Médiaszolgáltatás és CDN egy távközlési szolgáltató szemszögéből

JURSONOVICS TAMÁS
Deutsche Telekom
tamas.jursonovics@telekom.de

Kulcsszavak: CDN, streaming, automatizáció, TLS, IPTV

Az over-the-top streaming platformok terjedése új technológiai lehetőségeket teremt az internetszolgáltatók számára TV-rendszereik újragondolásához. A multicast MPEG-TS-alapú műsorszórás lassan háttérbe szorul az adaptív streaming egyre jelentősebb előnyei, az egységes kliensfejlesztés, a célzott reklámok, a WLAN-támogatás mellett, ami azonban határozott kihívást támaszt a felhasználószámmal arányosan növekvő sávszélességigény által. A Telekom által épített CDN bemutatása ezen kihívásokra kínál egy hatékony megoldást nyílt forráskódú szoftverek, konténertechnológia és teljes automatizáció formájában, mely egy extrém skálázható, hibatűrő, költséghatékony és flexibilis CDN-rendszert eredményez.

1. Bevezetés

Az over-the-top (OTT) szolgáltatások térnyerésével párosult innováció előtérbe helyezi az IPTV-szolgáltatók által használt műszaki megoldások időszerűségének és hatékonyságának kérdését. A közvetlen verseny az igény szerinti videó (VoD) területén és az élő műsorszolgáltatás lassú összefonódása a közösségi médiával arra ösztönzi az IPTV-szolgáltatókat, hogy folyamatosan keressék a megújulás lehetőségét helyzetük és szerepük megőrzése érdekében. Ezen törekvés egyrészről üzleti oldalon mutatkozik meg új előfizetői ajánlatok, értéknövelt szolgáltatások és szövetségek formájában, másrészről az IPTV műszaki újragondolásában. Tekintsünk csak a belső és külső hálózatokban használt különböző műsorszóró protokollok, az MPEG-TS multicast és az adaptív streaming által támasztott eltérő követelményekre, mint a lejátszó alkalmazások heterogén fejlesztési környezetére, a nem egységes *personal video recorder* megvalósításokra, a többféle hardverplatformra, és így tovább.

Ahhoz, hogy egy IPTV-szolgáltató felvehesse a versenyt az OTT-megoldásokkal, erőforrásait egy irányba kell összpontosítania. Ezt felismerve, a Telekom a jövőben felhagy a multicast alapú műsorszórással és kizárólag a HTTP-streaming használatára koncentrál, ami nagyban leegyszerűsíti az IPTV felépítését.

Ez azonban azt is jelenti, hogy a multicast által biztosított hatékonyságot nem lehet többé kihasználni, az előfizetőszámmal arányosan nő a műsorterjesztés hálózati erőforrásigénye. Ez a folyamatosan bővülő optikai hozzáférésekkel, illetve egy hatékony, saját tartalomelosztó hálózattal (CDN) ellensúlyozható, melynek mindenképpen ki kell szolgálnia az alábbi szempontokat: skálázhatóság több tíz terabit per másodperces tartományban, alacsony költségek, minimális fejlesztési erőforrások, flexibilitás, könnyű üzemeltethetőség.

A soron következő szakaszok ennek a CDN-nek a tervezése, építése során felmerült kihívásokat és az ezekre adott válaszainkat mutatja be: hogyan alkalmazkodtunk a belső és külső hálózatok eltérő igényeihez, miként tartottuk a fejlesztési erőforrásokat minimálisan, és használunk konténerizációt az egyszerű üzemeltethetőség érdekében, mely architektúra biztosít nagyfokú skálázhatóságot, miként választottuk meg az automatizálás mértékét, milyen lehetőségeink vannak a rugalmas skálázhatóság eléréséhez. Ezek után szó esik a jelenleg tapasztalt problémákról és ezek jövőbe mutató megoldásainak lehetőségeiről.

2. Külső hálózatok

A HTTP-streamingszolgáltatások alapvetően más igényeket támasztanak a tartalomelosztó hálózatokkal szemben, mint a webböngészés. Míg egy weblap megnyitása során az oldalon szereplő képek és scriptek letöltési sebessége nem kritikus tényező, azaz a felhasználók elfogadják, ha egy oldal néha lassabban jelenik meg az esetlegesen ingadozó elérhető sávszélesség miatt, addig tévénézés közben a legkisebb kiesés sem tolerált [1]. Az átviteli technológiától függetlenül az előfizetők elvárják az évek alatt megszokott kábeles/műholdas szolgáltatásminőséget. Hasonlóan, egy igény szerinti videó (VoD) késleltetett indulása még tűrhető, de a filmnézés közben egy esetleges képmegállás elfogadhatatlan.

Mindezek elkerülésére két megoldás terjedt el. Az első esetben – hasonlóan a multicast alapú műsorszóráshoz –, a videóátvitel IP-kapcsolatához rendelt szolgáltatásprioritási szint (DiffServ) biztosítja az elsőbbségi továbbítást, így a saját hálózat megfelelő méretezése mellett a szolgáltatás minősége garantálható. Mobil és Wi-Fi hozzáférés esetén a sávszélesség nem tervezhető, illetve külső hálózat esetén a szolgáltatásprioritás nem elérhető, ezért egyetlen járható út marad: kompromisszum keresése a felhasználói élmény tekintetében [2]. A videó felbontásának csökkenése, vagy tömörítési mér-

tékének növelése még mindig elfogadhatóbb a felhasználók szempontjából, mint a rövid szünetek és megállások, ezért a bevett gyakorlat szerint a multimédiás tartalmak több profilon (más sávszélességeken) is elérhetőek, melyek közül a lejátszó alkalmazás választ az aktuális sávszélesség alapján, ezzel minimalizálva egy esetleges képmegállás vagy újratöltés valószínűségét.

