



**Budapesti Műszaki és Gazdaságtudományi Egyetem**  
Informatikai Tudományok Doktori Iskola

# **Út a felhő alapú mobilitáskezeléshez**

Tézisfüzet

Leiter Ákos

Tudományos témavezető: Dr. Bokor László, Ph.D.

BUDAPEST, 2025

# Tartalomjegyzék

<b>Rövidítésjegyzék.....</b>	<b>3</b>
<b>Bevezetés .....</b>	<b>4</b>
1.1 Kutatási irányelvek .....	5
1.2 Kutatási módszertan.....	6
<b>Új eredmények .....</b>	<b>7</b>
1.3 Első téziscsoport: Folyam-alapú, hálózatközpontú dinamikus mobilitáskezelés PMIPv6 alapon .....	7
1.3.1 Architektúra javaslat .....	7
1.3.2 Jelzés üzeneti folyamatok .....	<b>Error! Bookmark not defined.</b>
1.3.3 Kiértékelés .....	8
1.4 Második téziscsoport: Felhő alapú Mobile IPv6 és Proxy Mobile IPv6 .....	14
1.4.1 Felhő alapú MIPv6 és PMIPv6.....	14
1.4.2 A felhő alapú környezet műszaki költsége PMIPv6 esetén.....	14
1.4.3 Mikroszolgáltatás alapú tervezési koncepció Cloud-native Mobile IPv6 Home Agent számára.....	16
1.5 Harmadik téziscsoport: Zárt láncú orkesztációs eljárások felhő alapú Mobile IPv6 esetén .....	21
1.5.1 A zárt láncú orkesztáció funkcionális és performancia elemzése felhő alapú Mobile IPv6 Home Agent esetén.....	21
1.5.2 Analitikus modellezés és vizsgálatok .....	23
<b>Alkalmazások és következtetések .....</b>	<b>26</b>
<b>Publikációim.....</b>	<b>27</b>
<b>Hivatkozások .....</b>	<b>29</b>

## Rövidítésjegyzék

ANDSF	Access and Network Discovery Selection Function
BA	Binding Acknowledgement
BU	Binding Update
CFR	Current Flow Information Request
CNF	Cloud-native Network Function
CN-MIPv6	Cloud-native Mobile IPv6
CN-PMIPv6	Cloud-native Proxy Mobile IPv6
CoA	Care-of-Address
FUR	Flow Update Request
HA	Home Agent
HDR	Handover Discover Request
LMA	Local Mobility Anchor
MAG	Mobile Access Gateway
MIPv6	Mobile IPv6
MN	Mobile Node
MPU	Mobility Policy Update
MTBF	Mean Time Between Failure
MTTR	Mean Time to Repair
MUA	Mobility Update Accept
MUR	Mobility Update Request
NFV	Network Function Virtualization
NPoA	Network Point-of-Attachment
O&M	Operation and Management
ONAP	Open Network Automation Platform
PBA	Proxy Binding Acknowledgement
PBU	Proxy Binding Update
PCRF	Policy and Charging Rule Function
PMIPv6	Proxy Mobile IPv6
PoA	Point-of-Attachment
RA	Router Advertisement
RS	Router Solicitation
SDN	Software Defined Networks
TA	Tracking Area
UE	User Equipment

## Bevezetés

A folyamatosan növekvő hálózati forgalom és az egyre gyakoribb mobilitási események új kihívások elé állítják a telekommunikációs hálózatokat. A hálózati funkciók virtualizálása (Network Function Virtualisation, NFV), a felhő és a szoftver alapú hálózatok (Software Defined Networking, SDN) teljesen megváltoztatta a telekommunikációs infrastruktúrák alapjait. A mobilitáskezelés sem kivétel az említett trendek alól. A disszertációm az IP-alapú mobilitáskezelésre fókuszál, melynek a fő célja, hogy hálózatváltás vagy hálózatkapcsolódási pont csere (Point-of-Attachment, PoA) esetén az IPv6 cím állandó maradjon. Ehhez a Mobile IPv6 protokollcsaládot fogom alkalmazni (xMIPv6, MIPv6 [1]) és annak egyik speciális változatát, a Proxy Mobile IPv6-ot (PMIPv6 [2]). Megjegyzendő, hogy a disszertációmban említett mobilitáskezelés a TCP/IP alapú hálózatok 3. rétegében operál, nem a gyakran gondolt első vagy második rétegben.

A MIPv6 egy felhasználócentrikus mobilitáskezelési protokoll, míg a PMIPv6 hálózatcentrikus. A különbség ott található, hogy a mobil végberendezés (Mobile Node, MN) részt vesz-e a mobilitáskezelésben vagy sem. Ebből az következik, hogy PMIPv6 esetén a mobil végberendezés nem is tudja, hogy mobilitáskezelésben részesül.

Amiatt, hogy az IP réteg független az alatta lévő hálózati technológiától, az xMIPv6 protokollcsalád képes technológiafüggetlen hálózatváltásra pl. Wi-Fi és mobilhálózati interfész között. Továbbá ez lehetővé teszi a terheléelosztást is: ha van elérhető Wi-Fi hálózat, némely forgalmat érdemes azon keresztül kiszolgálni.

Viszont nem minden felhasználó igényel minden esetben mobilitáskezelést. Ezzel a területtel foglalkozik a Dinamikus Mobilitáskezelés (Dynamic Mobility Management, DMM). Célja továbbá optimalizálni a mobilitáskezeléshez kapcsolódó jelzésforgalom nagyságát.

A Dinamikus Mobilitáskezelés egyik logikus kiterjesztése a folyam-alapú mobilitáskezelés. Mind a MIPv6, mind a PMIPv6 rendelkezik ezzel a Flow Binding-nak hívott speciális technológiával [3] [4]. Fontos, hogy PMIPv6 folyam-alapú mobilitáskezelés esetén viszont már szükséges a mobil végberendezés módosítása és részvétele a folyamatokban. Ezek a megoldások együttesen lehetővé teszik az intelligens, folyam-alapú forgalomelosztást, ha több hálózat érhető el egyszerre.

A disszertációm első téziscsoportja egy új eljárást fog bemutatni – beleértve az architektúrális kiegészítéseket – azt célozva, hogy miként lehet a 3GPP által definiált hálózatokban dinamikus, folyamalapú, hálózatközpontú IP alapú mobilitáskezelést végrehajtani. Az eljárás vizsgálatához készítettem egy megfelelő tesztrendszer is, továbbá MATLAB segítségével szimulációkat hajtottam végre a működés elemzésére,

kiemelve, hogy az új modellem mennyire tudja csökkenteni a jelzésüzenetek adta terhelést.

A második téziscsoportom a Home Agent (HA) felhő alapú környezetben való működésével foglalkozik. Ehhez megterveztem egy felhő alapú Home Agent-ot (cloud-native Home Agent), majd létrehoztam a PMIPv6 Local Mobility Anchor (LMA) és Mobile Access Gateway (MAG) hálózati entitások felhő alapú koncepcióját. A működés számszerű kiértékelését prototípus implementálásának segítségével végeztem el.

A harmadik téziscsoportom a szoftveres rendelkezésre állást vizsgálja, ha a hálózatban orkesztrátor is található. A mikroszolgáltatás alapú szoftveres tervezés a felhőinfrastruktúra megjelenésével növeli a komplexitást, mivel sokkal több különálló szoftver komponens keletkezik. Ezen probléma kezelésére használható egy új entitás, az orkesztrátor. A MIPv6 alapú működés támogatására létrehoztam új eljárásokat, melyek képesek az automatikus hibajavításra és kapacitásnövelésre. Ezek alapján bemutatom, hogy a rendelkezésreállítás növelhető intelligens és automatikus orkesztráció segítségével és egyidejűleg csökkenthető a redundáns komponensek használata, mellyel erőforrást lehet így spórolni.

## **1.1 Kutatási irányelvek**

Az egyik alapvető célkitűzésem az volt, hogy létrehozzam és verifikáljam az intelligens terheléelosztás új, operátor centrikus módszerét 3GPP hálózatok számára. A Wi-Fi hálózatok olcsóbb hozzáférést ígérnek, mint a mobilhálózati interfészek. Ha tudjuk azonosítani azokat az alkalmazásokat a mobil végberendezéseken, melyeket a helyi Wi-Fi hálózatra tudunk terhelni, rádiós erőforrást és költséget takaríthatunk meg a mobilhálózatokban. Ez még hatékonyabb tud lenni, ha figyelembe vesszük a mobilitáskezelést is azért, hogy folyamatosan tudjuk biztosítani és megszakításmentessé tudjuk tenni egy adott alkalmazás kommunikációját. Viszont a mobil felhasználók mozgásának mintázata merőben változó lehet. Ezt egy matematikai modell segítségével kezeltem, melyet Haijun Zhang et al. [5] munkája alapján készítettem és egészítettem ki. Két fajta esetet fogok vizsgálni a modellem segítségével: 1) a mobil végberendezés a Wi-Fi hatósugarába érkezik és ott fejezi be a kommunikációt, 2) a mozgó felhasználó keresztülmegy a Wi-Fi hálózaton, majd újra visszakapcsolódik mobilhálózatához. Figyelembe vettem a gyors és lassú mozgású felhasználók arányát is, hogy kezelni lehessen pl. a gyalogosok és a gépjárművek számát. A modellem képes jellemezni az ezekhez kapcsolódó jelzésüzeneti forgalom karakterisztikáját, ha dinamikusan kezeljük a mobilitási igényeket.

Ez elvezet a disszertációm másik témájához, az IPv6 alapú mobilitáskezelés felhő alapú működési kihívásaihoz. Az új futtatási paraméterek, mint pl. a virtualizáció és a felhőben történő orkesztáció, mobilitáskezelés szintjén is új megközelítéseket kívánnak. Ha mobilitáskezeléssel párosult intelligens terheléelosztást szeretnénk megvalósítani, akkor

a hozzájuk kapcsolódó entitásoknak és eljárásoknak is kompatibiliseknek kell lenniük az új környezettel. Ezért tettem javaslatot a felhő alapú MIPv6 és PMIPv6 (cloud-native MIPv6, PMIPv6) megalkotására [C1][C2].

