Műsorszórás: nem csak a helyi hálózaton (2. rész)

A VideoLAN projekt egyéb műsorszóró és különleges képességei.

Sorozatunk előző részében már bemutattuk a VideoLAN helyi hálózatokon alkalmazható műsorszóró képességeit, ám mint azt a cikk végén említettem, a dolognak ezzel messze nincs vége. Például nem használtuk ki a program nyújtotta apró, ám annál hasznosabb lehetőségeket, kényelmi megoldásokat, gondolok itt az adatfolyam valós idejű átalakítására, az egyidejűleg többféle protokollon történő sugárzásra, vagy épp a kényelmes webfelületről irányítható televízióállomás lehetőségére. Ezek mindegyikéről hamarosan szót ejtünk, de ez önmagában kevés lehet annak az olvasónak, aki számára szűknek bizonyulna a helyi hálózatok szabta mozgástér. Valóban akad a dolognak némi hátránya, ugyanis tömeges nézőtábort ezzel a módszerrel jelenleg még nem toborozhatunk. Azonban kár lenne azt hinnünk, hogy csak ilyen szűk keretek között mozoghatunk, ezért szeretnék néhány olyan módszert megismertetni, amellyel kitekinthetünk a helyi hálózatok világából. Részben ehhez kapcsolódóan fogunk megismerkedni azokkal a bizonyos, eddig nem használt képességekkel, és próbáljuk olyan módon felépíteni a képzeletbeli televízióállomásunkat, hogy az később megfeleljen az új követelményeknek. Ilyen új követelmény lehet például az, hogy HTTP protokollon keresztül távoli ügyfelek számára is elérhetővé váljon a sugárzott műsorfolyam. Látni fogjuk, hogy bizony létezik néhány körülmény, amely eltérő a helyi hálózatok esetében tapasztaltaktól. Vágjunk bele!

Átkódolás

Mint tudjuk, a videofolyamok erőteljes tömörítésen és átalakításon mennek keresztül, mielőtt elnyerik tárolható, különleges formátumukat. A tárolhatóság mellett némely formátum kifejezett célja, hogy könnyen sugározható legyen az internet kusza vezetékein is. A helyi hálózatokon ilyen gonddal megfelelő lejátszók esetében nem kell küszködnünk, azonban a fájlok tömörségét, mindenképp érdemes kihasználnunk, ezzel ugyanis sávszélességet takaríthatunk meg. Ezenkívül jó nekünk az, hogy létezik egy előre kialakított befoglalókeret, így az adatcsomagok kompakt, jól kezelhető egységeket alkotnak. Ezen okok, s részben a módszer egyszerűsége miatt a VLC az adatfolyamot a lemezen tárolt formában, az eredeti kódolással sugározza. Ez természetesen nem azt jelenti, hogy az adatfolyamot kivágja a hálózatra, csupán annyit, hogy nem nyúl a kép- és hangsávokhoz, „változatlan” állapotban továbbítja őket.

Felmerülhet a kérdés, hogy mi történik, ha valaki előre meghatározott formában szeretné a különböző módokon tárolt anyagait ilyen módon közzétenni? A megoldás az adatfolyam menet közben (on-the-fly) történő programbeli átkódolása. A módszer lényege, hogy a kép, illetve hangsávok nem a lemezen tárolt formátumban kerülnek sugárzásra, hanem dekódoljuk a kiszolgálón, s azonnal újrakódoljuk, majd az így kódolt adatfolyamot küldjük el a hálózaton. Ennek így elsőre nem sok értelme van, de ha jobban a dolgok mögé nézünk, látni fogjuk, hogy ez egy remek lehetőség.

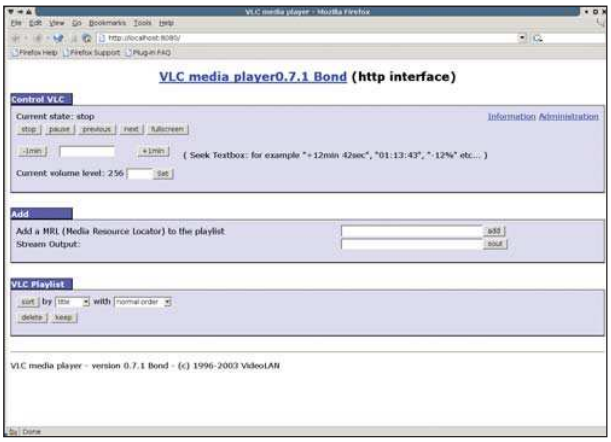
Egyik ilyen szükséglet a szabványosság. Ugyanis elképzelhető, hogy mi minden adatot ugyanolyan formában szeretnénk eljuttatni a nézőkhöz. Mint tudjuk, a digitális kép- és hanganyagok a lehető legváltozatosabb formában és kombinációban állhatnak rendelkezésünkre, tehát elég nagy a valószínűsége annak, hogy átalakításra szorulunk.