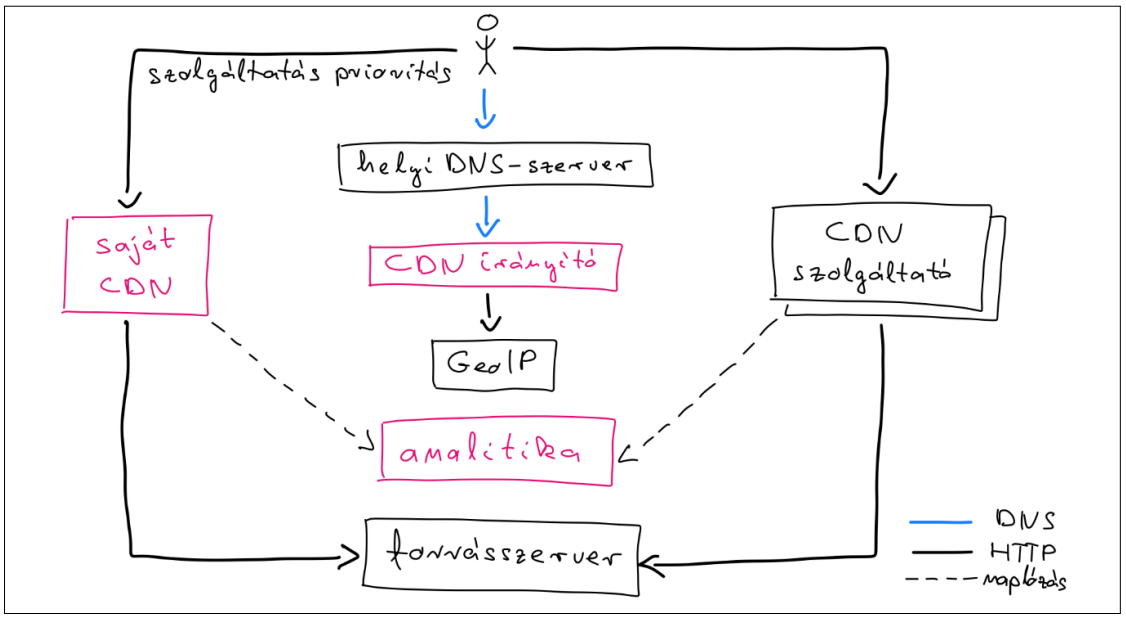
Mint látható, egy internetszolgáltatónak több megoldás is rendelkezésére áll a saját hálózatában a kívánt felhasználói élmény eléréséhez, azonban a külső hálózatok esetében a meghatározó tényező a sávszélesség. Ennek biztosítására az alábbi lehetőségek adódnak: peering kapcsolatok kiépítése vagy bővítése, azonban ez stratégiailag megfontolandó döntés, hiszen ezen kapcsolatok kétirányúak, melyek közvetlen, szabad utat nyitnak a szolgáltató hálózatába harmadik felek számára a peering partneren keresztül. Alternatívaként szóba jöhet CDN-kapacitás telepítése idegen szolgáltatóknál, de a felmerülő komplex üzemeltetési, biztonsági és versenytársi kérdéseket is figyelembe kell venni. Végső és egyben neutrális megoldásként a harmadik fél által biztosított CDN-szolgáltatások igénybevétele (Akamai, Cloudfront) említhető. Utóbbi előnye a tervezési és üzemeltetési feladatok kiszervezése, hátránya azonban a relatív magas díjak, melyek csökkentése érdekében több CDN-szolgáltatóra célszerű támaszkodni a köztük lévő versenyhelyzet kihasználása érdekében.

A Telekom IPTV-szolgáltatásának biztosításához a belső hálózatban IP-priorizálás, a külsőben két CDN-szolgáltató mellett döntöttünk *(lásd az 1. ábrát)*.

Ebből három további feltétel következett. Először lehetővé kellett tennünk, hogy a felhasználók egységesen kommunikálhassanak mindhárom CDN-rendszerrel. Itt elsősorban nem a HTTP-protokoll alkalmazására gondolunk, hanem a CDN-partnerek által előszeretettel ajánlott (és erősen reklámozott) értéknövelt szolgáltatásokra, mint a hitelesítés, jogosultságkezelés, címezhető TV-hirdetés és vízjelezés. Ezek első benyomásra megkönnyítik a szolgáltatás létesítését, azonban – szabadalmaztatott technológiáik révén – erős függőséget alakíthatnak ki a szolgáltató irányába. Ennek elkerülése érdekében a CDN-rendszer tervezése során az alapvető funkciókra koncentráltunk, és minden olyan szerepet, mely nem valósítható meg a HTTP-protokoll által biztosított tágabb keretek között, a CDN-en kívül realizáltuk. Szerencsére ez számunkra, egy IPTV-szolgáltató számára könnyű feladat, hiszen mind a kliensfejlesztés, mind a háttér-infrastruktúra a saját hatáskörünkben van, így a CDN-integrációt szabadon meghatározhattuk.