A kutatómunkám harmadik szakaszában az Open Network Automation Platform (ONAP)[6] segítségével mutattam be, hogy mit lehet nyerni a szoftveres rendelkezésre állás területén akkor, ha a felhő alapú mobilitáskezelési javaslatomat alkalmazzuk. Bemutatom az analitikus modellre épülő matematikai számításaimat és az aktuális implementációt is, hogy a redundancia helyreállítási idő gyorsasága hogyan növeli a rendelkezésreállást CN-MIPv6 esetén, miközben költséget csökkent a kevesebb redundáns elem használatával. Ezek az eredmények nem csak CN-MIPv6-ra vonatkozhatnak, hanem könnyen kiterjeszthetők más hálózati funkcióra is.

## 1.2 Kutatási módszertan

Minden téziscsoportom jól ismert statisztikai és analitikus módszerekre épül. A mobilitáskezelési és hálózatváltási valószínűségi modellem Haijun Zhang et al.[5] munkáira támaszkodik, azokat egészíti ki, és MATLAB-ban fejlesztettem le.

A CN-MIPv6 prototípusom C programozási nyelven lett implementálva, kiegészítve Python elemekkel az API hívások megkönnyítésére, Kubernetes felett, követve a mikroszolgáltatási irányelveket. A numerikus eredményeket ismert statisztikai eszköztárakkal elemeztem (pl. szórás, medián).

A szoftveres rendelkezésre állási számításaim a Varga et al. [7] munkájában bemutatott, telekommunikációra alkalmazott speciális módszerekre támaszkodik, mint pl. a Mean Time Between Failure (MTBF), és a Mean Time to Repair (MTTR). A diagrammok és a számítások Python segítségével lettek elkészítve. Az ONAP-t használtam orkesztrátornak, mely egy Python és JAVA alapú implementáció.

# Új eredmények

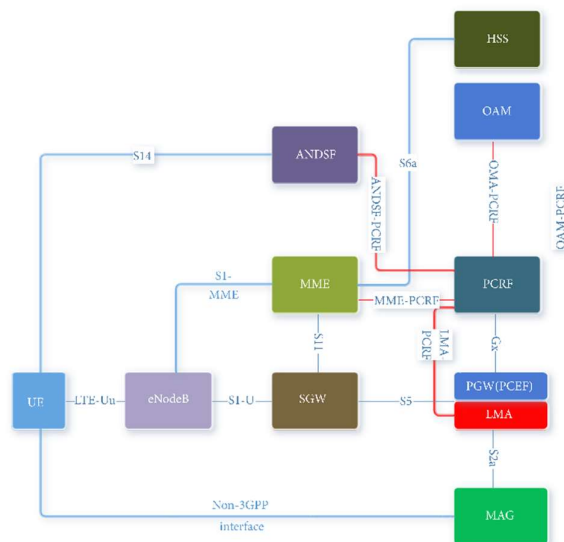
## 1.3 Első téziscsoport: Folyam-alapú, hálózatközpontú dinamikus mobilitáskezelés PMIPv6 alapon

A 3GPP által standardizált hálózatok nem rendelkeznek alkalmazáspecifikus, dinamikus mobilitáskezeléssel PMIPv6 felett. Ebben a téziscsoportomban bemutatom, hogyan lehet ezt az integrációt elvégezni és az intelligens terheléelosztást megvalósítani. Továbbá mérési eredményeimmel és a matematikai modellemmel mélyebben elemezem az általam javasolt megoldás teljesítményét.

### 1.3.1 Architektúra javaslat

**Tézis 1.1:** *Létrehoztam egy új architektúrát a folyam-alapú, hálózatközpontú mobilitáskezelés megvalósítására 3GPP alapú hálózatok számára Proxy Mobile IPv6 segítségével. Az új architektúra új IPv6 alapú interfészeket és kapcsolatokat tartalmaz a maghálózati elemek között és új funkciókat hoz be ahhoz, hogy támogassa az intelligens terheléelosztást is. Készítettem egy új eljárást, mely képes intelligens terheléelosztást indítani a 3GPP alapú mobil hálózati interfészekről Wi-Fi interfész felé Proxy Mobile IPv6 segítségével. A megoldás kompatibilis a meglévő IETF által standardizált jelzésűzenetekkel, de új elemeket is tartalmaz. [J1]*

Az 1.ábra bemutatja az új elemeket és interfészeket. A Policy Control and Charging Function (PCRF) nevű maghálózati csomópontot tekintem a központi elemnek, mely vezérli az intelligens terheléelosztást. Az ábrán pirossal jelölt új interfészek teszik lehetővé a végrehajtást és az információk cseréjét az egyes hálózati funkciók között.

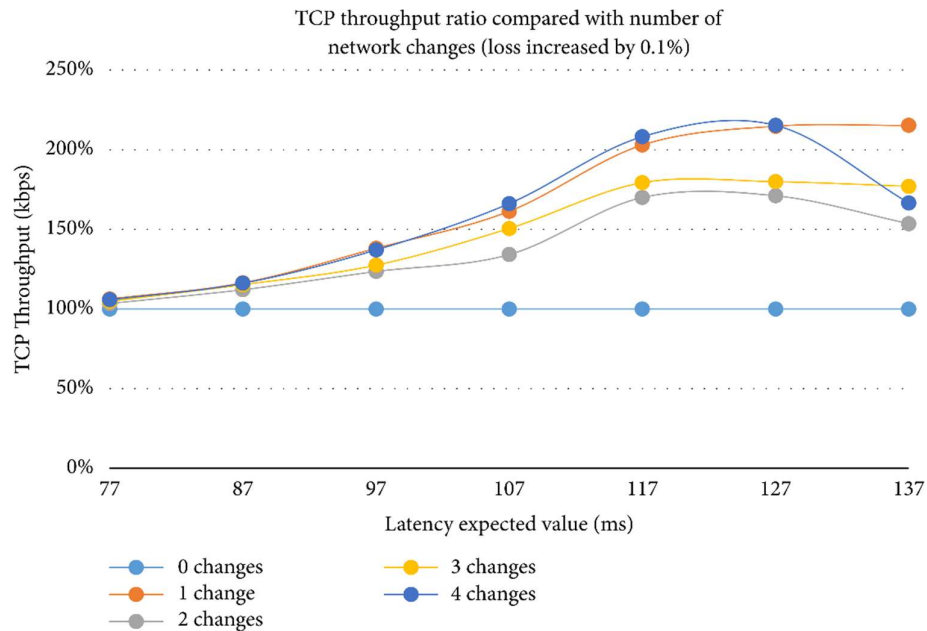


1. ábra - Architektúra javaslat folyam-alapú, hálózatközpontú dinamikus mobilitáskezelésre PMIPv6 segítségével

### 1.3.2 Kiértékelés

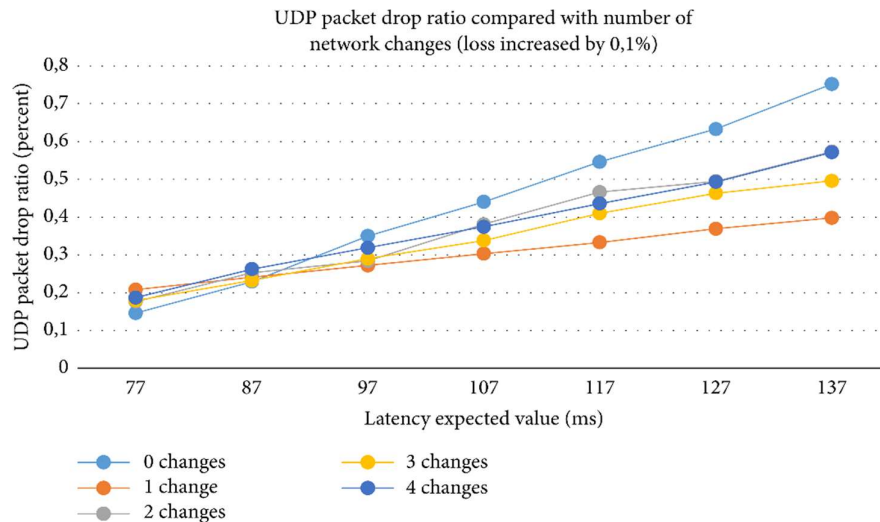
**Tézis 1.2:** Az elvégzett mérések során, valós teszrendszeren implementált modellen megállapítottam a TCP folyamatok hálózattváltásainak számszaki a korlátját egy adott időegységen belül, a TCP áteresztőképességének csökkenését véve határfeltételnek. Az elvégzett mérések során, valós teszrendszeren implementált modellen megállapítottam az UDP folyamatok hálózattváltásainak számszaki a korlátját egy adott időegységen belül: az UDP csomagvesztés lineárisan növekszik a hálózat váltások számához képest. [J1]

Ezen tézis fő megállapításait a 2. ábra mutatja be. Azokban az esetekben, mikor egyáltalán nincs hálózattváltás, a rendszer rosszabbul viselkedik, mint 4 hálózattváltás esetén. 3 váltás a maximum, amikor a TCP áteresztő képesség nem kezd el csökkenni.



2. ábra - TCP áteresztőképesség a hálózattváltások függvényében

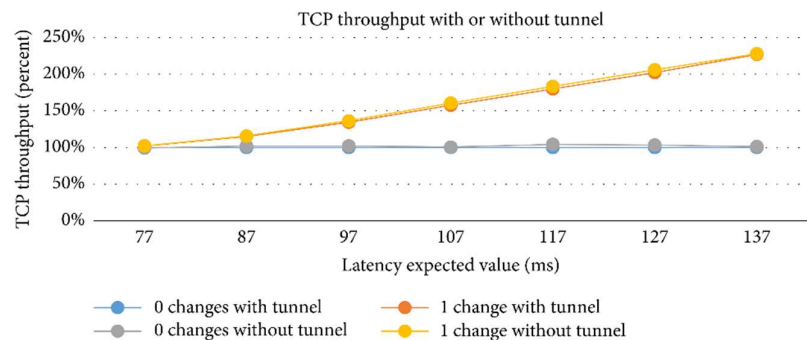
Azokban az esetekben, amikor semmilyen hálózattváltás nem történik, a legrosszabb a csomagvesztési arány (3.ábra). Akár 4 hálózattváltás is jobb egy percen belül, mint nem beavatkozni a rosszabbodó hálózati lefedettségbe.



