Önálló laboratórium beszámoló

Távközlési és Médiainformatikai Tanszék

készítette: **Schweitzer András Attila**
schweitzeraa16@gmail.com
neptun-kód: **TLEIB5**
ágazat: **Intelligens hálózatok**
konzulens: **Németh Felicián**
nemethf@tmit.bme.hu
konzulens: **Lévai Tamás**
levait@tmit.bme.hu

Téma címe: Többutas adatátvitel Media over Quic rendszerben (Media over Multipath QUIC)

Feladat:
A Media over QUIC (MoQ) egy új, még fejlesztés alatt álló protokollcsalád, amelyet az IETF szabványosít. Egyik megvalósítása a LibQuicR, amely a feladat alapjául szolgál. A feladat maga pedig nem más, mint a LibQuicR könyvtár módosítása oly módon, hogy az képes legyen kihasználni a multipath (többutas) adatátviteli képességeket, amelyeket a PicoQUIC nevű, QUIC protokollt megvalósító könyvtár biztosít. A munka során során a LibQuicR transzport rétegét úgy kell átalakítani, hogy az egy kapcsolaton belül több hálózati útvonal létesítésére és használatára legyen alkalmas a PicoQUIC multipath implementációján keresztül.

A módosításokat követően a rendszer működését egy demonstrációval igazoljuk: egy virtuális tesztkörnyezetben szimulált, több hálózati interfésszel rendelkező kliensek segítségével történik az adatátvitel, amely során aktívan használjuk a multipath képességeket. A demonstráció során egy olyan forgatókönyvet vizsgálunk, ahol az egyik aktív útvonalat szándékosan megszakítjuk. A cél annak bemutatása, hogy a LibQuicR – az új multipath támogatással – beavatkozás nélkül képes fenntartani az adatfolyam továbbítását, minimális késleltetéssel és megszakítással.

Tanév: 2024/25. tanév, II. félév

1. A laboratóriumi munka környezetének ismertetése, a munka előzményei és kiindulási állapota

1.1. Bevezető

Napjainkban az internetes adatforgalom túlnyomó részét különféle médiatartalmak – elsősorban videók és élő közvetítések – teszik ki. Egyes kutatások szerint szerint az internetes forgalom 65%-a ilyen típusú tartalom lehet [1]. Ezek hatékony és megbízható továbbítása azonban komoly technológiai kihívást jelent, különösen akkor, ha a felhasználói élményt is figyelembe vesszük: az alacsony késleltetés, a megszakításmentes lejátszás és a biztonságos adatátvitel alapvető elvárások.

E kihívásokra jött létre válaszként a QUIC protokoll, amely UDP-alapú, alacsony késleltetésű, titkosított és megbízható adatátvitelt tesz lehetővé. Erre épülve fejlődik jelenleg a Media over QUIC (MoQ) szabvány, amely kifejezetten médiatartalmak hatékony továbbítását célozza a QUIC képességeit kihasználva.

Ezzel párhuzamosan a felhasználói eszközök – különösen a mobiltelefonok és laptopok – egyre gyakrabban rendelkeznek több hálózati interfésszel (például Wi-Fi és 5G). Mégis gyakran tapasztalható kapcsolatmegszakadás, hosszú töltési idők vagy éppenséggel elérhetetlenné váló szolgáltatások, ha az egyik kapcsolat megszakad vagy instabillá válik. A többutas adatátvitel lehetőséget kínál ezen problémák áthidalására azáltal, hogy párhuzamosan több hálózati útvonalat használ az adatok továbbítására, növelve ezzel a robusztusságot és az elérhető sávszélességet.

A jelen munka célja ezen két technológia – a médiatartalmakra optimalizált QUIC-alapú továbbítás és a multipath kommunikáció – egyesítése. A cél egy olyan rendszer bemutatása, amely képes több útvonalat kihasználva, élő adatfolyamot hatékonyan és megbízhatóan továbbítani. Ez nemcsak a hálózati erőforrások jobb kihasználását segíti elő, hanem hozzájárul a végfelhasználói élmény javításához is.