A másik olyan kényszer, ami miatt átalakításra szorulhatunk: a sávszélesség szűkössége. Előfordulhat, hogy a tárolt videofájl bitrátája túl nagy a hálózatnak, amelyen sugározni szeretnénk. Ez lehet azért, mert túlterhelt, vagy épp azért, mivel egy „átjárót” létrehozva szeretnénk egy kimenő kapcsolaton másik hálózatba juttatni az adatfolyamot, és a távoli hálózattal szűk vonalon vagyunk összekötve. Ilyenkor az a bevett szokás, hogy az adatfolyamot nagyobb veszteséggel áttömörítjük, így kisebb méretű (ugyanakkor gyengébb minőségű) lesz.

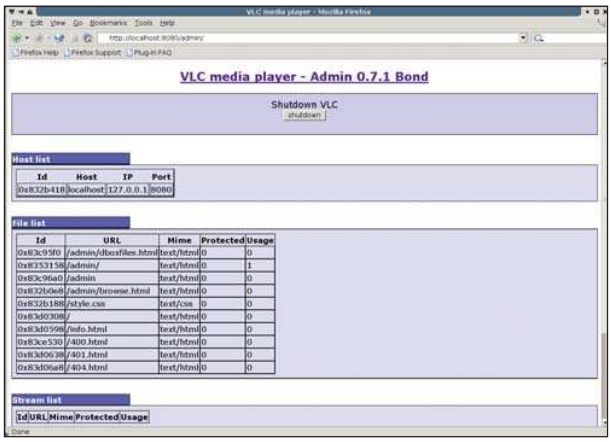
A VLC használata során remekül megoldhatjuk az ilyen jellegű nehézségeket, s ehhez mindössze néhány tíz karakterrel szükséges bővítenünk a parancssorban kiadott utasításunkat. Az előző cikkben említettem a VLC kimeneti moduljainak használatát, ezeket a modulokat nagyon egyszerű parancsfájl segítségével állíthatjuk be. A kapcsoló formátuma a következő:

```
--sout
#modul1{jellemző1=...,jellemző2=...}:#modul2
➥{jellemző1=...,jellemző2=...}:...
```

Adatfolyam átkódolása esetén nekünk épp egy ilyen kimeneti modul használatára lesz szükségünk, melynek neve: `transcode`. A `transcode` modul a bemeneti adatfolyamot magán átereszti és a kimeneten átkódolt adatfolyamként jeleníti meg, s mindezt valós időben teszi, azaz a bemenetet



1. kép A VLC HTTP-felülete



2. kép A VLC HTTP-felülete – felügyeleti oldal

menet közben (on-the-fly) alakítja át. A módszer egyetlen hátránya, hogy mivel a tömörítő algoritmus aszimmetrikus, a tömörítés meglehetősen gépigényes, így jó erős processzorra van szükségünk ahhoz, hogy az átkódolás folyamatos legyen.
Nézzünk most egy példát az átalakításra! Adjuk ki a következő utasítást a parancsértelmezőben:

```
vlc -vvv <bemeneti adatfolyam> --sout
➥#transcode{vcodec=mp4v,acodec=mpga,vb=800,ab=128,
➥deinterlace}:standard{access=udp,mux=ts,
➥url=239.255.12.42,sap=Probafolyam}
```

A parancs hatására a bemeneti adatfolyamból MPEG-4 formátumú képsávból és MPEG audio layer 2 formátumú hangsávból álló átviteli adatfolyam keletkezik (ez gyakorlatilag a szabványos DVB adatfolyam, de ilyennel találkozhatunk a DVD-ken tárolt videók esetében is), amely a 239.255.12.42-es csoportos üzenetküldési címre továbbítódik némi csatornainformációval fűszerezve. Számunkra most leginkább a `transcode` modul jellemzői az érdekesek, vizsgáljuk meg egy kicsit.