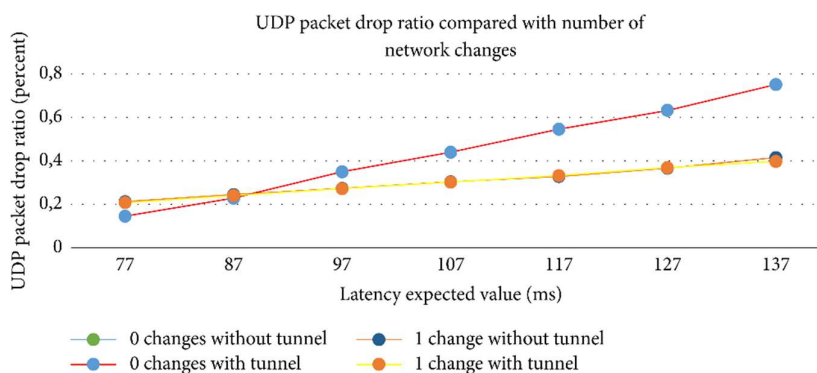
3. ábra - UDP csomagvesztési arány a hálózat váltások számának tükrében

**Tézis 1.3:** Az elvégzett mérések során, valós tesztrendszeren implementált modellen megállapítottam, hogy az alagutazási költség az adatsíkban elhanyagolható az architektúra javaslatomban mind a TCP, mind az UDP folyamatok esetén. [J1]

A 4. és az 5. ábra bemutatja az alagutazási költségeket. Ezek segítségével becslést lehet adni arra, hogy dinamikus mobilitáskezelés esetén műszaki költségben mit jelent az alagút használata.



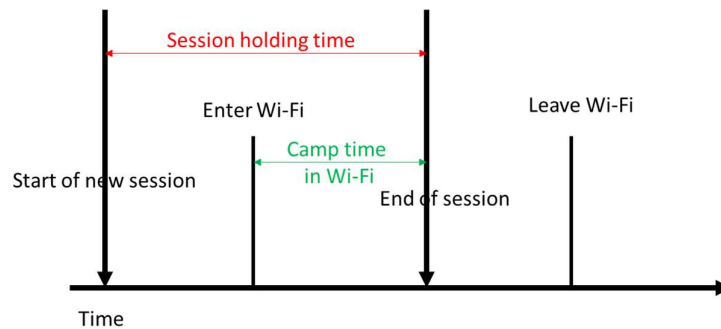
4. ábra - TCP alagutazás költsége



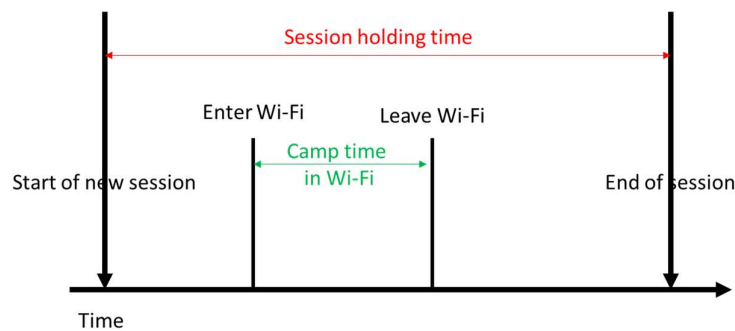
5. ábra - UDP alagutazás költsége

**Tézis 1.4:** Megállapítottam a jelzésüzenetek karakterisztikáját a különböző mobilitási forgatókönyvekben az általam készített modellben. Kimutattam, hogy a legnagyobb nyereség akkor következik be, mikor a lassan mozgó mobil végberendezések aránya a legnagyobb, de a legkisebb arányban vannak mobilitáskezelési igényű folyamatok. [J1]

Két esetet különböztettem meg (6. és 7. ábra), melyeket Haijun Zhang et al. [5] munkája alapján különítettem el: a mobil végberendezés keresztülhalad a Wi-Fi hálózaton vagy abban terminálja a forgalmát.



6. ábra - Eset #1: a felhasználó a Wi-Fi hálózaton fejezi be a forgalmazást



7. ábra - Eset #2: a felhasználó keresztül megy a Wi-Fi hálózaton

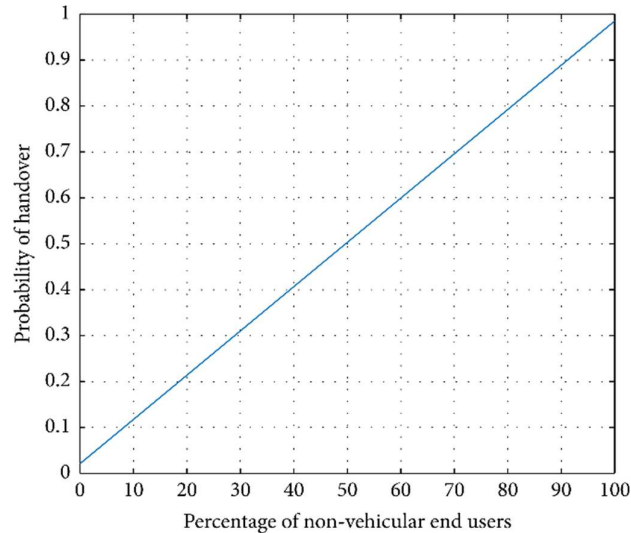
A modell kétfajta mobil felhasználót különböztet meg: alacsony mobilitási igényűt (non-vehicular,  $\alpha$ ) és magas mobilitási igényűt (vehicular users,  $1 - \alpha$ ). A modell feltételezi, hogy a makrocellában töltött idő (camp time) exponenciális eloszlású mindkét típusú felhasználói csoport számára. A makrocella átmérője 100x-osa a Wi-Fi cella átmérőjének ( $R, 100r$ ). Ehhez adtam hozzá azt a dinamikus mobilitáskezelést leíró arányszámot, ami megmondja, hogy egy adott végberendezésen futó alkalmazások hány százaléka igényel mobilitáskezelést.

A következő pszeudokód mutatja be a MATLAB implementációm releváns részeit:

```
# ratio of vehicular and non-vehicular users
LOOP from alfa=0 to 1 by 0.1
    CALCULATE handover probability for the two scenarios
    #ratio of sessions which requires dynamic mobility management in
    the following loop
    LOOP from 10 to 90 by 40
```

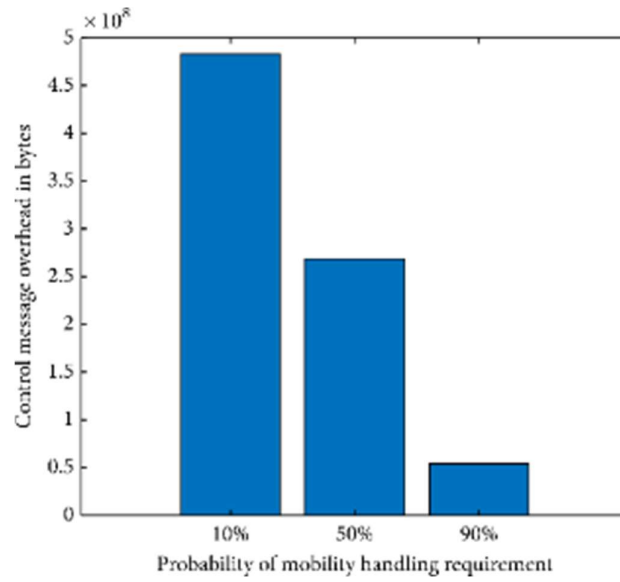
```
        CALCULATE control message overhead by actual vehicular user
        ratio, handover probability and mobility demand
    END of LOOP
END of LOOP
```

A 8. ábra mutatja a hálózatsváltási valószínűséget a magas és alacsony mobilitási igényű felhasználók arányában. A felállásban egy Wi-Fi cella van egy mobil cellába beágyazva, melyeken a felhasználó keresztülmegy és a Wi-Fi cellában hagyja abba az adatforgalmazást. A magas mobilitású felhasználó lehet egy jármű, míg az alacsony mobilitású egy gyalogos. A modell azt mutatja meg, hogy ha egy gyalogos keresztülmegy ezen az úton, akkor nagyobb valószínűséggel következik be hálózatsváltás, hisz a rendszernek lesz ideje ezt végrehajtani az gyalogos alacsony sebessége miatt. A modell azt is figyelembe veszi, hogy nem feltétlenül lesz hálózatsváltás akkor, ha valaki gyorsan mozog. Továbbá annak sem 1 a valószínűsége, hogy a gyalogos hálózatsváltásban részesül, hisz lehetséges, hogy pl. a bázisállomás körül köröz.

