmak, célok, megkötések)

Tehát hogyha valaki nem ismerné el kell mondanom hogy ős-linuxos, hogyha van a teremben aki ismeri azokat az embereket, hogy Görgő, Buci, Slapic gyakorlatilag ő is velük kezdete 100 évvel ezelőtt amikor még a linux is kisgyerek volt és azt gondolom nagyon kevés ember van az országban aki jobban ismerné ezt a rendszert mint ő. És szerencsére Most őt is Csillag Tamást megkaptuk egy egész napos előadási sorozatra. És ezt most végignézhethjük. Az első előadás az gyakorlatilag egy általános bevezető lesz, az internet és a környékbeli dolgokról. Nagyon nagy szeretettel átadom a szót az előadónak.

Köszönöm szépen a dicsérő szavakat. Azt hiszem ezek után bemutatkoznom sem kell. A legelején én is mondanék egy kis bevezetőt. Ha már itt három volt belőle vagy kettő. Miről lesz szó? Ahogy néztük a Linux az oktatásban konferencia tematikáját az előző alkalmakkal, a tűzfalazásról, internetről, a proxy-kről volt szó engem úgy kerestek meg hogy, Samba-ról tartsak. Valamiért elterjedt, hogy én nagyon értek a Samba-hoz. Ez különben nem igaz. Annyi az egész hogy az ominózus '96-os első Linux Konferencián, tényleg tartottam egy szakmai előadást, ahol még Buci, Slapic, és Madarász Gergő adtak elő, és azóta is még egypárszor, de azóta elég sok mindent csinállok, most már legkevesebbet a Samba-val foglalkozom.

Mondhatnám azt is hogy nekem a legegyszerűbb konfigurációim működnek amiket csinállok és semmi bonyolultba nem megyek bele, ezért aztán működik. Ezt a szellemiséget próbáljuk követni végig a barátommal. Kiterjesztettük ezt a Samba dolgot a belső hálózatra, tehát míg az előző Linux Konferencián az internet, tűzfal, és proxy-k és ilyen dolgok voltak és a nagyvilág felé azt mondtuk, hogy amit itt házon belül szolgáltatni kell, azokkal próbálunk meg foglalkozni. Így bevállaltuk az egész napos előadást. Mikor nekiálltunk slide-okat írni, ugye jó szokásunk szerint néhány nappal ezelőtt, akkor rájöttünk, hogy ez hihetetlen méretű téma. És az admin szekció meg elvileg egy elég részletes parancsoktól és szkriptektől sem mentes szekciónak számít. Nahát ez egy elég kemény feladata elé állított minket. Megpróbálunk ennek megfelelni. Mondhatnám azt hogy öles léptekkel fogunk haladni, viszont bármikor nyugodtan közbe lehet kérdezni. Megijesztek mindenkit, hogy bármelyik slide-unkról képesek vagyunk fejenként egy napot beszélni.

**Miről is lesz szó?** Próbáljuk meg rendszerbe önteni. Először is gondoljuk át mit is csinálunk, ilyen tervezési megfontolások. És akármennyire Linux Konferencia, szükség van foglalkozni a hálózattal, mert ma egy számítógép, hálózat nélkül semmit sem ér. Itt nagyon visszafogottan de foglalkozunk a hálózattal. A következő rész ugyanolyan eszköz mint a hálózat, és manapság nagyon divatos, ez a virtualizáció, amiről Tamás fog beszámolni, és egy általa használt megoldást bemutatni. Én mást használok, de ezt ő nyerte meg. Utána valamennyire bemutatjuk a háttér szolgáltatásokat. Tehát azokat a szolgáltatásokat, amiket közvetlenül nem használ a felhasználó, de roppant fontos, ami nélkül gyakorlatilag nem működik a hálózat, itt gondoljunk olyanra mint a DNS. Amit határeset, hogy használ-e a felhasználó vagy nem. Amikor nem működik a DNS szerver akkor a felhasználó megkérdezi, hogy ki kapcsolta ki az internetet, mert nem tud IP címek alapján weboldalakat nézni, és még jó néhány ilyen.

**A csoportmunka,** egy picit elnagyoltam ezt a részt, mert ez akkora téma és annyiféle, hogy ezt mindenkinek a saját cégigényeihez kellene illeszteni, mindenesetre mutatok egy olyan vállalati megoldást, ami kisebb cégeknél is működik (50-100 felhasználó). Utána azért nem ússzuk meg, ha már én beszélek, egy ki Samba-zást. Ígérem ez így SMB protokollokon és slide-okon lesz és előben nem. Fájlfeladat és nyomtatószerver és a végén megint lesz egy olyan, hogy lépünk vissza és ne csak a fákat lássuk hanem az erdőt is, azaz hogy helyezzük el ezeket a szolgáltatásokat, hogy csoportosítsuk és mire rakjuk. Itt fog előjönni eszközként a virtualizáció. Másrészt elindítunk egy flame war-t amit majd lehet folytatni este, olyat hogy hardver

kiválasztás. Nekem az egyik mesteremnek tekintett Buci azt mondta, ne bízz olyan számítógépbe, amit fel bírsz emelni, aminek szerintem nagy igazsága van.

**Érintünk olyan témát, amiről egy egész nap lehetne beszélni pl. a mentés.** Én Layer0-nak írtam, ugye a hálózati rétegek alatt van még egy. Ami manapság egyre fontosabb, de kevés irodalma van, mivel nem látszik annyira. Hova rakjam a gépemet, hogyan tereljem ki belőle a porcicákat, hogy kap az tápellátást, hogy tudom én azt hűteni? És még egykét dolog, de tényleg csak felvillantva a főbb problémákat. Körülbelül ennyi lesz a mai műsor. Előljáróban még annyit hogy, az hogy Linux szekció, megadja azt az irányt, hogy itt Linuxról lesz szó, és a Linuxon futó különböző alkalmazásokról. Mi az hogy open source és free software, remélem ezzel mindenki tisztában van. Itt elhangzanak különböző szoftver nevek config fájl részletek. Ez messze nem azt jelenti, hogy ez az egyetlen üdvözítő vagy legjobb megoldás. Az open source közösségben tipikus, hogy egy feladat megoldására több program áll rendelkezésre, és ezek közül választani kell. Az azonban biztos, hogy mi ezt választottuk valamilyen oknál fogva, és nekünk működik. Olyan szoftvert nem fogunk mondani, ami nem vált be. Biztos van még ezenkívül jó néhány amivel ezeket meg lehet oldani. Ennyi lett volna a bevezető.

**Alapszolgáltatások, és intranet.** Milyen szolgáltatásokra van szükség? A hálózatunkat nézzük át, először teljesítményre, másrészt meg biztonságra, hogy hogyan szervezzük a hálózatunkat. Azt hogy az intranet szolgáltatásokba ki mit ért, az mindenkinek a saját dolga. Sok mindent lehet belevenni, de én azt mondom ezeket a szolgáltatásokat bele fogjuk venni: elektronikus levelezés az kell, valamilyen fájlmegeosztás is, nyomtani is kell. A nyomtatás az mindig kritikus rész a számítástechnikában. Mivel a nyomtatás pénzbe kerül ezért jó lenne nyilván tartani, ez igaz a hálózati sávzsélességre is, főleg ha kifelé elég szűkös, akkor azt mondja a főnök, hogy ezt próbáljuk megoldani. Érdekes tanulság, hogy az a tudat hogy nyilvántartják a sávzsélességet, már elég, hogy csökkenjen a felhasználás. Kipróbáltuk több cégnél is. Bevezettük a hálózati naplózást, számoltuk a forgalmat majd három-négy hét múlva közöltük a felhasználóval, hogy na, most már mérjük a hálózati forgalmat és hirtelen egy 30 %-kal csökkent. Ez nyomtatásban is hasonló, ha közöljük, hogy tudjuk ki mennyit nyomtat, ez önmagában képes redukálni a nyomtatott mennyiséget. Nyilván egy teljesen más kategória a számlázás szintű nyilvántartás akár nyomtatásról akár hálózatról van szó. Azért ez bonyolultabb és ezt óvatosan kezelném, számlát kell nyomtatni, és mérni a nyomtatott lapok számát.

**A WiFi-ről túl sokat nem mondanék, egy-két slide-on majd oldalt látszik.** A legnagyobb divat ma, hogy nem kell kábeleznünk, mert van vezeték nélküli hálózat. Szerintem – mint hálózati szakember szerint – ez egy nagy tévedés. A vezeték nélküli hálózat egy nagyon kényelmes szolgáltatás, de nem ugyanaz mint a vezetékes, és nem is szabad keverni vele. A WiFi kilóg az épületből, és nem tudjuk kontrollálni, legfeljebb nehezíteni a dolgát a kint ülőnek. Tapasztalatok szerint nem igazán nehezítik a kint ülőknek az illetéktelen hozzáférést. Elég sok intézmény van, ahol nem csak az ott dolgozók, és az ottani emberek használnák ezeket a szolgáltatásokat, hanem jönnek vendégek, esetleg tanfolyamra jönnek, akár látogató, akár konferenciát rendeznek, akkor önkik milyen szolgáltatás kell. Nagyon nagy valószínűséggel, nem ugyanaz mint az ott dolgozóknak, és alapvetően téves koncepció őket ugyanabba a hálózatba rakni mit az ott dolgozókat, mert ennek elég komoly mellékhatásai lehetnek. Főleg, ha ezek szervertelepítést tanulnak, és csinálnak néhány DHCP szervert a hálózatba.

**Lehet tovább ragozni a szolgáltatásokat.** Lehet olyan eset is, amikor valaki otthonról akar dolgozni. Vagy olyan munkatársak akik járják az országot, vagy még messzebb, és onnan is el akarnak érni bizonyos dolgokat. Ezt hogyan tehetik meg? Erre az egyik megoldás lehet a virtuális magánhálózat.

**Lehet olyan is mint címlisták.** Ez lehet olyan is, hogy a cégen belül ki tudom keresni az e-mail címzetteket. Nyilván ez egy kis cégnél nem szempont, de mondjuk egy 20.000 diákos és 4000 tanárból álló egyetemen ez már nem triviális feladat, és akkor ennek már van értelme. Nem beszélve arról egy cégnél, hogy ügyfél nyilvántartás, és az ehhez köthető címlista. Lehet olyan is mint a tennivalólista. Mi praktikusnak vettük ezt a távoli fájllelérést, majd amikor a mail szerverről beszélünk akkor erre kitérnék.

**Naptár.** Van olyan cég ahol kimondottan kérik, és teljesen jól használják. Jól látható, hogy elfoglalt vagyok, főnök mikor mit tervez, terembeosztást lehet készíteni. De van olyan is ahol ott a naptár szolgáltatás és a cégen belül a munkatársak 5%-a sem használja. Ezt is lehet a végtelenségig bonyolítani, mikor az „okostelefon” próbál szinkronizálni a „buta” naptárhoz.

**Nem kellene elfelejteni a régi jól bevált levlistákat,** mint szolgáltatásokat, és mellé különböző alkalmazásokat. Rendszergazda szempontból jól lenne azt csinálni, hogy a kritikus alkalmazások nem a felhasználó gépén futnak, mert az egy megbízhatatlan környezet – például fel bírom emelni a gépét –, hanem valami központi helyen, ami jobban kontrollálható környezet. És akkor jön az alkalmazásszerver amiről gyakorlatilag semmit nem fogunk beszélni, hogy ez hogy lesz. Mivel attól függ milyen alkalmazás milyen alkalmazásszerveren, milyen platformon, és ezt így nem vállaljuk.

**Foglalkozunk a hálózattervezéssel.** Három szempontról van igazán szó a hálózattervezésnél: teljesítmény, biztonság és karbantarthatóság. Ezek ellentmondó követelmények is lehetnek, bár segítik is egymást időnként. Nyilván a teljesítmény és biztonság ellentmondó követelmény. A klasszikus idézet szerint: „a hálózat biztonságát a használhatatlanságig lehet fokozni”. Ennek a másik extrém példája

Hogy tudom védeni a számítógépet?

- Tegyéél rá vírusirtót és tűzfalat.
- Ez nekem túl bonyolult. Mít tegyek?
- Ne kapcsold be.

Beszéljünk egy kicsit a hálózati technológiákról, mert ezek rohamosan fejlődnek, és időnként a Linux listákon is azt veszem észre, hogy nagyon bátor rendszergazdák faragnak elég nagy rendszereket, és közben alapfogalmakkal nincsenek tisztában. A kiszolgáló gépek teljesítménye a hálózat felé. Ez egy nagyobb téma, ezért a legvégére tettük. A biztonságról, csak egy nagy kép van, amiről lehet beszélni, de trükkök nem lesznek. Annyiban van köze egymáshoz a két dolognak, hogy egy jó hálózat nem bonyolult, emiatt jó a karbantarthatósága, ugyanakkor biztonságos is. A bonyolult hálózat általában nem jó. És ez mindenféle megoldásra igaz, hogy próbáljuk egyszerűen tartani.

**Az első dolog, meg kellene határozni mekkora forgalom lesz a hálózatunkon.** Az irodalomból össze lehet szedni I/O művelet per másodperc, paramétereket. Azért az hozzá kell tenni, hogy a hálózati paraméterek meghatározására az angol kifejezés nagyon sok mindent elmond. Ezt úgy hívják ezt a procedúrát, hogy „educated guess”. Ami azt jelenti, hogy képzéssel elősegített kitalálás. Elég bátor hálózatteljesítményt jósolni. Itt nagy szórás van. Mít is jelent az átlagos irodai hálózat?

- Ez a tipikus titkárnő aki időnként ír egy levelet, ami nem okoz nagy terhelést a hálózaton.
- Az erős irodai az egy advanced user, aki már használ dolgokat, mint például: Nexon bér, egyetemeken Neptun, SAP, tehát ilyen kliens szerver, könyvel, gépel ezerrel.
- A fejlesztő itt azt jelenti, hogy ő programoz és fordít, de ugyanez tartozik aki CAD-ezik. Nekik semmi nem elég, tehát ide kellene a legerősebb gépek, a leggyorsabb hálózatok.

**Mi lehet a szűk keresztmetszet?** Hálózati technológiáknál meg szoktuk nézni a címkét, hogy hány megabitet tud? Felhívnám arra az egyszerű tényre a figyelmet, hogy a half duplexes hálózat – például a régi koax kábel – vagy a mostani UTP, ha ráraok egy HUB-ot – ami hivatalból half duplexet tud csak –, az jó esetben fele akkora sávszélességet tud csak mint a switch-el hálózat. Ez igaz még kevés gép esetén is. Ha pedig már húsz gépnél is több van, akkor már sokkal rosszabb a helyzet. Itt nem is mentünk gigabit Ethernet fölé, mivel ott már nem olyan egyszerű az élet. Ott már bejönnek olyan dolgok, hogy kevés az a számítógép amelyik pl. a 10 Gbit-et ki tudná tölteni. A switch-eknek, vagy a hálózati eszközöknek, van egy olyan, kategória hogy a so-ho (small office-home office) ahol ne legyenek illúzióink. A switch-eknek van egy nagyon fontos paramétere a hátlap, vagy backplane teljesítménye. Normál esetben ezt megadják gigabitben, hogy mekkora adatokat lehet átküldeni rajta, és korrekt esetben ezt még megadják akár frame/sec, (azaz hány keretet tud átküldeni) formában is. A kettő az két különböző dolog. Normál esetben, ha én fájlokat mozgatok két dolog között, akkor 1500 byte-os frame-ekkel dolgozom, ilyenkor valószínűleg a sávszélesség fog limitet okozni. Viszont ha bejön egy támadás, egy vírus ami elszabadul a hálózatba, az tipikusan 64 byte-os frame-ekkel fog dolgozni, akkor majd a másik paraméter jelenti majd a szűk keresztmetszetet. Ahol már láttunk olyat a switch dokumentációkban, hogy a backplain teljesítménye ennyi Gbit, és ennyi frame/sec-et tud továbbítani, az már akár használható is lehet. Az árákról, pedig ez a táblázat jól mutatja, hogy tudja egy műszaki szakember elsütni a főnökének, hogy neki igen is egy 70.000\$-os CISCO 6500-ra van szüksége. Hisz az a legolcsóbb. Tehát a diagramok azért vannak a mérnökök kezében, hogy a megfelelő helyeken alkalmazzák őket. Mindig attól függ, hogy mi a cél. Ha van egy olyan cégem ahova kell több száz Gbit-es port és ezt ki is kell szolgálni akkor olcsóbb lesz a dolog ezzel a nagy dobozzal,

mintha vennék három szobányi kicsit, és azokban hibát keresnék, mivel néhány biztosan rossz bennük, és nem tudom melyik, mivel nem menedzselhető.

**Kábelezés kiválasztása.** Manapság ebben nagy kihívás nincs. Ez a kategória 5-ös csavart érpáros kábel, ez elterjedt és használható, van egy olyan változata, hogy CAT 5e aminek a felépítése ugyanolyan mint a CAT 5-é, csak a szerelési utasításban tér el egy picit. Mekkora kanyarban hajlítható, mekkora kilogrammally nyújtható, de ez már Gigabitet is tud 100 méterre. Ennél bonyolultabb nem kell. Sajnos jelent meg olyan pályázat Magyarországon, hogy ha volt Cat 5, Cat 6, akkor legyen Cat 7. Amivel nincs is gond mivel brutálisan jó, mivel érpáronként külön árnyékolt, és olyan csatlakozója van ami nem RJ-45. Gyakorlatilag a csatlakozói annyiba kerülnek, hogy ugyanannyi pénzből optikát is húzhatunk. A Cat 7 nagyon vastag, sokkal nagyobb kábelcsatornák kellene, drága a csatlakozó, sőt nem is tudok rá rendesen csatlakozni a mai eszközeimmel. Szóval kell egy átalakító, és visszajutottunk oda, hogy lesz egy Cat 6-os kábelszakasz.

**Az optika nagyon elterjedt, nyilván pénzügyi okból.** Épületek közt sokszor látni rézkábeles átvezetést, ami elég kamikaze megoldás. Lehet épületek között korrekttül is átvinni rézkábelen jelet, csak vannak nagyon kemény feltételei, és ha ezeknek megfelelőnk akkor olcsóbb ha optikával csináljuk. Egy példa: az elektromos hálózatokat nagyon sokszor túlterheljük már manapság, ha a két épület külön földelésen van, és néhány kilométeren belül becsap egy villám, akkor a föld potenciálban lényeges különbség alakul ki a két épület között. Elkezd áram szaladni a két épület között, és ezt a hálózati eszközök nem tolerálják. Az hogy lecsapott egy villám és egy-két port meghal elég gyakori jelenség és erre figyelni kellene. Ezért épületek közt célszerűbb optikát húzni. Ez ügyben elég sok cég felbukkan, aki nem megfelelően tájékoztatja a felhasználóját, hogy mit is húzzon és mennyit. Most itt a különböző optikai szabványokról van szó, az látszik, hogy a rezes az tud 100 métert, az optikából többféle is van. A 62,5 mikronos legfőljebb 270 méterig képes elmenni, ami épületek közt nem sok, hamar el lehet érni. Az 50 mikronossal legfőljebb 550 méterig.

Ez nagyon lényeges különbség, úgy hogy árban nincs is. A 62,5-es az a régebbi kábel az 50-es az újabb. Ha valamelyik cégnek van régebbi, azt megpróbálja eladni a felhasználónak. De előbb-utóbb felhasználó ráfut arra, hogy a 100 Mbiten jó a 62,5-es mert tud 2 kilométert, de Gigabiten csak 220 métert, ami az épületen belül nem elég. Lehet még tovább ragozni a 70-100 kilométerre jó ZX-es de ott az eszköz az 6000\$-ba kerül.

**A másik kérdés amit meg szoktak kérdezni a rendszergazdától, hogy mennyi a WiFi hatósugara?**

Amire a helyes válasz, attól függ, és mérni kell. Fontos az, hogy milyen antennával van, milyen irányban mennyit szór, milyen épületen van. Ha van egy vizesblokkom tele csövekkel, és ez 10 méterre van az access pointtól, akkor az 10 métert fog szórni abba az irányba, és nem többet. Ez az egyik oldala. A másik, ha van egy Salgó polcos raktár, ahol nagyon szépen meg tud pattanni a jel (a jelerősséggel nincs probléma a raktár mögött), akkor annyi irányból és annyi félét hall, hogy a jel-zaj arány annyira elromlik, hogy közel használhatatlan lesz. Tehát WiFi telepítés előtt, hogy hány access point és hány antenna kell azt először is mérni kell. Ki kell rakni az access pontot a gyanús helyre, és célműszerrel – lehet laptop is – körbesétálni és megnézni mi van.

**A másik probléma manapság a vezeték nélküli hálózatokkal, hogy óriási nagy a zaj.** Mindenki csinálja és behallatszik. Ez ügyben pedig a legnagyobb galádság az, hogy még mielőtt szabvány lenne, elkezdtek a gyártók forgalmazni n-es eszközöket. Nem olvassák el általában a felhasználók, hogy minden gyártó odairja az n-es eszközére, hogy ez egy pre-standard, és nem fogadták el még szabványnak, és csúszik az elfogadása, mert elég komoly problémák vannak vele. De ha a szomszédnak van egy a/b/g-s access pontja, és nem vagyunk vele jóba akkor, vegyünk egy n-es és kapcsoljuk be. Ő nem fogja használni az a/b/g eszközét. Magyarországon a Wifi-nél ma van tizenhárom használható csatorna. Ezek közel vannak egymáshoz, és három van ami nem fedi el egymást, a 0-ás, 6-os, 13-as. Az n-es meg ezek közül néhányat azonnal bezajol. És ha valamelyik access pont dolgozik például a hatoson, és mellette nem messze egy a hetesen, akkor egymást csak zajként fogják értelmezni és nem normál forgalomként, és nem tudnak megegyezni, hogy osszák meg a médiát – a levegőt –, egymás között. Ezért elég óvatosan kell ezeket az eszközöket telepíteni, mivel erre is figyelni kell.

Ráadásul az ilyen access pont telepítésnél kettő követelmény van: normál esetben (pl. iskola), az a cél, hogy minél kevesebb access ponttal minél nagyobb területet lefedjen. Aztán jön egy konferencia, ahol megjelenik egy teremben ötven ember. A levegő viszont ugyanúgy osztott média, mint a régi koax kábel, és ott is húsz gép fölött a hálózaton érdekes jelenségek kezdtek megjelenni, ugyanez a helyzet itt is. Egy access point egy

helyen egyszerre 20 embert bír kiszolgálni. Ha van egy konferenciám ahol egyszerre ötvenen vannak, akkor két access pontot kell leraknom. Nyilván másik csatornára, hogy ki lehessen ezt szolgálni. Felhívom a figyelmet az „a” csatorna praktikusságára, ugyanis tapasztalat szerint a felhasználók 10-15%-a használja az „a” csatornát. Ezért ha konferenciára megyünk, célszerű átváltani az a csatornára mert a „b” csatornán ott megy a tolongás. Ráadásul a „b” csatorna hiába 54 Mbit-es, azt csinálják az access pontok, hogyha fellép egy régi „b”-s kliens akkor mindenki 11 Mbit-en forgalmaz. Ha meg mégis 54 Mbitet ír is ki, akkor is meg kell várni az időzítéseket amikor a 11 Mbit-es időt kap, amitől a hatékonysága ugyancsak romlik. Az 54 Mbit is elég erősen optimista dolog, és az ennél magasabbakat ígérőket, elég kétkedéssel kel fogadni.

**Még egy technika, amit nem nagyon használnak, és nagyon praktikus:** Virtuális LAN, amit VLAN-nak szokás rövidíteni. Viszont a vezeték nélküli hálózatot WLAN-nak hívják. Ez hangzásában a magyar nyelvben ugyanaz. Először is megnézzük hogyan működik egy switch, hogyan működik egy VLAN-t tudó switch, aztán foglalkozunk a trunking-gel, és azzal hogyan lehet ezeket a VLAN-okat a Linuxon konfigurálni.

**A hálózati switch abban tér el a HUB-tól,** hogy a HUB az egy Layer 1-es eszköz, biteket másol jobbra-balra, és ennél magasabb struktúrájú adatot nem ismer, ami egyben az összes előnye és hátránya. A switch az Layer 2-es működésű ahol már vannak frame-ek, azaz keretek, és a kereteknek van forrása (feladója) és célcíme is. Az adat már valamilyen borítékban szerepel. A switch értelmezi ezeket a frame-eket, ő nem biteket másol jobbra-balra. Az egyik lábán bejön egy frame, azt ő értelmezi, és ha minden rendben van akkor továbbítja. Az első lényeges dolog: ha minden rendben van. A switch-ek többsége lefuttat egy FCS-t (Frame Control Sequence) ami megnézi, hogy a frame megfelel-e az előírásoknak. Van-e fejléce, a fejléce olyan hosszú-e, a végén van egy CRC, kiszámolja a CRC-t ha minden jó akkor foglalkozik vele, ha nem akkor meg eldobja mondván, hogy ez nem frame csak egy frame-nek látszó valami. Ez okoz néha olyan galibát, hogyha fogom a SOHO kategóriájú hálózati eszközt ami működik a hálózaton. Berakok egy rendes eszközt, ahol az FCS már a hardverbe az Ethernet chipben lejátszódik, és egyszer csak eltűnnek gépek a hálózatból. Klasszikusan ilyen tud produkálni a Realtek 8029-es hálózati kártya. Ez még egy combo BNC/UTP csatlakozós. Ezen a hardver minimális, és ezt a szoftvere csinálja, a windowsos drivere pedig hibás. Van egy olyan előírás az Ethernetben, hogyha nincs is annyi közlendő adatom akkor is 64 byte-ig fel kell bővítenem a frame-et, Na, ezt nem teszi meg a Realtek. És ha olyan switch-em van ami nem ellenőrzi a frame-et, akkor továbbviszi a 48 byte-os frame-et, ha jó switch-em van akkor az meg kiszűri. Akkor van az, hogy egy noname switch-csel működik a hálózat, és ha berakok egy 40 ezer forintos Cisco-t akkor meg tűnnek el gépek a hálózatból, és azzal nem működik. Ez az egyik oka. A rossz hír az, hogy ennek a chipnek a következője a 8139-es is ezt csinálja, a jó hír az, hogy csak a Microsoft driverrel, de bármilyen D-link megjavítja a működését, a linuxos driver az viszont jól működik. Ott tartunk, hogy már viszonylag épkezláb frame-ek közlekednek. A switch annyiból jó, hogyha meghülyül egy hálókártyám és elkezd hibás frame-eket, vagy esetleg van egy hibás kábelem, és mennek a hibák, akkor csak arra a kis szegmensre vonatkozik. Gyakorlatilag a switch és az alatt lévő ütközés tartomány, ami jó esetben egy gép, vagy esetleg egy HUB-bal továbbosztott hálózat, de ebből a hibák már nem tudnak kiterjedni. Ez mindenképp a switch előnye, és javítja a hálózat teljesítményét. A másik az, hogy a switch kialakít a hálózatról egy elképzelést ki merre van. Amikor bekapcsoljuk akkor üres a feje, ha jön egy frame valahonnan, akkor felírja hogy bejött a frame az 1-es lábán, a földönak is felírja a címét, az 1-es lábán, ugye ez a cím, a kártya MAC address-e az egyes lábán látható. Azt nem tudja, hogy a cél cím merre van, tehát kiküldi az összes többi lábán, egy elárasztásos algoritmussal. Tehát, ha az egyes bejött egy frame, és nem tudja merre kell küldeni, akkor kiküldi a maradék 23 lábán. Jó esetben valamelyik számítógép válaszol, abban a forráscímében már az előző cél cím szerepel, és akkor felírja, hogy ezt itt van a 8-as lábán. Ha ez a két gép tovább beszélget, akkor a frame-ek bejönnek az egyes lábán és kimennek a 8-as lábán, és a maradék 22 láb erről mit sem tud jó esetben. Ez hálózatbiztonság szempontjából is jó – de nyilván ismerünk támadási módokat ahol ezt ki lehet trükközni –, és teljesítmény szempontjából is. Az ilyen switchek automatikusan működnek, semmit nem kell konfigurálnom, az egész jelenség automatikusan működik. Nyilván jönnek majd elő ennél bonyolultabb problémák, például, ha kihúzom a gépet akkor eltűnnek a címek, ha bedugom egy másik helyre akkor újra meg kell tanulni a címeket, stb.

Régebben volt kétfajta switch-elési mód ami 10 Gbiten visszajött, a technikai spirál környékén. A mai switchek a store and forward (tárol és elküld) elven működnek. Ez azt jelenti, hogy bejön a frame azt ellenőrzi és küldi tovább. Ez nyilván késleltetés a rendszerben, ki kel számolni, hogy a 100 Mbiten az 1500 byte-os frame-nek be kell jönni, majd ha megnézi ki kell sorjázni, és ez egy késleltetés a rendszerben. A másik módszer az a cut through. Ez annyit csinál, hogy megvárja az első 64 byte-ot, ebben már benne van

a fejléc információ. Tehát el tudja dönteni, hogy hol fog kimenni, meg sem várja a frame végét, hanem elkezd küldeni a másik lábon. Régi 10 Mbit-es switch-eknél lehetett ezt állítani, főleg ilyen token ring okokból. Viszont a 10 Gigabites data center a higher switch-eknél visszajött, ugyanis 10 Gigabittól a késleltetés már számít.