Másodszor a felhasználókat a megfelelő CDN felé kell irányítani. Ehhez igénybe lehet venni külső megoldásokat (Cedexis Openmix), vagy ahogy a Telekom esetében, egy saját fejlesztésű hiteles névszerverrel (authoritative nameserver) és egy GeoIP-szolgáltatással ez a funkció egyszerűen megvalósítható. A lényeg mindkét esetben, hogy az irányítás DNS-alapon történjen (CNAME mapping), mivel egy extra HTTP-átirányítás és tipikusan a vele járó TLS-kézfogás minden egyes csatornaváltáshoz hozzáadott késleltetése rontaná a felhasználói élményt. Megjegyzem, hogy míg HTTP-átirányítás során a felhasználó internetszolgáltatója (azaz az őt kiszolgáló optimális CDN) jó biztonsággal megállapítható a forrás IP-cím és egy GeoIP-adatbázis (Maxmind, Neustar) segítségével, addig a DNS-alapú irányításnál ez az információ csak akkor érhető el, ha a felhasználó által igénybe vett helyi DNS-szerver (resolver) támogatja az EDNS *client subnet extension*-t [3], mely lehetőséget biztosít a felhasználó IP-címének a CDN irányítója számára történő továbbítására. Természetesen az ISP a helyi DNS-szervereiben ezt egyszerűen aktiválhatja, illetve a Google által szabadon elérhető DNS-szerverek ezt automatikusan támogatják [4]. Mindezek ellenére számolni kell egy alacsony hibafaktorral, hiszen biztosan lesznek olyan felhasználók, akik hátrányukra változtatják meg a DNS-beállításaikat, így a számukra ideális CDN nem meghatározható.

Harmadszor pedig elengedhetetlen, hogy több CDN között dinamikusan irányított felhasználók esetén valós idejű naplózási és analitikai rendszert építsünk. Ehhez



1. ábra
A Telekom CDN-portfóliója

nem támaszkodhatunk az egyes CDN-ek saját megoldásaira, mert egy átfogó, korrelált képet kell alkotnunk az összes felhasználó streaming minőségéről, ezért egy saját rendszer létrehozása mellett döntöttünk. Ehhez jó megoldást biztosít az *ELK Stack/Search Guard,* melyek segítségével képesek vagyunk 60 másodperces késleltetéssel másodpercenként több millió naplóbejegyzés feldolgozására, keresésére, illetve elemzések és grafikonok biztosítására.

3. Fejlesztés

Egy ISP számára a saját hálózatán belüli CDN-szolgáltatás létesítéséhez a lehetőségek széles tárháza áll rendelkezésre [5]. Globális platformszolgáltatóktól kezdve (Akamai, Fastly, Cloudfront), külső fél által telepített és menedzselt CDN-rendszereken át (Qwilt, open caching), CDN-szoftverbeszállítókon keresztül (Synamedia), egészen a nyílt forráskódú rendszerekig (Apache Traffic Control). Természetesen egy teljesen új fejlesztésnek a lehetősége is nyitva áll. A megfelelő megoldás kiválasztása összetett kérdés, azonban a Telekom szempontjait figyelembe véve az *1. táblázat* szerinti értékelést vettük alapul.

Az *unicast TV* alapvető igénye az extrém skálázhatóság több tíz terabites tartományban. Míg ez a szolgáltatási kategóriában probléma nélkül elérhető peering kapcsolatokon vagy közvetlenül a szolgáltató hálózatába telepített infrastruktúrával, addig a menedzselt, beszállítói és nyílt forráskódú CDN-megoldásoknak tipikus problémája a központi, monolitikus kérésirányítás és -felügyelet, mely nehezen alkalmazkodik több milliós felhasználószámhoz. Ez számunkra határozottan az új fejlesztés irányába mutatott.

Egy CDN-szolgáltatás versenyképes alkalomszerű multimédia-továbbítás esetén, mint pl. labdarúgó VB vagy az olimpiai játékok, ahol nagy sávszélességigény csak rövid ideig jelentkezik, azonban a televíziózás során minden este 8 órakor és minden hétvégén megjelenik a sávszélességcsúcs, amely költséghatékonyan csak saját vagy osztott infrastruktúrával biztosítható. Így a szolgáltatói CDN-megoldásokat a saját hálózatban kizártuk.

Fontos szempont továbbá a rendszer flexibilitása, igény szerinti módosítása. Ez külső beszállítók esetében nem kívánt függőséget jelent. Mindig is alapvető és kritikus fontosságúnak tartottuk, hogy a megfelelő kompetencia és továbbfejlesztési lehetőség egy ISP oldalán is rendelkezésre álljon, ezért ebben a kérdésben is a beszállítói megoldásoktól távolodva, a nyílt forráskódú lehetőségek irányba billent a mérleg nyelve.

Egy CDN-rendszer semmiből történő kifejlesztése túlzott befektetést igényelne, ezért a számunkra optimális megoldást végül a nyílt forráskódú rendszerek és a teljes fejlesztés között találtuk meg: nem érdemes mindent nulláról kezdeni, de a nyílt forrás kódú CDN-projektek sem biztosítják a megfelelő fokú szabadságot, ezért köztes megoldásként a Telekom más szolgáltatásaiban már jelen lévő, nyílt forráskódú komponensekre támaszkodtunk. Saját fejlesztéssel ezeket egy CDN-rendszerbe szerveztük, így az erőforrásigényeket minimálisan tartottuk, mindemellett a flexibilitást maximálisan megőriztük (ld. következő szakaszt), nagyfokú automatizáció alkalmazásával pedig a menedzsmentkomponensek fejlesztését elkerültük (5. szakasz).

4. Architektúra

Az architektúrát alapvetően meghatározó szempontok az extrém skálázhatóság, illetve a magas rendelkezésreállás és hibatűrés. Ezek szerepét legegyszerűbben a tipikus CDN-ek problémáin keresztül tudjuk bemutatni. Ahogy azt a *2. ábra bal oldala* mutatja, egy CDN-rendszer fő komponensei közé tartozik a

• *request router,* mely a teljes CDN-infrastruktúra állapotát felügyeli, és DNS A/AAAA-választ adó bejegyzések (1), vagy HTTP 302/307 átirányítás (2) segítségével a felhasználók kérését egy optimális *edge cache*-hez irányítja (3). A cache találati arány növelése érdekében az egy helyre telepített cache-ek tárterületét *consistent hashing* segítségével [6] egy virtuális tárterületté szervezi.