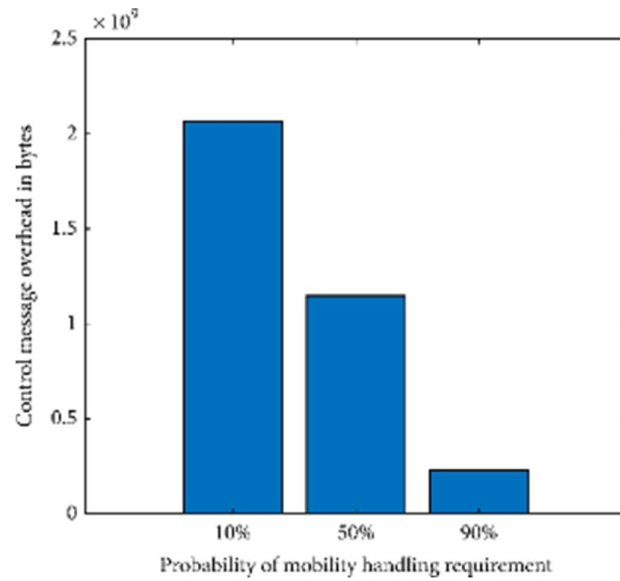


**8. ábra - Hálózatsváltási valószínűség a magas és alacsony mobilitási igényű felhasználók aránya alapján**

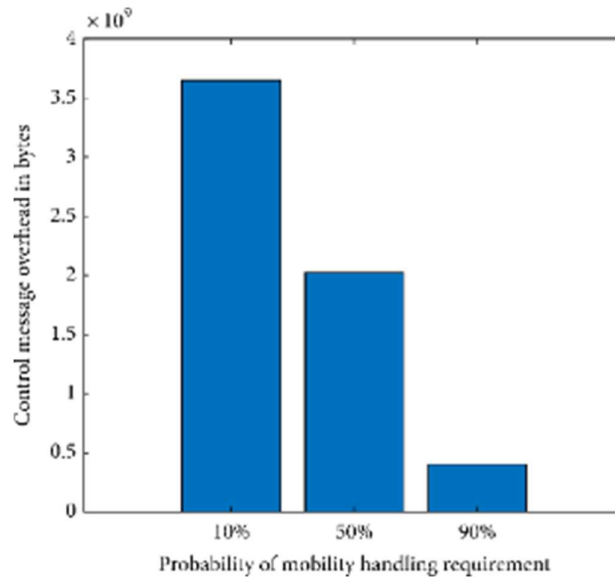
1000000 mobil felhasználóval és felhasználóként 30 adatfolyammal végeztem el a forgatókönyvek elemzését, ahol PBU és PBA üzenet váltások történtek (76 bájtos csomagméretet beállítva). Az x tengely mutatja azon alacsony mobilitási igényű felhasználók arányát, akik számára mobilitáskezelésre van szükség.



9. ábra - Jelzésüzeneti többlet 10%-os arányú alacsony mobilitási felhasználók esetén

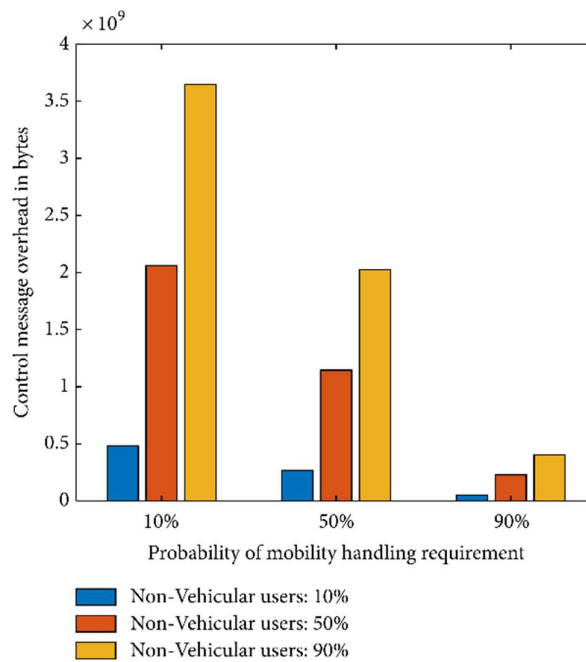


10. ábra - Jelzésüzeneti többlet 50%-os arányú alacsony mobilitási felhasználók esetén



11. ábra - Jelzésüzeneti többlet 90%-os arányú alacsony mobilitási felhasználók esetén

Kimutattam, hogy akkor érhető el a legnagyobb nyereség, mikor az alacsony mobilitásigényű felhasználók száma magas, de a legalacsonyabb azon forgalmi folyamatok aránya, melyek mobilitáskezelést igényelnek. A 12. ábra foglalja össze az eredményeket, melyek részletesen vannak kifejtve a 9. 10. és 11. ábrákon. Minél kevesebb a mobilitáskezelést igénylő felhasználó, annál nagyobb a felesleges jelzésüzenetek mennyisége, ha nincs a dinamikusság igénye is figyelembe véve.



12. ábra - Összefoglaló diagram a jelzésüzeneti többlet reprezentálására

## 1.4 Második téziscsoport: Felhő alapú Mobile IPv6 és Proxy Mobile IPv6

Ebben a téziscsoportban bemutatom, hogy miként lehet a MIPv6 HA, PMIPv6 LMA és PMIPv6 MAG funkciókat felhő alapú környezetben Kubernetes segítségével megvalósítani. A kiegészítéseim segítségével új funkciók és eljárások válnak lehetővé, mint például az automatikus hibajavítás és skálázás. Továbbá a mikroszolgáltatási tervezési irányelv lehetővé teszi a forgalom és a funkciók szeparálását, mellyel külön végrehajtóegységeket lehet hozzárendelni egy feladathoz. Így csak azt a komponenst kell skálázni, mely aktuálisan túlterhelt, amivel erőforrást lehet spórolni.

### 1.4.1 Felhő alapú MIPv6 és PMIPv6

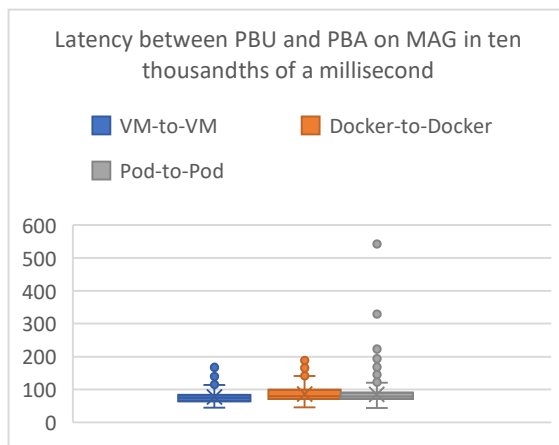
**Tézis 2.1:** *Megterveztem a MIPv6 HA felhő alapú architektúráját Kubernetes környezetben, melyet Cloud-native Mobile IPv6 neveztem el. Szintén megterveztem a PMIPv6 LMA és MAG felhő alapú architektúráját Kubernetes környezetben, melyet Cloud-native Proxy Mobile IPv6 neveztem el. [C2] [C1]*

A kialakított új architektúra követi a vezérlési és felhasználói adatfolyamok szeparációjának irányelvét (Control and User Plane Separation, CUPS [8]). Továbbá képes az egyes funkciókhoz külön feldolgozóegységet rendelni, amit Flow Bindings esetén is lehet használni.

### 1.4.2 A felhő alapú környezet műszaki költsége PMIPv6 esetén

**Tézis 2.2:** *Megállapítottam a konténerezés műszaki költségét PMIPv6 vezérlő sík esetén. Ehhez terveztem és készítettem egy tesztkörnyezetet, mellyel megmértem a releváns teljesítményértékeket. Az adatok statisztikai elemzésével megmutattam, hogy Docker és Kubernetes alapú telekommunikációs felhőinfrastruktúrákban szignifikáns átlagos késleltetésnövekedés (~11%) van jelen. Ez a műszaki költsége a konténerek használatának. [C3]*

Az eredményeket 13. ábra és az 1. táblázat foglalja össze.



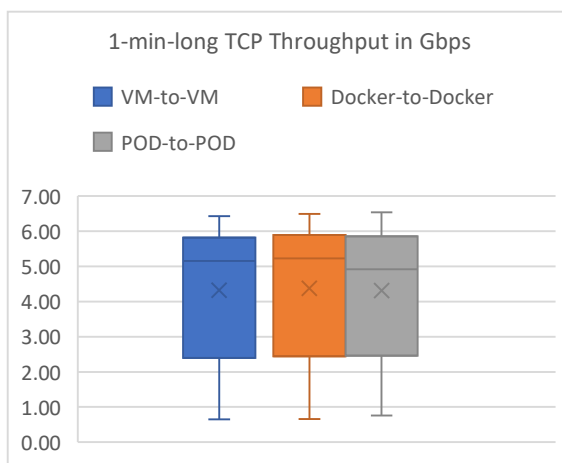
13. ábra - PBU és PBA üzenetek között eltelt idő (ms) a MAG-on

	Átlag (ms)	STDEV	MIN	MAX	Medián
VM-VM	7,7	1,8	4,5	18,6	7,5
Docker-Docker	8,6	2,4	4,6	20,5	8,0
POD-POD	8,5	2,9	4,4	54,2	7,9

1. táblázat - A vezérlőik mérésének numerikus eredményei (ms)

**Tézis 2.3:** Megállapítottam a konténerizáció műszaki költségét a PMIPv6 adatsíkra. Megmutattam, hogy nincs szignifikáns különbség a TCP áteresztő képességben, ha felhő alapú környezetet használok az IPv6-in-IPv6 típusú alagutazás esetén. [C3]

A 14. ábra és a 2. táblázat tartalmazza a részletes eredményeket.