**Jön a VLAN switch működése.** Hogyan tudunk virtuális LAN-okat csinálni. Ha veszünk egy VLAN switch-et és berakjuk a hálózatunkba, akkor ugyanúgy működik mint egy normális switch, csak többbe kerül. Viszont létre lehet hozni rajtuk virtuális LAN-okat, valami úton-módon, soros konzolon, telnet-en. El kell olvasni az eszköz elírását, hogy tudunk beszélgetni az eszközzel, legrosszabb esetben egy webes felületen keresztül. Ha nem ismerem egy eszközt, bemegyek egy webes felületre, és meg tudom nézni, hogy mit tud, akkor az jó. Az egér mozgást és kattintást elég nehéz lesz szkriptezni, és ha sok eszközöm van, akkor elég macerás lesz egy idő után. A másik hogy a legtöbb eszköznél nem lehet elmenteni a beállításokat egy fájlba, ami azért fontos, mert ha konfiguráltam egy eszközt, akkor fontos lenne lementeni a konfigurációját. Mert ha esetleg elromlik, akkor veszek egy hasonló eszközt, rátöltöm a konfigurációt és akkor már tudom is használni, és nem kell végigkattintgatni fél órán keresztül a beállításokat. Létrehozunk VLAN-okat, ezeknek adunk neveket esetleg számokat. Aztán a különböző switch lábakkal megmondjuk, hogy melyik VLAN-ba tartozik. Az egy VLAN-ba lévő gépek látják egymást, a különbözőkben lévők viszont nem. Ezt úgy képzeljük el, mintha virtuális switch-eket csinálnánk. Az 1-es VLAN az egy switch, a 2-es VLAN még egy switch,

de semmi köze nincs a VLAN-oknak egymáshoz, a frame sem megy át. Ahhoz hogy át tudjunk menni az egyik VLAN-ból a másikba egy router kell nekünk. Ezzel a módszerrel szét tudjuk szegmentálni a felhasználókat. Azt, hogy a portot hogy tesszük be egy VLAN-ba, az klasszikusan az hogy, odamegyünk az eszközhöz, és megmondjuk, hogy ez a lába a VLAN. De mára már szabványosított. Ami a vezeték nélkül felbukkant 802.1x az használható vezetékiesen is. Amikor elindul az eszköz kapok egy ablakot, user name /password, a Radius server megmondja a switch-nek, hogy melyik VLAN-ba mehet a felhasználó és kész.

Így néz ki egy hagyományos hálózat. Vannak osztályok, switch-ek, és egy routerrel összekötöm őket, amivel nyilván szűrni is tudok, hogy ki mit érhet el. Ha VLAN switch-ük van akkor egy switch-be össze tudom őket szedni, mégis szét lesznek választva. Most még három kapcsolatunk megy a router felé. Ez viszont nem az ideális, mert amíg még csak három VLAN-unk van akkor még jó, de ha van hetven VLAN, akkor az már nem jó, mert elvisz egy csomó portot, azért, hogy a VLAN-okat összekössem. Ezért nyilván nem ez a jó módszer.

**Úgynevezett trunk fogalma következik.** Kijelölünk speciális csatlakozásokat, az úgynevezett trunk-ök. A megoldás az, hogy VLAN-onként használunk egy uplinket ahol nem skálázódik a probléma normálisan. Vagy úgy teszem rá a frame-eket erre a kábelre, hogy színezem. Megjelölöm a frame-eket, kap egy piros címkét az VLAN1-es, zöld címkét a VLAN2-es, stb. Itt az a baj hogy a technológia a szabvány előtt jár.

Tehát van csövünk, amibe színes golyócskákat engedünk, azaz trunkol. Aztán jelentkeznek apró problémák, de már a Linuxon is megoldották. A szabvány keretek 1518 byte hosszúak. A 802.1q is 4 byte-tal hosszabbak. Ha buta switch-ünk van és nem végez keretellenőrzést, akkor továbbítja a trunk vonalat. Ha félig okos switch-ünk van és ellenőrzi a frame-eket, de nem ismeri a trunköt, akkor az ilyeneket el fogja dobálni, mert hogy az túl hosszú. Ez tipikus probléma, mert a rendszergazdánál, a pingetés megy 64 byte-tal de, jön a felhasználó, hogy nem tudja elolvasni a levelét, amihez egy nagy képet csatoltak, ami már nem jön le az útvonalon. Ez ügyben a leggaládabb eszköz az egy média konverter, állandóan debugoltunk, és másnap elővettük a felhasználói kézikönyvét. És feltűnt az a rész, hogy switching media konverter. A régi időkben a CISCO minden egyes VLAN-ba spanning tree példányt futtatott, a konkurencia meg nem, mivel azok csak egyet futtattak. A spanning tree a Layer 2-be kialakít egy elképzelést a hálózatról, és megszünteti a hurkokat. Ezért, ha volt egy CISCO-s hálózatom, és középre beraktam egy jól irányzott routert, akkor vagy látott vagy nem látott a hálózat másik fele, majd egy idő után ez váltott, hogy ki lát és ki nem. Ráadásul a CISCO kitalált egy VLAN trunking nevű protokolt, ami azt jelentette, hogy egy switch-en beállítom, és a többnek megmondja hogy nálam mi van, azaz mi a prioritása, hogy hívják, stb. A többiek, szintén futtattak egy spanning tree meg egy úgynevezett bar protokollt, ami butább mint a VTP, mert nincs a hálózatoknak neve. Ez úgy működik, hogy ha azt mondom hogy trunk, és megbízok a kapcsolatban, és a switch látja hogy jön a hatos VLAN-ba is frame, akkor fölveszi a saját fejébe a hatos VLAN. Ez így teljesen jól hangzik, mert megint nem kell konfigurálni, és terjed tovább az információ, viszont alapértelmezetten az ilyen switch-eknek az összes lába tanul VLAN-t, a másik, hogy a VLAN ID a keretben az 12 bites, és 4096 lehetséges értéke van, de a régebbi eszközök, csak 1024-et támogattak. De meg kell nézni a kézikönyvben, hogy hány VLAN támogatott, mert ez két különböző fogalom. Gyakori, hogy az olcsó eszközökbe, 64 VLAN

támogatott. Ami azt eredményezi, hogy a 65. esetén egyet eldob a régiek közül, és ezt az újat fölveszi. Remélem felismertétek a támadás lehetőségét. Bekapcsolok egy ilyen eszközt, és a rendszergazda rádug egy Linuxot, és kezdi konfigurálni a VLAN-okat, amikor megcsinálja a 65.-et akkor a switch eldob egyet. Manapság ez úgy változott, hogy van a multi spanning tree. Ez úgy van hogy van M spanning tree az N darab VLAN-hoz, ahol  $M < N$ , Elindítunk három spanning tree-t, és megmondjuk, hogy a 0-100 az egyesbe, a 101-200 a kettesbe stb. Ez praktikus és jó bizonyos szempontból. Ez annyiban lényeges, hogyha van két kapcsolatunk eddig fölfele, akkor az egyiket shutdown-ba tette a spanning tree, és a másikon ment. Ott volt a 2 Gbit-es linkem, de csak az egyiket tudtam forgalmazni. Ha átállítom a súlyokat ennél a multi spanning tree és azt mondtam, hogyha a 10 alattiak ezt a VLAN-t használják és a másikat teszi shutdown-ba, a 100-200 fordítva. Ha van két linkem akkor ki tudom használni a két linket, és ha szakadás van akkor is csak egy kisebb dugulás keletkezik.

A klasszikus kialakítás az, hogy a szervezeti felépítés az eltér a kábelezéstől. A kábelezést úgy csinálom, ahogy az épület diktálja. A szervezeti felépítés meg nem így néz ki. A másik az, hogy a kábelezés 10-20 évre készül, a szervezeti struktúra, még ennyi hónapot sem szokott kibírni, és akkor ezt VLAN-nal lehet követni.

**Nézzük Linuxon hogyan lehet ezt konfigurálni.** Nyilván a Linux támogatja, a 2.4 kernelnél az volt a probléma, hogy a gagyi kártyákon semmit nem kellett trükközni, a jobb hálózati kártyákon, ami már hardverben ellenőrizte a frame-eket, a régi kernelben még egy patchet kellett rakni, hogy engedélyezze az oversized frame-eket. Az hogy Linuxon hogy nézzenek ki a VLAN interface-ek az változtatható. Van egy VLAN nevezetű csomag Debianban, és vconfig utility, és meg lehet adni milyen legyen a nevezék. Általában ezzel nem kell foglalkoznunk. Igazából kettőnek látom értelmét, pl. VLAN5, így hívják az interface-t. Például, ha van egy szerverem, amiről lóg néhány gép akkor ez teljesen kellemes és jó. A routereken praktikusabb az interface+VLAN. Linuxon most már állítható, hogy az 1-es ethernet 5-ös VLAN-ja azonos-e a 2-es ethernet 5-ös VLAN-jával, ami nem biztos, attól függ hogy van a routeren, de ez konfigurálható. A szervereken egyszerűbb ez. Nem keverendő ez azzal hogy IP aliasing-gal amikor eth.0: pl. az iptables szempontjából ez nem külön interface. Debianban az eth0 az a nothing VLAN. A trunkon egy VLAN-t ki lehet választani nothing VLAN-nak, ami jelölés és színezés nélkül megy. Nagyon lényeges, hogy a kétoldali berendezés egyezzen ebben, mert ha nem, akkor át lehet tekerni az egyik VLAN forgalmát a másikba. Látszik, hogy az eth0 az a nothing VLAN. A következőnél annyit kell a Debianban megadni, VLAN.20, ami a 20-as VLAN, illik megadni azt, hogy VLAN, road device eth0, azaz, ez az eth0 belül lesz majd, ezt mindenféle setup-ból. Ebből be is fogja konfigurálni a vconfig-ból a nevezést. Amit még érdemes megadni pre-up-ban, ez az ifconfig eth0. Ugyanis, ha én valamilyen oknál fogva lekapcsolom az eth0-ról az IP címet, mert nekem nem kell, akkor lehet nem jön át az eth0 interface-en. Hiába konfigurálom be az eth0.20-at, nem fog működni, mert még az eth0 nincs up-ban. Tehát, ha ez kihagyom, és csak normál VLAN húzok, és nincs nothing VLAN-om, akkor van szükség erre a parancsra, hogy tegyék up-ba az interface-t.

**Biztonságpolitikát kellene írni,** a nem számítógépes problémákat (pornó, játék, video) ne próbáljuk megoldani tűzfalal, mert nem a tűzfal dolga. Az alapfogalmakról van egy jó könyv akkor Elizabeth D. Zwicky és társaitól a „Building Internet Firewalls” az O'Reilly-től ami nagyon jó. Tűzfal elrendezésekről nem beszélek. A klasszikus az, hogy van egy demilitarizált zónának (DMZ) nevezett hálózatunk, ami kifelé szolgáltat. Akkor kifelé és befelé külön tűzfalal szűrjük. Itt van amit lehet összevonni és van amit nem. Azzal, hogy VLAN-jaink vannak szétválak a fizikai és logikai elrendezés, ezért duplán kell rajzolgatnunk. Ez egy kisebb cégnél teljesen jól működik. Tipikusan van egy tűzfalgépünk kinn az internet fele, van egy szerverünk. Nagyon sokszor elég egyetlen szerver is, kicsi a hálózat. Ha nagy, akkor csinálhatunk egy nagyobbat, amit switch-ekkel kötünk össze, ami szolgáltat kifelé. Házon belül a költségek miatt is veszünk egy olyan switch-et ami tud VLAN-t, mondjuk van Gbit-es interface, erre rátesszük a szervert, vagy szervereinket, és ha szerencsénk van, akkor itt már olyan switch-eket, amin az egyiket a tanárok, laborok vannak. Még szerencsésebb, ha van egy laborunk 20 géppel, és van benn egy VLAN képes switch-ünk, mert akkor a tanári gépet tudjuk külön VLAN-ba rakni, mint ahol a diákok vannak. És így nagyon jól lehet osztani. Nyilván a költségek megnőnek, mivel itt már minden switchnek „okos switchnek” kell lennie.

**Itt egy kis elrettentés.** Azt szokták, hogy kifelé mindenki jól tűzfalaz, házon belül pedig nem. Pedig a támadások 70%, ami még optimista becslés, házon belülről érkeznek. Azt vegyük észre, hogy a hallgatói laborokat igenis tegyük más VLAN-ba, mint a könyvelést. Nyilván a hallgatókat, nem érdeklik a könyvelések, de ha bekapunk egy vírust és nem tudunk jelentést küldeni akkor az baj. Ebből következik,

hogy házon belül is szegmentáljunk. Ez vonatkozik a vezeték nélküli hálózatra is. Mindig jön a kérdés, hogy a user nyomtatni akar vezeték nélküli eszközről, és akkor megkérdezzük tőle, hogy az autóban lévő is nyomtatni akar? Itt a vezetői szándék a lényeg. Mert ha a vezető azt, mondja, hogy igen értem a problémát, és a vezeték nélküli nem azonos a vezetékessel, akkor meg tudja csinálni a rendszergazda. Ha a vezető ezt a problémát nem érti, akkor ez az egyéni érdekérvényesítő képességtől függ.

## Csillag Tamás: Virtualizáció

Itt is nagyon sok mindenről fogok beszélni. Először is szeretném elmondani mit is érdemes elkerülni, utána néhány alapfogalomról lesz szó. Utána folytatjuk, hogy miért is jó a virtualizáció. Több virtualizációs megoldást fogok bemutatni, viszont utána az OpenVZ-nél fogok kikötni, mert ezt nagyon jól lehet használni. Be fogok mutatni egy működő konfigurációt. Meg fogom mutatni, hogy ez egy könnyen kezelhető felülettel, egy menedzselhető felülettel appliance-szel, hogyan működtethető. És végén a biztonságról is ejtünk néhány szót.

Nekem volt ilyen szerverhez szerencsém. Úgy indul, hogy felrak az ember egy SMTP szervert, IMAP szervert, mert kell levelezni. Legyen rajta webmail is, mert ha már ott vagyunk milyen kényelmes, hogy nem kell mindenkinek külön klienst telepíteni. Legyen egy webmail, webes felületen. Akkor már a felhasználóknak is legyen PHP szerver, saját kis könyvtárak, weboldalak, meg szkriptek hadd fussanak. De akkor ezeknek a szkripteknek kell egy adatbázis is. Valaki PostgreSQL, valaki MySQL-t, valaki mind a kettőt. És akkor lehet látni, hogy szépen fejlődik a dolog. Ha az ember úgy áll neki, hogy van egy szerverem, elkezdjük szépen felrakni, az ember végigjár ezen az úton, amíg rájön, hogy jön a user hogy frissítsünk a PHP-n mert neki már PHP 5 kell. Az ember megfrissíti a PHP-t és akkor borul a webmail. És lehetne ezt fokozni. A lényeg az, hogy egy annyira összefüggő dolog jön össze, hogy bármihez hozzányúlunk, valami borulni fog. Mert bármilyen kis változtatás is nagy kihatással van rendszerre, és nem lehet upgrade-et tervezni. A másik az, hogy egy szerver, van és ha borul, akkor az összes szolgáltatás elszáll. Az én meglátásom szerint érdemes ezeket szeparálni, vagy particionálni. Úgy hogy ezek kerüljenek külön-külön szerverekre. És itt fog bejönni a virtualizáció. Mert jó hogy külön szerverre, de mi van akkor, ha nincs annyi szerverem.

**Néhány alapfogalom.** A fizikai hardvert hívjuk hostnak, ez az ami a virtuális gépeket futtatja. A virtuális gépeket guest-eknek hívjuk. A menedzselés a host felületről történik. Egy kontrollált területet adnak arra, hogy ezeket a guest-eket menedzseljük. Másrészt egy kontrollált területet arra, hogy mi az amit egy guest csinálhat. Miért jó ez?

Egyrészt költséghatékony, mivel már a legelején megállapítottuk, hogy nem jó az hogy ez mind egy szerveren fut, mert az hosszútávon komoly problémákat okoz és visszaüt. De ezzel egy szeparált egységeket hozhatunk létre, akár egy szerveren belül is.

Visszatérve a legelejére. Meg tudom csinálni azt, hogy leállítom a webmail részét, és ezt egy külön egységként frissítem. Aminek a következménye, hogy nem lesz 10 percig webmail. És le lehet vonni utána a következtetést, hogy sikerült-e vagy nem, esetleg vissza lehet térni az előző állapothoz. Ez egy teljesen független egység. Közben a levelezés, még megy, a webszerverek futnak, az a többi részre nincs kihatással. Az egyszerűbb menedzsment, szintén egy kétélű dolog. Ez egy egyszerűbb menedzsmentet tud biztosítani. Növelhető a redundancia, jobb a hardverkihasználtság. Hiszen, ha van egy olyan szolgáltatásunk, amihez, néha-néha odatéved egy felhasználó, akkor az nagyon minimálisan fogja a processzort használni. Ez esetben célszerű lehet ilyen módon virtualizálni, hogy több szerver illetve szolgáltatás osztozzon egy gépen. Még a skálázhatóság is felmerül. Most ugyanez negatív szemszögből megnézve. Ez egyszerű persze, de meg kell tanulni. Mennyivel egyszerűbb, hogyha tudjuk hogy kell egy szervert telepíteni. De hát meg kell tanulni. Redundáns, de a kialakítása többletmunkát igényel. Hiszen az alapértelmezés, hogy felrakjuk ezeket a virtuális gépeket szervereket egy gépre. Könnyen single point of failure lehet az eredmény. HA igen nem figyelünk oda, hogy tényleg ez több gépen osztozzon egyszerre, legyen valami redundancia, valami elosztás. Legyen terv arra, hogyha az egyik gép kiesik. Nagyon jó dolgok vannak erre, hogy backupból, pár másodperc alatt ezeket el lehessen indítani, de erre is fel kell készülni. A hardver kihasználtság is jó, de kifogyhatunk az erőforrásból. Ez is egy szempont amit figyelembe kell venni. Itt most néhány virtualizációs megoldást hasonlítok össze röviden.

**Teljes hardver virtualizáció.** Itt magát a fizikai hardvert emuláljuk, és ilyen módon módosíthatlan image-et, módosíthatlan operációs rendszerek futtatására is lehetőség nyílik. Meg tudjuk csinálni, hogy egy szerveren Windowst futtatunk, OpenBSD-t, meg Linuxot. És ezek teljesen natívan fogják ezt látni. És magán a host gépen dönthetjük, el ki milyen hálókártyát lát, ki milyen erőforrásokhoz fér hozzá, ki mekkora diszk területet használhat. Itt is két nagy típus van. Az egyik a hagyományos, a másik a hardveresen támogatott. Az újabb processzorokban (már 2004-ben jöttek ilyenek) amelyben van hardveres virtualizáció támogatás. Itt az a különbség, hogy a virtualizátornak, ami magát a virtualizációt végzi, könnyebb dolga van, mert a hardverben olyan utasítások vannak készen, aminek a használatával sokkal könnyebb ezt megvalósítani. A KVM, ami a Linuxnak a része, ezt használja ki, és ily módon nagy komplexitástól tudtak megszabadulni. A Linux kernel fejlesztése során, ez egy praktikus lépés volt, hogy ezt meglépjék, mert nagyon sok virtualizációs dolog épült be, és már csak egy kis lépés kellett, hogy a Linux ezt már alapból tudja. Amit hagyományosként említek meg, az nem azt jelenti, hogy ezek butábbak, pont az ellenkezője, mert ők ki tudják ezt a hardveres támogatást használni, viszont a KVM csak hardveres támogatással működik.

**Paravirtualizáció.** A paravirtualizációkor egy úgy nevezett hypervisor kerül be az guest-ek és az operációs rendszer közé, és minden egyes rendszerhívás, egy translációval átmegy a hypervisoron. Viszont van egy hátránya, hogy módosítani kell alapvetően a rajta futó operációs rendszert. Itt is vannak trükkök amivel ezeket ki lehet kerülni, és maga a hypervisor is úgy fejlődik, hogy minimális legyen a változtatás. A Xen ennek egy nagyon jó példája. A Xen az UML-nek bizonyos elődjének tekinthető. Hasonló koncepciók szerepelnek benne.

**Operációs rendszer szintű virtualizáció.** Ezt úgy kell elképzelni, hogy az operációs rendszeren belül vannak elszeparált környezetek, mondjuk azt, hogy chroot-ok. Ezek az elszeparált környezetek megerősített chroot-ként képzelhetők el. Nem látják egymás processzeit, nem látják egymás hálózati forgalmát, külön diszk terület allokálható nekik. Erre a gondolatra kell építeni, hogy az egyik megoldás az, hogy a hardvert emuláljuk, ez pedig egy ellentétes irány, rendben van ez egy operációs rendszer, de válasszuk el őket, amennyire lehet egymástól. Az OpenVZ az egyik ilyen megoldás, erről fogok részletesen beszélni, ennek a kereskedelmi változata a Virtuozzo. A Virtuozzo-t tették opensource-os változáttá, ebből született meg az OpenVZ, és nem titok, mint a legtöbb opensource-os programnál igazából ez egy előszobája a Virtuozzo-nak. A legtöbb dolgot ki tudják rajta próbálni, sokan használják, és ezt vissza tudják a Virtuozzo-ba is építeni, tehát ez egy értékes szimbiózis ennek a cégnek is. Van a VServer ami hasonló csak kevesebbet tud, erről nem fogok beszélni, és a Solaris Containers (Zones) és a FreeBSD jail hasonló ehhez a megoldáshoz.

**Az OpenVZ-hez kényelmes menedzsment eszközök érhetők el,** teljes hálózati virtualizáció oldható meg vele, tehát teljesen elkülönül az IP stack. Parancssal tudok IP címet odaadni egy gépnek, és onnantól csak azt az IP címet fogja tudni használni. Nincs is lehetőség a gépen belülről más IP címet felvenni. Ez más virtualizációs megoldásoknál máshogy van. Kétszintű kvóta lehetőség van, meg lehet adni, hogy egy gép milyen kvótát használhat és egy gépen belül hagyományosan a user ill. csoport kvóta lehetőségek továbbra is használhatók. Kétszintű ütemező van, ami azt jelenti, hogy meg tudom mondani, hogy bizonyos virtuális gépek mekkora szeletet kapjanak a kernelütemező alapján. Például meg tudom mondani, hogy van egy gép ami nagyon fontos, és elengedhetetlen szolgáltatások futnak rajta, az ilyen prioritással fusson, és meg tudom mondani, hogy van itt egy másik gép, amin csak egy user tesztel valamit, az pedig minimális prioritással fusson. Nyilvánvalóan ez akkor kezd játszani, amikor 100%-on fut a processzor, akkor ez esetben, ha van 2 gép, amelyik mindkettő 100%-on viszi a processzort, akkor olyan arányban fogja elosztani, hogy amelyik nagyobb prioritást kap, az mondjuk 90%-ot fog kapni, a másik pedig csak 10%-ot. Elég szofisztikált memóriaszabályozás van. Ami szerintem még nagyon fontos, az a checkpointing ill. live migration, amivel megoldható az, hogy egy virtuális gép egész környezete lementhető egy fájlba (memória, futó processzek, hálózati kommunikáció), ezt a saját menedzsment

eszközével át lehet vinni egy másik gépre, ott pedig pár másodpercen belül indítható.

Arra kell gondolni, hogy megy például egy médiaszerver esetében, vagy akár egy HTTP szerver esetében megy egy flash tartalom, médialejátszás vagy fájlletöltés, és igazából észrevehetetlenül megtörténik az átállítás, meg sem akad a kommunikáció.

**Nézzük, hogyan lehet ezt feltelepíteni.** Debian ill. Ubuntu esetében is hasonló utasítással lehet telepíteni azt a kernelt ami erre képes. Van hozzá két alapeszköz, a vzctl és a vzquota ami a menedzseléséhez kell. Ez után egy megfelelő template-et kell letölteni hozzá, ami maga a guest gépnek az image-e. Git-ben is le tudjuk tölteni, tehát a teljes kernel forrás elérhető. Ez nem része az alap kernelnek, viszont Git-ben ugyanúgy elérhető és naprakészen van tartva, több sorozat kernel széria van, van tesztelői és éles változat. Egy konténer létrehozásakor megadjuk a gép számát, milyen OS template alapján legyen generálva, ez nem más, mint egy tar.gz fájl. Van egy alaprendszerünk, amit vagy előre más által elkészített formában letöltünk, illetve lesz majd arra példa, hogy mi magunk hogyan tudunk létrehozni. Ez a tar.gz fájl valójában a root fájlrendszer betömörítve. Megadjuk a host nevet, IP címet, ezen parancs lefuttatása kb. 5-10 mp egy mai szerveren. Tehát gondoljunk bele, hogy milyen célra kell a gép. Kb. annyi idő alatt, mint eljutnánk a CD-hez a polcig, – még nem mentünk be a szerverszobába, a kulcsot még nem vettük elő, – annyi idő alatt már készen áll egy virtuális gép. El is kell indítani, ez újabb 5 mp körülbelül.

Nézzük, hogyan néz ki a hálózat. A hálózaton egy ún. „venet” nevű virtuális interfész jelenik meg, ezen látszódik az a virtuális IP cím, amit az előbb hozzáadtunk. Utólag természetesen meg tudjuk azt csinálni, hogy az ipadd paranccsal felveszünk még egy IP címet, akkor ugyanígy meg fog jelenni még egy IP cím. Tehát nagyon kényelmes ennek a menedzselése, ez egy fontos szempont.

**Diszk tárterület:** saját fájlrendszere van amin ez megjelenik, ez egy virtualizált fájlrendszer, de valójában ezek a fájlok a host fájlrendszeren ugyanúgy fájlok. Tehát ez csak egyetlen könyvtár a host gépen. Ezek után nagyon könnyű, mert nem kell fájlrendszereket létrehozni, ill. a backup is nagyon egyszerű, mert a host gép fájlrendszerének részeként tudjuk ezeket menteni is.

**Saját template készítése:** történhet debootstrap-pel, Debianon ez egy nagyon népszerű parancs, sokan használják. Telepíteni is lehet ezzel a paranccsal, egy Live CD-vel megadjuk a disztribúció nevét, megadjuk, hogy melyik könyvtárba rakja, valamint megadjuk egy mirror-nak a nevét. Ez kb. 2 perc alatt fut le, lesz belőle egy 142 MB-os image, tömörítve ennek a töredéke lesz. Alap image-ként ez használható, a továbbiakban ha ezen az image-en változtatunk, akkor az újabb image-ek a leszámazottai lesznek, tehát nagyon egyszerű egy olyan image-t létrehozni, ami már előre be van konfigurálva. DNS szerveret nem kell konfigurálni, hiszen az alaprendszerben vannak szkriptek,

ez része az alap hálózati konfigurációnak, illetve egy-egy image konfigurációjának. De például be tudjuk állítani, hogy mi legyen az apt-get sources.list-nek a tartalma, milyen LDAP szerveret használjon. Nagyon sok alapkonfigurációt meg tudunk tenni, és így teljesen testre tudunk egy olyan image-t szabni. Meg tudjuk mondani, ha webszerver kell, akkor ez az image menjen. Van egy webszerver image-ünk, minden előre felkonfigurálva, és 5 mp-en belül működőképes.

**Az overcommit** nekem is újdonság, a Linux memóriakezelésének egy érdekes tulajdonsága. Ezzel arra van lehetőség, hogy több memóriát allokáljon, mint amennyi van. Ez alapvetően durván hangzik, ha egy gépben pl. van 4 GB memória és 4 GB swap, akkor az 8 GB, akkor lehet allokálni mondjuk 1 TB memóriát, akkor néz az ember, hogy ez jó nagy hülyeség, majd biztos az lesz az eredménye, hogy amikor tényleg elkezdik használni, akkor elindul az out-of-memory killer, és szépen elkezdi leirtani ezeket a processzeket. Már el is hangzott az elárulószó, hogy „amikor elkezdik használni”. Nagyon sok program, pl. az Apache, de ez általános tendencia, allokálnak egy nagyobb memóriát a program indulásakor, és nagyon sokszor előfordul, hogy annak mondjuk 30%-át használja ki. Tehát az overcommit nagyon rossz ötletnek tűnik első ránézésre, viszont a gyakorlatban nagyon jól működik, hogy sokkal több memóriát tudunk allokálni. Igazából a mobil

szolgáltatók és a hálózattervezés is hasonlóképpen működik, tudják, hogy nem fog mindenki egyszerre telefonálni, ezért csak töredéke az amit ki tud szolgálni a hálózat, de nagyon jól működik a gyakorlatban, mert nincs a teljes kapacitásra valós igény. Ugyanez az overcommit lehetőség más formában megjelenik az OpenVZ-nél is. Az OpenVZ-nél meg tudjuk azt csinálni, hogy van egy gépünk 8 GB memóriával, és mondjuk kétszer ennyit kiosztunk a gépeknek. Ez ennyit lát, az annyit, ugyanúgy, ahogy megmondtuk mekkora diszk területet lásson, erről néhány slide-dal korábban volt szó. Van benne egy ún. garantált memória lehetőség, amivel azt tudjuk megmondani, hogy mennyi az, amennyit biztosan fog tudni használni egy gép, ez az az érték, amit garantálni fog neki az OpenVZ. Ha out-of-memory lehetősége áll fenn, akkor meg fogja nézni, hogy ez ami túllépett melyik processz lépett túl a legjobban, és azt, azokat a processzeket fogja megállítani.