`vcodec` (video codec): segítségével adhatjuk meg az új képsáv tömörítéséhez használandó kodeket.
`acodec` (audio codec): ezzel a jellemzővel állíthatjuk be a használni kívánt tömörítő eljárást a hangsávra vonatkozóan.
`vb` (video bitrate): meghatározza, hogy mekkora lesz a kimeneti adatfolyam mérete (kbit/másodperc). Gyakorlatilag ennek segítségével szabályozható, hogy a tömörítés során az eredeti képanyaghoz képest mekkora legyen a veszteség mértéke. Minél nagyobb ez a szám, annál nagyobb az adatfolyam mérete, s annál jobb a kép minősége.
`ab` (audio bitrate): ugyanazt jelenti, mint a vb jellemző, csak épp ez a hangsávra vonatkozik.

Egyéb fontos kapcsolók:
`width`, `height`: a kép méretét (szélességét, magasságát) adhatjuk meg vele, így méretezhetjük át az eredeti képet.
`croptop`,`cropbottom`,`cropleft`,`cropright`: ezekkel a kép szélein jelentkező vékony fekete kereteket vághatjuk le a négy irányból.

Terjedelmi okok miatt az adatfolyamok szerkezetéről, az alkalmazható kodekekről nem tudok szót ejteni. Erről egyébként egy teljes cikk jelent meg a Linuxvilág 15. számában „Digitális videózás” címmel. Azok az olvasóink, akik mélyebben érdeklődnek a téma iránt a Linuxvilág honlapján (vagy előfizetőként) elolvashatják.

Többszörös adatfolyam-sugárzás

A módszer lényege az, hogy a bemeneti adatfolyamot nem egyetlen kimeneten tesszük közzé, hanem egyszerre több címre, több kimenetre is elküldjük. Ez akkor lehet hasznos, ha például szeretnénk kilőni az adatfolyamot a helyi hálózatból egy általunk meghatározott gépre, IP-címre úgy, hogy közben a helyi hálózaton a többiek is lássák az adást. Példaképp álljon itt egy helyzet: tegyük fel, hogy van egy cégünk két telephellyel, két helyi hálózattal, s a két telephely valamilyen szűkebb kapcsolattal össze van kötve. Azt szeretnénk, hogy az egyik telephelyen sugárzott adást mindkét helyi hálózaton megtekinthessük. Mivel az UDP protokollon küldött csoportos üzenetek a helyi hálózatok határán érvényüket vesztik. Ha csak nincs beállítva egy csoportos üzenetek továbbítására alkalmas átjáró (gateway), nekünk egy külső IP-re, a másik telephely átjárójára is el kell küldenünk az adatfolyamot, amely a két telephely közötti közvetlen kapcsolaton halad át, ily módon gyakorlatilag kézzel hajtjuk végre az útvonalválasztást (route). Ezután a másik oldalon a bejövő adatfolyamot ismét szét kell dobni az ottani helyi hálózatra, s készen is vagyunk.
Ha valami hasonlót szeretnénk csinálni, akkor megint csak a fentebb említett modulokat kell segítségül hívnunk, közülük is a `duplicate` nevűt, amellyel különböző kimeneti célokat adhatunk meg egyazon bemeneti adatfolyamhoz. A modul neve némiképp sántít, ugyanis azt sugallja, hogy megkétszerezi a kimeneteket, de valójában n-szerezi, ugyanis nemcsak két kimenetet határozhatunk meg, hanem többet is.
Egy ilyen parancs a következőképpen fest:

```
vlc -vvv <bemeneti adatfolyam> --sout
➥'#duplicate{dst=standard{access=udp,mux=ts,url=
➥239.255.12.42,sap=Probafolyam},dst=standard
➥{access=udp,mux=ts,url=192.168.1.2}}'
```