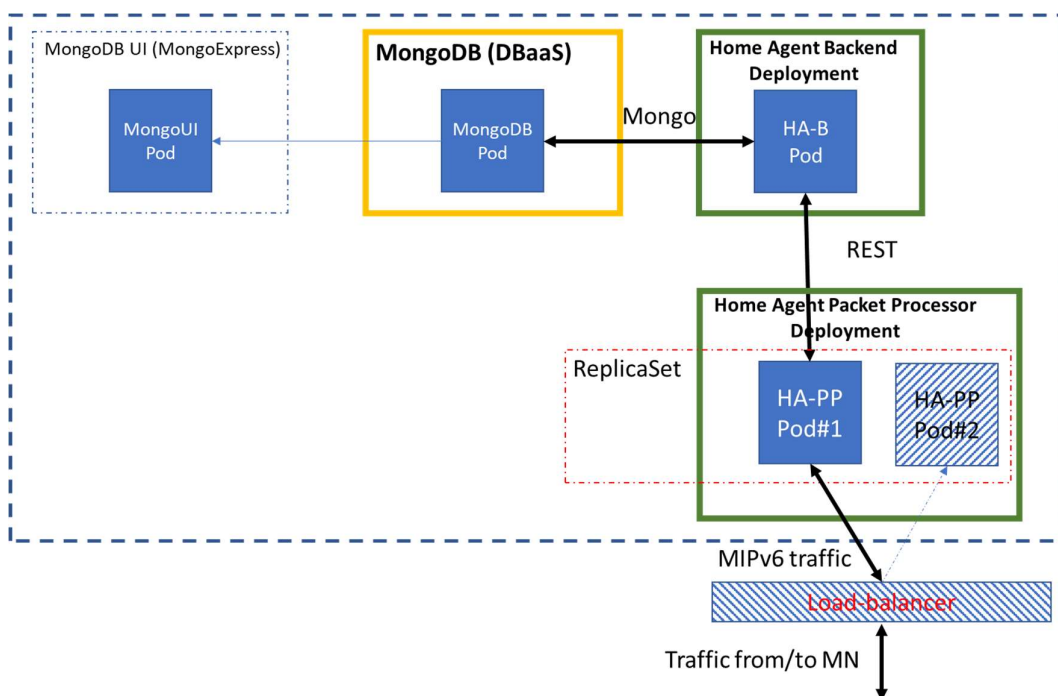
14. ábra – Áteresztőképesség 1 perc hosszú TCP folyamatok esetén IPv6-in-IPv6 alagutazás használatkor (Gbps)

	Átlag (Gbps)	STDEV	MIN	MAX	Medián
VM-VM	5.21	0.65	4.15	6.43	5.17
Docker-Docker	5.32	0.66	4.22	6.50	5.23
POD-POD	5.18	0.76	4.17	6.54	4.92

2. táblázat - Az adatsík vizsgálatának numerikus eredményei (Gbps)

### 1.4.3 Mikroszolgáltatás alapú tervezési koncepció Cloud-native Mobile IPv6 Home Agent számára

**Tézis 2.4:** Elkészítettem egy mikroszolgáltatási tervezési irányelveket követő szoftveres architektúrát a MIPv6 HA számára Kubernetes környezetben történő működés támogatására. Készítettem egy modellt és implementáltam hozzá egy teszrendszer, hogy megvizsgáljam a több interfész egyidejű használatát Kubernetes Pod esetén, támogatva a CN-MIPv6 HA vezérlő síkját. Megmutattam, hogy a MACVLAN-nal megvalósított több interfész támogatás növeli a késleltetés ingadozás mértékét (átlagosan 55%-al), de nincs szignifikáns hatással a késleltetésre (2%-os növekedés átlagosan). [C3]

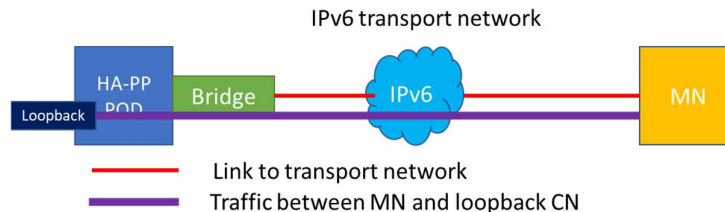


15. ábra - CN-MIPv6 mikroszolgáltatás alapú architektúrája

Kétfajta Kubernetes POD tartozik a kialakított architektúrához, melyet a 15. ábra mutat be. Az egyik a Home Agent Packet Processor (HA-PP), mely a MIPv6 standardizált forgalmának kezelésért felelős. A HA-PP JSON objektumok segítségével kommunikál a másik új entitással, amint Home Agent Backend (HA-B) névre kereszteltem. Ez felelős a külső szolgáltatásokkal történő kommunikációért és a Binding Cache kezelésért (MongoDB alapokon [9]). Annak ellenére, hogy van nem szabványosított forgalom is a rendszerben, külső, L3 kapcsolatok esetén teljesen megőrzi a szabványosított interfészt. A HA-PP egy különálló Kubernetes csomag, emiatt önállóan lehet skálázni vagy hibajavítási folyamatok alá vetni. Ezt a Kubernetes automatikusan is meg tudja oldani. A

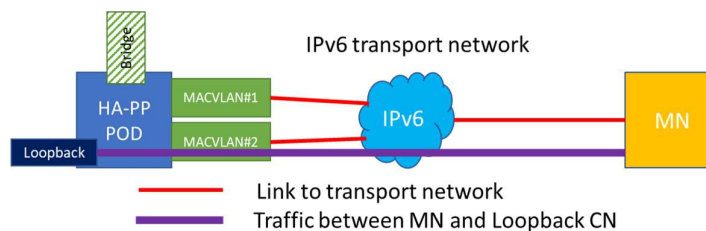
Kubernetes Readiness és Liveness nevű eljárásai folyamatosan figyelik, hogy az IPv6 Mobility Header RAW socket működik-e vagy sem, így szükség esetén azonnal be lehet avatkozni.

A 16. ábra mutatja be a Calico alapú tesztszrendszert. A HA-PP Pod-nak 1 db interfésze van. Itt nem csak MIPv6-al kapcsolatos üzenetek mennek, hanem a Calico saját belső üzenetei is.



16. ábra - Calico alapú tesztkörnyezet

A 17. ábra mutatja be a második tesztszrendszert, ahol már a Multus kiegészítés is használatos. Itt már el van választva a MIPv6 forgalom, mely a Multus interfészen megy, MACVLAN alapon megvalósítva.



17. ábra - Multus-szal kiegészített Calico alapú tesztkörnyezet

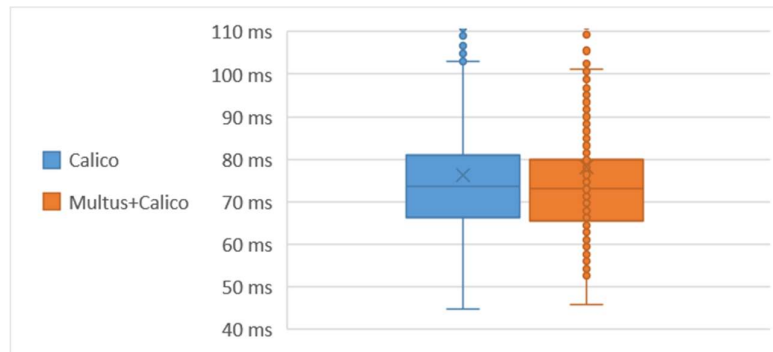
A fentiek alapján készítettem el a tesztkörnyezeteket. Minden forgalom az MN és a HA-PP-n levő loopback típusú interfészen terminálódik, kivéve az áteresztőképesség mérés forgalmát. Ez az a mérés, ami egyébként megmutatja a Calico alkalmazásának hátrányait is a későbbiekben.

A késleltetést körülfordulási időként definiáltam (Round-Trip-Time, RTT), ami a BU kiküldése és a BA megérkezése közötti időt fedi le az MN nézőpontjából, 10000-szer végrehajtva. Az #A mérés az új csomópontok hálózati csatlakozására fókuszál, míg a #B mérés a BC frissítésével foglalkozik, melyet a hálózatváltás vizsgálatára lehet használni.

#### **#A mérés: új csatlakozás vizsgálata**

**Cél:** megvizsgálni, hogy egy MN új regisztrációja késleltetési szempontból mit jelent CN-MIPv6 HA esetén.

**Következtetés:** a késleltetés szórása Multus használata esetén lényegesen nagyobb (18. ábra, 3. táblázat). Ez bizonytalanságot visz a rendszerbe új MN csatlakozása esetén.



18. ábra - RTT of MN új csatlakozása esetén (ms)

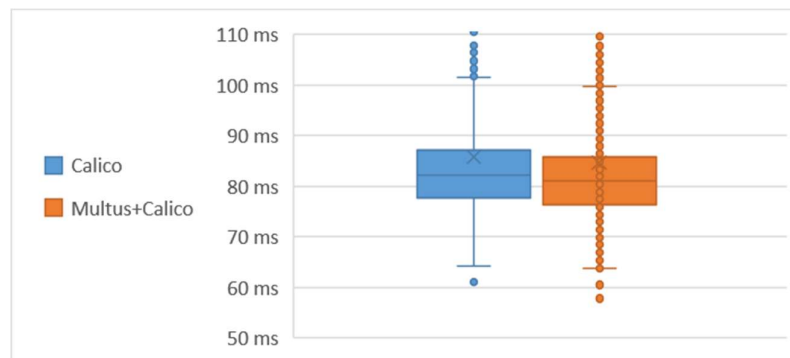
	Átlag (ms)	Medián	STDEV	MIN	MAX
Calico	76,26	73,7	102,73	44,9	5258,1
Multus+Calico	78,25	73,1	160,32	45,7	5255,2

3. táblázat - Numerikus eredmények új MN csatlakozása esetén (ms)

### Mérés #B: Meglévő csatlakozási kötés (binding) frissítése

Cél: megvizsgálni a késleltetést abban az esetben, mikor egy meglévő csatlakozást frissítünk a BC-ben.

Következtetés: Multus-szal kiegészített Calico használata esetén a késleltetés szórása nagyobb (19. ábra, 4. táblázat). Ez bizonytalanságot visz a hálózattváltási folyamatokba.