**Most a Proxmox nevű eszköztől mutatok képernyőképeket.** Ez nem más mint egy Debian-on OpenVZ és KVM egy webes felületen egy menedzsment felületbe összefogva, clustering képességgel. Ismerkedésre nagyon jó, én úgy csináltam, hogy felhúztam egy 64 bites Debiánt, és a saját repository-ból egy pár csomagot kellett felrakni, majd egy teljesen jól működő rendszert kaptam. Ha valaki szeretne ezzel ismerkedni, akkor látogassa meg a Proxmox honlapját, nézze meg, aztán, hogy utána kiköt-e ennél az eszköznél, és tényleg ezt fogja használni, vagy visszatér a parancssoros eszközökhöz, azt nem lehet tudni, de megnézni a lehetőségeket feltétlenül jó. Ezzel el lehet érni a virtuális gépeket, a template-ek előre elkészített formában elérhetők a webes felületen, a saját templateket lehet használni.

Ez a Pázmány Péter Katolikus Egyetemen a belső szolgáltatásokat ellátó gép. Látszik, hogy amikor ez a képernyőkép készült, akkor 2%-on volt a processzor kihasználva. Ezen több virtuális gép fut, a fizikai memória nemrég lett bővítve. Látszik, hogy a különböző feladatokat ellátó gépek miként vannak particionálva, vannak adatbázisszerverek, abból is többfajta, vannak különböző célú webszerverek, webmail, mindegyik egy-egy virtuális szerver. Pillanatok alatt bármelyiket le lehet állítani, újra lehet indítani, upgrade-elni lehet, nyilvánvalóan a felhasználók érezni fogják annak az egy szolgáltatásnak a kiesését, de a többi szolgáltatást ez nem fogja érinteni.

A menedzsment felület AJAX-os, tehát működés közben, amikor éppen a processzorhasználat ugrik, akkor a felületen ez élőben követhető, a memóriahasználat ugyanúgy. A grafikus és parancssoros felületen ugyanúgy hozzáférhető a gép, ami egy-egy guest-et valósít meg.

Virtuális gép létrehozásakor lehet konténert választani (az OpenVZ konténernek hívja a guest-eket), a másik lehetőség a KVM, látszik, hogy a swap-et, memóriát megadhatjuk, a virtuális ethernet beállítható. Természetesen bridge-nek a továbbadására is van lehetőség, akkor kevesebb a felügyeleti lehetőség a gép fölött.

A biztonság az egy érdekes kérdés, és az egyik kedvenc témám. Mihez képest biztonságos? Az első példában említett esethez képest, amikor mindenki ott van egy gépen, és a felhasználók mindent látnak, és a programhibákat próbálják kihasználni, ahhoz képest ez biztonságosabb. Más szempontból kevésbé tartom biztonságosnak, nagyon barátja vagyok a memóriavédelmi megoldásoknak, szóba jöhet a grsec, van az OVA projekt, az Exec-shield is szóba jöhet, de azt nem tartom olyan jónak. Több olyan megoldás van, amivel page ill. segmentation alapú memóriavédelmeket lehet beépíteni, ez a virtualizációval elveszik. Ugyanúgy, ahogy a XEN-nel elveszik, ezzel a virtualizációs OpenVZ-s megoldással is elveszik, igazából az a hardvernek olyan képességét használja ki, ami ezzel a virtualizációval nem érhető el. Ez sajnos megint egy olyan kompromisszum, amit meg kell tenni, illetve ismét egy olyan kérdés, amit el kell dönteni, hogy mi az amire egy virtualizált gépet használunk, és mi az amire nem. Ha van egy gép, szolgáltatás ami nagyon túl van terhelve, rárajuk egy ilyen gépre, akkor a többi szolgáltatás is ezt érezni fogja, mert tőle fog erőforrást elvinni.

Itt is szóba jöhet a claszterezési képesség illetve ennek a migrációs képessége, ha beállítjuk a felületen ezt a migrációs lehetőséget, akkor pár pillanat alatt át lehet rakni egy másik gépre, a legtöbb idő az image rsync-elésével telik el. Ez akkor kezd izgalmas lenni, amikor már több

gépünk van amiből egy ilyen clustert kiépítünk, és azon belül migráljuk a gépeket, illetve osztjuk el az erőforrásokat, hogy ki hova jut, ki mihez fér hozzá.

**Hallgató:** mit értesz itt cluster alatt?

**Előadó:** ez valójában nem OpenVZ-s, hanem Proxmoxos fogalom. A cluster esetében meg lehet azt csinálni, hogy több gépet (host-ot) összeköt, mindegyiken az a Proxmox fut, a menedzsment felületen be lehet állítani, hogy ez a kettő cluster lesz. Ekkor SSL kulcsok segítségével biztonságos kapcsolat lesz felépítve a két gép között, ez után közösen lehet menedzselni ezeket, illetve a két host között a gépek (guest-ek) mozgathatók.

**Hallgató:** ha az egyik host leáll, akkor a másikon automatikusan elindulnak a megfelelő guest-ek?

**Előadó:** ezt nem próbáltam, szerintem nincs ilyen lehetőség.

**Hallgató:** az OpenVZ-nek van konzolja, vagy csak SSH-n keresztül érhető el?

**Előadó:** igen van konzol.

OpenVZ esetében nagy előny, hogy mivel kernelből egy van, csak azon belül vannak szeparált környezetek, ennek az az előnye, hogy az IO minimálisan korlátozódik. Leginkább a beállítások határozzák meg azt, hogy ki mekkora szeletet kap belőle. Elég szofisztikáltan állítható be, meg lehet határozni, hogy ez a gép teszt, ezért alig kapjon valamit, a másik az mehet, maga a virtualizációs veszteség az teljesen minimális.

**Hallgató:** az OpenVZ más architektúrára is elérhető?

**Előadó:** nem tudom biztosan, hirtelen azt mondanám, hogy nem.

**Hallgató:** többmagos processzoroknál van valamilyen szabály?

**Előadó:** a Linux kernelnél van erre lehetőség, ami figyel arra, hogy egy processz ne ugráljon a processzorok között. KVM esetében meg lehet mondani, hogy melyik processzoron fusson, OpenVZ esetében meg lehet határozni, hogy hány processzort lásson, de hogy pontosan melyikén fusson azt nem.

## Lajber Zoltán: Hálózat és alap infrastruktúra

**Az alpinfrastruktúráról lesz szó.** A mottóról sokat nem mondanék: „Mindenki másképp csinálja.” Most is úgy kezdjük, hogy egy kicsit a tervezésről, nem biztos, hogy sorrendben meg rohanósan is megyünk. Nagyjából arról lesz szó, hogy hogyan kell syslog tehát naplófájlokat gyűjtögetni, hogy kell forgalmat naplózni, erről túl sokat nem mondok, de egy-két szkriptet elárulok.

Akinek még nincs tapasztalata és főleg a vezetőség, aki úgy látja, hogy működik a hálózat és ez teljesen természetes, nem is tudják, hogy van rendszergazda. A szerverek maguktól mennek, nem kell hajtani a biciklit, hogy ne aludjon ki, itt pedig a pingvines-jegesmedvés reklámra tudnék utalni. Vannak olyan szolgáltatások, amikor roppant fontosak. Nem látszik, de nekünk az NTP elég fontos, a felhasználók akkor veszik észre a DNS jelenlétét, amikor nem működik. Én még tudom fejből az index IP címét, tehát nagyjából el tudom tölteni a napomat, de ezt a felhasználók nagy része nem tudja. Ilyen, hogy tftp, már említettük, hogy úgy telepítem az új vasat, hogy bedugom a hálózatba a PXE boot, NFS, ehhez kell a tftp is, mint protokoll, szó lesz a TACACS+-ról és Radius-ról is, ha lesz időnk.

Nem ártana tudni, hogy mi történt a hálózatunkban, ez a Nagios, ami küldözgeti az SMS-eket szorgalmasan, azt megmutatom, hogy tudunk SMS-t küldeni konkrétan, ha nem is többet. A Nagios-ról nagyon sokat lehetne beszélni. Ha itt átrohanjuk, akkor javaslom az IPSZILON előadásorozatot ez az ingyenes programok szemináriumok és labor az NIIF keretében. Ott egy témáról egész napos előadás szól slide-okkal és videókkal.

A Nagios az azért jó, hogy ő szóljon, hogy nem megy a webszerver. Ha kérdezik, akkor tudjuk mondani, hogy tudjuk, magyarul pro-aktívak vagyunk. Nem ártana azt tudni, hogy mi fog történni, főleg a pályázatos világban, ahol a főnök megkérdezi, hogy mit szeretnél venni. Grafikonokat rajzolunk, aminek önmagában nagy jelentősége nincsen, ha egy napra rajzolgatunk, de ha van egy évre visszamenőleg, akkor azt tudom mondani, hogy hamarosan upgrade-elni kell a webszervert, kicsi lesz ez a sávszélesség, elkezdett nőni a mail gateway processzorterhelése, tehát el tudom tervezni a dolgokat. Illúzióm nincs a beszerzéseket illetően, én a hátizsák probléma algoritmusát alkalmazom, tehát megírom a toplistámat, felül a legdrágább tételek, és mikor van pénz, akkor a legdrágább tételt, ami belefér a keretbe megoldom belőle. Néhány év alatt egész jól működő algoritmus.

Tervezésről a hálózat menedzsmentre dedikáljunk egy gépet, klasszikus probléma, nem kell combosnak lennie, csak megbízhatónak. Mondjuk az előző generációs szerver vas az nekünk jó, amennyiben nem attól a gyártótól származik, ahol tudjuk, hogy 3 év után kiszáradnak a kondenzátorok. Volt olyan gyártó, ahol 8 db gépünknel, 3 év garancia volt rá plusz-mínusz két hónappal hullottak ki az alaplapi kondenzátorok, szerencsére 8-ból 6-ot még elszámoltak.

Hagyományosan ezeket a gépeket NOC-nak szokás nevezni, ez a Network Operation Center. Nagyobb hálózatokban erre külön subnet és külön cluster, van vagy valamilyen elosztott géprendszer. A NOC az kiemelt jogokkal rendelkezik, tipikusan az összes hálózati eszköz elérhető innen, emiatt aztán védenünk is úgy kell, ha valaki be tud törni a NOC-ra, akkor mindent megkaparintott, ott az összes config stb., bármit csinálhat. A védelme kritikus, azt mondanám, hogy a NOC-on ne legyen semmi kintről elérhető szolgáltatás. Szétválasztása a hálózatról, megint a VLAN-ok jöttek elő, ezt ugorjuk is, de például nálunk van egy egyes VLAN, ott vannak a hálózati eszközök, ez az alapértelmezett VLAN, oda nem is érdemes felhasználni több okból. Itt van a switch-eknek, mindenkinek a saját IP-je, amin keresztül lehet menedzselni. Ez nyilván egy belső hálózat, tehát RFC1918-as címmel (Address Allocation for Private Internets), amit nem is NAT-olok ki a nagyvilág felé, ha frissíteni akarom a szoftverét a switch-nek, akkor azt letöltöm a NOC-ra és onnan frissítem, nem a nagyvilágból frissítek rá. A szokásos dilemma, hogy a DNS névfeloldás csak a menedzsment gépeknek működik (ezzel a tartománnyal kapcsolatban), tehát a DNS, ha erről a tartományról kérdez, nem válaszol akárkinek, még a saját user-eimnek sem. Van egy külön

subdomain rá a Szent István Egyetemen ezt a subdomain-t úgy hívják, hogy MTZ, ez a Management Tool Zone.

Illik dokumentációt készíteni a hálózatról. Ki mit használ. A Visio-t ezen a szekción azt hiszem gyorsan ugorhatjuk, vele több probléma is van. A dia az nekem nagyon nem jött be, én korábban gépészmérnök voltam és néhány CAD-es szoftvert is ismertem, a diának teljesen más a gondolkodásmódja. Ha húzok egy vonalat, akkor utána ne álljon vissza más üzemmódba, maradjunk vonalhúzásban, de ez egyéni kérdés.

Tgifet használok, az nem mai gyerek, de amit láttuk már itt ábrákat, tűzfal elrendezést, az is tgif-fel készült. Jó pár dolgot tud, lehet saját ikonokat is belerakni. Gyakorlatilag rajzolok valamit, meg hozzárendelek attribútumokat, ezeket összecsoportosítom és úgynevezett szimbólum lesz belőle. Ha azt mondom a tgif-ben, hogy ctrl-i router kiválasztom, akkor a router-nek van neve, típusa, amit rákattintással át tudok írni és rejtett attribútumként például, ha van neki egy URL-je, a tgif tud úgy menteni, hogy HTML imagemap-et, ami ott van a rajzon. Ha erre rákattintok, milyen elegáns megkapom azt az eszközt vagy Munin-ban azt a szerveret, amire kattintok. Ennyire nincs megcsinálva az én dokumentációm, de az elvi lehetőséget már kidolgoztam. Itt vannak az ikonok, amiket használunk, gyakorlatilag ez a neve, típusa meg rejtett attribútuma, ami az URL.

**Eseménynaplózás:** sokat erről nem beszélnék. Nagyon fontos lehet. Ha már betörtek, akkor jó lenne tudni, hogy mi történt. Ilyenkor elég gyakran sikálják a dolgokat, jó lenne távolra logolni. Ilyen szempontból a legbiztonságosabb a nyomtató, mert onnan törölje a logot a betörő, ha tudja, ez viszont nem a leggyorsabb. Nem tudom kinél mennyi log keletkezik, ez a másik probléma, hogy lelkesen logol mindent a rendszergazda, aztán meg se nézi. Vannak logfeldolgozó eszközök, ebbe nem mennék bele. Nem kell ahhoz nagy hálózatnak lenni, hogy ilyen napi 100 000 sorok keletkezzenek, ebből kiszűrni, hogy mi rossz. Erre mondta az egyik tűzfalas mérnök barátom, hogy ha egész nap logokat nézel, akkor hazamész és kiirtod a családot, és ebben van valami.

A másik, hogy a túl részletes napló kezelhetetlen, ne essünk abba, hogy naplózzunk mindent, aztán majd jó lesz valamire. Ez az egyik. A másik az, hogy a logolás az elég processzorigényes folyamat, a legtöbb hálózati eszköz és tűzfal a forgalmat még elbírná. Volt már olyan támadásban részem, hogy az eszköz el bírta volna a forgalmat, de a naplózásba belepusztult. Ezért kb. ez a szerver illetve vasfüggő is, egy iptables linux esetén is már a 100 Mbites hálón nagyon tanácsos a log rate limit-et rakni. A technikáját meg lehet nézni az IPSZILON-on Kadlecsik Józsi előadásában. De az a lényeg, hogy az iptables-ben van rate limit, egész komoly burst-öl stb. Nagyon jól meg van csinálva. Kötelező. Jártam már én is úgy, hogy keresem a hibát, akkor ne dobjon el néhány logot vagy ne limitálja. Direkt van ilyen láncom, hogy log drop, tehát úgy ugrok az iptables-be, hogy ezt most logold és dobd el, meg van a log drop debug, ahol nincs rate limit, és nyilván felejtettem már ott ezt, amikor megérkezett a támadás...

A logot kell rotálni, borzasztó dolgok vannak, ha elkezdenek nőni a végtelenségig. Erre szerencsére a logrotate, egy nagyon jó eszköz. Bár ott is vannak tréfás dolgok, mert lehet olyat csinálni, hogy include-olok könyvtárakat. Viszont a logrotate hajlamos arra, ha kétszer definiálok ugyanarra a fájlra, akkor megáll és a mögötte lévőkkel már nem foglalkozik. Ha elég nagy számú géped van, akkor ezt akkor veszed észre, mikor betelt a diszk. Itt jön az a megfontolás telepítéskor meg kvótázáskor, hogy egy szervernél, ez egy klasszikus kérdés, hogy particionáljam, ma már azt lehet mondani, hogy nem particionálunk, hanem LVM-et csinálunk. Tehát fogom a gépet, általában van két diszk, amiről bootol, van egy sima, nem LVM, semmi trükk, egy rendes RAID 1, amiről bootol az op. rendszer, ez a root partíció, a /boot-ot külön nem szoktam használni, van egy swap és a maradék egy LVM. Tehát egy nagy volume group és telepítéskor ennek kb. 50-60%-át osztom ki. Aztán amikor valamelyik fogy, azt megnövelem, hál istennek ez xfs-sel meg lw-vel ez leállás nélkül megoldható. Ha induláskor ki kell osztani az összes diszket, akkor ott valamit elrontottunk a beszerzésnél.

Logolásról annyit, hogy a syslog-ng nevezetűt érdemes használni, tehát az eredeti unixos alap syslog az felejtős. A syslog-ng az szép és kényelmes, ráadásul magyar termék. Még annyit, hogy a syslog-ng-nél még alapfeltétel, hogy együtt járjanak a gépek és berendezések órái, inentől lesz fontos az NTP. Semmi értelme öt gépről összeszedni a logokat, ha van köztük 5 tized másodperc eltérés, mert akkor az események sorrendje már látszólag megváltozott és nem fogjuk tudni, nyilván csak akkor, amikor a nemzetbiztonsági hivatal ott toporog meg a rendőrség.

Syslog-ng konfiguráció részletek macera is. Lehet csinálni különböző célokat, például, hogy abba a fájlba fog logolni, meg lehet mondani, hogy ki legyen a tulajdonosa, milyen jogosultságai legyenek, itt például az adminok nézegethetik stb.

Lehet filtereket csinálni. Itt látszik egy olyan filter, hogy nem csak arra szűrök, amelyik gép küldte, ez a host, hanem szerencsére például ez az eszköz olyan, hogy a log üzenetben mindig van egy olyan szám, ami a prioritást írja le, és így meg tudom mondani. Van olyan eszköz, ahol nem ilyen a string, de meg tudom mondani, hogy az az esemény milyen syslog-ng prioritásra kerüljön stb. Nagyon sok lehetőség van, nem mennék bele. Utána az egészet összerakom, mert azt mondom, hogy ha a netről jön és átmegy a filteren, akkor tedd abba a célba. Ez alapján látható, hogy szét lehet szedni különböző csoportokra. Tudok olyat csinálni, hogy az összes Apache log ide menjen, de külön szét tudom szedni gépenként is.

Ami még megfontolandó, mennyi dolgot őrzök? Mennyi ideig és hogyan rotálok? Nyilván amelyik napi két sort termel, azt nem érdemes naponta rotálni, mert a gzip header több lesz, mint az eddigi log tömörítve, de a nagyon nagy logot meg macerás kezelni. Van olyan, hogy ne tömörítse be azonnal, hanem a .0 az még tömörítetlen legyen, a Linuxon nincs igazán jelentősége, mert ott van a zgrep. Például az alap Solarisnál, ha nem rakom fel, akkor nincs zgrep, akkor érdemesebb, ha kell még az előző napi logom, így kezelni.

Itt van a logrotate részletek, például ez azt csinálja, hogy a /var/log/routers alá került összes log fájl naponta átpakoljuk a /var/log/routers/old-ba és 365 napig őrizzük és tömörítjük. A logot nagyon szépen lehet tömöríteni, átlagban tizedére össze lehet nyomni. Emiatt a NOC-on célszerű, ha van egy külön /var/log partíció, mert ha mégis elrontom a szkriptet, történik valami, egyéb szolgáltatás nem hal be miatta.

Alapvetően ez egy állandó dilemma, hogy mennyire particionáljuk szét a gépet. A desktop gépemen van egy partíció és kész vagy egy fájlrendszer és azon van minden. Abból nem érdemes fragmentálni a rendelkezésre álló területet. Szerveren viszont úgy kell gondolkodni, hogy ha valami el tud szaladni, akkor nyilván idővel el is fog méretben, az lehetőleg minél kevesebb szolgáltatást rántson magával. Klasszikus, ha például nincs külön partícióm és a /-on vagyok, akkor nem fogok tudni ssh-val bejelentkezni olyan egyszerűen, marad a konzol.

Forgalom naplózásról még annyit, hogy alapvetően 3 módszert használok. Nem szeretem azokat a linuxos megoldásokat, amelyiknél az iptables-be plusz sort kell azért beírni, hogy a naplózás működjön. A tűzfal szabályrendszere e nélkül is épp eléggé bonyolult, a forgalom mérés ilyen alapon nem az ő dolga. Én ezt a netacc nevű alkalmazást használom, ez mai vasakon néhány Gigabites interfészig teljesen jó. Már tud adatbázisban is dolgozni, az enyém még TXT fájlban és logokban dolgozik és arra vannak kész szkriptek, mutatók is valamilyen.

A kisebb router-eknél lehet olyat csinálni, ez a span, ez gyakorlatilag azt jelenti, hogy forgalom kitérítés, tehát ha például átmegy egy switch-en, mondjuk a switch-en a giga0/1 a nagyvilág, akkor a giga0/2-n ki tudom másolni ugyanezt a forgalmat. Ott nem megy forgalom, nem lehet gépet dugni rá, de a NOC-nak a második interfészével, a pcap-pel tudom nézni a forgalmat, és ebből lehet gyártani, erre 26 eszközt lehet találni a neten. Kaposváron egy srác, amikor a kollégiumi sávszélesség még limitált volt, akkor csinált erre egy olyan eszközt, hogy a NOC-os gépük mérte a kollégiumi kvótát, és aki elérte a kvótáját, akkor megírta a tűzfalszabályt és letiltotta azt a user-t, tehát ezt ennyire lehet automatizálni. Másnap meg kezdte újból. Nyilván itt a debug az elég fontos, mert a tűzfalon elkezd nyitogatni egy program, az messzire is vezethet.

A nagyobb Cisco dobozok azok tudnak egy úgynevezett NetFlow nevezetű exportot, sőt újabban megjelent IEEE szabványként ennek egy utóda. Ez annyiból jó, hogy hardverből az eszköz maga ad tipikusan öt paramétert, ami egy flow-t azonosít, ez a cél és forrás IP, cél és forrás port és protokoll.

Ez az ötös azonosít egyértelműen egy flow-t, megmondja, hogy hány byte, meg hány csomag ment át rajta és ezenkívül olyan információt, ami a tűzfalon is nagyon lényeges, hogy melyik interfészen jött be, melyiken ment ki stb. Tud olyat is, hogy melyik as-be távozott. Ezt egy elég tömörített formátumba el tudja rakni, ez most már nem 45, hanem 80 Gbyte egy hónapnyi, de elfér. A flow-tools az úgy működik, hogy meg lehet neki mondani, hogy ebbe pakolj. Be lehet azt állítani, hogy évente egy könyvtár, havonta egy könyvtár, naponta egy alkönyvtár és óránként egy fájl. Ez konfigurációs opció, hogy mennyire darabolja. Nálunk kb. ez a használható méret, hogy lehessen hova tenni, ez Gbit-es internet sávszélesség és napközben olyan 2000-2500 bekapcsolt PC hálózaton ezt így bírja. Ezt is szépen lehet tárolni.

Rengeteg eszköz van ennek a feldolgozására. Például nagyon praktikus az, tapasztalat, hogy a vírusos gépek, nagyon nagy számú, de nagyon kis byte-os, tehát byte-ra kevés a forgalom, de nagyon nagy számú flow-t kezdenek. Benyelt valami csúnya kártevőt és az elkezd próbálgatni a hálózatot. Van is ilyen szkriptünk, ami nem byte-ra nézi a top 10-et, hanem flow számra és gyönyörűen korrigál. Azért kollégiumi kontrollálatlan gépeknél képzeljük el, hogy mikor szimultán 25 000 UDP kapcsolata van a tűzfalon a user-nek, elfogyott a NAT, mert egy IP címre 32 000 vagy 65 000-t lehet NAT-olni, de többet nem nagyon. Kiment a kolléga, nem betörő, 1,8 GHz-es gép és mondja, hogy nem gyanús a géped, hogy lassú? Azt mondja, hogy ha a Wordben akarok gépelni, akkor ki kell húzni a netből, de nem. Halványan utalt a kolléga, hogy vírusirtó, tűzfal? Azt mondta a barátom, hogy az lassítja. És a frissítés? Azt mondta a barátom, hogy az elronthatja a Windowst. Egy teljesen frissítetlen, vírusirtó, tűzfal nélküli gép a kollégiumban. Akkor amikor átlagban egy ilyen Service Pack 3-as XP-nek statisztikailag van 5 perce arra, van 90% valószínűséggel valami olyan szoftver is legyen rajta, amit nem a gazdája telepített.

Egy kis szkript awk nyelven, az eredményét mutatom, nekem az egyik kedvenc programnyelvem ez, log és egyéb feldolgozásokra. Itt is látszik, hogy a paraméter és felhasználó kezelés az első két oldal és a program az egy oldal. Azt mondom neki, hogy net-attck IP-ből a cél, ugye destination alapján csinálja a logot, a top 10-et kiírja, itt csak a hármát mutatom. Ugyanúgy forráscímekre, látszik, hogy lehet küldeni a technikust a 132-esre meg a 205-ösre, mert elég csúnyán viselkednek. Főleg úgy, hogy a negyedik, amit itt már nem látunk az néhány százas nagyságrendben van.

Reggel ezt oda lehet adni a technikusnak, hogy ezeket a gépeket nézze meg és csináljon velük valamit. A harmadik változat az ugyanúgy csak önkényesen /24-es netmask-ra dolgozik. Ez inkább akkor érdekes, hogy honnan ér minket valami zargatás. „Támadnak, mit tiltsak ki?” típusú kérdésre válasz ez. Hasonló szkript van, ami portokra nézi, nem IP-kre, ugye itt a protokollok. Az awk gyönyörűsége, hogy ez az érdemi rész, ez átmegy a logon, minden soron lefut és amikor végzett, akkor ezt kiírja. Igazából ez a 3 sor a log feldolgozás a többi az a paraméter és egyéb kezelés. Ilyen statisztikákat mutat. Hogy cél vagy forráspont szerint melyek a legnépszerűbb portok. Ugye az 53-ast sokan használják, de a 25-ös az gyanús, hogy mit akar, mikor úgysem engedem ki.

NTP. Említettük már, hogy milyen fontos, hogy az órák együtt járjanak, legalább a szervereknél. Visszakanyarodva a XEN-nek van egy olyan tulajdonsága, hogy hiába NTP-zek a guest-eken, az óra összevissza jár, ha a host-on nem jó. Az a lényeg, hogy a dom0-kon jó legyen az óra és innentől kezdve majdnem mindegy, hogy mit csinálunk a guest-eken, ha a dom0-n jó az óra. Ez a Network Time Protocol, mi ezt úgy csináljuk, hogy van egy edge router ami a külvilágot kezeli, meg azon van a DMZ, és van egy core router ami a belső hálót kezeli. Ez a nagyvilágban néhány órához szinkronizál, általában ez a default gateway, és ehhez lehet szinkronizálni. Ez az alapfeltétel, nyilván akkor nem jó a default gateway, ha tűzfal van. Tűzfalon nem szolgáltatunk NTP-t sem, az fekete lyuk, – erről megoszlanak a vélemények, kis hálózaton lehet ilyet csinálni, sőt akár DNS is lehet rajta, – és utána mindenki ezt az egy NTP kiszolgálót használja.

**DNS szolgáltatás.** Meglepően sok szolgáltatás függ tőle. Tapasztaltam én is, mikor bedőlt a hálózat, a DNS szerver elhasált, fel akartam húzni egy gépet, és például a PXE boot és az NFS root-os dolog nem futott rendesen, mert a DNS timeout-ok később jártak le, mint az NFS timeout-ok. Ssh: ha ég a

ház, gyorsan kellene ssh-znod, de ha nincs DNS, akkor az ssh is ültet a kispadon, mielőtt beenged, ha nem tudja a DNS-t feloldani. Tehát nagyon-nagyon fontos a megbízható és redundáns DNS.

Régebben volt egy DNS szerverünk, ami a mi zónáinkat szolgáltatotta a nagyvilágnak meg feloldott a usereknek. Aztán ismertté vált ez a rekurzív DNS probléma és illet hogy kifelé ne válaszoljunk mindenféle rekurzív kérésre. Én azt mondom, hogy DNS szerver ügyileg érdemes szétválasztani, hogy van egy DNS szerverünk a DMZ-ben, ami a zónáinkat szolgáltatja, és bent van valami kis cache only DNS szerverünk, aki a felhasználóknak feloldja az összes nevet, ha kell akár a mi szerverünkön.

Logikus nevezéktan. Nálunk Zeus, Héra, Sziszifusz meg ilyen gépnevek voltak, erről tudni kell, hogy melyik mit csinált. Tudni kell, hogy a volt főnököm anyukája ógörög-bölcsész-filológia stb., és telepítettünk egy gépet és nevet kellene neki találni. Mi a hülyék istene? Hívtuk a bennfentest, mondta, hogy ott Taigetosz van, kis csúsztatással, így mégse hívhatjuk a gépet, így Sziszifusz lett a neve.