• *Edge cache,* mely magát a tartalom közbülső tárolását és kiszolgálást biztosítja több hálózati helyszínen (point of presence, PoP), ideális esetben közel a felhasználókhoz.

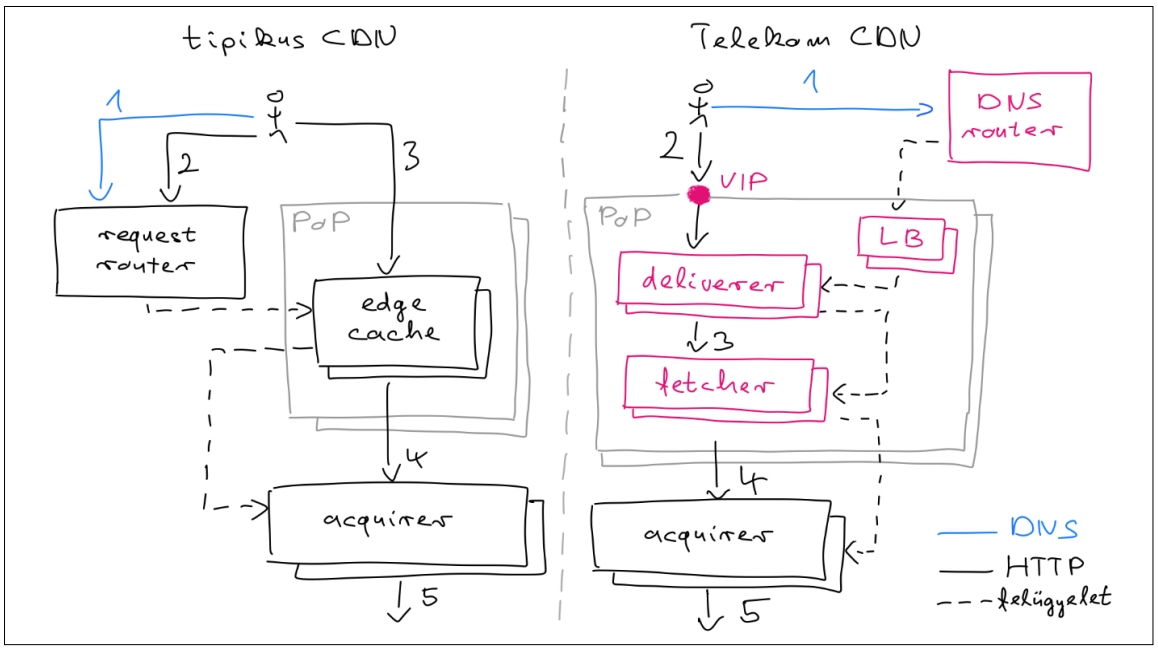
• *Acquirer,* mely feladata egy közbenső tárolási réteg kialakítása az *edge cache*-ek és a forrásszerver *(origin)* között (4, 5), utóbbi további tehermentesítése érdekében.

A legfőbb problémát a request router összetett és központi szerepköre, mely esetleges meghibásodása a teljes CDN-rendszerre hatást gyakorol, illetve ezen szerepkör mind a felhasználók, mind az edge cache-ek számával történő arányos skálázódása jelenti. Ahhoz, hogy a CDN-rendszer extrém és megbízható módon bővíthető legyen, a request router drasztikus egyszerűsítésére volt szükség.

A *2. ábra jobb oldalának* megfelelően, önfelügyelő és hibatűrő PoP-ok létrehozásával a központi felügyeletet elosztottuk a rendszeren. Az egy PoP-hoz tartozó komponensek önmagukban képesek a redundancia és hibajavítás biztosítására minden külső funkció nélkül, en-

1. táblázat
CDN létesítési alternatívák

	szolgáltatás	menedzselt	beszállítói	nyílt forráskódú	fejlesztés
skálázhatóság	+++	++	++	++	+++
költségek	+++	++	++	+	+
flexibilitás	+	+	++	+++	+++
tudás	+	+	++	++	+++
erőforrásigény	+	+	++	++	+++
üzemeltetés	+++	+++	++	+	



2. ábra
Architektúra

nek köszönhetően a request router már nem egyedileg felügyeli az egyes edge cache-eket, elegendő számára csupán a PoP-okra, mint logikai egységekre tekinteni. Így a request router méretezése függetlenedik a CDN méretétől.

Másfelől az edge cache-ek eléréséhez *direct routing* alapú terhelés elosztást [7] alkalmaztunk, mely lehetővé teszi az egyetlen virtuális IP-címről történő szolgáltatásnyújtást, függetlenül a PoP-ba telepített szerverek számától. Ez mindamellett, hogy nagyfokú skálázhatóságot biztosít, a request router szerepkörét tovább csökkenti, így a request router-nek már csak egy alapvető DNS-irányító feladatkört kell ellátnia, mely során a felhasználók kéréseihez (1) egyszerűen a közelükben lévő PoP-ok virtuális IP-címéhez rendeli. A HTTP-átirányítás szerepét veszti. Köszönhetően a DNS-infrastruktúrában alkalmazott köztes tárolási funkciónak, a request router (immár DNS-router), a felhasználók számával sem skálázódik.

Az előbbiek lehetővé teszik a több ezer edge cache, vagy akár a több száz terrabites tartományba való skálázhatóságot, azonban így egy fontos előny is elveszik, mégpedig a consistent hashing által biztosított virtuális tárhely lehetősége. A DNS-irányítás során a felhasználó által elérni kívánt URL, azaz tartalom nem ismert, ezért a DNS-router-nek nincs lehetősége az egyes tartalmak edge cache-ekhez történő rendelésére. Ez egy nélkülözhetetlen funkció a CDN költségeinek alacsonyan tartásához, ezért ennek biztosítása érdekében a Telekom architektúrájában az edge cache-ek szerepét, hasonlóan a Fastly által szervezett PoP-okhoz [8], ketté kellett választani.