19. ábra - RTT meglévő BC bejegyzés esetén (ms)

	Átlag (ms)	Medián	STDEV	MIN	MAX
Calico	85,8	82,3	114,15	1,1	215,2
Multus	84,65	81	125,05	7,8	269,7

4. táblázat - Numerikus eredmények meglévő BC bejegyzés frissítése esetén (ms)

**Tézis 2.5:** A tesztrendszer alapú modellem segítségével megvizsgáltam a több interfész alkalmazásának hatásait Kubernetes Pod-ok esetén CN-MIPv6 HA tekintetében, mikor párhuzamosan több IP folyam is használatban van. Megmutattam, hogy több interfész

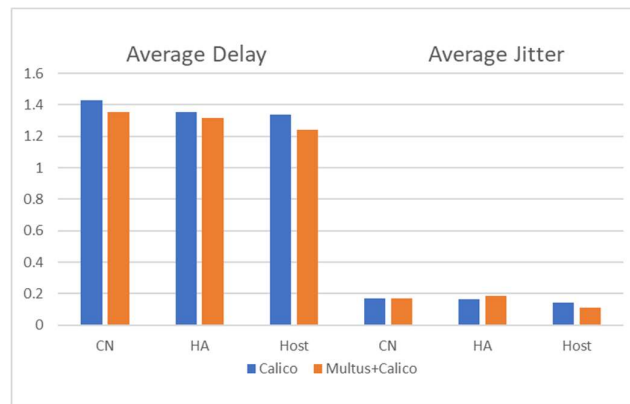
használata esetén a TCP áteresztőképesség szignifikánsan, átlagosan 52%-al nagyobb. [C3]

Az adatsík IPv6inIPv6 alagutazási technikát alkalmaz. A két tesztkörnyezetben a késleltetést, a késleltetésingadozást és az áteresztőképességet vizsgáltam. Az UDP és TCP forgalmakat D-ITG [10] segítségével generáltam, kivéve az áteresztőképesség vizsgálatot, amire Iperf3-at [11] használtam. A forgalom mindig az MN-ről indult és a CN-en végződött.

### #C mérés: késleltetés és késleltetésingadozás

**Cél:** megvizsgálni a késleltetést és késleltetésingadozást UDP forgalom esetén, majd kiértékelni a konténerizáció műszaki költségét.

**Következtetés:** késleltetés szempontjából nem állapítható meg szignifikáns különbség a konténerizált HA esetén a nem konténerizálthoz viszonyítva (20. ábra, 5. táblázat).



20. ábra - Átlagos UDP késleltetés és késleltetésingadozási értékek a mobil végberendezés és a CN, HA, majd a hoszt között

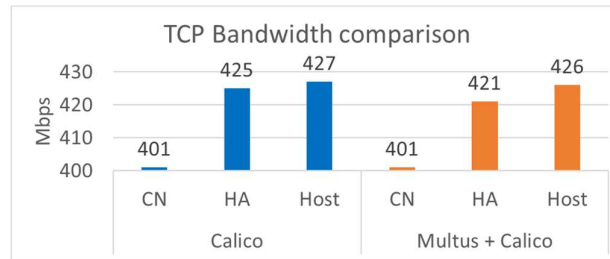
	Fogadó	Max késleltetés	Min késleltetés	Átlagos késleltetés	Átlagos Jitter
Calico	CN	9,497	0,946	1,431	0,17
	HA	9,398	0,957	1,352	0,166
	Host	8,538	0,988	1,338	0,144
Mulus	CN	7,846	1,016	1,353	0,171
	HA	7,817	0,957	1,317	0,186
	Host	7,781	0,942	1,24	0,109

5. táblázat – A késleltetés és a késleltetésingadozás numerikus eredményei (ms)

### #D mérés: TCP áteresztőképesség

**Cél:** megvizsgálni a TCP áteresztőképesség műszaki költségét konténerizáció esetén.

**Következtetés:** TCP áteresztőképesség szempontjából nincs különbség a két megoldás között (21. ábra), viszont Multus használata esetén lehet több interfészt is hozzáadni a Podhoz.



21. ábra - TCP átteresztőképesség vizsgálata

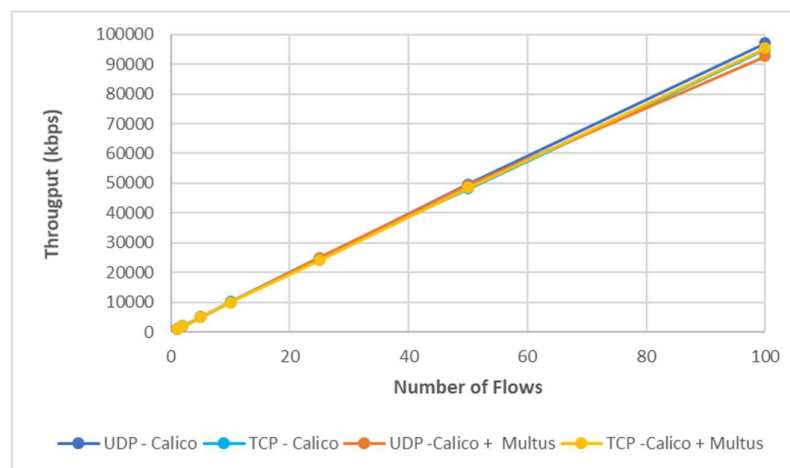
**Cél:** megállapítani a TCP átteresztőképességet a Multus által támogatott több interfész használata során, a Calico teljesítményéhez viszonyítva. Ebben az esetben a CN egy virtuális gép, és külön interfész van dedikálva a ki- és a bemenő forgalomra. 10 perc hosszú TCP folyamatot használtam, Iperf3 segítségével generálva.

**Következtetés:** az átlagos átteresztőképesség normál Calico esetén 271 Mbps, míg Multus esetén 412 Mbps. Szignifikánsan jobban teljesít a több interfészes beállítás Multus-szal.

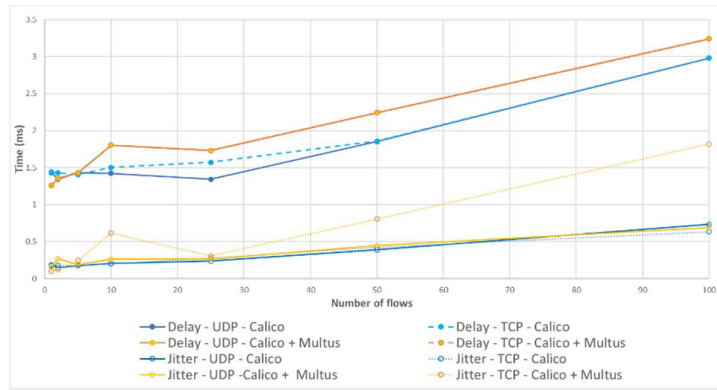
**#E mérés: folyam alapú viselkedés vizsgálata**

**Cél:** megvizsgálni, hogy miként viselkedik a rendszer több folyam egyidejű használata esetén, információt szolgáltatva a Flow Mobility alkalmazások optimalizációjához.

**Következtetés:** az átlagos késleltetésingadozás szignifikánsan nagyobb Multus esetén és TCP forgalom használata során. Minél több TCP folyamat használlok, annál nagyobb a késleltetés ingadozás. Az UDP forgalom nem mutatja ezt a szignifikáns különbséget. Átteresztőképesség szempontjából nincs szignifikáns különbség a két modell között (22. ábra, 23. ábra).



22. ábra - Átteresztőképesség a folyamatok számának arányában



23. ábra - Átlagos késleltetés és késleltetésingadozás a folyamatok számának arányában

## 1.5 Harmadik téziscsoport: Zárt láncú orkesztációs eljárások felhő alapú Mobile IPv6 esetén

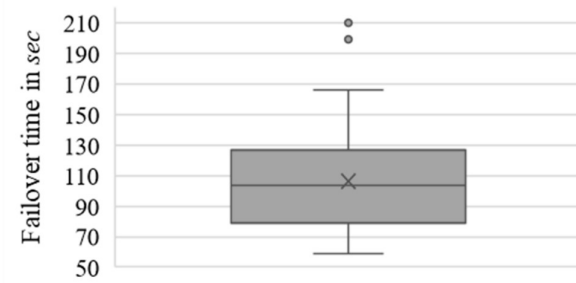
### 1.5.1 A zárt láncú orkesztráció funkcionális és performancia elemzése felhő alapú Mobile IPv6 Home Agent esetén

**Tézis 3.1:** *Javasoltam és elkészítettem egy zárt láncú orkesztrációs keretrendszert a felhő alapú Mobile IPv6 számára, Open Network Automation Platform segítségével. A keretrendszer képes detektálni a hibajavítási és a skálázási igényt MIPv6 HA esetén. [J3]*

A hibajavítás azon folyamat, amikor a rendszer érzékeli, hogy egy adott hálózati funkció nem megfelelően működik és a forgalmat automatikusan egy másik hálózati funkcióra tereli át. A skálázás folyamata során több új végrehajtó hálózati funkciót állítunk hadrendbe, hogy ki lehessen szolgálni a forgalomnövekményt. A méréseim során ezen folyamatokat modelleztem valós implementációt használva és vizsgáltam meg a felhő alapú MIPv6 kontextusában, ahol a HA-PP típusú Pod viselkedését mértem.

**Tézis 3.2:** *Modelleztem és megvizsgáltam a hibajavítási és skálázási folyamatokat a felhő alapú MIPv6 esetén. Megmutattam, hogy egy telekommunikációs környezetben, ahol több fizikai lokáción is található adatközpont, a hibajavítási idő akár kevesebb mint 1/30-ada is lehet az általános adatközponti értékeknek. A skálázási folyamatokat vizsgálva kimutattam, hogy akár 67%-al is több kapacitás adható a TCP áteresztőképességhez. [J3]*

Az eredményeket a 24. ábra és a 6. táblázat foglalja össze. A mérések során tapasztalható volt a használt ONAP implementáció belső komponenseinek az instabilitása, ami magyarázza a magasabb szórásértékeket.