1.2. Elméleti összefoglaló

1.2.1. A QUIC protokoll

Az internetes multimédiaátvitel technológiai fejlődése során egyre nagyobb igény mutatkozik olyan protokollokra, amelyek képesek rugalmasan és hatékonyan kezelni a valós idejű adatfolyamokat, még változékony és megbízhatatlan hálózati körülmények között is. [2] A QUIC protokoll – mint UDP-alapú, titkosított és kapcsolatorientált transzportprotokoll – alapjaiban újraértelmezi a hálózati kommunikáció lehetőségeit, különösen olyan kiterjesztésekkel, mint a *multipath* adatátvitel [3]. A multipath képesség lehetővé teszi több párhuzamos hálózati útvonal (**path**) egyidejű vagy váltott használatát egyetlen logikai kapcsolat keretén belül, ami különösen hasznos mobil eszközök, redundáns hálózatok vagy edge hálózati környezetek esetén.

A QUIC protokoll és a HTTP/3 működésének vizsgálatára fejlesztették ki hozzá a qlog formátumot, amelynek célja, hogy strukturált és gépileg olvasható formátumban rögzítse a QUIC kapcsolatok eseményeit és metrikáit. Ez tulajodnképpen egy JSON-t generál a kapcsolat bármely végpontjában ami tartalmaz minden olyan adatot amely a kapcsolat során generálódik a QUIC rétegben, beleértve a csomagok küldését és fogadását, a kapcsolat állapotát, a késleltetéseket és a hibákat is.

1.2.2. QUIC multipath alapjai

A multipath koncepció technikai alapja, hogy a QUIC kapcsolat során nemcsak egy, hanem több **path** hozható létre, amelyeket a transzport réteg párhuzamosan vagy dinamikusan képes használni. Egy *path* definíció szerint egy adott forrás-cél IP-cím és port kombináció, azaz különböző fizikai vagy logikai hálózati kapcsolatok reprezentálása. A projektben használt PicoQUIC könyvtár egy QUIC megvalósítás, amelynek fejlődő multipath támogatása lehetővé teszi, hogy egy kapcsolat során több ilyen path aktív legyen, és az alkalmazás (quic-et megvalósító rétegében egy belső algoritmus) meghatározza, hogyan ossza meg az adatforgalmat ezek között [4].

PicoQUIC esetében a multipath kezelés alapvetően decentralizált és eseményvezérelt. A kapcsolat felépítése során egy adott interfészen elindított kapcsolat kiegészíthető további **PATH CHALLENGE / RESPONSE** (útvonal próba / útvonal válasz) üzeneteken keresztül feltérképezett útvonalakkal. Ha egy új path válik elérhetővé (például egy új IP-cím vagy interfész aktiválódik), a rendszer felismeri azt, és lehetőséget

biztosít az adatküldés ezen az útvonalon történő elindítására. A path-ek állapotát folyamatosan figyeli a protokoll (RTT, veszteség, állapotváltozás), így lehetővé válik az útvonalak közötti dinamikus váltás vagy forgalommegosztás a kapcsolatok megszakítása nélkül. Ez különösen fontos valós-idejű alkalmazásokban, mivel lehetővé teszi a megszakítás nélküli adattovábbitást, még hálózati hiba vagy mobilitás esetén is.

1.2.3. A Media over QUIC (MoQ) protokoll

A Media over QUIC (MoQ) egy új, még fejlesztés alatt álló protokollcsalád [5], amely a QUIC nyújtotta lehetőségekre építve biztosít alacsony késleltetésű, valós idejű médiatovábbítást. A MoQ rendszerében a szerepkörök három alapvető típus köré szerveződnek: a **Publisher** (szolgáltató) az, aki a médiatartalmat (például videó vagy hangfolyam) létrehozza és továbbításra bocsátja, a **Subscriber** (fogyasztó) a végponti fogyasztó, aki ezt a tartalmat fogadja és feldolgozza/lejátsza, míg a **Relay** (továbbító) köztes szereplőként funkcionál, amely a számára elérhető tartalmat hirdeti és továbbítja más résztvevők felé, jellemzően a késleltetés, terheléselosztás és elérhetőség optimalizálása érdekében. Ezek a szereplők struktúrált MoQ adatfolyamokon keresztül kommunikálnak egymással, amelyek sávokból (*track*) azon belül pedig objektumokból(*object*) épülnek fel, lehetővé téve a médiatartalom rugalmas azonosítását és replikációját.