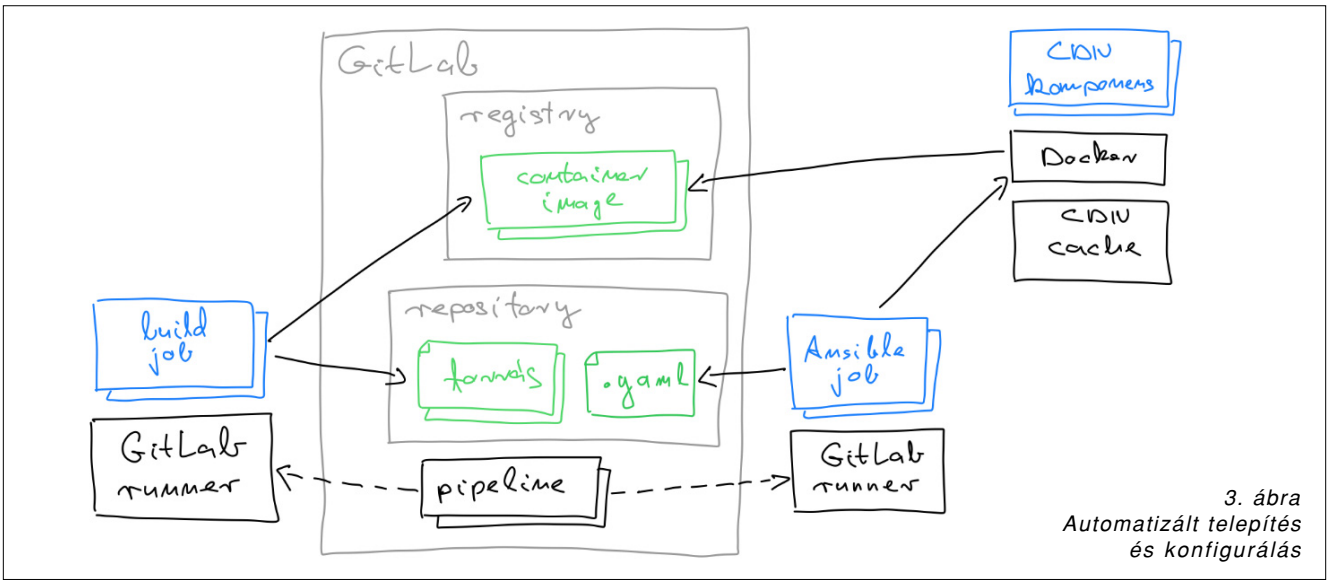
A felhasználókat közvetlenül a *deliverer*-ek szolgálják ki (2), melyek a top tartalmak közül is a top-ot tárolják. Mivel a kéréseket véletlenszerűen rendeljük az egyes deliverer-ekhez, ezért ezen tartalmak többszörösen, minden deliverer-en tárolásra kerülnek, így az erre allokált tárhely alacsonyan van meghatározva. Viszont a deliverer-ek már értelmezik a HTTP-kéréseket, így számukra lehetséges a *consistent hashing* alkalmazása a *fetcher*-ek irányába (3), ahol az egyedi tárhelyek már egy közös virtuális tárhellyé vannak szervezve a magas találati arány fenntartása érdekében. Természetesen e két funkciókhoz nem szükséges külön szervereket fenntartani, egy komponensben megvalósítva, logikailag szétválaszthatóak.

A kompromisszum, ami ezzel az architektúrával jár, az a kérések egy részének kettős kezelése, ami azonban megfelelő *deliverer/fetcher* méretezéssel az IPTV-szolgáltatások esetén elfogadható szinten tartható. A fetcher-ek szerepe és feladatköre nem változott, továbbra is a forrásszerverek tehermentesítését végzik (4, 5).

A rendszer felépítéséhez a következő komponenseket választottuk:

- Operációs rendszer: Ubuntu Linux.
- DNS-router:
  - dnsdist: DNS-terheléselosztás, túlterheléses támadás elleni védelem,
  - PowerDNS: hiteles névszerver,
  - FastAPI: RESt APi szerver a kérésirányítási logika megvalósításához.
- Deliverer, fetcher, acquirer:
  - Varnish Cache: nyílt forráskódú, közbenső tároló (reverse caching proxy),
  - HAProxy: TLS-végződtetés vagy kezdeményezés,
  - keepalived: PoP-terheléselosztás,
  - Python-alapú erőforrás felügyelet.

Az üzemeltetés további egyszerűsítése érdekében az összes alkalmazás Docker alapon, konténerekben fut. Célunk nem egy edge-felhő létrehozása volt, hiszen ez jóval túlmutatna a CDN-rendszer keretein, hanem egy HW/OS-absztrakció megvalósítása. Ez lehetővé teszi a fizikai komponensektől és a gazda operációs rendszertől független patch-menedzsmentet, és a függőségek konténeren belüli egyedi és flexibilis kezelését, mely a rendszer fejlesztését, tesztelését és automatizációját nagyban megkönnyíti.