Ez különben kezdődik a kábelezésnél. Ugye kábelezést nem csinálunk úgy, hogy lerakjuk az asztra a switch-et, aztán elvisszük a gépig, mert valaki belebotlik, meghúzza a gépet, rántja az egészet. Patch panelt kell csinálni és akkor lehet kereszt/egyenes kábel, meg ilyeneket könnyen cserélgetni. Viszont ha már patch panel van és fali aljzat, akkor lássuk el címkével. Na, de hány darab 27-es fali csatlakozó van az épületben, és az hol van? Nálunk a F2I-001 azt jelenti, hogy a főépület második emeleti „I” rendező szekrények közül az 1-es kábel. A hozzá tartozó switch-ről (asw-f2i1.mtz.gau.hu) látni, az asw azt jelenti, hogy access switch, az F2I rendezőszekrényben lévő egyes switch és ugye az mtz.gau zóna, amit ugye már említettem. De hasonlóan például a ups-syma.mtz.gau.hu, ez azt jelenti, hogy ez egy szünetmentes tápegység, symetra és azok közül az „a”. Kettős látásra már utaltam, van belőle egy „b” is.

Ez azért is egyszerű, mert később a Nagiosban megjelent már az, hogy például csillaggal, wildcard-okkal tudunk neveket adni és azt mondjuk, hogy access switch-ek csoport, ki kapjon SMS értesítést az access switch halálról, asw\*-ról SMS-t kapnak a netadminok. Ezt így könnyen lehet utána kezelni.

A szervereken elég gyakori ez a beépített menedzsmet eszköz, azokat is úgy nevezzük el. Ez a pityu név az úgy jött, hogy jöttek ezek az új vasak, és már virtualizált volt, tehát nem tudjuk elnevezni webszerver, mert ez egy dom0 és mi legyen a neve? Mondta ott az egyik srác, hogy legyen pityu, lásd a „Sövényen túl” című klasszikust, és így pityuból van nekünk összesen 8, 0-7.ig, ahogyan azt kell. Időnként némelyiket kivesszük, mert virtuális gépek és lehet költöztetni, ha valami projekt indul, akkor kivesszünk egy pityut, elnevezzük valaminek, aztán vagy vesszünk helyette egy újat vagy visszakapjuk. Virtuális gépek rohangálnak és volt, hogy kolléga megállt a rack szekrény előtt: „Emberek, a webszerver most melyik pityun van?” Minden pityunak van szerviz processzora, arra be lehet lépni és onnan lehet konzolt szerezni.

TACACS+, a magyar adminok azt mondják, hogy „takács”. Ez egy nagyon praktikus eszköz gyakorlatilag a RADIUS a rokona, de az az ökölszabály, hogy az eszközre belépés, és az ottani parancsokat TACACS-szal csináljuk, az átmenő forgalmat pedig RADIUS-szal autentikáljuk. A TACACS-nak az az előnye, hogy usernév password-del lehet belépni az eszközre, – a céges méretben használhatókról beszélünk, – és képes arra, hogy minden kiadott parancsot naplózzon. Ezt elteszi a log fájlba.

Ez roppant praktikus, nem azért, mert én nem bízok meg a Sanyi barátomban, mert tényleg ő a layer1, csak tipikus, hogy telefonál a user, hogy nem megy az internet. Mondhatom, hogy tudom, én kapcsoltam ki, mert nálam van a kulcsa. De mióta? Két hete. Mit csináltunk két héttel ezelőtt a hálózatunkban? Van 70 db switch-ünk, melyik switch, miről van szó? Ha bemondja a fali csatlakozó számát, ez a következő kérdés. Ha nem tudja, akkor megkérdezzük a MAC address-t, ezek után általában megmondja a fali csatlakozó számát. Bár egyszer a kolléga megkapta azt, hogy nekem nem MAC-em van, hanem PC-m. Alkalmazott bölcsészkar volt, neki elnéztük ezt a problémát. Az a lényeg, hogy megtudjuk miről van szó, és így elég hamar kiderül, hogy mondjuk 2 héttel ezelőtt a Sanyi átrakta 12-esből 16-os VLAN-ba ezt a switch-et. Tehát fel tudom hívni Sanyit, hogy ezt meg

miért csinálta, és akkor kiderül, hogy azért, mert ő mondta. Ilyen szempontból fontos ez a naplózás, mert kiderül, hogy végül is mi történt itt.

A RADIUS-ról annyit, hogy VPN-t autentikálunk vele és a 802.1x-es vonalat. A RADIUS mögött LDAP van. Az LDAP-ba bekerülnek a diákjaink meg a dolgozóink, és mindig van kivétel, akit bele kell rakni. Erre a RADIUS-ban van egy user fájl, ahol konferencia stb., meg tudunk oldani, anélkül, hogy az LDAP-ot szanaszét kellene hekkelni.

TFTP: van a Cisco eszközeinken egy alias, ezzel le lehet menteni egy központi helyre a konfigurát. Gyakorlatilag az a menet, hogy ha valaki változtat valami konfigot, lementi az adott eszköz memóriájába, hogy áramszünet után az jöjjön vissza, és lementi a központi helyre, ahol megy WEB és lehet a TAPACS logot sem biztos, hogy kell piszkálni.

A tftp-t nem csak erre használjuk, hogy a konfigot le lehet írni, az op. rendszer frissítés a hálózati eszközökön szintén itt történik, és PXE boot-os NFS-es, mondjuk rescue környezetünk, ahol próbáljuk uniformizálni a gépeinket, hogy ne legyen túl sokféle. Van egy desktop rescue, van egy szerver rescue és még szerverből van háromféle vasunk, minden eszköz be van fordítva a kernelbe, ami PXE-n jön és tudjuk használni. Tftp-ben nincs autentikáció, ez nagyon lényeges. Viszont írni akkor tud csak, ha mindenki által írható a fájl. Konfigoknál ott 666 joggal kell megcsinálni, toucholni kell, hogy le tudja írni a hálózati eszközt. Ha már létezik, akkor ebből nincs probléma, de egy új eszköznél figyelni kell erre. Ezt félretettük a /var alá valahova és nem tudom, nem a /boot vagy /ether/boot vagy hová teszi gyárilag a linux, ezt nem ott használjuk.

**Nagios:** nagisoról néhány tévhitet. Azt mondják, hogy majd pingeti helyettem a host-okat a Nagios és majd szól. A Nagios nem host ellenőrzésre, hanem service ellenőrzésre szolgál. Ha egy host-ot definiálunk http, smtp, mysql service-t, akkor próbálja ellenőrizni, és optimalizálás során könnyen előfordulhat, hogy nem is ping-eti meg a host-ot. Minek ping-esse, ha HTTP-n a TCP kapcsolat felépül, akkor ez valószínű ping-ethetné is akár. Alapvetően a Nagios service-ekben gondolkodik és nem host-okban. A host-ot akkor ellenőrzi, ha nem érhető el egy szolgáltatás, akkor megnézi, hogy a host elérhető-e, ha az sem, akkor a parent host-ja elérhető-e, azt mondja, hogy down vagy unreachable. Ez egy lényeges logikai dolog.

Mi adjuk meg configban, hogy kinek ki a felmenője, hogy épül össze a Nagios. Azt is kell tudnunk dönteni, hogy layer2 vagy layer3 alapján gondolkozunk. Ez azért lényeges, mert a Nagios ahhoz elég okos, hogy ne szóljon nekem, hogy host down, ha előtte egy hálózati eszközzel esett, mert akkor unreachable-t küld. A kérdés az, hogy engem a router érdekel vagy a switch, ami előtte van. A legrosszabb, ha elkezdem keverni.

A Nagios nem fekete-fehér, megy/nem megy állapotú, hanem négy állapota van: van az OK, a warning, a critical és az ismeretlen állapotú. Az utolsó állapot ez akkor fut le, ha a Nagios valami hibával találkozik, például nincs elég helye stb., valamiért nem futott le a plugin, nem tudta megállapítani a service-t. Ezt jelzi, hogy nem biztos, hogy a service-szel baj van, csak nem tudom, hogy mi van.

Van 3 konfigurációs paraméter, ezeket ne piszkáljuk, illetve csak akkor piszkáljuk, ha tudjuk, hogy mit csinálunk. Ez nagyon lényeges a Nagios működése szempontjából. Vannak „hard” és „soft” állapotok. Ez is nagyon lényeges, és hogy mikor küld nekünk értesítést. Úgy működik a dolog, hogy van egy normál intervallum, amikor mondjuk 5 percenként ellenőrzi, hogy minden rendben van-e. „Hard OK” állapotban vagyunk. Az egyik ellenőrzés itt az időben 10-essel jelzett, az sikertelenül zárult, mondjuk lehalt a HTTP szolgáltatás. Ekkor „soft critical”-ba kerül az állapot, nem szól még senkinek a Nagios, nem küld SMS-t, nem küld emailt, bár sárgul, ha nézegetjük. Viszont a szokásos 5 perces, tehát a normál intervallum-t azt felveszi egy retry intervallum-ra, ez tipikusan 1 perc, és az alapértelmezés szerint ötször rápróbálkozik, magyarul innentől kezdve a kezét rajta tartja a dolgon. Gyorsan megnézi és ha ötödszörre is bukik az ellenőrzés, akkor „hard critical” állapotba kerül ez a szolgáltatás és kimennek az üzenetek, meg elindulnak az eszkalációk és egyéb dolgok, amiket tud. Innentől visszaáll a normál ellenőrzési ütemre és amint megjavul, akkor azonnal „hard OK”-ba fog kerülni a dolog. A critical-ra nem fog alaphelyzetben üzenni. A critical az olyan, hogy ping-eti, de

nagyobb a loss, mint a beállított érték, ezt meg lehet adni a ping service paraméterénél, hogy mennyi az, amit warning-nak és error-nak tekint. Ugyanez diszk ellenőrzésnél például 92% felett warning és 95% felett error state.

**Nagios.cfg:** ilyen objektum orientált jellegű dolog. Nagyon lényegesek a kontaktok. Megmutatom, hogy én hogyan küldök SMS-t Nagiosból. Az értesítés kimehet több útvonalon, ha igazán nagy gáz van, akkor az e-mail nem megy ki. Így nézett ki a Nagios, ha egy olyan user lépett be, aki nem volt kontakt minden géphez. A Nagios az figyel arra, hogy ki kihez kontakt, hogy amikor belép a felületre csak azokat a gépeket látja, amikhez köze van. Utaló jelek vannak a többire is.

Vettünk egy ilyen kis kütyüt, soros porton lehet rákötni a szerverre, persze ma már nincs a szervereken soros port, ezért egy USB-RS232 átalakító is bekerült a vonalba, aztán antennacsatlakozó van rajta. Mi ezt szofisztikált mérésekkel hajlítottam gemkapocsból egyet neki és ez elég. Meg megláttuk az árlistán, hogy a kábel és az antenna mennyibe kerül.

Ha valakinek hekkelnie kell bármilyen SMS küldőt, akkor a kissg-nek az oldalát (<http://gatling.ikk.sztaki.hu/~kissg/gsm/>) ajánlom mindenki figyelmébe, nagyon sok berendezésnek és általában az ilyen GSM modemek vezérlőkódjai, ami a modemen az ATZ, annak megvannak a rokonai, az ott nagyon szépen le van írva. Van egy gsmsms tools apt-get install típusú csomag, de egy kicsit átszabtam az indítását. A lényege az, hogy itt látszik egyrészt, hogy a /dev/ttyUSB0-n keresi az eszközt, a bejövő SMS-eket erre az e-mail címre küldi, de ami még lényegesebb, hogy spool könyvtárból dolgozik. Van egy /var/spool/sms/out könyvtár.

Ez a program nem akkor indul, amikor a Nagios küldeni akar, hanem ő percenként poll-ozza ezt a könyvtárat, ha ott talál egy fájlt, akkor az első sor a telefonszám, a többi meg az üzenet, és kilapátolja. Nagios-ba pedig így van bekonfigurálva, kell egy notify-by-sms parancsot csinálni, ezek a változók szépen le vannak írva a Nagios-nak a doksjába, és ez alapján látszik, hogy ez egy jól irányzott echo parancs, összerak egy szép hosszú valamit és echo-zza a /var/spool/sms/out-ba.

Figyeljünk arra, hogy a fájl neve nem mindegy, mert hiába akarok kiküldeni 80 SMS-t, ha egymást felülírja és a végén egy fájl marad ugyanolyan néven, ezért a fájl nevet is úgy rakom össze, hogy a kontakt neve is benne van, a host neve és a szolgáltatás is. Ezzel szépen el lehet küldeni az SMS-t.

Az első sor az a contact pager, utána új sorba nyomja a többi szöveget. Egyre felhívnam a figyelmet, mielőtt nagy karbantartás van ilyen UPS és betáplálás környékén, húzzuk ki ezt a berendezést. Lehet olyat is csinálni, hogy a Nagios-ba előre begépeli az ember scheduled downtime. Általában ez akkor jut eszünkbe, hogy odaérünk még kihúzni, de már begépelni a Nagios-hoz nem.

Tud passzív szervíz ellenőrzést, elosztott rendszert, sok mindent a Nagios, sőt tud olyat, hogy eszkzaláció. Ha kiesik valami, akkor szólok a rendszergazdáknak, ha egy óra múlva sem vagy ötödik értesítés után sem történt még javítás, akkor szólok a főnöküknek, ha még öt után nem történik, akkor szólunk a HR-nek.

SNMP monitorozásra én Cricket-et használok, sok egyforma eszköz monitorozására praktikus. Szép diagramot rajzol ez az nrpg leszármazottja végül is. Lehet például, hogy UPS szünetmenteshez definiálok, hogy így néz ki egy APC és amikor már használni akarom, akkor ennyi a konfigurációs fájl, ha érkezik egy új switch, amit monitorozni akarok, akkor ide egyetlen egy sorba beírom, amiatt mert ez könyvtárszerkezetben is tök jól van, a részleteket meg majd itt a slide-on. Viszont leszoktunk arról, hogy a host-okat is.

Régen még volt ilyen, hogy mrtg-vel néztük a CPU/diszk terhelést a szervereken, erről leszoktunk, Munin-t használunk, ez úgy működik, hogy van a központban egy a crontab-ban föléled ötpercenként végigkérdezi a konfigurált host-okat, a klienseken meg fut egy kis daemon és akkor meg van, hogy ki jöhet hozzá, lehet SSL kulcsot is cserélni stb. Akkor a központ megkérdezeti ezeket és egy rrd-ben eltárolja stb. Az, hogy milyen paramétert figyel, azt roppant egyszerűen a kliens tudja konfigurálni. Azért praktikus, mert ha felhúzzok egy virtuális gépet és odaadom a lepketenyésztő tanszéknek, akkor a lepketenyésztő tanszék maga állítja be, hogy mit monitoroz,

ahhoz csak az ő gépét kell átállítani, egyszerűen symlink-eket kell létrehozni Munin plugin könyvtárába, hogy ezt is használom azt is használom.

Például az interfész statisztikáknál általában a neve is számít, tehát egy if\_ a program neve, ha eth0, akkor az eth0-t monitorozza, ha VLAN5, akkor a VLAN5-öt monitorozza. Nagyon jó az API-ja, könnyű plugin-t írni. Így veszem föl, csoportokba lehet szervezni, hogy ki kicsoda, tehát hogy szolgáltatás stb., hova tartozik. Ilyen debug dolgok, hogy lehet ilyet írni. Van egy WEB-es felület, ahol így látom körülbelül a gépeimet, nagyon kellemes, mert ami sárga az warning-ban van, ami piros, az meg error. Ez egy reggeli kis rutin, hogy megnézem a Munin-t utána megnézem a linux-aimat és utána lehet dolgozni.

Látszik, hogy a pityu5-ön a ventilátor fordulatszáma az túl alacsony. Például a ventilátor az úgy van megcsinálva, hogy egy tól-ig fordulatszám között normális, ha afelett van, akkor warning, mert melege van a gépnek. Ha még nincs melege a gépnek, de már nem tudja tovább emelni a fordulatszámot, akkor hamarosan melege lesz a gépnek. Viszont egy bizonyos fordulatszám alatti az error, mert az azt jelenti, hogy az a ventilátor kiszállt a játékból valami miatt. Nyilván ez olyan vas, hogy odarohan az ember és kicseréli anélkül, hogy leállítaná a gépet, de ez egy másik történet. Így azonnal lehet látni, hogy mi történik.

Még egy dolgot, így néz ki az, amit mondtam, ez egy UML-es gép története. Lehaldoklott és látszott, hogy a memória kezd kevés lenni, virtuális gépnél egyetlen jól irányzott paranccsal megnöveltük a memória méretet és túléltek. Látszik, hogy egy-két csúcs esetén beleszalad a 256 MB-os limitbe, akkor nyilván megpusztult szegény gép. Tud e-mailt küldeni a Munin is, ha warning vagy error van, de ez lényegesen butább, mint a Nagios, össze is lehet kapcsolni.

**Nyilvántartások:** hány és milyen gépünk van? Erre a legegyszerűbb válasz az úgynevezett arpwatch nevezetű program. Ha egy broadcast domain-ben van a tartományom, akkor elindítom, promiscuous módba lépteti a kártyát és figyeli az arp kéréseket. Ha felbukkan egy új páros, akkor küld róla egy e-mailt a, roppant borzasztó egy bizonyos méreten túl. Az újabb verzióban van olyan is, hogy SNMP-n lekérdezi például a router-nek az arp tábláját, ez már hasznosabb egy szétszedettebb házon is és egy idő múlva kialakulnak olyanok, hogy MAC address IP cím párosítások. Akkor jó, ha telefonál a user, hogy IP cím ütközés van és akkor ebből vissza lehet nézni, hogy ki az, aki rosszkodott. Most látható a MAC address, azt éppen látod az arp táblában, hogy ez, de ki volt ez régebben? Ezt így akkor meglehet nézni. Ennél nagyságrenddel szofisztikáltabb a netdisco. Hasonlít így ránézésre ez a grafikon a Nagios-hoz. Lényeges eltérés van a kettő között, a Nagiosnál én mondom meg, hogy ki kinek a parent-je, és úgy rakja össze az ábrát. A netdisco pedig könyörtelenül begyalogol a hálózatba és földeríti. Ez viszont a valóság. Volt már olyan ábránk, hogy keresztben átment egy 10 megás link, és kiderült, hogy a user a két rendezőszekrény két végéből össze patch-elte, tehát valahogy csak elintézte a dolgot.

A Muninhez még egy szót. Ha elég sok host-ot beraktok, mert a Munin az úgy megy, hogy lefut öt percenként, begyűjti az adatot, utána feldolgozza a warning-error-t, meg legyártja az ábrákat. Ez néhány tíz gépig teljesen jól működik, de a fölött kezd belepusztulni a gép, hogy az RRD-ből PNG-ket gyárt. Van egy olyan üzemmódja, hogy on-demand, tehát amikor ránézek, akkor állítja elő csak az ábrát. Normál cron futásnál nem csinál ábrákat, az RRD-ket gyűjti és amikor odamegyek a weblapra, ellenben ez akkor nem lesz túl gyors, akkor állítja elő az ábrákat, nyilván cache-eli. Viszont elég agresszíven cache-eli, tehát a shift-reload-ról ne feledkezzünk meg.

Kicsit csal a netdisco. Nem tudom ismeritek a Cisco-discovery protokollt, ez a Cisco saját protokollja. Régebbi HP eszközök is tudták. Most van helyette egy IEEE szabvány új és tudni fogja a következő netdisco is. Az a lényeg, hogy ez egy layer2 protokoll, üzenetnek egymásnak hello-kat és lehet tudni, hogy ki kinek a szomszédja. Tehát, ha nekem egy Ciscos hálózatom van, feltételezzük, hogy az SNMP community-k, remélem senkinél nem a public meg private gyári alapértelmezett, de mondjuk lehet egyforma egy társaságba tartozóknál, de be lehet konfigurálni, megmondom az SNMP community-t neki meg egy induló device-nak az IP címét, és ő onnan

feltérképezi a hálózatot. Csinál teljes eszközléltárt, hardver IOS verzióval stb.-vel szépen nyilvántartást.

Itt van például egy régebbi állapot a mi hálózati eszközeinkről, tehát ilyen szinten. Ez a Products 695, meg itt lejjebb ezek az akkor még számára nem ismert 2960-as Cisco-k voltak. Ezt utána be kellett írni SNMP NIP-be és utána tudta, hogy mit jelent ez az eszköz. Tehát csinál egy ilyet.

Aztán ami roppant hasznos, hogy SQL-be dolgozik, PostgreSQL-be. Nagyon világos és egyszerű a táblaszerkezete, mindenféle dokumentáció nélkül el lehet igazodni rajta, hihetetlen jó dolgokat lehet csinálni és archiválja. Leszedi a layer2-ről a MAC táblát, rendszeres időközönként és a layer3-ról az ARP táblát és ezt összegyűjti. Ezt kihasználva adatbázis alapján layer2 traceroute-ot tud csinálni. Hibakeresés esetén adatbázisból megvan néhány másodperc alatt és ez egy jó feature. A mostani utolsó stable release-ben ez a layer2 traceroute a menüben HTML szinten ki van kommentezve, nyugodtan vissza lehet rakni.

Van aki a Cricket helyett a Cacti-t használja inkább. A netdisco és ilyeneknek a környékén ez a jffnms, ez a Just For Fun Network Management System-nek a rövidítése. Érdekes elnevezés. Az NUS pedig megpróbálja összegyűjteni, amit eddig láttunk, Cricket, Nagios és ilyeneket. Mi még azért nem tértünk át rá, mert a Munin plugin-eket nem tudja kezelni.

## Lajber Zoltán és Csillag Tamás: Csoportmunka (webmail, címtárak, naptár, üzenőfal, nagy fájlok küldése, stb.)

**Csoportmunka eszközökről lenne szó**, de ebben nem vesznék el a részletekben. Ez az a rész, ahol túl sokat beszél bele a user, hogy mit akar, úgyhogy ezt a témát picit nagyoljuk, de találunk benne olyan részt, amivel lehet időt húzni. Az elején próbáljuk azt, hogy mit értünk csoportmunka alatt. Manapság ez egy nagy buzzword és mindenki szereti, hogy együtt működés, meg csoportmunka. Szerintem tapasztalat alapján, néhány kisebb-nagyobb cégnél tanácsadóként is sikerült dolgoznom, arra mindig ügyelek, hogy a userek ne tudják a telefonszámom és ne legyen közöm a desktop gépekhez, tehát van egy interfész a userek a desktop gép és köztem. De van 10 fős, 50-100 fős, van amelyik kimondottan műszaki beállítottságú valamelyik teljesen bölcsész orientációjú cég és ez alapján, anélkül, hogy statisztikákat és piaci diagramokat mutogatnék, így beszélnék róla.

**Most akárki akármit gondol a csoportmunkára, nagyrészt ez ma mailt, levelezést jelent szépen magyarul.** És a levelezés ma már olyan infrastruktúra szolgáltatásnak megy lassan, hogy legalább olyan fontos, sőt fontosabb, mint a web. Sajnos nagyon sokan keverik és arról tudnánk regélni, hogy az SMTP és FTP betűszavak mit jelentenek, és a userek ezt nem tudják, ezért fájlátvitelre használják a levelezést. Ez az egyik probléma, a másik pedig az, hogy a legtöbb felügyeleti szerv sincs tisztában azzal, hogy az SMTP, egy nem megbízható, tehát protokoll definíció szerint nem megbízható szolgáltatás, hanem best effort. Tehát ha én elküldök egy emailt, akkor a user vagy megkapja vagy nem. Ha kérek visszaigazolást, akkor vagy a levelet nem kapja meg vagy én nem kapom meg a visszaigazolást, és egyébként is az összes kombináció előfordulhat. Még az is, hogy megkapom a visszaigazolást, de a user nem kapja meg a levelet. Persze van arra informatikai megoldás, hogy teljesen korrektül, megbízhatóan és hitelesen üzeneteket továbbítsunk, de szerintem senki nem akar X.509-et implementálni és üzemeltetni.

A munka dandárját nekünk a levelezés adja és ezzel is éppen elég gondunk van, egyetlen egy szót említenék a levelezés környékén: spam. Előfordulnak olyanok, hogy címlisták, itt mutatok majd egy trükköt, hogy az általam használt rendszerben, ami instalkor nem biztosítja ezt, hogyan lehet csinálni egy egyéni és egy céges közös címlistát, amiben könnyen lehet keresni. Idáig még használják, ez egy gyakori dolog. Kevés helyen használják már például a tennivalólista dolgot. Elvileg ez jó lenne, mert priorizálni lehet, tovább lehet passzolni másnak, de ezt komolyabb rendszereknél úgy hívják, hogy ticketing, hibajegy-kezelés magyarul. De az ehhez hasonló dolog, a mi szempontunkból. A felhasználó szempontjából nem biztos. Néha még használják a naptárt, de ahhoz képest, hogy szinte az összes általam telepített rendszerben ott van a naptár funkció, most nem úgy, hogy beírom a cal parancsot a bash-ba, bár ez nem rossz különben, de van ilyen webes különböző protokollon szinkronizálható stb., megosztható naptár, legfeljebb a 10-15%-a használja a felhasználóknak. Bár van olyan cég, ez részben vezetés részben a hát őt már nem is hívnám titkárnőnek, mert ennél sokkal jobban képzett, kézben tartja a céget, csinálja a naptárakat. Amikor elkezdett belepiszkálni mások naptárába, meg a főnök is azt mondta, hogy most lesz a megbeszélés és akkor azt mondta valaki, hogy nem tud jönni, akkor azt mondta, hogy ez nem látszott a naptáradból, akkor kialakult és ők használják is a naptárat anélkül, hogy túlbonyolítanák és teljesen jó dolgokat csinálnak, de ez elenyésző sajnos.

Az előbb említett SMTP, FTP feloldása jött az, hogy a spam filterünk és vírusfilterünk védelmére korlátozzuk a küldhető maximális levélméretet. Inkább mi dobjuk el, mint másvalaki szó nélkül, tehát inkább mi küldjük ki a felhasználónak, mint más valahol szó nélkül vesszen el. Viszont ilyenkor reklamálnak, hogy mit tudnak csinálni. Erre van a nagy fájlok küldése vagy más néven poste restante. Elvileg GPL-es, gyakorlatilag egyetlen PHP fájl és egy PostgreSQL, de különben mindegy milyen SQL, egy egyszerű tábla és ennyi az egész. Alapvetően válasszuk ketté a dolgot, ez nem mindig sikerül. Ugye van a Mail Transfer Agent és van a Mail User Agent. Hogy csináljuk mi ezt a levelezést? Csináljuk azt, hogy a DMZ-ben lévő gépünkbe berakunk egy SMTP gateway-t, ahol nincsen se user, se semmi és ott folyik a spam és víruszűrés, tehát oda esnek be a levelek és ha

ezeken átjutott, akkor bentre odaadjuk valamelyik belső szervernek. Ott meg vannak a felhasználók. DMZ-be ne nagyon rakjunk olyan gépet, ahol például felhasználói adatbázis van, mert ha véletlenül sikerül betörni, legalább ne legyen címlistájuk. Az, hogy ki milyen MTA-t használ, mondhatnám, hogy ízlés szerint, ma gyakorlatilag a Postfix, Exim, Qmail versenyez. Az, hogy ki melyiket használja, ez gyakorlatilag olyan, mint a linux disztribúció, mikor a linux kezdőn mindig előjön, hogy melyik disztribúciót használja, mert a helyes válasz: amit a hozzánk legközelebbi linux szakértő használ. Azzal boldogulsz leghamarabb. Nagyjából mindegy, bár én a Qmaillel eléggé, túl sokat nem dolgoztam vele hál isten, de elég bonyolult a karbantartása, pach-elés és ilyesmi. Azt hogy Postfix vagy Exim teljesen más a gondolatmenete a konfigurálásnak, vagy az egyik vagy a másik tetszik valakinek, az ritka, hogy mind a kettő tetsszen neki. A jelenlévő előadók Postfix-eznek. Ennek az az egyik oka, hogy előtte még Sendmail-ezett, de az már régen volt mikor sendmail.cf-et kézzel írtunk. A legjobb Sendmail-es történetem az volt, hogy IRIX 5.2-es op. rendszeren Silicon Graphics unix sendmail.cf-et írtam kézzel, mert olyan volt a cég, hogy egyik irodája itt volt Budapesten, a másik Amerikában és a házon belüli levelezés úgy ment, hogy, hogy mondjam, tehát az amerikai idő szerinti csúcsidőn kívül betárcsázott az amerikai gép ide UUCP-n a céges leveleket átmásolta a queue-ból, amúgy meg interneten levelezett mind a két cég a saját gateway-én.