A kidolgozás alatt álló szabvány szerint egy MoQ-alapú rendszerben a relay képes egyszerre több publisher és subscriber felé is kapcsolatot fenntartani, és akár multicast-szerű módon továbbítani a médiatartalmat. A hálózati topológia e szerepkörök között rengetegféle lehet, mivel a relayek közötti kapcsolat is támogatva van ami lehetővé teszi a tartalom replikációját és elosztását a különböző relayek között. Ezáltal a gazdag struktúrabeli opciók lehetőséget biztosítanak a tartalom dinamikus és optimalizált terjesztésére, azonban egyben igényli azt is, hogy a transzport réteg rugalmasan tudjon alkalmazkodni a változó hálózati viszonyokhoz – például egy útvonal meghibásodásához, vagy új alternatív útvonal megjelenéséhez.

1.2.4. Multipath célja a MoQ-ban

A multipath működés bevezetése ebbe az architektúrába jelentős előnyt nyújthat, különösen olyan eszközök esetén, amelyek egyszerre több hálózati interfésszel rendelkeznek (például Wi-Fi és mobil adatkapcsolat szimultán használata). A relay-ek és végpontok (subscriber, publisher) több interfésszel rendelkező környezetben történő működése során a multipath támogatás nemcsak a redundanciát növeli, hanem lehetőséget biztosít egyfajta **hálózati adaptivitásra is**, amely révén például a relay automatikusan kiválaszthatja a legjobb elérhető útvonalat egy adott irányba. Ez az architektúra ideális alapot teremt olyan rendszerek számára, ahol fontos a magas rendelkezésre állás, az alacsony késleltetés és az automatikus hibatűrés.

A MoQ protokoll jelenlegi állapota még fejlesztés alatt áll, és jelenleg egyik implementáció sem támogatja a multipath képességeket.

Bár más MoQ megvalósítások is léteznek, azért a LibQuicR könyvtárra esett a választás, mert a QUIC protokollt a PicoQUIC könyvtár segítségével implementálja, amely jelenleg is folyamatosan bővül, és multpath támogatása is követi a legújabb szabványosítási irányokat.

1.3. A munka állapota, készültségi foka a félév elején

A munka félév kezdeti állapota röviden összefoglalható azzal, hogy saját tapasztalatom nem volt sem a QUIC protokollal, sem a LibQuicR kódjával kapcsolatban, valamint a téma is új, tehát nem volt mire építeni.

2. Az elvégzett munka és az eredmények ismertetése

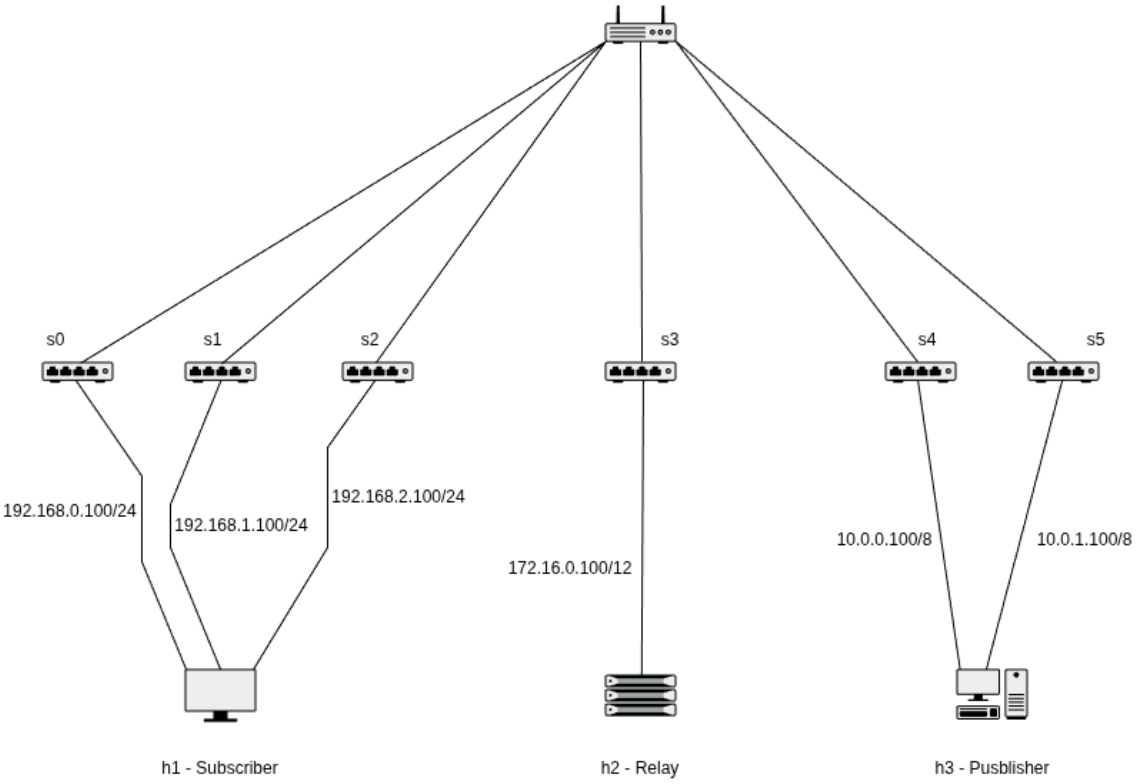
2.1. A fejlesztés és elemzés lépései egy multipath demonstrációhoz