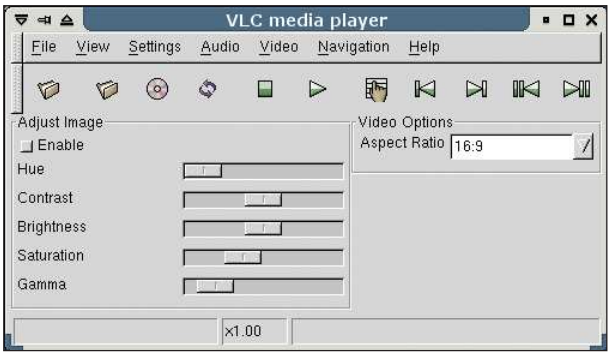
Ha közelebbről megvizsgáljuk a modult, láthatjuk, hogy a `dst` (destination = célállomás) jellemzővel adhatjuk meg egymás után a kimeneteket, vesszővel elválasztva. Láthatjuk, hogy a `dst` jellemző után a teljes adatfolyam-szerkezetet meghatározhatjuk, azaz minden adatfolyamot különböző kapcsolókkal láthatunk el, mintha valóban különböző kiszolgálókról származnának.
A fenti két lehetőséggel, tehát az átalakítással és a többszörös adatfolyam-sugárzással, azaz inkább ezek kombinálásával mindenféle érdekes dolgot és számtalan lehetőséget kaphatunk. Mivel egy adott kimenetet a teljes adatfolyam-szerkezettel jellemezhetünk, megoldható, hogy a helyi hálózaton az eredeti, kiváló minőségű ám nagy sávszélesség-igényű adatfolyam csordogáljon, ugyanakkor a kimenő, kis sávszélességű vonalon egy menet közben (on-the-fly) átalakított, rosszabb minőségű, kisebb méretű műsor továbbítódjon, s mindegyik adatfolyamot tetszés szerint testreszabhatjuk.

A VLC irányítása HTTP-n keresztül

A VLC-t a fejlesztők egy egyszerű kis HTTP-kiszolgálóval is felszerelték. Ez a kiszolgáló végzi egyrészt a VLC távoli irányítását, másrészt a HTTP alapú adatfolyam-sugárzást. (Érdekes tény, hogy a SHOUTcast esetében is épp egy ilyen felépítésű és szerepű HTTP-kiszolgálóval találkozhattunk, talán nem véletlenül. (Linuxvilág, 35. szám. 38–41. oldal, Bemutatkozik a SHOUTcast)) Nézzük előbb a webről történő vezérlést!
Ha igényünk támadna, hogy távolról szeretnénk irányítani a VLC-t kiszolgálóként, akkor ezt részben elintézhetjük valamilyen távoli terminálról. Ez azonban nem ad lehetőséget a pozícionálásra, és a sugárzott adatfolyam kicserélése is bajos lehet, hiszen a VLC grafikus felületéhez azon a bizonyos idegen gépen egy X-kiszolgálónak kell futnia, és egyébként is nehézkes egy ilyen rendszer kezelése. Ennek megoldására a fejlesztők ellátták a VLC-t egy-két apró CGI parancsfájllal, ennek segítségével a kiszolgálóhoz csatlakozhatunk, mint webkiszolgálóhoz, s immáron grafikus felületről, kényelmesen kezelhetjük kiszolgálónkat, szinte bárhonnan a világból, s nem kell hozzá semmi egyéb, csak egy böngésző.
Nekünk ehhez semmi mást nem kell tennünk, mint a kiszolgálón elindítani a VLC-t az alábbi utasítással:

```
vlc -I http --http-host 192.168.0.102:8081 --sout
➥udp:239.255.12.42
```

Ezután, ha valamelyik másik gépen (vagy akár magán a kiszolgálón) ellátogatunk a ***http://192.168.0.102:8081*** címre, akkor egy kezelőfelületen találjuk magunkat.
A felület vízszintesen három részre tagolódik. A legelső a pillanatnyi adatfolyamot kezeli. Lehet benne pozícionálni, ide-oda tekergetni és tud váltani a lejátszási lista elemei között. A következő részben adhatjuk hozzá a sugározni kívánt fájlokat a lejátszási listához. Ehhez meg kell adnunk a fájl teljes elérési útját. Lehetőség nyílik még a kimeneti forrás meghatározására, ám ez valamiért nem működik, talán nincs még belerakva a kódba, épp ezért már az indításkor célszerű megadni, mint ahogyan azt meg is tettük a fenti paranccsal. A legutolsó részben a lejátszási listát kezelhetjük, megváltoztathatjuk az elemek sorrendjét, törölhetünk közülük, s ha bármelyikre rákattintunk, a kiszolgáló onnantól kezdve azt kezdi el sugározni.