Az IMAP az annyiból jobb, hogy a szerveren lehet tárolni további foldereket is. Ez a fel bírom emelni a gépet meg a user környezet nem kontrollált. A POP-on leszedi a saját gépét, a saját gép az kihál, elviszi, nem ott van, hazamegy, nem tudja elolvasni. Legyen IMAP-en és fenn a szerveren, azt én majd mentem, visszatöltöm, meg intézem, semmi köze ne legyen a felhasználónak ehhez lehetőleg. Nyilván archiválni lehet, local folder és leírom, az a user dolga. Régebben én Couriert használtam, nem MTA-nak, hangsúlyozom, migráció során találkoztam csak vele. Viszont nagyon sokáig IMAP-re Couriert használtam, van POP része is, de aztán több helyen teljesítmény problémám volt és megkérdeztem egy szakértőt, aki azt mondta, Dovecot-ot hogy kell felrakni és ez bevált. Mindenféleképpen a maildir formátuma hasznos, igazán nagy levelek nálunk nincsenek, tehát egy user mailbox ritkán tartalmaz, egy folder ritkán tartalmaz 4-5000 levélnél többet és ilyen folderből néhány tíznél több nem szokott lenni, ezeket jól ki tudja szolgálni. A user adatbázis viszont valamilyen formában (van példa rá, kisebb esetben, hogy a szabvány unixos userek) inkább valami adatbázisban tartsuk, az LDAP ilyen szempontból roppant kellemes. Mert a mail gateway is tudja használni és a nem létező userének érkező levelet már ott el tudom szórni és nem kell beküldenem és vissza, nem kell kétszer járatnom stb. A UNIX user sem teljesen elvetendő ötlet, annak ellenére, hogy itt nagy rendszerekben gondolkozunk. Több kisebb cégnél van úgy, hogy a belső szerver, amelyik egyben Samba is, meg sok minden másra használják, azon igen is UNIX-on léteznek a userek. Vagy megcsinálom úgy a UNIX-om, hogy LDAP-ot használ, de ehhez meg már régen készültek a gépek, hogy LDAP-pal menjen a UNIX is.

**Mi történjen a telephelyen kívül tartózkodó felhasználókkal?** Erre én azt mondom, hogy használják a webmailt. A mai spames világban annyiféle szűrést, szolgáltató hogyan szűr, ki mit enged, hogy ez roppant nyűgös kifelé szolgáltatás. Az SMTP-t kellhet, hogy hiába is autentikáltatom, mert mondjuk az internetszolgáltatója már szűri. Nekünk ez elég, hogy elérem a mailboxomat IMAP-en és keresek valami SMTP relayt ahonnan én elküldöm. Az itt ülők ezt meg tudják oldani, az átlagfelhasználó az, hogy IMAP meg SMTP, az bonyolult dolog, használja a webmailt. Azt a stratégiát követjük, hogy vannak ugyan tiltott dolgok a hálózatban, de ebből viszonylag kevés legyen, tehát ne legyen az, hogy hivatali allűrjei vannak az informatikának. Vannak határok. Ezt mi úgy próbáljuk a legtöbb helyen megoldani, hogy vannak a támogatott megoldások, ilyen például a webmail. Ha egy user azt mondja, hogy nem tudja elolvasni a leveleit, akkor bemegy a webmailre, ott van a levél, látja. Ha ezen kívül Outlookkal levelezél és nem működik, akkor így járt. Mert ezt nem támogatjuk, bár expliciten nem tiltjuk. Oldja meg! Az Outlook-osokat a legegyszerűbb úgy lefegyverezni, hogy jön a user, hogy nem megy a levelezés, mivel levelezél, Outlookkal. Visszakérdezünk, hogy Outlook vagy Outlook Expressz?

Ugye azt a user nem tudja, hogy ez két teljesen különböző termék, itt általában megállt a tudomány és innentől nyerő helyzetben vagyunk és mi jövünk. Tehát SMTP autentikációval lehet a Dovecot-Postfix páros elegánsan néhány soros konfigurációval. Van még a submission port, az annyival szerencsésebb, hogy egyrészt butább, mint az SMTP, tehát kisebb veszélyeket hordoz, másrészt a szolgáltatók még nem szűrik, úgyhogy ez is egy javasolt menetirány, de azt sem javaslom. Vagy ha olyan cégünk van, hogy tele vagyunk road warriorokkal, akik járnak szanaszét a világban és mindenfélére használjuk, akkor csinálunk VPN és azon át megy, amit mi akarunk.

Teljesen életszerű, hogy a szolgáltató kiszűri az SMTP-t. Igen, de van egy külön submission port, az 587-es port, azt is persze ki lehet szűrni, de ezt általában nem szokták úgy szűrni, nem véletlen, hogy a különböző netbios, meg különböző smb, ezek ki vannak szűrve, mert a szolgáltató egyrészt a userét is így próbálja védeni, de nyilvánvalóan az 587-nek más a felhasználása. Ez az, aminek a Postfix main.c-ben kell szerepelnie. Ez meg, ami a Dovecotban szerepel, a Dovecotban van egy socket, ezen a socketen figyel a Dovecot, és ezen a socketen nagyon egyszerű szöveges protokollal, nem Courier, hanem Cyrus. A Cyrus az egy jó rendszer, csak nagyon összetett és komplikált, ez az én megközelitésem, ott nagyon nehéz bináris protokollal kell kommunikálni, óriási nagy librarykat include-olni, itt egy egyszerű szöveges protokollal megy az autentikáció. A Postfixhez fut be a kérés, ő SMTP szinten ezt lekezeli, ugye SSL felett, meg is van mondva, hogy csak SSL felett lehet autentikálni illetve TLS felett. És TLS felett bejön a user, megmondja a felhasználónevét, jelszavát és a Postfix meg fogja kérdezni a Dovecotot hogy mondj erre valamit, mond egy felhasználónevet, meg egy jelszót, a felhasználónév általában a domainnel együtt van, mindenhol így van kezelve, hogy a teljes domain nevet adja meg userként, így több domaint is lehet kezelni és ez alapján a Dovecot azt mondja, hogy igen vagy nem. Szóval az SMTP port az teljesen felejtős, mert csomó küzdelmünk lesz vele, ez a submission port ez még járható.

Illetve még ami szóba jön az Outlook Expressz meg a társai miatt, az a wrapper módban az SMTP port, az a 465-ös porton van, ott eleve SSL-lel megy, mert itt a TLS-en, start TLS van, amikor azt mondja a kliens, hogy én most TLS-en szeretnék beszélni veled.

Az a mail gateway ami a DMZ-ben ül, ne relay-ezzen a hostoknak, az csak a mail szervernek vagy mail szervereknek a bentről, tehát jól definiált környezetnek tűnik ezzel. Volt már, hogy roppant praktikus volt, hogy kifelé spam filterezünk. Ez a phishing időszak, jött az email, hogy küldjétek el a jelszavakat meg a user neveteket, szerencsére van egy olyan userünk, aki az ABC-ben elég elől szerepel, szerencsére jól képzett és gyorsan szólt, hogy jött ilyen levél, gyorsan ráütöttünk egy szabályt, hogy ezt a levelet gyorsan elfogjuk. A log szerint négy ilyen levél kiment, amiről nem tudjuk, hogy mit tartalmazott, viszont a többit megfogtuk. Hál Isten azt tudom mondani, hogy a 6000 aktív userből 10-en válaszoltak, ami nem rossz arány, és ezek is mind olyat válaszoltak, hogy ezt most komolyan gondoljátok vagy hogy van. Ez a Google-lel fordított hihetetlen magyarságú, borzasztó szöveg volt. De volt már ilyen, szomszéd intézmény kapott be angol nyelvűt, ahol jól képzett kutatók válaszoltak. Aztán azt vették észre, hogy belépnek a webmailre, megváltoztatják a jelszavukat, felvesznek a címlistába ilyen küldőlistákat és egy óra alatt 200 000 levél simán kiment a rendszerből. Szerencsére kicsit öreg vas volt és ennyit tudott kilapátolni egyszerre.

Gödöllő, Szent István Egyetem: Hol csináljuk mi a levelezést? Egyetlen központi mail gateway-ünk van, ez rektori utasításban megvolt, hogy csak ezen keresztül mehet ki-be levél. Ezt a rektor akkor írta alá, mikor a nemzetbiztonsági hivatal emberei még nem értek ki a kapuhoz. Addig mi hiába írtunk ilyen szabályzatokat senki nem gondolta, hogy ez komoly, amikor bejött három ember és azok közül kettő ilyen füldugóval: igen, már a helyszínen vagyunk jelentkezések mentek, utána már lehetett beszélni a vezetőséggel, hogy mit szeretnénk és hogyan kellene ezt csinálni. Amúgy nem kívánom senkinek a főnök lelkivilágát, amíg az alagsorból felért a harmadikra a rektorhoz, hogy itt van a rektor és a nemzetbiztonságiak, gyere fel légy szíves. Kicsit átgondolta, hogy mit üzemeltet és mi lehet a probléma.

A legnagyobb balhénk teljesen tréfás dolog volt, szintén migráció során talált Mercury levelező, gondolom ismerős, MTA Novell, rendszergazdája régen távozott az intézményből, egy kis sufniiban ez a 3.11-es Novell Mercury ez évek óta ment, persze open relay volt, a Mercurynak ez szokása.

Spam az dőlt ki róla, ezt már tudtuk is, mert gyanús volt a forgalom. Egy napig őriztük a leveleket, majd rájöttünk, hogy ez a 10 fős tanszék nem fog napi 30-40 000 levelet küldeni, tehát egyszerűen lekaszáltuk olyan szinten, hogy háló szinten tiltva volt az IP. És kész, ennyi. Majd megjelent a nemzetbiztonsági, ugyanis egy olyan spam ment ki erről, de jellemző, hogy hogyan jönnek rá, hogy hogyan és honnan jöhettek rá? Elhagyott a barátnőd? Nem jó a kedved? Add meg a Visa kártyád számát és 10 dollár értékben küldünk neked marihuánát. Most erre a marihuána szóra egy kereső valahol magas helyről rázuhant, ugye lefelé terjedés, nő a pofon, és így jutottak hozzánk, hogy ki hol árulnak az egyetemen marihuánát. Ez volt az első kérdés? Erről szó nem volt. Amúgy ez a levél a Visa kártya szám megszerzéséről szólt és nem a marihuánáról, de mindegy. Ezek után lehetett tárgyalni a vezetőséggel.

Egyetemhez illően ott fut a linux kezdő, a linux lista, linux flamet nem is mertem felírni a számokat róla. 2005-ben volt 2553 tag és körülbelül napi 100 levél minimum, a linux kezdőn is van, hát most annyi nincs, de a flame, az húzós ilyen szempontból. Mennek levlisták is és azokat nem olyan egyszerű. Meg hát 93 óta internetezik az egyetem, pont ezek a gau.hu-s régi domain nevek, és hát egy email cím az nem olyan, hogy csak úgy megszűntetek vagy domain, azt egy darabig kezelni kell. Elég szép számú listán szerepeltünk, úgyhogy foglalkozni kellett vele. Az SMTP gateway története: hát az első volt egy DEC MicroVAX MXmail, ha valaki ismeri ezt a rendszert, elég felhasználóbarát, csak elég rendszeren megválogatja a barátait. Aztán lett egy SUN Enterprise Ultra 250-es, gondolom ismerős gép, ez se mai gyerek, egy Solaris volt rajta, Sendmail Trend Micro VirusWall, ez régi. A mostani termékeket nem akarom minősíteni. Részben a forgalmazó is bűnös volt benne, de a termék is, sokba került az open proxy. Gyanús volt, hogy miért rakunk ki egy nem korrekt MTA-t 25-ös portra. Jó, hogy víruszűrő, de ne ez hallgasson a 25-ös porton, ott egy rendes MTA hallgatózzon. Tehát így csinálták meg, mert mondták, hogy másképp nem lehet. Akkor főnökkel nekiláttunk, felcsaptuk a kézikönyvét a Trend Micronak és másnapra futott úgy, hogy szendvicsbe rendszeren, ahogyan kell két Sendmail között.

Ezek után is sorba kerültünk be a különböző adatbázisokba, míg rá nem jöttünk, hogy a menedzsment felületének a portja open proxy autentikáció előtt. Nem is tudtuk, hogy azon a porton egyáltalán figyeltek, de ugye ehhez kellett valami log. Ráadásul a Trend Micro minden programrésze csak root-ként fut. A bigadminon csak úgy hívják, hogy mother-of-bufferoverflow. Aztán áttértünk egy másik megoldásra, gondolom ismerős ez a dolog, meghagytuk a Trend Micronak a víruszűrő részét csak szépen beszendvicseltük a Postfixbe. Beraktuk a spamfiltert, mert a környék processzora nem lett volna elég különben, ahogy nőtt a forgalom, hogy a víruskergető az kegyetlenül kezdett belassulni.

Azzal, hogy a spamfiltert elé raktuk, ettől fellélegzett a víruskergető, mert a levelek nagy részét már megfogta a spamfilter. Ez így tök jól működött egy darabig. Elkezdtük betanítani, amíg például egy járműgépész tanszéknek, 25 ember betanítottuk a spam adatbázisát. Nem olyan egyszerű, nagyon könnyű túltanítani, túl kevés van, túl sok van. Hát matematikus ismerős hölgy, az nagyon jól kezeli, olyan programokat írt rá, de ez tamagocsi jelleget öltött, hogy mindig spamet tanítunk és ez ugye ilyen viagrás stb. hormonos korszakban, képzeljétek el, amikor bejön az állatorvos tudományi kar. Akkor a gyógyszernev meg az, hogy pénisz, az nem spam. Itt körülbelül feladtam a spamfilter dolgot.

Ezt a Postfixes dolgot linuxon teljesen könnyű telepíteni, Solaris-szal kicsit motorozgatni lehetett, de lehetett, még ilyen Solaris 8, Solaris 9-ről van szó. Szabályrendszer hangolásokkal küzdöttünk, de ebbe nem mennék bele. Kezdtük kinőni ezt a hardvert, olyat kerestünk, ami Solarison fut. Mire eldöntöttük és sikerült megvásárolni, addigra Linuxra tettük és nem Solarisra, de mindegy, kiválasztásnál szempont volt, hogy Solarisnál is fusson. Teszteltük, ez kereskedelmi termék, nem open source, itt látható a neve, ezt teszteltük és bevezettük. Nincs túl jó véleményem a pénzes szoftverekről, az a termék, aminek ott látszott a neve, meglepően jó. Például nem spilázza túl: van egy webes kattintgató felülete, mikor legelőször látod még hasznos is, de vannak Perlben írt magas szintű parancsai, és például a webes felület is, tudod, hogy itt van a Sieve szűrő és abba vannak Sieve nyelven leírva a szabályok. Kirakja egy fájlba, kiírja a webre, hogy ez a fájl, odamész vi-ga vk megnyomom a régi whitelist-emet, szed vissza a fájlt és mehetünk tovább az editálásba.

Tehát teljesen korrekt és a man is úgy néz ki, hogy kattints ide, ha nem itt van, akkor ez a magas szintű parancs vagy ebbe a fájlba ez az opció. Végig le van írva szépen. A supportja az azt mondta a harmadik email után, hogy itt a Skype azonosító csörögjek be. Bejelentkeztem Írországba majd a második mondatnál a srác megkérdezi, hogy beszélhetnénk-e magyarul? Persze.

Azért óvatosan duhajkodunk ennyi felhasználó email rendszerével, úgyhogy a tesztüzem már eleve megvolt korábban, a régi mail gateway maradt az éles, de néhány, kevésbé kritikus SMTP szerver forgalmát az új gép, ugye tudtuk a Postfixen állítani, hogy ki a transport table-ben, merre menjen. Akkor ott tudtuk, hogy ott viszonylag intelligens userek vannak stb., az a subdomain menjen erre, de úgy, hogy nem szűrte meg karanténzott, csak jelölt subjectbe, header-be stb. Úgy volt, hogy jól működik, úgyhogy akkor áttértünk az éles üzemre, ilyenkor már el is dobhatott leveleket. Abszolút nem volt szempont, viszont bevált és abszolút rászoktak a usereink, hogy karanténolni tud, amit ő spamnek vagy vírusnak minősít, azt berakja egy karanténba. Ez úgy működik gyakorlatilag, hogy egy post-queue-ban ottmarad, félremásolva egy queue formátumba és egy PostgreSQL adatbázisba leindexeli, amit weben keresztül a user menedzselhet, mivel LDAP-ban vannak a usereink, be tud menni, meg tudja nézni a karanténját, kipipálja és tovább tud menni. A vírusost is tovább küldheti magának.

A karantén alatt most nincs túl sok diszk, mondjuk 20 giga van erre és 14 napot tudunk tárolni. Erre én azt mondom, hogy amiről 14 nap alatt nem derül ki, hogy fontos email, az nem volt fontos email. Az a lényege különben, ez egy PMX Filter nevezetű, ő tud telepíteni Postfixet vagy a tiedet használok. Most már az övét használjuk, mert volt egy-két probléma még az oldstable debianos Postfixszel például, nem szerette, ha egy levélben 3268-nál több main szeletből állt, volt ilyen levelünk, ami jött és a szeme fennakadt kicsit időnként. Itt van egy összehasonlító táblázat, hogy hogyan szűr. Ha úgy állítod be a szűrőt, hogy a 80%-át megfogja a spamnek, akkor nagy valószínűséggel nem lesz fals pozitívod. Ugye ez sokkal szimpatikusabb, mintha azt mondja neked valaki, hogy minden spamet kiszűrünk, az oké és még mit? Az nem nagy tudomány minden spamet kiszűrni, de mellette...

Ez jó régi, egy 2005-ös vason, ez egy SUN Fire v20z Opteronnal, 4 Giga RAM-mal, azóta már nem ez van alatta persze, ez mondom akkori állapot. Tehát 17 000 tiszta levél 18 000 spam, ez a napi termés körülbelül. Ráadásul hétvégén a usereink pihennek és kevesebb tiszta levél megy át, ellenben a spamek ugyanúgy jönnek, ez is látszik. Azóta, ezek nagyságrendet változtak ezek a mennyiségek, a gép is 8 Giga RAM-os, dual Opteronos, kétmagos, SUN Fire ugyanúgy csak 4100-as van most alatta. Körülbelül másodpercenként 10 email esik be, most bekapcsoltuk azt, hogy policy szinten 25-ös porton, ahhoz, hogy eldobja a levelet, mert hagyja tornázni az ügyfelet, a hello-nál eldől, hogy szóba állunk-e vele vagy sem, de nem elzavarja ilyenkor, hanem hagyja, hogy beszélje végig, oszt úgyis eldobja a levelet. Sőt ilyenkor felveszi a timereket annyira, hogy hadd dolgozzon, tehát mindig megvárja az SMTP timeout-ok előtt szól, tehát lefogja egy kicsit a küldőt, minket annyira nem, mert mi várunk és úgyis eldobjuk utána. Itt körülbelül a 10-ből 7 elvérzik SMTP helló szinten (ez Postfix), a bejövő 3-ból kettő spam, másodpercenkénti mennyiségeket mondtam. Nem tökéletes az EML kiterjesztést, azt nem szerette átengedni, volt ebben igazság, de erre rá kellett beszélni. A jelszavas fájlokat alapesetben, tehát jelszóval tömörített ZIP vagy a jelszóval védett egyéb fájlokat azonnal berakja a suspect, tehát gyanús címkével a karanténba és akkor tovább kell küldeni. A régebbiek még csak kiterjesztés alapján szűrtek, már egy éve legalább olyan, hogy bele is néz a fájlba, tehát EXE-t nem tudnak küldeni, hiába nevezik át DOC-nak. LDAP-pal nagyon jó összegyógyítható. A partíciókkal ilyen Munin bug, lehetett látni, hogy voltak küzdelmek, hogy melyik partíció mennyire telített és mennyi van. Van Postfix plugin a Muninban, ez itt a mailq mérete, a kék vonal az a deferred queue, nem tudom mennyire ismerős a Postfixnek a működése, ugye vannak a normal queue és amit nem tud azonnal elküldeni, az a deferred-be került. A deferred queue mérete és a spamfilter hatásfoka között nagyon nagy korreláció van, gyakorlatilag a spam filteremet úgy tudom minősíteni, hogy melyike lesz kisebb.

Itt a piros nyilaknál, itt kezdtük el tanítani az AMaViS-t. Itt egy kicsit keményítettem a szabályrendszeren. Itt túl sokat sirtak a usereim és lazítottam a szabályrendszeren és itt adtam fel. Itt kezdtük el bevezetni ezt a terméket és itt álltunk át éles üzemre. Például van olyan probléma,

hogy a mailq-ba LDAP azonosítóval belép a user, akkor az ő mail címére, ami mail attribútum, arra menő leveleket látja, de a rendszer használ néha alternate address és mailenq address attribútumokat is, mint email címekeket, ezeket nem kezeli. Tehát az ilyen címekre menőt nem látja. Az admin látja, sőt az admin is több rétegű, és lehet olyan jogot adni, hogy látja a karantén fejléceket, de a levél tartalmát nem nézheti meg. Van ilyen jogosultság is, az überadmin az megnézheti a levél tartalmát is akár. Az admin felületen van olyan is, hogy vagy továbbküldöm a usernek vagy forwardolom egy adott email címre az ilyen levelet. Ez elég jól meg van csinálva. Nem kell saját hatáskörben szűrni. Az AMaViS-nál az volt, hogy betettük a subjectbe, hogy spam vagy kisebbség a headerbe stb. és ilyen szűrőgetések. Voltak ilyen megosztott folderek, ahonnan én szkripttel összeszedtem a pozitív meg negatív átcúsúzottakat stb., ezek mind megszűnhettek gyakorlatilag. Ráadásul a mail kvótában nem szerepelt az, ami a karanténban csücsül. Ez így roppant kellemes, hogy megvan a mail kvóta a usernek, de ami a karanténban ül spam, az nem számít bele a kvótájába. A karanténra meg egy globál partícióméret vagy filesystem-méret limit, azt be kell löni, ahhoz kell egy kis rutin azért. Ez működött. Itt ez az URL, ha hagyjuk róla az utolsó fájlrészt, akkor ott meg lehet találni, hát mondjuk most még nem, de mire hazaérünk addigra ott lesznek a mostani slide-ok is meg a régebbi elkövetett slide-jaim is. Vissza gruppenre, ha már erről volt szó.

**Mail User Agent:** én erre azt mondom, hogy webes a támogatott, azon kívül meg mindenki azt használ, amit akar. Például nálunk is van egy webes, ami támogatott, én konkrétan pine-ból használom. Más levelező nincs tiltva, de nem adunk hozzá segítséget, oldja meg ő maga, ami eléggé bevált nekem, bár volt egy rossz korszaka, stabilitásügyileg, az a Horde nevezetű groupware rendszer, PHP-ban írták, és ezen belül az IMP nevezetű alkalmazás az, ami a levelezést csinálja. Tehát a horde az egy keretrendszer, egy PHP alapú egész nagy keretrendszer, és ebből az IMP, ami a levelezés, van több ilyen alkalmazás, lesz mindjárt szó néhányról. Ez viszonylag jól bevált nekem, elég sok mindent tud. Ha csak tisztán levelezés kell és nem groupware, akkor gyanúsított most még a queue nevezetű, AJAX-os, egész pofásan kinéző. Az a szépséghiba most nálunk, hogy nem tudja kivenni LDAP-ból, tehát LDAP autent IMAP autentikációval belép, de nem tudja LDAP-ból kivenni a from-ot, tehát minden usernek magának meg kell csinálni, hogy mi legyen a from vagy a replay-to cím vagy ilyesmi.

Alapvetően úgy van beállítva a rendszer, hogy a Horde-ba nem hozunk létre usereket, hanem IMP auth-ot használunk, tehát úgy van beállítva a Horde, hogy eleve a levelező rendszer a belépő alkalmazás és ilyenkor a login ablakban gyakorlatilag IMAP auth-tal. Tehát azzal a usernév-passworddel kell menni, amivel az IMAP-et is eléri, utána persze a Horde rendszerben létrejön az ilyen nevű user és tárolja az adatbázisban a beállításait, preferenciáit, mindent, de nem foglalkozunk külön Horde user gyártással. Van erre a Horde-ban keretrendszer, de nem szeretem, mert ott olyan is, ha nincs, akkor csinállok. Be lehet állítani több mailboxot is, sőt az elején mindjárt lehet azt csinálni, hogy szerveret választani, tehát ha valakinek több szervere van, akkor ezt meg lehet csinálni. Legtöbb helyen ez konflúziót okoz a usereknek, tehát menjen az egy helyre és kész. Ami kellett és egy darabig nem volt megoldva, az a vacation, a szabadságüzenet, hogy elmentem szabadságra. Erre van egy csomag, fel kell rakni, tud procmail filtert vagy Sieve jellegűt, szépen le van írva a doksijában, hogy hogyan kell felrakni.

**Címlisták:** másik nagy téma vagy még közvetlenül kapcsolódik a levelezéshez. A Horde-ban van egy Turba nevű alkalmazás, ami ezt csinálja, gyakorlatilag PostgreSQL-be saját object tároló rendszerben tárolja ezeket. Az a tapasztalat, hogy célszerű, hogy minden embernek van egy saját levlistája és egy céges levlistája. Az a lényeg, hogy a sajátba mindenki azt ír be, amit akar és azt vesz fel, akit akar, a cégest, azt meg tipikusan a titkárnő vagy valamelyik menedzser karbantartja, és az összes alkalmazott, tud olyat csinálni, hogy dolgoznak mondjuk egy x projekten, akkor van egy x nevezetű alias és a projektben résztvevő mindenkinek tud levelet írni. De ez csak néhány usernek írható, a többieknek readonly ez a lista, egyik listából a másikba mozgathat át például elemeket. Nyilván ha van írásjoga a céllistára. Ez nem teljesen triviális ennek a két címlistának a megcsinálása a Horde-ba vagy a Turba-ba, tulajdonképpen le van írva a doksijában, de kicsit következtetni is

kellett. Ez egy olyan megoldás, ami még az upgrade-ket is túléli. Tehát működik. Nyilván ésszel. Csak config részletek, ugye van egy saját címtár, amit SQL-ben tárolunk, szokásos SQL beállításokat, tehát adatbázis szerver stb. neve az jön, és itt a lényeg, hogy adunk neki egy tábla nevet, hogy melyik táblában tárolja. Nyilván lusta ember voltam és a gyárilag adott Turba aláhúzás out-ex táblát azt én leklónoztam egy ilyen néven és az már működött is. És itt megmondjuk azt, hogy ez nem publikus, nem readonly, és használhatja a user. Ez a saját, személyes, mindenkinek van egy ilyen levlistája, itt a Kis Pista tud betenni a Kis Pista levlistájába, ugye mindig beírja ebbe a táblába, hogy ki ennek a jegyzéknek a tulajdonosa és akkor csak ő tud ezen változtatni, sőt csak ő látja ezt a címlistát. Készül egy közös is, ez a központi táblában tárolódik, közös címtár a címkéje, ezt látja, amikor kiválasztja a felületen. A lényege az, hogy nem az lesz a tulajdonosa, tehát nem csak a tulajdonos kezelheti ezt, nem foglalkozunk azzal, hogy ki a tulajdonosa ennek a bejegyzésnek, bárki hozzáférhet ehhez a címlistához. Viszont publikus, de read only. Magyarul mindenki, akinek van accountja hozzáfér readonly módon ehhez a címlistához, kivétel az adminok, akik írhatnak bele. Itt egyszerűen user neveket kell megadni, nyilván, ha ez domain-es, akkor azt a user nevet kell megadni, akik ezt írhatják is. Ennyi a trükk benne. Ehhez az kell, hogy amikor lefutott a Horde install, akkor azt a táblát nyilván le kell duplikálnom, most, hogy ezt hogyan csinálom, SQL stb., ezt kell megcsinálni és akkor ez működik.

**A TODO-hoz van a Horde-nak egy nag2 alkalmazása,** nem nagyon láttam használni. A naptárt azt néhányan használják, az egy kronolith nevezetű alkalmazás. Az egész dolognál ott jön a buktató, hogy hogyan és mihez szinkronizálom. Ilyen group szoftvernek mostanában eléggé nyomul az Zimbra.

Van fizetős és open source változata. Én feltettem a nem tudom, lehet, hogy nem open source, de free változatát. Hát ha valaki letölti a Debian csomagokat és megnézi, hány debian csomag és mi, nekem ugye ott nem tetszett, hogy az MTA-tól kezdve mindent az Zimbra csinál, tehát így én a free verziójának nem látom értelmét. Ugyanakkor nagyon jó terméknek tartom a fizetős változatát. Például, ha Exchange-ről kell migrálni, akkor a pénzes változatban csomó olyan eszköz van, ami ilyen konvertálásokra való, PDA szinkronra való, ezzel szinkronra való, azzal szinkronra való, de az csak a fizetősben van benne. Úgyhogy ha komplett szolgáltatást akarok, akkor van support minden, tehát az valószínű jó, bár ilyet nem üzemeltetek sehol, de jónak tűnik. Én úgy látom, hogy ahhoz, hogy az Zimbra igazán jó legyen, ahhoz a pénzes változata kell, az ingyenes nem nyerte el a tetszésemet.