2.1.1. Mininet szimulációs környezet

Első sorban szükséges volt egy hálózat virtualizálására alkalmas környezet, amely lehetővé teszi a különböző hálózati topológiák és viszonyok szimulálását, hiszen csak egy kontrollált környezetben lehet érdemben tesztelni a multipath működést. E célból a Mininet nevű eszközre esett a választás, mivel a használatához szükséges tudás elsajátítása nem igényelt sok időt.

A Mininet segítségével könnyen létrehozhatók virtuális gépek (továbbiakban hostok vagy h1/h2/h..), amelyek a futtató gépen létrehozott hálózati névterek, amelyeken virtuális hálózati interfészeket és azokhoz hálózati konfigurációt rendelhetünk. Ezekben a névterekben lehet futtatni a tesztelni és megfigyelni kívánt programokat. A Mininet emellett lehetővé teszi a hálózati topológia szimuláció közbeni megváltoztatását is, így kapcsolatok megszűnését is könnyen szimulálhatjuk vele. [6]

Az alábbi topológiát hoztam létre a Mininet segítségével, amellyel a multipath működését teszteltem (1. ábra):



1. ábra. A Mininetben létrehozott topológia

Ahogy említettem a h1, h2, h3 jelölik a hostokat, s0-s5 jelölik a switch-eket, és r0 jelöli a routert. h1 három interfésszel rendelkezik, egyenként csatlakozik a s0, s1 és s2 switch-ekhez, azok pedig a r0 routerhez. Hasonlóan h2 is, csak egy interfésszel, a h3 pedig kettővel csatlakozik.

A hostok és a router között azért van szükség switch-ekre, mert a kapcsolatok szimuláció közbeni megszakítása problematikus, ha a megváltoztatott kapcsolat közvetlenül a host-okhoz csatlakozik. Így az alábbi topológián egy switch és a router közötti kapcsolat megszakításával könnyen lehet szimulálni a kapcsolat megszakadását.

Címzés szempontjából a hostok eltérő alhálózatokban vannak, (a h1-nek még emellett mindhárom interfésze eltérő alhálózatban van, hogy az alapértelmezett átjáró eltérő lehessen, ezzel pontosabban szimulálva a valós környezetet),

A munkát és a tesztelést felgyorsítva mindezt egy Python szkript segítségével automatizáltam, amely a mininet API-ját használja a virtuális gépek és kapcsolatok létrehozására, konfigurálására, kezelésére, beleértve a hostokon futtatott demonstrációs programokat és a Wireshark csomagelkapást is. Tehát egy script futtatásával pillanatok alatt megfigyelhető a kívánt program működése.

2.1.2. Megismerkedás a picoQUIC könyvtárral

A féléves munka kezdetén első célom az volt, hogy alaposan megismerjem a PicoQUIC könyvtárat, mivel ez képezte a későbbi fejlesztések technikai alapját. A megismerés első lépése a **picoquicdemo** példaprogram tanulmányozása és futtatása volt, ami segített megérteni a könyvtár felépítését, működését, de különösen a multpath kezelés módjait.

Maga a pcioquicdemo felépítése elég egyszerű, olyan értelemben hogy ez egy darab C fájl, amely a picoQUIC függvényeit használja, és a szerver, valamint a kliensek funkcióit is ellátja egyaránt.