3. ábra
Automatizált telepítés és konfigurálás

5. Automatizáció

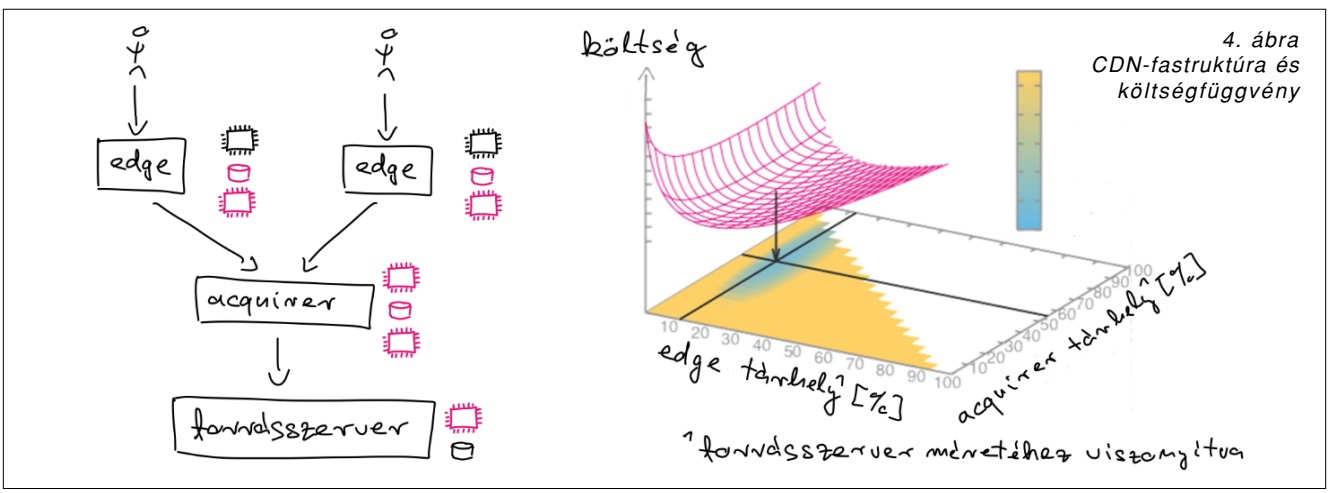
Az *unicast streaming* által igényelt jelentős kapacitás biztosításához nemcsak egy erősen skálázható és robusztus rendszerre van szükség, hanem sok szerverre is. A Telekom CDN-rendszerét több tucat telephelyen több száz szerver alkotja. Ahhoz, hogy egy ilyen méretű rendszer könnyen üzemeltethető, illetve a fejlesztési erőforrásigény alacsonyan tartható maradjon, egy új szemléletre volt szükség a rendszermenedzsment területén: teljes telepítési és konfigurálási automatizációra, illetve a kezelői felület hiányára. Tekintsük át ezeket részletesebben.

Ha egy CDN-rendszer már meglévő szoftverkomponensek integrációjával épül, akkor a legnagyobb fejlesztési erőforrást ezen komponensek távoli konfigurálása és adminisztrációs felülete igényli. Ha jól megnézzük ezeket a feladatköröket, észrevehetjük, hogy az adminisztrációs felületnek a feladata nem más, mint a rendszer tulajdonságainak egy absztrakt leírása, a távoli konfiguráció pedig ezen leírás átfordítása a komponensek egyedi vezérlésére. Erre már léteznek szabadon elérhető, kiváló megoldások, mint Ansible, Puppet, Chef, Salt stack.

A Telekom CDN tervezése során a választásunk a *yaml* formátumra esett, mely könnyen érthető és kényelmes absztrakciót biztosít. Az automatizációra számításba vettük a *Puppet* környezetet, azonban ennek nem túl elterjedt programozási környezete miatt végül az *Ansible* segítségével valósítottuk meg. Ez kiválóan integrálható a Telekomnál rendszeresített *GitLab*-környezetbe és a *Molecule* [9] keretrendszer segítségével teszt automatizációt is lehetővé tesz *(3. ábra)*.

6. Méretezés és rugalmas skálázhatóság

CDN-rendszer tervezése során alapvető kérdés a méretezés és költségoptimalizálás. Dedikált hardver-infrastruktúrát feltételezve ezt csúcsterhelésre kell számolni a cache-ek erőforrásigényei alapján processzor, tárhely (ideértve a memóriát is) és hálózati interfészek mentén. Alacsony igényei miatt a CDN-menedzsment és kérésirányítás, illetve a belső hálózaton olcsón és nagy sávszélességen elérhető hálózati kapcsolatok miatt a hálózati interfészek is elhanyagolhatóak, így a méretezést alapvetően 2 ortogonális dimenzión kell elvégezni a cache-ek processzor- és tárhelyigényei alapján.



4. ábra
CDN-fastruktúra és költségfüggvény

Figyeljük meg a CDN fastruktúrájú felépítését a *4. ábra bal oldalán!* Könnyen megérthető, hogy az *edge processzor* erőforrásigénye egyenes arányban áll az éppen szolgáltatást igénybe vevő felhasználók számával, hiszen egy streaming kapcsolat elsősorban a processzort terheli (TLS-kézfogás, titkosítás, TCP-stack, adatok mozgatása a memóriából/térhelyről a hálókártya irányába). Csúcsterhelésre ez egy jól becsülhető, fix érték. Az *edge cache* tárhely méretezése viszont már egy függő változó, mely elsősorban a cache találati arányt, ezen keresztül pedig a tartalom beszerzéséhez szükséges processzor igényt és az *acquirer*-en megjelenő kapcsolatok számát, azaz az acquirer processzorigényét határozza meg. Hasonló módon belátható, hogy az acquirer tárhely megválasztása az acquirer és a forrásszerver processzorigényét határozza meg. Magától értetődően a forrásszerver tárhelymérete a teljes tartalommal megegyező, mely szintén előre, jól becsülhető, fix érték.

Ebből következően a CDN méretezése során feladatunk ezen szabad paraméterek optimális megválasztása, mely nem triviális feladat, eléréséhez két dolog szükséges. Egyfelől meg kell határozni a cache találati arány értékét a cache tárhely méretének függvényében. Ez a tartalomfogyasztási szokásoktól és a cache-ben implementált *eviction*-algoritmustól függ, azonban az esetek nagy többségében ez nem írható le zárt algebrai formában. Ezen dilemma feloldása érdekében a CDN méretezése során egy parametrizálható cache-modellt alkottunk Python-alapokon, mely alkalmas a tartalomfogyasztási statisztikák alapján a bejövő forgalom becslésére. Ezen egyedi cache-modellek hálózatba szervezésével a teljes CDN-rendszer szimulálhatóvá válik, így már meg lehetett becsülni a szükséges erőforrásigényeket, azaz a CDN költségét egy választott szabad paramétercsoportra.