**Nagy fájlok küldése:** egy usernek azt is nehéz megmagyarázni, hogy a mail rendszerben a mail kvóta bruttó 50 mega. És hogy mit jelent az, hogy bruttó? Header plusz mime-coding. Mi ezt tudjuk, hogy ez mit jelent, de hogy mondd ezt magyarul? Hát olyan 20 megás csatolmányok még átmennek. Kis rátartással, mert úgyis 25 megásat fog és az még átmegy. Ismerjük a user-t ennyire. Nagy fájl küldésére, erre teljesen rászoktak az oktatók is, például slide-okat raktak fel stb. Ez úgy működik, hogy van egy weblap, LDAP autentikációval fel tud tölteni egy fájlt és ami fájlt feltölt, az kap egy szám végű URL-t. Utána ezt cut&paste berakja a levélbe, elküldi, és ott van a fájl és töltheti le. Méret limit expliciten nem nagyon van ezen, bár mondjuk egy komplett CD-image-t nem lehet felrakni, ennek több oka van, azt már nem engedi, de inkább az van, hogy ADSL-ről, otthonról ki timeout-ol, nincs felvéve PHP-ba a szkript, hogy mekkorát tölt föl, ugye ismerős, PHP-ban be lehet állítani input, timeout stb. és akkor eldobja. Tehát ez egy limit igazából méretre. Nem kell neki túl nagy tárhely, 10 gigán elvan szépen. Ami nagyon lényeges, föltöltéskor be lehet állítani a fájlra az élettartamot 2 és 14 nap között, default-on 2 nap. Annyit csalunk, hogy nem óra, percre nézzük, hanem hajnalban fut le a karbantartó szkript, tehát ha valaki reggel feltöltött, annak majdnem fél napot még bónuszban ott van a fájlja, de ennyi. Hajnalban jön a szkript és kitakarítja a régi fájlokat. Még egy funkció van, lehet jeligét hozzátenni, mert elvileg az URL, az olyan, hogy nekiállhat a ráérő diák és próbálkozhat és valamit csak kap. Nem jelszónak, hanem jeligének hívjuk, hogy érezze a user, hogy ez nem jelszavas védelem, nincs olyan komoly védelme, de lehet adni jeligét, és

csak az tudja letölteni azt a fájlt, aki tudja azt a jeligét. Magyarul amikor e-mailben küldöm az URL-t, akkor küldöm vele a jeligét és akkor le tudja szedni.

Egyetlen egy komoly feature request áll a queue-ban, hogy ez fordítva is működjön, tehát, ha kintről, aki nincs az én LDAP-omban, ő akar nagy fájlt küldeni nekem, akkor én mondjuk tudjak egy slotot létrehozni, ahová ő feltöltheti a fájlt, de ez elég bonyolult programozás és használja a saját poste restante-jét, különben is open source és GPL és vigye. Most nem tudok tar.gz-t adni. Kell egy PHP, szerintem teljesen mindegy, hogy milyen PHP, kell egy kis tárhely, meg kell mögé, most PostgreSQL van, de valószínűleg más SQL-lel is megy. Ennyi a nagy fájlok küldése.

A mailmanban be lehet azt állítani, levlista szolgáltatás, nem elhanyagolható funkciói vannak a mailmannak, moderátor, admin stb., tehát ne felejtsük el ezt a levlista alapszoftvernek mondható azt hiszem a mailman. Abba van ilyen feature, hogy ha csatolmány jön, akkor beteszi az archívumba, a levlista archívumba, de a kimenő levelekben csak az archívum URL van. Ezt a mailman már elég régóta tudja, nem tudok most verziót mondani, de elég régóta tudja ezt a funkciót, két éve legalább. A másik, hogy azt nem tudom, hogy a Dovecot tudja-e, de amit mi használunk levelező rendszert, az például, ha ugyanaz a csatolmány jön több usernek, akkor egy fájl és elintézi a mailbe symlinkkel. Nem triviális a feladat, mert mi van, ha elkezdik törölgetni? Hol éli túl meg stb.? De megoldja. Ez működik.

## Lajber Zoltán: Fájlf- és nyomtatószerver (Samba - tipikus esetek, alapproblémák)

Erről a témáról először 1996-ban, Schönherzben, a Linux Konferencián, majd 2001-ben Sopronban a Network Shopon beszéltem, bár ott nem a Samba-ról, hanem az SMB protokollról, ezt annyival foglalnám össze, hogy az SMB protokoll nem követi a hálózatban szokásos layeres tervezési architektúrát, és van benne néhány furmányos dolog. Utána két Y előadást tartottam, annyiból érdekesek az Y előadások, hogy arról készül ilyen videofelvétel, az egyikén csak hang, a másikon csak kép készült, apró technikai problémák voltak. Az sokkal részletesebb, reggel 10-től délután 4-5 óráig tartó dolog. Most nem vesznék el a Samba részleteibe, ismételten elmondom, hogy nekem működik a Samba, de mindig nagyon egyszerű konfigurációkra törekszem és nem csinállok semmi különleges dolgot benne. Ez egy alapvetően jól működő stabil rendszernél egy lényeges dolog. Miért hívják ezt Samba-nak? Azért, mert az SMB protokoll egyik megvalósítása, a másik megvalósítása a dark side-ról jött.

**Az előadás tematikája:** röviden beszélünk a Samba-ról, aztán névfeloldás, böngészés, ebből mindenkinek van kalandja, akinek nem egy subnet-en belül vannak a gépei. Ausztrálok kezdték el fejleszteni, elég régen már különben, hogy windowsos szerű megosztást tudjunk nyújtani. Gyakorlatilag bármilyen TCP/IP képes op. rendszerre portolták már, elképesztő mennyiségű helyen fut már, a Mikrotiken nem biztos, de a StrongARM-on már igen kategóriájú. Van hozzá jó pár adminisztrációs és migrációs segédeszköz. Valamelyik listán fordul elő, hogy milyen konfigurációs eszközt használnak a guruk? És mondták, hogy vi pontosabban vim, tehát grafikusan Gnome-ban futtatott vim és azért ebben van némi igazság. Hál Istennek azért innen is öregednek ki dolgok, a sötét oldal is fejlődik, tudunk elfelejteni dolgokat. Volt egy olyan common internet fájlrendszer nevezetű kezdeményezés, ez különben nagyon bájos volt, mert erre a fejlesztői konferenciára meghívták egyébként a sambás társaságot is. A Samba fejlesztői levlistán még fenn voltam akkoriban, végre beszélhetünk a fejlesztőkkel, mert eddig úgy működött a Samba fejlesztés, hogy előkerült egy szoftver, tcpdump, megnézték, hogy mit beszél és akkor megírták az ellen oldalt, hogy erre válaszolunk. Ez nem a leghatékonyabb fejlesztési módszer, lássuk be, főleg úgy, hogy a másik oldal bármikor tud változtatni anélkül, hogy szólna. Különben ez megtörtént. Tehát amikor kijött az NT4 domain feature-ökkel a Samba kiderült, hogy viszont a konkurens cégnek a szervere csúszik és akkor a Samba így megverte volna, előnybe került volna, hogy domain kontrollerként tud működni, ezért kijött egy service pack, valami frissítés, ahol ez a DEC RPC, tehát nem a SUN-os DEC RPC hívások vannak ebben az SMB protokollban, hiszen átszámolták a függvényeket. Ennyi. Akkor a sambásoknak volt még egy hetük ezt kicsomagolták, megoldották és utána már megint működtek, de addigra kijött az a termék, ami különben csúszott és mégsem az volt a hír értéke. Mondjuk azt, hogy ez az SMB protokoll ez nem teljesen úgy működik, ahogy egy józan unixos hálózatos ember elképzei. Először is, első számú biztonsági funkció, hogy SMB-n csak akkor áll szóba velünk a gép, ha megfelelő néven szólítjuk meg. A gépnek van netbios neve, és ha én véletlenül más néven szólok hozzá, akkor nem válaszol. Az más kérdés, hogy megfelelő módon ezt a nevet pikk-pakk ki tudom, deríteni, de ez egy dolog. Ennek a nyelvnek rengeteg dialektusa, verziója és változata van, erre legutóbb ott futhattatok rá, ha a Sambat upgrade-eltétek 3.02x valahova, akkor elkezdtek nem működni a win kilencven akárhányas kliensek, mert Samba-ban már defaultban az egyik protokoll dialektust már nem használták és azt be kellett kapcsolni, sőt, ha annyira új volt a rendszered, akkor az SMB password-okat újból kellett generálnod, mert abból is több változat van, hogy hogyan kódolódik. Van a LAN menedzser, a régi DOS-os vagy OS/2-es LAN menedzser típusú meg az NT típusú kódolás, ami nem átjárható egymásba persze. Megegyeznek a dialektusban és akkor jön egy erőforrás, a tree connect ettől csak egy nüánsnyi a fájlmegosztás, ez lehet nyomtató, de lehet levelezés meg bármi. Ha újabb erőforrás kell, akkor megint egy tree conect, és itt van a gond, hogy mi van, ha ennek a usernek van joga a fájlmegosztáshoz, de nincs joga a nyomtatóhoz? A message echo meg server info autentikáció

nélkül elérhető. Ez is egy kicsit historikus jellegű, de tudni kell. Régen volt olyan még a win kilencven akárhányasban, hogy annak az operációs rendszer szerű valaminek nem volt olyan, hogy user. Ezért a megosztáshoz adtunk jelszót. Na, itt aztán macerás ez a megosztás szintű dolog, a közösnek a jelszava ez, és van a felhasználói szintű autentikáció, amikor már volt olyan fogalom náluk, hogy felhasználó, akkor lehetett ehhez jelszót és akkor van joga, nincs joga stb. Tehát itt már név és jelszó kell az autentikációhoz. A telepítésről, itt van egy Samba konfiguráció, mondhatnánk ez majd ha nézik a slide-okat, ez a minimum konfiguráció, amivel el lehet indulni. Javasolom az interfacest, főleg, ha ez egy több interface-szel rendelkező gép, hogy melyiken fülelek. Aztán arra is figyeljetelek, ha egy másik gépre átmásoljátok ezt a konfigurációs fájlt, és a hátsó részt gondosan beállítjátok, ezt ne felejtsetek el, lehet vele debugolni pár percet. Ennél trükkösebb a host allow, hogy kivel állok egyáltalán szóba. A többlábú gépeken azért hasznos lehet, hogy csak az egyik interface-n szolgáltatók Samba-t. Ma már a nyomtatásnál nincs nagy vita, ezt a Common Unix Printing System-et (CUPS) kell használni. Régebben volt ilyen lpr, meg lpr-ng, meg ilyenek.

Talán az F. Ható Katalin-féle Word 2.0, 1988-ban kiadott könyvben van egy nagyon aranyos nyomtatás című fejezet, hogy a nyomtatás speciális előkészületeket igényel. Például küldjük ki a szobából azokat a kollégákat, akik azt hiszik, hogy sosem káromkodunk, tegyünk el az asztalról minden tárgyat, amivel kárt tehetnénk a nyomtatóban vagy a számítógépben. Gondolom, mindenki ismeri a feelinget, mikor beesnek, hogy már rohannék csak ezt az egy lapot akarnám kinyomtatni, és melyik nyomtató hol és miért ferdén és miért nem jön ki stb.

Itt már van egy-két érdekes dolog, de ebbe most nincs időnk belemenni, de majd utalok a nyalánkságokra. Itt ésszel kell bánni. Az OS levelről annyit, hogy ez egy demokratikus rendszer, amit mi rendszergazdák nem szeretünk, mert mi vagyunk az urak. Az SMB protokollban a gépek választgatnak meg szavazgatnak, ez nem egy hatékony rendszer, amint azt tudjuk. Itt többféle súlyozó faktor van, az OS level az egyik ilyen, minél nagyobb, annál nagyobb eséllyel én leszek a főnök. Ezt már sokáig nem nagyon lehet növelni ezt a 254-et, 8 bit van rá. Az NT4 workstation az ilyen 31, a Windows 2000 szerver az 35, tehát ez a 254 az elég jó ráhagyás. Persze, ha a cégnek van egy gondosan üzemeltetett active directory és architektúrája, akkor óvatosan ezzel a számmal, mert bele lehet nyúlni.

A preserve key is érdekes dolog, rendes operációs rendszeren case sensitive a fájlrendszer, na most a másik oldalon, ez nem így van.

Azt mondják, hogy case retention van, de nem sensitive. Magyarul, ha kis-nagybetűvel lementem, akkor próbáljam megőrizni, de nem tesz különbséget a fájlnevekben. Arra felhívnom a figyelmet, hogy lényeges különbség van ennek a fájlrendszernek a működése (most nem a linuxos, hanem a másik oldaliban), ugyanis ez attól függ, hogy ez olyan fájl rendszer, amiben vannak 8.3-as régi nevek vagy nincsenek, hálózati-e vagy lokál, és ezeknek tetszőleges kombinációja befolyásolja, hogy hogyan kezeli a neveket. Ha mindenki Windowson ír le és azon olvas vissza, akkor nincs baj, de ha FTP-n, winscp-n valamin is csinálja és megcsinálja a nagyon fontos fájl meg az n betűs nagyon fontos fájlt, akkor melyiket látjuk és van olyan, hogy ha egyik megnyitja, egyiket látja, a másik a másikat látja, mind a kettő lementi és nem azt látják.

**Password backend-ről annyit, hogy többféle úton-módon tudja tárolni a jelszavát a Samba,** vagy ezt a helyi fájl rendszeren tárolt speciális DB formátumot vagy az LDAP-ot javasolom manapság használni. Aztán jönnek a megosztás definíciók. Tapasztalatom az, hogy egy dupla könyvtár szerkezet alakul ki minden ilyen fájl szerveren, mindenkinek van egy saját home-ja, az hogy most a create mask és az directory mask ilyen az inkább céges policy lehet. Lehet olyan is, hogy ez tényleg önálló és csak ő látja és senki más nem lát bele, ezt tartom különben a jobbnak. Gyakorlatilag kialakul munkacsoportonként egy közös könyvtár, ahol gyakorlatilag még egyszer létrejönnek ugyanazok a home könyvtárak egy idő után, vagy legalábbis, aki dolgozik, azoknak a usereknek a home könyvtára. Tehát érdemes egy közöset csinálni, ahol sokan írhatnak. Törölhetek is, de ez egy másik történet. Fontos fájl meg tegye olyan helyre, ahonnan nem tudják törölni. Itt a közös terület, felhívnom két-három trükkre a figyelmet.

A Samba alapvetően úgy dolgozik, hogy ha megtörtént a Samba autentikáció, akkor az SMB processz az forkol, és valamilyen userként el kezd ténykedni a linuxban. Normál esetben, ha én bemegyek lajbi userként, akkor egy idő után az SMB processzem lajbi userként ténykedik, az összes jogosultságával, amivel rendelkezik. Szoktak ilyet sikítani, hogy ha én a /home/lajbi helyett a /home/cstamast nézem, akkor látom. Igen, tehát ha én azt mondom, hogy nem a saját /home-omat, hanem beírom a share-ba és felveszem, akkor én látom readonly, ha amúgy a Unixban látom readonly, a tipikus unixnál így szokott lenni. Ezen ne lepődjön meg senki, majd megmutatom, ezt hogy lehet elkerülni, legegyszerűbben a fájljogokkal. Ez a force user és a force group arra való, hogy ha már autentikált a Samba, akkor ilyen userként fog tovább tevékenykedni a fájl systemen. Magyarul, ha a közösben létrejön egy fájl, akkor azt látom, hogy a közös user-demo csoporttal hozta létre, nem fogom tudni megmondani, hogy a Józsi vagy a Pista írta-e oda. Ez a mellékhatása, de ennek kikerülésére is van lehetőség. A másik az, hogy nem csinálom force user/force group-ot, hanem olyan create maskokat csinálom, hogy én láttam, hogy Józsi hozta létre, de ettől még Pista törölheti is akár. Bár itt megint van egy trükk, hogy a törlés az UNIX logika szerint működik, ugye a fájl törlése a UNIX logika szerint nem a fájlön végzett művelet, hanem a könyvtáron végzett művelet és a könyvtárírás jog kell ahhoz, hogy a fájl törljem. A másik oldalon a fájlra kell írásjog, ahhoz, hogy a fájl törlhessem. Ez olyan, ahova mindenki jöhet, egy userként létrejön és kész. Ez nem az éles webszerver, hanem minden normális esetben van a www.cégnév.hu és mellé van egy w3cégnév.hu, ahol a fejlesztők tornáznak, ez mondjuk arra tökéletesen jó és mondjuk innen rsync-kel. Nem javasolt automatán, mert mi van, ha nem fejezte be a fejlesztést és még nem tették ki a pontosvesszőt a sor végére? Akkor széthullik a weben, de az a lényeg, hogy megadom, hogy hova működik a force user, force group és itt van a valid users utasítás, ehhez a share-hez az csatlakozhat, aki a webmaster csoport tagja. A kukac, az általában csoportokra vonatkozik. Tehát a webmesterek itt ügyködhetnek ebben a fájlrendszerben. Amennyiben ma még arra van dolguk, hogy fájl system szinten piszkáljanak. Ugye, ha ez web, akkor nem árt a mask-okat és a mode-okat átállítani, hogy a webszerver is tudjon vele mit kezdeni.

**A printers-ről csak annyit, hogy a mai linuxokban tipikus, hogy a /tmp az a memóriában shmfs.** Nyomatásnál ebből nagyon izgalmas doksik jöhetnek ám, mert valaki fogja a 270 oldalas Word doksiját és berakja azt /tmp-be nyomatáskor a Samba, nem fér be és elfogy a memória. Ezért érdemes a path-t beállítani a printersnél is, hogy ezt hova teszi. Ilyen helyeken érdemes ezt átgondolni, hogy kényelmes és gyors, amíg kicsi a /tmp, de lehet olyan szerverszolgáltatás, ahol a /tmp nő. Akkor vegyem ki ne az shmfs-ben legyen, csináljak neki vagy külön fájl systemet vagy ugye a /var/tmp jó ilyenre. Már csak azért is jó és nem érdemes máshova nagyon elzavarni viszont, mert ezt bootoláskor törli és nem árt, ha már egyszer újraindult a nyomtatószervert, akkor inkább dobja ki a jobot, mint még egyszer nekiálljon küldeni. Bár tipikusan nem az egyoldalásra fognak sokat ülni.

**WINS és NetBT.** Csináljuk úgy, ha nekünk kis hálózatunk van, hogy a netbios nevek és a DNS nevek passzoljanak. Ugye például a DNS mask nevezetű program, ami teljesen jó DNS szerver és DHCP-nek is, az tud olyat, hogy bentre valami nevet szolgáltat, a domain név ilyen esetben mindegy, a host nevek, azok legyenek azonosak a netbios névvel és akkor a Samba tud olyat, hogy a WINS kéréseket átfordítja DNS kérdésre és vissza és akkor kész, megoldott egy csomó problémát. Ha ennél bonyolultabb a helyzet, akkor így jártunk. A sötét oldalnak a kedvenc szórakozása az, hogy a kliens gép bejegyezi magát a DNS-be. A DNS-be a DNS admin ír és nem a kliens gép, mert átnevezi esetleg a kliens gépét és úgy akar adminisztrálni, na mindegy, erre megvannak a technikák, a DNS mask nagyon jól tudja szűrni ezeket a kéréseket. DNS szerver logot szoktuk nézni, ugyanis tele van azzal, hogy xy desktop gépbe akart írni, de nem hagytam, a BIND tipikusan szereti ezt logolni. Nagyon jó, hogy ha a BIND-unk elé egy DNS maskot, azzal ezt ki lehet szűrni. Csak egy config opció és nem engedi tovább az ilyen jellegű kéréseket és kész. Nem beszélve akkor, ha az ilyen csacsogásokról, ha azok WiFi, mikro egyéb lassú linken mennek át, akkor még érdekesebb ezt csinálni. Na, most itt két dolgról van szó. Ez olyan szinten működik, hogy a

tcpdump egy ideig azt mondta, hogy malformed DNS packet, tehát hibás DNS csomagnak látta a NetBT kéréseket, tehát hasonlított az a DNS-hez, csak nem teljesen az volt.

A NetBT name service és a Windows name service variánsai ugyanannak a szolgáltatásnak, de nem ugyanazok. A netbios name service az broadcast alapú, magyarul, ha bekapcsolok egy gépet, az UDP broadcast-ek. Olyanokat tudott csinálni, az én gépemet úgy hívják, hogy „gép”. Feléled, megkérdezi, hogy kit hívnak úgy, hogy „gép”? Vár egy kicsit, senki nem válaszol. Engem úgy fognak hívni, hogy „gép”. Vétó? Ha nincs, na, akkor úgy hívnak, hogy „gép”. Ez így 3 broadcast, de ha ez elég régi hálózat és van benne Ethernet 2, 802.2, és 802.3 frame, meg IPX is, akkor az összes frame-mel az összes protokollra megcsinálja ezt. Ha ez még valahogyan kicsorog egy WiFi hálózatra, annak annyi, még nem is forgalmaztuk, de már ki van tömve.

A WINS az egy unicast, tehát ott van WINS szerver és megbeszéltem vele, hogy mi történt ott, azzal nincs semmi gond. Emiatt viszont a NetBT nem megy át a routeren, legalábbis kultúr-router ilyesmit nem továbbít, a WINS meg minden további nélkül elmegy másik subnetre is, merthogy unicast és lehet routolni és kész. És, hogy igazán könnyű legyen a dolgunk idézőjelesen a két szolgáltatás teljesen független és soha nem cserélnek egymással adatot, bár egy gépen futnak. Ha WINS szerver nincs beállítva, akkor a WINS kliensek nem szerepeltetik magukat a listába, ettől függetlenül netbiosban lehet, hogy látjuk vagy fordítva. Ez ilyen. Hogy jó legyen a netbios, az igazából csak humán célokat szolgál, tehát az, hogy én a browsing listán látom-e a gépet vagy nem, az a humán megnyugtatása, hogy tényleg elérem a szolgáltatást az meg a WINS, egyszerűsítve a helyzetet. Tehát ha a browsing nem megy, az egy dolog, sőt jobb is, ha nem megy, mert kevesebbet csacsog a hálózaton, csak attól még fixen be kell tudnom írni mondjuk a gépnév megosztás nevet és akkor kész.

A kliensek többféle üzemmódban tudnak dolgozni. Nem tudom ebben a körben ez a HKLM (HKEY\_LOCAL\_MACHINE; Windows registry) ismert-e? Ott van az az érték, hogy broadcast-ol, nem broadcast-ol, kevert vagy hybrid. Samba-n lehet állítani, hogy mi legyen a resolve order, milyen protokollokat használjon, ha megpróbál megkeresni egy netbios nevet. Az lmhosts az a LAN menedzser host fájl, a host az a host fájl, a WINS az a WINS, a broadcast az meg a broadcast. Ez a computer browser, ami a humán interface-es.

**Master browser:** ez amit mondtam, hogy szavazással döntenek el, hogy ki a master browser. UDP tartományban is van, tehát helyi, tehát minden UDP broadcast tartományban van egy master browser. Azonkívül meg egy tartománynak lehet, az Active Directory-ját kezdi jól kezelni. Miket beszélgetnek? A computer browser frissítések UDP 138-on, bootoláskor 3 broadcast, de ha van három frame-ünk, akkor ez háromszor három, ha van két protokollunk, akkor ez 3x3x2. A local master bejelenti magát 12 percenként, az workgroup bejelenti magát 15 percenként, a hostok bejelentik magukat 12 percenként és a master browser 15 percenként. Ez egy ilyen bérelt vonalon jól néz ki. A browser listákat is cserélgetik, a tartomány master browser a helyi browsereket cserélgeti, ez már unicast és egész jó. Sokat fejlődött mostanában a Samba. A lényeg az, hogy nagyon sok cég marad a jó öreg NT4 jellegű domaineknél és nem biztos, hogy rohanni kell. Láttam nagy cégnél olyan tanulmányt miután a régi NT4-ről upgradáltak Active Directory-ra, hogy ennyi pénzből és energiából tetszőleges címtárt, autentikációt bevezethettek volna. NT4-es domaint teljesen jól tudja csinálni, akkor is, ha windows a domain controller stb. Az biztos, hogy jól működik. Az Active Directory-ban részt tud már venni a Samba, de ő nem tud Active Directory domaint létrehozni. De ha van egy domain szerverünk, akkor abba a domainbe be lehet rakni például egy nyomtató szerveret tökéletesen ki lehet váltani belőle. Olyan apró cég, aki nyomtatásban nem is olyan jelentős, mint a Xerox, például így használja a rendszerét.

**Domain controller:** itt be kell állítani néhány dolgot. Vannak profile-ok, login könyvtárak. Egy apróság: szervertes profil editorral lehet editálni, és ha a „pol”-t tehát policy-t átnevezem man-ra, mint mandatory, akkor az kötelező policy és lenyomja a user torkán. Tehát utána a user belép, bármit állítgat a policy-jén, vagy amikor újból belép, akkor az van, amit én hagytam régen, ez ilyen labor PC-ken nagyon hasznos tud lenni. Meg aztán ez a share, direkt van egy olyan Samba

opció, ha hisztis a Windows, nem írható jog a profil könyvtárra, viszont van egy olyan, hogy fake write, azt hazudom, hogy írhat rá, de közben nem írhat rá, és akkor szépen továbbmegy a Windows és azt hiszi, hogy ír rá, de nem ír rá.

**Netlogon share:** itt egy dologra hívom fel a figyelmet, ha ide valaki batch fájlt ír, akkor konvertálja át a sorvégi jeleket, mert a másik rendszer mást használ. Ékezetek: ez folyamatosan előjön, nem triviális. Elvileg az új protokollokban az a jó, mármint a kliensek részéről, hogy negotiation van és megbeszéljük a kódolást, csak néha hazudnak. Samba-ban elég sok opció van arra, hogy a fájl system milyen kódolású és a kliensnek milyen kódolást mondok. Nagyon könnyen elképzelhető olyan scenario, hogy nem találok olyan közös nevezőt, hogy mindenkivel működjön. Ilyenkor azt lehet csinálni, hogy van egy olyan trükk, hogy vannak ilyen makrók, hogy %n, az a gép netbios neve meg stb. dolgok, hogy a config fájlban úgy include-olok, hogy win98-as gépek másik néven szólítják meg a szervert, és akkor nekik másik config fájlt használok, nekik 852-öt mondok, az NT-knek 1250-est mondok, különben meg az újabbaknak, az XP-nek azt mondom, hogy UTF8. Amúgy meg a fájlrendszer linux alatt UTF8. De azért itt lehet kalandozni. Aztán jön a japán vendéghallgató, aki csinál egy fájlt, na az izgalmas. Volt rá példa, én lementettem, de ő nem tudja betölteni. Milyen codepage van? Japán. És neki? Koreai. Na, most akkor mi a közös nevező?

Jelszó tárolási módokról: LDAP célszerű.

**Access controllról, nevekről:** az elvileg 254-es fájlnevek a Windowsban 224 hosszúak. Ugye a szabvány open-save dialog boxban van egy kis bug és nem enged hosszabbat. Elvileg a doksi szerint 254 hosszú lehet, de nem. Ha dialog boxszal hozod létre, akkor tényleg csak 224. Ha 254-esen hozom létre azt dialog boxból nem lehet megnyitni, de ha paraméternek van elindítva akkor meg lehet nyitni. Az xcopy kezeli, de például a Word nem nyitja meg. Szép, mikor az ember ilyenre fut rá és fél napot olyan szépen debugol, hogy csak na.

Itt ugyanaz a force group, force user, erről már volt szó, readonly, yes-no. Nagyon hasznos a readonly, ha tényleg olyan share, akkor kapcsoljuk be, mert lockingba, op-lockingba meg ilyenekbe sebességnövekedést nagyon masszívat, ilyenekbe elérhetünk. Van egy megosztásunk, ami effektíve readonly, akkor ezen nagyot dobhatunk. Ugye valid userrel meg tudjuk mondani, hogy ki az, aki csatlakozhat, az invalid userrel, hogy ki, az, aki nem csatlakozhat stb. Van most már POSIX acl kezelés ugye a fájl systemekben, ez egész jól működik, és tovább lehet adni. Control panel, adminisztrative tools, computer management elindítva az action, connect to another, gép kiválasztása vagy system tools, shared folder, na itt lehet állítani.

**Locking:** ez egész jól működik a Samba-ban, bár cluster fájl systemekkel elég érdekes dolgokat lehet csinálni. Op-lock: gyakorlatilag az op-lock azt nem az alkalmazás, mert ugye lockingot alkalmazás kéri, hogy lockolok egy fájlt vagy nem lockolom stb. Az op-lockot az op. rendszer maga, a kernel intézi fájl system dolgaiban. Registry-n keresztül lehet szabályozni a működését és azért csinálták, hogy gyorsabb legyen a történet. Gyakorlatilag amiatt van, hogy masszívan cache-elhessen a kernel, tehát, ha olvas egy hálózati megosztásról és van neki egy rendes op-locja, akkor cache-elhet oda-vissza írásra, olvasásra. Sőt még van olyan helyzet is, hogy a lockingot is cache-elheti, tehát amikor az alkalmazás lockot kér, akkor helyben szól neki az op. rendszer, a szervertel meg sem kell, hogy beszélje. Emiatt látjuk, hogy ha ez ilyen jó és ilyen gyors, biztos van pár buktatója. A level1 op-loc az, amikor mindent szabad. A level2 az hasonló, de már csak olvasásügyileg, írásra nem lockolhat annyira és lehet olyan, hogy filter op-lock, hogy nem is engedem meg.