Sajnos pont emiatt az átláthatóságot nem feltétlen egyszerűsíti meg, de összességében a munka során kiderült, hogy két dolog szükséges a multipath működéshez:

- a kezdeti multipath negotiation (egyeztetés) - ami arra szolgál, hogy a két fél közötti első kapcsolat felépítésekor megegyezzenek az összes többi transzport paraméter mellett abban is, hogy mind a két fél támogatja-e a többutas kapcsolatot
- illetve az "extra" útvonalak tényleges kiépítése

Ezek körül az utóbbit megvalósító kódrészletek könnyen fellelhetőek, mivel ezen a téren a program sok fajta opciót biztosít, amelyek közül a legegyszerűbb eshetőség az alapszintű multipath, amikor több interfész használatával több útvonalat alakítunk ki, amelyeket a picoQUIC könyvtár automatikusan kezel.

Jelenlegi munka szempontjából ez az egyszerű multipath a leginkább releváns, mivel amíg az nem működik megfelelően, addig a komplexebb multipath forgatókönyvek nem is igazán tesztelhetők/használhatók. Szerencsére az ezt ellátó kódrészlet a picoquicdemo-ban jól elkülönül, így jó alapot is képzett a LibQuicR könyvtárban való implementálásához.

2.1.3. Működés elemzésének módjai

Munkám során elég korán előjött a kérdés, hogy hogyan tudom a picoQUIC működését elemezni, mivel a tényleges működést nehéz folyamatában nyomon követni, hasznos lenne picoQUIC kapcsolatot menedzselő adatfolyamot is látni.

A picoQUIC rengeteg funkciót és metrikát biztosít a működés elemzésére, első sorban a qlog fájlok generálása lenne a leginkább logikus lépés, és a protokoll alapszintű működésének megértése szempontjából valóban hasznos a qlog, de nem minden esetben elegendő, mivel a QUIC protokoll sokféle eseményt és metrikát generál, amelyek közül sok nem feltétlenül releváns a multipath működés szempontjából és maga a multipath is megnehezíti mivel az egyetlen qlog vizualizáló amit találtam az a qvis [1] volt, ami sajnos nem tudja könnyen és átláthatóan kezelni a multipath eseményeket, így ezzel nem tudtam megfelelően elemezni a kapcsolatokat.

Ezt a nehézséget kiváltandó a Wireshark program választása sokkal hasznosabbnak bizonyult, mivel a QUIC protokollt is támogatja, illetve tanulmányaim során is találkoztam már vele. Azonban ez olyan kihívást állít a qlog-gal szemben, hogy míg a qlog fájlokat a program maga generálja, tehát a titkosítást ki tudja kerülni, addig a Wireshark magukat a hálózati interfészen megjelenő csomagokat rögzíti, így a QUIC csomagok titkosítása megnehezítette a folyamatot, főleg a multipath miatt, mivel a másodlagos útvonalon küldött és fogadott csomagokat a Wireshark nem tudja hozzákötni automatikusan az első útvonal interfészén létrejött kapcsolat titkosított csomagjaihoz.

A picoQUIC beépítetten támogatja a SSL keylog fájlok generálását a "SSLKEYLOGFILE" környezeti változó beállításával, ami segítségével fel lehet oldani a quic által használt tls titkosítást, így a Wireshark képes dekódolni a csomagokat, és megjeleníteni azok tartalmát (legalábbis a kapcsolat kezeléséért felelős quic réteg szempontjából).

Ez a megoldás lehetővé tette számomra, hogy a Wireshark segítségével generált pcapng fájlokat átkonvertáljam dekódolt formátumra az alábbi paranccsal:

[1] qvis: webes alkalmazás, amely a qlog fájlok vizualizálására szolgál, és sajnos a multipath használata könnyen eltorzítja a két oldal kapcsolatának a szinkronizációját.

```
editcap --inject-secrets tls,<keylog_file> <input_pcap_file>
  <output_pcap_file>
```

Ez a parancs a Wireshark által generált TLS titkosítást használó QUIC csomagokat tartalmazó pcapng fájlokat dekódolja, a keylog fájl segítségével, így azok bármilyen kontextusban, a keylog fájl nélkül megtekinthetőek és elemezhetőek egy Wireshark programban.