3. kép A VLC grafikus felülete kiterjesztett módban

HTTP alapú műsortovábbítás

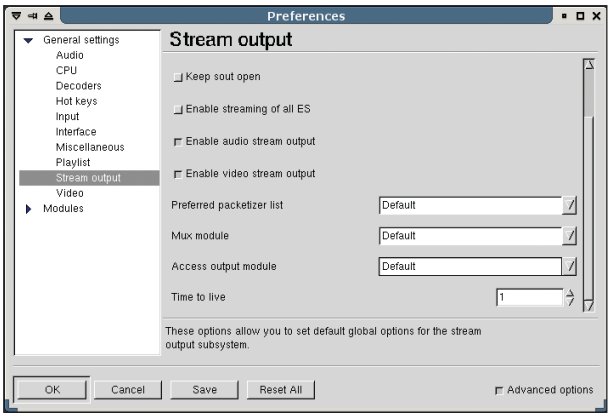
Gyakorlatilag ez az a módszer, amelyet a webrádióknál már megismertünk. Ilyenkor az adatfolyam úgy érkezik az ügyfél gépére, mintha csak egy HTML dokumentumot töltene le valamelyik kiszolgálóról. Itt természetesen nincs lehetőség a csoportos üzenetküldésre, így ehhez hatalmas sávszélességre van szükségünk, ugyanakkor megvan az az előnye, hogy könnyedén kilép a helyi hálózatok nyújtotta szűkös területről, feltéve, ha az ügyfélig elegendően nagy a sávszélesség.
Ha ilyet szeretnénk, akkor kimeneti forrásnak ezt a bizonyos beépített HTTP-kiszolgálót kell megadnunk. Figyelem, ez nem ugyanaz, mint az előbbi webről történő vezérlés. Ott csak egy felületet biztosítottunk a beavatkozásra, itt viszont azon fogjuk továbbítani az adatfolyamot, természetesen a gép egy másik kapuján (port).
Próbáljunk indítani most egy ilyen szolgáltatást. Adjuk ki az alábbi parancsot:

```
vlc -vvv <bemeneti adatfolyam> --sout
➥#standard{access=http,mux=ts,url=localhost:8080}
```

Az URL helyére természetesen azt a címet írjuk, amelyen a gépünket az adott adatfolyamhoz való hozzáférés érdekében elérni szeretnénk. Arra ügyeljünk, hogy ilyenkor a HTTP-kiszolgáló az URL-t virtuális kiszolgálóként kezeli, azaz csak az adott cím beírásával érhető el, ha mondjuk, a gép IP-jét címezzük, nem fogunk látni semmit.
Ha ez megvolna, csatlakozzunk az előbb létrehozott kiszolgálóhoz, először VLC-vel az alábbi módon:

```
vlc http://localhost:8080
```

A parancs hatására megkezdődik a videó lejátszása, mi pedig örülhetünk, hogy ezt bárki akár távolról is elérheti. Azonban itt is szükség lesz a VLC médialejátszó-képességeire, s az ügyfél gépén is jó, ha ez fut, ugyanis bármilyen formátumú HTTP-vel továbbított videó lejátszására képes. Egyéb esetekben, például MPEG-1 vagy MPEG-2 (VCD, SVCD) továbbításakor az MPlayer nevű lejátszó is minden gond nélkül boldogul a HTTP-ről történő lejátszással, de



4. kép A VLC grafikus beállítópanelje

gondolom ezen senki nem lepődik meg. Az viszont már meglepő, hogy egy szabványos, szakaszolható (streamelhető) MPEG-1 adatfolyamot ily módon sugározva a Windows Media Playerrel nem sikerült megnyitnom. Mint tudjuk, a weben elérhető különböző videotartalmakat teljesen hasonlatosan játssza le menet közben a Windows Media Player, ha a böngészőben rákattintunk akár az adatfolyam, akár egy fájl hivatkozására. Ennek ellenére a program állandóan hibaüzeneteket küldött. Folyamatosan tudta átmenetitárazni (cache), böngészőből a lemezre menteni, onnan is képes volt megnyitni, tehát az adatfolyam hibátlanul eljutott a célgéphez, ám menet közben nem volt képes az adatfolyam lejátszására. Gondolom közben mindenki kitalálta, hogy ezzel azt szerettem volna elérni, hogy az ügyfél-oldalon ne kelljen telepíteni „különleges” programot azért, hogy tévézhessünk a webről, de sajnos a legelterjedtebb lejátszó ezt nem tudja. Feltételezem, hogy egyéb programokkal megvalósítható a dolog, de ha már úgyis telepíteni kell, akkor használjuk inkább a VLC-t, Windows alatt az a legalkalmasabb – talán még filmnézésre is.