Másfelől a CDN költségének minimalizálásához optimálisan kell megválasztani ezen paramétereket. Ezt egyszerűen egy Monte-Carlo-szimulációval oldottuk meg, mely segítségével minimalizálható a CDN költségfüggvénye *(lásd a 4. ábra jobb oldalán)*. Erről leolvasható a teljes költség különböző *edge* és *acquirer* tárhely tükrében. Az ábra által szemléltetett példa-CDN-konfigurációról jól megállapítható, hogy a CDN költségének minimalizálása érdekében az edge tárhely méretét 12%-ban, az acquirer-t pedig 54%-ban kell választani, valamit a költség érzékenyebb az edge tárhely változására, mint az acquirer-ére.

A fenti gondolatmenetet folytatva érdekes kérdés e költségmodell időbeli alakulása. A fent vázolt megoldás egy pillanatnyi optimumot ad, azonban a felhasználók tartalomfogyasztása, így a CDN-rendszer erőforrásigénye is időfüggő *(5. ábra)*. Az eddigi méretezés jól alkalmazható dedikált infrastruktúra esetén, azonban, ha egy edge felhő által adott lehetőségeket szeretnénk kihasználni, vagy a CDN-rendszer energiafogyasztását is számításba szeretnénk venni, akkor további költségcsökkentést érhetünk el az egy időpontban szükséges erőforrások igény szerinti felszabadításával vagy lekötésével. Ennek modellezésére további tényezők figyelembevételére van szükség. A tárhely felszabadítása során a cache találati arány csökkenni fog, ami megnöveli a relatív tartalombeszerzési sávszélességet, azaz az alsó rétegek terhelése nő, illetve tárhelyfoglalás esetén az újonnan kapott tárhely „üres”, ennek tartalommal való feltöltése ideiglenesen szintén rontja a találati arányt. Ezek függvényében a fent bemutatott cache modell átdolgozására van szükség, melynek képesnek kell lennie a felhasználók tartalomfogyasztási szokásainak előrejelzésére.

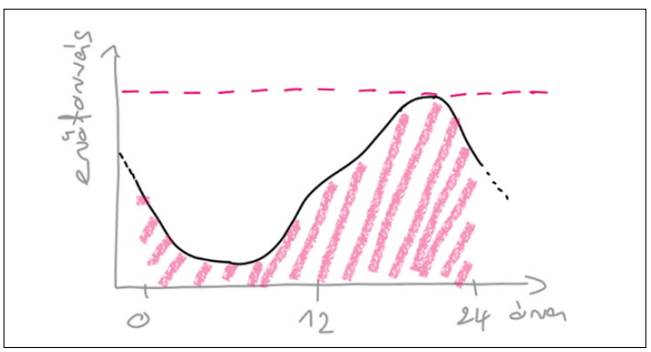
7. Továbbfejlesztés

A nyílt forráskódú komponensekre épülő rendszereknek folyamatosan alkalmazkodniuk kell a változó hardver- és szoftverkörnyezethez. Ez a Telekom CDN-rendszerével is így van, a jelenlegi funkcionalitás az elmúlt öt év során alakult ki az újonnan megnyíló (és rég elfeledett, de újra felfedezett) lehetőségek mentén. A cikk írása időpontjában két fontos lehetőséget említünk meg, melyek kiaknázását fontosnak tartjuk.

Elsőnek a *Varnish Cache* legfőbb hiányosságát, a tartós (nem felejtő) tárhely használatának hiányát említem. A Varnish Cache elsősorban egy kiváló RAM-cache, azonban a memória túl drága nagy méretű videótár hatékony tárolásához. Szerencsére, ebben a témakörben előre tudtunk lépni, és jelenleg tesztelés alatt áll a nemrég nyilvánosságra hozott „buddy” és „fellow” *storage plugin* [10], mely az *io_uring* könyvtár [11] alkalmazásával ezt a problémát orvosolja.

Másodiknak a TLS implementálásának hiányát emeljük ki. Ez, ahogy az architektúrából is látszik, könnyen áthidalható egy TLS-terminátor segítségével, azonban ebben az esetben elveszik az IP-réteg közvetlen és dinamikus kezelésének a lehetősége, azaz az edge cache nem látja közvetlenül a TCP-kapcsolatokat és ezek paramétereit, statisztikáit. Valamint nem tudja közvetlenül az L3/L4-es protokollokat vezérelni, úgymint dinamikus szolgáltatásosztályhoz (ToS) rendelést, vagy a TCP *congestion control* meghatározását. Célul tűztük ki, hogy a nyílt forráskódú Varnish Cache is képes legyen a TLS natív támogatására és terminálisára, ami egyben megnyitja a lehetőséget a kernel TLS használatához. Ezzel a technológiával reményeink szerint a processzor szignifikánsan tehermentesíthető [12].