2.1.4. libquicr felépítése és átalakítása

2.1.5. Elvégzett munka eredményei

<Én magam (nem a társam) a félév során következőket olvastam el / programoztam / készítettem el / teszteltem / dokumentáltam / néztem át / tanultam meg, stb. Tételes leírása és felsorolása mindannak, ami a félév során történt, alátámasztandó azon állításom a konzulens/tárgyfelelős felé, hogy összességében mindent beleértve tényleg dolgoztam a TVSZ szerint kreditenként 30 órát, azaz a heti 2 kontakt órás tárgy esetében min. $2,5*30 = 75$ munkaórát, illetve a heti 6 kontakt órás tárgy esetében min. $8*30 = 240$ munkaórát...>

Ebben a részben a hallgató az általa elvégzett munkát mutatja be. Hangsúlyosan a saját munka bemutatása a cél, hiszen a hallgató ezzel igazolja a témavezető és a tárgyfelelős irányába, hogy – folyamatosan fejlődve és egyre több és jobb munkát végezve – a szakdolgozatát/diplomadolgozatát képes lesz megírni. A beszámoló nem munkanapló, nem arra vagyunk kíváncsiak, hogy mit mikor csinált a hallgató és mennyi időt töltött vele, hanem egy eredmény-centrikus beszámolót szeretnénk olvasni. De itt is fontos tudni, hogy megosztott feladat esetén ki-mit csinált, mekkora részt vállalt.

Az egész beszámoló elkészítésénél törekedni kell a magyar nyelv szabályainak követésére és a műszaki dokumentáció/tudományos közlemény írásával kapcsolatosan kialakult közmegegyezés szerinti formai követelmények betartására. (Tehát nem kell többes számként hivatkozni saját magunkra, kerülni kell a furcsa megfogalmazást, passzív és egyéb kifacsart mondatszerkezeteket. Az egy szót határozatlan névelőként történő használatakor ne írjuk ki számként.)

A beszámoló természetesen nem csak szöveget tartalmazhat, hanem képleteket, táblázatokat, ábrákat és még sok minden mást. Ezek kapcsán az alábbi elvek irányadók:

- Az ábráknak, képeknek és táblázatoknak mindig van számuk és címük. (A cím nem ennyi: „1. ábra”, hanem azt írd le, ami látható rajta.)
- Az ábrákra, a képekre és a táblázatokra a szövegben hivatkozni kell, és a szövegben elemezni kell azokat. Például a 2. ábrán látszik, hogy a vizsgált félévben még két napos csúszással is lehetett jeles érdemjegyet szerezni a tárgyból, de a pontosság még nem garancia a jó jegyre: öten nem kaptak jelest, noha nem késtek a leadással.
- Az ábrák, képek és táblázatok mérete a szükségesnek megfelelő legyen: elég nagy ahhoz, hogy kinyomtatva is olvasható és értelmezhető legyen, de nem nagyobb annál, mint amit szerepe indokol.
- A grafikonoknak a tengelyeken legyenek feliratai és ha releváns, a mértékegység is.
- A képletek esetében nem minden képletre történik hivatkozás, de ahol igen, ott a képletet a műszaki irodalomban jellemző módon a sor végére tett kerek zárójelben lévő számmal jelöljük meg. A képleteket ne képként illeszd be a szövegbe.
- Kódrészleteket, ha nem relevánsak, ne illeszd be képként, főleg ne rossz minőségben. Nyugodtan teheted függelékbe és hivatkozd be a szövegben, mint a képeket, például: Az 1. számú függelékben található az adatbeolvasó kód, melyet C++ nyelven készítettem el.

Az írásbeli beszámolót a témavezető és a tárgyfelelős is értékeli. A tárgyfelelősi értékelés szempontjai az alábbiak:

1. Megfelel-e az elvégzett munka a félév elején kiadott feladatnak?
2. Megfele-e a beszámoló a formai követelményeknek? Ezen belül:
    - a. Megfelelő-e az elméleti bevezető és az irodalomjegyzék?