Ha a kliens oldali cache-elés jó, akkor használjuk. Ha egy user használ egy share-t, például a home-on, ott nagyon jó szintén, mert más nem piszkál bele, had dolgozzon az op. rendszer, had cache-eljen, mert tényleg hasznos. Amikor közösbe többen is írhatnak, ott már megosztlanak a tapasztalatok, hogy mennyire jó vagy nem. Ha lassú, messze van a szerver, másik telephely stb., akkor nagyon nagyot gyorsíthatunk ezen, de bukhatunk is vele, majd mindjárt meglátjuk,

hogy miért. Mert a megbízhatatlan és túlterhelt hálózaton elveszhet. Ha force userünk van, akkor megy egy op-lock break eddig, mert akkor vált egyet a tulajdonos, de ez nem mindig jut át.

Op-lock példa: gép1 megnyitja a fájlt, kér op-lock-ot, mivel semmi más nem használja, odaadja neki. Gép2 is megnyitja a fájlt, kér op-lock-ot. Erre mivel a gép1 még nem írt vissza a fájlba, ezért a szerver kéri, hogy az op-lock1-ről menjünk az op-lock2-re, tehát szól a gép1-nek, hogy lépjen kicsit vissza. A gép1 ekkor leírja a write-cache-ét és visszalép (ezt op-lock braking-nek hívják) op-lock2 szintre és ekkor megnyithatja a másik gép a fájlt, mert a write-cache leíródott. Ha bármelyik gép ezek után ír még a fájlba, akkor szól a szerver, az összes, két gépnek, hogy felejtsetek el a read-cachet is, hagyjuk ezt az op-lock-ot és rendesen olvasson mindenki. Főleg az ilyen bérelt vonalas, meg microval átlótt vonalakon, az a jó, hogy ez nem TCP alapú meg nem olyan, tehát nincs ellenőrzés, simán elveszhet akár az op-lock kérés, akár a válasz.

**Nyomatás:** egy dologra hívnám fel a figyelmet, hogy a Microsoft SDK-ban van egy Postscript driver, ami, valahogyan elrontották és open source lett és a CUPS-osok gyakorlatilag megcsinálták ebből a Postscript driverüket. Ismerős a probléma, aki Windowst üzemeltet, hogy terminál szerveren mekkora gáz a nyomtató, milyen driver, hogy van, ki hova nyomtat. Mindenféle driverek kellene és egészen el lehet rontani egy terminál szerver stabilitását ezzel a sokféle nyomtató driverrel, mindjárt mutatom, hogy miért. Ilyenkor a legjobb megoldás az, hogy CUPS nyomtatószervert csinálunk valahol, egy Samba-s linuxos nyomtatószervert, amin CUPS fut, és minden windowsos kliensen lenyomjuk, hogy a CUPS-os Postscript drivert használja. Innentől kezdve a terminál szerveremen egy nyomtató driver van, ami különben stabil, nem bántja az op. rendszert, és majd a CUPS konvertál és nyomtat, ahogyan kell. Ráadásul mivel Postscript az oldalszámolás is egész jól működik.

A nyomtató driverek különböző könyvtárakban tanyáznak. A w40/0 könyvtár alatt a win kilencven akárhányas driverek tanyáznak, a w32x86/2 alatt az NT kernel módban futó driverei és a /3 alatt a userspace-ben futó driverek. Mi van, ha egy /2-ben lévő driverben memory leak van. Az szokta hanyatt lökni a terminál szervert is. Amúgy meg kezelésére vannak parancsok stb. Még egy apróság a Samba-val kapcsolatban: jó, de ki törölte azt a fájlt, ki töltötte fel stb. Van a Samba-ban egy virtual fájl system egy plusz réteg és ez ilyen stack-elhető objektum modulok és ráadásul egymásra építhetők. Önmagában a stack felépítése nem okoz jelentős teljesítménynövekedést, nyilván, ha ez egy víruskereső, akkor azért el kell olvasni a fájlt. A sorrend az számít, hogy milyen sorrendben építem egymásra, mondjuk a víruskeresőt nem érdemes alulra építeni, mert a vírusos fájlt dobjuk már el a lelegején, hacsak nem akarjuk naplózni, hogy ez ki volt, de mindegy.

**Van olyan, hogy audit modul,** ez például a kapcsolódásokat, a dir open/create/remove és fájl open/close/rename/unlink/chmod nyomja syslogba. Magyarul szépen naplózódik, hogy ki melyik fájlommal mit csinált. Van egy extended audit, ami még többet tud, ebbe most nem mennék bele.

Van egy fake permission (fake\_perms), bármit hazudhatunk a klienseknek, úgymint elhiszik alapon. Van egy recycle, ez is kellemes, ez nem a windowsos recycle, hanem a Samba unixos fájl szinten, egy elrejtett helyre, amit különben nem mutat, csinál egy recycle-t, ahonnan vissza lehet bányászni fájlokat. Ez a régi novelles NCP-ből (NetWare Core Protocol), ha valaki ismeri, a novelles fájlrendszerben van hasonló funkció. Be lehet állítani a limitet stb., ezt azt. Nyilván a user akkor fogja törölni a DVD gyűjteményét, mikor előtte törölt egy nagyon fontos dokot, aztán azt meg nem lehet visszaállítani. A netatalk az az Apple felé. A shadow\_copy lehet még egy érdekes, gyakorlatilag snapshotokat lehet csinálni egy share-ről.

**Van egy database fájl system, ami tulajdonképpen SQL query-k eredményét mutatja,** mint könyvtárakat a fájl systemre. Ez ideális az mp3 gyűjteményre. Akármilyen szerint nézegethetem és olyan playlisteket rakok össze, amelyet akarok. A vscan az open antivírusosok csinálták víruskergetőre, valószínű be lehet ide rakni más víruskergetőt is. Az más kérdés, hogy ez mekkora kézféket jelent az I/O teljesítményben, hogy én most azonnal akarom nézni, mert kontrollálatlan gépeim vannak vagy elég, ha időnként cron-ból időnként átfutom és megnézem víruskergetővel a

fájlokat. Szerintem ez alapvetően attól függ, hogy mennyire kontrolláltak a kliensek. Ha nekem van egy olyan megosztásom, ahová két ember ír, mármint a boldog meg a boldogtalan, akkor azt biztos, hogy ellenőriztethetem ilyennel. Mert legfeljebb lassan másol fel. Ez nem a nap mint nap használt munkakönyvtárakat, hanem az ilyen gyűjtőhelyeket, azokat valószínű érdemes ilyennel piszkálni.

Van kifordított is, az a winbind. Mint mondtam a Samba azt szereti, ha van UNIX user vagy például UNIX-ról esetleg be akarok lépni windowsos accounttal, akkor ez fordítva lehetséges. Gyakorlatilag usernév-passwordöt kell csinálni. Felhasználó látja más home-ját is. Mondtam tipikus problémát cd.., akkor látja. De ezt ki lehet úgy védeni, hogy a home-nál felsoroljuk a valid userbe a %s, az a user, aki autentikált, annak a neve. Magyarul a home az egy speciális, mert, aki autentikált, azt mindig úgy hívják, tehát ha én megyek lajbiként, akkor lajbi a share neve és ezt csak a lajbi nevű user láthatja. Így kivédtük azt, hogy megnézi a másik userét is, de inkább a creat-askot csináljuk. Ennyi lenne tömören a Samba.

A Samba minden további nélkül tud több interface-t és alhálózatot kezelni, sőt, mivel a Samba-ban elég jó WINS szerver van és belóg minden hálózatba, helyre rakja a browsingot meg az ilyeneket is. Ha egy gépre össze tudod teregni, akkor ez jó. De ha virtualizációról van szó, akkor nem egyszerű. Nem egyszerű nagy rendelkezésre állású Samba-t csinálni. Alapvetően a probléma, hogy egyrészt a user adatbázist nem egyszerű szinkronizálni, ezt az LDAP-pal ki lehet kerülni. Egy tdb adatbázist nem lehet, hiába sync-elgetem, cache-el nem ír le stb. De mondjuk LDAP-ból egy user adatbázis oké.

A probléma még a fájl locking, hiába van nekem cluster fájl systemem, mondjuk az OCFS (Oracle Cluster File System) nem tud lockingot, mint olyant, nem használható semmire szerintem, arra jó például, hogy az /etc/xen-ben a configokat lássam egy helyről. A GFS2-vel (Global File System) és a legutóbbi körözésemnél, ami azért nem up-to-date, mert volt egy éve, kis instabilitást tapasztaltam.

A GFS1-gyel meg patch-elni kell a kernelt. Na most a XEN-nel is kell patch-elni és nem olyan egyszerű olyan keresztmetszetet találni, hogy olyan kernelt találjak, hogy a XEN patch után belemegy a GFS, vagy a GFS után a XEN.

A másik jellemző, nem szoktak ezzel dicsekedni a rendszergazdák, de nekem a termelő (production) legnagyobb XEN rendszerem az 2.6.16.33-as kernellel fut és 3.04-es XEN-nel. Nem kell rohanni a verziószámokkal, működni kell a rendszernek, és kell egy tesztrendszer, ahol kipróbáljuk az újakat. Hogy most miért 16.33, lehetne 17.04 is, de ettől nagyobb tartományt nem tudok elképzelni a vasaimmal, ami nekem jó lenne. Van elég sok ok rá.

A Samba-ban össze lehetne rakni egy olyan rendszert, ha már virtuális gépezünk, két fizikai vas, mindegyikre rakunk fájlserver gépeket, amik hozzáférnek egy storage-hoz, és azt osszák ki. Kell egy jó cluster fájl system alájuk, itt van az egyik nagy kutya elásva. De nem ezek lennének a domain controllereink vagy az OWA szervereink, ezek csak úgymond fájlserverek. Amelyik csoportnak kell egy Samba, azoknak csinállok egy külön domain controllert, csinállok egy virtuális gépet, amin elindítok egy Samba-t, és odaadom a tanszéki adminisztrátornak, hogy konfigurálja oda, ahová akarja. Akár LDAP szinten, akár gép szinten csináljon rá megosztásokat. De a megosztások nem igazán azon a gépen vannak, hanem van egy olyan, hogy Microsoft Distributed File System, MSDFS. Ez durván úgy néz ki ezen a kis gépen, ami szervernek látszik, mondjuk a home megosztás az symlink a két másik szerver home-jaira. Tehát az MSDFS-t durván így csinálja meg a Samba, hogy ott van egy fájl, amiben benne van a szervermegosztás, szervermegosztás a fájlnevre. Bármelyik a kettőből működik, akkor elérhető. Kezd már egész HA szintű lenni. Főleg úgy, hogy van két fizikai vasam, van két úgymond fájlserverem, ami kiszórja a fájlokat, van két LDAP szerverem, amik nyilván egymás replikái. Csinállok ilyen kis gépeket, akár egy példányban, mert ha gáz van, akkor vagy live migrationban még mielőtt beüt a krach, átküldöm a másik fizikai gépre, vagy ha beütött a krach, akkor úgyis ott van a share media a fájl systemen kívül és onnan

bebootolom. Rossz esetben egy reboot idejére esett ki a dolog. Ami amúgy sem zavarja a usereket, mert az MSDFS miatt a futó fájlserverek közül valamelyiken eléri.

Elvileg ez járható, hogy ilyen kis virtuális Samba-kat csináljak, szerintem a kulcs ebben az, hogy a kis virtuális Samba-kat adminisztrálja az aki akarja, és mögötte van két nagy fájlserver. Na most, ha egy van, az működik, de akkor nem redundáns, tehát megint van a Single Point of Failure esete. Mindezeknek messze előfeltétele az LDAP, nyilván abból is van replikám, és abból is kettő van. Végül is az LDAP az lehet a Samba serveren, főleg a replika. Az is jónak tűnik, hogy van egy csak LDAP serverem, amit konfigurálok, és a két egyben fájlserver az replika, ez így már jónak tűnik.

**Hallgató:** Egyébként sikerült olyan cluster fájlrendszert találni, ahol normálisan működik a bootolás?

**Előadó:** A GFS-en működik, ATA over Ethernettel összerakott egy egész komoly és működő rendszert, van benne egy-két tanulság, például hálózatra az ATA over Ethernet nem tolerálja a frame vesztést, eléggé be tud lassulni, de amúgy működik.

Ami miatt a kernel verzióhoz ragaszkodom, hogy van egy Fibre Channel kártya a gépben, van egy Fibre Channel switch és van egy MD storage, abban két storage processzor van, tehát két útvonalon látszik ugyanaz a LUN, ez a multipath dolog. Tök jól működik, átáll, visszaáll stb., de amikor elkezdtem ezt csinálni, akkor ez nem is a device mapperben volt, hanem a szoftver RAID-be, abba volt egy multipath target. Majdnem működött, ehát felállt, átállt csak nem akart visszaállni. Ilyenkor mit csinál az ember fia? Írtam Molnár Ingónak: tudom, hogy már nem ezt csinálod, de valamikor te írtad ezt az md-t? Visszakérdezett egyet, ez a klasszikus Mingo-féle levelezési thread, írok egyet, akkor ő küldi, hogy ezt a kprint-et rakd be ide meg ide a forrásba, és küldd el a kimenetét. Elküldtem, majd visszaküldte, ha ezzel patch-eled meg a kernelt, akkor jó lesz. A harmadik levélben ott van a patch, ami megoldja a problémát.

**Hallgató:** windowsos nyomtató server esetén hogyan lehet megegetni az olyan nyomtatókat, amihez csak windowsos driver van?

**Előadó:** biztos meg lehet etetni valami úton-módon ez a GD nyomtatókra, tehát a windowsos Graphic device izékre gondolsz? Láttam már ilyen megoldást. Nálam a nagyobbik cégnél sikerült, ilyen Canon 2020 alatt nem állunk szóba, tehát az ilyen, ekkora és hálózatos és van hozzá driver és kész. Azért ezt el lehet érni, nem könnyű harc, néhány év. Kell nyomtató? Akkor van.

**Hallgató:** meg hogy megkérdezzék, hogy mit érdemes megvenni!

**Előadó:** azt nem értik, hogy nyomtató vásárlásnál mit szokott mondani a user? A3-as, színes, szkennelni, faxolni meg mit tudom én még mit ne tudjon. E helyett például fő paraméter nyomtatóra a lapterhelés. Rendes nyomtatóhoz odaírják, hogy havi vagy napi ennyi oldal nyomtatásához tartják optimálisnak. Egy klasszikus Deskjet 500-as tintasugaras esetében küldtek rá napi 400 oldalt, és csodálkoztak, hogy mennyibe kerül, Meg hogy hónapok alatt meghal, szegény jól bírta, tovább, mint kellett volna, de nem erre készült.

A Xerox az elég nagy szegény volt. Ugye tudjuk, hogy az Ethernethez elég sok köze van a xerox cégnek, és nyomtatókat is gyárt, csak kicsit kisebbeket, asztalra is fel lehet tenni, szétrázta az asztalt egy idő után, tehát külön kellett tenni, de mindegy. Viszont olyan botrányos volt a szoftvere, hogy ha azt mondtam neki, hogy Windows nyomtatás, akkor csak SMB netbeui, ha azt mondtam, hogy Novell, akkor csak IPX és ha azt akartam, hogy IP felett nyomtasson, akkor csak unixos lp. Semmi logika nincs benne, mert egyrészt a windowsos nyomtatás lehet IP felett is, mint az már többünknek sikerült már, ezenkívül viszont a Cisco switchek állandóan levágtak, ugyanis malformed Eternet frame, mintha nem tudná a szabványt a Xerox. Azok régi nyomtatók voltak hál Istennek, ellenben olcsók voltak, mondjuk 120 ezer forintért vették, majd ugye sikított, sárgán világított és gombnyomásra nyomtatott és hívták a szakembert, hogy miért? Mondták, hogy szerviz csomagot kell telepíteni, most már nem csak toner kell bele. Kijött a szervizes és mondta, hogy az első szerviz csomag 230 ezer, akkor rájöttek, hogy a Canon az nem biztos, hogy olyan drága.

## Csillag Tamás és Lajber Zoltán: Rakjuk össze az egészet (hardver kiválasztás, tippek, trükkök szerver üzemeltetéshez)

Nagyjából összeszedtük a szolgáltatáslistánkat, mit is akarunk mi itt szolgáltatni, hogy melyik szolgáltatás hol fusson, nyilván ehhez kell hardverválasztás. A szerver, hardver méretezésbe és választásába nem akarok nagyon belemenni, egy-két tapasztalatot azért szeretnék megosztani veletek. Egy slide erejéig felelgetem, hogy egy ilyen virtualizált több felől hozzáférhető környezetben milyen nehézségekbe jutunk a mentéssel. Ha már hálózatról beszélünk, akkor én beszélnék a layer0-ról. Ez a hova teszem a szerveremet és honnan veszi a tápot például.

**Melyik szolgáltatás hol fusson?** Nem spiliázom nagyon túl a dolgot, mert szét kell osztani. Erről már a legelején, a virtualizációnál szövegezt Tamás, itt összeütöttem egy kisebb táblázatot. Lapozok egy régebbi ábrára, amit én lényegesnek tartok: ez néhány százfős méretig betakarja a követelményeinket egy ilyesmi hálózat. Azt tudom mondani, hogy egy tűzfalgép, mellé úgymond DMZ-be, alapesetben mondjuk egy gép, nyilván ez ragozható igény és teljesítmény szerint, hogy mennyire szedem széjjel, de nyilván virtuális gépekkel megint az a jó, hogy megbízható gépek általában elég nagy teljesítményűek és így a virtualizációval meg tudom azt oldani, hogy megbízhatóan, de több és szeparált rendszerem van. Középen van egy VLAN képes és legalább egy belső szerver, nyilván ez is osztható. Ebben a felállásban gondolkodunk, ha mást nem mondok. Innentől kezdve, hogy egy fizikai vas vagy nem, az már részletkérdés. Az autoratív DNS, a webszerverünk és az SMTP gateway az a DMZ-be való, erről nem nagyon tudunk vitát nyitni.

Az előbb említett slide-on van egy olyan, hogy mi mivel vonható össze és például a külső router és külső szerver összevonható, ez ugye arra, amit a kolléga is mondott, hogy néha a tűzfalon is kénytelenek vagyunk szolgáltatást futtatni, tehát ha nekem ott van egy olyan tűzfal gépem, akkor azon futtattam szolgáltatást, virtualizációval ez javítható, mert amelyik NAT-ol, azon nem kell webszervert futtatnom. Mi van, hogy ha mail szemét dől és bele akar döglenni a szerverem, ne haljon le az SMTP-m. Ilyen megfontolásokkal valahogyan eloszszuk.

A következő dolog az ilyen alapinfrastruktúrák, az LDAP, DNS, NTP, DHCP és például az SQL szerver, amit közvetlen nem látni. A legtöbb SQL szerver a webszerveren is használja, meg bentől is akárki használhatja. Nyilván lehet odáig fokozni, hogy a külső az ne az legyen, annak legyen egy külső adatbázis szervere és ez meg egy belső, egy másik. Tehát mondom, hogy ennek nincs vége, hogy hol szedem széjjel. Amekkora a probléma az, hogy mindent rakjunk egy gépre majdnem ugyanakkora probléma, mintha túlzottan szét daraboljuk. Ez függ a virtualizáció technikájától is, mert még az ilyen OpenVZ szerű virtualizációknál viszonylag egyszerűen sok gépet létre lehet hozni, addig XEN esetén az összes fizikai memóriát oszthatom széjjel, nincs overbooking. Tehát nem tudok akárhány virtuális gépet csinálni, mert a fizikai memória az korlát. Mondjuk ma, amikor 32, 64 giga RAM-ok férnek az Intel architektúrára ez nem akkora probléma, de azért ennek ára van. Az ára az nem a beszerzési ára, hanem majd utalunk, hogy mi.

A másik nagy feladatcsoport az a Notes. Ezt azért célszerű fizikailag külön gépre rakni. Aztán jönnek még olyanok például, hogy DNS és DHCP, a sok VLAN-om van, akkor sok interface-e lenne. Ha most XEN-ben gondolkozunk és elkezdünk felhúzni virtuális interface-eket a dom0-m config-jából, fálnak megyek mire összekonfigurálgatom. Képzeljük el, hogy a dom0 eth0.20-at össze kell bridzselnem a virtuális géppel. Én ezt úgy oldottam meg, hogy 4 PCI ethernet kártya van a szerverben, az egyik kártyát odaadtam annak a dom0-nak, ami a DHCP-t csinálja, meg akkor ez nyilván tud DNS-t is csinálni stb. Tehát akkor egész PCI device-t delegálok a virtuális gépnek. Az egy trunk, VLAN és mehet. A Notes-ot én külön vason tartom, nyilván nem kicsi a hálózat, de viszonylag régi, de megbízható és bevált vas. Az is egy kérdés, hogy az SMTP szerveren és a webmail az egy gépen van vagy külön, elég jól elfér azért egybe. Hát akkor a fájlnyomtató és lehet ez az egész egy belső szerver egy kisebb esetben. Nyilván itt skálázhatósági problémák vannak.

A másik pedig az, hogy hiába van sok virtuális gépem, ha a vas elromlik alatta, akkor behalt mindenem. Tehát lehet, hogy emiatt érdemes, hogy ezt két gépre tegyük. Ettől függetlenül

virtualizálni és ha kell akkor át tudom egyszerűen lapátolni, legfeljebb a diszket áttologatom, vagy ha nincs is mögötte olyan storage-om, valahogy át tudom játszani az image-eket, más nem a mentésből az előző napi állapotot, de valami van. A virtualizáció még egyre jó, tesztelésre. Elég elterjedt az, hogy a www.akármi mellé van egy w3 akármi, amelyiken az új verzió, az új design, az új CSS akármi kipróbálható. Erre is nagyon jó a virtualizáció. Kap 2 giga RAM-ot az éles szerver és kap 256 megát a tesztszerver és lehet rajta két embernek megnézni és, ha jó, akkor rsync-en átmásoljuk vagy akármi. Ez nem recept, hogy így szedjük széjjel, de csináljunk egy hasonló táblázatot és gondolkodjunk. Aztán visszatérünk rá, hogy mi számít. Egyrészt a szolgáltatás és biztonság miatt osszuk meg, de ne essünk át a ló túl oldalára. Ugyanaz, mint a particionálásnál, szedjük szét külön fájl systemekre a szerverünket, de ne daraboljuk szét a diszk területeinket, mert lehet, hogy mindenhol van 500 mega szabad helyünk, de ha nekünk 1 giga kellene és nem egybefüggő, akkor baj van. Ezért is mondtam, hogy az LVM-nél az elején kisebbet osszunk ki a volume group-ból és növeljük, mert például az XFS-t nagyon jól lehet menet közben növelni, de nem lehet csökkenteni. Az ext3-at lehet oda-vissza növelni, de alapvetően offline.

Aki járt már hosting szolgáltatónál, az látta a polcon a Codegen kék LED-del világító kiállítást, na azok nem szerverek. Sávszélesség értékekről így hirtelen, kicsit szokatlan, az első itt a nagy MB/sec meg a byte meg a bit dolgok. Ezekre mindig a helyes válasz, hogy attól függ. Gondoljuk végig kicsit a problémát. Azért a régebbi processzoroktól csodát ne várjunk. Most például az, hogy az Opteronnál miért írtam ilyen nagy számokat és a Xeonnál miért csak 1x10 gigabitet? Ha fog valaki egy alaplap rajzot, ugye korrekt alaplapokhoz adnak olyan rajzot, hogy a chipset hogyan van bekötve. Ez már nagyon sok mindent elárul, mert kiszórja a piacon kapható alaplapok 80%-át. Ott azt látjuk, hogy a Xeon processzor az be van kötve egy úgymond északi híd és arra megy minden és ott ez a sávszélesség. Az AMD-nél pedig, ha 10 gigabit az van a rendszer felett, 2x8 gigabit a két processzor között, ha kétprocesszoros rendszer és még van a memória sávszélesség. Tehát egy kicsit más jellegű az architektúra, és itt bizony van egy kis előnye az AMD-nek. Két évvel ezelőtt azt mondtam volna, hogy messze nyerő, hál Istennek, ez a két cég harcol egymással, és volt olyan időszak, hogy jóval nagyobb teljesítmény kevesebb áramfogyasztással egyértelműen az Opteron tudta megoldani, ma már ez nem igaz. Most az, hogy melyik a nyerő, arra a helyes válasz, hogy attól függ, az alkalmazás jellegétől, a felhasználás módjától függ.

A másik, amit el szoktak felejtani és ez főleg a kisebb kategóriájú gépeknél, hogy mit tudnak a slot-ok, a bővítőkártyák? Ha nekem egy PCI-os alaplapra épített SATA vezérlőm van, akkor ennyi, amit tudok és ez nem túl sok. Magyarul jó esetben ki tudok lapátolni 1 Gbitet, de az már ideális eset. Nyilván a mai ez meg PCI kérdések felírtam az egyszerűest, a mihez tartás végett, a 16-szoros meg elég ritka, tipikusan ma 4 és 8-szoros dolgok vannak, de egyszerű szorzással ezt ki lehet hozni, lineáris, bár elméleti. Azért van olyan, hogy több buszrendszer ezen átmegy, arra megy. Azt tudom mondani, hogy a 10 Gbit-es kártyán 70-75% fölé menni macerás, sőt eddig nem sikerült. Ha valaki összerak egy rendszert, 10 giga Ethernetet belerak, az egy gigányit kilapátol egy mai rendes szerver minden további nélkül. Kicsit körülnéz, fél nap állítgatás után 4 giga, és a következő egy hónap, fel lehet menni 4-ről 7-re. De ez már alkalmazás és ilyen függő. Túl vagyunk már egy eljárásán, meg lehet oldani, de nem egyszerű.

**A winchesterekről, a hard diszkek különböző sebességéről.** Maga, egy diszk mit tud? Erről is lehet vitát nyitni. SATA, SATA2, itt csak megemlítettem a két dolgot, ez a drága, ez meg a nem annyira elterjedt dolog a Fibre Channel és az InfiniBand. Az InfiniBand az egy brutális dolog, ennek van még egy négyszeresen összefogott sebessége is, tehát 10 és 40 Gbit-es a tipikus InfiniBand manapság. És van benne, mondjuk azt, hogy DMA. Az InfiniBand-dal összekötött egyik gépnek a diszkvezérlője a másik gép memóriájába tudja rakni az adatokat. Itt aztán gyönyörűen lehet virtualizálni, mert tudok olyan virtuális gépet csinálni, hogy ennek a vasnak a diszkvezérlője, annak a vasnak a hálózati kártyája és ennek a vasnak a processzora, meg annak a négynek a memóriája. Ha itt megdől valami az megint egy izgalmas dolog. Ezt lefordítva, ha egy PCIx-es, ez egy tipikus, szervernek soha nem rakjuk be a legújabb technikát, ez gondolom evidens, hogy nem a most megjelent legfrissebb dolgot kell berakni szervernek, egy kicsit várjunk még,

amíg leülepedik és az összes cég megtanulja elolvasni egyformán azt a szabványt, ami most jelent meg, mert ezt nem szokták. Tehát ha van egy jó diszkvezérlő, ez a hagyományos SCSI diszkvezérlő, azt 3-4 darab 10 000-es fordulatszámú diszkkal tömöm ki, tehát addig nő a sebessége, utána persze növelhetem még, de már nem fog nőni a sebesség. Itt van még néhány feltétel, hogy csinálom a RAID-et stb. SATA-ból kicsit jobb a helyzet. A PCIe 8-szoros vezérlővel akár 13-14 diszkiig el tudok menni, nem nagyon van 16 diszknél nagyobb doboz. Nem teljesen véletlenül.

**A hardver minősége:** az egyszerű benchmarkokban egy nagyon olcsó eszköz is tud közel olyat hozni, mint egy drága. Nem akarom fetisizálni a nagy márkaneveket, de még desktop részében is van olyan tapasztalat, hogy ha kell egy 50-100 gép egyforma, akkor azt csak márkából szabad, mert az ilyen összeszereltekben látszólag egyformát szállítanak, aztán az alaplap egy verzióval arrébb van, más driver, borzasztó dolgok vannak. Tipikus olcsó szerver, mostani állapotra mondok valamit, az nem túl combos, de baj nincs vele. Érdemes lenne még a Linux listákról összehozni, hogy milyen vasakkal milyen borzasztó... pl. SIS chipsetes szerver alaplap. Ugyanez a hálókártya. Broadcom kártyánál nem ilyen egyszerű, mert van belőle egész gyenge is.

Szerverekben sajnos felbukkan az nVIDIA, az nem olyan meggyőző dolog. Sajnos olcsóbb szerverben tipikus az az elrendezés, hogy két darab Broadcom és két darab nVIDIA mert olyan chipset, amiben benne van az nVIDIA. Akkor én ezt úgy csinálom, hogy a Broadcomot használom, amíg lehet és az egyéb dolgokra ott van az nVIDIA.

Egy combosabb elrendezés, látjuk az alaplapra épített Fusion SAS vezérlőt, az már egész szép dolgokat tud és van mellette 4 darab Intel kártya. Ennek az alaplapnak, mikor már lerajzolják, hogy ez a chipset hová van kötve. Eggyel olcsóbb kategóriákban is találni olyat, hogy ugyanaz a chipset, ahol mondjuk a drága gépben egy PCI-os csatornára kötik az egyik diszkvezérlőt, a másikra az Ethernetet, az olcsóbban meg egy csatornára kötik az összes perifériát. Óriási különbség van, amikor kezdem elérni a limitet. Ahol nincs rajz, ott ne legyen illúzió, hogy hogy kötötték.