5. ábra CDN időbeli erőforrásigénye



Mindezek mellett egy ellenpéldát is szeretnénk felhozni. Az elmúlt évek során a hardverplatform-kapacitás lendületes fejlődésének lehettünk tanúi: Intel Scalable Gen 4, PCIe5, 400GE hálózati kártyák, a több mint 100 magos processzorok is könnyen elérhetővé váltak. Sorra dőlnek meg az egy szerverrel kiszolgálható sávszélesség-rekordok [13], ugyanakkor egy CDN esetében nem biztos, hogy érdemes ezt a trendet követni, és a lehető legtöbb kapacitást egy cache-ben összpontosítani. Gondoljunk csak arra, hogy egyetlen, 800 Gbit/s-ot kiszolgáló szerver kiesése, még UHD-videó sávszélességét feltételezve is, egyidejűleg 50.000 felhasználóra lenne hatással. IPTV-szolgáltatás nyújtása során maximálisan törekedni kell a felhasználói élmény megőrzésére, ezért – meglátásunk szerint – az új technológiák megjelenése inkább a kompaktabb szerverek felé fogja terelni a CDN-infrastruktúrát. Az egy szerverrel kiszolgált felhasználók száma ezzel tetőzni fog.

8. Összefoglalás

Az előző szakaszokban bemutattunk egy nyílt forráskódú komponensekre épülő, skálázható, flexibilis és könnyen üzemeltethető tartalomelosztó hálózatot, mely moduláris felépítése és nagyfokú automatizációja révén költséghatékony megoldást biztosít streamingszolgáltatások nyújtásához. Azonban cikkünk igazi lényege nem ez, hiszen a felhasznált komponensek szabadon elérhetők, az alkalmazott technológiák zöme pedig több mint húszéves múltra tekint vissza, – azt is mondhatnánk; nincs itt semmi különös látnivaló!

Amit át szeretnénk adni az olvasónak, az egyfelől a nyílt forráskódú szoftverekben rejlő potenciál felismerése. Meglátásunk szerint ma már elértük azt a szintet, hogy egy közösség által készített és támogatott szoftverekkel igenis lehet nagy megbízhatóságú rendszereket építeni és akár kritikus szolgáltatásokat biztosítani, ha nyitottak vagyunk e közösségekben önzetlenül részt venni és a közös célt sajátunknak tekinteni.

Másfelől, – de nem utolsósorban –, a telko-világban a kreativitás fontosságát és szerepét szeretnénk kiemelni. Sajnos, az eddigiekben sok olyan helyzettel szembesültünk, amikor a szaktudás és kockázatvállalás hiánya miatt egy adott beszállítóra túlzott módon hagyatkoznunk kellett, ami nehezen feloldható függőséget és további tudásvesztést okozott. El kell kerüljük az ilyen helyzeteket, lehetőséget és teret kell engednünk a tanulásra, valamint vállalnunk kell néhány hibánk következményét, mert csak így fejlődhetünk és teremthetünk olyan megoldásokat, melyek maradéktalanul szolgálják saját céljainkat. Egy beszállító valamiyen szempontból mindig a saját érdekét fogja szem előtt tartani...

„Mérnökök vagyunk,
merjünk szabadon alkotni és építeni!”

Hivatkozások

[1] K. Kerpez, D. Waring, G. Lapiotis, J.B. Lyles and R. Vaidyanathan, “IPTV service assurance,” IEEE Communications Magazine, Bd.44, Nr.9, pp.166–172, 2006.

[2] S. Michael, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld and P. Tran-Gia, “A survey on quality of experience of HTTP adaptive streaming,” IEEE Communications Surveys & Tutorials, Bd.17, Nr.1, 2014.

[3] C. Contavalli, W.v.D. Gaast, D.C. Lawrence, W. Kumari, RFC 7871 – Client Subnet in DNS Queries, 2016.

[4] Google, “EDNS Client Subnet (ECS) Guidelines,” [Online]. *https://developers.google.com/speed/public-dns/docs/ecs*

[5] D. Rayburn, “Updated List Of CDN Vendors and History of All CDNs To Date,” 16.01.2023. [Online]. *https://www.streamingmediablog.com/2023/01/cdn-list.html*

[6] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine and D. Lewin, “Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web,” 1997.

[7] W. Zhang, “Linux virtual server for scalable network services,” in Ottawa Linux Symposium, 2000.

[8] Fastly, “Clustering,” [Online]. *https://developer.fastly.com/learning/vcl/clustering*

[9] Red Hat, Inc., “Ansible Molecule,” [Online]. *https://molecule.readthedocs.io/en/latest/*

[10] N. Goroll, “Two new storage engines for Varnish-Cache,” 06.02.2023. [Online]. *https://varnish-cache.org/lists/pipermail/varnish-announce/2023-February/000757.html*

[11] “Efficient IO with io_uring,” [Online]. *https://kernel.dk/io_uring.pdf*

[12] S. Shwartsman, T. Jursonovics, “Kernel TLS and TLS hardware offload,” [Performance]. EuroBSDCon, 2019.

[13] D. Gallatin, T. Jursonovics, “The ‘other’ FreeBSD optimizations used by Netflix to serve video at 800Gb/s from a single server,” [Performance]. EuroBSDCon, 2022.

A szerzőről

JURSONOVICS TAMÁS a Budapesti Műszaki Egyetem Villamosmérnöki és Informatikai Karán szerezte diplomáját (2003) és védte meg doktori disszertációját (2014) IPTV-rendszerek témakörben, illetve a Hochschule Darmstadt – University of Applied Sciences-en szerzett MBA-fokozatot (2015). 2003-ban csatlakozott a Deutsche Telekom csoporthoz, kezdetben a mobil TV-rendszer kiépítésén dolgozott a Westel900/T-Mobile Magyarországnál, majd több, a Magyar Telekomnál vezetett nemzetközi IPTV integrációs projekt után 2011-től Németországban, a Deutsche Telekom-nál fejleszti a Magenta TV szolgáltatást támogató CDN-rendszert. Számos médiatartalom-átvitellel kapcsolatos publikáció szerzője, az ELTE Adattudományi és Adattechnológiai Tanszékével közös CDN-optimalizációs kutatások aktív résztvevője, az MTA köztestületi tagja, elkötelezett híve és szószólója a mérnöki tudás és tapasztalat népszerűsítésének.