Az írásbeli beszámoló beadásának napja a szóbeli beszámolóhoz képest (munkanapban)	A „b” faktor értéke
-4. munkanap	0.04
-3. munkanap	0.09
-2. munkanap	0.20
-1. munkanap	0.30

1. táblázat. Az írásbeli beszámoló késedelmes beadásával kapcsolatos hanyagsági faktor értéke

- b. Egyértelmű-e, hogy mi volt a hallgató saját munkája?
- c. Megfelelő-e a dokumentum technikai színvonala?

Ezen kívül a tárgyfelelős veszi figyelembe az értékelés során kialakult félévi jegyre vonatkoztatva az ún. „hanyagsági faktor” értékét, amelyet (1) szerint állapítunk meg:

$$F_{hany} = 1 - a - b \tag{1}$$

2. ábra. Hallgatók érdemjegyeinek eloszlása az írásbeli beszámoló késése függvényében

Az (1)-ben szereplő a szám a munkaterv beadásában történt késedelemre, míg a b szám az írásbeli beszámoló beadásában történt késedelemre vonatkozik. Utóbbi értékeiről az 1. táblázat tájékoztat.

A beszámoló értékeléséről részletesebben írunk [10]-ban.

A beszámolóban bizonyára szerepelni fognak rövidítések. Ezeket a rövidítéseket, betűszavakat néhány, az infokommunikáció területén nagyon ismert és gyakran használt kifejezéstől (például IP, TCP, GPRS, UMTS) eltekintve ki kell fejteni logikusan az első használat alkalmával (például így: „A GPS (Generalized Processor Sharing) egy ideális folyadékmodellen alapuló csomagütemező eljárás.”).

A beszámoló készítése során előfordulhat, hogy a hallgató úgy érzi, hogy alfejezetekkel tagolva jobban olvasható és érthető lenne a beszámoló. Ennek akadálya nincs, de érdemes arra figyelni, hogy a túlzott tagolás sem tesz jót egy írásműnek, illetve hogy a címsorokban a rövidítések és a hivatkozások használata tilos. Tartalomjegyzéket készíteni nem szükséges a beszámolóhoz, de nem is tilos, kivéve azt az esete, amikor nyilvánvalóan terjedelemnövelési célokat szolgál.

A beszámoló terjedelme tárgyanként változhat. Általános szabály, hogy 1 hüvelyknél nagyobb margókat ne használjunk. A szöveg legyen egyszeres sortávú, sorkizárt és 12 pontos betűméretű. A bekezdések kezdődjenek behúzással a minta szerint.

2.2. Összefoglalás

A félévi munka során elért új eredmények ismételt, vázlatos, tömör Ebben a részben az adott félévre vonatkozó, az *Önálló laboratórium tárgy keretében elvégzett munka során* elért **új** eredmények ismételt, vázlatos, **tömör** összefoglalását várjuk, lehetőleg nem felsorolásként. Itt még egyszer ki lehet térni a leglényegesebb eredményekre, valamint a félév során felmerülő nehézségekre, de meg lehet említeni a továbbfejlesztési irányokat, lehetőségeket is.

Ezt a részt tagolható a következő pontok megválaszolásával:

- Mi volt az **aktuális kérdés**, probléma, amivel a félév során foglalkoztál?
- Mi a dolgozat **célja**, miért érdekes egyáltalán ezzel a problémával foglalkozni?
- Milyen **módszereket** használtál a probléma megoldása érdekében?
- Mik a legfontosabb **eredmények**?
- Milyen **következtetéseket** lehet levonni?

Ha valaki elolvassa ezt a részt, képet kell kapnia az egész dolgozatról. Ne legyen az absztrakt szó szerinti ismétlése.

Fontos, hogy az itt megadott sablontól el lehet térni, használata nem kötelező, csak segítséget jelenthet, viszont a fedőlap lehetőleg maradjon ugyanez és tartalmilag egyezzen meg a sablon irányelveivel. A beszámoló felépítésében nem érdemes eltérni a *Bevezető – Féléves munka és eredmények bemutatása – Összefoglaló* hármastól.