Nagyon hasznos a szerverszobában van ventilátor, klíma, nem komfort klíma, tehát senki nem szeret hosszan a konzol előtt állni, úgyhogy én nagyon értékelem, ha van ez a szerviz processzor, hogy távolról elérhető, be is lehet rá ssh-zni. Az mondjuk tipikus, az összes Xen-en, dom0-n úgy van megcsinálva, meg azon is, ahol nem virtualizált, hogy a szerviz proc-ra be tudok ssh-zni és egy start konzollal átveszem a sorost és minden alatta lévő gépen soros konzolon van. Tehát még ha lehalt a hálózati interface is, bedugult stb., akkor is soros konzolon el tudom érni a gépet. Ez a szerviz processzor elérhető weben grafikusan is. Nekem most a tipikus kernel nem is szól grafikus felületre nincs framebuffer driver és ilyenek, de soros konzolon elérhető. Ezek a gépeken a BIOS is olyan, hogy soros konzolon, ha újra bootolom a gépet és a szerviz processzorról tudom újra bootolni és lekapcsolni stb., tudok például memória modult kikapcsolni, ha ráfutok egy hibás memória modulra, és ezeket mind távolról el lehet intézni. Sőt van olyan, ahol modemet lehet erre tenni. Ez out-of-band menedzsment jellegre külső telephelyen egy-két autózástól megmenekülhetünk. Kikapcsolt állapotban be lehet lépni és újraindítani. Abban a pillanatban, ahogy bedugod a szerviz processzor bebootol. Aztán vannak olcsóbbak, ahol van szerviz processzor, de tényleg csak akkor indul, ha bekapcsoltam a gépet és tud még egy-két dolgot, a semminél jobb, de az azért más. Általában a szerviz processzor is az IPMI nevű protokollon elérhető, gyakorlatilag ez nevezhető szabványnak olyan ügyben, hogy ventilátor, fordulatszám, hőmérséklet ilyeneket lekérdezgessek és nagyon jól működik. Van is erre Muninhoz plugin-ünk, ipmitools, de sok minden mást is lehet csinálni ezen keresztül.

Firmware upgrade NFS-en. Tehát ha NFS-en exportálok a firmware-t látja, hogy ott a firmware, és bootol a gépem, akkor előbb ráhúzza a BIOS frissítést és utána bootol be az op. rendszeren. Az összes ilyen brand szereti úgy kezdeni, hogy bejelentek valami hibát, hogy nem megy a gépen. Firmware a legújabb? Addig szóba sem állnak velem. Akkor upgradeljük a firmwaret és akkor persze még mindig hibásak. Ez a firmware upgrade lecseréli az alaplapét, a szerviz prociét, az mondjuk egy külön, mert akkor a szerviz processzor is újraindul, diszkeken, meg ha van Fibre Channel kártya ilyeneken. Egy kicsit ellenkezik azzal, hogy egyszerre egy dolgot változtassunk, mert utána,

ha nem megy, akkor mitől nem megy? Tipikus buktató, linux listákról szűrtük le, még onnan, hogy „A 640 kbyte mindenkinek elég kell, hogy legyen”-nek a továbbcsengése. A 3,5 gigabyte-nál is van egy memóriarés. Általában a legtöbb alaplap a PCI I/O-t, azt a 32 bites address, tehát címtartomány tetejére oda 3,5-4 giga közé „meppeli” be. Emiatt van az, hogy szerver szempontból gagyi gépbe veszek 4 giga RAM-ot és 3,5-et lát. Ez a jobbik eset különben, aztán van olyan, hogy szépen „átmeppeli” úgy, hogy megcsinálja azt, hogy 3,5 gigáig ott a RAM és 4 fölött megint látszik, de ez a 32 bites op. rendszereken nem nyert. Tehát ezzel nem biztos, hogy tud mit kezdeni, a 64 bites simán lekezelet. Ez sem olyan szép, meg lehet ezt oldani másképp is. A legdurvább, amire eddig egy kolléga ráfutott, mellesleg viszonylag viszonylag márkás géppel, igaz, hogy annak a márkának a legolcsóbb szerverkategóriája. Senkinek nem szólt, 2 giga RAM-mal virtuális gépekkel gyönyörűen működött, nyilván kevés lett, raktak bele még 2 giga RAM-ot, pár napig elfutott, majd amikor kitalálták a dom-ok, hogy ők most memóriát használnak, akkor szanaszét kócolta a diszket. Ugyanis kiderült, hogy nem „meppeli” ki, tehát a 4 giga RAM ott volt, de ott volt a PCI I/O is és a memóriatartományt I/O utasításoknak véve, szétsikálta a diszkeket. Ma általában 64 illetve 128 gigánál többet nem nagyon lehet belepréselni a gépekbe. A 32 bites rendszer címtartománya 4 giga, ezért rakják oda. Elvileg 4 giga a címtartomány, annak a tetejére berakták. A 32 bites Windows például nem tud vele mit kezdeni.

**Gépkiválasztás:** van, ahol egyszerűen olyan a BIOS nem, régen a laptopokban volt egy olyan korszak, hogy a processzor már tudta a virtualizáció HVM (HardwareVirtualMachine) támogatást, de a BIOS ezt bootoláskor letiltja, és volt egy olyan feature, hogy ezt nem lehetett visszakapcsolni. Tehát volt olyan, hogy a laptop elvileg képes volt rá, gyakorlatilag nem lehetett vele HVM-t.

**Mentés:** szokták keverni a mentés és archiválást, ami két teljesen különböző dolog. Továbbá a RAID az nem számít mentésnek. Mindenhol használok RAID-et, tipikus az, hogy a gépek két diszkről RAID1-gyel bootolnak, a felhasználók meg általában nem RAID1-en, hanem régebben RAID5, RAID6, RAID10. Van olyan elrendezés, hogy co-RAID, hogy itt RAID0 itt RAID1 elrendezések, a fiókok a rackszekrényben szépen összefésülgetve, ezt lehet csinálni. Különböző sebességű adattárolók érhetők el, beszélgettünk erről, hogy a storage-ot is érdemes úgy pakolni, hogy lehet 10 vagy 15 000-es fordulatszámú Fibre Channel diszkekkel, ami ugye két darab Fibre Channel csatlakozó hátrafelé 15 000-et forog és 146 gigásak a diszkek, ez a gyors. És lehet 750 vagy 1 terásat a diszkekből egy másik fiókot. Akkor naponta használt adatok, azok ott vannak a gyorsan és a ritkábban használtak, de még jó, ha online kategória, az ott van a lassabb tárolón. Tehát ezt már a storage cégek mondogatják, hogy életciklusa van, és hogy tegyük el valahova. A szalagon lévő adat csak katasztrófa esetén visszakereshető, az nem online van és az sem biztos, hogy akkor visszajön, de mindegy. Kiderült, hogy a cégnél lévő DAT az fél-analfabéta volt, írni tudott, olvasni nem, ez a jobbik eset. A helyi rendszergazda szépen cserélgette a kazettákat, címkézgette gyönyörűen, az elődje csinálta a rendszert, ő már csak ennyit értett hozzá és crash-elt. Hívtak minket, megpróbáltuk helyreállítani, SUN-os DLT-t hoztunk és visszadugtuk tölteni, de mi van, ha fordítva lett volna? Az egy kicsit nagyobb kaland. Azt mondom most, hogy diszke mentek alapvetően a mai diszkárok mellett. Kis cégeknél tipikus, hogy 8 darab Supermicro 8 diszkes ház, az első két diszk rendszer és bootol RAID1-be, a következő 5 diszk az RAID5, azon vannak a user adatok, a következő meg egy vagy két számmal nagyobb winchester, hot-swappes, tehát pénteken fogja a helyi rendszergazda kihúzza, betolja a másikat és ezt elviszi mondjuk egy másik telephelyre vagy haza magához, nem rárakja a gép tetejére Mert a villám is egy helyre jár. Nem beszélve a takarítónő vizes vödöréről, meg egyéb katasztrófákról. Az USB az nem vált be, előtte még olcsóbb a Firewire-ös külső diszk, nagyobbakba például másik telephely olcsó storage és átnyomjuk. Az már megint messzire vezet, hogy ha storage-om van, akkor block szinten kezdek mentetgetni, erre megint vannak gyönyörű nagy datacenteres szoftverek és izék vagy ésszel csinálom fájlrendszer szinten, nyilván kevesebb adatom lesz fájlrendszerrel általában egy block device-om nincs teljesen tele, meg aztán ott már lehet különbséget is kezelni. Bár a storage-ok is tudnak szépen snapshot-okat meg ilyeneket. Valószínű a jó megoldás a kettő között van, ilyen copy-on-write snapshot,

ugye tipikus, hogy van egy 1 terrás szeletem, tudok egy olyat csinálni, hogy egy pillanatfelvétellel leállok. Például egy adatbázismentésem fut órákig, az adatbázis meg nem állhat órákig. Akkor tudok egy olyat csinálni, hogy azt mondom, hogy snapshot, akkor mondjuk szétkapcsolom. Ha ezek után a rendszer írni akar vagy bárki erre a diszkre, mielőtt felülírja, azt kimásolja a snapshot-ra fenntartott területre, leírja az eredeti helyére. Ez általában arra elég, tapasztalat szerint, hogy 10-20% terület elég, tehát ha van egy 1 terrásom, akkor 250 gigással bőven megvagyok még egy intenzíven használt adatbázis szerveren is, arra, hogy legyen egy ilyen snapshot-om és utána ezt ráérek egy nap alatt lementeni, másolgatni, tömöríteni stb. Ez a snapshot-os dolog így eléggé hasznos. De ezek előtt masszívan gondolkozni kell, mikor, mit, hová, mikor, hogyan, mire? kérdésekre kell válaszolni.

Visszaállítási teszt: DRP (Disaster Recovery Plan), mi csináltattunk ilyen és meglepő dolgok derültek ki. Hiába van storage, ez meg az, kellett egy pendrive, amin például a call router configja meg a Fibre Channel switch configja rajta van, mert amíg az nincs, addig nem tudunk visszaállítani semmit. Persze jött a tanácsadó cég, hogy játsszuk el, hogy a mail szervert újra telepítjük. Ez működik. Egyes számú szabály: ne rontsd el, ne javítsd meg, ha működik. Menjen el a fenébe, de legalább ők leültek, azért jó, mert ő külső szakember, nem azért, mert mi hülyék vagyunk, és nem látjuk, de ő más szemmel látja és úgymond hülyéket kérdez, ami teljesen jogos: azt a configot honnan veszed? Beírhatjuk fejből is, mondjuk van 3 db 48 portos kártya meg két 16 portos kártya, ennek a configját egzaktnak nem biztos, hogy tudom hajnali kettőkor felébresztve.

Mentésről meg mentési stratégiákról, ketté szedtem a mentéseket: rendszer meg felhasználói adatmentés. A rendszer mentésre nálam 97%-ban Debian linuxok futnak. Ha elhalt egy gépem és produkálnom kell nulláról, vissza kell töltenem nagyjából a rendszert, fel kell tennem a webszervert, az adatbázis szervert, ilyeneket, hogy egyáltalán az adatbázis szerverbe vissza tudjam tölteni a mentést. A rendszermentés az, hogy van egy működőképes operációs rendszerem, felhasználói adatok nélkül. Ezt a képen látható szkripttel csinálom, lehet kövezni. Mit csinál? Kitalálja a fájlnevet, beteszi a dátumot, látszik, hogy tgz lesz belőle, ez ugye a mai dátum, gyakorlatilag ez a 2009-04-03 típusú, azért kiírom a fájlnevet. Leszedem a telepített csomaglistát egy fájlba, fogom a csomaglistát meg az etc-t, összetarolom és fölmásolom oda, egy központi helyre. Az hogy az hogyan megy szalagra, meg mit csinálnak vele, az egy másik történet. Nyilván a backup usernél ez a kulcs ott van stb., ssh. Ezek nem túl nagy dolgok, nyilván, ha nekem olyan rendszerem van, hogy az /usr/local-ban vagy valahol vannak mentendő dolgaim, azt értelem szerűen. Például a Notes-on én tipikusan a /var/lib/tftp-t is szoktam menteni, drasztikusan növeli, ott van a hálózati eszköz configjai, ez más user adat felé menő dolog, de ez annyira esszenciális és annyira alacsony szintű, hogy beleraktam. Ennyi, ezt viszont nem automatizálom, nem fut le hajnalban stb. Mielőtt nagyobb konfigurálok a rendszeren előtte nyomok róla egy mentést. Azt azért figyelni kell, hogy a dátumban nincs idő, csak nap, ha egy nap kettőt nyomsz, akkor nem lesz meg az előző állapotod. Le lehet ezt kezelni, de majd a biorobot megoldja.

Adatmentés: ez nagyon bonyolulttá is tud válni. A piacon is elképesztő rendszerek vannak ilyen kezelésre, hogyan van konzisztens állapotba, ki dump-ol az adatbázisomból, hová rakom, hogy tudom visszatölteni, milyen alkalmazásom van, hogy tudom azt megállítani, mikor konzisztens az adata? Itt kérdések vannak sorban, ezt nem egyszerű megoldani.

Programok a tar.gzip tar.bzip mindenki számára ismerős. A cpio az egy elfelejtett nagyon régi és meglepően jó utility, én például gép költözésre (valaki kérdezte, hogy csináljunk nem RAID-es gépből RAID-est) ott is a cpio-t használom például, ahol másolgatni kell. Van már 2 Lenny-s gépem, a laptopomon Ubuntu van. Van még Sarge egy-kettő, de az nagyon fogy, azokból simán lett Etch. Ott azért voltak markáns különbségek, de ahová nem kellett a Woody-ról mennem a Sargeről, ott nem kellett Etch-re sem. Nem kell annyira rohanni a verziókkal, most nem azt akarom mondani, hogy ne olvassuk a bugtrack-et és ne frissítsünk lelkesen, ha kell, de tudjuk, hogy mikor kell.

**Hallgató:** ha kialakít valaki egy jó mentési eljárást, amit ráépít arra, hogy cpio, és utána elkezd ACL-eket használni?

**Előadó:** akkor az kellemetlen. A legtréfásabb, ha nem ő, hanem a userei kezdik el használni a POSIX ACL-eket, az ezzel nem mentődik el. Van rá megfelelő eszköz xfscopy, snapshot stb., meg lehet ezt oldani.

**Hallgató:** megoldás lehet, hogy a getacl-lel kinyerni az összes ACL-t egy fájlba.

**Előadó:** ami nekem bevált, van egy kicsi kis 200/400 gigás LTO-nk (Linear Tape-Open), arra mennek az igazán esszenciális adatok időnként, azt tob (Tape Orienteted Backup) nevezetű bash szkripttel csinálom, a config fájlja is egy bash szkript, működik. Mellesleg afio-t használ. Az afio és a tar között az a lényeges különbség, hogy ha tarral csinállok egy nagy fájlt, és zip-elem és valahol megsérül akár a szalagom, akkor a hibától hátrafelé gyakorlatilag buktam a történetet. Az afio az kicsit kifordítja, ő szeletenként állít elő header-t és fájlonként tömörít. Tehát ha megsérül a szalagom, akkor valószínű, hogy a sérült fájlt buktam, és mögötte is még helyre tudom állítani. Amúgy a paraméterezése eléggé hasonlít, egy átmenet a cpio és a tar között, maradjunk ennyiben. Ez nem túl ismert.

Az rsync is nagyon kellemes, bár ott is vannak olyan opciók, amivel ésszel kell, törölje az extra fájlokat, ami nem létezik, de melyik irányból és hova? Ezzel mellé lehet nyúlni rettenetesen. Ez a tob, ez nekem működött, szalag, címkéz, tud stb., nem túl bonyolult, ellenben azt, amit kell, azt tudja. Többen használják a dirvish nevezetűt, nekem nem jött be annyira, de ez jó direkt ilyen directory mentés. Aztán van az rsnapshot, ez jó. Ez tulajdonképpen rsync, csak egy kicsit okosabb. Dátum alapján csinálja a könyvtárakat, ha kell, akkor hard link-eket használ, tehát én láthatom a tök full backup-omat, és ha nem változtak a fájlok, akkor annyit használ, amennyit, és amikor változik, akkor szétkapcsolja a linkeket, és teljesen jól kezeli.

Csinálunk teljes hard link erdőt a meglévő fájlokra, és arra rá rsynceli az újat, ugye az rsync algoritmus eleve úgy csinálja, hogy letörli a fájlt és helyette egy újat csinál, és ez így teljesen jól működik. Tehát csak, ami megváltozott az lesz meg még egyszer, inode-ból lesz több.

Felhívom a figyelmet arra, hogy a fájl systemek default beállításai esetén beszélünk storage-ről, meg kezdenek röpködni a terrák. Az op. rendszer illetve az NKFS feltételez valami inode adat arányt, és ha olyan fájlrendszerem van, ahol nagyon sok a kicsi fájl, futottam már rá arra, hogy 15% fájlrendszerig telt föl helyügyileg, de elfogytak az inode-ok, mondjuk ez UFS volt, de figyeljünk erre, hogy ha 1-2 terra, meg ennél nagyobb fájl systemeket csinálunk, akkor fontoljuk meg, hogy mire használjuk. Ott gondolkozunk egy kicsit. A linuxban van 2 terra környékén egy block device limit, ami config opció és azonnal túlléphető. 32 bites op. rendszernél egy 16 terrás összefüggő filesize limit van, amivel nem tudunk mit kezdeni. 64 bites rendszerrel nem is merem kiszámolni, hogy ez hova ugrik, de jóval arrébb. Nem is látom értelmét, nekem nem nagyon van 16 terra környéki fájl systemem, inkább valami oszd meg és uralkodj, egy-két terrások vannak maximum. ZFS-n van olyan, hogy egy poolon összefüggően 16 terra, de az is ilyen 1 gigás kvótákra szét van szabdalva.

**Hallgató:** az inode-ok számát nem lehet növelni?

**Előadó:** de, csak amikor létrehozod a fájlrendszert akkor, magyarul, ha van egy 1 terrás fájlrendszered, és rájöttél, hogy kevés az inode-od, akkor ezt félre kell másolnod, újraformáznod és visszatöltened. Ha az 1 terrás fájlrendszered 60%-ig tele van, akkor nem biztos, hogy van helyed, hogy hova másold félre. Amikor ezt mi csináltuk a maildir-rel, akkor maga az újraformázás mondjuk 10 másodperc volt, csak mire a maildir-eket rsyncron-nal áttöltöttük az 4 óra, meg mire visszatöltöttük az a másik 4 óra.

**Layer0 problémák: hova tegyük a gépeket?** Ha fizikailag hozzáférnek a géphez, akkor annyi. Azt nem lehet megvédeni, itt most nem a betörőre kell gondolni, csak jön porszívózni hajnalban a takarítónő és nincs szabad konnektor, akkor kihúzza. És természetesen zárlatos a porszívó, vagy a pirosba dugja, amiben az UPS van.

A combosabb gépekbe, főleg ha diszk is van benne, kell ventilátor, márpedig ezek zúgnak. Ezekkel egy légtérben dolgozni nem egészséges, tulajdonképpen mindenféle előírásnak megfelel, de tegye fel a kezét, aki számítógép monitor előtt dolgozva óránként 15 perc szünetet tart. Nálam a deep-hacking üzemmód, amikor jön egy telefonhívás, hogy tényleg, oda sem mentem el, meg enni is kellene valamit stb. És már pár órája nyomod a billentyűzetet.

A hálózati eszközöknek is jobb egy külön zárható hely, meg hát az embernek sem árt. Magyarul legyen egy szerverszobánk. Na, de milyen? Ugye vannak por meg egyéb veszélyek. Aztán hova tegyük a szervert? Rack-elhető vagy polcos? Sajnos, ha azt mondom, hogy rack-elhető, ez az árcédulán azonnal látszik. A rack-elhető ház, ez ugrik, de egy bizonyos darabszám felett ez a járható út. Egyedül ez a járható út.

**Szellőzés: első kérdés, hogy kell-e klímaberendezés?** A válasz az, hogy igen, kell, de ezt a főnök nem mindig engedi. Ha nincs klímaberendezés? Kisebb teljesítményű és viszonylag nagy légtérnél, mondjuk a kettő aránya számít, elég lehet a légáram is. De ne görcsöljünk, a meleg levegő felfelé akar szállni, ezt segítjük. Tehát ha van egy 10-12 nm-es szobánk, ahová állítunk 4 toronygépet, ez akár jó is lehet, ha az ajtó alján van elég szellőzőnyílás, be tud jönni a levegő és valahol fenn meg ki tudom fűjni és van elég légáram és ahonnan jön, ott nem 35 fokos. Meg nincs benne annyi por, hogy egy év után tönkremenjen tőle a gép. Itt azért jönnek a „ha”-k. Jobb azért az, hogy alul szívónyílás, fölül kifúj. Ennek megint van ám zaja a kifújó ventilátornak, tehát megint ne oda fűjja, ahol ül alatta az ember. Szűrőt azt lehet rá tenni, de azt tudni kell, hogy ezek az axiális ventilátorok, most előjött belőlem az áramlástanos gépészes, ne menjünk bele a jelleggörbéjükbe, de az olyan, hogy nagyon nagy mennyiségű levegőt tudnak szállítani, hanemics ellennyomás. Ha megnő az ellennyomás, akkor drasztikusan csökken a szállított levegőmennyiség.

Ha fogok egy desktop PC-t és 20 cm-re tolom a faltól, akkor kb. 30%-kal csökkent a fűjt levegő mennyiség, ha még 10 cm-t tolok rajta, akkor még megfeleződik a szállított levegő mennyisége. Ha dolgozni kell, arra másfajta szivattyúkat találtak ki, ami nagy nyomást tud előállítani. Egy szerverszobában vannak olyan gépek, amelyik előről hátra fűj, valamelyik innen-onnan, arra, tehát próbálnék kialakítani a gépházon belül is, ha én rakom össze, meg úgy általában erre a kémény hatásra rádolgozni, és hogy egy irányba menjen. Láttam már olyan gépet, közel a halálához, hogy a tápventilátor kifele fűjt, a másik is kifele fűjt, de nem volt honnan kapjon levegőt, vagy még rosszabb, mikor mind a két ventilátor befelé fűj. A befelé fűjtés az azért jó, mert túlnyomásos a ház és nem megy befelé a por. Így csinálják a harci járműveknél is a védelmet, de az egy másik kategória azért.

**Hűtésről:** még egy keverés szokott lenni, a komfort klíma és az ipari klíma egész más, tehát, amit én a szobában lerakhatok kis toronyklímát, az úgymond komfort klíma, az nem fog nagyon hideg levegőt fűjni, az huszon-egynéhány fokos levegőt fog befűjni. Viszont az ilyen, gépterem hűtésre szánt klímák azok mondjuk 8 fokos levegőt fűjnek be az álpadló alá, meg száraz, nem foglalkoznak dolgokkal. Aki ez alá oda áll hosszán, ugye rutinos rendszergazda miről ismerzik meg? Nyáron is ott van a laptop táskájában a termopulcsi, mert ha be kell menni a szerverszobába, például a nagy kánikulában egy nap egyszer megyek a szerverszobába, mert 35 fokban ki-be rohangálok, az tuti megfázás és a strandon hova teszem a zsebkendőt?

A klímaterhelésről egy hozzávetőleges számítási mód, hogy mi kell. Amit elektromosan betáplálunk, azt biztos, hogy elhűtjük. Tehát ha nekem van szünetmentesem, és tegyük fel, hogy elég is, akkor azt el fogom hűteni. Egy dolog még itt bejön, ez a cosinus phi. A szünetmentesek általában voltamperesek, a gépek meg wattosak, régen az ökölszabály az 0,7 volt, amai, úgymond korszerű kapcsolóüzemi tábláknál 0,9 körül van a cosinus phi, de ez most senkit ne zavarjon. Az a lényeg, hogy számolhatunk, meg kell egy kis mérnöki rátartás is, a voltampert azt vegyük wattnak, tekintsünk el a váltóáramtól. Tehát, ha nekem van 2x3000 wattos szünetmentesem, akkor 6 kwatt klíma kell mindenféleképpen. A jó hír az, hogy a 6 kwatt hűtési teljesítményű klíma az 1,5 kwatt elektromos áramot fogyaszt, de ezt még pluszban biztosítanom kell a betápnál. Jönnek a kis 25

amperes megszakítók a szerverszoba mellé. Mondtam már, hogy kettős betáplálás, mert kettős látásom van?

Számítás: kint van 35 fok, bent átlagos épület és 35 fok, akkor 1 kwatt hűtési teljesítmény kell ahhoz, hogy 26 fokot tudjak tartani 1 köbméteren. Úgy kell számolni, hogy a betáplált villamosteljesítmény+az igény. Ha például ablaka van a helyiségnek, akkor a sugárzással jelentős hő jön, tehát azt hozzá kell adni. Ha a szellőzésem kintről levegőt szív be és az nem hideg, akkor annak a levegőnek a fajhőjével, légárammenységével is kell számolni. Nem kell félni a számítástól, ki kell hívni egy klímás szakembert és ő megnézi a géptermet, mond egy klímát és az jó lesz. Csak nem komfortklíma szerelővel, hanem ipari klímás szerelővel kell ez ügyben beszélni. Ennyit a klímáról.

**Hallgató:** ha több órája leállt a klíma, akkor ne fogjuk meg a kilincset, mert égési sérülést lehet szerezni, a gépekről leolvadnak a műanyag alkatrészek.

Nálunk leállítási program van, az a táblázat, amin láttuk, hogy mit hova osztunk, az nálunk úgy van a virtuális gépek mellé. Hogy gold-silver-bronze. Ha áramszünet van, hiába van UPS, a klíma leáll. Tehát ügyeletes, nagios, sprint, fél órája van, hogy beérjen, az ügyeletesnek városon belül kell maradnia és bronze kategória leáll. Ezek után körbe lehet telefonálni, hogy mekkora lesz az áramszünet, ha hosszabb, akkor a gold kategória leáll, Oracle, adatbázisok leállnak, nehogy elmenjen és a webszerver meg a mailszerver megy, amíg az UPS bírja, csökkentett üzemmódban. De az első körnek azért kell hálnia, mert tényleg tud olyat csinálni, kint 35 fok, volt klímaproblémánk áramszünet esetén az első 5 percben 1 fokot fel tud menni a hőmérséklet, 21-ről 36 fokig simán felmegy 20 perc alatt.

**Erősáram:** tényleg elfeledkeznek róla. Kétirányú betáp. Nálunk úgy van megcsinálva, hogy két szünetmentes van a pincében, persze mindegyik a maga fejében redundáns és kétirányú betáppal és a rack szekrények alján ott van 3 kontaset, a nem szünetmentes, az egyik szünetmentes a másik szünetmentes és így dugdossuk be a gépeket. Fontos gépnél mind a két tápját szünetmentesbe, kevésbé fontos gépet az egyik tápba és így tovább.

**Szerverszoba:** ez 60 nm, ott fönt az a két kicsi 2x10 kwatt klíma. Ez a hálózat, itt van a call-router, itt az edge router, ez meg az emc, négy sorban rack szekrények, az első a szolgáltatók beállási pontja. Szervereket lentre kell rakni, mert lentről jön a hideg, a padlóból, tehát felülre olyan szervert rakjunk, amit hamar le lehet állítani, ha klímaprobléma van. A légáramlás nagyon kacifántos, kikísérleteztük, hogy viszonylag mély szekrények vannak, és ha az első ajtót behajtuk és alulról jön be a hideg levegő, ez a jobb nekik, nem a nyitott ajtó, mert az előző szekrény soré akkor fölfele megy. Elöl van három klíma, ami befúj a padló alá, a szekrényből jön a levegő, a szekrény hátul kifúj, megy föl és később beszereltük azt a két klímát. Kialakul egy ilyen áramlás a gépteremben, és ebbe az áramba építettük bele ilyen terelő lemezzel, azt a két 10 kwattos klímát, ami besegít. Ez azt a levegőt hűti le, amit a 3x15 kwatt beszív és benyom a padló alá. Ezek olyan minimális nyomással dolgoznak, hogy tesztelni sem egyszerű, mert lehet azt mondani, hogy füsttel kipróbálom, de van tűzjelző berendezés is, úgyhogy óvatosan. Minden szekrény sor tetején van optika meg réz, ez megy oda a központi hálózatos szekrény sorba és így tudom pecselgetni, hogy mit hova akarok kötni. A „v” az a szünetmentes egyik szünetmentes, másiktól jövő táp, ez minden szekrény aljában így van. A szünetmentes 1,4 tonna, van két vezérlőegysége a szünetmentesnek, 4 kvoltamperes vezérlőmodulokból 5 db és így 16 kvoltampert tud, mert egy a redundancia miatt mindig tartalékban van. A második torony az csak az áthidalási időt növeli, körülbelül teljes terhelésen olyan 1 órát bírunk. De mivel a klíma miatt úgyis le kell állítani, a b-tervben már sokkal jobb a helyzet és úgy általában a minimál konfigurációban 6-8 órát át bírunk hidalni. Akkor nem megy a klíma, de néhány szerverünk megy.