A grafikus VLC

Bár eddig mindvégig parancssorból vezéreltük a VLC-t, de lehetőségünk van a grafikus felület igénybevételére is, ezen jelentős mennyiségű kapcsoló kiváltása lehetséges, vagyis a beállítások palettája meglehetősen színes. Bár van, amit csak parancssorban tudunk megadni, azért rengeteg testreszabási lehetőséget ad a VLC grafikus felülete. A fentebb említett modulok szinte mindegyikét beállíthatjuk, ezenkívül a program egyéb részei is kiválóan vezérelhetők. Ahhoz, hogy mindezt elérjük, ne felejtsük el a beállítópanel jobb alsó sarkában engedélyezni a haladóknak szóló nézetet.

VOD

Ez a módszer (Video-On-Demand= igény szerinti videózás) egy kicsit el is tér az eddig tárgyalt műsorszórástól, hiszen az eddigiekkel ellentétben itt nem az adatfolyamnak ugyanahhoz a szakaszához kapcsolódik mindenki, hanem lehetőség nyílik igény szerint egy adott szakaszhoz kapcsolódni. Ez a gyakorlatban annyit jelent, hogy bármikor elkezdhetjük nézni a filmet, előre és visszatekerhetünk, megállíthatjuk stb. Majdnem olyan, mint a tévéadás és a VHS kazetta nézése közötti különbség, leszámítva azt az aprócska tényt, hogy itt nem áll rendelkezésünkre a kazetta.

A VideoLAN által javasolt megoldás igen egyszerű, így hangzik a receptjük: végy egy tetszőleges webkiszolgálót, például az Apache-ot, tégy elérhetővé egy videofájlt, ezzel a kiszolgálóoldalon meg is tettünk minden szükséges lépést. Amit hozzáad a VideoLAN, az annyi, hogy a VLC képes lejátszani az itt található fájlt anélkül, hogy az egészet letöltené.

Az egyszerűség annak köszönhető, hogy a HTTP protokollra bízzák a VOD módszer közben felmerülő gondok megoldását. Ez azt jelenti, hogy a HTTP 1.1 lehetőséget biztosít a `GET` művelet során a kiszolgálón lévő fájlokban történő pozícionálásra. (Ilyen pozícionálás történik akkor is, ha például befejezünk egy megszakadt letöltést.)

Ha tehát ezzel a megoldással videót nézünk, gyakorlatilag a webről töltünk le, csak épp nem írjuk lemezre a letöltött adatfolyamot, s eközben még ide-oda ugrálunk is benne.

Próbaképp tegyünk elérhetővé a gépünkön lévő webkiszolgálón keresztül egy videofájlt, lehetőleg valamilyen MPEG-et. Ez azért hasznos, mert szabványos és szakaszolható, s az sem árt, ha a többi módszerrel történő sugárzás esetén is mindig MPEG adatfolyamok továbbítására törekszünk.

Ha megvolnánk a fájlok közzétételével, az alábbi módon csatlakozhatunk az adatfolyamhoz VLC-vel:

```
vlc -vvv http://localhost/filmek/film1.mpg.
```

Mondanom sem kell, hogy a szolgáltatás sávszélesség-igénye mellett az eszközigénye is óriási. Itt ugyanis annyi szálon kell olvasni a merevlemezt, ahányan csatlakoznak hozzánk, annyiszor kell szétválasztani a kép- és hangsávokat, egyszóval minden művelet n-szereződik, arról nem is beszélve, hogy ebben a helyzetben a csoportos üzenetküldésnek még az elve is kizárt, hiszen nincs két egyforma csomag. A sok nehézségért cserébe viszont korlátlan szabadságot kapunk, mint tévénézők.

Tanulság

Ha jobban megnézzük, lehetőségeink szinte korlátlanok, s mindezt teljesen ingyen igénybe vehetjük. Kellő szakértelemmel egy valódi videóstúdió építhető néhány Linuxszal és a VideoLAN segítségével, amelyben talán az a legszebb, hogy elérhető közelségben van, bárki megvalósíthatja.

A dolog egyetlen szépséghibája, hogy közönség terén igencsak hadilábon állunk, erre valódi tévéműsort alapozni még nem lehet, ezért kezdetben maradni fog a helyi hálózatokon történő, ideiglenes, ám rendkívül hasznos próbálkozás, hogy aztán később szélesebb tömegekhez is eljuthasson a digitális adás, hiszen minden kétséget kizáróan ez a műsorszórás jövője.

Komáromi Zoltán
(komi@kiskapu.hu)
23 éves, a BME hallgatója, mellette PHP-programozóként dolgozik. Kedvenc területe a multimédia.