3. Irodalom, és csatlakozó dokumentumok jegyzéke

3.1. A tanulmányozott irodalom jegyzéke

[1] Douglas Karr, *Live Streaming Trends and Statistics (2024)*, https://martech.zone/live-streaming-trends-statistics/

[2] Saleh Alawaji, *IETF QUIC v1 Design, 2021. dec. 15.*, https://www.cse.wustl.edu/~jain/cse570-21/ftp/quic/index.html

[3] QUIC Working Group, *Multipath Extension for QUIC, draft-ietf-quic-multipath-14, 2025. apr. 24.*, https://datatracker.ietf.org/doc/draft-ietf-quic-multipath/14/

[4] Christian Huitema / Private Octopus Inc., *PicoQUIC GitHub repository*, https://github.com/private-octopus/picoquic

[5] IETF, *Media over QUIC Transport, 2025. apr. 28.*, https://datatracker.ietf.org/doc/draft-ietf-moq-transport/11/

[6] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks (Hotnets-IX)*, pages 19–24, Monterey, California, 2010. Association for Computing Machinery. https://doi.org/10.1145/1868447.1868466.

[7] Quicr, *LibQuicR GitHub repository*, https://github.com/Quicr/libquicr

[8] Umberto Eco, *Hogyan írjunk szakdolgozatot?*, Kairosz Kiadó, 2000, ISBN: 9639137537.

[9] Esterházy Péter, *Termelési-regény (Kisssregény),* Magvető Könyvkiadó, 2004, ISBN: 9631423948.

[10] *Tájékoztató a Műszaki Informatika Szak önálló laboratórium tantárgyainak 2008/9. tanév I. félévi lezárásáról a BME TMIT-en (VITMA367, VITMA380, VITT4353, VITT4330),* http://inflab.tmit.bme.hu/08o/lezar.shtml, szerk.: Németh Felicián, 2008. november 5.

[11] Wikipedia contributors, *Wikipedia:Academic use*, Wikipedia, The Free Encyclopedia, 2011 Nov 11. Available from:
http://en.wikipedia.org/w/index.php?title=Wikipedia:Academic_use&oldid=460041928

Itt jegyezném meg, hogy a tanulmányozott irodalmat hivatkozni kell a szövegben. Szükség esetén többször is. Az irodalomjegyzék célja (lásd a 3.1 fejezetet) ugyanis kettős[2]:

1. Az olvasó tájékoztatása, hogy a dokumentumban ki nem fejtett dolgoknak, a tudottnak vélt ismereteknek hol lehet bővebben utánanézni, így ott kell meghivatkozni az irodalmat [8, 9], ahová az irodalom kapcsolódik.

2. Megmutatni a tárgyfelelosnek/konzulesnek az elolvasott irodalom mennyiségét

Javasoljuk, hogy a hallgatók tanulmányozzák, hogyan néznek ki a hivatkozások a villamosmérnöki/informatikai szakma vezető szakmai folyóirataiban megjelenő cikkekben. Ebben a témavezető is biztosan tud segíteni. A hivatkozás teljességére és egyértelműségére tessék ügyelni. Például, ha egy könyvnek több, eltérő kiadása is van, akkor azt is meg kell jelölni, hogy melyik kiadásra hivatkozunk. A webes hivatkozások problémásak szoktak lenni, de manapság egyre több az olyan dokumentum, ami csak weben lelhető fel, ezért használatuk nem zárható ki. Itt is törekedni kell azonban a pontosságra és a visszakereshetőségre. A weben található dokumentumoknak is van címe, szerzője, illetve érdemes megadni a letöltés/olvasás időpontját is, hiszen ezek a dokumentumok idővel megváltozhatnak.

A wikipédiás hivatkozások használata nem javasolt, mert a wikipedia másodlagos forrás. Tájékozodjuk a wikipédián, de aztán olvassuk el az adott oldalhoz megadott hivatkozásokat is. A wikipedián külön szócikk foglalkozik azzal, hogy miért nem szerencsés tudományos munkákban a wikipédiára hivatkozni [11].

Nem publikus dokumentumok hivatkozása nem javasolt és csak kivételes helyzetben elfogadható!

[2] Akárcsak ennek a fejezet hivatkozásnak, ami a `\aref babel` parancsot demonstrálja

3.2. A csatlakozó dokumentumok jegyzéke

https://graphonline.top/

<A munka ezen beszámolóba be nem fért eredményeinek (például a forrás fájlok, mindenképpen csatolni akart forráskód részlet, felhasználói leírások, programozói leírások (API), stb.) megnevezése, fellelhetőségi helyének pontos definíciója, mely alapján a az erőforrás előkereshető – értelemszerűen nem nyilvános dokumentumok hivatkozása nem elfogadható.>