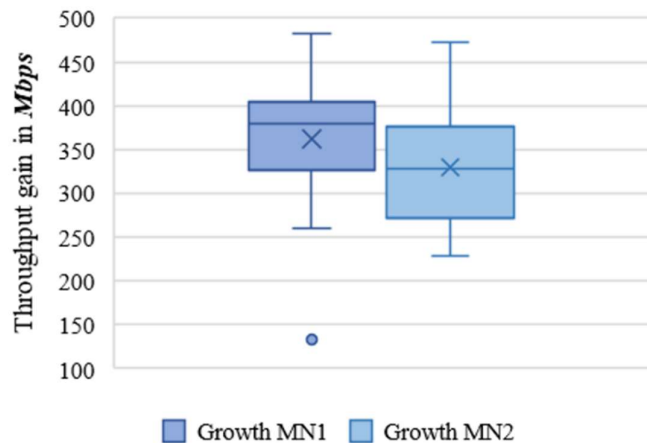
24. ábra - Az automatikus hibajavítás végrehajtási ideje (sec)

MIN	Átlag	Medián	MAX	STDEV
58,83	106,26	103,92	210,19	30,42

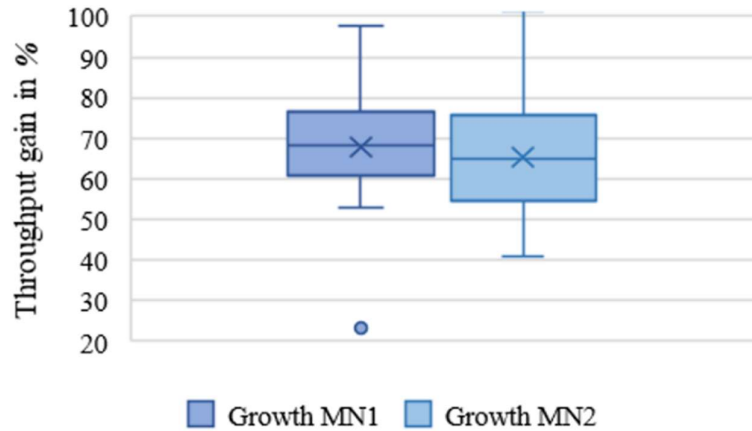
6. táblázat - Az automatikus hibajavítás numerikus eredményei (sec)

Skálázás esetén két mobil végberendezést használtam (MN#1, MN#2), melyek egy darab HA-PP végrehajtóhoz voltak csatlakoztatva. Amikor az ONAP észreveszi, hogy az első számú HA-PP (HA-PP#1) túlterhelődött, létrehoz egy újat (HA-PP#2), melyhez az MN#2 újra regisztrál és kiépítik az alagutat. Minden külső elem (CN) mostantól a HA-PP#2-n keresztül éri el az MN#2-t.

Minden Pod hálózati interfésze 1 Gbps-re van limitálva, hogy a kapacitáskorlátot is lehessen szimulálni. A mérés húsz alkalommal történt. Az eredmények a 25. és 26. ábrán láthatóak dobozábraként, míg a numerikus eredményeket a 7. táblázat foglalja össze. Az MN#2 áteresztőképesség növekménye azért kisebb az MN#1-hez képest, mert az MN#2 az újregisztrálás során nem forgalmazott.



25. ábra - Áteresztőképesség skálázás után (Mbps)



26. ábra - Áteresztőképesség skálázás után (százalék)

MN1 throughput nyereség (Mbps)	MN1 throughput nyereség (%)	MN2 throughput nyereség (Mbps)	MN2 throughput nyereség (%)
362,1 Mbps	67,94%	329,5 Mbps	65,4%

7. táblázat - Numerikus eredmények skálázás után

## 1.5.2 Analitikus modellezés és vizsgálatok

A telekommunikációs hálózatok egyik legfontosabb minőségi ismérve a rendelkezésre állás, melyet többek között a hibajavítás automatizálásával lehet javítani. Az alapvető képlet az alábbi:

$$(1) A_s = \frac{MTBF}{MTBF + MTTR}$$

ahol a hibák közötti időt MTBF-nek nevezzük (Mean Time Between Failures), és a hibajavítás idejét MTTR-nek (Mean Time to Repair).

Magas rendelkezésre állású, de alacsony késleltetésű (Ultra Reliable Low Latency Communication, URLLC) rendszerek esetén a követelmények jelentősek, emiatt érdemes redundanciát rakni a rendszerbe. Nem léteznek széleskörűen és egységesen elfogadott értékek szoftver MTBF és MTTR esetére, így tartományokat használtam: MTBF esetén 1 hónaptól két évig, MTTR-nél 5 perctől 45 percig. Különböző redundanciamegközelítéseket alkalmaztam, hogy 99,9999% fölé menjen a rendelkezésreállás. Mivel alacsonyabb MTTR érték minden esetben növeli a rendelkezésreállást, ezért kevesebb redundáns elem is elég ahhoz, hogy ugyanazt a rendelkezésre állást biztosítsuk. Emiatt erőforrásokat lehet megspórolni, hisz a gyors automatikus hibajavítási eljárások ezt lehetővé teszik.

2N és 3N redundancia esetén az alábbi egyenleteket lehet használni:

$$(2) A_{2N} = 1 - (1 - A_S)^2$$

$$(3) A_{3N} = 1 - (1 - A_S)^3$$

**Tézis 3.3:** *Kiszámoltam a felhő alapú Mobile IPv6 rendelkezésre állását azon eredmények alapján, amiket a Tézis 3.2-ben mértem. Megmutattam, hogy az automatikus hibajavítás miatt ONAP környezetben 3N helyett 2N redundancia is elég, melynek segítségével erőforrást lehet megtakarítani. [J3]*

A Tézis 3.2-ben megmutattam, hogy az átlagos automatikus hibajavítási idő 106,23 sec. Ezt az értéket veszem MTTR értéknek. A célom, hogy a számításokkal megmutassam, hogy a „öt kilences” rendelkezésre állás biztosítható kevesebb redundáns elemmel is (27. ábra).

#### 1N redundancia nélküli eset:

A fenti egyenletek alapján kiszámoltam az MTBF értékeket úgy, hogy legalább 99,999% legyen a rendelkezésre állás. Erre 123 napot kaptam, ami azt jelenti, hogy a rendszer összeomlása nem történhet gyakrabban, mint 123 naponta egy évben. Ezt veszem alapértelmezett értéknek.

#### 2N redundáns eset:

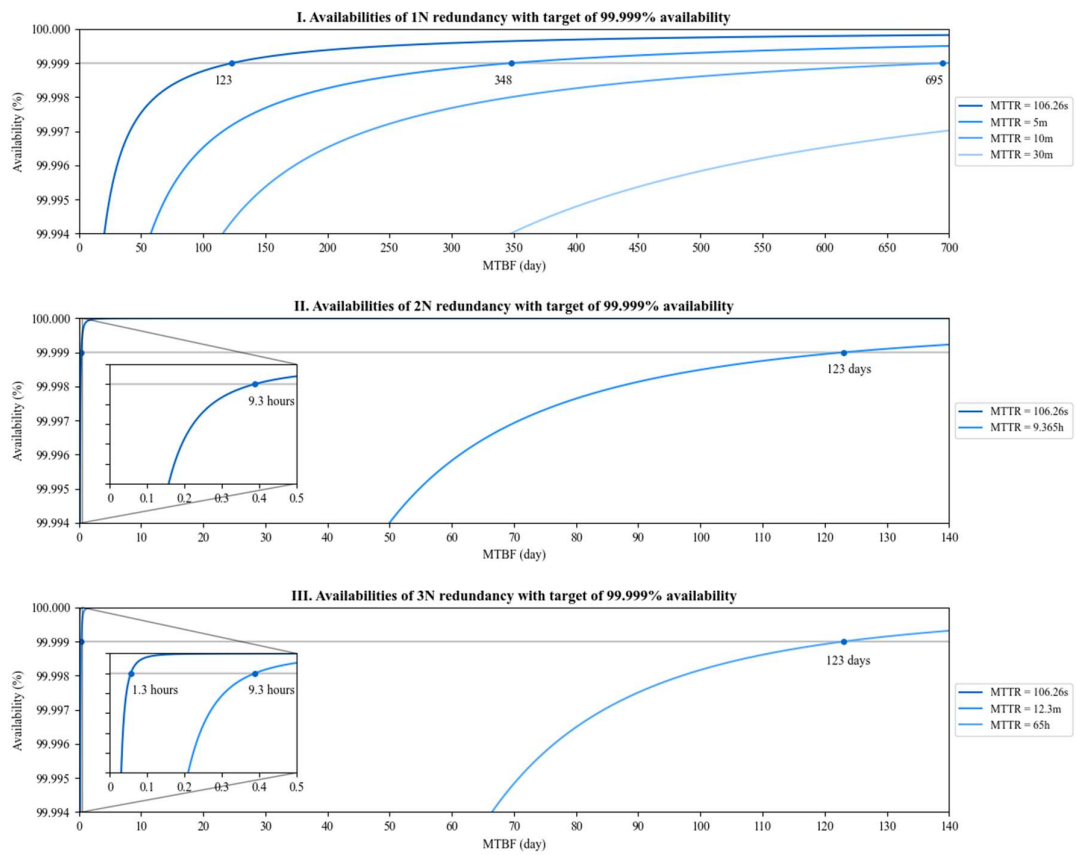
2N redundancia és MTTR=106,23 sec esetén  $MTBF_{2N}=9,3$  értékű lesz, ha a 99,999%-os rendelkezésre állást kell tartani. Tehát, ha minden 9,3 órában van egy hiba, akkor még a rendszer képes a 99,999%-os rendelkezésre állást biztosítani.

#### 3N redundáns eset:

3N redundancia esetén és MTTR=106,23 sec értékű, akkor az  $MTBF_{3N}=1,3$  óra lesz, ha a 99,999%-os rendelkezésre állást tartjuk. Így, ha az  $MTBF_{3N}$  értéke 1,3 óra vagy kevesebb, akkor a rendszer már nem teljesíti a 99,999%-os rendelkezésre állást. Tehát maximum 1,3 óránként jöhet hiba és a rendszer így 99,999%-os rendelkezésre állású marad.

Akkor tudunk erőforrást spórolni, ha kevesebb redundáns elem kell a rendszerbe. Ha tudjuk, hogy a rendszerünk átlagosan minden 123. napon hibázik, akkor nézzük meg, hogy mi az a maximum MTTR érték, ami még 2N redundancia esetén megtartja a 99,999%-os rendelkezésre állást. Ez  $MTTR_{2N}=9,365$  óra. Ez azt jelenti, hogy minden 2N rendszert, melynél az  $MTBF=123$  nap, le lehet egyszerűsíteni 1N rendszerre, ahol az MTTR minimum 9,365 óra. De 1N rendszer használata során még lesz szolgáltatás kiesés. Emiatt a 3N esetet érdemes megvizsgálni, hasonló logikával. Ha az  $MTBF=123$  nap, akkor számoljuk ki a minimum MTTR értéket, ami már nem ad több rendelkezésre

állást a 99,999%-on felül 2N-hez képest. Minden 3N rendszer esetén elég 2N rendszert használni, ha az MTTR 12 perc és 65 óra között van.



27.ábra – Redundancia elemzés

# Alkalmazások és következtetések

A forgalom átirányítása 3GPP alapú interfészről helyi (pl. Wi-Fi) interfészre képes erőforrást spórolni a rádiós rendszerekben. Ezek a rádiós csatornák az egyik legnagyobb költségelemei a mobilhálózatoknak. A disszertációm új elemeket és lehetőségeket ad a kommunikációs rendszerek terheléselosztással foglalkozó részeihez, hogy még hatékonyabban és pontosabban lehessen a vonatkozó folyamatokat végrehajtani. Eredményeim alkalmazásának segítségével megszakadásmentessé lehet tenni a terheléselosztásban részvevő forgalmakat, ami új szintre emelheti a rendszer dinamikusságát. Az általam javasolt az alkalmazás- és hálózatközpontú megoldással az IP folyamatok mozgatása kiegyensúlyozottabbá válik és a felhasználók kevésbé fogják érzékelni a negatív hatásait.

Valós környezetben figyelembe kell venni, hogy ki tulajdonolja a Wi-Fi és a mobilhálózatot: egyszerűbb, ha mind a kettőnek ugyanaz a tulajdonosa. Ez egy gazdasági limitáció, mely szűkítheti az alkalmazhatóságot. Ahogy a disszertációban is említem, különböző maghálózati elemek együttműködése szükséges az általam vizsgált és bemutatott rendszer teljes értékű kihasználására (pl. User Plane Function, Policy Control Function, Non-3GPP Interworking Function). Ezeket együttesen kihívás lehet karbantartani üzleti és szervezeti okok miatt. A szabványok használata kulcskérdés a megoldás során, ahol több szervezetnek is együtt kell működnie, pl. IETF, IEEE és 3GPP.

A konténerizáció segítségével könnyebben üzemeltethető alkalmazások jöhetnek létre, még annak ellenére is, hogy az atomikus elemek száma növekszik és orkesztráció is szükséges. Az ebből adódó műszaki költségeket bőven kompenzálja az új funkcionalitások és a hálózatautomatizáció ereje. Ezek nem csak egyes elemek funkcióinak fejlesztését jelentik, hanem a teljes rendszer működésére hatással vannak. A javaslataimmal a rendelkezésre állás költsége csökkenthető, hisz az azonnali és gyors beavatkozás a rendszerbe feleslegessé teszi a tartalék elemek egy részét. A bemutatott automatizációs eljárások tovább fejleszthetőek és alkalmassá tehetik a rendszert prediktív karbantartásra, ami szintén az üzemeltetési költségeket csökkenti.

## Publikációim

[J1] Ákos Leiter, László Bokor, "A Flow-Based and Operator-Centric Dynamic Mobility Management Scheme for Proxy Mobile IPv6", *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 4567317, 21 pages, 2019. <https://doi.org/10.1155/2019/4567317>

[J2] Ákos Leiter, Mohamad Saleh Salah, László Pap, and László Bokor, "Survey on PMIPv6-based Mobility Management Architectures for Software-Defined Networking", *Infocommunications Journal*, Vol. XIV, No 2, June 2022, pp. 2-18., <https://doi.org/10.36244/ICJ.2022.2.1>

[J3] Ákos Leiter, Edina Lami, Attila Hegyi, József Varga and László Bokor, "Closed-loop Orchestration for Cloud-native Mobile IPv6", *Infocommunications Journal*, 2023 1<sup>st</sup> issue, DOI: 10.36244/ICJ.2023.1.5

[C1] Ákos Leiter, László Bokor, and István Kispál. 2020. An Evolution of Mobile IPv6 to the Cloud. In *Proceedings of the 18th ACM Symposium on Mobility Management and Wireless Access (MobiWac '20)*. Association for Computing Machinery, New York, NY, USA, 137–141. <https://doi.org/10.1145/3416012.3424633>

[C2] Ákos Leiter, Nándor Galambosi, and László Bokor. 2021. An Evolution of Proxy Mobile IPv6 to the Cloud. In *Proceedings of the 19th ACM International Symposium on Mobility Management and Wireless Access (MobiWac '21)*. Association for Computing Machinery, New York, NY, USA, 107–115. <https://doi.org/10.1145/3479241.3486684>

[C3] Ákos. Leiter Dániel Huszti, Nándor Galambosi, Edina Lami, Mohamad Saleh Salah, Péter Kulics, László Bokor: "Cloud-native IP-based mobility management: a MIPv6 Home Agent standalone microservice design," 2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Porto, Portugal, 2022, pp. 252-257, doi: 10.1109/CSNDSP54353.2022.9908059.

[C4] Ákos Leiter, Edina Lami, and László Bokor. 2022. Towards Cross-domain Mobility Management in the Edge: New Elements of Cloud-native Mobile IPv6 Full-scale Implementation. In *Proceedings of the 20th ACM International Symposium on Mobility Management and Wireless Access (MobiWac '22)*. Association for Computing Machinery, New York, NY, USA, 119–122. <https://doi.org/10.1145/3551660.3560919>

[C5] Á. Leiter, P. Böösy, M. Kis, and L. Bokor, 'Performance costs for IPv6-based mobility management on the top of Kubernetes', in 2023 IEEE 9th International Conference on Network Softwarization (NetSoft), 2023, pp. 217–221. doi: 10.1109/NetSoft57336.2023.10175456.

[C6] Á. Leiter, A. Hegyi, N. Galambosi, E. Lami, and P. Fazekas, 'Automatic failover of 5G container-based User Plane Function by ONAP closed-loop orchestration', in NOMS

2022-2022 IEEE/IFIP Network Operations and Management Symposium, 2022, pp. 1–2. doi: 10.1109/NOMS54207.2022.9789799.

[C7] Ákos Leiter, István Kispál, Attila Hegyi, Péter Fazekas, Nándor Galambosi, Péter Hegyi, Péter Kulics, József Bíró: ‘Intent-based 5G UPF configuration via Kubernetes Operators in the Edge’, in 2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN), 2022, pp. 186–189. doi: 10.1109/ICUFN55119.2022.9829576.

[C8] Á. Leiter, A. Hegyi, I. Kispál, P. Bödösy, N. Galambosi, and G. Z. Tar, ‘GitOps and Kubernetes Operator-based Network Function Configuration’, in NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium, 2023, pp. 1–5. doi: 10.1109/NOMS56928.2023.10154212

[C9] Á. Leiter, "Cloud Stock Exchange for Telecommunication," 2018 International Symposium on Networks, Computers and Communications (ISNCC), Rome, Italy, 2018, pp. 1-4, doi: 10.1109/ISNCC.2018.8531052.

[C10] Á. Leiter and L. Bokor, "Stock market trading strategies and cost estimation for telecommunication resource management," 2019 10th International Conference on Networks of the Future (NoF), Rome, Italy, 2019, pp. 126-129, doi: 10.1109/NoF47743.2019.9015018.

[C11] Á. Leiter and L. Bokor, "A Study on Use Cases and Business Aspects of Cloud Stock Exchange," 2020 International Conference on Information Networking (ICOIN), Barcelona, Spain, 2020, pp. 732-737, doi: 10.1109/ICOIN48656.2020.9016536.

## Hivatkozások

- [1] C. Perkins (Ed.), D. Johnson, and J. Arkko, ‘Mobility Support in IPv6’. in Internet Request for Comments, no. 6275. RFC Editor, Fremont, CA, USA, Jul. 2011. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc6275.txt>
- [2] S. Gundavelli (Ed.), K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, *Proxy Mobile IPv6*. in Internet Request for Comments, no. 5213. Fremont, CA, USA: RFC Editor, 2008. doi: 10.17487/RFC5213.
- [3] G. Tsirtsis, H. Soliman, N. Montavont, G. Giaretta, and K. Kuladinithi, *Flow Bindings in Mobile IPv6 and Network Mobility (NEMO) Basic Support*. in Request for Comments, no. 6089. IETF, 2011. [Online]. Available: <http://www.ietf.org/rfc/rfc6089.txt>
- [4] C. Bernardos (Ed.), ‘Proxy Mobile IPv6 Extensions to Support Flow Mobility’. in Internet Request for Comments, no. 7864. RFC Editor, Fremont, CA, USA, May 2016. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc7864.txt>
- [5] H. Zhang, W. Ma, W. Li, W. Zheng, X. Wen, and C. Jiang, ‘Signalling Cost Evaluation of Handover Management Schemes in LTE-Advanced Femtocell’, in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*, May 2011, pp. 1–5. doi: 10.1109/VETECS.2011.5956481.
- [6] ‘Open Network Automation Platform (ONAP)’. Accessed: Nov. 05, 2023. [Online]. Available: <https://www.onap.org/>
- [7] J. Varga, A. Hilt, C. Rotter, and G. Járó, ‘Providing Ultra-Reliable Low Latency Services for 5G with Unattended Datacenters’, in *2018 11th International Symposium on Communication Systems, Networks Digital Signal Processing (CSNDSP)*, 2018, pp. 1–4. doi: 10.1109/CSNDSP.2018.8471756.
- [8] ‘3GPP TS.23.314 - Architecture enhancements for control and user plane separation of EPC nodes’. Accessed: Jan. 22, 2024. [Online]. Available: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3077>
- [9] ‘Mongodb’. Accessed: Nov. 05, 2023. [Online]. Available: <https://www.mongodb.com/>
- [10] S. Avallone, S. Guadagno, D. Emma, A. Pescape, and G. Ventre, ‘D-ITG distributed Internet traffic generator’, in *First International Conference on the Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings.*, 2004, pp. 316–317. doi: 10.1109/QEST.2004.1348045.
- [11] *iPerf - The network bandwidth measurement tool*. [Online]. Available: <https://iperf.fr>